**28 ASISTENTE DE SALUD PERSONALIZADO**

Un agente inteligente que monitorea constantemente la salud del usuario mediante wearables y proporciona recomendaciones personalizadas para dietas, ejercicios, y recordatorios de medicación basándose en el historial médico y las condiciones actuales del usuario.

**Identificar qué tipo de agente, justifique su respuesta.**

Caso de estudio específico:

Un asistente inteligente para corredores de maratón que monitorea la hidratación, fatiga y riesgo de lesiones en tiempo real, proporcionando alertas y recomendaciones personalizadas durante el entrenamiento y la competición.

Se identifica como un agente basado en la utilidad, no solo responde a estímulos inmediatos, sino que evalúa múltiples factores y selecciona la mejor acción para optimizar el rendimiento del corredor.

Puede calcular la mejor estrategia de hidratación, ritmo cardiaco y prevención de lesiones en función de métricas como temperatura, fatiga acumulada y frecuencia cardiaca.

También puede ser aprendizaje autónomo, podría aprender con el tiempo y mejorar sus sugerencias con cada entrenamiento o carrera.

puede llegar a tener característica de agente reflejo con estado modelo debido a que recoge almacena datos históricos (frecuencia cardíaca, hidratación, ritmo de carrera) para tomar mejores decisiones en el futuro.

Si detecta patrones de fatiga o riesgo de lesiones, ajusta sus recomendaciones.

**Identificar y describir el ambiente(s), que utiliza el agente.**

Parcialmente observable:

El agente no tiene acceso a toda la información en todo momento.

Aunque usa sensores(wearables), estos pueden ser limitados. Como por ejemplo no mide con exactitud la hidratación interna, solo la sudoración estimada.

Factores como el estado mental del corredor o su nivel real de agotamiento puede no ser 100% detectables.

Estocástico:

El rendimiento del corredor depende de muchas variables incontrolables (clima, emociones, hidratación real, fatiga acumulada).

El mismo esfuerzo en diferentes días puede producir distintos resultados.

Secuencial:

Cada mala acción afecta el estado futuro del corredor como ejemplo una mala hidratación el km 10 impacta el desempeño en el km 3.

No puede tomar decisiones de forma aislada, hay una progresión de eventos.

Dinámico:

El corredor cambia constantemente mientras el corredor avanza en la carrera.

La fatiga, el clima y el rendimiento no son estáticos, evolucionan con el tiempo.

Continuo:

Las variables (ritmo cardíaco, hidratación, velocidad) cambian en un rango continuo y no valores fijos.

No se toma decisiones en pasos discretos, sino que la información fluye constantemente.

Agente único:

Solo el agente interactúa con el corredor.

**Diseñar la arquitectura del agente.**

Estado

Sensores

Wearables (HR, GPS), Sensores de hidratación, Datos de clima (API)

Estado del mundo

Función de utilidad

Basándose en la función de utilidad, el agente decide la acción más adecuada y la comunica al corredor.

Entorno

Actores

Permite que el agente tome decisiones informadas y no solo reaccionarias.

Predice cómo evoluciona la condición del corredor basándose en los datos recopilados

Aquí se evalúa la situación del corredor con base en múltiples variables para determinar la mejor acción a tomar.

Factores externos: clima, altitud, terreno.

Condición del corredor: fatiga, hidratación y ritmo cardiaco

Construir PEAS:

**Performance Measure (P) – medida de desempeño**

El rendimiento del agente se mide según su capacidad para mejorar la experiencia y seguridad del corredor.

Detecta niveles de hidratación bajos y alerta al corredor antes de que afecte su rendimiento.

Recomendaciones de consumo de agua en momentos claves.

Gestión de fatiga:

Evaluar la fatiga en función del ritmo cardiaco, el esfuerzo y la duración de la carrera.

Sugerir cambios en el ritmo o pausas estratégicas para evitar agotamiento extremo.

Prevención de lesiones:

Monitoreo continuo del impacto en articulaciones y musculo.

Analisis la forma de correr (si se tiene sensores de zancada o pisada) para detectar patrones de sobrecarga.

Detecta patrones de riesgo de lesiones y sugerencia de corrección.

**Environment (E) Entorno**

El agente opera en un entorno parcialmente observable, dinámico y continuo compuesto por:

Condiciones del corredor:

Ritmo cardiaco

Variabilidad de la frecuencia cardiaca.

Nivel de hidratación (estimado por sensores de sudoración y consumo de líquidos.

Estado de fatiga muscular.

Condiciones ambientales:

Temperatura y humedad ambiental

Altitud y presión atmosférica

Velocidad del viento.

Características del recorrido:

Tipo de terreno (asfalto, tierra, inclinación, descanso)

Distancia recorrida y pendientes del camino

Historial del corredor:

Datos de entrenamiento y carreras previas

Patrones de hidratación y fatiga

Frecuencia y tipo de lesiones anteriores

Actuators (A) Actores:

El agente interactúa con el corredor mediante diversos medios:

Notificaciones en smartwatch

Vibración y alerta visuales sobre hidratación, ritmo y fatiga.

Mensaje de voz:

Recomendaciones audibles durante la carrera para mejorar la experiencia sin necesidad de mirar la pantalla.

Ejemplo reduce la velocidad tu frecuencia cardiaca es alta.

Ajuste dinámico del plan carrera:

Basado en el estado actual del corredor y las condiciones del entorno

Ejemplo si la temperatura aumenta ajustar la recomendación de hidratación.

Dashboard de análisis post-carrera:

Estadísticas detallas sobre el desempeño, sugerencia de mejorar y posibles ajustes para futuras carreras.

Senors (S)- Sensores:

El agente recopila informacion en tiempo real:

Sensores corporales (wearables):

Frecuencia cardiaca (ritmo y variabilidad, clave para medir fatiga)

Sensores de hidratación (estimación basada en sudoración y consumo de líquidos).

Sensores de movimientos (para analizar el ritmo y el impacto en articulaciones).

GPS y acelerómetro:

Medición de velocidad y distancia recorrida.

Detección de cambios brusco en ritmo o inclinaciones del terreno.

APIs externas:

Datos meteorológicos en tiempo real (temperatura, humedad, altitud).

**Definición del “State Space” de la arquitectura:**

El espacio de estado representa todas las posibles condiciones en las que puede encontrase el corredor en un momento dado. Se define mediante un conjunto de variable que describen su estado fisiológico, el entorno y su desempeño en la carrera.

1. Estado (S): Configuración posibles del sistema.

Es el conjunto de todas las configuraciones posibles en las que puede encontrarse el sistema.

El caso del agente un estado especifico podría ser:

S = {HR = 165 BPM, Hidratación = 55%, Fatiga = 70%, Ritmo = 4:30 min/km, Distancia = 15 km, Temperatura = 30°C}

HR = Ritmo cardíaco

Hidratación = Nivel de líquidos en el cuerpo

Temperatura = Condición ambiental

Fatiga = nivel de 0= mínimo, 100= máximo

Ritmo = Velocidad del corredor

Distancia = Kilómetros recorridos

Este estado indica que el corredor tiene frecuencia cardiaca alta, hidratación baja, fatiga elevada, un ritmo fuerte y está a 15 km de la maratón con clima caluroso

1. Estado inicial ( Punto de partida en el espacio de estado.

Es el estado en el que comienza el agente antes de ejecutar cualquier acción.

Un ejemplo: al inicio de la carrera el corredor podría estar completamente hidratado y sin fatiga.

s₀ = {HR = 90 BPM, Hidratación = 100%, Fatiga = 0%, Ritmo = 0 km/h, Distancia = 0 km, Temperatura = 20°C}

1. Estado meta o finales ( Estados que representan soluciones aceptables.

El objetivo del agente es ayudar al corredor a alcanzar un estado óptimo al final de la carrera, minimizando la fatiga y los riesgos.

S\_goal = {HR ≤ 150 BPM, Hidratación ≥ 70%, Fatiga ≤ 50%, Ritmo ≈ 5:00 min/km, Distancia = 42 km}

1. Acciones (A) Conjunto de operaciones o movimientos permitidos.

El agente puede realizar diferentes acciones para optimizar el rendimiento del corredor y minimizar riesgos.

Acciones disponibles:

A = {Reducir ritmo, Acelerar, Mantener ritmo, Hidratarse, Detenerse}

Un ejemplo acciones

Si HR > 160 BPM→ reducir ritmo

Si hidratación < 50%→beber agua

Si fatiga >80% →tomar una pausa breve

Si ritmo es estable y sin riesgo → mantener velocidad

1. Función de transición (T(S,a) → s'): regla que define como un estado cambia a otro al aplicar un acción.

En este estado se define como cambia el estado actual S a un nuevo estado actual S´ después de aplicar una acción A.

Si S = {HR = 170 BPM, Hidratación = 45%, Ritmo = 4:10 min/km}

y la acción A = "Reducir ritmo"

entonces S' = {HR = 160 BPM, Hidratación = 50%, Ritmo = 4:40 min/km}

aquí en este ejemplo reducir la velocidad mejora la hidratación y baja frecuencia cardiaca.

1. Costo o recompensa (C(s ,a ,s´)) valor asociado a cada transición, usado en problemas de optimización.

Se usa para evaluar si un estado es favorable o no. En un agente basado en utilidad, cada estado recibe una puntuación según qué tan beneficioso es. Asignando un costo o recompensa a cada transición de estado.

U(S) = - (Fatiga + Riesgo de deshidratación + Riesgo de lesión)

Aquí vemos que los estados óptimos reciben puntuación alta (baja fatiga, buena hidratación)

Estados peligrosos reciben puntuación baja (alta fatiga, deshidratación)

Ejemplo costo y recompensa

Si S' mejora la hidratación y reduce fatiga → Recompensa +10

Si S' aumenta fatiga y riesgo de lesión → Costo -15

Aquí permite que el agente optimice sus decisiones basándose en la mejor utilidad posible.

**Describir la solución utilizando los algoritmos de búsqueda (justificar detalladamente su respuesta).**

Para el agente inteligente para corredores de maratón puede modelarse como un problema de búsqueda, donde el objetivo es encontrar la mejor secuencia de acciones que lleve al corredor desde un estado inicial hasta un estado meta.

Para darle solución, se evaluó con distintos algoritmos de búsqueda. Se llegó a la conclusión el algoritmo más adecuado es A\* (A-star-Search).

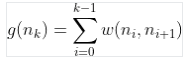
El agente debe encontrar la mejor secuencia de acciones para evitar fatiga extrema, deshidratación y lesiones.

El algoritmo A\* procede con una evaluación y encontrar la mejor trayectoria mediante tres componentes clave: g(n), h(n), f(n), estos componentes trabajan juntos para guiar el proceso de búsqueda hacia los caminos más prometedores.

Podemos decir que selecciona el mejor camino combinado:

g(n) el costo real acumulado del esfuerzo del corredor hasta el momento (fatiga, hidratación, ritmo cardíaco).

Matemáticamente para una trayectoria a través de los nodos (nodo inicia) a nodo actual podemos expresar g(n) como



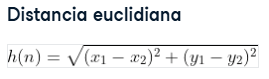
Donde representa el peso de la arista que conecta

h(n) una estimación del esfuerzo restante necesario para completar la maratón en buenas condiciones.

Matemáticamente para cualquier nodo n dado, la estimación heurística debe satisfacer la condición h(n) donde h\*(n)es el coste real del objetivo.

Las funciones heurísticas comunes incluyen la distancia manhattan y distancia euclidiana. Las coordenadas del nodo actual y del nodo meta, estas distancias se calculan.





f(n)=g(n) + h(n) valor que guía la búsqueda del mejor camino, considerando tanto lo que ha pasado como lo que falta.

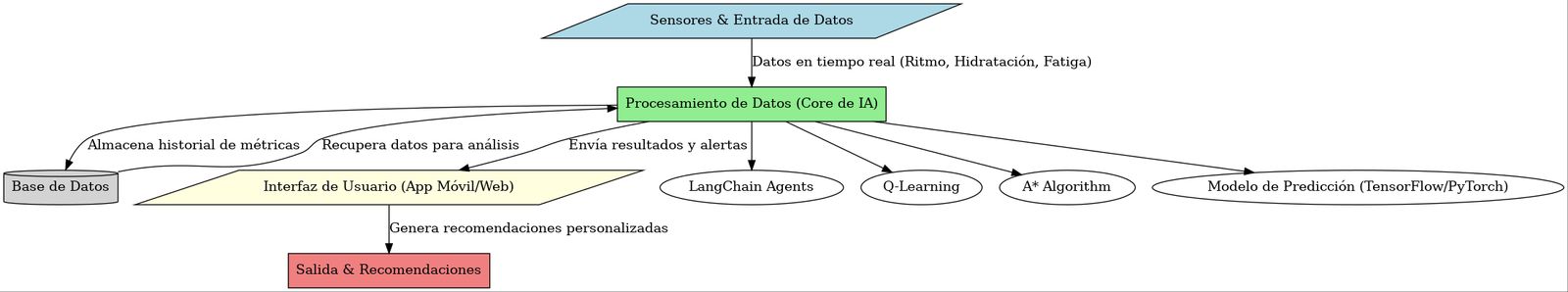
La toma de decisiones del algoritmo A\*, que combina tanto el coste real de la ruta como la estimación heurística para evaluar el potencial de cada nodo.



g(n) representa como lo hemos mencionado antes el coste real desde su inicio hasta el nodo actual.

h(n) representa el coste estimado desde el nodo actual hasta el objetivo.

**Diseñar la lógica del agente mediante diagramas de actividades.**

****

**Análisis críticos sobre las herramientas actuales de Agentes Inteligentes (comparar, contrastar y concluir).**

**Comparación:**

|  |  |  |  |
| --- | --- | --- | --- |
| Herramienta | Descripción | Ventaja | desventaja |
| CrewAI | Framework para la colaboración entre múltiples agentes especializados. | \*Enfoque en agentes colaborativos.  \*Integración con modelos de IA.  \* Modular y escalable. | \*Relativamente nuevo.  \* Puede requerir ajustes para tareas específicas. |
| AutoGEN | Plataforma para la creación de agentes colaborativos que optimizan la comunicación entre múltiples LLMs. | \*Permite interacción entre agentes autónomos.  \*Diseñado para optimización en diálogos IA. | \* Requiere configuración avanzada.  \*Dependencia de modelos externos. |
| Langchain Agents | Extensión de Langchain para agentes inteligentes que interactúan con herramientas y memoria. | \*Gran flexibilidad y personalización.  \*Integración con bases de datos y APIs. | \*Complejo de configurar.  \*Alto consumo de recursos. |
| AutoGPT | Agente autónomo que puede desglosar tareas y ejecutarlas sin intervención humana. | \*Totalmente autónomo.  \* Uso de memoria y herramientas externas. | -\*Puede ser ineficiente sin restricciones.  \*Consumo alto de tokens. |
| BabyAGI | Implementación ligera de un sistema de inteligencia general autónoma. | \*Más eficiente que AutoGPT.  \*Mejor control sobre | \*Funcionalidad limitada en comparación con AutoGPT. |
| MetaGPT | Modelo para la colaboración estructurada entre múltiples agentes especializados en tareas empresariales. | \*Optimizado para flujos de trabajo complejos. \*Enfoque en colaboración en proyectos. | \*Requiere configuración específica. |
| SuperAGI | Plataforma para la creación de agentes avanzados con acceso a herramientas y memoria persistente. | \* Más eficiente que AutoGPT.  \* Soporte para herramientas avanzadas. | \*Configuración más compleja. |

**Contraste:**

Autonomía vs. Control:

AutoGPT y BabyAGI son los más autónomos, pero pueden ser difíciles de controlar sin restricciones.

Langchain Agents y SuperAGI ofrecen mayor control sobre los agentes.

CrewAI, AutoGEN y MetaGPT están diseñados para colaboración entre agentes.

Eficiencia y Consumo de Recursos:

AutoGPT consume muchos tokens al procesar tareas sin restricciones.

BabyAGI y SuperAGI mejoran la eficiencia con mayor control.

Langchain Agents y MetaGPT permiten optimización personalizada.

Casos de Uso Ideales:

AutoGPT y BabyAGI: Tareas totalmente automatizadas con mínima intervención humana.

CrewAI y MetaGPT: Proyectos que requieren colaboración entre múltiples agentes especializados.

Langchain Agents y SuperAGI: Integración con herramientas externas y procesos empresariales.

AutoGEN: Optimización de interacciones entre múltiples agentes.

Conclusión:

Para diseñar este agente inteligente es crucial elegir herramientas que ayuden a optimizar la gestión de datos, toma de decisiones en tiempo real y llevar acabo respuestas personalizadas. Por tal motivo después de comparar todas estas herramientas se optó por hacer una combinación Langchain, SuperAGI, AutoGEN.

Se usaría Langchain para que facilite la integración con multiples fuentes de datos.

Utilizaríamos memoria contextual para adaptar recomendaciones basadas en el comportamiento del atleta.

AutoGEN nos ayudaría con la colaboración entre agentes asegurando una comunicación eficiente.

Puedo optimizar la toma de decisiones en escenarios dinámicos.

Puede personalizar la interacción con el atleta.

SuperAGI nos ayuda con proporcionar la gestión de multiples agentes avanzados

Puede ofrecer escalabilidad y optimización.

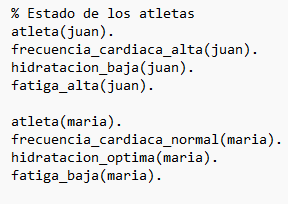
Diseñar, modelar y programar un agente inteligente para un caso de uso específico, aplicando conceptos clave como la arquitectura del agente, su espacio de estados y los algoritmos de búsqueda. Además, se debe implementar el agente utilizando **Lógica de Primer Orden (FOL)**.

**Lógica de primer Orden (FOL)**

Esta lógica nos permitirá representar el conocimiento del agente mediante hechos, reglas y inferencia. Esto permitirá al agente de salud pueda razonar y tomar decisiones lógicas sobre el estado del atleta.

Empezaremos con los hechos datos del atleta,

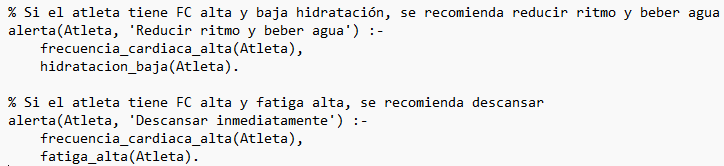
Estos hechos son la información sobre el estado actual que tiene atleta, por ejemplo:



Esto nos indica que el atleta juan tiene frecuencia cardiaca alta y baja hidratación. Por otro lado, la atleta maría está en buen estado.

Reglas en FOL (alertas):

Aquí podemos definir cuando el atleta recibe una alerta o recomendación



Esta regla le permite al agente detectar situaciones de riesgo y dar recomendaciones.

Inferencias en FOL decisiones.

El agente consulta los hechos y reglas para generar una recomendación



Aquí el agente con los hechos y reglas está generando una recomendación para el atleta juan.



Arroja dos recomendaciones o alertas por su estado el agente determina que es **crítico.**

**Asistente inteligente personalizado**

**GitHub:** [**https://github.com/angeloperezcorrea/Asistente-de-salus-personalizado.git**](https://github.com/angeloperezcorrea/Asistente-de-salus-personalizado.git)