Decisiones basadas en la arquitectura

**Objetivo1: Creación de los micro-servicios**

Para dividir las entidades en dos microservicios coherentes, aplicaremos principio de **Dominio Limitado** (Bounded Context) de **Diseño Dirigido por el Dominio** (DDD). Esto implica identificar los límites naturales dentro del dominio de negocio y agrupar las entidades que comparten funcionalidad o datos.

En en nuestro caso, hay dos contextos principales:

1. **Gestión de Residuos** (Waste Management)
2. **Autorizaciones de Centros de Residuos** (Waste Center Authorizations)

Por lo que se decidió aplicar la siguiente división:

**Microservicio de Gestión de Residuos**:

* WasteManagerEntity: Contiene la información principal del gestor de residuos.
* WasteManagerAddressEntity: Relacionada con la dirección del gestor de residuos.

**Microservicio de Autorizaciones de Centros de Residuos**:

* WasteCenterAuthorizationEntity: Maneja las autorizaciones de los centros de residuos.

La entidad WasteManagerEntity ahora tiene una referencia a las autorizaciones a través de un identificador o clave foránea, en lugar de contener la lista completa de WasteCenterAuthorizationEntity. Esto ayuda a mantener los microservicios desacoplados y facilita la escalabilidad y el mantenimiento.

# // Microservicio de Gestión de Residuos

public class WasteManagerEntity {

private Long id;

private String nombre;

private String nif;

private WasteManagerAddressEntity wasteManagerAddressEntity;

// Referencia a las autorizaciones por ID

private List<Long> listOfWasteCenterAuthorizationIds;

private Boolean isEnabled = Boolean.TRUE;

private Long version = 0L;

private Date createdDate;

private Date lastModifiedDate;

}

public class WasteManagerAddressEntity {

private Long id;

private String direccion;

private Boolean isEnabled = Boolean.TRUE;

private Long version = 0L;

private Date createdDate;

private Date lastModifiedDate;

}

# // Microservicio de Autorizaciones de Centros de Residuos

public class WasteCenterAuthorizationEntity {

private Long id;

private String authorizationNumber;

}

Con esta división, cada microservicio tiene su propia base de datos y lógica de negocio, lo que permite que se desarrollen, desplieguen y escale de manera independiente. Además, la comunicación entre microservicios se puede realizar mediante llamadas a API REST.