Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Hidalgo

M3. Actividad

Por:

Diana Guadalupe García Aguirre | A01276380

Emmanuel Bolteada Manzo | A01276310

José Herón Samperio León | A01276217

Asignatura:

Modelación de sistemas multiagentes con gráficas computacionales

Quinto semestre

Ingeniería en Tecnologías Computacionales

Profesores:

Mtro. Alfredo Israel Ramírez Mejía

Dr. Joselito Medina Marín

Pachuca de Soto, Hidalgo; a 2 de diciembre del 2021

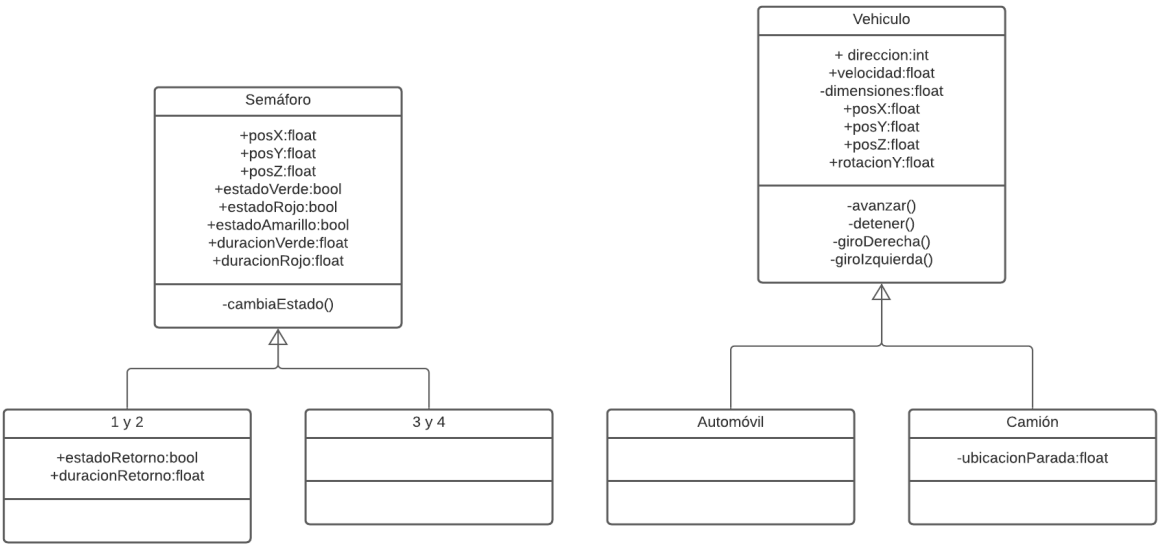
En el primer avance se comenzó con la implementación del servidor local utilizando Python, Flask y MESA para conectarlo con el modelo en Unity que ya estaba previamente completado.

En este segundo avance se realizó la corrección del funcionamiento que permite a los vehículos detectar la posición de los semáforos y detenerse en el momento apropiado. Se continúa trabajando en la generación de autos en las 4 direcciones de la circulación, pero, una vez que se logre realizar estos podrán detectar satisfactoriamente los semáforos y detenerse para dar paso a los otros vehículos.

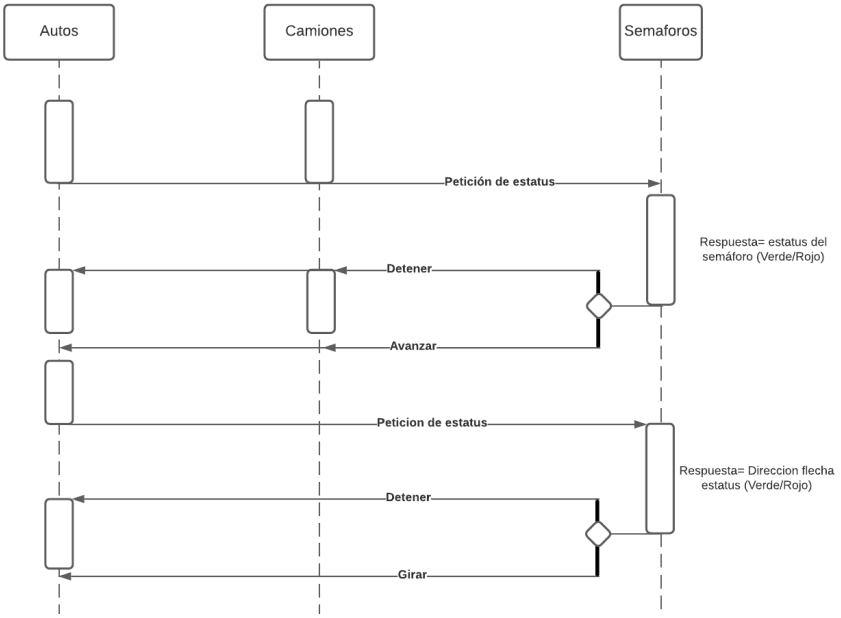
Así como en el avance anterior, se sigue considerando dejar fuera de las funcionalidades los giros y el cambio de carriles de los vehículos, pues se prioriza la sincronización de la circulación con los agentes avanzando en una sola dirección.

A continuación, se presenta la versión actualizada de los diagramas del reto tomando en consideración las modificaciones que se le han hecho por el poco tiempo disponible.

Diagrama de clases



Protocolo de interacción

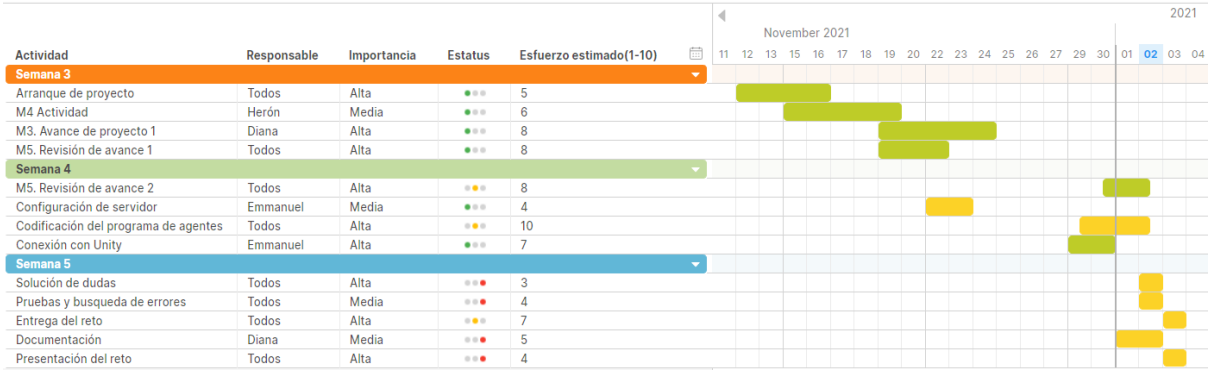


Código de implementación de los agentes

Por el aumento en la longitud del código en este avance, se optó por dejar el enlace al repositorio donde se encuentran los archivos RandomAgents.py, server.py y ControlTrafico.cs los cuales contienen el funcionamiento general de todo el reto.

El proyecto completo y código se encuentra en el siguiente repositorio:
https://github.com/DianaA96/agentes-computacionales

Plan de trabajo actualizado



Actividad	Responsable	Importancia	Estatus	Esfuerzo estimado(1-10)
Semana 3				
Arranque de proyecto	Todos	Alta		5
M4 Actividad	Herón	Media		6
M3. Avance de proyecto 1	Diana	Alta		8
M5. Revisión de avance 1	Todos	Alta		8
Semana 4				
M5. Revisión de avance 2	Todos	Alta		8
Configuración de servidor	Emmanuel	Media		4
Codificación del programa de agentes	Todos	Alta		10
Conexión con Unity	Emmanuel	Alta		7
Semana 5				
Solución de dudas	Todos	Alta		3
Pruebas y busqueda de errores	Todos	Media		4
Entrega del reto	Todos	Alta		7
Documentación	Diana	Media		5
Presentación del reto	Todos	Alta		4

Aprendizajes adquiridos

- Diana Guadalupe García Aguirre
    - La función que permite que los agentes ejecuten diversas tareas de manera coordinada, siguiendo el paso que dicta el modelo (y el ecosistema en el que persisten), es algo muy importante, que debemos entender para poder lograr que los agentes interactúen de manera adecuada. Lograr hacerlo con Mesa fue uno de los hitos de esta etapa, que nos permite avanzar con el resto del proyecto de manera significativa.
- José Herón Samperio León
    - Las características de POO en python y los métodos específicos que lo componen así.
    - La instancia de múltiples objetos y las peticiones que realizan a su entorno que definen su comportamiento y que la documentación es muy importante al igual que ejemplos de referencia que son muy limitados

- Emmanuel Bolteada Manzo
    - Para poder utilizar el servidor de Flask y manejar las peticiones desde Unity es necesario utilizar métodos de tipo Corutina, de los cuales aprendí su funcionamiento y apliqué para la sincronización de los semáforos después de x segundos.
    - La manera en que la librería Mesa instancia los agentes y maneja sus posiciones es algo compleja y ambigua, encontrando poca documentación al respecto, pero, una vez entendido el modelo bajo el que trabajan los “pasos” que da es más sencillo manipular los agentes generados y hacer su representación gráfica en Unity.