ARTI 4109 - Arquitecturas de Software

Cuaderno de Trabajo
Reto 1

Grupo 5
Camilo Alejandro Nossa Calderón
Brayan Felipe Rojas Bernal
Elmar Santofimio Suarez
Juan David Forero Rodríguez
Carlos Fidel Rodríguez Alarcón

Universidad de los Andes
Facultad de Ingeniería

Trabajo en Grupo

El propósito de este trabajo es identificar los requerimientos arquitecturalmente significativos del reto 1, producir un diseño para satisfacer dichos requerimientos, utilizando estilos y tácticas de arquitectura asociadas al desempeño y experimentar dichas decisiones.

Paso 1- Revisar las entradas: Tener claro el propósito del diseño a realizar, tener claros los requerimientos funcionales y tener claras las restricciones.

Utilice el formato de requerimientos de calidad para completar al menos 4 ASRs asociados al reto 1

ASRs

ASRs seleccionados de latencia y escalabilidad para la realización del experimento

+

Latencia	
Unidad:	Segundos
Respuesta esperada:	0.15 Segundos

Escalabilidad	
Unidad:	(# de eventos / minuto) * minutos
Respuesta esperada:	6500/minuto * 30 minutos

Unidad:	
Respuesta esperada:	

Actor:	**Usuario con suscripción premium**	**Estímulo:**	**Evento (ofertas de órdenes de compra y venta) ocurrido en un libro de órdenes de interés al usuario.**
Ambiente:	**Operación normal (1300 eventos/min)**	**Artefacto:**	**Sistema**
Respuesta esperada:	**El sistema genera la notificación y las envia a la cola de entrega en 0,15 segundos a los usuarios con suscripción premium.**		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Atributo de Calidad	
Unidad:	Segundos
Respuesta esperada:	<= 0.5

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Actor:	Propietario (vendedor)	**Estímulo:**	Registro de oferta de venta
Ambiente:	Operación normal (500 órdenes por minuto)	**Artefacto**	Receptor de ofertas de venta | ME
Respuesta esperada:	Registra en el libro de operaciones/ordenes Notifica a todos los interesados sobre la oferta disponible		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Atributo de Calidad	
Unidad:	Segundos
Respuesta esperada:	<= 0.3

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Actor:	Interesado/Comprador	**Estímulo:**	Registro oferta de compra
Ambiente:	Operación normal 800 órdenes de compra por minuto	**Artefacto**	Intermediario | ME
Respuesta esperada:	Registro de la oferta de compra | solicitud de compra en el libro de ordenes		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Atributo de Calidad	
Unidad:	Segundos
Respuesta esperada:	<= 0.2 (200 milisegundos)

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Actor:	Intermediario | ME	**Estímulo:**	Emparejamiento (tipo estocástico)
Ambiente:	Operación normal 1000 emparejamientos por minuto	**Artefacto**	ME
Respuesta esperada:	Matching exitoso Orden se empareja y confirma		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Atributo de Calidad	
Unidad:	Minuto
Respuesta esperada:	<= 1000 emparejamientos

Atributo de Calidad	
Unidad:	Minuto
Respuesta esperada:	800

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Actor:	Intermediario | ME	**Estímulo:**	Emparejamiento
Ambiente:	Operación normal 500 ordenes de venta por minuto 800 ordenes de compra por minuto	**Artefacto**	ME
Respuesta esperada:	Matching Se procesan y se confirman las ordenes		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

\+

Atributo de Calidad	
Unidad:	Tiempo en segundos
Respuesta esperada:	Tiempo de confirmación <= 500

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Atributo de Calidad	
Unidad:	
Respuesta esperada:	

Actor:	**Vendedor**	**Estímulo:**	**Envía una orden de venta**
Ambiente:	**Operación normal (500 órdenes de venta/minuto)**	**Artefacto**	**ME**
Respuesta esperada:	**Se registra**		

\-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Latencia	
Unidad:	Segundos
Respuesta esperada:	0.5 Segundos

Escalabilidad	
Unidad:	(# de Consultas / minuto) * minutos
Respuesta esperada:	10.000/minuto * 30 minutos

Unidad:	
Respuesta esperada:	

Actor:	**Usuario**	**Estímulo:**	**Consulta en un libro de órdenes**
Ambiente:	**Operación normal (1000 usuarios)**	**Artefacto:**	**Sistema**
Respuesta esperada:	**El usuario selecciona un libro de órdenes, realiza una consulta y el sistema retorna el resultado de la consulta.**		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Latencia	
Unidad:	Segundos
Respuesta esperada:	1.5 Segundos

Escalabilidad	
Unidad:	(# de Consultas / minuto) * minutos
Respuesta esperada:	6500/minuto * 30 minutos

Unidad:	
Respuesta esperada:	

Actor:	**Usuario con suscripción normal**	**Estímulo:**	**Evento (ofertas de órdenes de compra y venta) ocurrido en un libro de órdenes de interés al usuario.**
Ambiente:	**Operación normal (1300 eventos/min)**	**Artefacto:**	**Sistema**
Respuesta esperada:	**El sistema envía a sus usuarios con suscripción normal una notificación del evento ocurrido en tiempo con retardo.**		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

ASRs

+

Latencia	
Unidad:	Minutos
Respuesta esperada:	1 minuto

Escalabilidad	
Unidad:	(# de Suscripciones / minuto) * minutos
Respuesta esperada:	250/minuto * 120 minutos

Unidad:	
Respuesta esperada:	

Actor:	**Usuario (comprador)**	**Estímulo:**	**Compra de una suscripción (tipo estocástico)**
Ambiente:	**Operación normal (100 suscripciones/min)**	**Artefacto:**	**Sistema**
Respuesta esperada:	**Un usuario adquiere una suscripción y obtiene los beneficios de las notificaciones despúes de haber pagado con un tiempo de 1 minuto**		

-

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Impacto	

Atributo de calidad	
Prioridad	
Respuesta esperada:	

Paso 2- Establecer los motivadores de la iteración: Una ronda de diseño debe tener un objetivo alcanzable, permite decidir un criterio de terminación de la ronda y estable las actividades de diseño que comprenden la ronda

Motivador	Descripción
Emparejamiento (de ordenes de venta y compra)	Este proceso, busca en cada libro de órdenes y trata de encontrar la forma de satisfacer las diferentes órdenes de venta y de compra, que cumplan con las reglas establecidas. Este proceso se puede llevar a cabo, cada vez que una orden es registrada en un libro de órdenes, o de forma periódica, haciendo el matching sobre todos los libros de órdenes.
Servidor de notificaciones	Se busca generar la notificación sobre las nuevas ofertas que se den por parte de los usuarios vendedores/compradores, manteniendo a los usuarios premium informados en tiempo cercano al real sobre las transacciones realizadas en el sistema y a los usuarios estándar con un retraso en la recepción de la notificación.
Analítica (Consulta)	Se detalla la rapidez y capacidad del motor para traer la información de un libro, reflejando el comportamiento de los activos en las ordenes, dándole valor agregado a la subscripción informando al usuario de tendencias.

Paso 3- Escoger uno o mas elementos del sistema para refinar: Defina la granularidad de los elementos de arquitectura. Se siguen principios de descomposición Top-Down / Bottom-Up. Los elementos seleccionados deben buscar satisfacer un requerimiento específico

Elemento a Refinar	
Componentes Identificados	Descripción
Sistema/servicio de notificaciones	Se implementará un filtro para priorizar a los usuarios premium, esto se puede lograr a través de un cache que contenga los ID's de los usuarios premium.
Broker	Se añadirán colas en este servicio, para usuario premium, para que disminuya la latencia en la generación de notificaciones.

Paso 4- Seleccione conceptos que satisfagan los motivadores: Establecer alternativas de diseño. Seleccionar estilos, patrones y tácticas a ser utilizados

Estilo Arquitectura	Justificación
Arquitectura basada en eventos	1. El uso de colas nos permite priorizar eventos respecto al tipo de cliente, premium o estándar. 2. Al tener desacoplamiento entre componentes, permite que cada uno pueda escalar de manera independiente y responder de manera satisfactoria ante eventos estocásticos

Paso 4- Seleccione conceptos que satisfagan los motivadores: Establecer alternativas de diseño. Seleccionar estilos, patrones y tácticas a ser utilizados

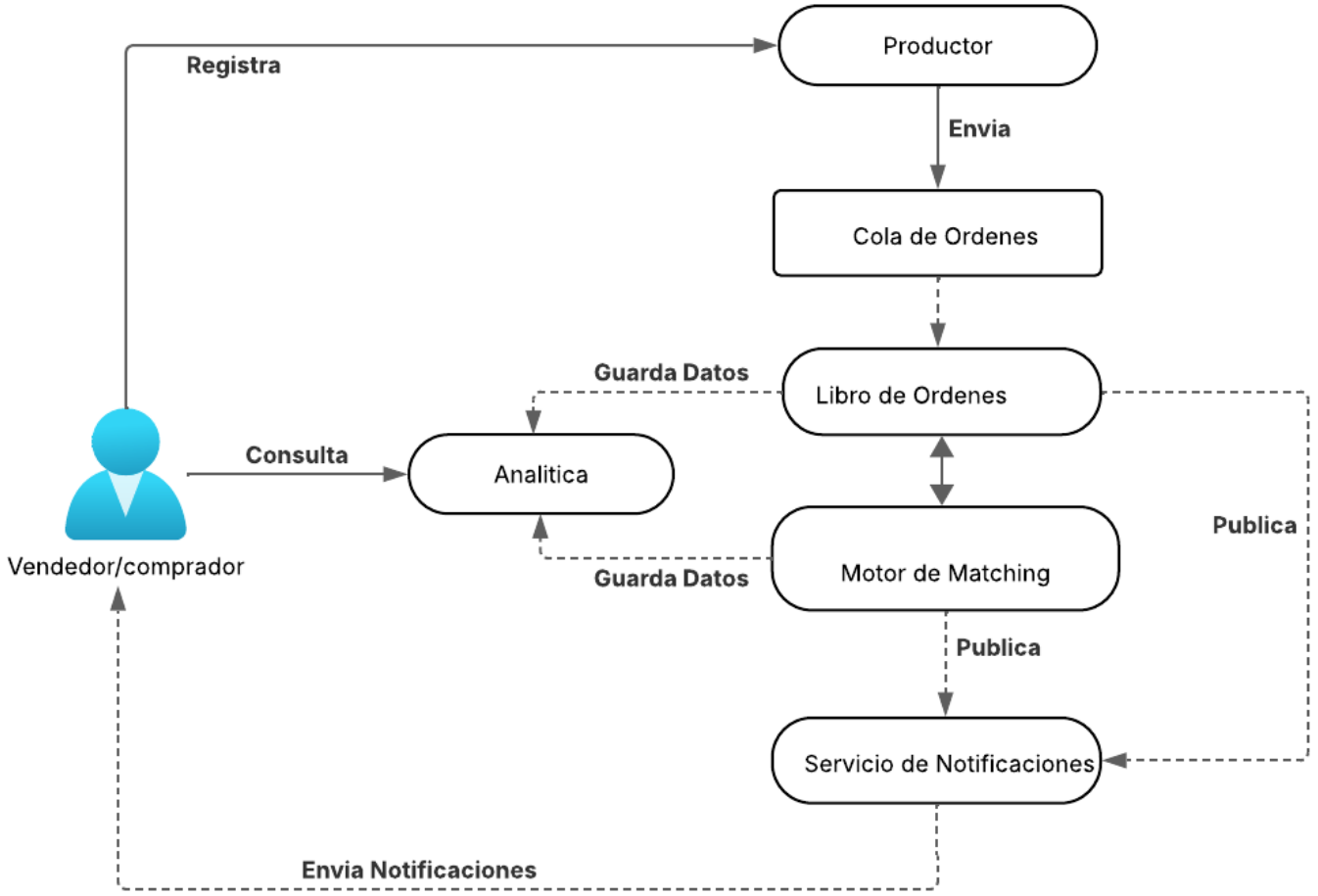
Tácticas de Arquitectura	Justificación
Mantener múltiples copias de computación	Esta táctica arquitectónica permite escalar automáticamente el sistema mediante la ejecución simultánea de múltiples copias del proceso de cómputo, sin necesidad de gestionar servidores ni infraestructura adicional, esto se logra por la capacidad de auto-escalado que nos provee el servicio Lambda de AWS.
Priorización de eventos	La priorización de eventos permite establecer un orden lógico y estratégico en el procesamiento de mensajes, garantizando que aquellos de mayor relevancia como las notificaciones dirigidas a usuarios premium se atiendan primero.
Reducción de muestreo	Esta táctica nos permite acceder a los datos de una manera más rápida, reduciendo la demanda de procesamiento y el almacenamiento. Esto lo logramos implementando un cache donde se consultarán los ID's de los clientes premium a los que se les generara la notificación.

Paso 5- Instanciar elementos de arquitectura, asigne responsabilidades y defina interfaces: Materializar los conceptos de diseño del paso 4 en elementos concretos de la arquitectura. Dejar claras las responsabilidades de dichos elementos. Definir las relaciones entre los componentes identificados.

Paso 6- Diseñe bosquejos iniciales de vistas y de conocimiento: Revisar la coherencia y completitud de las vistas de arquitectura. Este proceso ocurre varias veces de forma iterativa, la coherencia se va logrando con las iteraciones.

Utilizar la herramienta Lucidchart (usuario uniandes) para diseñar la arquitectura del sistema. Organice sus modelos en las vistas estudiadas en clase.

Vista de Contexto – Modelo de Contexto



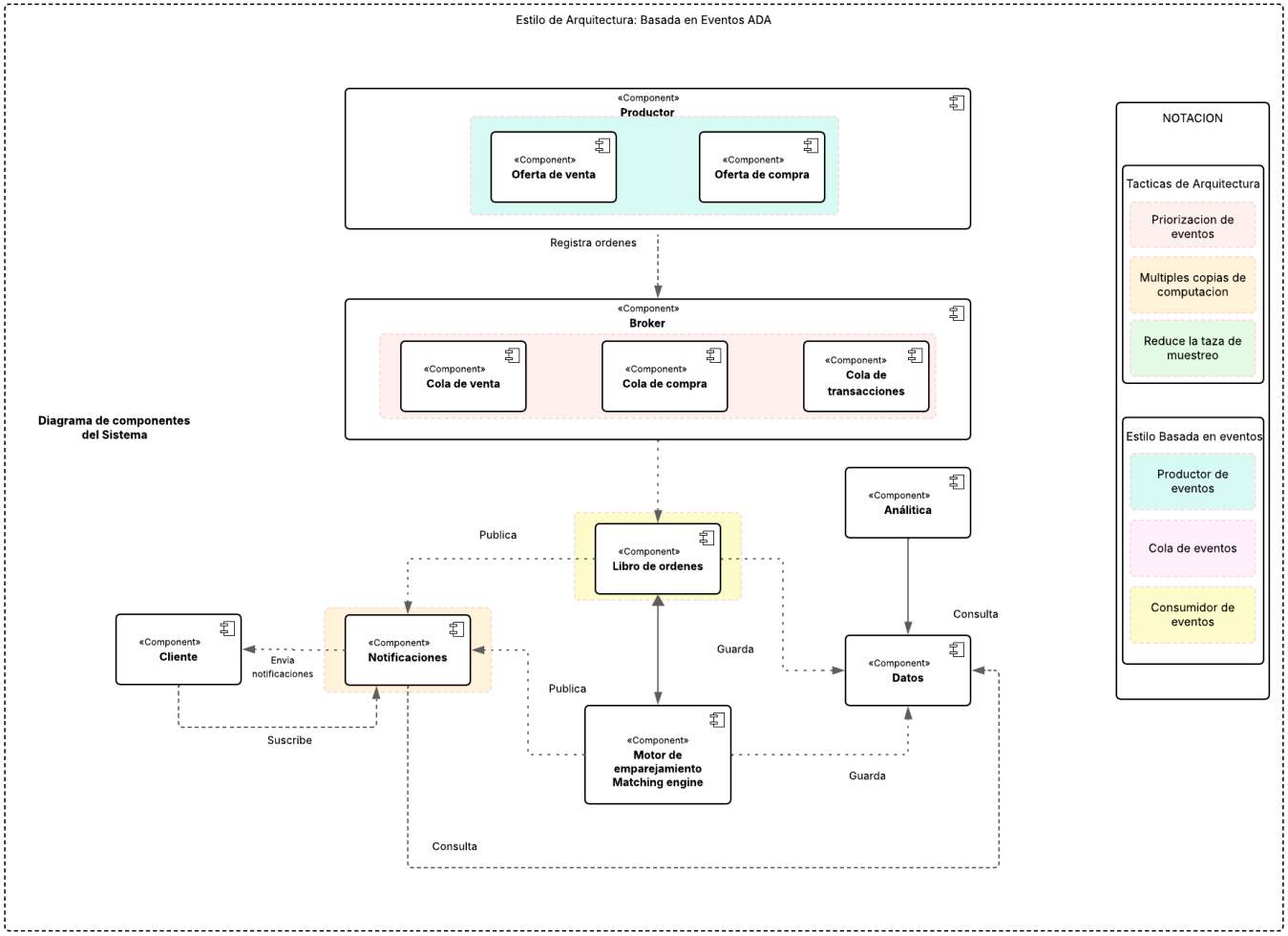
Hipotesis de experimento

Comprobar que una arquitectura basada en eventos con las siguientes tacticas:

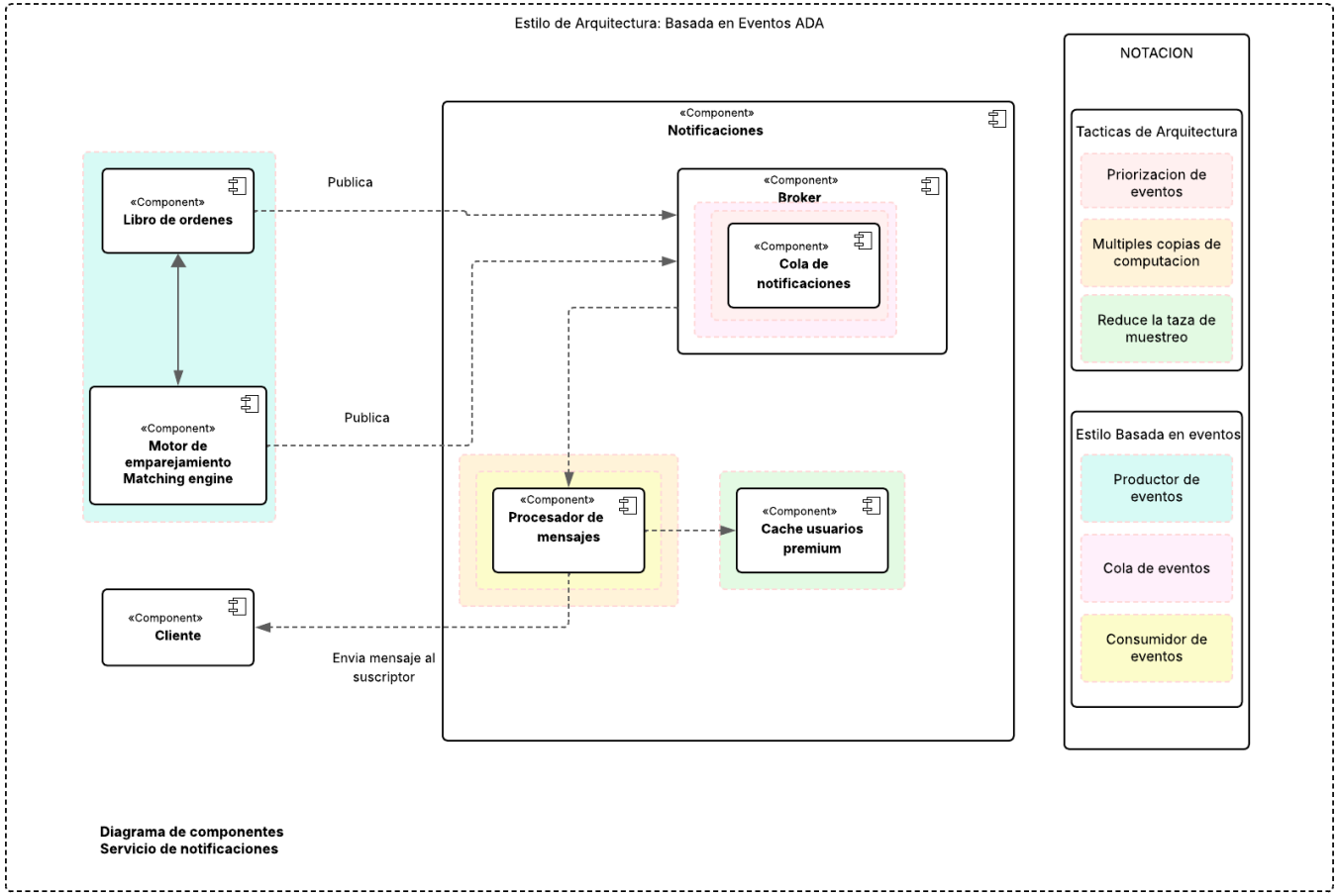
1. Multiples copias de computación
2. Priorización de eventos
3. Reducción en la tasa de muestreo

Logran que nuestros ASRs en el punto de sensibilidad cumplan con lo pactado.

Vista Funcional - Diagrama de componentes



Vista Funcional - Diagrama de componentes – Zoom de componente de Notificaciones



Vista de concurrencia - Diagrama de concurrencia

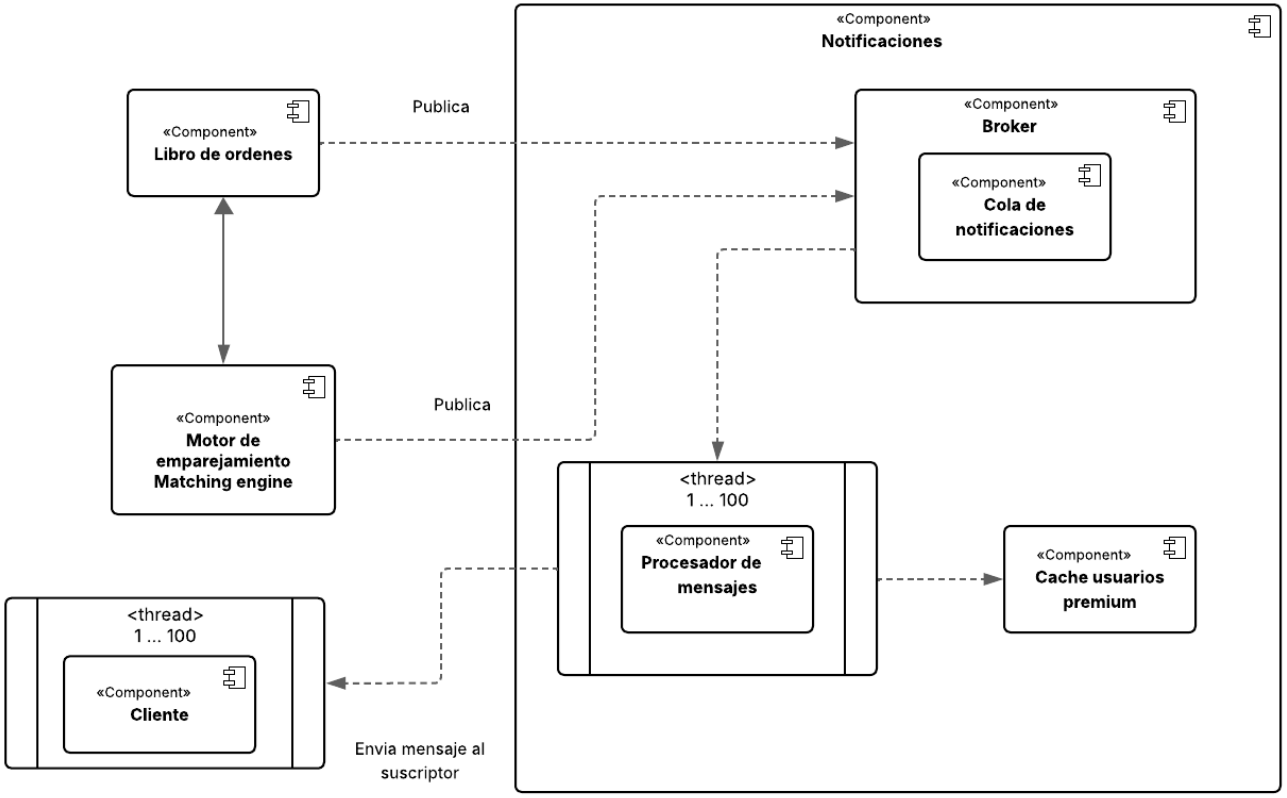
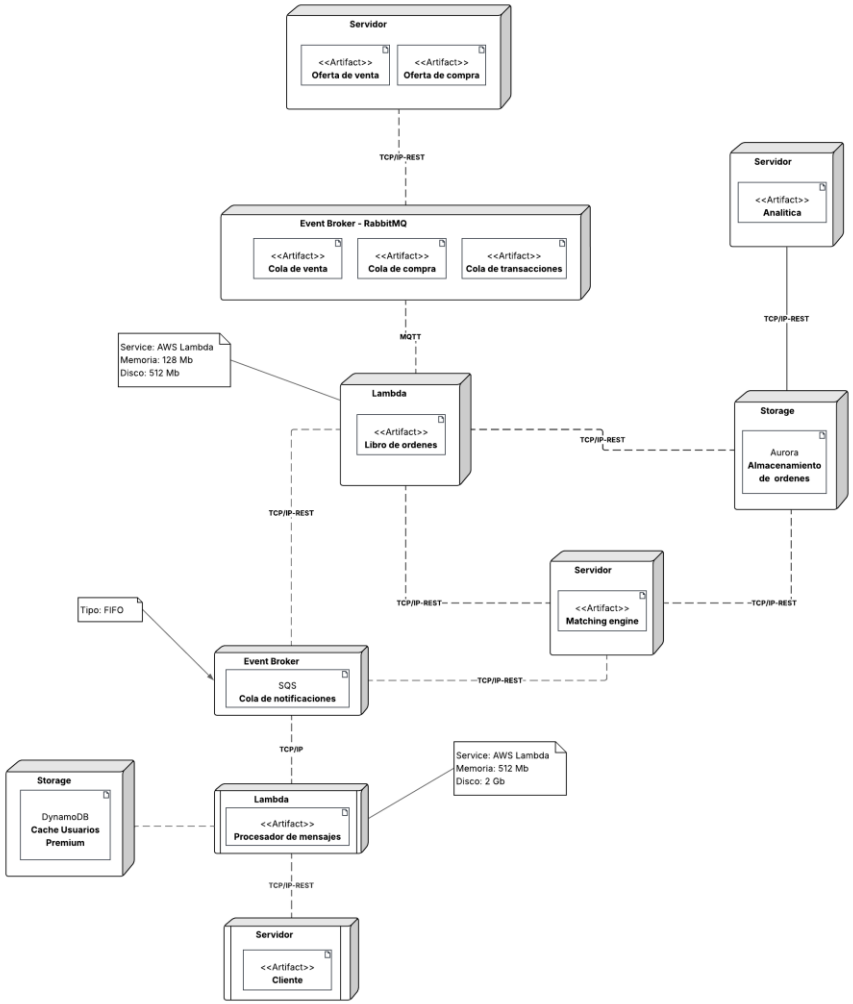


Diagrama de concurrencia
Servicio de notificaciones

Vista de Despliegue - Diagrama de despliegue



Costos de arquitectura (AWS)

Amazon SQS

Nivel gratuito: 1 millón de solicitudes al mes.
Precio estándar:

- USD 0.40 por cada 1 millón de solicitudes adicionales.
- Cada llamada a la API que envía, recibe o elimina mensajes cuenta como 1 solicitud, con hasta 10 mensajes y 256 KB totales.
- Cada fragmento de 64 KB se cobra como 1 solicitud (por ejemplo, 256 KB = 4solicitudes).

Transferencia de datos: sin cargos cuando los recursos están en la misma región.

Amazon Lambda

Nivel gratuito:

- 1 millón de invocaciones al mes.
- 400 000 GB-segundos de tiempo de cómputo al mes.

Precio por invocaciones: USD 0.20 por cada 1 millón de invocaciones adicionales.
Precio por tiempo de ejecución: USD 0.0000166667 por GB-segundo consumido.

- Se calcula como (memoria asignada en GB) × (duración en segundos) × (tarifa GB-s).

Costos de arquitectura (AWS)

Escenario	Mensajes/min	Duración	Total mensajes
Operación normal	1300	30 días	56'160.000
Pico de gestión	6500	30 min	195.000

*1300 event/min continuas
*6500 event/min x 30 min

*Memoria asignada: 512 MB
*Duración 0.15seg

Servicio	Concepto	Cálculo	Costo (USD)
Lambda	Invocaciones	(55.16 M /1M) * 0.20 USD	$11.03
	Cómputo	56.16M * 0.00000125 USD/INV	$70.20
SQS	Mensajes ON	(56'160.000 * 2) -1'000.000	$ 44.53
	Mensajes pico	(195.000 msj * 2) - 1'000.000	$ 0.16
Total			$125.25

Paso 7- Validar el cumplimiento de los motivadores y analizar el diseño producido: Validar con los stakeholders los objetivos cumplidos al final de una iteración o ronda de diseño. Revisión de pares. Decidir si el número de iteraciones y revisiones de diseño son suficientes o se debe refinar algún elemento

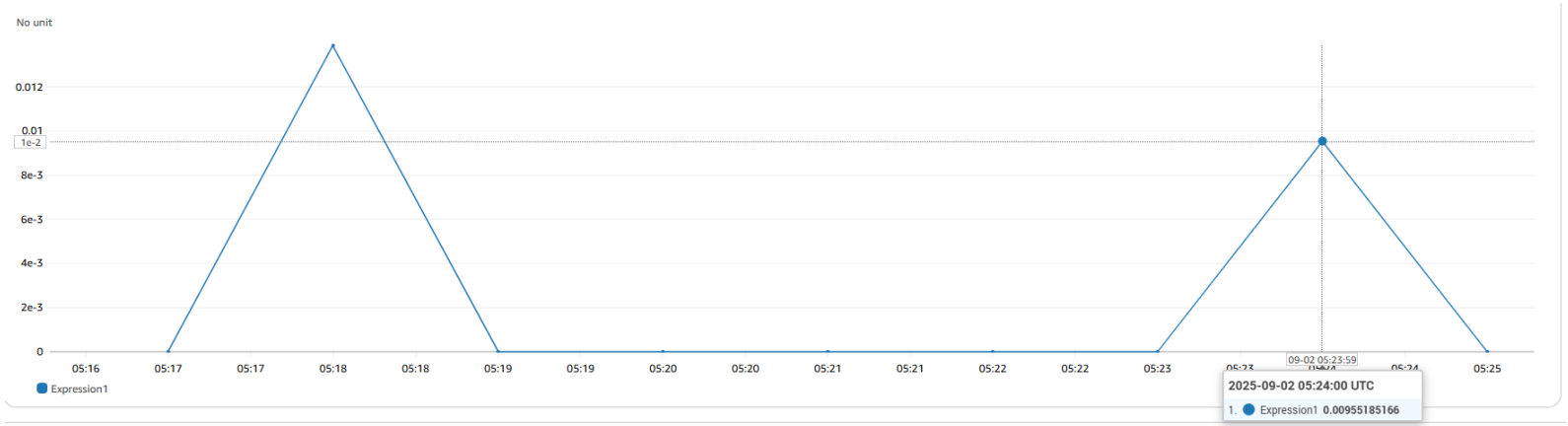
Título del experimento	Latencia en la creación y envío de notificación a la cola
Propósito	Medir el tiempo transcurrido desde que la función Lambda recibe un mensaje una orden hasta que publica en una cola de notificaciones.
Resultados esperados	Se espera que al tomar la información generarla y enviarla a la cola de notificaciones tenga un tiempo de duración de 0.15 segundos .
Recursos requeridos	Se requiere tener acceso a AWS y a la lambda creada con las colas correspondientes.
Elementos de arquitectura involucrados	El ASR asociado requiere que una notificación sea generada y enviada a la cola de notificaciones en un tiempo de 0.15 segundos, por lo que se buscara hacer uso de tácticas como múltiples copias de cómputo y priorización de eventos, ya que tenemos como punto de sensibilidad la lógica de creación de la notificación y llamada a la cola de destino.
Esfuerzo estimado	48 horas

Paso 7- Validar el cumplimiento de los motivadores y analizar el diseño producido: Validar con los stakeholders los objetivos cumplidos al final de una iteración o ronda de diseño. Revisión de pares. Decidir si el número de iteraciones y revisiones de diseño son suficientes o se debe refinar algún elemento

Título del experimento	Escalabilidad en la creación de mensajes a una alta demanda
Propósito	Evaluar la capacidad del sistema para generar y enviar simultáneamente 6500 notificaciones por minuto durante 30 minutos, midiendo cómo AWS Lambda escala en concurrencia bajo condiciones de carga elevada.
Resultados esperados	El sistema debe procesar al 1300 mensaje por minuto en una operación normal mientras que en una operación máxima, este debe tener la capacidad de generar 6500 mensajes por minuto en un tiempo de 30 minutos sin experimentar errores. Se esperan curvas de concurrencia que muestren un escalado lineal hasta el límite de configuraciones de Lambda
Recursos requeridos	Se requiere tener acceso a AWS y a la lambda creada con las colas correspondientes.
Elementos de arquitectura involucrados	El ASR asociado requiere generar 6500 notificaciones de maneras estocástica por minuto en 30 minutos , por lo que se buscara hacer uso de tácticas como múltiples copias de cómputo, haciendo uso de los beneficios que dan las lambdas al copiarse para poder procesar grandes cantidades de notificaciones.
Esfuerzo estimado	48 horas

Graficas del experimento

1300 event / min (Operacion Normal)

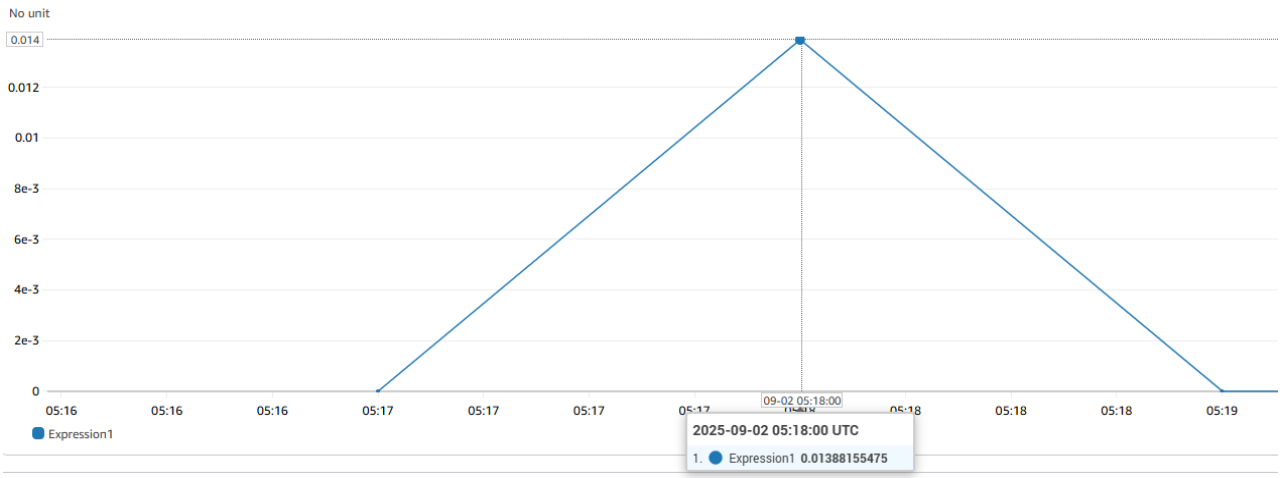


Latencia Esperada <= 0.15seg

Latencia Obtenida = 0.0095seg

Graficas del experimento

6500 event / min * 30 min (Pico)

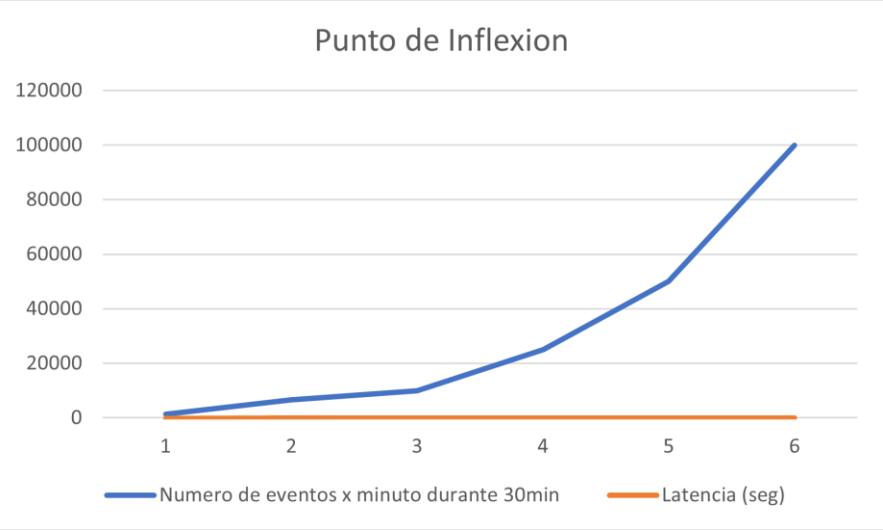


Latencia Esperada <= 0.15seg

Latencia Obtenida = 0.014 seg

Graficas del experimento

Punto de Inflexion



Numero de eventos x minuto durante 30min	Latencia (seg)
1300	0,06
6500	0,108
10000	0,1408
25000	16,43
50000	27,47
100000	42

El punto de inflexión es cuando tuvimos 25000 eventos * min / 30 min, ya que la latencia es superior al ASR propuesto

Paso 8- Iterar si es necesario: Repita los pasos 2 a 7 de ser necesario. Use el riesgo como una medida para decidir si parar o continuar con una iteración más.

Conclusión del experimento

Los resultados confirman la hipótesis de diseño: el uso de colas de prioridad, múltiples instancias de cómputo y cache de usuarios premium permite que el sistema cumpla con los atributos de calidad definidos (latencia y escalabilidad).

La arquitectura basada en eventos con AWS Lambda y SQS validó los ASRs de latencia y escalabilidad. Se alcanzaron latencias muy por debajo de 0.15 s (0.0095 s normal y 0.014 s en pico) y el sistema soportó 6500 notificaciones/min durante 30 min sin degradación.

El punto de inflexión encontrado es de 25000 eventos * min / 30 min debido a que la latencia es mayor a lo pactado en el ASR de latencia.

La decisión arquitectónica de emplear una arquitectura basada en eventos con AWS Lambda y Amazon SQS es adecuada para dar solución a los ASRs planteados.

NOTA: Se pueden ver en este repositorio https://github.com/juanforo31/Reto1_ASRs_Grupo5 los archivos, video del experimento, diagramas, código, entre otros.