Tecnológico de Monterrey

Actividad integradora - Sistemas multiagentes con GC

Luis Fernando Valderrábano García | **A01644530**

José Pedro Gastélum Beltrán | **A00227608**

Mario Feng Wu | **A01644768**

Octavio Sebastián Hernández Galindo | **A01638993**

Angel Gabriel Camacho Pérez | **A01743075**

Jonathan Roman Velasco | **A01644522**

TC2008B.302

Fecha de entrega:
24 de noviembre del 2025

Tecnológico de Monterrey

1. Diagramas de clases

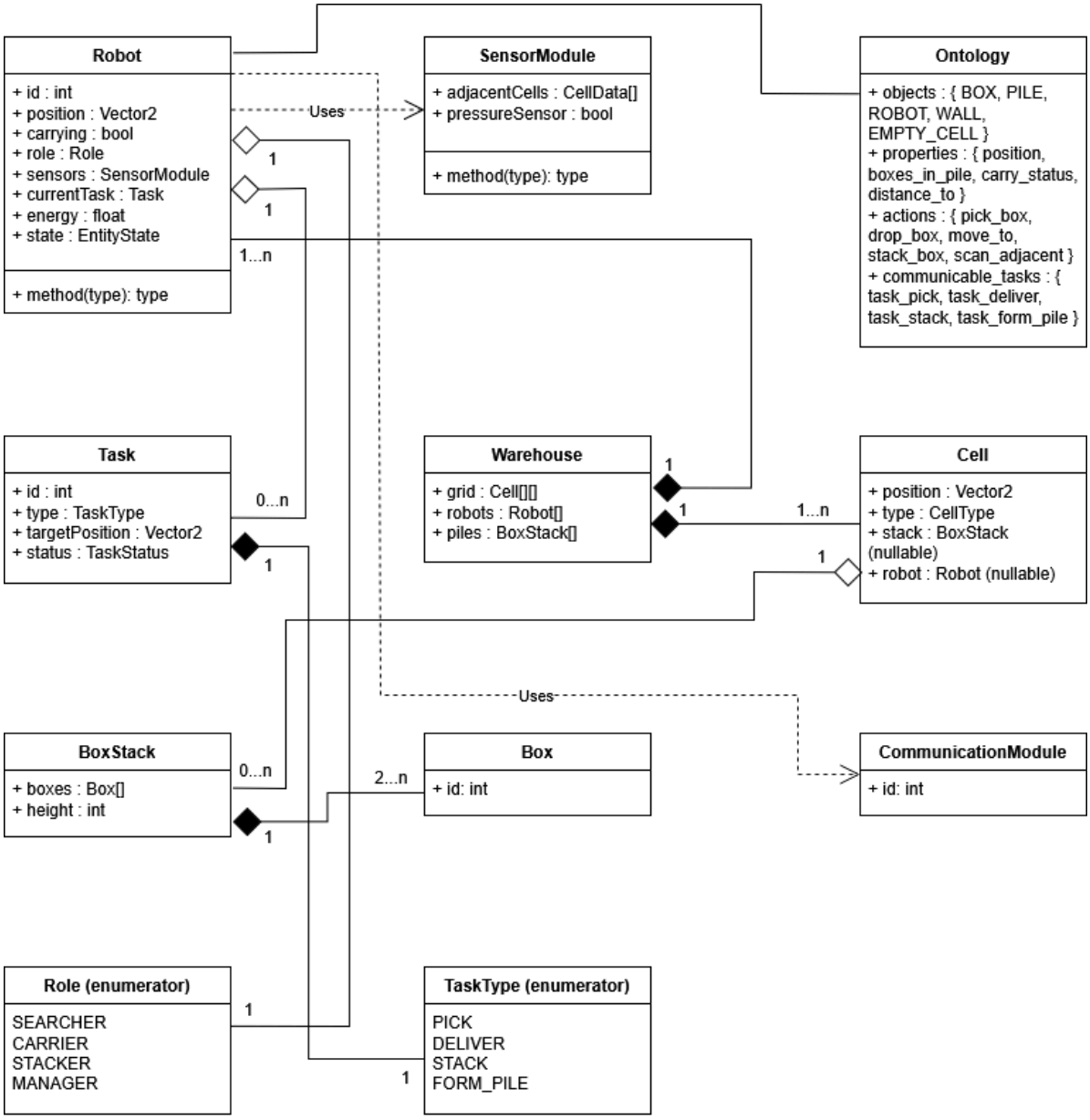

Figura 1. Diagrama UML. Creado en Draw.io [1]

La figura 1 muestra el diagrama de clases con la estructura principal del sistema de multiagentes dentro de un almacén automatizado. Incluye entidades como *Robot*, *Task*, *Warehouse*, *Cell*, *Box* y *BoxStack*, junto con módulos asociados como *SensorModule*, *CommunicationModule* y *Ontology*. El diagrama representa las relaciones entre los robots, las celdas del almacén, las pilas de cajas y las tareas asignadas, además de los roles y tipos de tareas posibles en el sistema. En conjunto, modela cómo los robots perciben su entorno, ejecutan tareas de manejo de cajas y se coordinan dentro del entorno del almacén.

Figura 2. Diagrama AUML parte 1. Creado en Draw.io [1]

Tecnológico de Monterrey

Figura 2 el diagrama muestra la estructura interna del agente Robot según AUML. El robot posee roles dinámicos (Searcher, Carrier, Stacker, Manager), que determinan su función según el contexto. Sus creencias (Beliefs) almacenan lo que percibe del entorno, como mapa local, cajas detectadas y pilas. Sus deseos (Desires) representan los objetivos que intenta cumplir: encontrar, mover y apilar cajas, y completar tareas. Las capacidades (Capabilities) describen lo que es capaz de hacer: moverse, percibir, manipular cajas y comunicarse. Finalmente, las acciones (Actions) son las operaciones concretas que ejecuta, como moverse, recoger o soltar cajas, apilarlas y enviar/recibir mensajes.

Figura 3. Diagrama AUML parte 2. Creado en Draw.io [1]

La figura 3 muestra una lista de protocolos que el agente Robot puede utilizar en el sistema. Cada protocolo corresponde a un tipo de interacción o mensaje entre robots: asignación de tareas, reporte de descubrimientos, ejecución de tareas, notificación de pilas completas, evitación de colisiones, actualización de estado, cambio de rol y verificación periódica (heartbeat). Es simplemente un listado de los nombres de estos protocolos dentro del modelo AUML.

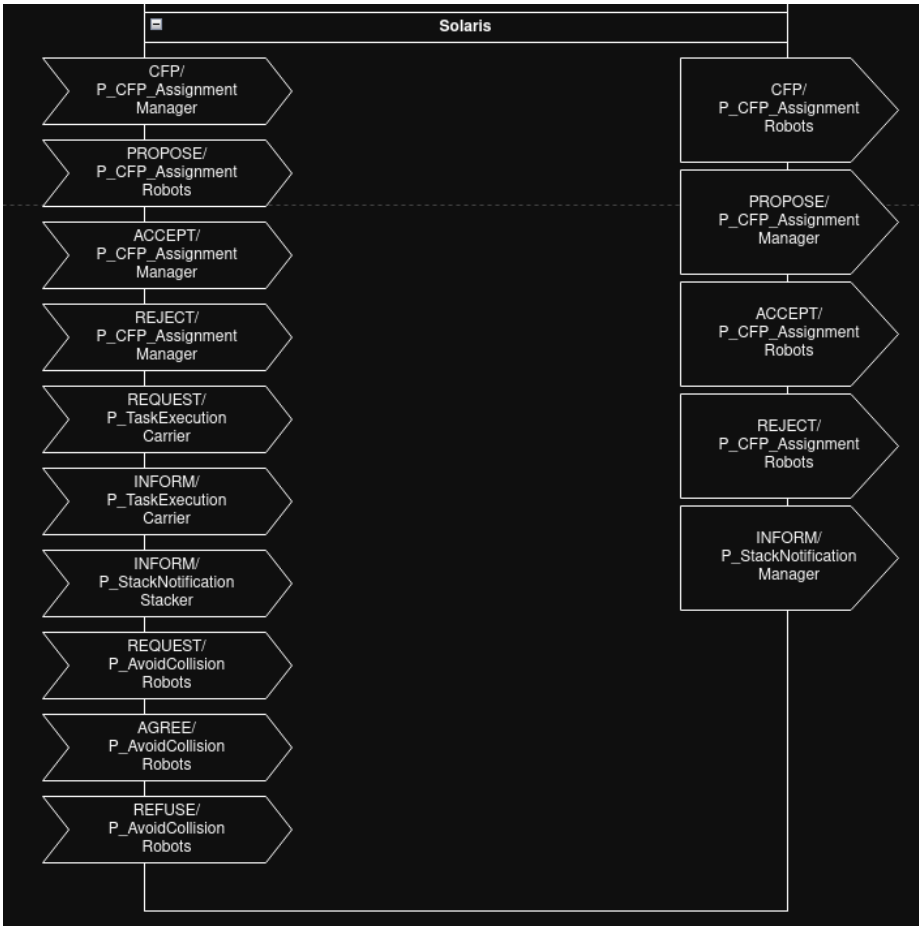

Figura 4. Diagrama AUML parte 2. Creado en Draw.io [1]

La figura 4 muestra un diagrama AUML de interacción entre robots en el sistema. A la izquierda aparecen los mensajes enviados por el Manager, Carrier, Stacker y otros robots, y a la derecha los mensajes que los otros agentes responden. Los mensajes están organizados por protocolos (como P_CFP_Assignment, P_TaskExecution, P_AvoidCollision y P_StackNotification) y por performatives (CFP, PROPOSE, ACCEPT, REJECT, REQUEST, INFORM, AGREE, REFUSE). En conjunto, el diagrama ilustra el flujo de comunicación típico entre los robots según su rol dentro del sistema.

2. Protocolo de agentes

2.1. Vocabulario

Para que todos los robots “hablen el mismo idioma”, se define la siguiente ontología:

BOX, PILE, ROBOT, WALL, EMPTY_CELL, position(x,y), boxes_in_pile, carry_status (true/false), distance_to(obj), pick_box, drop_box, move_to, stack_box, scan_adjacent, task_pick(position), task_deliver(position), task_stack(position), task_form_pile(position)

2.2. Performatives (Actos Comunicativos)

Se utilizarán los siguientes actos comunicativos para el intercambio de mensajes entre agentes.

- CFP — Call For Proposals
- PROPOSE — propuesta con un costo
- ACCEPT — aceptar propuesta
- REJECT — rechazar propuesta
- REQUEST — pedir acción directa
- INFORM — informar datos o estado

2.3. Protocolo de interacción

Participantes

- Manager: agente coordinador encargado de detectar tareas y asignarlas.
- R1, R2, R3, R4, R5: robots candidatos a ejecutar la tarea.

Objetivo del Protocolo

Determinar qué robot ejecutará la tarea de:

1. Moverse hacia la caja detectada.
2. Recogerla.
3. Transportarla a la pila asignada.

4. Colocarla en la pila.

Descripción del Protocolo

1. Detección de una caja sin asignar

El Manager identifica una caja disponible para ser procesada y decide iniciar el proceso de asignación.

2. Emisión de CFP (Call for Proposals)

El Manager envía un mensaje CFP(task_pick pos=(x,y)) a todos los robots (R1–R5), indicando que existe una tarea de recolección disponible en la posición (x, y).
Cada robot recibe el CFP.

3. Generación de Propuestas (PROPOSE)

Cada robot calcula su costo para ejecutar la tarea considerando principalmente la distancia a la caja:

- R1 → PROPOSE(cost = d1)
- R2 → PROPOSE(cost = d2)
- R3 → PROPOSE(cost = d3)
- R4 → PROPOSE(cost = d4)
- R5 → PROPOSE(cost = d5)

Los costos son enviados al Manager usando el protocolo PROPOSE del Contract Net.

4. Selección de la Mejor Oferta

El Manager evalúa todas las propuestas recibidas y selecciona la de menor costo.
Este robot será quien ejecute la tarea.

5. Envío de ACCEPT y REJECT

- El robot con el costo más bajo recibe:
  ACCEPT(task_id = N)
- Los demás robots reciben:
  REJECT(task_id = N)
  confirmando que no fueron seleccionados.

6. Ejecución de la Tarea por el Robot Ganador

El robot asignado realiza la tarea en cuatro pasos:

1. move_to(box_position) — se desplaza hacia la caja.
2. pick_box() — recoge la caja.
3. move_to(target_pile) — lleva la caja hasta la pila asignada.
4. drop_box() — coloca la caja en la pila.
5. Informe de Finalización

Una vez completada la tarea, el robot ganador envía al Manager:

INFORM(status = complete)

Lo cual indica que la caja ya fue recolocada en la pila correspondiente.

6. Resultado del Protocolo

- La caja es movida exitosamente a su pila.
- Solo un robot realiza la tarea.
- Los demás robots quedan libres para futuras asignaciones.

2.4. Diagrama de protocolo de interacción

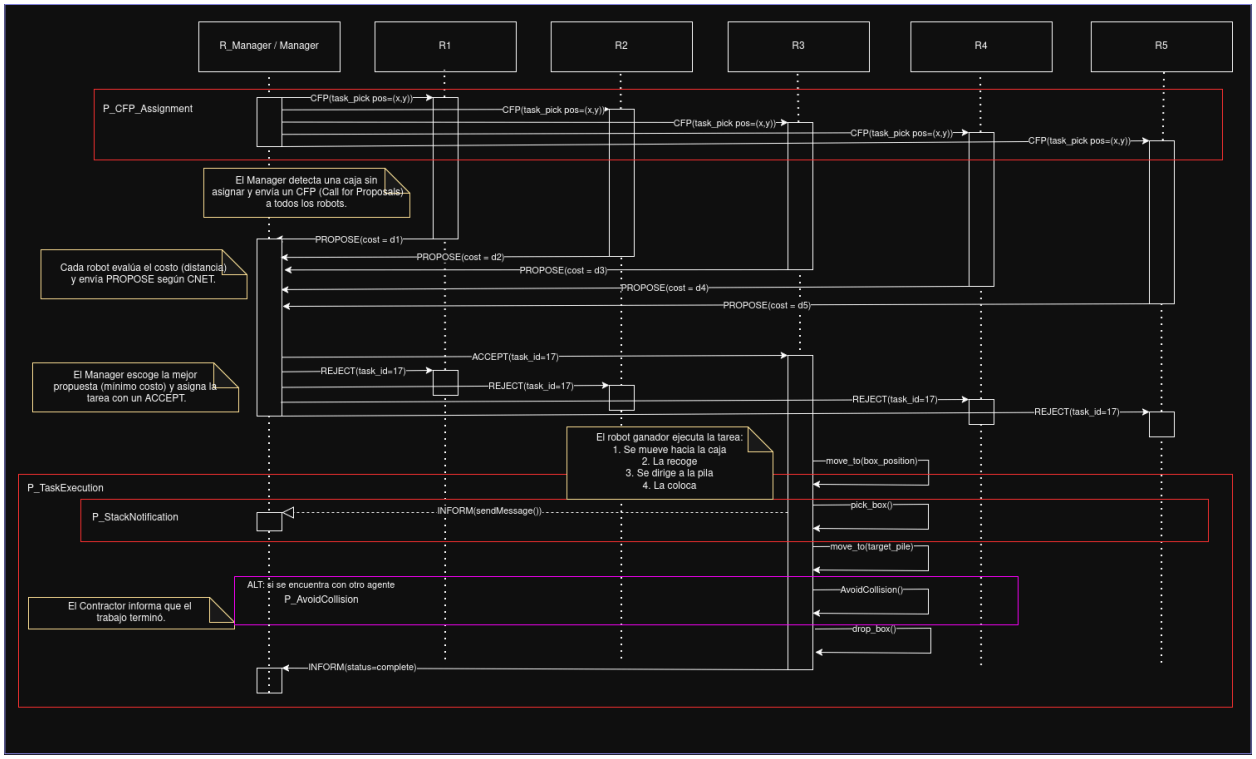

Figura 5. Diagrama AIP. Creado en Draw.io [1]

La Figura 4 muestra un protocolo de interacción entre agentes basado en Contract Net, donde un agente en rol de Manager envía una convocatoria (CFP) a los demás robots, estos responden con propuestas (PROPOSE), y posteriormente el Manager selecciona al mejor candidato mediante mensajes ACCEPT y REJECT. También se representan las acciones que realiza el robot asignado para completar la tarea y los mensajes INFORM que notifican el resultado. En conjunto, el diagrama ilustra la secuencia de comunicación y coordinación necesaria para asignar y ejecutar tareas dentro del sistema multiagente. Cabe recalcar que en el diagrama se encierran partes del proceso para poder visualizar los protocolos, por ejemplo, TaskExecution, CFP_Assignament, etc…

3. Estrategia cooperativa

Interacción 1 “Evitar colisión en pasillos”

Agentes: Robot A y Robot B(cualquier rol: searcher, carrier o stacker)
Acciones:

- C: Ceder / desviarse: Ejecutar Protocolo P_AvoidCollision
- D: Seguir recto / no ceder: Mantener velocidad y trayectoria

Patrones de juego identificado: Juego del Gallina
Justificación: Si ambos no ceden (D,D), ocurre un choque físico o bloqueo lógico, lo cual sería el peor resultado posible ya que se puede dañar el robot y perder mucho tiempo. Si uno cede y el otro pasa (C,D), el sistema fluye aunque el que cede sacrifica un poco de tiempo.

Matriz de Pagos:
Ponedarion: Tiempo

	Robot B: C	Robot B: D
Robot A: C	-1,-1	-2,5
Robot A: D	5,-2	-10, -10

Interacción 2 “Asignación de cajas"

Agentes: Carriers
Acciones:

- C (cooperar) : Enviar un PROPOSE con la distancia real hacia la caja
- D (No cooperar): Enviar un PROPOSE con una distancia falsa para “robar” la caja

Patrones de juego identificado: Dilema del prisionero
Justificación: A los robots les conviene recoger la caja (D) para aumentar su productividad individual, sin embargo, si todos los robots mienten sobre sus distancias para intentar tomar más cajas, la eficiencia global del almacén colapsaría ya que robots que están lejos, consumirían tiempo y energía de más. La cooperación de todos(C,C) maximiza la utilidad del almacén y el grupo a largo plazo.

Matriz de Pagos:
Eficiencia Energética

	Robot B: C	Robot B: D
Robot A: C	5,5	0,7
Robot A: D	7,0	-5,-5

4. Relaciòn a còdigo

Problema Almacén	**Scripts Unity Actuales**
Robot	Bot.cs
Caja (Box)	Plant.cs
Celda (Cell)	Cell.cs
Almacén (Warehouse)	Map.cs
Zona Segura	SafeZone.cs
Pilas de cajas	BoxStack (esta funcionalidad dentro de Bot.cs)

5. Texturas y FBX

Para la descarga de nuestras texturas y FBX implementados en el escenario de Unity se descargaron de AssetStore [2], la cual es un paquete ya integrado con varios elementos con textura, como por ejemplo; tomates, cajas, etc. Al igual que se agregó un FBX de un robot simulando los agentes [3].

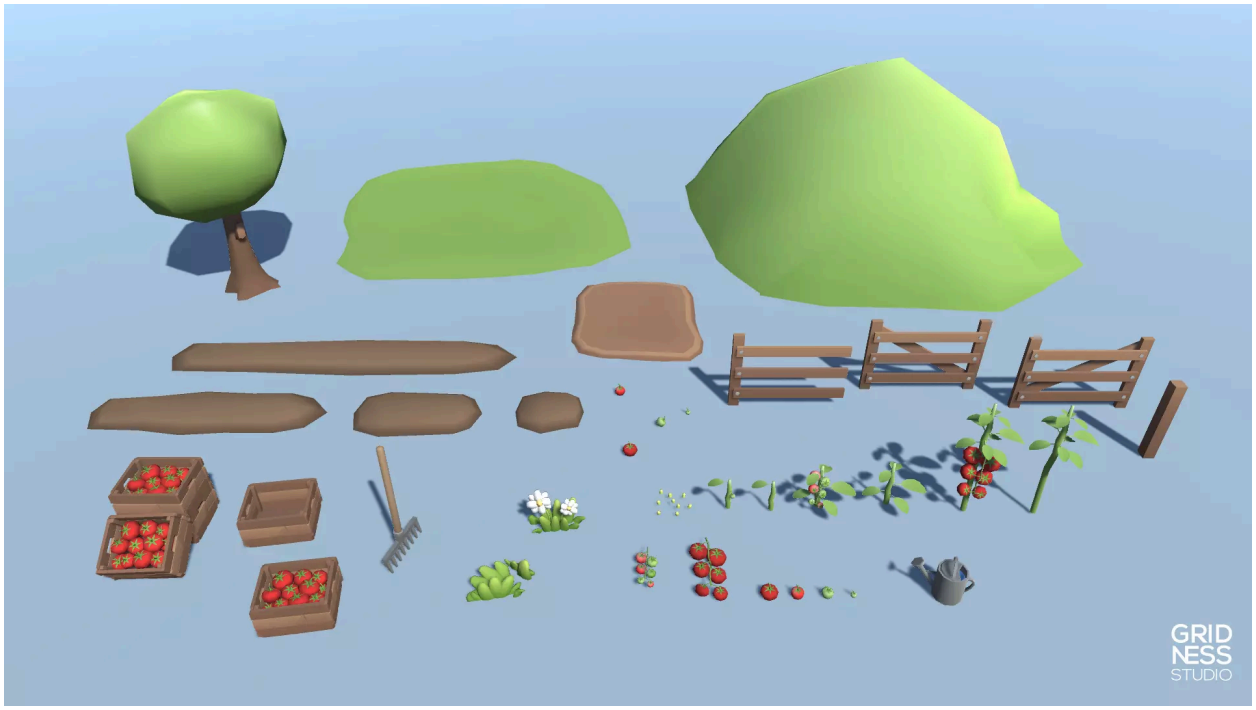

Figura 6. Modelo tomates. Creado en AssestStore [4]

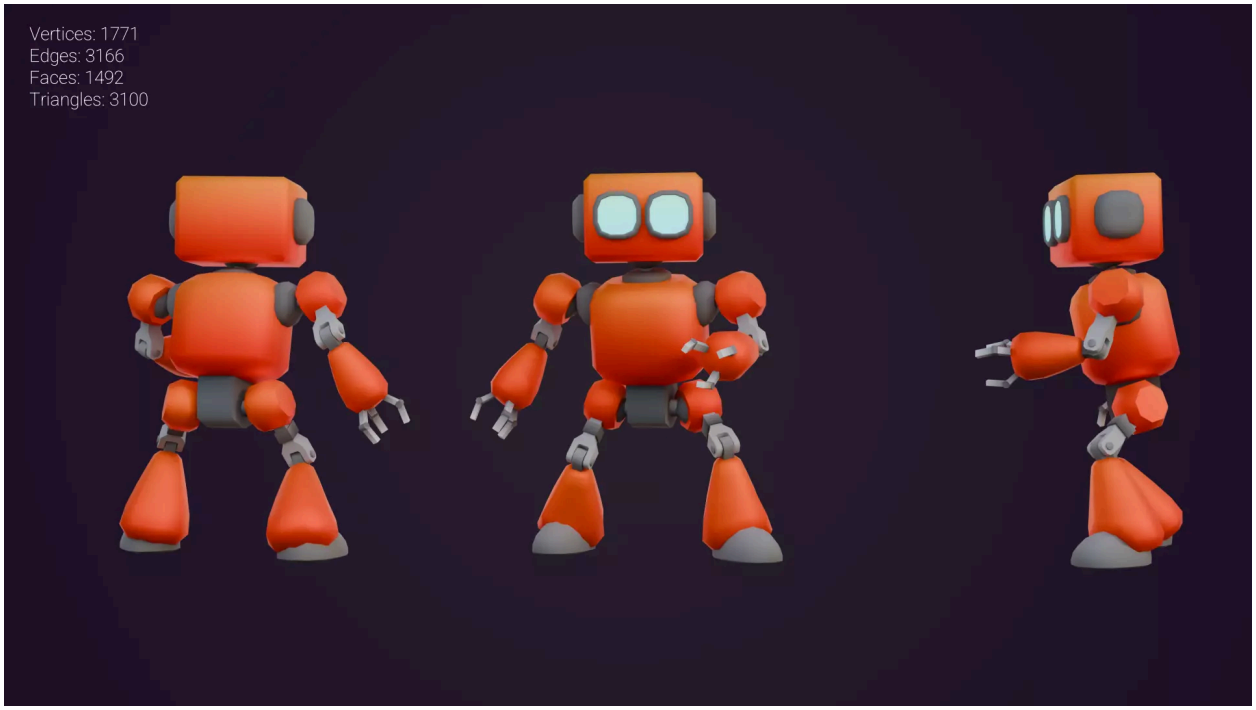

Figura 7. Modelo tomates. Creado en AssestStore [4]

6. Referencias

[1] draw.io. diagrams.net. Aplicación de diagramación en línea. https://www.drawio.com/
[2] Unity Technologies. *Lite Farm Pack – Low-Poly 3D Art by Gridness*. Paquete 3D de entorno industrial. Unity Asset Store. https://assetstore.unity.com/packages/3d/environments/industrial/lite-farm-pack-low-poly-3d-art-by-gridness-243315.
[3] Unity Technologies. Animated Low-Poly Robot (Free Stylized Sci-Fi Character). Paquete 3D de personajes. Unity Asset Store. https://assetstore.unity.com/packages/3d/characters/robots/animated-low-poly-robot-free-stylized-sci-fi-character-318201
[4] Unity Technologies. *Unity Asset Store*. Sitio web. https://assetstore.unity.com/