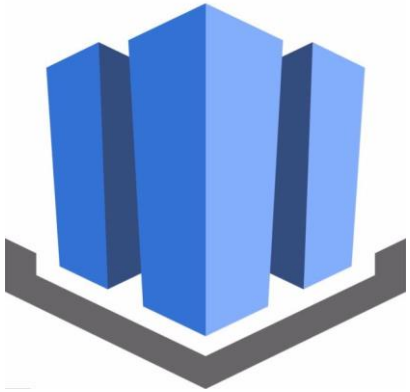
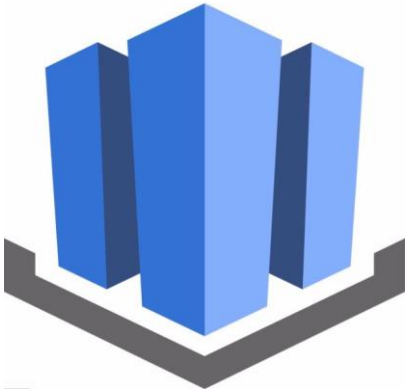


SISTEMA DE DECISION



NEREA CASTELLANOS RODRÍGUEZ
CATHERINE CASTRILLO GONZÁLEZ
SANDRA FRAILE INFANTE
STOYCHO IVANOV ATANASOV
JULIA MARTÍNEZ VALERA
GASPAR RODRÍGUEZ VALERO

SKYSCRAPERS Ver.: 2.0

Tabla de contenido

Introducción

El sistema de decisiones es aquel que se encarga de gestionar y decidir que realizara en la siguiente actualización del bucle general del juego. Por ello y la importancia que tiene, este sistema fue actualizado a una versión más óptima y de nuestra opinión mejor que el sistema anterior

NPC Agresivos

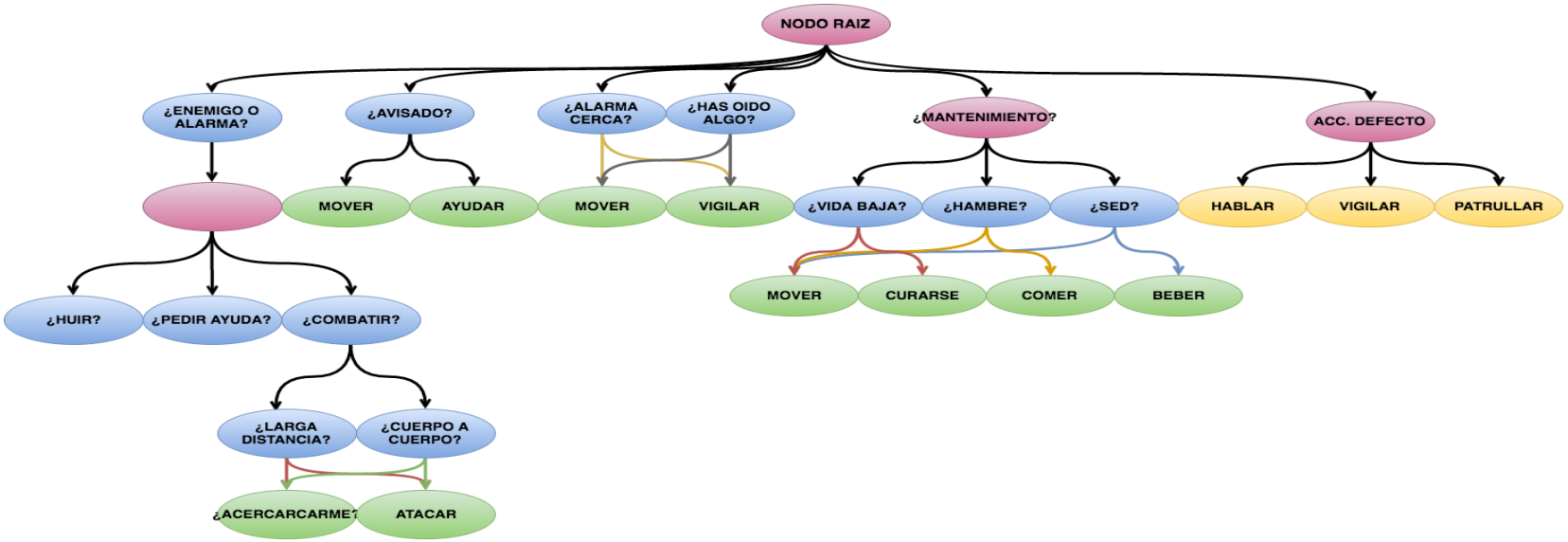
Behaviour Tree

Los behaviour trees se encargan de gestionar y seleccionar cual es la decisión que tomara la IA y en consecuencia realizar las acciones que dicha decisión acompañan. El sistema anterior comprendía un total de 4 arboles gestionados por 4 estados distintos lo que provocaba una complejidad innecesaria para las acciones que la IA debía tomar. Por ello se ha realizado un único árbol que gestionara las acciones de la IA haciendo el sistema más simple y con la misma cantidad de acciones y decisiones.

Este sistema es susceptible de ser modificado en el futuro por posibles mejoras o cambios en futuras iteraciones. Aquí se puede ver el nuevo árbol y los nuevos nodos que lo conforman.

Antes de dar paso a la explicación de los nodos debemos conocer los diferentes estados o valores de vuelta que pueden gestionar y así comprender las posibilidades del sistema de forma completa. Los estados o valores que pueden gestionar son los siguientes:

0. **FALLO**: Este valor es devuelto por cualquier nodo cuando su condición, acción o ejecución no se ha podido llevar a cabo por no cumplir los requisitos
1. **EXISTO**: Este valor es devuelto cuando la condición, acción o ejecución se ha llevado a cabo sin ningún problema.
2. **RUNNING**: Este estado es devuelto cuando la acción del nodo o su ejecución no ha podido ser terminada y devuelve el valor para que en la próxima ejecución prosiga con su trabajo.
3. **RESET**: Este valor está valorado de forma teórica para el caso de que un nodo requiera del reinicio del árbol y este se ejecute de forma estándar, eliminado posibles estados de running.



Los estados anteriores son aquellos definidos en nuestro arboles no siendo lo únicos estados y pudiendo si es necesario crear nuevos valores por necesidad del programa. Prosiguiendo con la explicación de los nuevos nodos, en este árbol los tipos han sido modificados a otros distintos por ello a continuación vamos a describir los distintos tipos de nodo:

- **Nodo Secuencia**: Ejecuta de forma ordenada los nodos hijos que les pertenece, pero a la primera respuesta negativa o estado running termina su ejecución.
- **Nodo Acción** (VERDE): Se trata de un nodo final donde se lleva la ejecución de la acción, aunque también puede devolver running.
- **Nodo Condición**: Controla si cumple los requisitos para continuar adelante por la rama del árbol.
- **Nodo Secuencia Positiva** (ROSA): Ejecuta cada uno de sus hijos hasta alcanzar un nodo hijo que devuelva una respuesta positiva o un estado running.
- **Nodo Condición Secuencial** (AZUL): Es la unión de un nodo de secuencia y uno nodo secuencia, busca que primero se pase la parte de la condición y después ejecutar sus hijos hasta que uno le devuelva un valor negativo o un estado running.
- **Nodo Subrutina** (AMARILLO): Se tratan de acciones cuya configuración no concuerda con un nodo simple, si no agrupa diferentes acciones y condiciones. Actualmente estos nodos se encuentran en una etapa temprana.

Algunos de los nodos tratados anteriormente no tienen asignado ningún color, este es así pues no forman parte del árbol, pero su comportamiento y su función son necesarios para entender el resto del árbol o de donde provienen otros nodos. Tras la explicación general de los nodos a continuación vamos explicar más profundamente que realiza cada uno de los nodos en su programación.

Funciones del sistema

En el sistema anterior gestionábamos los diferentes eventos del mapa a modo de estados teniendo los 4 estados que teníamos entonces:

- ESTANDAR
- ALERTA
- AGRESIVO
- ASUSTADO

Pero tras la valoración vimos que el resultado era demasiado complejo para el resultado obtenido por lo que se valoró unir todos los estados y posibilidades en un único árbol donde se decidió plantearlos de la siguiente manera.

- ¿ENEMIGO O ALARMA?: Este nodo se encuentra en la cumbre de la parte del árbol que se encarga del combate y la alerta del NPC, en este se hacen las valoraciones respecto a dos FLAGS, si ha visto al enemigo o a sonado la alarma, de este pasa a un nodo en blanco, cuya función es ejecutar todas las posibilidades y verificar si alguna de estas se puede llevar a cabo.
    - ¿HUIR?: Este nodo se trata de un nodo condición secuencia, actualmente al igual que algunos de sus compañeros no contienen hijos pues se siguen valorando cuales pueden componer. Este nodo valora si se cumplen las condiciones para que deba de huir del combate. Nuestro NPC de las diferentes variables, las que más pesan son sus FLAGS, salud, sed, hambre y moral. En este caso se hace uso de la moral para realizar este valor, pero el valor de esta variable no es almacenado si no que es calculada cada vez que es solicitada y su fórmula es bastante simple: $\text{salud} * 0.5 + \text{hambre} * 0.3 + \text{sed} * 0.2 = \text{moral}$ y si esta se encuentra por debajo de un 15% es cuando decide huir.
    - ¿PEDIR AYUDA?: Como el nombre nos sugiere gestiona las peticiones de ayuda de nuestro NPC hacia otros NPC o la activación de la alarma. Su condición es parecida al nodo anterior siendo el umbral el 40% de la moral y tras cumplir la condición el NPC realizará tres peticiones en el orden siguiente:
        - AYUDA CERCANA: Solicitara ayuda a sus compañeros cercanos.

        - AYUDA RADIO: Si no recibe respuesta a la primera petición, realizara la misma petición por radio, lo que permite un mayor rango de búsqueda.
        - ACTIVAR ALARMA: Y en el último caso activara la alarma como ultima forma de avisar a los demás y posibilitando a que se acerquen aquellos no muy alejados de la alarma a vislumbrar quien ha activado la Alarma.
        - Si cualquiera de los casos anteriores se cumple se le aplicara este NPC un congelamiento de la función de petición ayuda, para que esta aplicación no se ejecute de forma reiterada, creando un bucle de peticiones de ayuda.
    - ¿COMBATIR?: Al igual que sus compañeros la condición a evaluar será la moral siendo esta superior al 40% o en el caso de congelar el nodo de peticiones de ayuda deberá ser superior al 15% antes de que huya. Esta gestiona el combate y sus hijos recogen las diferentes formas de afrontar dicha gestión.
        - ¿LARGA DISTANCIA?: Este nodo gestiona si debe realizar el ataque a larga distancia y para ello se mide la distancia que tiene el NPC del jugador.
        - ¿CORTA DISTANCIA?: Este nodo realiza la misma gestión, pero a contra distancia y realizara los cálculos hacia unas condiciones de proximidad.
            - En ambos nodos se pueden ver que tienen los mismos hijos y en el mismo orden, la razón de esta ejecución es la siguiente: Ambos nodos deben calcular la distancia del jugador al mismo y valoran de forma similar siendo los umbrales o los limites diferentes, pero realizando los mismos cálculos. En el caso de larga distancia busca que se encuentre entre 5 y 15 cm del jugador (Puede parecer una media pequeña pero la escala del juego es pequeña) y distancias menores ataque cuerpo a cuerpo, aunque siempre que este alejado más de 1 cm deberá acercarse para llevar el ataque a cabo.
- ¿AVISADO?: Verifica si tiene el FLAG activado que referencia a las peticiones de ayuda de sus compañeros y en el caso positivo ejecutar sus hijos.

- MOVER: Este nodo de acción es el encargado de gestionar la pila de posición que contenga y desplazarse a cada una de ellas hasta que la pila se halle vacía, en cuyo caso su funcionamiento habrá concluido y pasara al siguiente. Ejemplo:
  - Este nodo se menciona de forma reiterada en árbol, pero su funcionamiento es siempre el mismo

NODO MOVER
vector<posiciones> pila;
Si la pila está vacía = FUNCIONO
Si no
Spos = pila.begin();
Mov = Mipos – Spos;
Si MOV ≈ 0 = SIGUIENTE POS
Si no RUNNING

- AYUDA: Este nodo su funcionamiento aún continúa siendo teórico, pero se busca que el nodo emisor de la petición le envié el evento que lo coloco en alerta y provocara dicha acción y que el receptor se encargue del evento, creando una comunicación simulada entre NPCs.

- ¿ALARMA CERCANA? ¿HAS OIDO ALGO?: La explicación de estos nodos se trata junta debido a sus similitudes y sus pocas diferencias al igual que sus nodos hijos. Ambos nodos gestionan sus propios FLAGS para conocer si deben actuar o no, siendo el primero alarma y el segundo ruido, pero el resto del funcionamiento es similar, ejecutaran sus hijos, desplazaran al NPC a la posición del evento y vigilaran para conocer o verificar con sus sensores que ha podido ser lo que provocara dicho evento.
- ¿MANTENIMIENTO?: Este nodo de secuencia positiva es el encargado de ejecutar a sus hijos para comprobar si el NPC tiene la salud baja o los niveles de sed y hambre altos.
  - Al igual que los últimos nodos que precedían a este sus funcionamientos son similares verificando los diferentes estados de las variables del NPC y actuar en consecuencia acudiendo al objeto de mundo que pueda solventarlas.
- ACC. DEFECTO: Este nodo verifica cuál de las acciones por defecto se puede llevar a cabo, cuando se alcanza esta parte del árbol quiere decir que el nodo no tiene nada prioritario que hacer y procederá a realizar dichas acciones hasta que aparezca un nuevo evento que gestionar.

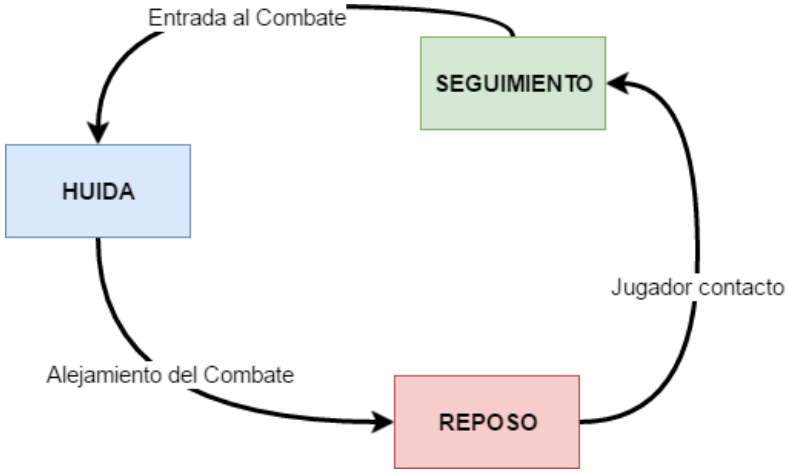
NPC Amigables

En el caso del nuestro proyecto el único NPC amigable se trata del rehén de cada nivel el cual desea ser rescatado. Toda la parte redactada a continuación se encuentra en desarrollo, se tienen notas y planteamientos teóricos, pero aún estamos con la primera iteración.

Máquina de Estados

Buscamos que el rehén fuera gestionado por una máquina de estados simple ya que la cantidad de acciones o estados del mismo son reducidas. Hemos decido que tendrá tres estados simples:

- **Reposo**: En este estado se encuentra cuando está esperando al NPC a lo largo de la partida o cuando tras ser salvado por el jugador, el mismo entra en combate, el rehén huya y vuelva ese estado hasta que se encuentre de nuevo con el jugador, es decir, el rehén sale de este estado cuando tiene contacto físico con el jugador lo que provoca el cambio de estado al estado de seguimiento.
- **Huida**: Cuando el rehén se encuentra involucrado en un combate, esta huira hasta encontrar sitio seguro. La huida se realizará con el Pathplanning pero de forma sencilla, el rehén solicitara el cálculo del nodo del grafo más cercano y utilizará el mismo para calcular tres nodo más que lo alejen y tras llegar a ellos, entrará en el estado de reposo.
- **Seguimiento**: En este estado, el rehén seguirá de forma continua al jugador hasta el destino final del nivel o hasta que sufran un combate. Para simplificar el funcionamiento del rehén, el mismo no tendrá un sistema de movimiento a la hora perseguir al jugador si no que mediante físicas se enganchara al personaje haciendo la persecución fácil.



Notas

Versión 1.0

Primera versión funcional del sistema de decisión, muchas de las decisiones se hallan deshabilitadas a espera de la creación de un Trigger System y un Blackboard dinámico con su correspondiente arquitectura. En esta versión se han comprobado:

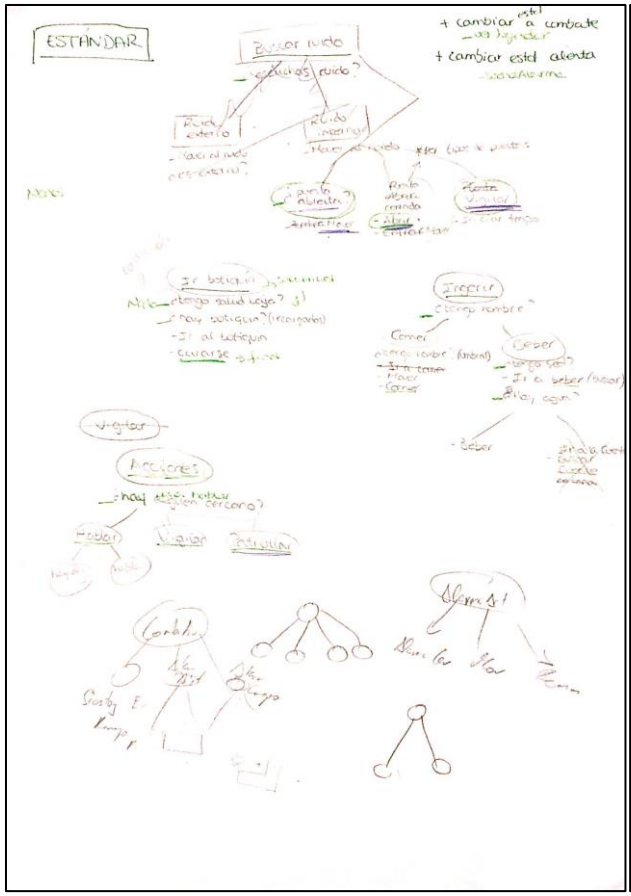
- Arboles de decisión probados (Orden correcto y gestión del estado running optimizado)
- Primera versión del nodo mover (nodo critico que gestiona mucha de las decisiones del sistema)
- Comprobada la máquina de estados para la gestión de árboles.
- Árbol estándar testeado en sistema 3D
    - Se cura, alimenta o bebe cuando lo requiere
    - Realiza su rutina cuando debe.

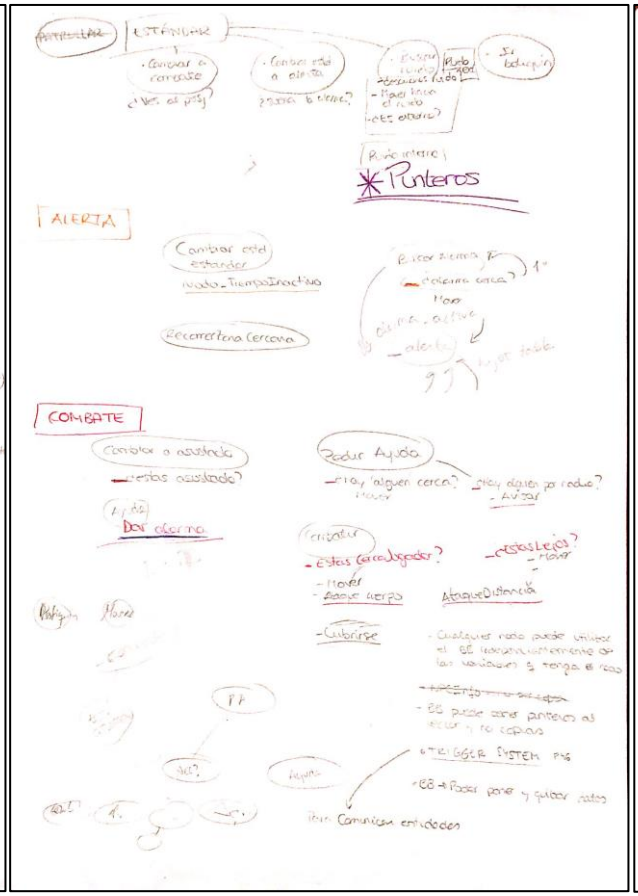
Versión 2.0

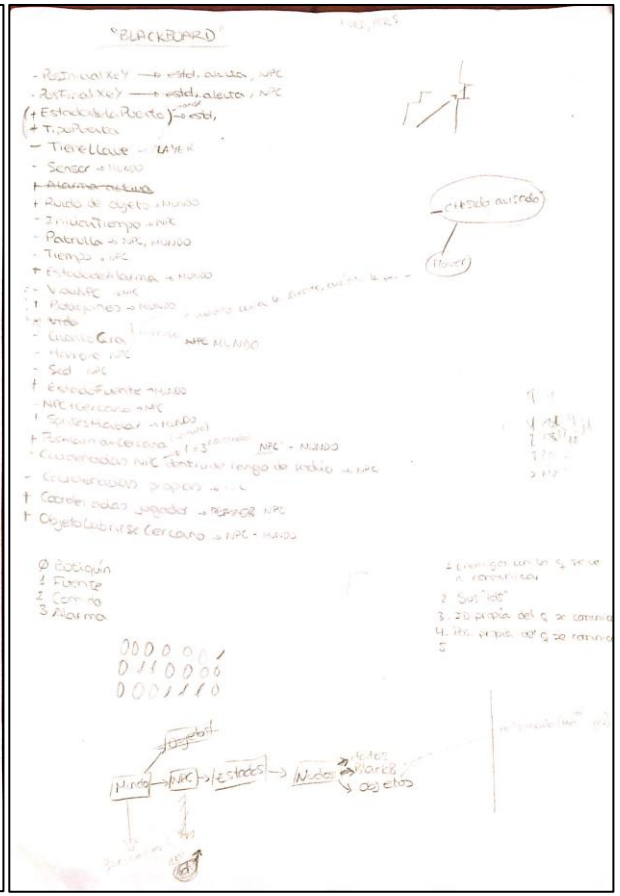
Segunda versión funcional del sistema de decisión, las decisiones respecto al sistema de combate se hallan sin implementar. Se ha implementado los distintos blackboard y el trigger system. Se ha eliminado la parte de los estados y los arboles han sido reducidos a un único árbol. En esta versión se ha comprobado:

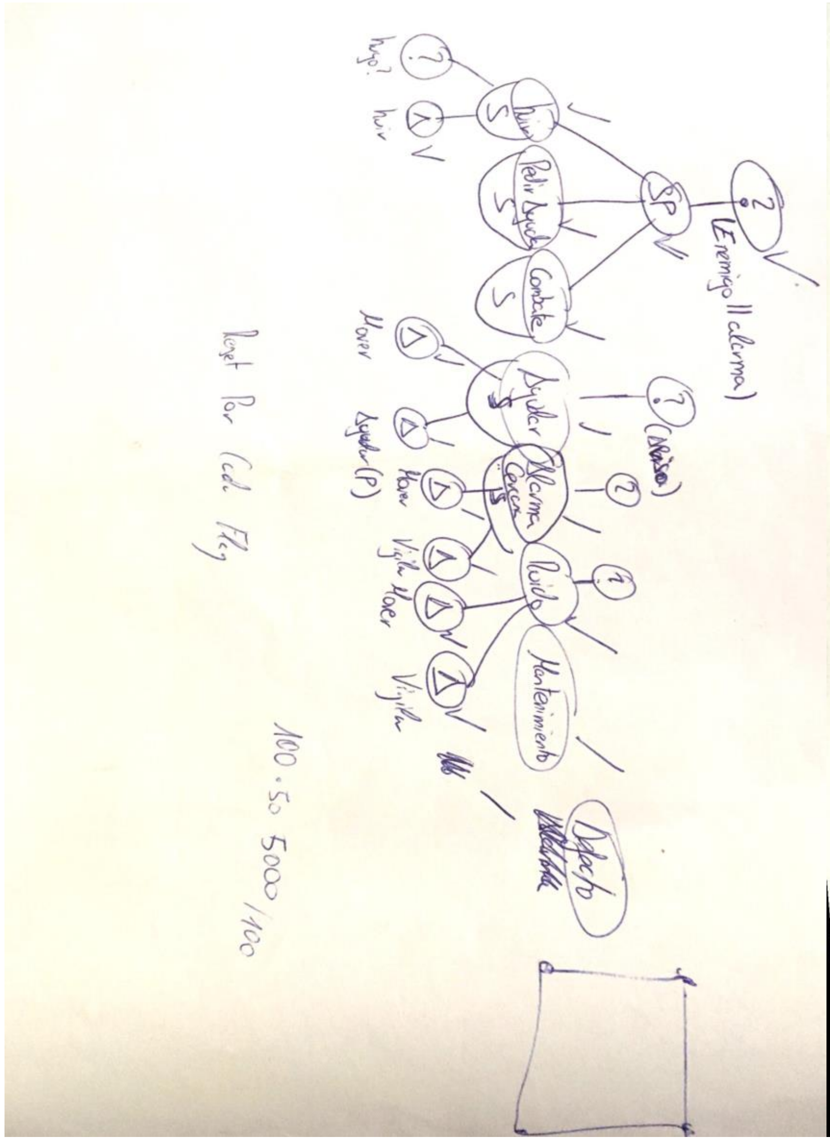
- Árbol de decisión comprobado, tanto orden de ejecución, como el estado running y el reset del mismo.
- Segunda versión del nodo mover (gestión de múltiples posiciones a modo de pila) y cálculos simplificados.

Bocetos y Anotaciones









Enemigo true   00000

ruido desactiva
Alarma no
aviso/radio si (tiempo)

Alarma true   (Combate Alarma Huir)

aviso, ruido si

Sensor: Enemigo, Alarma activada, Ruido
Comunicación: Pedir Ayuda, Ayudar
ruido
Asustado → huir
Combate   Alerta

Estandar ≃ Alerta (+ ...)

if (moral ≤ 40%) && (Enemigo || Alarma)
Pedir Ayuda
if (moral ≤ 10%) &&
huir
moral

huir ← Pedir Ayuda ← Combate
-moral   ↓   -moral
corta distancia   larga   Alarma

if (Enemigo && moral > 40%)
Combate

Comerse   Alimentarse   Beber   Accion Pred.
+ prioridad