Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Puebla

Modelación de sistemas multiagentes y gráficas computacionales

TC2008B.1

Reporte Final

Alumnos:

José Antonio Bobadilla García	A01734433
David Zárate López	A01329785
Karen Rugerio Armenta	A01733228

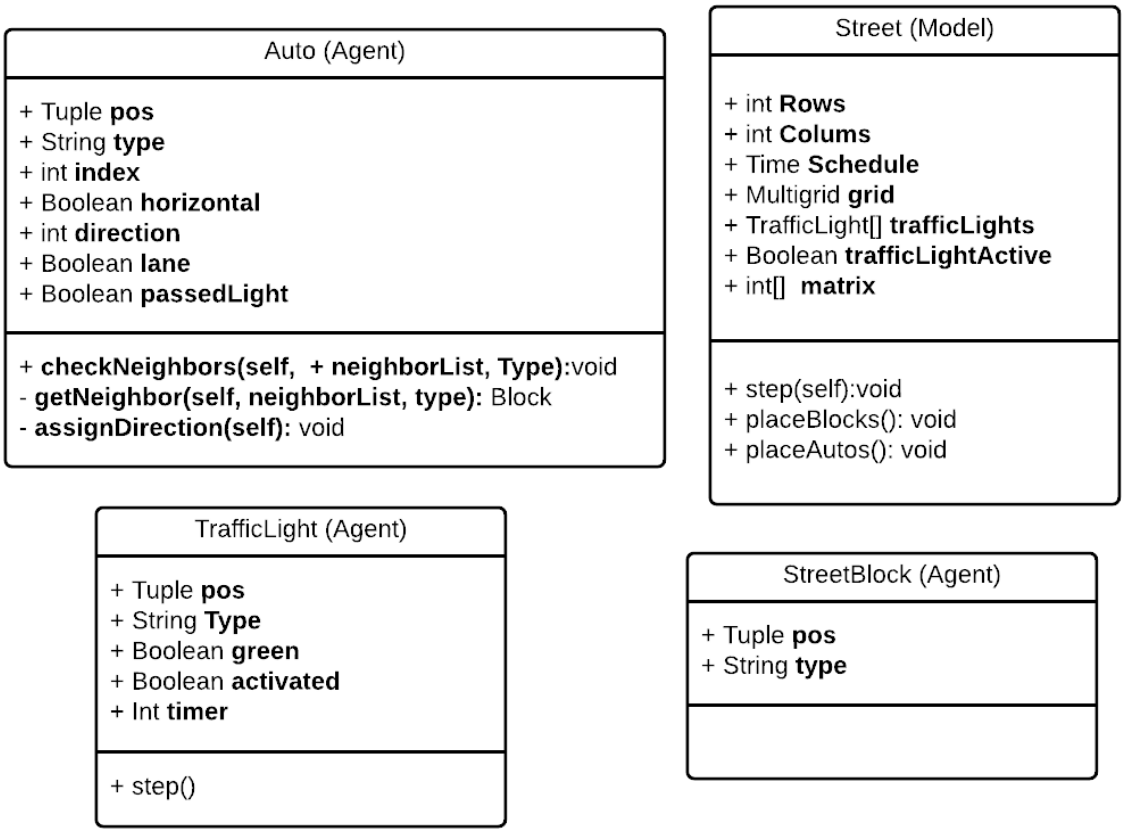
Fecha:
26 de Noviembre del 2021

I. Descripción del reto

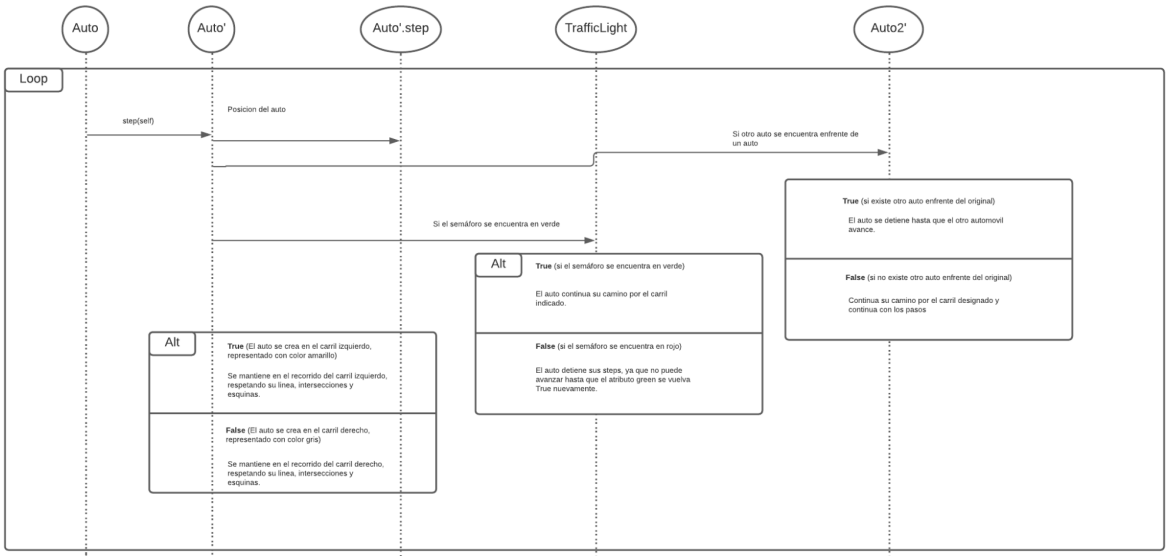
La movilidad de los diferentes automóviles en un país es fundamental para el desarrollo económico y social, junto con la mejora de la calidad de vida de sus habitantes. Al haber demasiados automóviles en una ciudad, o en un país, se vuelve un problema ya que esto afecta a las áreas que mencionamos anteriormente. Si las personas no pueden llegar a sus trabajos, puede haber pérdida de estos, si la materia prima no llega a las industrias, hay pérdida económica, si las medicinas no pueden llegar a ciertas zonas debido al gran cantidad de tráfico que existe en ciertas ciudades, afecta a la calidad de vida de las personas.

Es de gran importancia resolver este problema de movilidad de tal manera que en el caso del reto, México pueda estar entre las economías más grandes del mundo. El reto consiste en darle solución a un problema de movilidad urbana en nuestro país, mediante un enfoque el cual reduce la congestión vehicular al simular de una manera gráfica el tráfico, representando la salida de un sistema de multi agentes.

II. Diagramas de clase



III. Protocolos de interacción



IV. Implementación de los agentes

Para comenzar la implementación de una solución para el reto de movilidad urbana, se definieron cuántos y cuáles iban a ser los agentes involucrados en este. Primeramente se creó un agente *Auto*, el cual simulará los automóviles que se moverán en las calles de la ciudad. Posteriormente tendríamos a otro agente el cual simulará cada bloque transitable de calle, el cual llamamos *StreetBlock*. Por último tenemos al agente que simulará cada uno de los semáforos, los cuales llamamos *trafficLight* y realizarán la tarea de indicarles a los agentes Auto cuándo detenerse y cuándo avanzar. Esta implementación se desarrolló en *Python*, usando la librería de *mesa*, que es un framework para el modelado de sistemas multi agentes.

Cada uno de los agentes cuenta con atributos y métodos propios los cuales le permiten tomar decisiones e interactuar dependiendo de su entorno, por ejemplo, según su ubicación el agente auto comienza a moverse en la dirección de su carril, así mismo, gracias a sus métodos se detiene si hay otros agentes de tipo auto en frente de él para evitar colisiones, y si detecta un agente de tipo semáforo, interactúa con él para poder tomar la decisión de detenerse si se encuentra en rojo, o avanzar si su estado es verde, etc.

En cada step que el modelo genera, el agente Auto verifica si este no se sale de la cuadrícula general y posteriormente si el bloque que tiene enfrente es un bloque normal, un semáforo u otro agente del mismo tipo (auto). Si este agente se encuentra con un semáforo, a su vez, este verifica su estado actual y decide si avanzar o no, si este no está activado, verifica en cuál carril se encuentra, esto quiere decir si está en horizontal o vertical. Posteriormente se usan varios condicionales para decidir en qué momento debe de dar vuelta y mantenerse en su carril asignado.

V. Implementación de gráficas computacionales

Para realizar el ambiente en la cual se desarrolla la simulación, fue decidido crear una ciudad, en la que se utilizaron prefabs obtenidos de la Unity Store y que se encuentran a escala de las medidas de la matriz del modelo desarrollado. Para mantener aspectos de la vida real en la simulación, se colocaron diferentes edificios y adornos dentro de la ciudad.

Una vez obtenido el modelo multiagente con Python, se procedió a conectarse mediante el uso de Flask, framework de desarrollo web, un backend que provee las diferentes posiciones de los automóviles y de los semáforos que se van actualizando a medida que se ejecutan movimientos en el modelo. Estos, son mandados como un JSON a Unity y se utilizan en un archivo de C# para generar los agentes en 3D.

Para ello, inicialmente se crean 10 automóviles (preestablecidos) en el origen de la ciudad y una vez creada la conexión con el backend, se actualizan sus posiciones para empezar cada movimiento que realizan. Así, es posible visualizar de la mejor manera, los diferentes agentes que toman lugar en el sistema y la interacción que tienen entre ellos. Esto, se realiza en la función de Update en el código de C# mencionado anteriormente. De igual manera, el script permite que cuando el automóvil cambie de dirección, este sea rotado para simular las vueltas que se realizan en la ciudad.

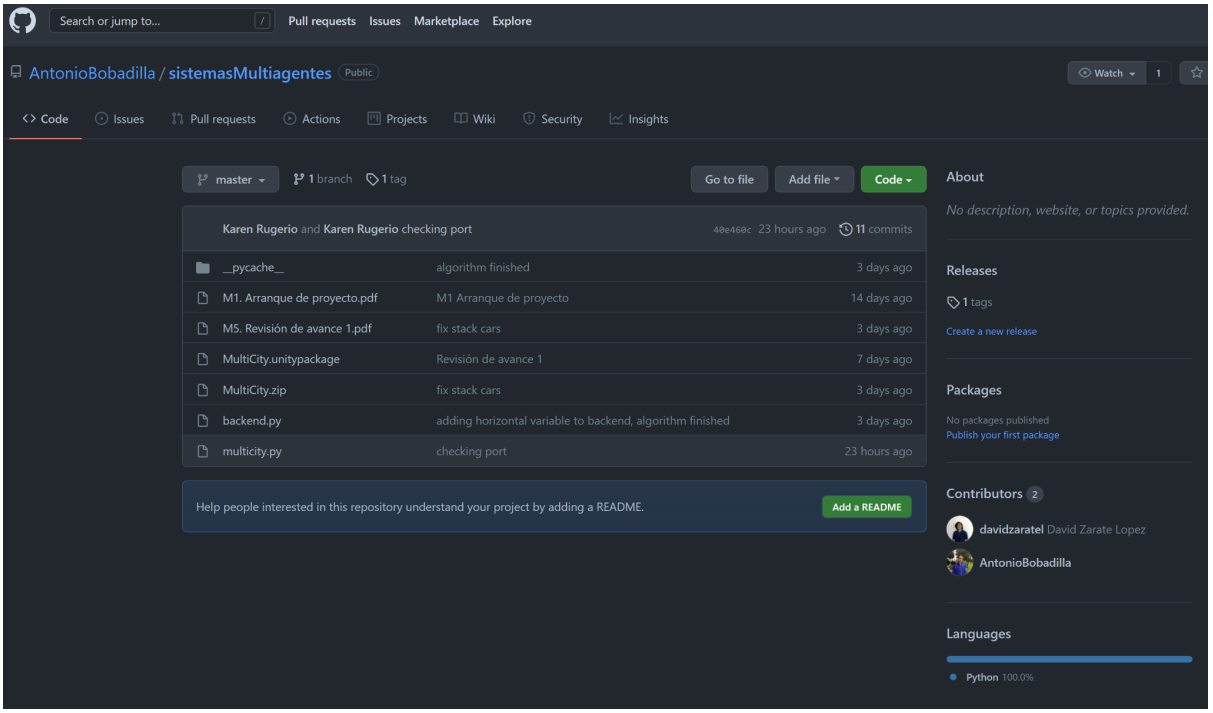
VI. Código de Agentes

Como herramienta colaborativa se creó un Github que nos permite trabajar con el código de Mesa en Python. En cuanto al código de los agentes, se encuentra un avance significativo de un 80%, teniendo listo el comportamiento de los agentes Auto y TrafficLight, teniendo en cuenta dos carriles en la ciudad que tienen un sentido contrario para poder simular una situación de la vida real.

Los automóviles pueden desplazarse de una casilla a otra sin despegarse de su carril y pueden dar vuelta en las esquinas que encuentren. De igual manera, se respetan los estados del semáforo por los automóviles y sólamente se activa un semáforo a la vez para evitar la colisión de los agentes, algo también tomado de la vida real.

Por otro lado, todavía se necesita implementar la conexión del modelo con los gráficos para poder visualizarlo en Unity, sin embargo, esto será realizado en el periodo de tiempo establecido en el plan de trabajo.

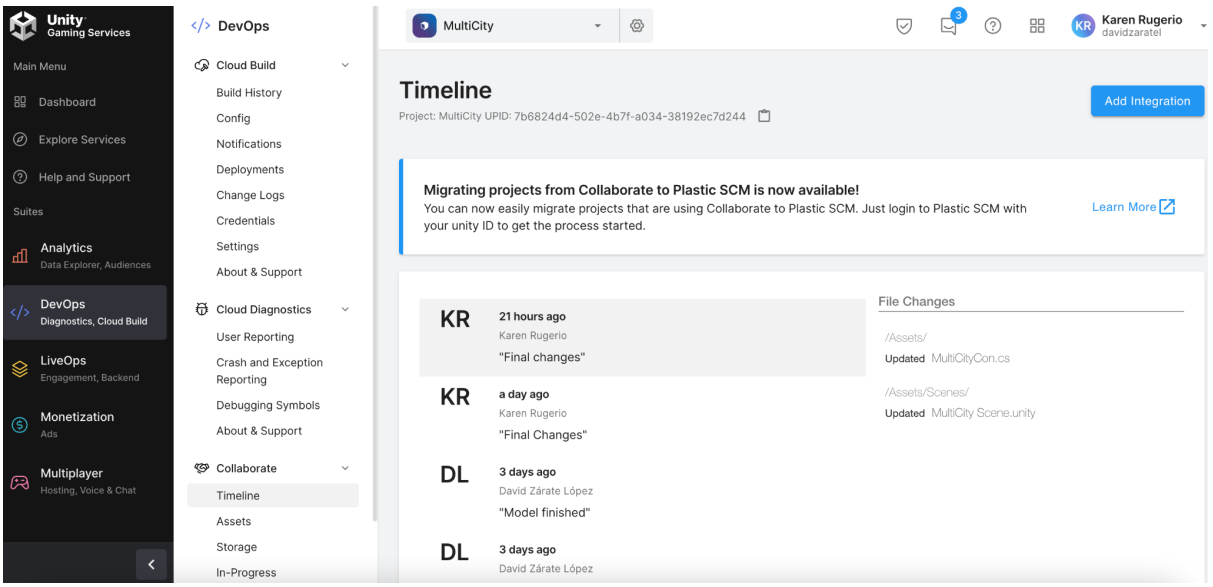
A continuación se muestra el link del repositorio en Github así como una captura de pantalla del mismo: https://github.com/AntonioBobadilla/sistemasMultiagentes



VII. Código de Gráficos

Se creó un proyecto colaborativo en Unity Cloud el cual tiene por nombre *MultiCity* el cual nos servirá como herramienta de manejador de versiones en nuestro reto y documentación. En cuanto al avance de los gráficos, todos los componentes en 3d han sido colocados, sin embargo, todavía no se implementa la conexión entre el código de los agentes y Unity.

A continuación se muestra el link del proyecto en Unity Cloud así como una captura del avance más reciente:

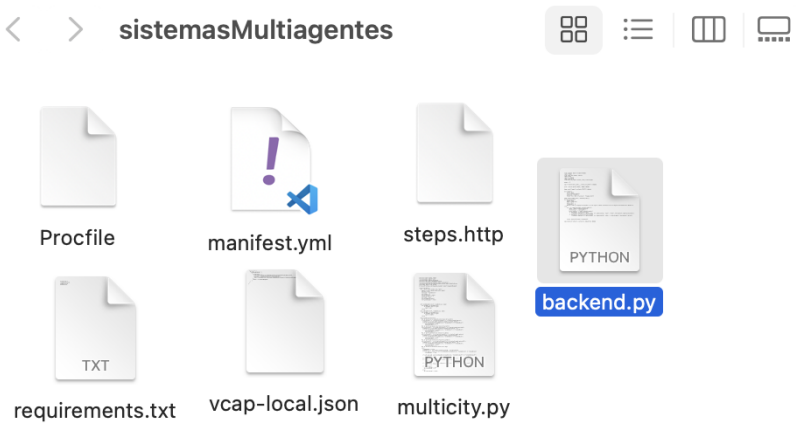


https://dashboard.unity3d.com/organizations/4673289701478/projects/7b6824d4-502e-4b7f-a034-38192ec7d244/collaborate/timeline

VIII. Conexión con IBM

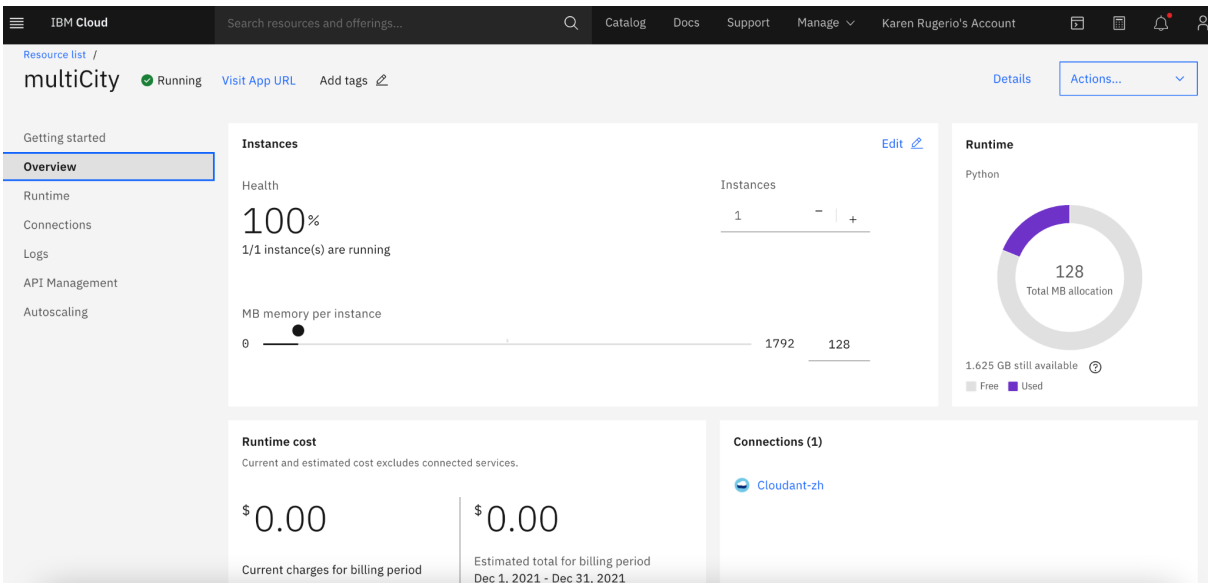
Al finalizar está implementación en Python se desarrolló igualmente una API la cual crea un servidor en Flask el cual se conecta con la implementación de multi agentes y crea una instancia del modelo y manda los datos de cada step, en este caso las coordenadas de los diferentes agentes y diferentes atributos a esta API mediante un formato JSON.

Los archivos publicados se muestran a continuación:



El archivo de *Proctfile* contiene el archivo que se ejecutará en el servidor, *manifest.yml* el nombre del Cloud Foundry app, que en este caso es Multicity, su ruta y el tamaño que se le ha sido asignado en el servidor. Posteriormente tenemos *requirements*, en donde se define las librerías que serán utilizadas para el correcto funcionamiento de la simulación. *Steps* es una forma de poder comprobar que la conexión es exitosa y muestra los post y gets que se tienen en el servidor de IBM. Finalmente se agregan los códigos con los que se ha trabajado a lo largo de todo el bloque, el código de *multicity.py* incluye toda la lógica del programa, la creación de agentes, su posicionamiento en el modelo, etcétera. Y finalmente el *backend.py* contiene la codificación para poder obtener la información del programa principal y transformarla en un JSON, que será recibido por Unity.

Muestra del reto publicado en el servidor:



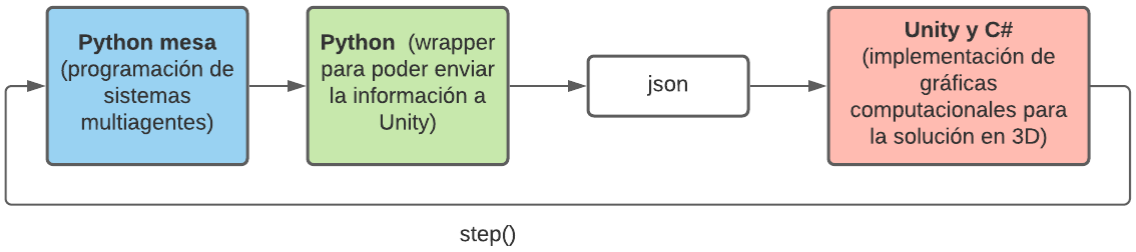
IX. Documentación describiendo el proceso de instalación

Para poder dar solución al reto se hizo uso de tres tecnologías computacionales, la primera fue implementando el código de toda la lógica computacional para poder programar los sistemas multiagentes, en Python llamado *multicity.py* y utilizando la librería de mesa. En esta parte de la solución, se declaran los agentes, se define la posición de los autos, así mismo se generan los semáforos y se definen sus posiciones en el espacio. Toda la lógica computacional con la que se mueven los agentes se encuentra en el código de python mesa.

Posteriormente se trabaja con un wrapper de Python llamado *backend.py* que actúa como mediador entre estas dos tecnologías, este se llama cada vez que los autos avanzan, es decir, realizan un step y llama a las funciones involucradas en python mesa. Así mismo, recibe datos del programa principal y los convierte en un json, que posteriormente ocuparemos para simular las graficaciones en Unity.

El último paso es crear con la herramienta de Unity un ambiente gráfico para poder simular la ciudad, los semáforos y los autos, así como la simulación de sus movimientos en 3D. La combinación de estos elementos gráficos al final, se convierte en un prefab y se le asigna el código de movimiento en el lenguaje de programación C#. En este código se definen las variables con las que se trabajará, las cuales son recibidas gracias al wrapper antes mencionado y haciendo uso de esta información se crea una función la cual nos ayudará a definir el comportamiento gráfico de los agentes en el Unity.

Representación gráfica de lo que sucede en cada step entre los programas para lograr la simulación:



X. Aspectos éticos a considerar

Al haber trabajado en una simulación completamente virtual, es necesario contemplar los aspectos éticos que esta solución conlleva en la vida real, puesto que no es posible controlar totalmente las acciones que llevan a cabo los usuarios y personas en su día a día. Es decir, a pesar de que los semáforos sean programados de la mejor manera posible, siempre puede llegar a pasar que los conductores no respeten los señalamientos de tránsito y puedan ocurrir accidentes.

Por otro lado, al trabajar en una situación que se lleva a cabo a cada momento del día, como lo es el tráfico, puede existir algún error al momento de implementar dicho sistema en un lugar real, por lo cual, es indispensable que cada componente y producto desarrollado no peligre contra la seguridad de las y los usuarios y que de igual manera, se pruebe de manera exhaustiva antes de salir al mercado.

A su vez, también se tienen que considerar otros aspectos éticos como el uso responsable de estas tecnologías con multi agentes, los cuales pueden dar paso para ser utilizados de una manera en que no sean beneficiosos para la sociedad en general.

XI. Reflexiones

José Antonio Bobadilla García: En esta materia pude aprender sobre diferentes tecnologías y ámbitos los cuales me parecen muy importantes e interesantes, como por ejemplo el desarrollo de sistemas multi agentes, con los cuales en este caso se simuló un cruce de calles para darle solución a un problema de movilidad urbana. Esta implementación se desarrolló en Python con el framework mesa, el cual nos ayuda a diseñar multi agentes en Python. A su vez, esto se complementa con conocimientos pasados con APIs para poder realizar la conexión entre Unity y Python desarrollando esta igualmente en Python. Me gustó mucho la conexión entre estas 2 áreas, tanto el modelado tridimensional, junto con el desarrollo de multi agentes. Considero que aprender de estas tecnologías es muy valioso para mi formación profesional ya que pienso realizar la concentración de inteligencia artificial.

David Zárate López: Considero que en esta solución se pudo atacar un problema que existe constantemente en nuestro día a día, situaciones en las cuales nos hemos visto afectados y fue realmente interesante poder trabajar en ello. De igual manera, considero que el modelo desarrollado es exitoso y siento una gran satisfacción al haber podido aprender cómo es que los modelos multiagentes

pueden ser conectados a herramientas externas por medio de un backend, lo cual nos permitió aprovechar al 100% tanto el framework de Mesa con Python y por otro lado, Unity con la graficación computacional. Finalmente, me gustaría mencionar que fue realmente enriquecedor poder aprender sobre Machine Learning y cómo es que podemos realizar agentes que adquieren conocimientos e inteligencia para poder realizar acciones como las que realizamos nosotros día con día.

Karen Rugerio Armenta Considero que esta es una de las problemáticas que más agravan a las grandes ciudades. Ya que en muchas de las ocasiones, los cuellos de botella se hacen en los cruces de semáforos. Por lo que al utilizar los agentes AI, así como gráficas computacionales, se puede modelar dicho entorno, mostrando una propuesta de solución para hacer el tráfico vial más rápido, teniendo como beneficios un crecimiento económico en la región. Me siento muy complacida con el resultado obtenido, ya que los semáforos establecen una excelente comunicación entre ellos y los autos interactúan de manera correcta con el entorno y con los otros agentes involucrados, haciendo la movilidad más óptima. Durante estas cinco semanas, descubrí mi interés por la inteligencia artificial, ya que este es un tema que siempre me había llamado la atención, y este bloque me permitió tener un pequeño acercamiento introductorio al mundo de los agentes y de las gráficas computacionales.

De igual manera conocer de manera teórica la parte de los agentes, es algo que abrió mi perspectiva de las cosas y de los diferentes enfoques que existen. Así como identificar y relacionar que el mundo físico y las leyes de las física son implementadas dentro de los gráficos computacionales, aspectos tan importantes que yo creía sencillos, tienen un valor y complejidad, para el momento de su desarrollo en código, por lo que conocer de ello hacen que tenga interés por seguir aprendiendo y preparándose para entregar excelentes resultados. Todo lo anterior contribuye de una gran manera a mi desarrollo profesional, ya que tengo el compromiso del autoaprendizaje, así como la audacia para resolver problemas de una manera rápida y eficaz. Sin duda alguna, este reto fue algo que disfruté demasiado; estaré esperando gustoso las futuras materias relacionadas con AI, para saber si esto es algo en lo que me gustaría dedicarme a futuro.