## Favorecer la composición sobre la herencia.

Se debe favorecer la composición frente a la herencia, aun siendo los interfaces más pesados que la herencia en sí, ya que nos asegura un código menos propenso a errores al no tener clases que dependan de clases padre, esto hace que cuando se tenga que realizar algún cambio no influya a todas las clases hijas y dejen de funcionar.

Proporciona un código mantenible y poca deuda técnica.

Caso de uso:

Tenemos un centro comercial en el cual disponemos de parking, tiendas y restaurantes. Debemos organizar todo de manera que se puedan agregar tantas tiendas, restaurantes y parking como se quiera, y se deberá agregar estos desde el centro comercial ya que estos forman parte del mismo. En las relaciones hemos puesto que las interfaces tienen una agregación porque tiendas, restaurantes y parking tienen sentido sin estar dentro de un centro comercial, aunque en nuestro caso de uso no tenga sentido.

## UML

