



Instituto Tecnológico de Estudios Superiores de Monterrey

Modelación de sistemas multiagentes con gráficas computacionales

Grupo 101

Equipo Druids

Arturo Ramos Martínez A01643269

Adolfo Hernández Signoret A01637184

Bryan Ithan Landín Lara A01636271

Diego Enrique Vargas Ramírez A01635782

Luis Fernando Cuevas Arroyo A01637254

Campus Guadalajara

Lunes 2 de septiembre de 2024

Conformación del equipo de trabajo (Fortalezas y debilidades) :

Arturo Ramos Martínez:

Fortalezas:

- **Conocimiento Técnico:** Experiencia en programación, con lenguajes como python o c++
- **Resolución de Problemas:** Habilidad para encontrar soluciones a problemas complejos.

Áreas de oportunidad:

- **Gestión del Tiempo:** Necesita mejorar en la planificación y distribución de tareas para evitar retrasos en la entrega de sus responsabilidades.
- **Delegación:** Asumir demasiadas responsabilidades, lo que puede llevar a una sobrecarga de trabajo.

Adolfo Hernández Signoret:

Fortalezas:

- **Desarrollo del Endpoint:** Sabe utilizar fast API.
Control del estado de las variables compartidas entre unity y python

Áreas de oportunidad:

- **Tiempo de dedicación.** Podría dedicar más tiempo para obtener los resultados no tan cercanos a la fecha de entrega.

Bryan Ithan Landín Lara:

Fortalezas:

- **Diseño de Interfaces de Usuario:** Sabe utilizar Unity para crear escenas realistas y que reflejan la simulación adecuadamente.

- **Creatividad:** Habilidad para agregar componentes que resuelven conflictos de una manera que no a todos parece obvia.
- **Colaboración:** Facilidad para trabajar en equipo, aportando ideas y ayudando a otros miembros a superar obstáculos.

Áreas de Oportunidad:

- **Conocimiento Técnico:** Podría beneficiarse de una mayor familiarización con los conceptos avanzados de programación de sistemas multiagente.
- **Priorización:** Necesita mejorar en la identificación de las tareas más críticas para enfocarse en ellas primero.

Diego Enrique Vargas Ramírez:

Fortalezas:

- **Analítica:** Habilidades para el análisis de datos y la interpretación de patrones, lo que es clave para mejorar la eficiencia del sistema de patrullaje.
- **Optimización de Algoritmos:** Capacidad para optimizar algoritmos, logrando una mejor performance del sistema en tiempo real. Implementando estrategias como el star path finding.

Áreas de Oportunidad:

- **Documentación:** Necesita mejorar en la documentación de su trabajo para que otros miembros del equipo puedan entender y seguir su progreso.
- **Multitasking:** Puede tener dificultades para manejar múltiples tareas simultáneamente, lo que podría afectar su rendimiento.

Luis Fernando Cuevas Arroyo:

Fortalezas:

- **Gestión de Proyectos:** Capacidad para planificar y gestionar proyectos, asegurando que el equipo se mantenga en línea con los objetivos y plazos acordados.
- **Comunicación efectiva:** Habilidad para comunicar ideas y estrategias de manera clara, tanto al equipo como a otros participantes como el profesor.
- **Negociación y Resolución de Conflictos:** Experiencia en manejar conflictos dentro del equipo y llegar a soluciones que beneficien a todos.

Áreas de Oportunidad:

- **Profundización Técnica:** Se puede mejorar los conocimientos técnicos para entender más a fondo los desafíos que enfrenta el equipo en términos de programación y desarrollo.
- **No ceder:** A veces cede demasiado aunque sabe que él está en lo correcto con tal de evitar un conflicto.

Listado de lo que esperamos lograr y obtener como equipo de trabajo en el presente bloque, así como nuestros compromisos para lograrlo:

- Entender y aplicar conocimientos sobre los agentes, así como de los sistemas multiagentes.
- Lograr comunicar las distintas tecnologías, en este caso la comunicación entre python y unity con C#, por medio de una api con Fast API para manejar las peticiones de unity.
- Mejorar nuestra planeación de proyectos sobre agentes, aprendiendo a diseñar los distintos diagramas (poner los tipos de diagramas).
- Mejorar nuestra organización como equipo.

- Presentar una simulación que cumpla con lo que se nos pide. sobre todo apegándose lo más posible a la programación orientada a los agentes.
- Y sobre todo aprender todo lo que podamos en el proceso.

Compromisos:

- Comunicarnos lo mejor posible, es importante que cualquier duda o propuesta sea expresada, para estar claros con nuestra propuesta para el reto, fechas de entrega y avances. Esto por medio del grupo de whatsapp y en las reuniones de equipo.
- Reunirnos todos los días por lo menos 3 horas hasta la fecha de entrega. Además de cada quien trabajar por su cuenta.
- Avisar al equipo cuando se haga alguna actualización en el código (cuando se sube algún cambio al repositorio de github)
- Comprometernos y responsabilizarnos con las partes que a cada integrante del equipo le toca desarrollar.
- Crear un ambiente ideal para el trabajo colaborativo, dejando de lado actitudes que puedan afectar la comunicación.

Herramientas de trabajo colaborativo:

- El repositorio de github donde guardaremos la información es el siguiente:
https://github.com/ArturoRM22/drone_agents
- Para la comunicación entre los participantes optamos por un grupo de WhatsApp para una comunicación más clara y directa.

Propuesta formal del reto:

- **Descripción.**

El proyecto consiste en diseñar e implementar un sistema multiagentes que simule un patrullaje coordinado en una fábrica/mega-almacén. El objetivo principal es lograr una cobertura efectiva de vigilancia, por medio de un dron, cámaras y un guardia de seguridad. El flujo será el siguiente:

- **Agentes involucrados**

- **Dron:** Tendrá la tarea de seguir una ruta de patrullaje predefinida, con el objetivo de vigilar la zona; en caso de detectar algo sospechoso, el dron notificará/alertará al guardia, para que el delibere si se trata o no de un peligro o de alguna situación que no debería estar pasando.
- **Cámaras:** Tendrán un rol muy parecido al del dron, pero estas en lugar de comunicarse con el guardia al detectar que algo anda mal, por ejemplo si ven algún intruso, lo que harán será comunicarse con el dron, para que este se acerque a la zona y decida si comunicarse con el guardia o no.
- **Guardia:** Tendrá la capacidad de tomar control del dron cuando este se comunique con él, para hacer una inspección más detallada y decidir si encender o no la alarma del lugar.
- **Intruso/Empleado:** El funcionamiento de este agente es demostrar la comunicación de los agentes encargados de la vigilancia, agregamos un 4 agente, el cual actuará como intruso o empleado; esté estará rondando por el lugar, hasta que el dron, o alguna cámara lo detecte, en ese momento, tanto el guardia, el dron y las cámaras, establecerán una comunicación y decidirán qué hacer.

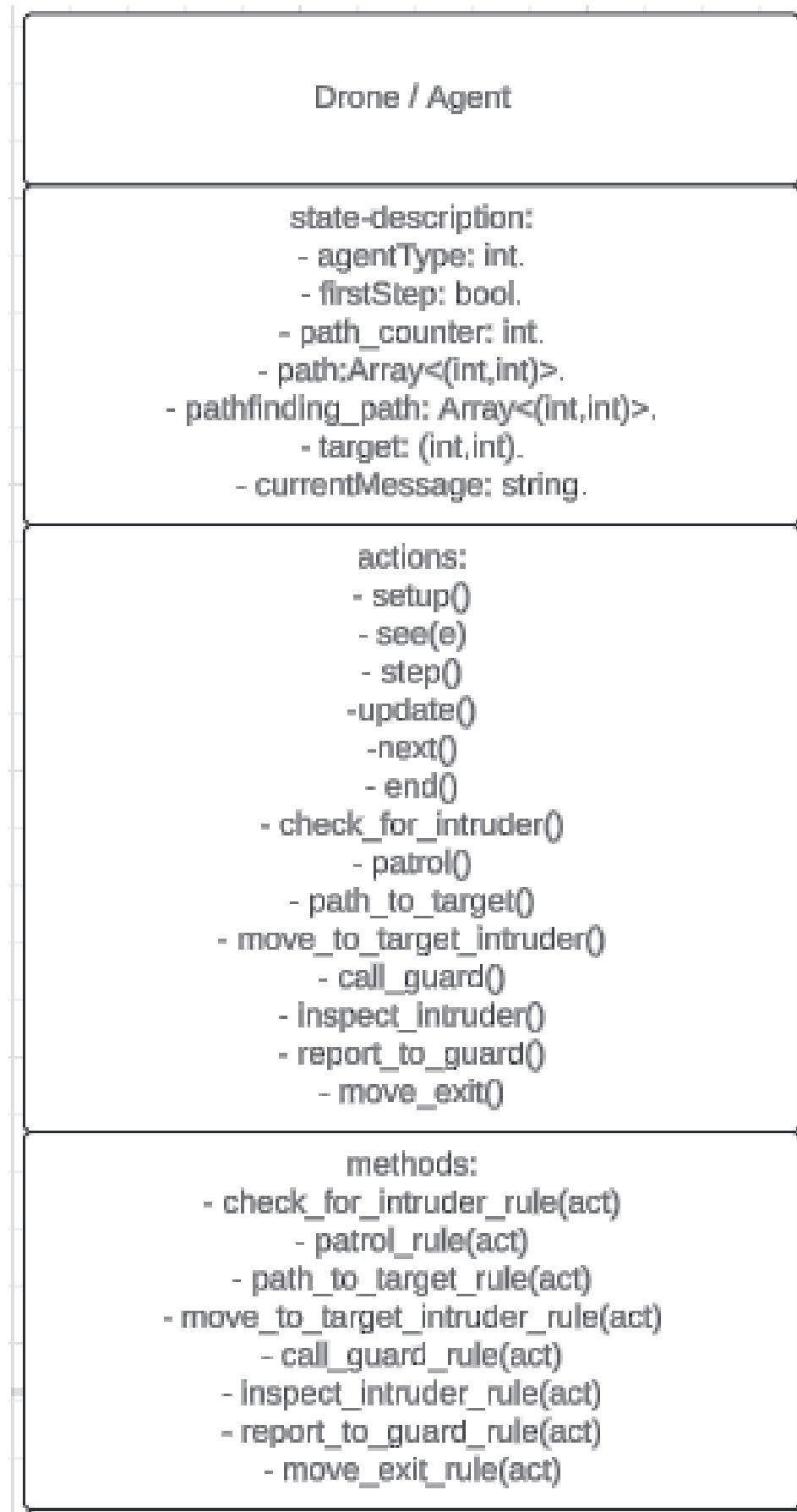
- **Propiedades de los agentes involucrados:**

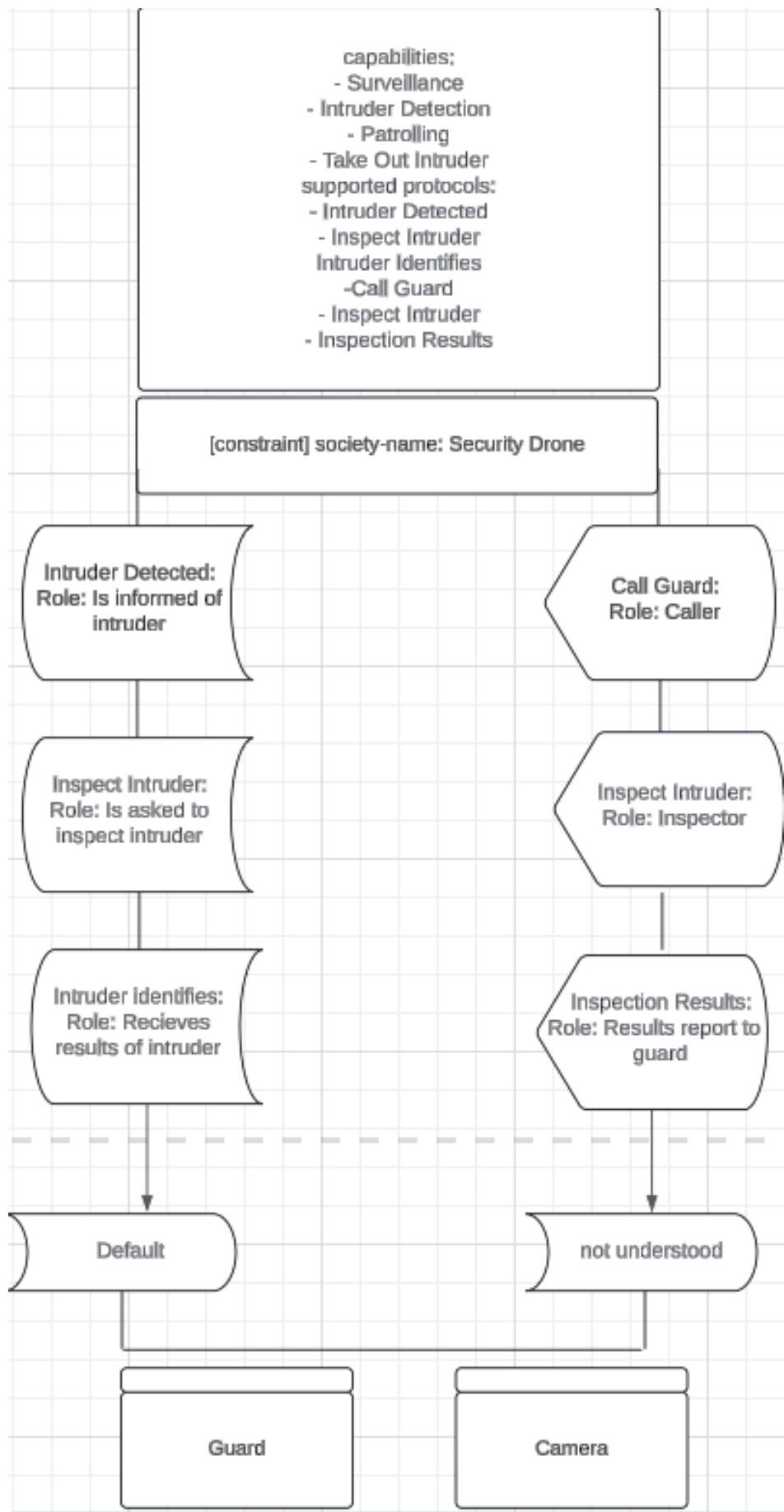
- **Dron:**

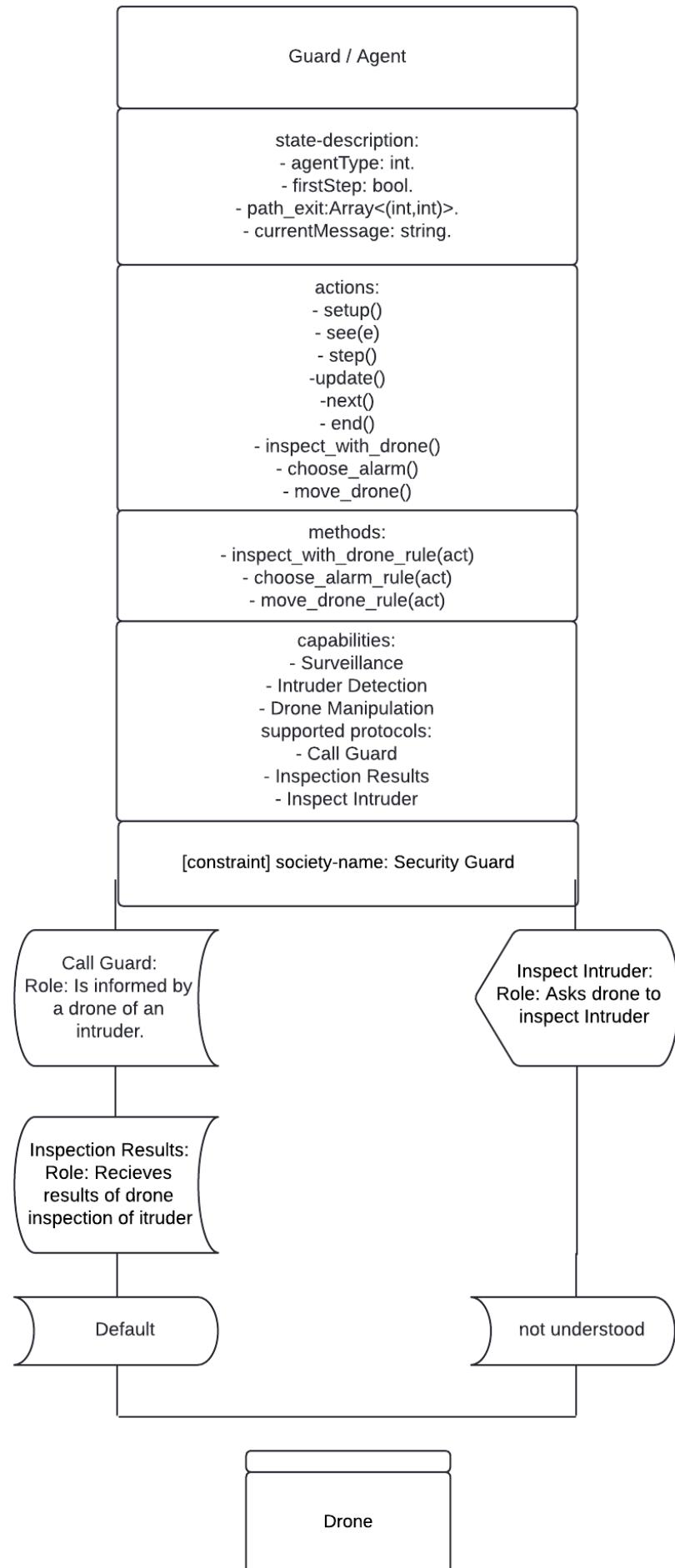
- **Autonomía:** El dron patrulla por sí solo e incluso puede iniciar la lógica de mensajes al detectar un intruso o ser alertado primero por las cámaras.
 - **Reactividad:** El dron reacciona a los eventos de vigilancia y posibles intrusos detectados por él mismo o por las cámaras.
 - **Proactividad:** Realiza patrullajes de manera autónoma siguiendo una ruta predefinida, anticipándose a posibles incidentes.
 - **Socialización:** Se comunica con las cámaras y el guardia, notificando sobre posibles amenazas y colaborando para tomar decisiones conjuntas.
- **Cámaras:**
- **Autonomía:** Las cámaras si bien actúan por sí cuenta al detectar intrusos, este es el límite de su autonomía. El resto de acciones dependen de detectar a un intruso.
 - **Reactividad:** Detectan la presencia de intrusos y responden de manera inmediata, alertando al dron para que verifique la situación.
 - **Proactividad:** Aunque las cámaras no patrullan, están activamente monitoreando la zona y pueden detectar actividades sospechosas de manera continua.
 - **Socialización:** Se comunican con el dron, solicitando que inspeccione áreas donde se detecta una posible intrusión.
- **Guardia:**
- **Autonomía:** El guardia y toda su lógica es dependiente de que se detecte a un intruso dentro del mapa, sino este no es capaz de actuar.
 - **Reactividad:** Responde a las notificaciones del dron o cámaras y decide si activar o no las alarmas en función de la información recibida.

- **Proactividad:** Aunque su papel es principalmente reactivo, puede tomar control del dron para investigar más a fondo cuando lo considere necesario.
- **Socialización:** Se comunica tanto con el dron como con las cámaras, recibiendo y procesando la información antes de tomar una decisión sobre las alarmas.
- **Intruso:**
 - **Autonomía:** El intruso tiene una serie de objetivos en el mapa a los cuales llegar, este se mueve a ellos de manera independiente.
 - **Reactividad:** No responde de manera directa a los eventos de vigilancia, pero su presencia desencadena las acciones de los otros agentes.
 - **Proactividad:** Se mueve de manera autónoma por el área vigilada, simulando el comportamiento de un intruso o empleado.
 - **Socialización:** No se comunica directamente con los otros agentes, pero su detección provoca que los demás agentes colaboren para gestionar la situación.

Diagrama de clase presentando los distintos agentes involucrados







Wall / Agent

state-description:

- agentType: int.

actions:

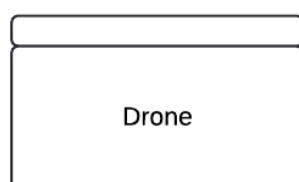
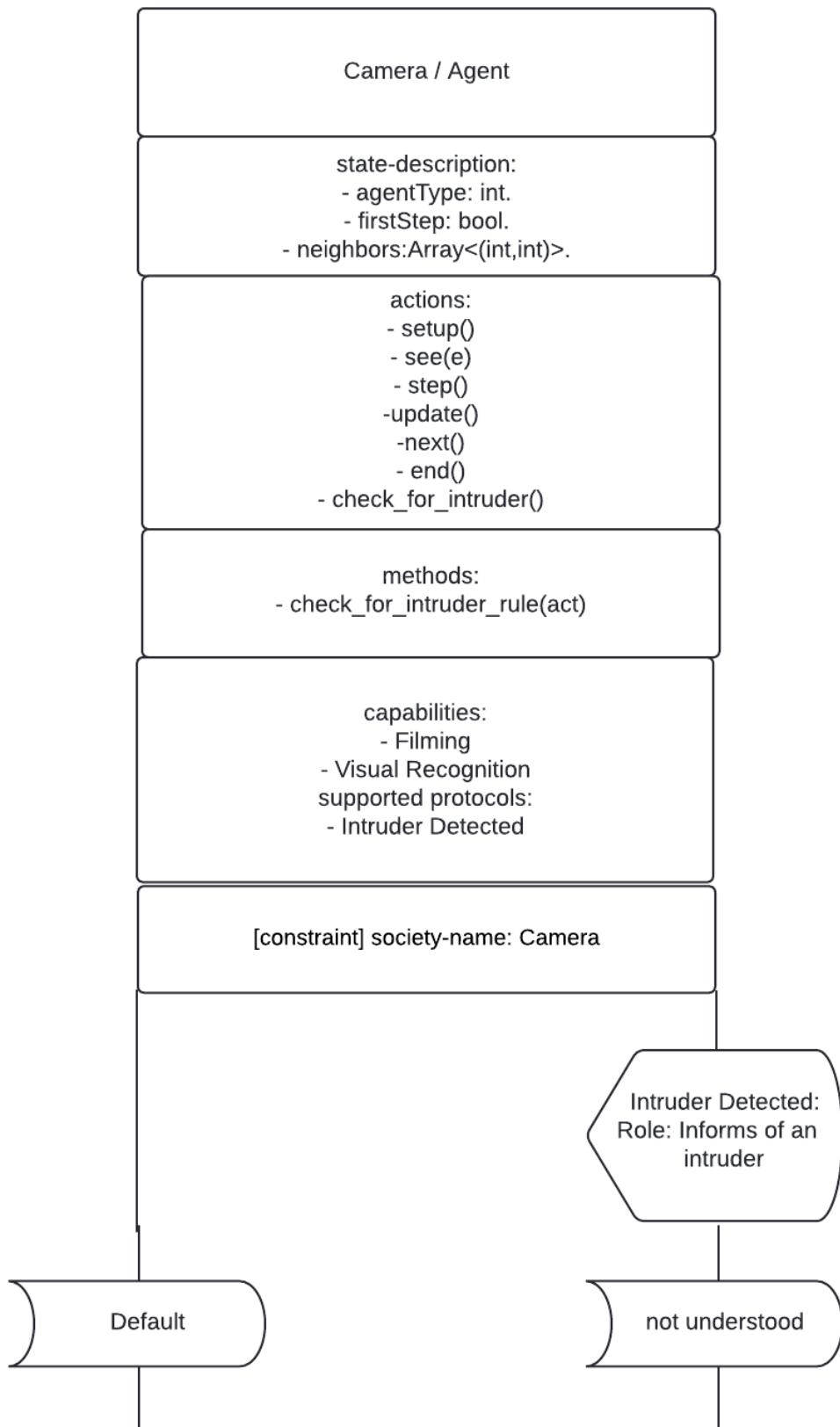
- setup()
- see(e)
- step()
- update()
- next()
- end()

methods: none

capabilities: none

supported protocols:none

[constraint] society-name: Wall



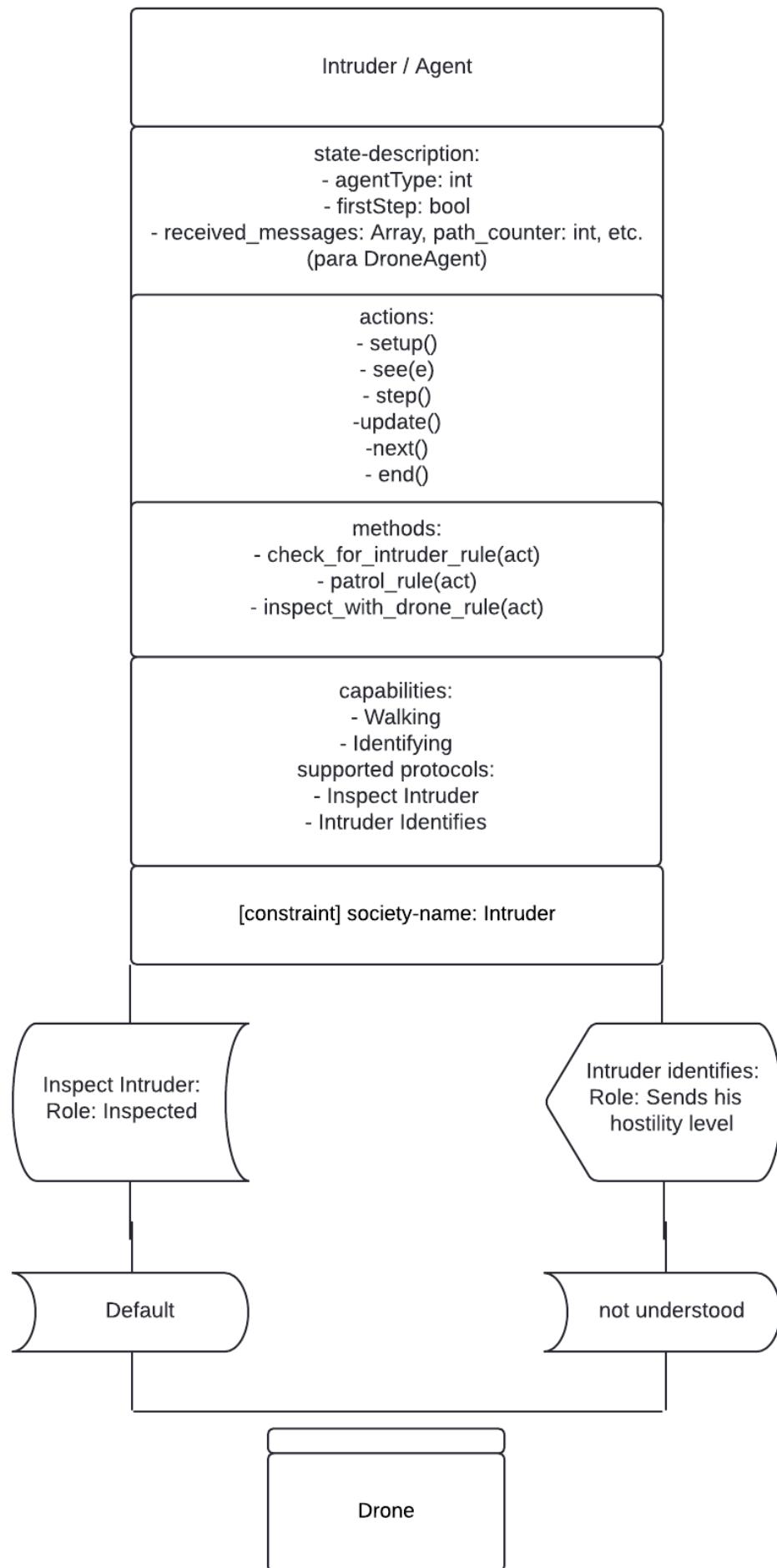
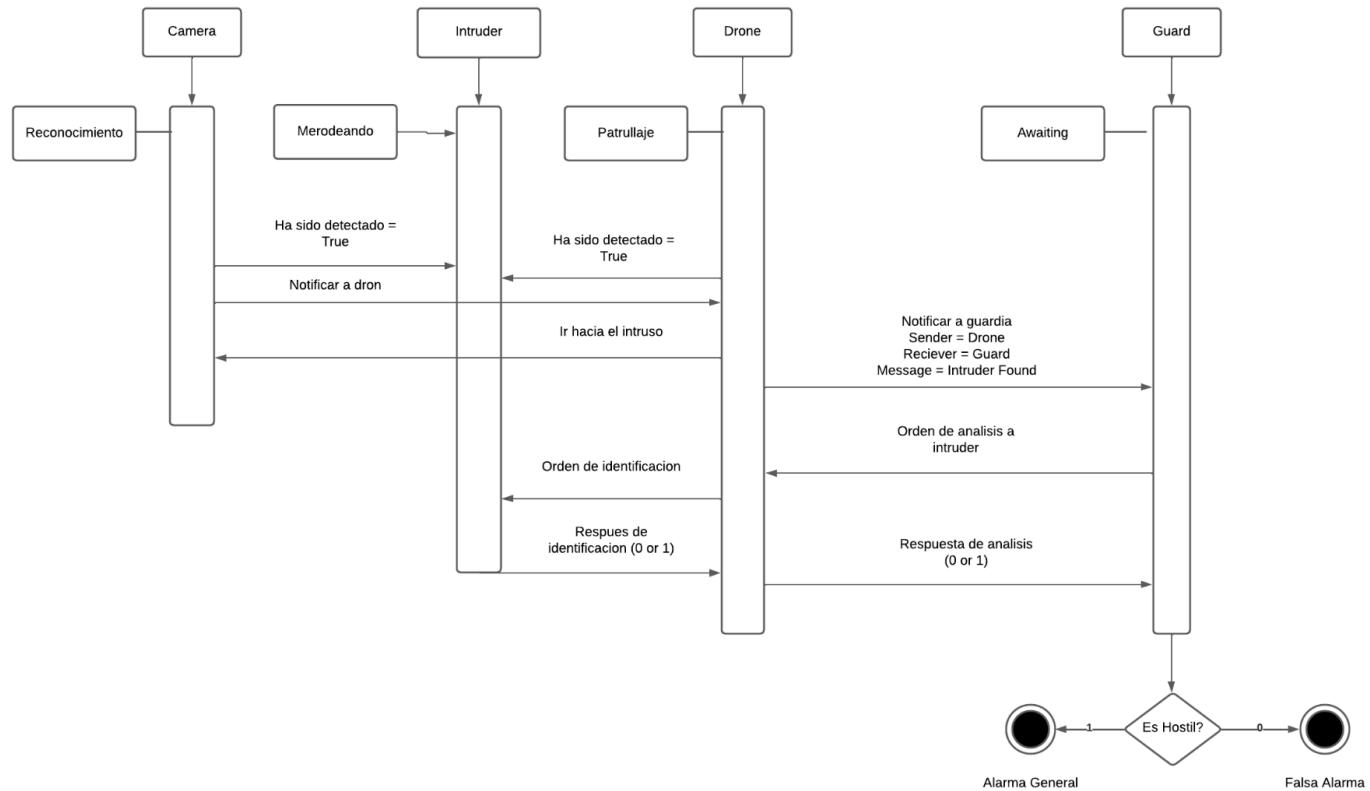
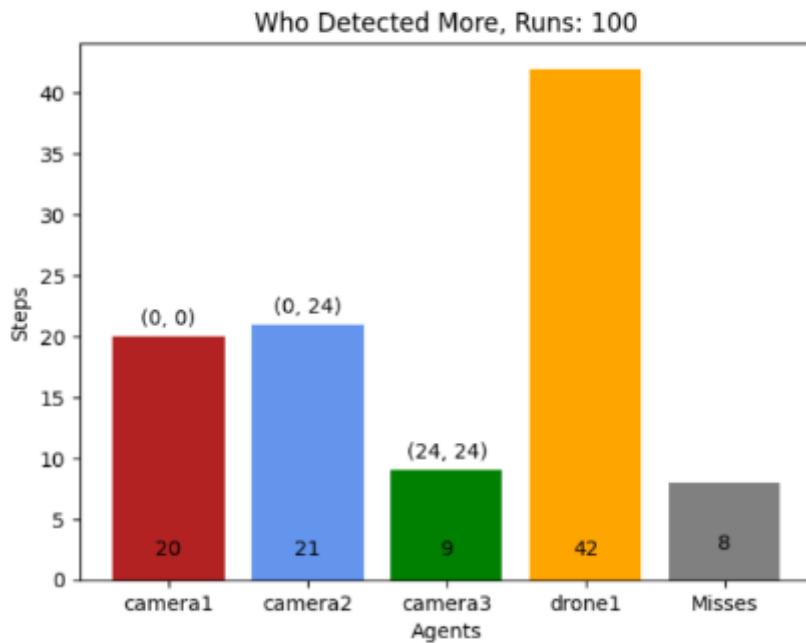


Diagrama de protocolos de interacción.



Métrica de utilidad o éxito

Para identificar si la simulación ha tenido éxito, se realiza la comparación de si al final de los pasos indicados, la posición del dron y la del intruso son la misma. Desde un punto de vista de cada agente, tiene mayor éxito el que detecta más intrusos. Desde el punto de vista de todos los agentes como comunidad, fracasan si ninguno detecta al intruso.



Plan de trabajo y aprendizaje adquirido:

1. Plan de Trabajo y Aprendizaje Adquiridos

Responsable: Luis Fernando Cuevas Arroyo

Fecha: 2 - 6 de septiembre de 2024

Tareas:

Elaborar un plan de trabajo detallado, incluyendo actividades pendientes, responsables, fechas y estimación de esfuerzo.

Resumir el aprendizaje adquirido por cada miembro del equipo durante el proyecto.

2. Desarrollo del código

Responsables:

Diego Enrique Vargas Ramírez: Implementación de los agentes.

Arturo Ramos Martínez: Implementación de los agentes.

Adolfo Hernández Signoret: Comunicación entre python y unity.

Fecha: 2 - 6 de septiembre de 2024

Tareas:

Implementar el código necesario en Python y Unity para cumplir con los objetivos.

Más específicamente, lograr una simulación acorde a los requerimientos, y apegada a la programación orientada a agentes. Además, lograr llevar esto a unity, por medio del protocolo HTTP y el manejo de entrada y salida de datos para el correcto funcionamiento y visualización de la simulación.

Subir el código al repositorio de GitHub.

2. Desarrollo del código para la visión computacional**Responsables:**

Bryan Ithan Landín Lara

Luis Fernando Cuevas Arroyo

Fecha: 2 - 6 de septiembre de 2024

Tareas:

Implementar los scripts necesarios en unity para poder llevar a cabo la visión computacional y lograr conectar el servidor, para que pueda analizar frame por frame en el proyecto, para que en base a eso, los agentes puedan interactuar y comunicarse entre ellos.

Subir el código al repositorio de GitHub.

4. Presentación Final**Responsables:**

Luis Fernando Cuevas Arroyo

Bryan Ithan Landín Lara

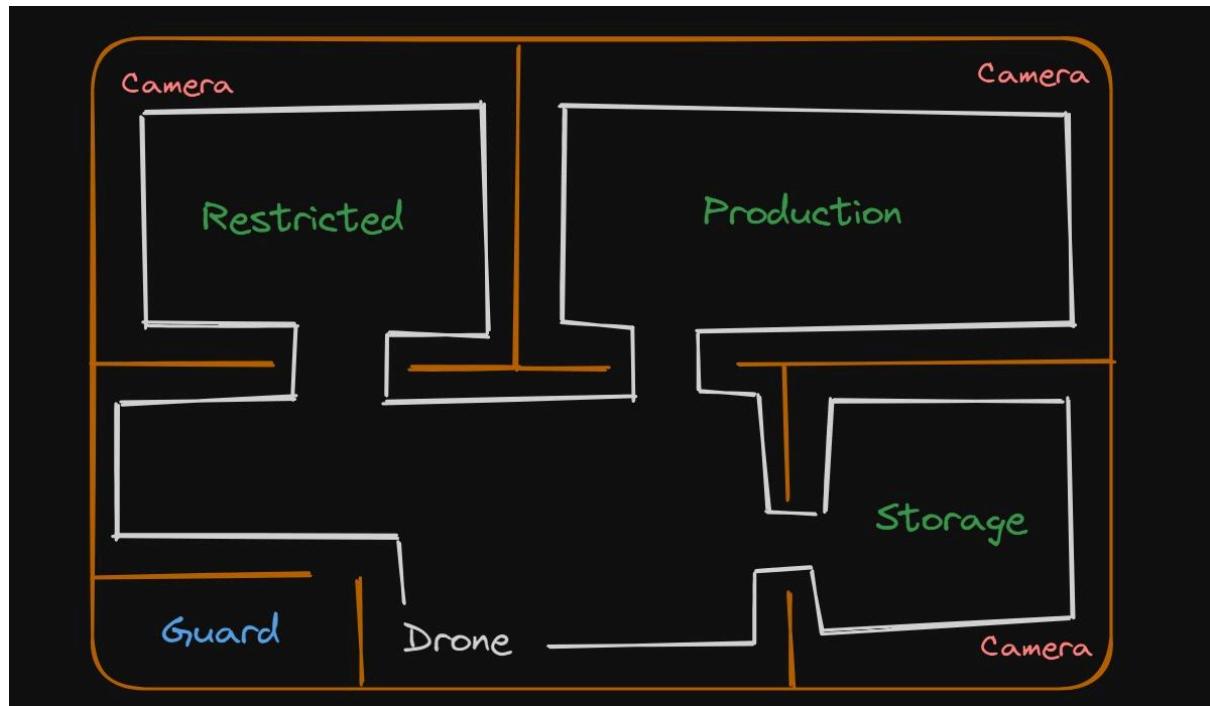
Fecha: 3 - 6 de septiembre de 2024

Tareas:

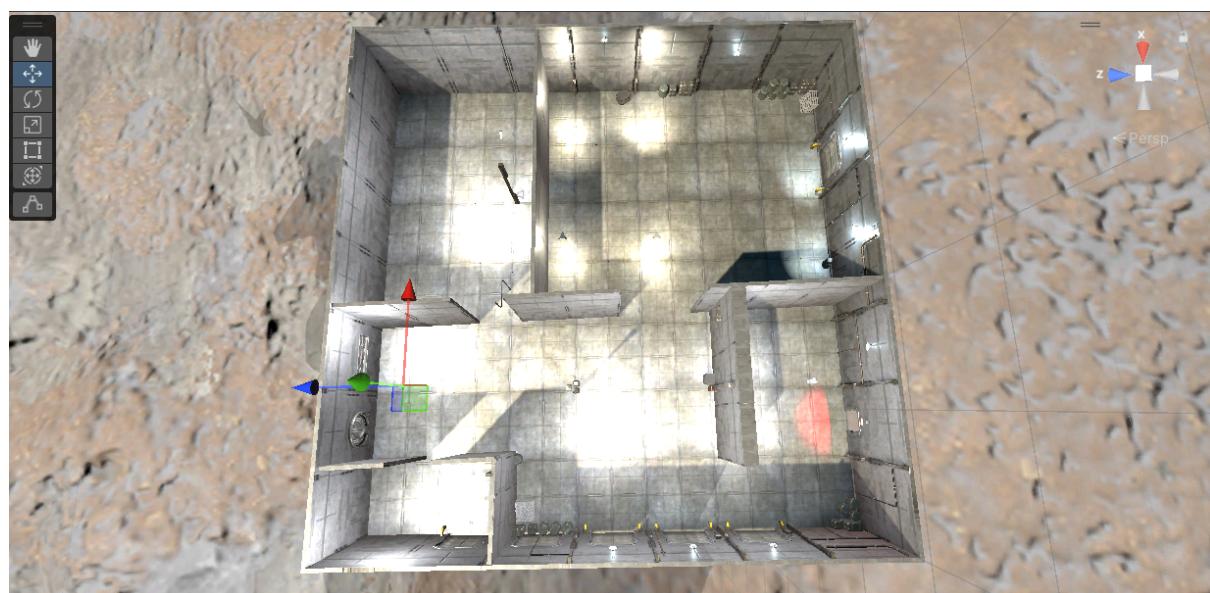
Crear y organizar la presentación final del proyecto.

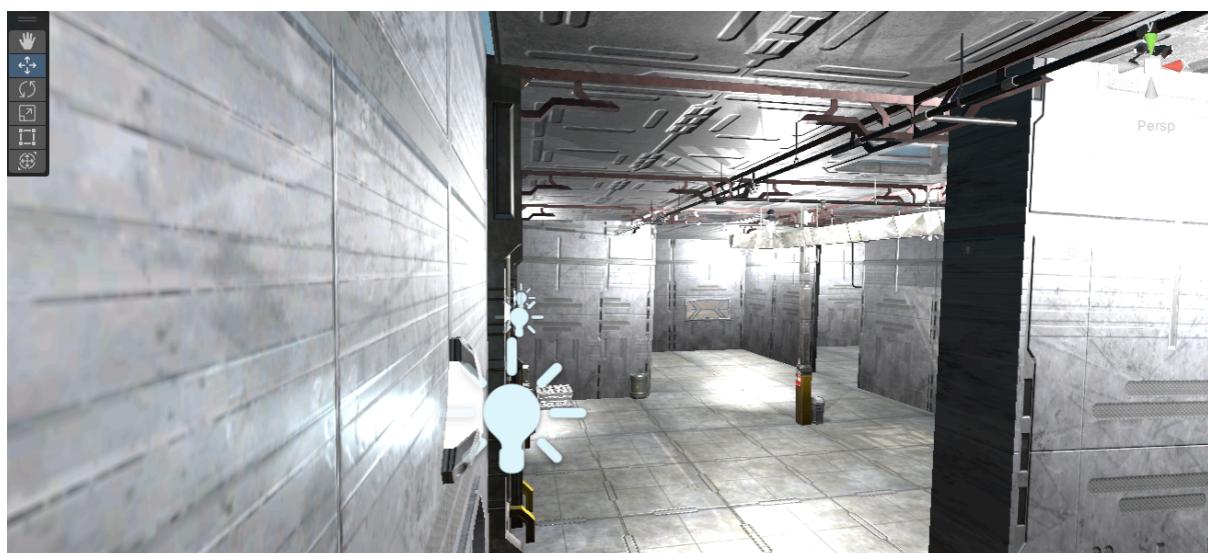
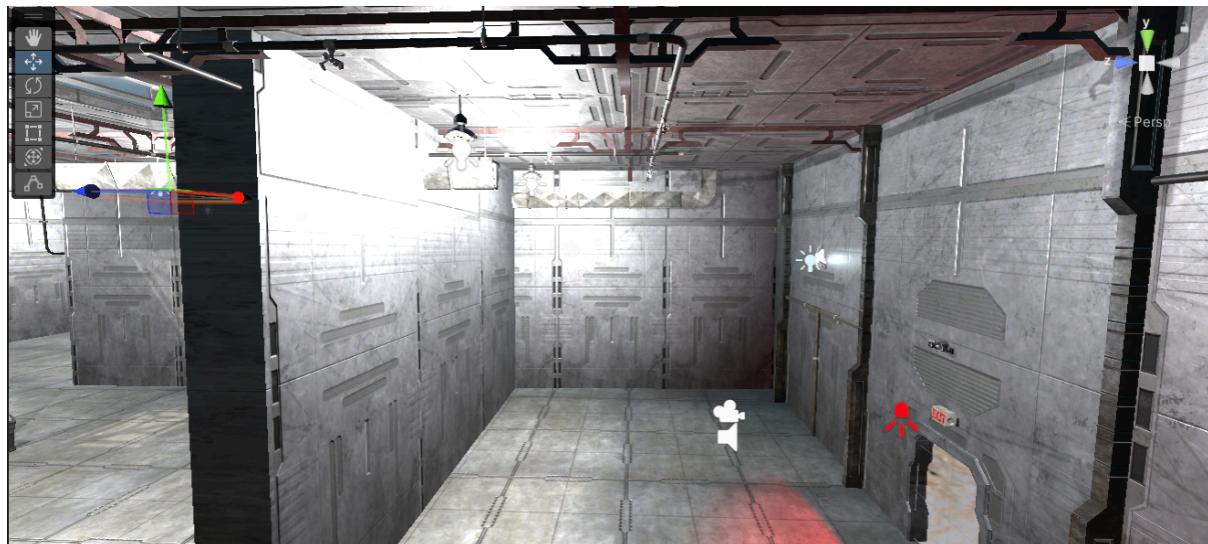
Asegurarse de que toda la documentación mantenga consistencia gráfica (fuentes, colores, márgenes, etc.).

Bosquejo inicial del ambiente:



Fotos del proyecto en Unity :







Fotos del servidor de Python funcionando desde Postman

Inicia simulación con begin = 1

POST Call Post Python X

POST ▾ http://127.0.0.1:8000/	Send ▾	200 OK	23 ms	35 B	Just Now 02-Sep
Body (JSON) ▾		Preview	Header	Request	Tests
JSON <pre> 1 ▼ { 2 "begin": 1 3 }</pre>		<pre> 1 ▼ { 2 "dron": [3 24, 4 10 5], 6 "intruder": [7 11, 8 12 9] 10 }</pre>			

```

sign0ret@Sign0ret-msi-alpha15:~/dev/python/drone_agents/droneAPI$ uvicorn app:app --reload
INFO:     Will watch for changes in these directories: ['/home/sign0ret/dev/python/drone_agents/droneAPI']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [17202] using StatReload
INFO:     Started server process [17204]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
vivo
Step: 0
Message: 0
Path found: [(11, 12), (10, 12), (9, 12), (8, 12), (7, 12), (6, 12), (5, 12), (4, 12), (3, 12), (2, 12), (1, 12), (1, 13), (1, 14), (1, 15), (1, 16)]
INFO:    127.0.0.1:53614 - "POST /step HTTP/1.1" 200 OK

```

Foto de el dron en la posición del intruso

POST ▼ http://127.0.0.1:8000/ Send ▼ 200 OK 5 ms 35 B Just Now | 02-Sep

Body (JSON) ▼

JSON

```
1 ▼ {  
2     "begin": 0  
3 }
```

Preview Header Request Tests

```
1 ▼ {  
2     "dron": [  
3         21,  
4         15  
5     ],  
6     "intruder": [  
7         21,  
8         15  
9     ]  
10 }
```

INFO: 127.0.0.1:33562 - "POST /step HTTP/1.1" 200 OK
sigo vivo
Step: 37
Message: 0
INFO: 127.0.0.1:33562 - "POST /step HTTP/1.1" 200 OK
sigo vivo
Step: 38
Message: 0
CAMINO DE DRON A INTRUSO
[(22, 14), (21, 14), (21, 15)]
MOVER DRON
INFO: 127.0.0.1:33562 - "POST /step HTTP/1.1" 200 OK
sigo vivo
Step: 39
Message: 0
MOVER DRON
INFO: 127.0.0.1:33562 - "POST /step HTTP/1.1" 200 OK
sigo vivo
Step: 40
Message: 0
MOVER DRON
Drone reached the target and is calling the guard.
INFO: 127.0.0.1:33562 - "POST /step HTTP/1.1" 200 OK