



SIEEE

[Final DDS 20230729.pdf](#)

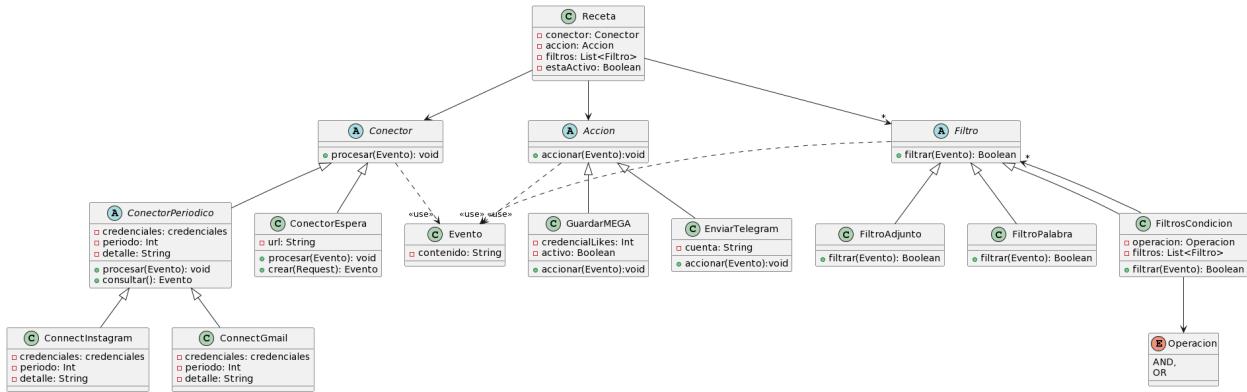
Clase de

consulta:<https://drive.google.com/drive/folders/1hGD8rBAx210FjYOW8dH0ps9PbIdCHy3V>

Hecho por: Chabela Lamas

Modelado de objetos

Diagrama de clases

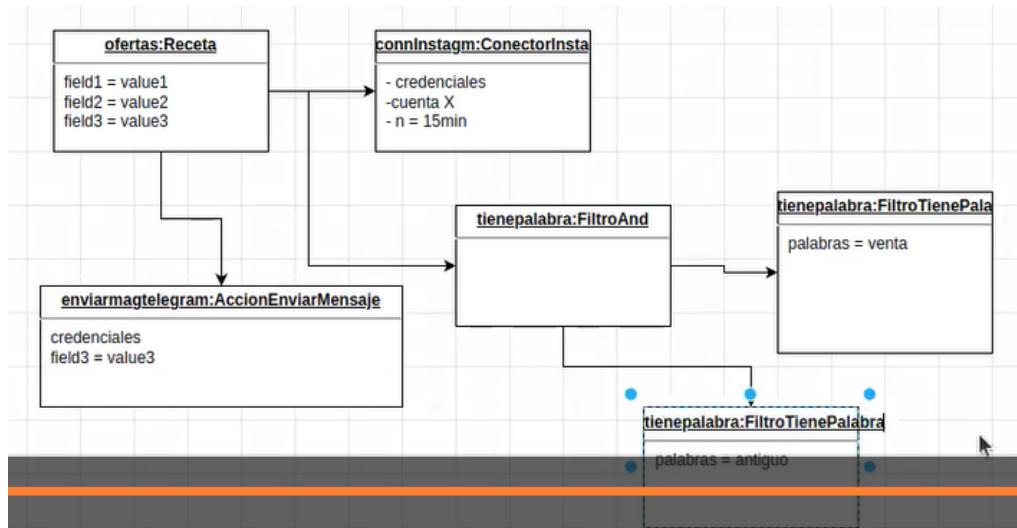


Decisiones de diseño

Se decide utilizar la estrategia de clase abstracta para los conectores, acciones y filtros, debido a que por un lado las interfaces no se pueden persistir al ser STATELESS y por el otro lado, cada una de sus clases hijas tienen accionar distinto.

De la misma manera se implementa un patrón Composite con el Filtro Condición permitiendo establecer las operaciones lógicas del AND y OR. Por consiguiente, se mejora la extensibilidad para crear nuevos compuestos y hojas que agreguen comportamiento y se simplifica la forma de tratar la estructura de objetos compuestos.

Mostrar sobre un diagrama de objetos o esquema conceptual, el ejemplo de “Ofertas” del contexto

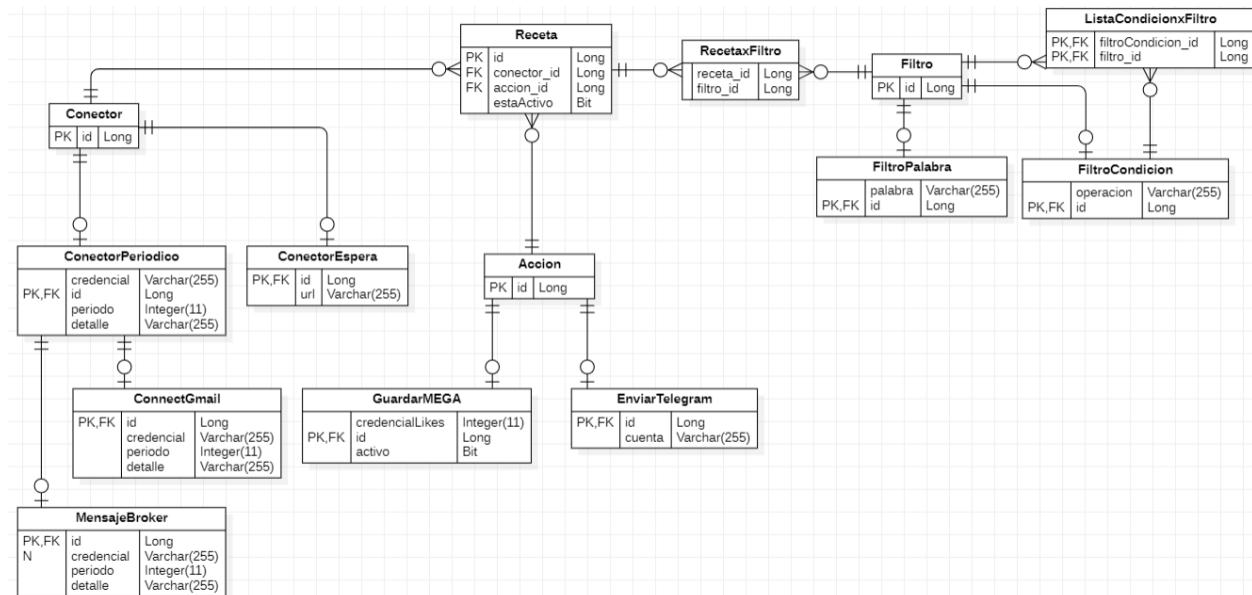


Explique con un diagrama de secuencia o pseudocódigo, cómo el escenario anterior procesa una publicación que tiene esas palabras “venta” y “antigüedad” en su contenido.

El conector de Instagram va al consultar del conector periódico. Se crea el evento. Del evento se procesa a la receta. De ahí se llaman a los filtros, si es que hay. Si cumplen los filtros, se ejecuta, luego, la acción.

Persistencia

Modelo de datos



Decisiones de diseño

En primer lugar, se decide no persistir el evento ya que la consigna menciona: “No hay ningún requerimiento que solicite datos sobre los eventos procesados por los usuarios, de hecho el acceso a los mismos post procesamiento del evento no debería permitirse nunca”.

En segundo lugar, se decide utilizar la estrategia de herencia: JOINED. faltaría justificar por qué

Entidades necesarias a persistir

- Conektor
- ConektorEspera
- ConektorPeriodico
- Accion
- Filtro
- FiltroCondicion

Impedance mismatch

Tipo de dato objetos	Tipo de dato relacional
LocalDateTime	SMALLDATETIME
String	VARCHAR(255)
Int	INTEGER(11)
String	TEXT

Respecto al impedance mismatch de Identidad, tuvimos que agregar un campo "id" (clave subrogada) para identificar únicamente a cada una de nuestras entidades.

Modelo desnormalizado

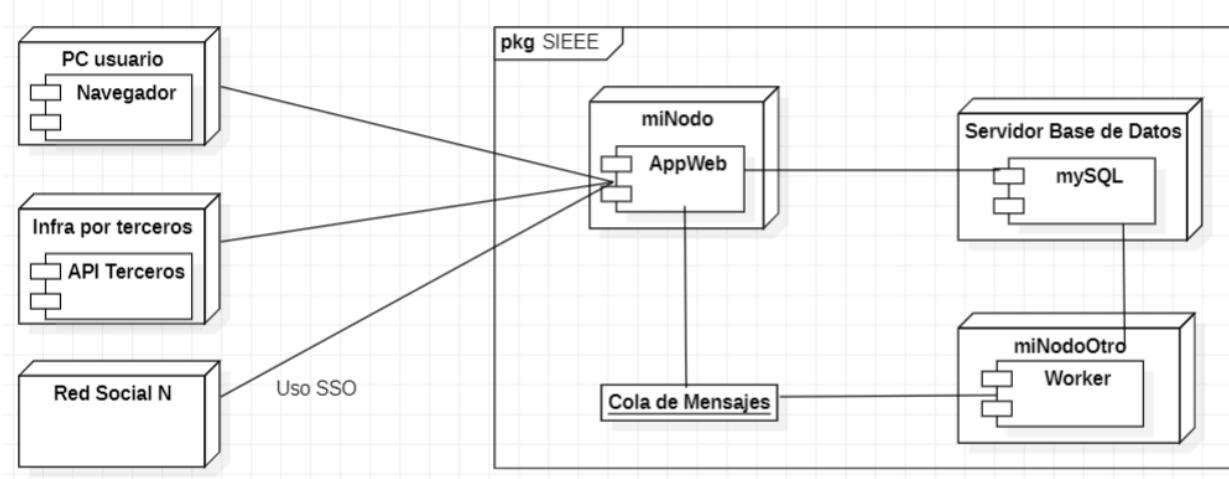
No se desnормaliza por performance ni por consistencia de datos

El sistema se integrará contra uno de facturación a través de la base de datos. Se quiere que ello tenga un impacto mínimo sobre la operatoria del sistema (es decir no degrade su rendimiento para los usuarios). Dicho sistema de facturación quiere conocer cuantas acciones/eventos se procesaron por usuario y cuanto se demoro en cumplirlas, para cobrar por el servicio

Persistís el usuario, sumas un contador, con el tiempo que tarda. Otra opción es desnormalizar la receta.

Arquitectura

1. Proponga una arquitectura física para el sistema, mediante un diagrama de despliegue o esquema general:
 - a. Deben figurar los componentes a utilizar, los sistemas externos, con qué protocolos todos se comunican, y cualquier aclaración que considere relevante.
 - b. Tenga en cuenta que pueden llegar muchos eventos a la vez y que no queremos saturar nuestros recursos tratandolos todos al mismo tiempo. No apuntamos a manejar información crítica, si se demora "mucho" en realizar una acción o pierde algún paquete, los usuarios están cordialmente advertidos
 - c. Se quiere que el usuario pueda comenzar a usar el sistema rápidamente, con lo cual se deben aceptar credenciales de redes sociales u otros sitios para registrarse/ingresar al sistema



Se comunican por TSPL por ser web. Los componentes son: base de datos, worker, mensajería y app web. La infraestructura de terceros es bidireccional. Mensajería recibe evento y después la aplicación lo va leyendo y procesando (worker).

2. Se observó que muchas de las recetas observan exactamente los mismos eventos, ya sea porque siguen las mismas cuentas en las redes sociales, u observan los mismos cambios en los recursos WEB, con lo cual los conectores tienen EXACTAMENTE los mismos datos.

Teniendo en cuenta esto:

- ¿qué se puede implementar para mejorar el rendimiento del sistema? Explique su funcionamiento
- Dé un ejemplo concreto

Se podría utilizar una Caché: memoria intermedia entre aplicación y servidor. Se pregunta antes de ir al servidor si se encuentra en este componente y si es así, te ahorras tiempo de búsqueda de la receta en el servidor.

