Justificaciones de diseño - E3

- Elegimos, en lugar de usar una API, hacer una clase CalculadoraDistancia ya que encontramos la fórmula para calcular la distancia entre dos direcciones (con coordenadas) y nos pareció innecesario utilizar una API solamente para esa funcionalidad. Al mismo tiempo, le agregamos un método obtenerDireccionMasCercana que, dada una dirección original y una lista de direcciones, nos brinda la dirección más cercana de la lista a la dirección original; esto fue pensado sabiendo que debíamos calcular qué técnico se encuentra más cercano a una heladera.
- Sobre el tema de los colaboradores, pensamos en usar la misma tarjeta que para las personas en situación vulnerable. De esta forma nos queda solo el atributo de usosActuales de la tarjeta en null cuando el usuario es un colaborador. También usamos una herencia pequeña para el registro de los distintos usos, para evitar tener múltiples listas.
  Para resolver la comunicación a la heladera de que va a ser abierta por X persona, le agregamos al colaborador un método programarColaboración que crea una solicitudDeApertura que le envía a la heladera (esta la guarda en una lista de solicitudes pendientes), en paralelo a esto se genera un CronJob que al tiempo de expiración de la solicitud (inicialmente de 3 horas pero podría variar) si la persona no realizó la acción la elimina de la lista y queda inválida la solicitud de apertura. Entendemos que este dato viene de un archivo de configuración.
- Sobre las suscripciones, decidimos generar una interfaz Suscripción, que conoce tanto al Notificador como a los tres tipos de suscripciones estándar posibles (quedan N viandas, faltan N viandas y desperfecto en X heladera). Para las primeras dos, se generan a demanda según sean solicitadas (si alguien se suscribe a faltan 5 viandas y una persona anteriormente ya se había suscrito a esa misma cantidad de viandas faltantes se reutiliza la suscripción, en caso contrario se genera una nueva).
  El Notificador antes mencionado genera datos de tipo Notificación a través de la interfaz medioDeComunicacion, que dejó de ser un Enum para pasar a ser una clase con su propia lógica para notificar cierto medio.
- La estructura de los sensores cambió para poder usar un broker, ahora el receptorSensor se encarga de analizar los datos recibidos y accionar (a través del accionador) en caso de ser necesario.