



Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Puebla

Modelación de sistemas multiagentes con gráficas
computacionales - TC2008B.1

Profesores:

Jose Eduardo Ferrer Cruz

Luciano García Bañuelos

Reporte final de solución al Reto

Daniel Flores Rodríguez A01734184

Jonathan Josafat Vázquez Suárez A01734225

Daniel Esteban Maldonado Espitia A01657967

Fecha:

3 de diciembre del 2021

● Movilidad Urbana

En los últimos años, el uso de automóviles ha ido en aumento en nuestro país, lo que ha traído consecuencias como el aumento de los accidentes automovilísticos, contaminación del aire y congestión vehicular. Este último punto, también genera consecuencias negativas sobre la economía del país, ya que es necesario que se puedan transportar tanto bienes y productos, como las propias personas a sus actividades laborales sin perder mucho tiempo en los trayectos.

La solución a este Reto consistirá en la simulación de la movilidad urbana, el transporte de los autos dentro de un ambiente natural, simulando en base a sistemas multiagentes las reacciones correspondientes e individuales que realizan cada automóvil, utilizamos modelos asíncronos de trabajo ante un setup y update, trabajando desde mesa (Python), utilizando servicios en línea como IBM cloud que te permite funcionalidades de API generando un MVC.

● Propuesta de solución

Nuestra propuesta consiste en simular una intersección de 4 carriles en cuatro direcciones diferentes, donde cada carril cuente con un total de cuatro autos que se dirigen siempre hacia la misma dirección, es decir, no dan giros o vueltas hacia otros carriles. La intersección también contará con un semáforo por cada carril para indicarle a cada auto el momento en el que pueden cruzar sin que se choquen con los demás automóviles. Cabe destacar que los autos disminuirán su velocidad cuando haya un automóvil o peatón justo en frente de ellos.

Así mismo, nuestra simulación contará con peatones los cuales estarán cruzando por las vías siempre y cuando los autos se lo permitan. Dichos peatones también caminan hacia la misma dirección en todo momento, y se deben detener cuando la calle que quieran cruzar tenga autos en movimiento o haya otro peatón justo en frente de ellos.

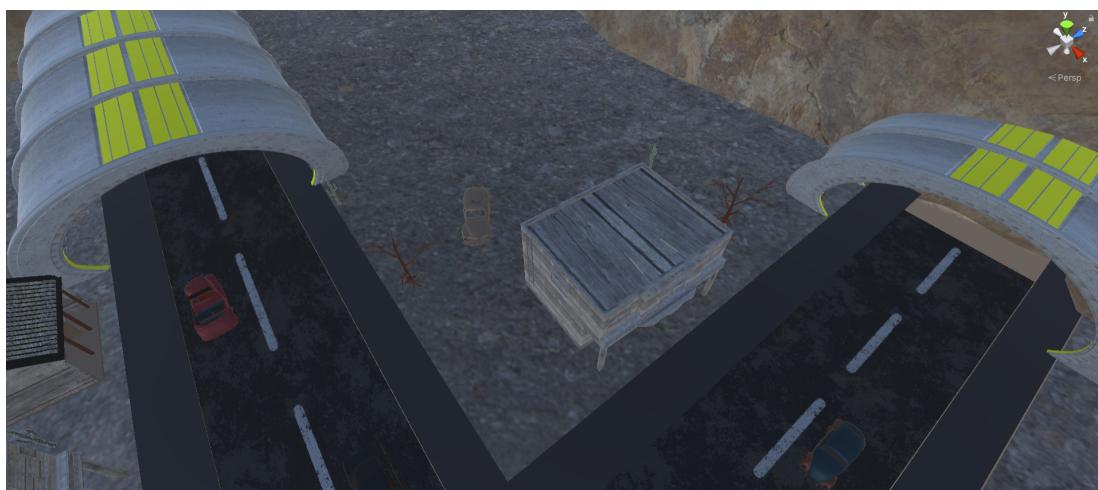
Por ello, los agentes que estarán involucrados en nuestro sistema son los automóviles, peatones y semáforos, y estos estarán siendo emulados haciendo uso del Framework Mesa y modelándolos en 3D usando Unity. Cabe destacar que, para que Unity obtenga los atributos de cada agente, este hará consultas a una API creada con Flask y publicada en un servicio de IBM Cloud la cual le proporcionará un JSON con dichos atributos como la posición y estado de cada agente en cada fotograma de la simulación.

● Elementos de la simulación

La simulación consta de la construcción de un cruce de 2 carreteras las cuales solo cuentan con un carril por cada dirección. Utilizando la herramienta ProBuilder, se crearon tanto los elementos como la colocación de las texturas.



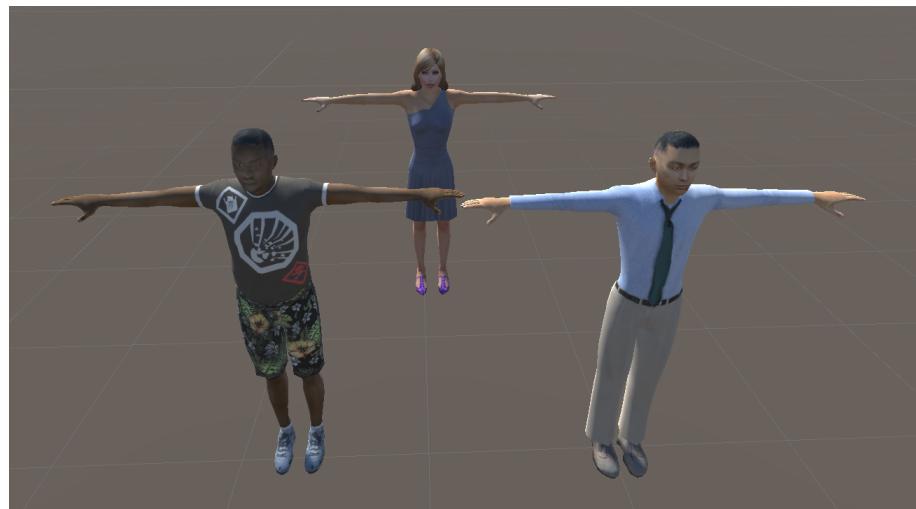
Se escogió una ambientación que simulara las afueras de una ciudad ya que estas zonas también suelen tener congestión vehicular y le da cierta originalidad a nuestro proyecto.



Estos son algunos ejemplos de los modelos de automóviles empleados en la simulación final.



Para nuestros peatones, usamos modelos prefabricados gratuitos de internet, los cuales además contaban con la animación de caminar, por lo que se vería más realista su movimiento dentro de la simulación.



Igualmente contamos con 4 semáforos y cruces peatonales en la intersección para que los agentes móviles interactúen adecuadamente entre ellos y no haya colisiones entre ellos.



- **Agentes involucrados**

Autos

Un agente de Auto tendrá el trabajo de trasladarse entre 2 puntos, siguiendo una ruta establecida por el constructor, basándose en densidad de agentes en ciertos puntos del mapa y el camino más corto junto a las alternativas.

Por medio del step, interactúa directamente con agentes del mismo tipo, evitando colisiones entre estos, además interactúa con otro tipo de agentes, como lo es el semáforo, el cual habilita o no el continuo movimiento siguiendo la ruta establecida. Esta interacción es por parte del agente Auto solo al momento de llegar a una intersección dentro del modelo.

Por último, los peatones están directamente relacionados con el movimiento del auto sin importar si ha llegado a una intersección o no. Esto le brinda la posibilidad a los vehículos de detenerse en caso de que un peatón no haya terminado de cruzar la vía

Semáforos

El semáforo es un agente el cual modifica su estado en cada step siguiendo la lógica correspondiente. Estos están establecidos dentro del modelo en cada una de las intersecciones, cuando un auto se aproxima a una intersección, el modelo le proporciona el estado del semáforo en esa ubicación.

Cada semáforo tiene una duración de 10 segundos entre cambio, pero como también contamos con luz amarilla que dura 3 segundos, cada luz verde dura como máximo 7 segundos, mientras que las luces rojas sí pueden durar 10 segundos por completo.

Peatones

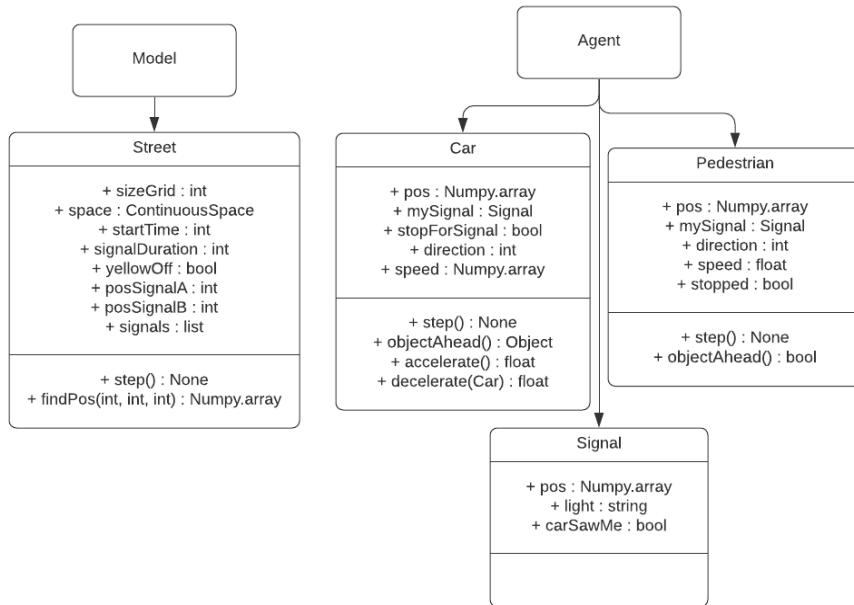
El peatón, al igual que el auto, tendrá una ruta asignada por su constructor que lo traslade entre 2 puntos, el viaje de este será a través de los bordes de la carretera, por lo que no afecta a los autos en su trayecto normal.

El step se trata de movimiento dentro del modelo, al momento de cruzar una carretera, este agente interactuó con los agentes tipo auto, para saber si alguno se aproximaba y dar la instrucción a este agente auto para detenerse.

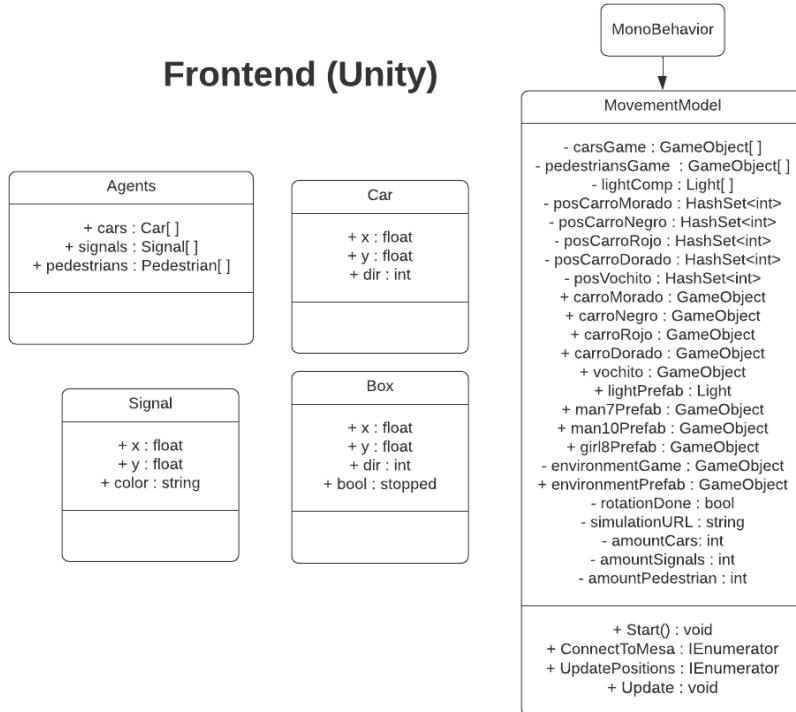
Los peatones también interactúan con los semáforos, ya que estos deben detenerse cuando los autos de la vía que quieren atravesar tienen la luz verde, y deben poder avanzar, si la vía que van a atravesar está en rojo.

- **Diagramas de clase**

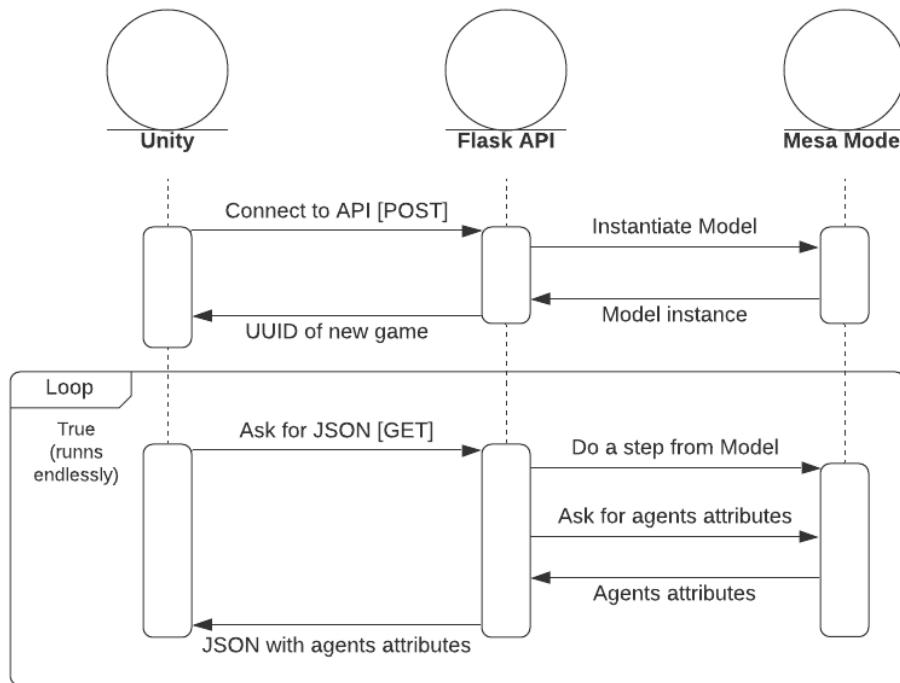
Backend (Mesa & Flask)



Frontend (Unity)



- **Diagrama de protocolos de interacción**



- **Particularidades**

Consideramos que la propuesta de solución que planteamos es original ya que el reto se nos presenta como una problemática con un enfoque más urbano y como una situación que se vive solamente en las ciudades. Sin embargo, en nuestro país, los pueblos o las afueras de las urbes también suelen sufrir este tipo de problemas principalmente porque cuentan con avenidas con menos carriles para cada dirección. Por esta razón, como equipo pensamos que sería mejor hacer la simulación en un entorno menos urbano pero que igual presenta este tipo de problemas de tráfico.

De igual manera, decidimos agregar a los peatones como agentes dentro de la simulación debido a que estos también suelen interactuar en las vías automovilísticas, sobre todo en los cruces que les corresponden. Por esta razón, agregamos cruces peatonales dentro de la intersección para que estos agentes tuvieran un espacio especial en el que pudieran pasar al otro lado de la calle.

- **Consideraciones éticas**

Para este caso podemos analizar que una de las consideraciones éticas que se deben tomar en cuenta al momento de trabajar con simulaciones computacionales de este tipo es aquellas consideraciones al momento de que aplicar estas en la vida real, es decir, tener en cuenta todas aquellas consecuencias e implicaciones que se tienen al momento de llevar este código a la vida real, en donde los automóviles autónomos son cada vez más comunes circulando en

carreteras y calles de ciudades, en donde de por medio, se ven implicadas personas y otra clase de elementos a las cuales su integridad física debería ser fundamental, pues al hablar de sistemas multiagentes, lo primero que debemos de tomar en cuenta es que al ser entes que se tienen la capacidad de tomar decisiones por sí mismos, cualquier error que suceda podría ocasionar una accidente que podría llegar a causar daños que tengan una repercusión económica millonaria, e incluso llegar a lesionar o incluso llegar a causar la muerte de personas que se vean involucradas en dicho incidente.

Para esto, algunas de las consideraciones importantes que nosotros como ingenieros debemos tener en cuenta al momento de tratar de implementar esta clase de código en autos autónomos que simulan ser nuestros agentes que se mencionaron anteriormente, una de estas y la más importante es la elaboración de un plan de pruebas.

Con este plan de pruebas, nosotros como ingenieros en sistemas, tendríamos que programar una solución que sea capaz de pasar todas las pruebas propuestas por expertos en el tema de vialidad, seguridad peatonal y civil, con esto se disminuye exponencialmente las probabilidades de que suceda algún accidente.

Sin embargo, como lo hemos podido comprobar en varias ocasiones, aunque las probabilidades sean bajas, nunca se está exento de que llegue a suceder algún accidente, y por lo tanto, lo más conveniente para todos sería definir, con ayuda de abogados expertos en el tema, una declaración del tema que tenga como objetivo definir aquellas obligaciones que cada uno de los sujetos que se vean involucrados deben de tener y conocer cuando llegue a suceder algún accidente, sin embargo, es en este punto cuando entramos en un cuestionamiento de un tema que ha sido muy controversial desde la entrada de automóviles autónomos en el mercado. ¿Quienes serían los responsables de las consecuencias al momento en que un accidente se produzca, aquel que es dueño del vehículo e iba conduciendo en ese momento, o el programador que realizó el código que, básicamente le da vida al automóvil autónomo? Es un tema que se ha reflexionado por varios años, pero que sin embargo, desde nuestra perspectiva, debería ser un tema que se trate desde otro punto de vista, es decir, viendo cómo es que estas nuevas tecnologías deberían de llegar a ser consideradas como una nueva extensión en términos éticos y jurídicos, en donde no solo debería de ser culpado aquel que implementa dicha tecnología, sino que también a aquel sujeto jurídico que se directamente se haya visto involucrado al momento de que pasa un accidente y este es el presunto culpable de está.

● Reflexiones individuales

Daniel Flores:

Al haber cursado este curso, pude reforzar los conocimientos que adquirí a lo largo de las clases y durante las actividades realizadas a lo largo del curso de sistemas multiagentes y modelación de gráficos tridimensionales.

El simple hecho de haber planeado desde cero una simulación de sistemas multiagentes me ha dejado una experiencia que sin duda puedo aplicar en mi especialización más adelante en

mi carrera profesional. Pues gracias a esto, pude aprender sobre el uso de algunas de las librerías más importantes de Python para implementar Inteligencia Artificial y trabajar con Big Data, como lo fue el framework Mesa y la librería Pandas. Las cuales nos ayudaron a definir y dar vida a nuestros agentes más importantes, es decir, los automóviles.

Aunado a esto, pude enriquecer mi experiencia, haciendo uso de una de las herramientas tecnológicas más importantes que existen en el mercado actualmente: IBM cloud, con la cual pudimos literalmente, hacer que nuestra simulación funcionará desde cualquier lugar con acceso a internet, por el uso de tecnologías en la nube, en donde un servidor se encuentra ejecutando nuestro código que backend implementado para esta solución.

Finalmente, considero que también pude aprender bastante sobre el uso de unity para poder realizar simulaciones computacionales en 3D, haciendo uso de herramientas útiles como lo es ProBuilder, el uso de prefabs, el uso de packages y la programación de recursos desde un código de C Sharp.

Daniel Maldonado:

A través de este Reto, tuve la oportunidad de reforzar los conocimientos adquiridos en la materia sobre los sistemas multiagentes y la modelación de gráficos tridimensionales. El haber planeado e implementado desde 0 un sistema que involucra la interacción de diferentes agentes con diferentes características, me pareció algo muy enriquecedor para mi formación profesional, ya que creo que dichos sistemas se pueden percibir en muchos ámbitos del mundo digital.

Por otro lado, me resultó muy llamativo el haber optado por emplear una tecnología en la nube como lo es IBM cloud, la cual a pesar de que en un principio me parecía algo compleja de usar, después de probarla un par de veces me resultó mucho más sencilla de lo que pensé y muy útil para tener cualquier servicio web y poder consultarla en cualquier momento.

También, considero que el haber escogido a Mesa como el framework de multi agentes fue una decisión acertada por parte de los profesores ya que me permitió aprender POO en Python y, después de mucha práctica, resultó muy sencillo entender cómo manejar los diferentes agentes y hacer que interactúen entre ellos.

Jonathan Josafat:

<texto>

Video de presentación final:

<https://youtu.be/kIqrypmB-GY>