ESCUELA COLOMBIANA DE INGENIERIA
PROGRAMACIÓN ORIENTADA A OBJETOS
PROYECTO INICIAL Ciclo No. 4 2020-2
REFACTORING Y EXTENSIÓN

El proyecto inicial tiene como propósito desarrollar una aplicación que permita simular una situación inspirada en el **Problem E** de la maratón de programación internacional 2019 **Dead End Detector.** En esta versión vamos a identificar las ciudades por colores.

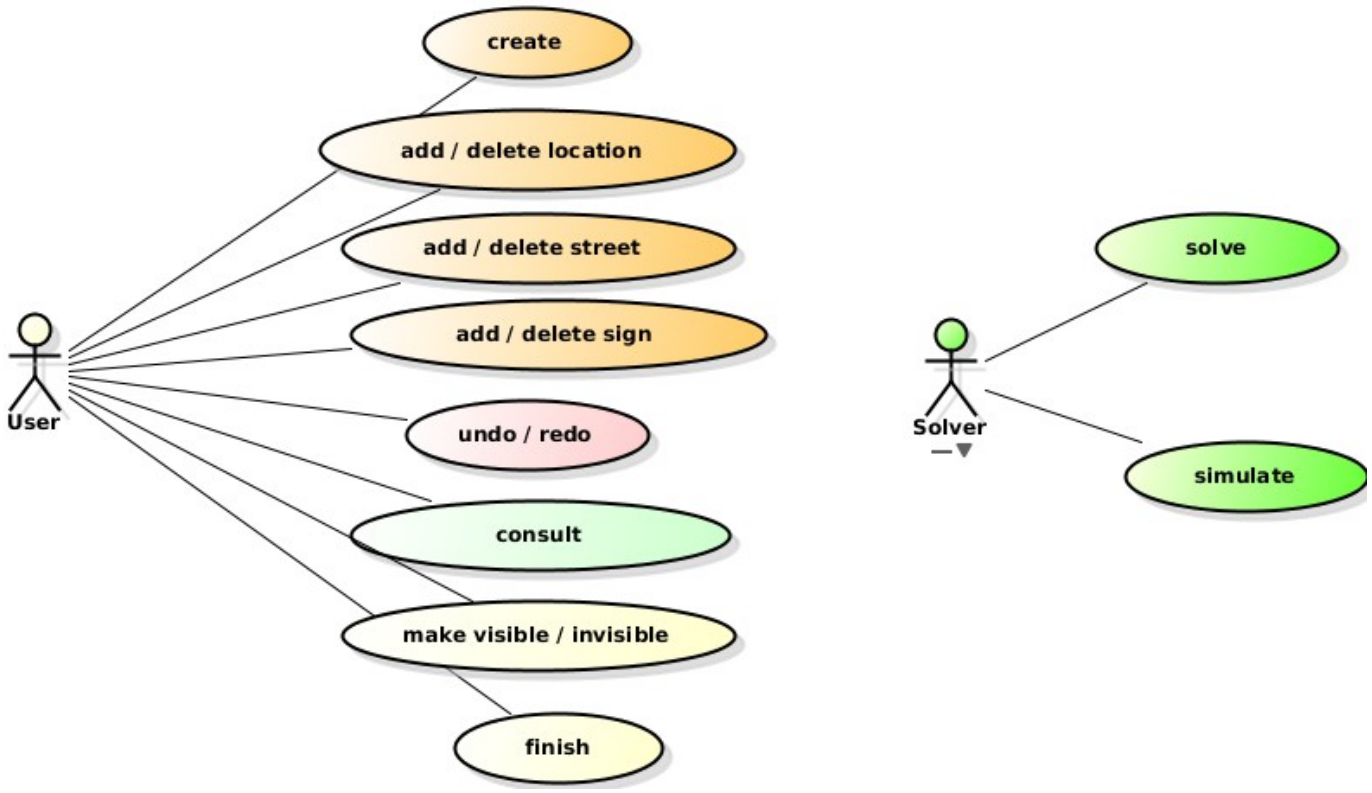
CUARTO CICLO

El objetivo de este ciclo es perfeccionar y extender el simulador para garantizar la calidad del mismo considerando los criterios de funcionalidad y extensibilidad.
Para esto se espera que en esta entrega se tenga:

1. Estructura del sistema en dos paquetes: `shapes` y `town`
2. Refactorización del paquete `shapes` aprovechando el mecanismo de herencia. El paquete debe figurar completo.
3. Refactorización y extensión del paquete `town` para incluir diferentes tipos de localidades, carreteras y señales.

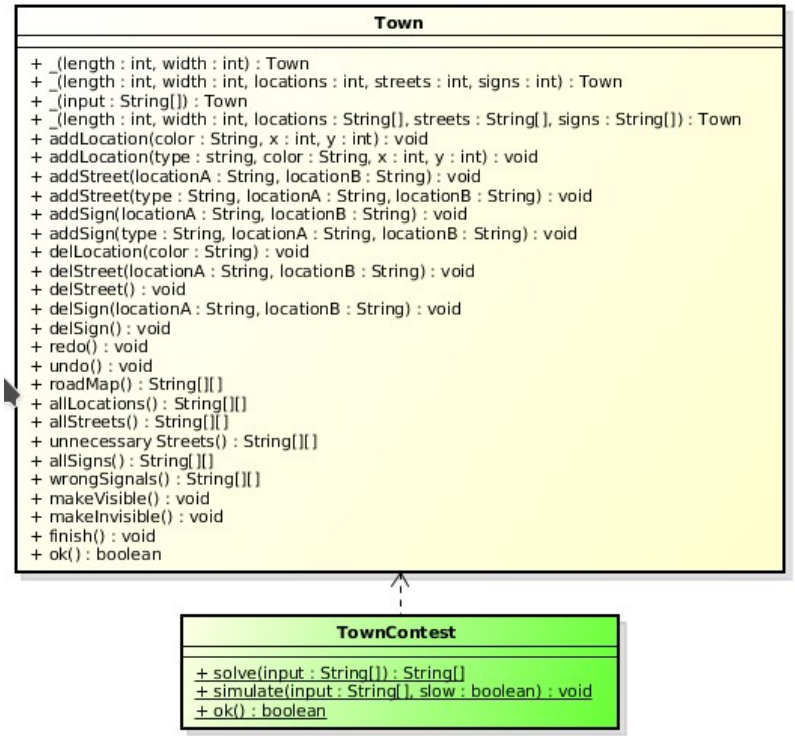
NUEVOS REQUISITOS FUNCIONALES

El simulador debe poder manejar diferentes tipos de localidades: **normal** (la que tenemos), **reverse** (intercambia las posiciones) y **isolated** (no permite que le coloquen carreteras); diferentes tipos calles: **normal** (la que tenemos), **silent** (no permite ser señalizada) y **prudent** (si le colocan una señal inmediatamente coloca la otra y una vez puestas no permite que retiren las señales) y diferentes tipos de señales: **normal** (la que tenemos), **bouncy** (si la van a retirar salta a la otra ciudad, si puede) y **fixed** (no se dejan retirar).
Defina un nuevo tipo de elemento: localidad, calle o señal.



Extensión: **create, add/ delete location, add/delete street, add/delete sign**

REQUISITOS DE DISEÑO Y CONSTRUCCIÓN



El input y townContest sólo usa los elementos normales

Los productos esperados para esta entrega son:

1. Diseño completo en la herramienta astah
   Diagrama de paquetes de los dos paquetes con sus relaciones..
   En astah, crear un diagrama de clases (cambiar el nombre por Package Diagram0)
   Diagrama de clases del paquete `town` con atributos y métodos privados y públicos.
   En astah, crear un diagrama de clases en el paquete (dejar el nombre por omisión).
   Diagrama de clases del paquete `shapes` con atributos y métodos públicos.
   En astah, crear un diagrama de clases en el paquete (dejar el nombre por omisión).
   Diagrama de secuencia de cada uno de los métodos las clases `town y townContest`
   (Parar en los componentes de **shapes**)
   En astah, crear los diagramas de secuencia en los métodos (dejar el nombre por omisión)
2. Código siguiendo los estándares de documentación de java.
   No olviden que el código de los métodos no debe ocupar más de una pantalla.
3. Código de pruebas de unidad que cubran los métodos desarrollados. Las pruebas se deben preparar en modo invisible. La clase `townC4test` será una creación colectiva usando el wiki correspondientes
   No olviden diseñar las pruebas considerando dos preguntas: ¿qué debería hacer? ¿qué no debería hacer?
   Los nombres de los casos de prueba deberán incluir la identificación de los autores. Por ejemplo, segunDAdeberia (DA: Iniciales de los primeros apellidos en orden alfabético).
4. Dos pruebas de aceptación para la sustentación escritas en BlueJ , con las esperas necesarias y con pregunta al usuario sobre si la acepta. `townC4Atest`
5. Documento de retrospectiva. (7 preguntas ver ciclo uno)
   Es necesario incluir la retrospectiva de este ciclo y de los anteriores.

REQUISITOS DE USABILIDAD

Los elementos de diferentes tipos debería poder distinguirse claramente.

REQUISITOS DE ENTREGA

Los productos los deben publicar en el espacio preparado en moodle en un archivo .zip con un nombre igual a la concatenación de los apellidos de los autores, ordenados alfabéticamente.
Publicar productos : Semana 10 Martes 13 de Octubre