Evidencia 1. Actividad Integradora

Modelación de sistemas multiagentes con gráficas computacionales

Omar Jiménez Armendáriz	A01732097
Francisco Rocha Juárez	A01730560
Alejandro Alfonso Ubeto Yañez	A01734977

1.- Introducción

1.1.- Resumen

En el presente documento se muestra la implementación del equipo 5 para la resolución de la actividad integradora de la unidad de formación Modelación de sistemas multiagentes con gráficas computacionales utilizando el entorno de ejecución de JavaScript, Node Js.

Accede al repositorio en BitBucket del proyecto y también a su versión de GitHub. Estos son repositorios que recopilan los contenidos de la unidad de formación y evidencias de la misma. En ambos se ha realizado la invitación a los docentes a cargo del bloque para que sean capaces de revisar el progreso del equipo en cualquier momento.

1.2.- Explicación de actividad integradora

Asumiendo el papel de un orgulloso propietario de **5 robots nuevos** y un almacén lleno de cajas, el equipo deberá hallar la forma de organizar el desorden que dejó el dueño anterior de un almacén utilizando robots para organizar las cajas apilando una sobre otra para iniciar un negocio exitoso.

Cada uno de estos robots está equipado con ruedas omnidireccionales y, por lo tanto, puede conducir en las cuatro direcciones, además pueden recoger cajas en celdas de cuadrícula adyacentes con sus manipuladores, luego llevarlas a otra ubicación e incluso construir pilas de hasta cinco cajas.

Todos los robots están equipados con la tecnología de sensores más nueva que les permite recibir datos de sensores de las cuatro celdas adyacentes. Por tanto, es fácil distinguir si un campo está libre, es una pared, contiene una pila de cajas (y cuantas cajas hay en la pila) o está ocupado por otro robot. Los robots también tienen sensores de presión equipados que les indican si llevan una caja en ese momento.

Debido a que estos carecen de un software de gestión de agentes múltiples de última generación la actividad integradora consistirá en enseñar a sus robots a colaborar para ordenar las cajas del almacén para que terminen apiladas en pilas de cinco.

2.- Desarrollo

2.1.- Diagrama de clases

El modelo contendrá un total de 4 tipos de agentes diferentes:

- **Robot**: agente que se moverá por todo el espacio en búsqueda de cajas para ordenar.
- **Caja**: agente que será detectado por un robot y que será transportado a un estante que aún no esté lleno.
- **Estante**: agente que almacenará en su interior hasta un máximo de 5 cajas y que sirve como destino para los robots que las transportan.
- **Wall Block**: agente que sirve como obstáculo para el trayecto de los robots, viéndose obligados a esquivarlos para llegar hacia un estante.

Sus atributos y métodos se muestran a continuación en el siguiente diagrama de clases:

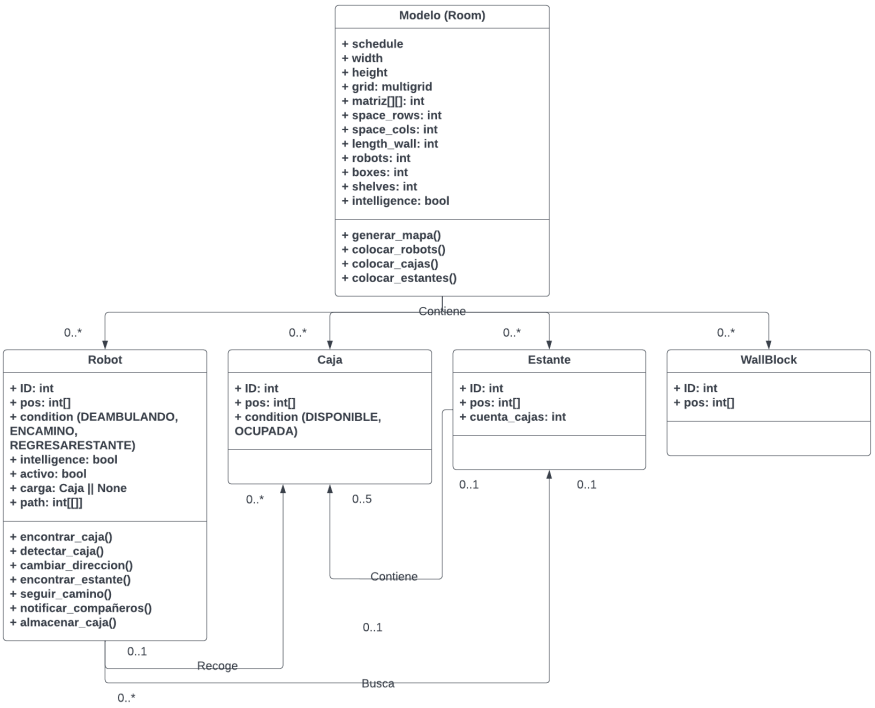

Imagen 1.- Diagrama de clase presentando los distintos agentes involucrados

2.2.- Diagrama de protocolo de interacción

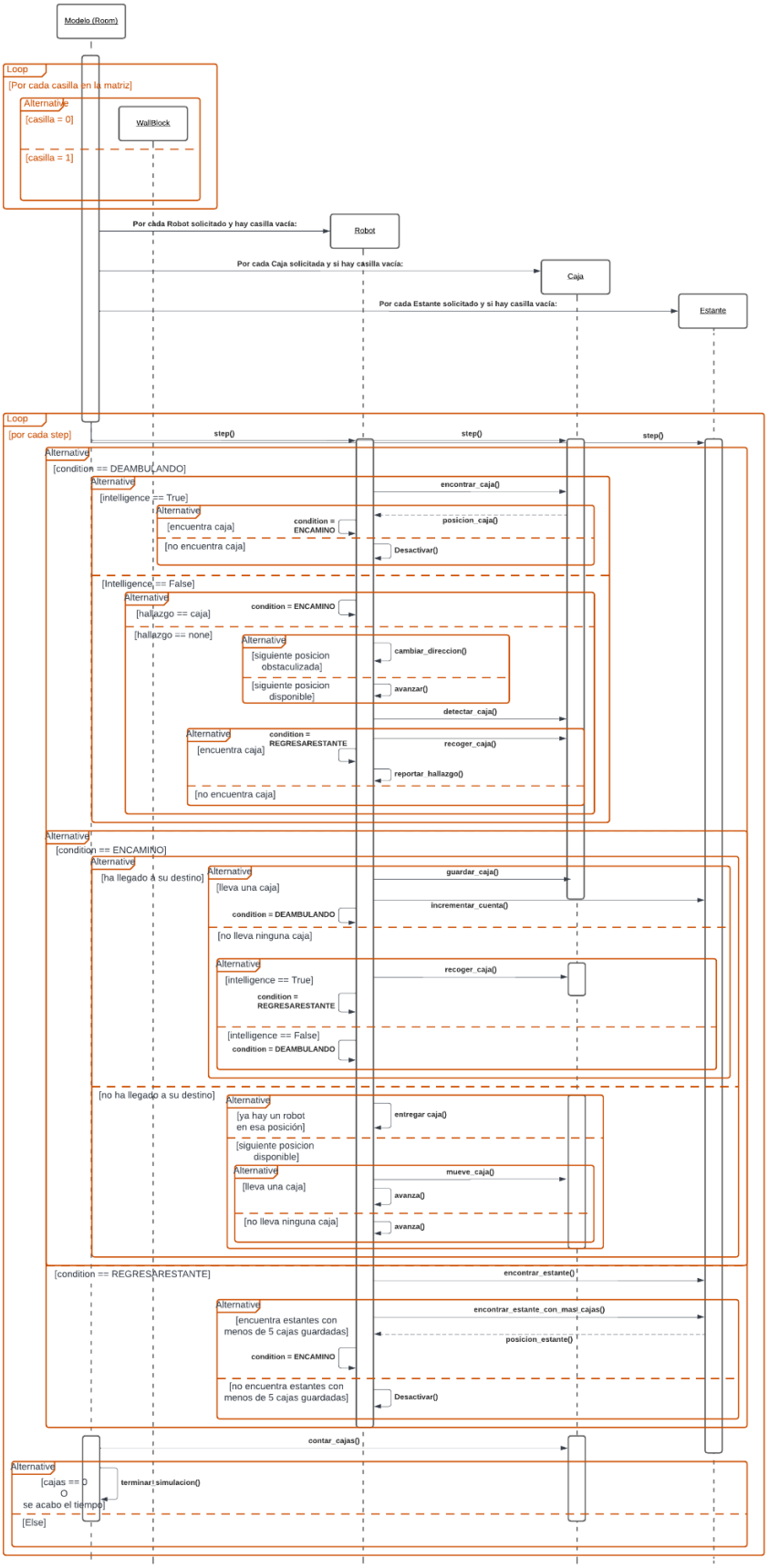

Imagen 2.- Protocolos de interacción entre los agentes involucrados

2.3.- Estrategia cooperativa para la resolución del reto

- Los robots implementados en esta solución cuentan con dos tipos de comportamiento disponibles: **modo omnisciente** y **no omnisciente** y la estrategia que adopten para solucionar la problemática planteada depende de su comportamiento activo.

  En ambos modos existen las siguientes estrategias en todo momento:

  - La primera estrategia es que cuando exista una colisión entre agentes, ya sea porque sus caminos se atraviesen o exista una cola para llegar a una estantería o caja, los dos agentes involucrados intercambian instrucciones y cajas. De esta manera, van pasándose las cajas hasta que llegue al objetivo o intercambian instrucciones hasta que un agente sea capaz de seguir la orden sin problemas.

  - La segunda estrategia es que cuando un agente se encuentre atrapado entre cajas y no pueda encontrar un camino hacia una estantería óptima, sus compañeros lo ayudarán despejando las cajas que lo obstaculizan hasta que pueda salir.

- **Modo omnisciente:** en este modo los robots saben previamente dónde están todas las cajas y las estanterías:

  - La estrategia adicional que adoptan es ponerse de acuerdo sobre qué caja va a recoger cada uno y a qué estantería la va a llevar. De esta forma, cada robot sabe específicamente a qué caja ir y a qué estantería llevarla, de forma que no existan dos robots que tengan como objetivo la misma caja y que no coloquen cajas de más en las estanterías.

- **Modo no omnisciente:** en este modo los robots saben dónde están las estanterías y el estado de estas, pero no tienen información sobre las cajas, así que proceden a buscarlas de acuerdo con la siguiente heurística:

    - Avanzan en línea recta hacia una dirección en concreto hasta que se topen con un obstáculo. En cuanto se topen con uno, cambian de dirección hacia la siguiente. Las direcciones se eligen de forma aleatoria entre las siguientes: derecha, arriba, izquierda, abajo, derecha.

    - Si en algún momento encuentran una caja a su alrededor durante su recorrido, interrumpen su movimiento para recoger la caja y posteriormente se dirigen hacia una estantería óptima.

  - La estrategia adicional que adoptan es que en cuanto un robot encuentra una caja y la recoge, le notifica al resto de robots que se encuentren deambulando

(es decir, que no tengan un rumbo definido) sobre su posición y estos se dirigen hacia este punto, agarrando cualquier caja que encuentren en su camino. De esta forma, los robots adoptan un comportamiento de grupo que se beneficia de los números para cubrir un mayor terreno y encontrar cuántas cajas sea posible.

2.3.1- Estrategia alternativa para disminuir tiempo de ejecución y movimientos requeridos.

- Una forma de disminuir el tiempo requerido para que todos los robots coloquen las cajas en pilas de 5, así como sus movimientos para realizar esta actividad, es implementando el **modo omnisciente**. Este modo incrementa drásticamente la eficiencia de los robots, ya que las rutas que siguen están hechas por medio del algoritmo A* y en ningún momento divagan en su camino. Si se ven obstaculizados y no encuentran un camino, permanecen en su lugar hasta que sus compañeros les despejen el paso.

- De esta manera no existe incertidumbre ni movimientos adicionales, a diferencia del **modo no omnisciente**, en el que los robots pueden repetir movimientos o moverse sin un rumbo concreto.

- Sin embargo, aún es posible realizar optimizaciones. Por ejemplo, el algoritmo de pathfinding empleado es A*, pero el fin de este algoritmo es encontrar un camino válido rápidamente, no necesariamente el más corto. Por lo tanto, para reducir los movimientos realizados, podría implementarse un algoritmo que encuentre el camino más corto posible, como Dijkstra.

- Así mismo, para reducir el tiempo de ejecución es posible realizar diversas optimizaciones en el comportamiento de los robots, ya que actualmente cada robot realiza únicamente una acción por cada step, ya sea moverse, recoger caja, intercambiar instrucciones o trazar un camino hacia un objetivo. Es posible hacer que un agente realice múltiples acciones en un mismo step, aunque con un costo computacional adicional. Un robot podría trazar un camino y dar el primer paso en el mismo step, por ejemplo; o intercambiar instrucciones y comenzar a seguirlas inmediatamente.

2.4.- Participación de integrantes del equipo

Backend	
Actividad	***Encargado***
Programación de clases de agentes	Omar Jiménez Armendáriz
Programación de clases de modelo	Omar Jiménez Armendáriz
Codificación de archivo backend.py	Omar Jiménez Armendáriz

Frontend	
Actividad	***Encargado***
Creación de espacio 3D, renderización y cámaras	Francisco Rocha Juárez
Importación de modelos 3D	Francisco Rocha Juárez
Conexión entre Backend y Frontend	Francisco Rocha Juárez
Diseño html y css de página de simulador index	Alejandro Alfonso Ubeto Yañez

Documentación	
Actividad	***Encargado***
Diagramas de clases y protocolo de interacción	Omar Jiménez Armendáriz
Introducción	Alejandro Alfonso Ubeto Yañez
Estrategia cooperativa	Alejandro Alfonso Ubeto Yañez
Guía de instalación y ejemplos	Alejandro Alfonso Ubeto Yañez
Resultados	Alejandro Alfonso Ubeto Yañez

3.- Resolución de la problemática

3.1.- Guía de instalación y ejecución

1. Instalar virtualenv a través de pip:

```
$ pip install virtualenv
```

2. Clonar el repositorio del proyecto:

```
$ git clone
https://aleubeto@bitbucket.org/aleubeto/modelacion-de-sistemas-multiag
entes-con-graficas.git
```

3. Crear un entorno virtual con tu versión actual de python usando VirtualEnv:

```
$ cd sistemasMultiagentes_graficasComputacionales
$ virtualenv venv --python=python3.8.10
```

4. Activar el entorno virtual e instalar dependencias de requirements.txt:

```
$ source venv/bin/activate
$ pip install -r requirements.txt
```

5. Ejecutar en una terminal el backend:

```
$ cd Integradora/Backend
$ python3 backend.py
```

6. Instalar dependencias de node js usando npm:

```
$ npm install
```

7. Ejecutar en una terminal diferente el frontend usando npm:

```
$ cd Integradora/Frontend
$ npm run dev
```

3.2.- Resultados

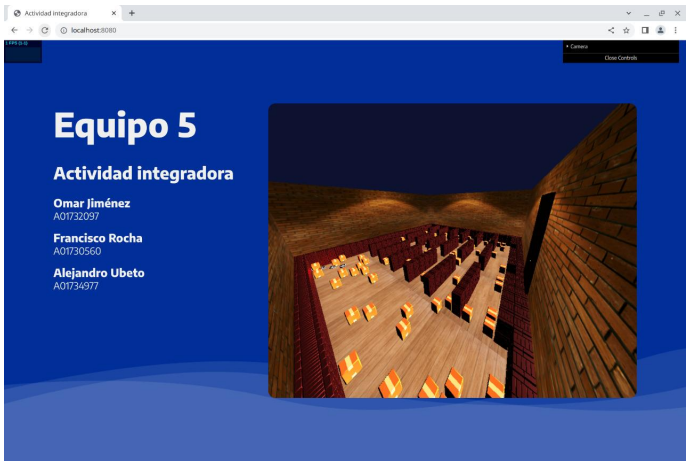

Imagen 3.- Ejecución inicial del modelado 3D

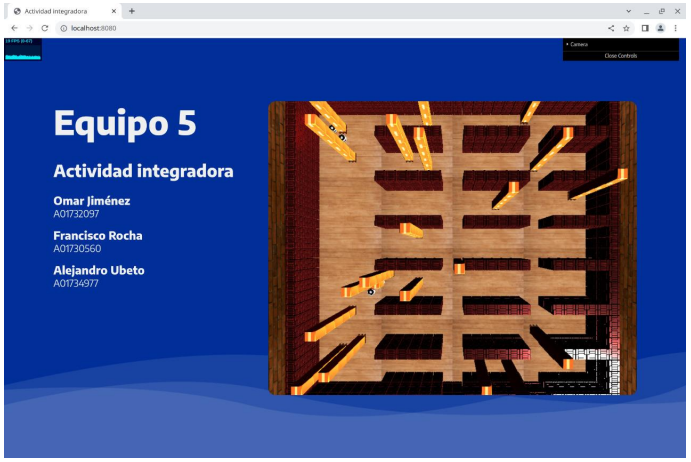

Imagen 4.- Pilas de cajas organizadas tras fin de ejecución

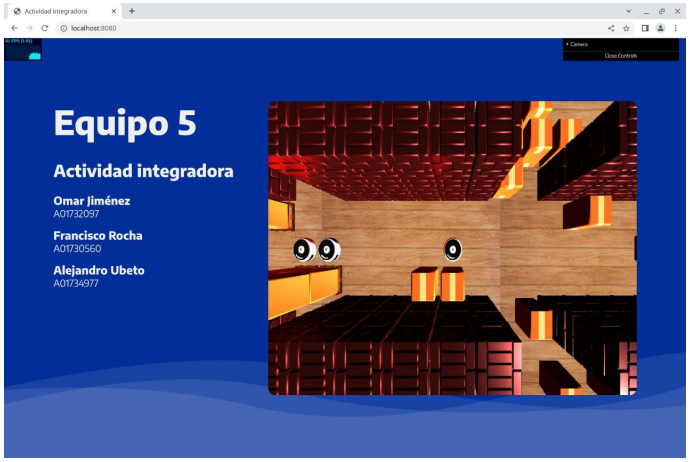

Imagen 5.- Cámara siguiendo a agente Robot (tecla 3)

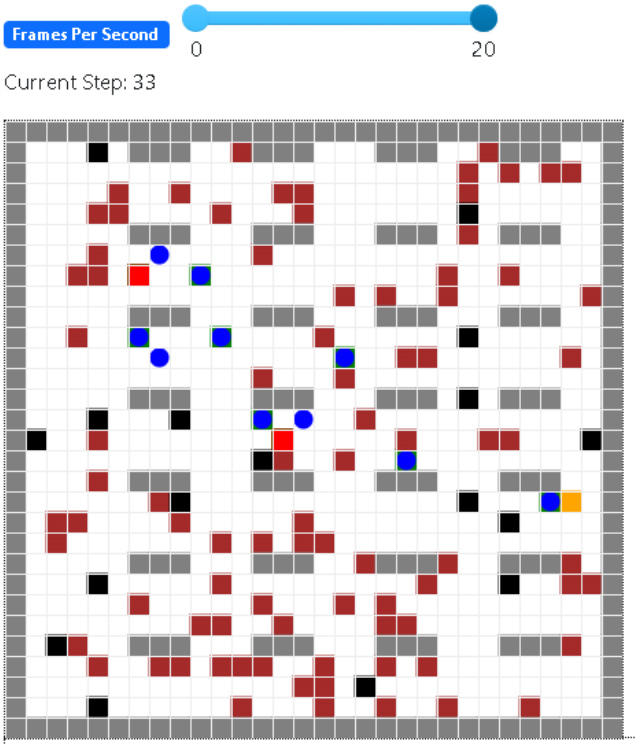

Imagen 6.- Modelo ejecutado en servidor de Mesa

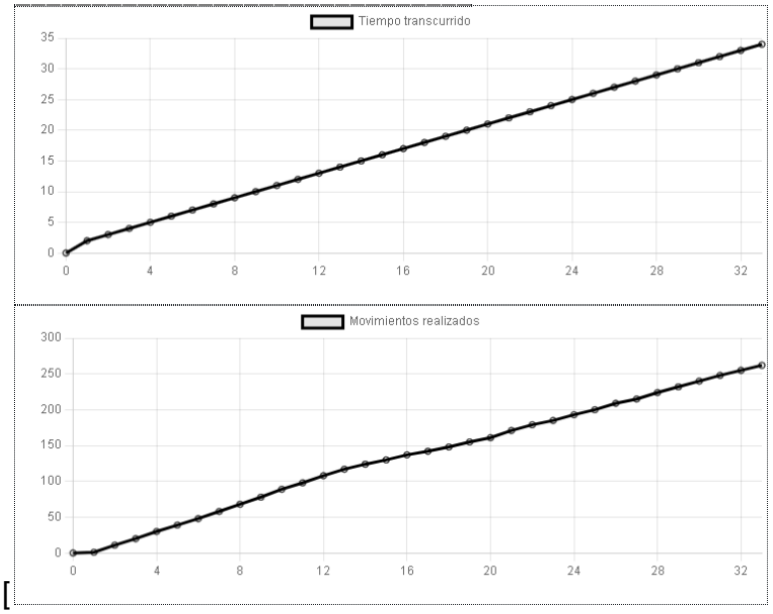

Imagen 7.- Gráficos obtenidos en el servidor de Mesa