



Evidencia 2 - Entrega final reto

Arturo Montes González A01798012

Ares Ortiz Botello A01747848

Andrés Iván Rodríguez Méndez A01754650

Rosa Itzel Figueroa Rosas A01748086

30 de noviembre, 2023

Modelación de sistemas multiagentes con gráficas computacionales

Jorge Adolfo Ramírez Uresti

Oriam Renan De Gyves López

Índice

Revisión 1 - Arranque	3
Identificación de fortalezas y áreas de oportunidad de cada uno	3
Expectativas que tenemos del bloque	4
Listado de lo que esperamos lograr y obtener como equipo de trabajo en el presente bloque	4
Creación de herramientas de trabajo colaborativo	5
Repositorio de Github en el que se guardará toda la documentación y código generados	5
Herramienta de comunicación entre los participantes	5
Propuesta formal del reto	5
Descripción del reto a desarrollar	5
Identificación de los agentes involucrados	6
Plan de trabajo	7
Aprendizaje adquirido y reflexiones individuales	8
Revisión 2 - Modelación agentes	9
Plan de trabajo	9
Descripción del medio ambiente	9
Descripción PEAS de cada agente	10
Diagramas de agente usando AUML	13
Diagrama de organización SMA	14
Diagrama de interacción entre agentes	15
Aprendizaje adquirido y reflexiones individuales	16
Revisión 3 - Avance del 60%	18
Plan de trabajo	18
Aprendizaje adquirido y reflexiones individuales	22
Evidencia 2	23
Análisis de la solución desarrollada	23
Proceso de instalación y ejecución de la simulación	24
Video demostrando el proceso de instalación	26
Video demostrando la simulación	27
Análisis individual de la solución y reflexión de aprendizaje	27

Revisión 1 - Arranque

Conformación del equipo

Integrantes del equipo de trabajo

Arturo Montes González A01798012

Ares Ortiz Botello A01747848

Andrés Iván Rodríguez Méndez A01754650

Rosa Itzel Figueroa Rosas A01748086

Identificación de fortalezas y áreas de oportunidad de cada uno

Arturo: Mi conocimiento en algoritmos y estructuras de datos es avanzado, así como mi habilidad para resolver problemas. También, considero que tengo una buena comunicación y capacidad para ayudar a los demás miembros del equipo a resolver problemas.

Ares: El haber trabajado con Unity en uno de mis semestres anteriores podría representar una ventaja debido a que la curva de aprendizaje no será tan grande, sin embargo, hay una gran área de oportunidad en cuanto al modelado en 3D, debido a que sólo he trabajado con modelado en 2D. Asimismo, considero que puedo mantener una buena comunicación con mi equipo, lo que facilitará el intercambio de ideas y la propuesta de soluciones en caso de ser necesario.

Andrés: El semestre pasado ya tuve la experiencia de trabajar en Unity por ello considero que es una de mis fortalezas, también para mí es muy interesante trabajar en temas relacionados con la inteligencia artificial así que será fácil adquirir los conocimientos para crear el sistema, considero que una de mis fortalezas es el trabajo en equipo, se organizarme bien.

Rosa: Considero que una de mis fortalezas dentro de este proyecto es mi conocimiento previo en modelación 3D y de Unity. De la misma forma, considero que mi desempeño dentro de los trabajos en equipo es bueno, con el objetivo de siempre mejorar y desarrollar un buen proyecto.

Expectativas que tenemos del bloque

Arturo: Obtener conocimientos fundamentales para el funcionamiento y desarrollo de algoritmos de gráficas computacionales, tales como: APIs gráficos, matemáticas, procesamiento paralelo, etc. También espero que entender algoritmos para simular agentes sea de gran utilidad para esclarecer mi futuro profesional, ya que me parece interesante.

Ares: Durante este bloque espero poder adquirir un buen conocimiento sobre el manejo de sistemas multiagentes y gráficas computacionales. A pesar de que esas áreas de la tecnología no sean algo que busque ejercer en mi futuro laboral, me intriga poder entender conceptos como pipelines gráficos, rendering, rasterización, primitivas geométricas, agentes inteligentes, entornos dinámicos, etc. Además, me interesa mucho el poder simular una ciudad y modelar cómo será el flujo vehicular en ella.

Andrés: Espero aprender mucho sobre los sistemas multiagentes para poder aplicarlo en futuros proyectos, también espero aprender sobre las gráficas computacionales, específicamente sobre su funcionamiento y creación, para así poder entender cómo es que funcionan en los proyectos en videojuegos.

Rosa: Aprender sobre gráficas computacionales, creación y desarrollo de agentes para simular un entorno de nuestra vida cotidiana. Me interesa comprender a un nivel más profundo el funcionamiento de agentes inteligentes y cómo es que se pueden aplicar en diversas simulaciones.

Listado de lo que esperamos lograr y obtener como equipo de trabajo en el presente bloque

- Demostrar cómo un sistema de transporte público bien diseñado hace más eficiente el flujo de tránsito. Específicamente, demostrar que al disminuir la cantidad de agentes, se disminuye la complejidad en la comunicación y, en consecuencia, la complejidad del sistema disminuye, obteniendo un mejor funcionamiento en el sistema.
- Presentar un modelo realista del mundo con ayuda de gráficas computacionales y algoritmos para la simulación de agentes inteligentes.

Compromisos personales para lograrlo

Arturo: Me comprometo a dedicar 2 horas por semana al área de gráficas computacionales, esto implica: practicar con herramientas y lenguajes para gráficas computacionales, estudiar teoría y aplicar lo aprendido en las materias.

Ares: Me comprometo a cumplir en tiempo y forma con todas las actividades y labores que se me designen. Consultaré con mi equipo cualquier duda y/o cuestión que surja a lo largo del desarrollo del proyecto, y me aseguraré de tener un buen entendimiento de los medios y herramientas con los que trabaje así como con los que mis compañeros usen.

Andrés: Me comprometo a realizar todas las tareas en las que esté asignado para la elaboración del proyecto, así como a aprender las nuevas herramientas que estaremos utilizando.

Rosa: Para este bloque me comprometo a ayudar a mis compañeros, cumplir con mis deberes tanto del proyecto como de actividades individuales y aportar propuestas e ideas de mejora dentro de la organización y desarrollo de nuestra entrega final.

Creación de herramientas de trabajo colaborativo

Repositorio de Github en el que se guardará toda la documentación y código generados

[Repositorio Equipo 1](#)

Herramienta de comunicación entre los participantes

- Whatsapp
- Hive
- Discord

Propuesta formal del reto

Descripción del reto a desarrollar

El reto a desarrollar consiste en plantear una propuesta de solución ante la problemática de la congestión vial en la Ciudad de México con el objetivo de reducirla, para ello tenemos que implementar una representación gráfica de un sistema de multiagentes que programaremos con el uso de python específicamente usando la librería Mesa, pudiendo visualizar la

simulación con el uso de Unity. Dentro de nuestra ciudad implementaremos distintos medios de transporte público como metrobús y camión. Cumpliendo nuestro objetivo de eficientar el tránsito el metrobús contará con estaciones y carril exclusivo. El camión tendrá paradas designadas para eficientar el tiempo y evitar la generación de congestión vehicular.

Identificación de los agentes involucrados

Agentes:

- Vehículo: los vehículos podrán tomar decisiones sobre los caminos que puedan tomar, así como también decidirán detenerse o avanzar, también serán autónomos porque nadie los estará controlando, serán capaces de comunicarse con los distintos agentes haciendo uso de sus luces direccionales.
- Camiones: los camiones tienen rutas designadas a seguir, pero pueden moverse libremente entre carriles. También deben hacer paradas y bajadas en sitios designados. Podrán comunicarse con los demás agentes.
- Metrobús: los metrobuses tienen rutas designadas y únicamente pueden moverse en su carril designado y deberán hacer paradas y bajadas en lugares designados. Podrán comunicarse con los demás agentes.
- Peatones: los peatones se podrán mover por donde ellos quieran lo que demuestra su autonomía, también tomarán decisiones sobre cómo pasar la calle o por dónde caminar.

Relaciones:

- Agentes - Semáforo: Los agentes actúan dependiendo del color del semáforo, avanzan cuando es verde, se detienen cuando es rojo. Esto puede generalizarse a la interacción de cualquier agente con una señal de semáforo. Por ejemplo, tanto una persona como un vehículo se detiene al ver un semáforo en rojo y avanzan cuando el semáforo muestra la señal verde, contemplado esto, cada agente actúa de acuerdo a la señal de semáforo que le corresponde, siendo los peatones quienes interactúan con el semáforo peatonal y los vehículos con el semáforo vehicular.
- Transporte - Peatón: un transporte no puede atropellar a un peatón, el peatón se puede subir y bajar de un transporte.
- Transporte público - Persona: el transporte público se detiene para bajar y/o subir a una persona o más cuando llega a una parada.

- Vehículo - Vehículo: sea *A* un vehículo y *B* un vehículo distinto:
 - *A* debe guardar una distancia mínima respecto a *B*.
 - Si *B* frena, entonces *A* debe frenar al ver sus luces.
 - Si *B* va a girar para la izquierda o derecha, entonces *B* activará sus luces y *A* disminuirá su velocidad para dejar maniobrar a *B*.

Plan de trabajo

Para llevar a cabo el arranque del proyecto, diseñamos el siguiente plan de trabajo, asegurándonos que la distribución de tareas fuera justa.

Actividades:

- Reunión de inicio del proyecto para definir objetivos y roles
- Crear el documento inicial
- Identificar fortalezas y áreas de oportunidad de cada integrante
- Definir expectativas y metas para el proyecto
- Definir compromisos personales
- Crear repositorio de Github
- Identificar los agentes de nuestro proyecto y definir sus relaciones/interacciones
- Crear los diagramas de clase y protocolos de interacción
- Redactar aprendizajes obtenidos y reflexiones individuales

Se llevó un monitoreo de lo mencionado anteriormente con ayuda de la herramienta “Hive”, la cual nos permitió visualizar todas las tareas, a quién fueron asignadas y así, poder trabajar en ellas de forma organizada y eficiente. El tiempo que estimamos para realizar todas las actividades definidas fue de 3 días como máximo, dedicando de una a dos horas al día en el documento.

The screenshot shows the Hive application interface. On the left is a sidebar with navigation options: My actions, Project Navigator, Apps, Hive University, Goals, Pinned, and Projects. Under Projects, there is a section titled 'Sistemas multiagente...' which is currently selected. The main area displays a list of tasks categorized by status: 'No empezada (2)', 'En progreso (2)', and 'Completada (5)'. Each task card includes a title, priority level (Carga Baja, Carga Moderada, Muy importante), due date, and a small icon representing the number of users assigned to the task.

Estado	Tarea	Prioridad	Fecha	Usuarios Asignados
No empezada (2)	Hacer entrega del URL en Canvas	Carga Baja Muy importante	Nov 7	1
No empezada (2)	Diagramas de clase (identificación de agentes)	Carga Moderada Muy importante	Nov 7	1
En progreso (2)	Identificar fortalezas y áreas de oportunidad	Carga Baja	Nov 6	2
En progreso (2)	Commit del documento al repositorio	Carga Baja Muy importante	Nov 7	1
En progreso (2)	Diagramas de protocolos de interacción	Carga Moderada Muy importante	Nov 7	2
Completada (5)	Crear el documento inicial	Carga Baja Importante	Nov 5	1
Completada (5)	Reunión inicial para definir roles	Carga Baja Muy importante	Nov 6, 3:00 pm	2
Completada (5)	Crear repositorio de Github	Carga Baja Muy importante	Nov 6	1

Figura 1. Tareas establecidas en Hive

Aprendizaje adquirido y reflexiones individuales

Arturo: Comprendí en una mayor medida que es un agente y, más importante aún, cuando es bueno considerar a algo como agente o no. Es importante modelar únicamente lo que nos interesa, ya que esto nos permite tener un objetivo más claro y entender la aportación de cada agente al sistema que se quiere modelar.

Ares: Logré aprender sobre qué es un agente y la forma en que podemos estructurarlo para posteriormente desarrollarlo dentro de nuestra simulación. Aprendí que para realizar una simulación es necesario considerar diversos factores dentro nuestro plan de trabajo para poder realizar una simulación con los objetivos planteados.

Andrés: Aprendí sobre las características que debe tener un agente, así como de las relaciones e interacciones que los agentes pueden tener entre agente y objetos, en general me di cuenta que para hacer el proyecto debemos tener en consideración muchos factores para que la simulación salga como se tiene planeado.

Rosa: Durante esta entrega logré aprender sobre el diseño de agentes, el desarrollo de diagrama de clase de cada agente, las relaciones entre agentes y sus respectivos diagramas de interacción. Finalmente, pude reflexionar sobre todos los pasos y procesos necesarios para desarrollar este proyecto, así como las herramientas que debemos utilizar.

Revisión 2 - Modelación agentes

Plan de trabajo

En esta etapa del proyecto, se definieron diversas actividades para poder lograr tener el entregable en tiempo y forma.

Actividades:

- Reunión con el profesor para revisión y retroalimentación de los diagramas realizados en la entrega anterior
- Edición del documento para agregar los puntos de la revisión 2
- Agregar un índice
- Corregir los diagramas necesarios
- Hacer la descripción del medio ambiente
- Hacer la descripción PEAS de cada agente
- Diseñar un diagrama de organización SMA
- Documentar aprendizajes adquiridos y reflexiones individuales
- Subir documento a nuestro github
- Hacer la entrega en canvas

Descripción del medio ambiente

El medio ambiente de la simulación de congestión vial en la Ciudad de México presenta características distintivas que impactan directamente en el rendimiento de los agentes involucrados (automóviles, metrobuses, camiones y peatones). Estas son esenciales para comprender cómo los agentes interactúan y toman decisiones en su entorno dinámico. A continuación se brindará una descripción y justificación detallada de las cinco características de nuestro entorno:

1. Accesible vs Inaccessible

El medio ambiente es 100% accesible dado que los sensores de los agentes permiten recopilar información relevante del entorno. Los agentes pueden percibir semáforos, señales de tránsito, otros vehículos y peatones, lo que facilita la toma de decisiones informadas.

2. Deterministic vs Non-deterministic

Nuestro entorno es en gran medida no determinista (70%) debido a la naturaleza dinámica del tráfico urbano. Aunque los agentes pueden anticipar y reaccionar según las señales y eventos detectados, las acciones de otros agentes y las condiciones del tráfico pueden cambiar rápidamente, lo que introduce cierta incertidumbre en el sistema.

3. Episodic vs Non-episodic

Nuestro medio ambiente es 75% episódico, puesto que cada agente experimenta episodios que incluyen percepciones y acciones específicas. Los eventos, como semáforos que cambian de color, señales de tránsito y movimientos de otros agentes, definen los episodios. No obstante, los episodios no son totalmente independientes, ya que las interacciones entre agentes y las condiciones del tráfico pueden influir en eventos futuros.

4. Static vs Dynamic

El entorno es 100% dinámico debido a que las condiciones del tráfico, la presencia de peatones y otros vehículos, así como los cambios en las señales de tránsito y semáforos, se ven modificadas constantemente. Aunque el entorno puede no cambiar a un ritmo extremo, la dinámica del tráfico implica una constante adaptación por parte de los agentes.

5. Discrete vs Continuous

El medio ambiente tiene aspectos continuos (80%), ya que las variables como la velocidad de los vehículos y la posición de los peatones, pueden variar de manera continua. Las acciones, aunque discretas en ejecución (avanzar, detenerse, cambiar de carril), pueden ser influenciadas por factores continuos, como la velocidad actual del vehículo o la posición relativa de otros agentes en el entorno.

Descripción PEAS de cada agente

Automóvil

Performance

- El automóvil tiene la capacidad de trasladar personas eficientemente por la ciudad, obedeciendo las normas de tránsito y asegurando la seguridad de los ocupantes. Su rendimiento se evalúa en términos de eficiencia en la navegación, respeto de las señales y semáforos y la prevención de colisiones.

Environment

- Se enfrenta a un entorno urbano complejo con calles y avenidas, semáforos, señales de tránsito, vehículos y peatones. La capacidad del automóvil para interpretar y reaccionar ante estos elementos determina su rendimiento en la reducción de la congestión vial.

Actuators

- El automóvil está equipado con un motor para la propulsión, frenos para detenerse, luces para indicar su estado, direccionales para cambiar de carril, y claxon para comunicar algún tipo de mensaje con otros agentes del entorno, como alertar a peatones u otros automovilistas.

Sensors

- Utiliza sensores para percibir el entorno, detectando semáforos, señales de tráfico, otros vehículos y peatones. Estos sensores son esenciales para tomar decisiones informadas sobre la velocidad, la detención y las maniobras.

Metrobús

Performance

- El metrobús busca proporcionar un transporte público eficiente, deteniéndose en las estaciones designadas, respetando semáforos y garantizando la seguridad de los pasajeros. Su rendimiento se evalúa según la puntualidad, la seguridad y la eficacia en la gestión de pasajeros.

Environment

- Opera en rutas de transporte público con estaciones, semáforos, peatones y otros vehículos. La capacidad del metrobús para integrarse en este entorno dinámico es crucial para reducir la congestión.

Actuators

- Utiliza un motor para la propulsión, frenos para detenerse, puertas para permitir el embarque y desembarque de pasajeros, y un claxon para alertar a peatones y automovilistas.

Sensors

- Los sensores de un metrobus detectan semáforos, estaciones y la presencia de peatones para coordinar la detención y apertura de puertas, asegurando una interacción segura con el entorno.

Camión

Performance

- Un camión tiene la tarea de proporcionar transporte público, gestionando paradas, semáforos, vehículos y garantizando la seguridad de los pasajeros. Su rendimiento se mide en términos de eficacia en el transporte y la gestión del tráfico.

Environment

- Opera en rutas de transporte público, enfrentándose a paradas, semáforos, peatones y otros vehículos. La capacidad del camión para adaptarse a situaciones complejas en el tráfico es esencial.

Actuators

- Utiliza un motor para la propulsión, frenos para detenerse, direccionales para cambiar de carril, puertas para permitir el embarque y desembarque de pasajeros, y un claxon para alertar a peatones y automovilistas.

Sensors

- Los sensores de un camión detectan paradas, vehículos parados, semáforos y peatones, facilitando así la toma de decisiones informadas sobre la velocidad y la detención.

Peatón

Performance

- El peatón tiene como objetivo llegar a su destino de manera segura, respetando semáforos, señales y evitando conflictos con vehículos. Su rendimiento se evalúa en términos de eficiencia y seguridad en la navegación peatonal.

Environment

- Un peatón se desplaza por calles, cruces peatonales, semáforos, señales y banquetas. La capacidad del peatón para interactuar con estos elementos de manera segura es esencial para reducir riesgos.

Actuators

- Utiliza sus piernas para caminar, adaptando su velocidad según las señales y condiciones del entorno. Asimismo, utiliza sus brazos o manos para pedir el paso o pedir la parada de algún transporte público.

Sensors

- Sus sentidos humanos (como el de la vista y oído) le permiten percibir semáforos, señales, vehículos y pasos de peatones, tomando decisiones informadas sobre cuándo cruzar y cuándo detenerse

Diagramas de agente usando AUML



Figura 2. Diagrama de clase de automóvil y metrobús

Humano
Grupo: Peatón
Rol: Llegar a su destino
Eventos:
Detectar vehículos
Detectar parada/estación
Detectar semáforos
Detectar señales
Detectar banqueta
Evento-acción:
Vehículos -> Esperar a que pare para cruzar
Parada -> Esperar al camión para subir o bajar en la parada
Semáforo -> Si es rojo parar, si es verde seguir
Señal -> Cruzar
Banqueta -> Caminar solo por ese espacio

Figura 3. Diagrama de clase de humano

Diagrama de organización SMA

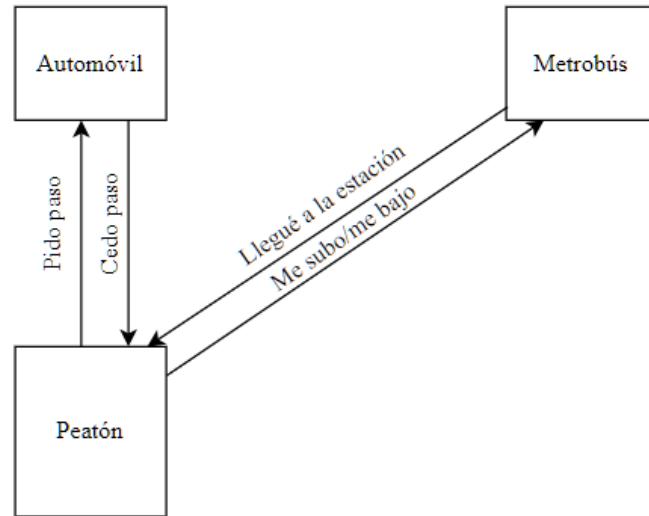


Figura 4. Diagrama de organización de nuestro SMA

Diagrama de interacción entre agentes

Agentes (Metrobús, Automóvil y Persona)

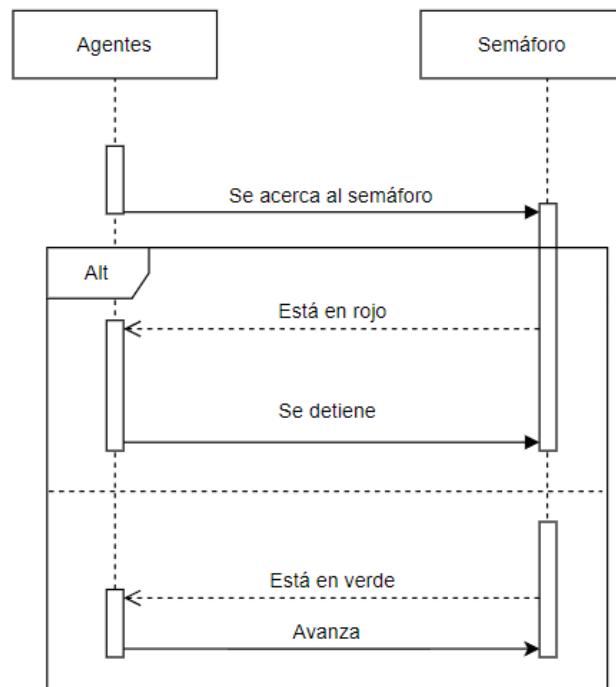


Figura 5. Diagrama de secuencia Agentes (Metrobús, Automóvil y Persona) - Semáforo

Transporte(Metrobús, Automóvil) - Semáforo

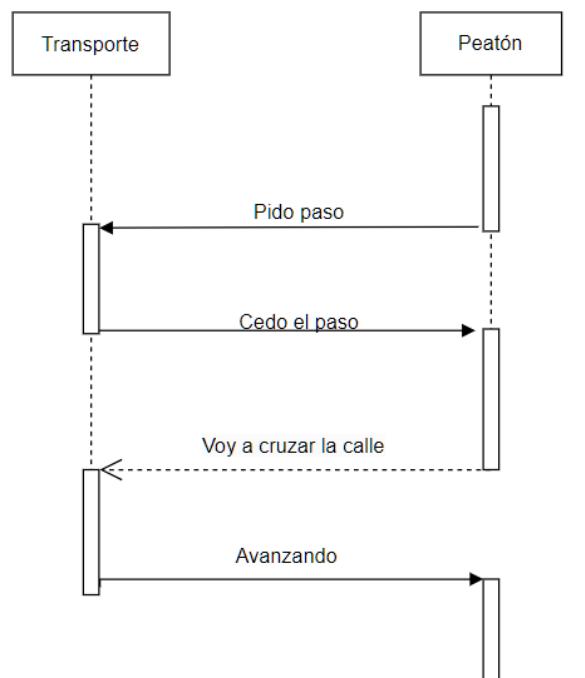


Figura 6. Diagrama de secuencia Transporte - Peatón

Transporte público(Metrobús) - Semáforo

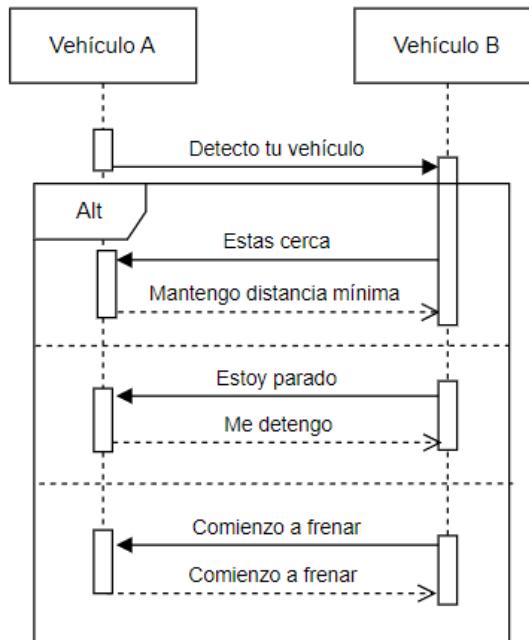


Figura 7. Diagrama de secuencia Vehículo - Vehículo

Aprendizaje adquirido y reflexiones individuales

Arturo: En el desarrollo de esta entrega pudimos esclarecer en mejor medida la arquitectura del sistema. Al definir de manera clara los actuadores, sensores y ambiente, quedan más claros los componentes de sistema y la información que cada agente debe enviar y recibir para comportarse de manera esperada.

Ares: Al realizar este entregable pude aprender más sobre los aspectos que definen a un medio ambiente y las características necesarias para describir a nuestros agentes (performance, environment, actuators, sensors). Me pude dar cuenta que tener en cuenta esos dos aspectos es muy importante para lograr un mayor y mejor entendimiento de nuestro sistema y cómo se darán las interacciones en él. Asimismo, comprendí cuál es la manera correcta de hacer diagramas de interacción entre agentes y diagramas de organización.

Andrés: A lo largo de este entregable noté que definimos de una manera más detallada algunas de las funcionalidades que nuestros agentes van a tener, también del cómo debemos tener nuestro ambiente para que nuestros agentes puedan realizar los movimientos que deben hacer, con ayuda de este entregable en equipo pudimos definir más estrategias para tener un correcto funcionamiento de nuestro sistema multiagentes, también entendí cómo hacer un diagrama de interacción de manera correcta.

Rosa: Durante este entregable definimos más el entorno en el que desarrollaremos la simulación de la Ciudad de México. De la misma forma, logramos definir más las acciones e interacciones entre todos los agentes dentro de la ciudad. En términos generales, esta entrega tuvo el objetivo de ayudarnos a plantear todas nuestras ideas relacionadas al diseño de la simulación que queremos desarrollar con la meta de hacer una ciudad más eficiente y sin congestión vial.

Revisión 3 - Avance del 60%

Plan de trabajo

Para esta tercera revisión fue necesario establecer las actividades pendientes y el tiempo en el que se realizarán, así como los responsables de llevarlas a cabo, el intervalo de esfuerzo estimado y la fecha en las que se realizarán y terminarán. Adicionalmente, mostraremos cuáles son las actividades que ya han sido finalizadas, realizando una comparación de diferencia del tiempo que nos tomó realizarlas y el tiempo que inicialmente estimamos.

Actividades pendientes					
Número de actividad	Actividad	Responsable	Esfuerzo estimado	Tiempo estimado	Fecha de inicio
1	Actualizar el plan de trabajo	Ares Ortiz Botello	1 día	30 minutos	20/11/2023
2	Definir las nuevas tareas por hacer	Todo el equipo	1 día	30 minutos	20/11/2023
3	Hacer las correcciones necesarias de la descripción del medio ambiente	Ares Ortiz Botello	1 día	15 minutos	21/11/2023
4	Definir el aprendizaje adquirido como equipo	Todo el equipo	1 día	15 minutos	22/11/2023
5	Realizar reflexiones individuales	Todo el equipo	1 día	15 minutos	22/11/2023
6	Subir documento de revisión 3 al Github	Andrés Iván Rodríguez Méndez	1 día	5 minutos	22/11/2023
7	Subir el link de Github a canvas	Andrés Iván Rodríguez Méndez	1 día	5 minutos	22/11/2023
8	Tener un avance del 90%	Rosa Itzel Figueroa Rosas	5 días	3 horas cada día	22/11/2023

	en Unity				
9	Tener un avance del 95% en Python y MESA	Arturo Montes González y Andrés Iván Rodríguez Méndez	4 días	2 horas cada día	22/11/2023
10	Empezar con el servidor y API	Ares Ortiz Botello	2 días	1 hora y 30 minutos	23/11/2023
11	Tener un avance del 80% en el servidor y API	Ares Ortiz Botello	4 días	3 horas cada día	23/11/2023
12	Hacer pruebas unitarias	Todo el equipo	2 días	2 horas y 30 minutos cada día	25/11/2023
13	Hacer pruebas de integración	Todo el equipo	2 días	2 horas y 30 minutos cada día	25/11/2023
14	Agregar personas y metrobús en Python y Mesa	Arturo Montes González y Andrés Iván Rodríguez Méndez	1 día	3 horas	23/11/2023
15	Conectar Unity con el servidor	Ares Ortiz Botello y Rosa Itzel Figueroa Rosas	1 día	1 hora	23/11/2023
16	Conectar Python con el servidor	Arturo Montes González, Ares Ortiz Botello y Andrés Iván Rodríguez Méndez	1 día	1 hora	24/11/2023
17	Realizar pruebas/peticiones del servidor a Unity	Ares Ortiz Botello y Rosa Itzel Figueroa Rosas	2 días	2 horas y 30 minutos cada día	24/11/2023
18	Tener el reto completo en un 100%	Todo el equipo	7 días	4 horas cada día	21/11/2023
19	Diseñar la presentación final	Todo el equipo	2 días	1 hora cada día	26/11/2023

20	Elaborar documentos de entregas finales	Todo el equipo	2 días	2 horas cada día	28/11/2023
21	Subir evidencias a elumen y canvas	Todo el equipo	1 día	30 minutos	29/11/2023

Figura 8. Tabla de actividades pendientes

Actividades finalizadas					
Número de actividad	Actividad	Responsable	Tiempo estimado	Tiempo real	Diferencia entre tiempos
1	Reunión de inicio del proyecto para definir objetivos y roles	Todo el equipo	1 hora	1 hora	Ninguna
2	Crear el documento inicial	Ares Ortiz Botello	5 minutos	3 minutos	2 minutos
3	Crear repositorio de Github	Arturo Montes González	10 minutos	5 minutos	5 minutos
4	Identificar fortalezas y áreas de oportunidad de cada integrante	Todo el equipo	20 minutos	10 minutos	10 minutos
5	Definir expectativas y metas para el proyecto	Todo el equipo	30 minutos	20 minutos	10 minutos
6	Definir compromisos personales	Todo el equipo	20 minutos	10 minutos	10 minutos
7	Identificar los agentes de nuestro proyecto y definir sus relaciones/interacciones	Todo el equipo	45 minutos	45 minutos	Ninguna
8	Crear los diagramas de clase y protocolos de interacción	Todo el equipo	1 hora 30 minutos	2 horas	30 minutos
9	Redactar aprendizajes obtenidos y reflexiones individuales	Todo el equipo	15 minutos	10 minutos	5 minutos
10	Empezar a modelar los agentes en Python con el uso de MESA	Arturo Montes González y Andrés Iván Rodríguez	2 horas	2 horas	Ninguna

		Méndez			
11	Empezar a modelar la ciudad en Unity	Rosa Itzel Figueroa Rosas	2 horas	2 horas	Ninguna
12	Subir documento "Revisión 1" a Github	Andrés Iván Rodríguez Méndez	5 minutos	3 minutos	2 minutos
13	Entregar link de Github en canvas	Andrés Iván Rodríguez Méndez	5 minutos	2 minutos	3 minutos
14	Reunión con el profesor para revisión y retroalimentación de los diagramas realizados en la primera entrega	Todo el equipo	2 horas	2 horas	Ninguna
15	Creación del documento para la Revisión 2	Ares Ortiz Botello	5 minutos	3 minutos	2 minutos
16	Agregar un índice	Rosa Itzel Figueroa Rosas	15 minutos	15 minutos	Ninguna
17	Corregir los diagramas necesarios	Rosa Itzel Figueroa Rosas y Ares Ortiz Botello	45 minutos	40 minutos	5 minutos
18	Hacer la descripción del medio ambiente	Todo el equipo	50 minutos	40 minutos	10 minutos
19	Hacer la descripción PEAS de cada agente	Todo el equipo	50 minutos	50 minutos	Ninguna
20	Diseñar un diagrama de organización SMA	Rosa Itzel Figueroa Rosas y Ares Ortiz Botello	30 minutos	40 minutos	10 minutos
21	Documentar aprendizajes adquiridos y reflexiones individuales	Todo el equipo	15 minutos	10 minutos	5 minutos
22	Subir documento al repositorio de Github	Andrés Iván Rodríguez Méndez	5 minutos	5 minutos	Ninguna
23	Hacer la entrega del link en canvas	Andrés Iván Rodríguez Méndez	5 minutos	3 minutos	2 minutos
24	Continuar y avanzar	Rosa Itzel	1 hora	1 hora	Ninguna

	con la modelación de la ciudad en Unity	Figueroa Rosas			
25	Continuar y avanzar con la programación del SMA en Python con el uso de MESA	Arturo Montes González y Andrés Iván Rodríguez Méndez	1 hora	1 hora	Ninguna
26	Hacer la entrega del link en canvas	Andrés Iván Rodríguez Méndez	5 minutos	3 minutos	2 minutos
27	Crear documento para la Revisión 3	Rosa Itzel Figueroa Rosas	5 minutos	3 minutos	2 minutos

Figura 9. Tabla de actividades finalizadas

Aprendizaje adquirido y reflexiones individuales

Arturo: Se ha aprendido mucho de convertir de un modelo con coordenadas discretas a uno con coordenadas continuas. También se ha aprendido en la sincronización y envío de datos entre dos sistemas dependientes para tener una simulación en tiempo real.

Ares: Al haber realizado esta actividad pude darme cuenta de lo importante que es llevar un plan de trabajo actualizado para cada entrega, puesto a que así hay una mayor organización y noción de lo que se tiene que hacer y cómo. De igual forma, con los continuos cambios y correcciones que se han hecho en los diagramas de agentes, he podido entender con más profundidad lo que caracteriza a un SMA y al medio ambiente en el que se desarrolla. Finalmente, empecé a aprender sobre la creación de un servidor usando flask, para así lograr una comunicación entre todos nuestros programas.

Andrés: En general he aprendido mucho sobre la implementación de un sistema multiagentes, gracias al desarrollo de los diagramas y las correcciones que hemos realizado he podido pensar en posibles mejoras para la implementación del reto.

Rosa: Durante esta revisión logré profundizar mis conocimientos y poner en práctica el uso de distintas herramientas dentro del entorno de Unity. Por otro lado, también pude entender de una forma más clara el proceso de desarrollo de los diagramas necesarios para crear un Sistema Multiagente.

Evidencia 2

Análisis de la solución desarrollada

El modelo de multiagentes que utilizamos fue el *Multi-Agent System* (MAS). Decidimos usar este modelo porque necesitábamos un modelo que permitiera una independencia entre los agentes para velocidad y una fácil implementación. Además, todos nuestros agentes compartían un comportamiento similar, por lo que la implementación fue más simple. Las variables que tomamos en cuenta para la implementación de la solución fueron, el número de vehículos particulares que están circulando simultáneamente, el número de personas que circulan por las banquetas para que pudieran ir de un sitio a otro, ver si el metrobus con carril exclusivo aportaba a la mejora del flujo vehicular y a la disminución de vehículos particulares. Todos estos agentes podían interactuar unos con otros, e inclusive interactúan con objetos como semáforos y cruces peatonales, por ejemplo los peatones podían interactuar con los vehículos, los vehículos siempre deben de ceder el paso a los peatones para no atropellarlos, también los peatones interactúan con los semáforos, de esta manera si el semáforo estaba en rojo para los vehículos el peatón podía pasar sin afectar la circulación; otra interacción importante es entre los vehículos, si un vehículo detectaba un vehículo enfrente de él se detenía para evitar una choque entre ellos, contemplando estas interacciones podíamos mejorar el flujo vial porque todos avanzaban sin ningún retraso.

En el apartado gráfico, decidimos usar una representación visual en mesa como prototipo de lo que representamos en Unity. Mesa facilitó este primer prototipo, ya que con las construcción de las clases y sus movimientos era automático. Tras comprobar el buen funcionamiento de este prototipo, procedimos a modelar la ciudad a escala en Unity. Hicimos uso de probuilder y assets de Unity. El uso de Unity nos permitió ver en 3D todo lo que sucedía y de manera más realista, además de que agregamos cámaras en puntos importantes de la ciudad para hacer un mejor análisis.

Elementos destacables de nuestro modelo son el nivel de abstracción de los distintos agentes y su comportamiento. La implementación del comportamiento es una clase heredada a los distintos agentes móviles, por lo que el mantenimiento y la agregación de agentes es muy simple en caso de querer expandir el alcance del proyecto. Además, dejamos un fundamento sólido en la visualización para que, de igual manera, se pueda ampliar fácilmente la visualización

Por otra parte, aunque nuestro modelo es sólido también hay lugar para mejorarlo. Se podría agregar más reactividad a los agentes, para que no tengan un comportamiento tan *fanático*, sino uno más reactivo y realista. También, hay lugar para agregar puentes peatonales y hacer semáforos inteligentes que respondan a la demanda de tránsito de cada avenida. Finalmente, creemos que también hay lugar para mejorar la fluidez de la simulación en Unity.

Proceso de instalación y ejecución de la simulación

Para poder correr y visualizar nuestro proyecto, primeramente se deben de tener instalados Unity (versión 2023.1.18f1), Python (versión 3.12.0) y Flask (versión 3.0.0).

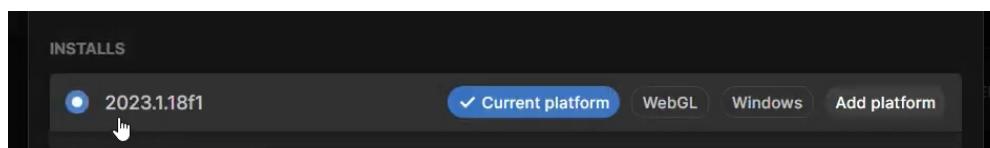


Figura 10. Versión de Unity

A screenshot of a terminal window. The title bar shows 'MINGW64:/c/Users/rousf/OneDrive/Documentos/GitHub'. The command '\$ flask --version' is run, and the output 'Python 3.12.0' is displayed. The terminal interface includes icons for file operations like star, copy, and paste, and tabs for 'NAME', 'CLOUD', and 'M'.

Figura 11. Versión de Python y Flask

Una vez teniendo estas herramientas instaladas con las versiones especificadas, podemos proceder a clonar el repositorio de Github donde se encuentra nuestro proyecto: [Repositorio Equipo 1](#)

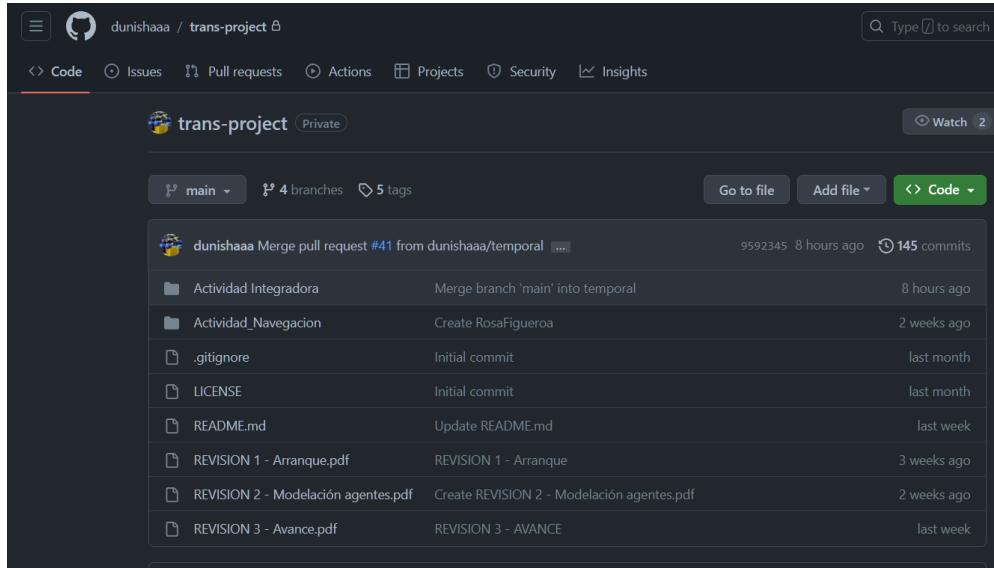


Figura 12. Repositorio de Github

Una vez clonado, abrimos la terminal e ingresamos a la carpeta de Actividad Integradora y posteriormente se debe de entrar a la carpeta de MultiAgentesPython donde se encuentra el servidor.

```
MINGW64:/c/Users/rousf/OneDrive/Documentos/GitHub/trans-project/Actividad Integradora/MultiAgentesPython
Flask 3.0.0
werkzeug 3.0.1

rousf@Rose MINGW64 ~/OneDrive/Documentos/GitHub
$ cd trans-project/

rousf@Rose MINGW64 ~/OneDrive/Documentos/GitHub/trans-project (main)
$ cd Actividad\ Integradora

rousf@Rose MINGW64 ~/OneDrive/Documentos/GitHub/trans-project/Actividad I
ntegradora (main)
$ cd MultiAgentesPython/
```

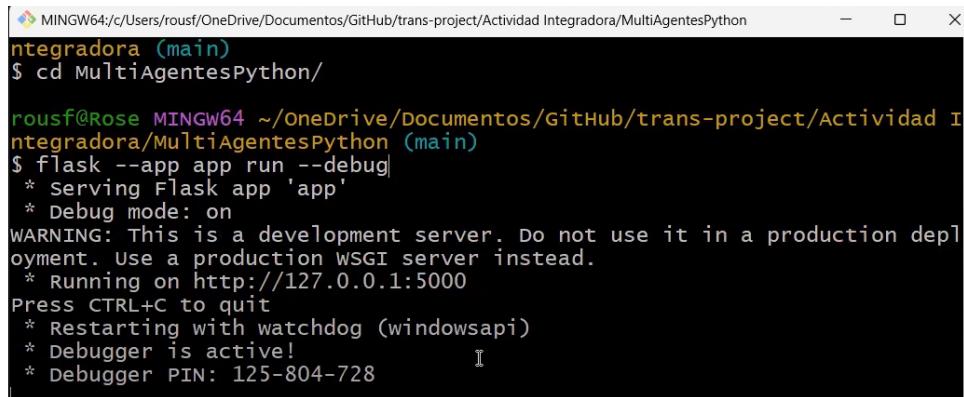
Figura 12. Terminal Gitbash con las carpetas de nuestro proyecto

El siguiente paso será correr el servidor con el siguiente comando:

```
rousf@Rose MINGW64 ~/OneDrive/Documentos/GitHub/trans-project/Actividad Integradora/MultiAgentesPython (ciudadFinal)
$ flask --app app run --debug
```

Figura 13. Comando para correr el servidor

Cuando este se encuentre corriendo, se podrán visualizar los siguientes mensajes en la terminal:



```
MINGW64:/c/Users/rousf/OneDrive/Documentos/GitHub/trans-project/Actividad Integradora/MultiAgentesPython
integradora (main)
$ cd MultiAgentesPython/
rousf@Rose MINGW64 ~/OneDrive/Documentos/GitHub/trans-project/Actividad Integradora/MultiAgentesPython (main)
$ flask --app app run --debug
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 125-804-728
```

Figura 14. Terminal mostrando que el servidor está corriendo

Después de esto, se deberá de importar el proyecto clonado a Unity, seleccionando la carpeta de trans-project\Actividad Integradora\ModeladoCiudad\Ciudad

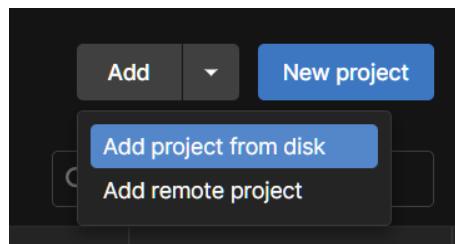


Figura 15. Importación del proyecto desde Unity Hub

Finalmente, cuando este se haya terminado importar, solo se deberá de iniciar la simulación con el botón de “play” que se encuentra en la parte superior media de la pantalla.



Figura 16. Botón de play en Unity

Video demostrando el proceso de instalación

[Proceso de instalación: Video](#)

Video demostrando la simulación

[Simulación de la ciudad](#)

Análisis individual de la solución y reflexión de aprendizaje

Arturo: Personalmente, la realización de este reto me ayudó a solidificar mis conocimientos ya que fue un proyecto en el que fueron muy importantes el uso de algoritmos, diseño de clases y medios de visualización. Además, encontrar este bloque me ayudó a sentar bases sólidas en gráficas computacionales, el área en la que quiero especializarme.

Ares: En conclusión, la realización de este proyecto y el haber cursado este bloque en general, me sirvió como base para empezar a aprender sobre sistemas multiagentes y cómo estos pueden ser modelados con el uso de distintos programas y frameworks. El haber trabajado con Unity para el modelado de la ciudad me permitió conocer herramientas como probuilder y UV Editor, las cuales fueron de gran utilidad para darle personalidad a nuestros prefabs. Además, toda la teoría correspondiente al modelado de agentes, como diagramas de interacción y características PEAS, me permitieron ver que el uso de estos conceptos en la vida real son de suma importancia para la resolución de diversas problemáticas

Andrés: Como conclusión personal del proyecto puedo decir que es bastante importante implementar este tipo de soluciones para encontrar estrategias al problema de la congestión vial en México, el uso de herramientas como unity y mesa nos permite apegarnos mucho a la realidad y demostrar si la solución es viable o no, en lo personal creo que la solución que propusimos es viable ya que nos permite observar que el flujo vial mejora al no tener vehículos que hacen paradas continuas como el transporte público.

Rosa: Para concluir, durante este bloque logré aprender el uso, implementación y diseño de la modelación de agentes. El desarrollo del proyecto me ayudó a poner en práctica las herramientas para el diseño de la ciudad dentro de la herramienta de Unity. En esta entrega tuvimos que arreglar muchos detalles para que la funcionalidad mejorará por lo que identifiqué diversas formas en la que se puede mejorar nuestra entrega.