ESCOLA SUPERIOR DE ENXEÑARÍA INFORMÁTICA

UNIVERSIDADE DE VIGO

SISTEMAS INTELIGENTES

Domestic Robot: Memoria segunda parte

Grupo 21

Eva Castiñeiras Puente
Miguel Diéguez Vilas
Hugo Fernández Fernández
Jaime Villota Martínez

Índice

Introducción

En este documento se presenta la segunda entrega del Domestic Robot para la asignatura de Sistemas Inteligentes desarrollado por el grupo SI-21 para el curso 2024-2025.

Es importante destacar que los objetivos logrados en este proyecto se recogen en la tabla de trazabilidad al final del documento, que se citará a lo largo de la explicación.

El objetivo principal consiste en diseñar un sistema multiagente capaz de suministrar medicación al propietario a la hora indicada por la pauta proporcionada y controlar las existencias de la misma.

Disponemos de una cuadrícula de 12x24 representando el hogar y que está dividida en habitaciones. Consta de una entrada, una cocina, un salón y tres dormitorios, aunque la mayor parte de la actividad será realizada en las tres primeras.

A continuación proporcionamos una breve descripción de los agentes y los objetos presentes en el hogar con relevancia en la consecución de los objetivos.

Agentes

Owner:

El propietario debe seguir unas pautas de medicación definidas. Estos medicamentos serán suministrados por la enfermera. Entretanto, el propietario se moverá libremente por la casa, tomando alimentos o bebidas de la nevera, descansando en los sofás y sillas o simplemente caminando. En ocasiones, el propietario podrá decidir ser él mismo quien tome la medicina que necesita. Además, podrá comunicar a la enfermera los cambios en su pauta de medicación añadiendo o eliminando medicamentos de forma dinámica, para ello disponemos de los planes.

Enfermera:

La enfermera es la encargada de suministrar las medicinas al propietario atendiendo a la pauta proporcionada. Debe hacerse con las medicinas necesarias y entregarlas en el momento exacto que deben ser tomadas. También requiere estar al corriente del control de existencias de la medicación aunque no se encargue de reponerla y, en caso de que sea el propietario quien decida ir en esa ocasión a por la medicación, comprobar que, efectivamente, ha consumido el medicamento.

Auxiliar:

Este agente, aunque no puede suministrar medicación directamente al propietario, participa controlando las existencias y la caducidad de los medicamentos. Cuando algún medicamento se va a caducar o quedan pocas existencias, se dirige a la zona de recogida para tomar los nuevos medicamentos y reponer el stock en el kit. Aunque no sea utilizado en este proyecto, el auxiliar también podría coger medicamentos para dárselos a la enfermera.

Objetos

Kit: Está situado en la cocina. En él se guardan los medicamentos y es accesible desde las casillas adyacentes ortogonalmente. OBJ4

Delivery: Corresponde a la puerta de entrada. Es utilizada por el agente auxiliar para recibir los pedidos de medicinas.

Charger: Zona destinada a la carga de robots. Se encuentra en la esquina superior derecha del hall. OBJ11.

Fridge: Contiene los alimentos y bebidas que el propietario tomará libremente. Es accesible desde las casillas adyacentes ortogonalmente.

Movimiento de los agentes y desplazamiento por el hogar

Un elemento común a todos los agentes es su capacidad de movimiento, para desplazarse siguen los siguientes pasos:

Como primer paso para desplazarse hacia el destino, se comprueba si el agente se encuentra en la misma habitación. En caso afirmativo, se desplaza hacia él sin más comprobaciones. En caso negativo, se encontrará la puerta más cercana que debe cruzar para llegar a destino y se moverá hacia ella. Los agentes se mueven casilla a casilla utilizando el método *moveTowards*.

Para controlar el recorrido del agente, se utiliza la distancia de Manhattan, que permite calcular el camino más corto a seguir entre dos puntos en una cuadrícula.

Los agentes se desplazan casilla por casilla siguiendo el camino más corto, aunque no podrán moverse por todas ellas libremente. Existen ciertas posiciones de la cuadrícula ocupadas por paredes, camas, sofás y otras piezas de mobiliario que no pueden atravesar. Esta condición será manejada por el método *canMoveTo*.

En el momento en el que se encuentra un obstáculo, cambia su ruta al destino y encuentra un nuevo camino. Para evitar que repita un recorrido ya realizado y vuelva

a tropezar con el mismo obstáculo, se almacenan en un HashMap *localizacionesVisitadas* todas las celdas por las que ha pasado el agente. Una vez el agente llega a su destino, el HashMap se vacía, preparado para moverse a un nuevo destino.

Con la implementación del movimiento de los agentes se cumple parte de los objetivos OBJ1 y OBJ6.

BDI

El modelo BDI (Belief-Desire-Intention) es una arquitectura de software para agentes inteligentes basada en conceptos de filosofía y psicología humana. Su nombre proviene de los tres componentes principales que rigen el comportamiento del agente:

- Beliefs (Creencias): Representan el conocimiento del agente sobre el mundo que le rodea, que puede ser incompleto o incluso erróneo.
- Desires (Deseos): Objetivos o metas que el agente quiere alcanzar.
- Intentions (Intenciones): Planes o acciones que el agente ha decidido llevar a cabo para lograr sus deseos y teniendo en cuenta sus creencias.

En el contexto de este proyecto, el modelo BDI se emplea para simular un comportamiento autónomo y racional de los agentes.

Limitaciones del proyecto

Se han asumido ciertas condiciones ideales en relación a situaciones que podrían darse en el mundo real por cuestiones de complejidad y tiempo.

- El agente propietario no tiene comportamiento no racional, es decir, no interfiere negativamente en el funcionamiento de la suministración de la medicación y aporta a la solución. No manipula los medicamentos, se deshace de cajas y demás acciones que podrían entorpecer el correcto desarrollo de la actividad por parte de la enfermera o el auxiliar.

- No se ha implementado una previsión del consumo energético. No se realiza una planificación anticipada del gasto de batería en función de las tareas pendientes o distancia a recorrer, sino que se actúa de forma reactiva. Cuando un robot alcanza un nivel de batería por debajo de un límite, se le ordena recargarse de forma preventiva, teniendo en cuenta que dicho umbral es suficiente para cubrir todas las acciones en un marco habitual.

Diagramas

Con el objetivo de representar de manera visual la estructura y funcionamiento del proyecto, se han elaborado una serie de diagramas que permiten comprender de manera más visual la organización y comportamiento de los agentes.

Funcionamiento

Funcionamiento general

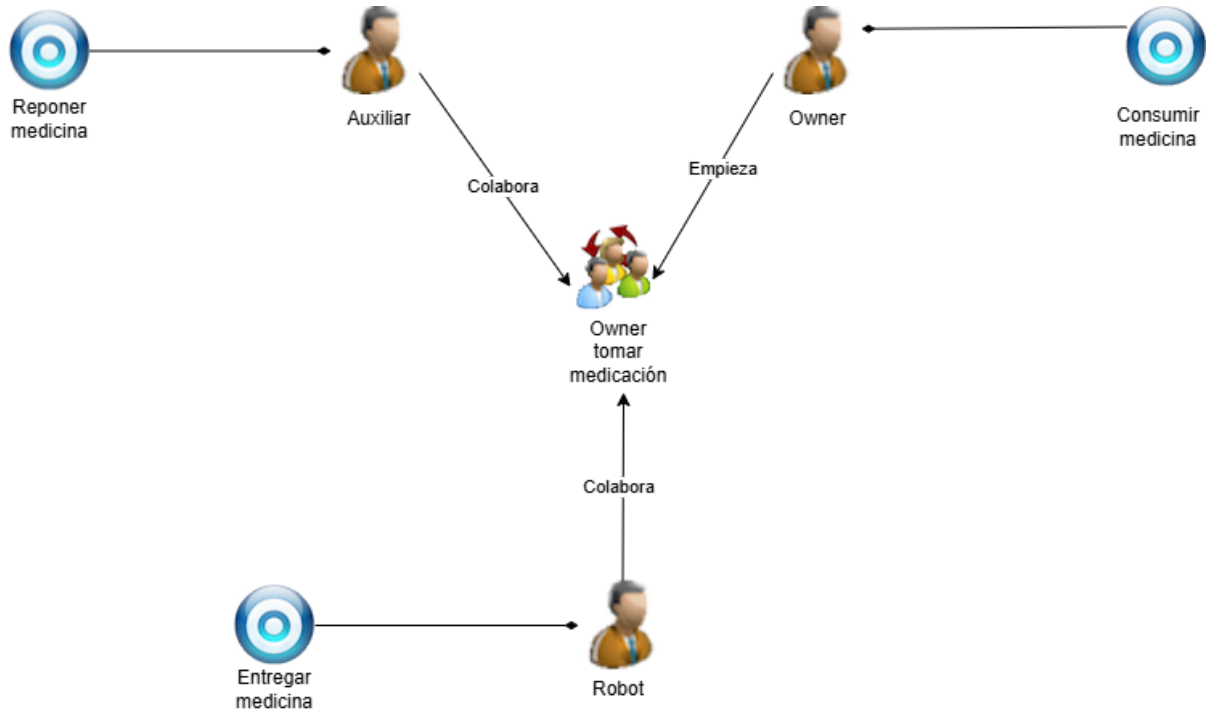

Diagrama 1. Diagrama de organización

En el Diagrama 1 (organización) se describe la jerarquía del sistema multiagente, representando cómo se relacionan entre sí el owner, el auxiliar y la enfermera, así como las responsabilidades de cada uno de ellos en el sistema.

La finalidad principal del programa es que el owner consuma su medicación. Para ello se verá ayudado por el auxiliar y la enfermera. La reposición del stock es realizada por el auxiliar. La enfermera será la encargada de suministrar la medicación al owner.

Inicio de la ejecución

En el comienzo del programa, el propietario comunica a la enfermera y el auxiliar tanto la prescripción de la medicación como la caducidad de dichos medicamentos (presentes en sus creencias). Después, ordena que la enfermera y el auxiliar comienzan su ejecución.

Cuando la enfermera y el auxiliar reciben junto con la orden para iniciar su actividad, las creencias necesarias para su ejecución las que controlarán la suministración de los distintos medicamentos en los momentos indicados (en caso de la enfermera) y el manejo de la caducidad de los mismos (en el caso del auxiliar). De igual manera, se iniciarán los planes que controlarán las existencias de cada uno de ellos y el manejo y gasto de la batería.

En este apartado, se cumple el objetivo OBJ2.

Entorno

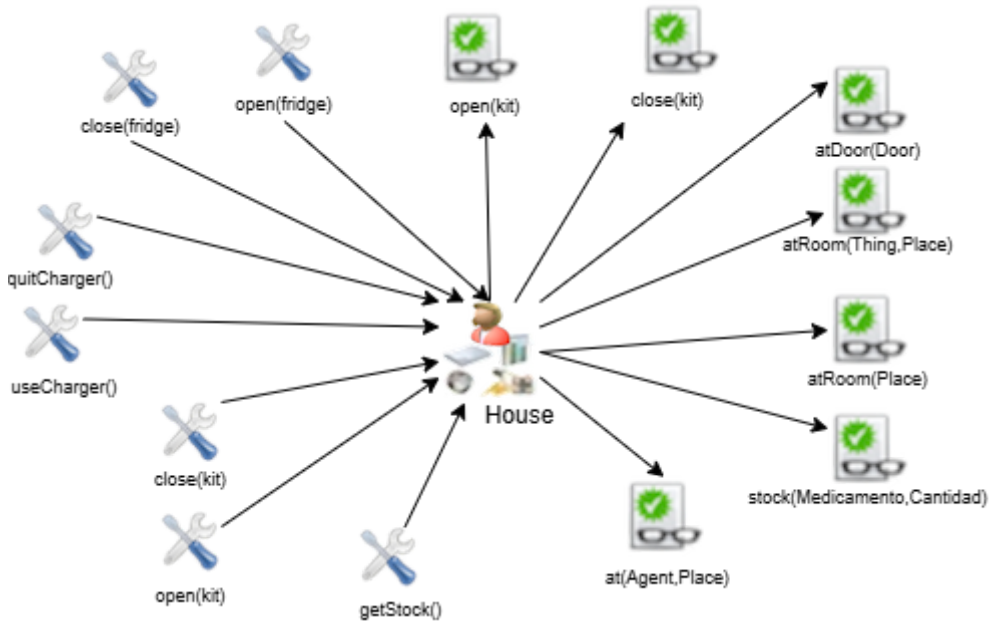

Diagrama 2. Diagrama del entorno

En el diagrama 2 (entorno) se representan los elementos del entorno con los que interactúa los agentes y las acciones disponibles en dicho entorno. Como ya se ha nombrado anteriormente, el funcionamiento está compuesto por tres agentes: Owner, Enfermera y Auxiliar, que precisan obtener del entorno las siguientes percepciones:

- *at(Agent,Place):* permite saber dónde se encuentra un agente. Por ejemplo, dependiendo de su ubicación en la casa, el agente percibirá *at(Agent, fridge)* o *at(Agent, owner);* o por supuesto, no percibirá nada en absoluto en caso de que no esté en ninguno de esos lugares.

- *atRoom(Thing,Place):* determina en qué habitación de la casa se encuentra *Thing*. *Thing* puede ser tanto un agente como un mueble de la casa.

- *atRoom(Place):* sirve para calcular en qué sala se encuentra el agente.

- *stock(Medicamento,Cantidad):* se emplea para conocer las existencias del medicamento indicado.

- *atDoor(Door):* indica la puerta en la que se encuentra el agente.

- *close(kit):* permite saber si el kit está cerrado.

- *open(kit):* determina si el kit está abierto.

Además de estas percepciones, los agentes precisan comunicarse con el entorno mediante una serie de acciones externas:

- *open(fridge):* el owner indica al entorno que abre el frigorífico.
- *close(fridge):* el owner indica al entorno que cierra el frigorífico.
- *open(kit):* el agente comunica al entorno que abre el kit.
- *close(kit):* el agente comunica al entorno que cierra el kit.
- *quitCharger():* la enfermera y auxiliar indican al entorno que abandonan la zona de carga.
- *useCharger():* la enfermera y auxiliar indican al entorno que acceden a la zona de carga.
- *getStock():* la enfermera y el auxiliar se informan sobre la cantidad de medicamentos que hay en el kit.

En este apartado se cumple el objetivo OBJ2.

Objetivos

Objetivo principal ideal.

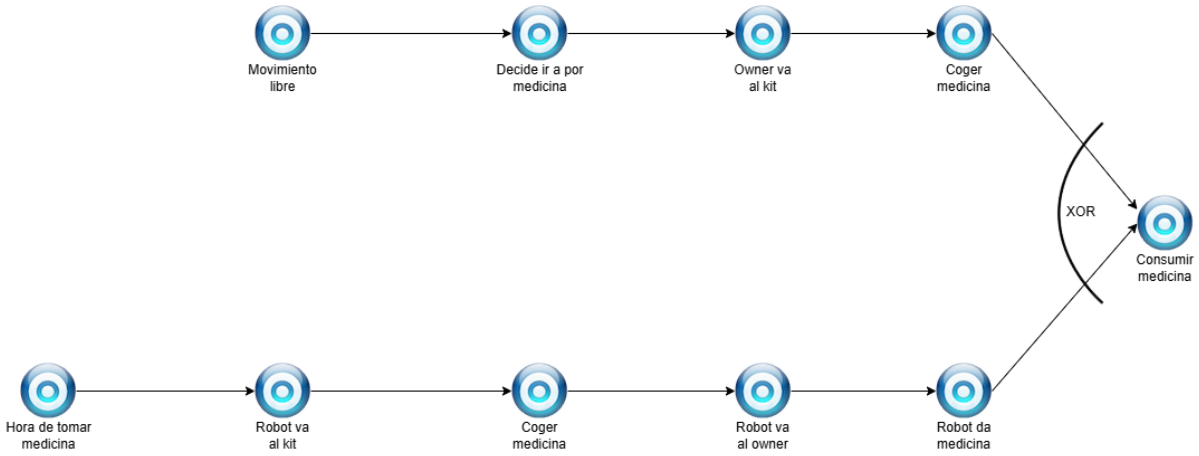

Diagrama 3. Diagrama de objetivos ideal.

Dividimos en dos diagramas, diagrama 3 y diagrama 4, la descripción de los objetivos del programa y los distintos pasos a seguir para lograrlos.

El objetivo principal (diagrama 3) puede ser llevado a cabo de dos maneras. En la primera, el owner se encarga de tomar la medicación por su cuenta y, en la segunda, es la enfermera quien se la suministra.

En el primer caso, el owner se encuentra libre recorriendo la casa, sentándose en el mobiliario hasta que decide ir a por la medicina de forma aleatoria. Cuando esto sucede se dirige hacia el kit y toma la medicina correspondiente. Finalmente, la consume.

Para que este caso no colapse el funcionamiento del programa o se produzca una suministración incorrecta, el owner envía un mensaje a la enfermera para avisarle de que es él quien irá por la medicación. Posteriormente, la enfermera comprobará que realmente ha consumido el medicamento.

En el segundo caso, la enfermera, atendiendo a las pautas de medicación proporcionadas por el owner, se dirige a por la medicación antes de la hora de suministración, recoge la necesaria y se desplaza hacia el owner. De esta manera, sin importar la distancia a la que el propietario se encuentre, la enfermera suministrará la medicación a la hora apropiada, esperando a su lado el tiempo restante necesario.

En el caso de que el owner vaya a por la medicación le envía a la enfermera un mensaje para que no vaya y viceversa. Y posteriormente la enfermera comprobará que se ha tomado la medicación correspondiente.

Objetivo alternativo

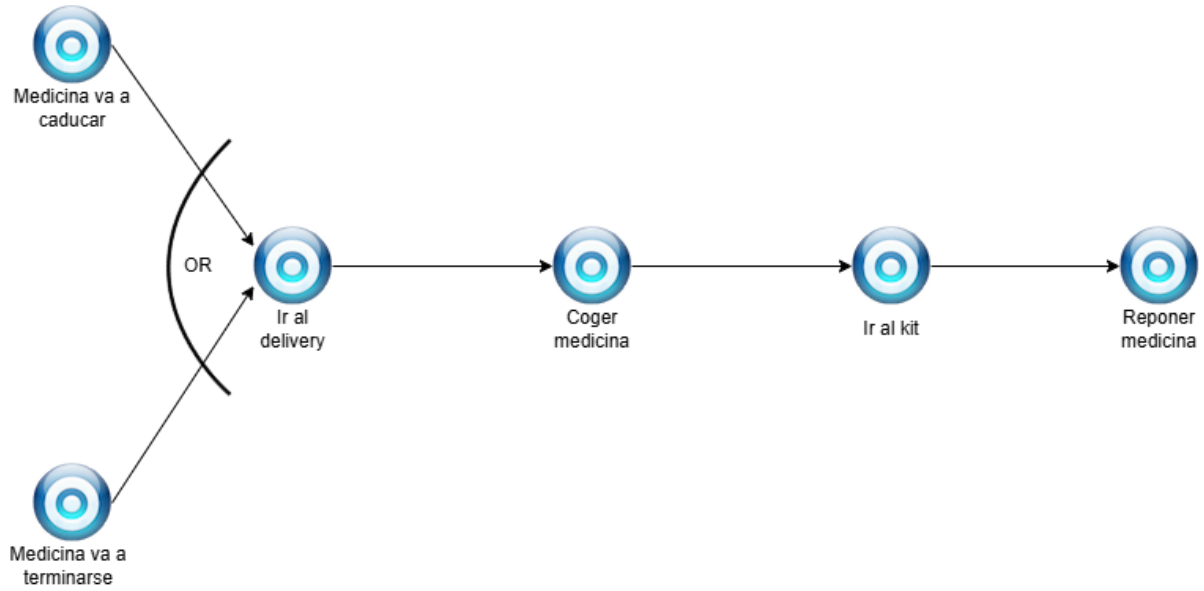

Diagrama 4. Diagrama de objetivo reponer medicina.

El diagrama anterior (diagrama 4) aborda la situación en la que un medicamento o bien caduque o se termine, debe ser repuesto. El objetivo principal del auxiliar es ocuparse de esta problemática. Cuando alguna medicación está próxima a caducar o quedan pocas unidades, se dirige al *delivery*, recoge las medicinas que necesita y las traslada al kit.

Lógica de los agentes

En este apartado se incluye una descripción más detallada del funcionamiento de cada agente valiéndonos diagramas de agente para cada uno de los presentes en el desarrollo del programa.
El comportamiento de todos los agentes se han dividido en dos diagramas para cada uno de ellos. En el primero se tratarán las creencias y planes principales de cada agente, los que se encuentran en la primera capa de ejecución y llaman a otros planes secundarios para cumplir sus objetivos. En el segundo, se tratarán tareas secundarias que soportan a dichos planes principales. La mayoría de los planes en los diagramas secundarios serán explicados de forma breve y concisa y sólo serán abordados con mayor profundidad aquellos más relevantes.

Owner

A continuación se muestra el diagrama de agente principal para el agente Owner:

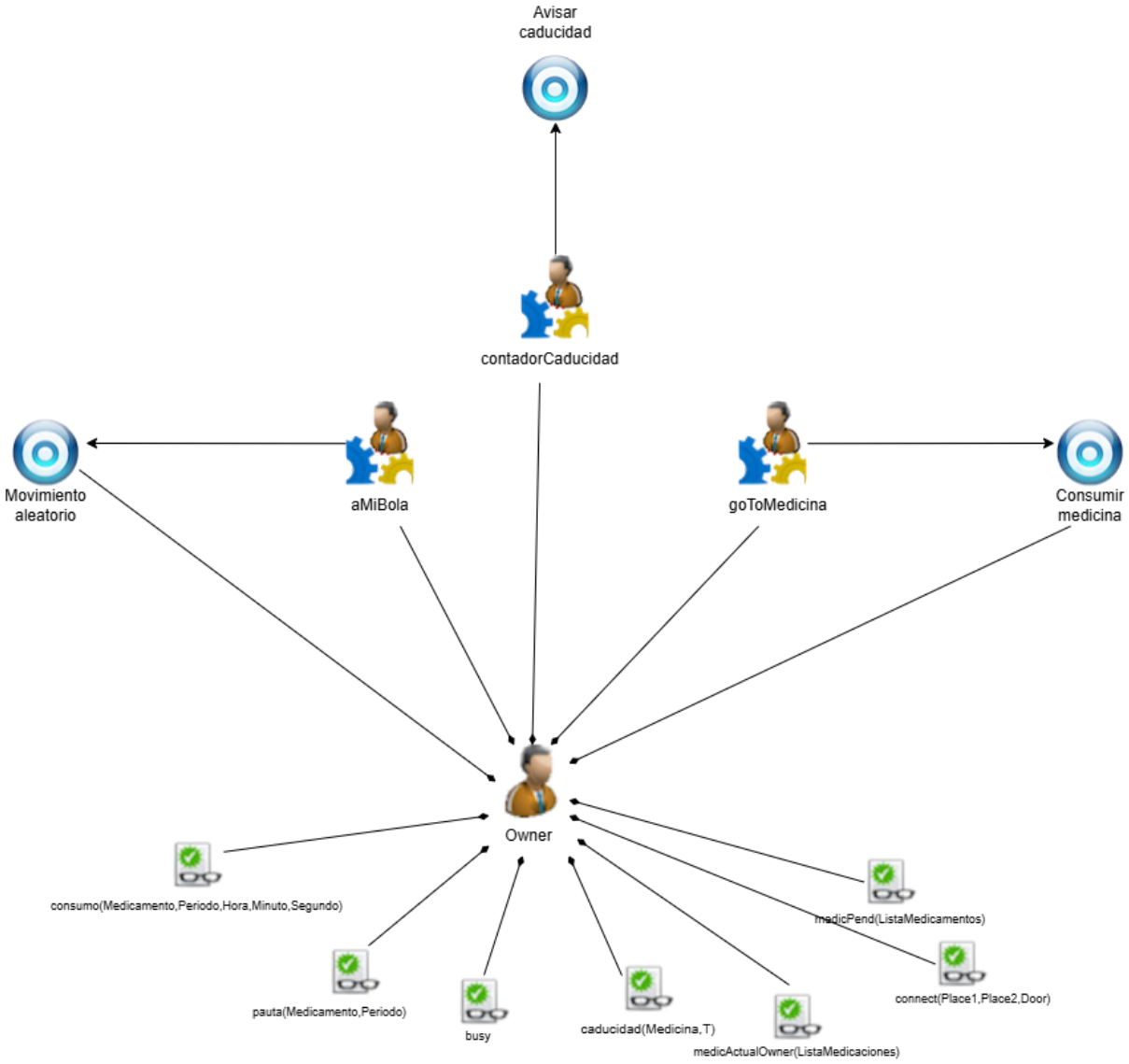

Diagrama 5. Diagrama de agente Owner

En el diagrama se observa que, aparte de las creencias presentes en el *environment* el *owner* contiene también las siguientes:

- *consumo(Medicamento,Periodo,Hora,Minuto,Segundo):* permite conocer el momento en el cual se debe consumir el medicamento referido.

- *pauta(Medicamento,Periodo):* creencia inicial donde se refleja la prescripción de los medicamentos.

- *busy*: indica que el agente se encuentra realizando una actividad.

- *caducidad(Medicamento,T):* permite saber cuanto falta la caducidad del medicamento.

- *medicActualOwner(ListaMedicaciones):* sirve para conocer las medicaciones que ha tomado el owner.

- *connect(Place1,Place2,Door):* indica qué puerta es la que conecta dos habitaciones.

- *medicPend(ListaMedicamentos):* guarda los medicamentos que debe consumir el owner.

El conjunto de todas estas creencias permite llevar a cabo las tareas a realizar por el owner:

***goToMedicina*:** es el plan al que se llama cuando el owner decide aleatoriamente ir a por las medicinas. Existen dos casos:

- Si el owner está ocupado, se elimina la acción que esté realizando (en nuestro caso sólo puede ser *sit*). Posteriormente, se ejecuta el plan *aPorMedicina*, el cual, de forma resumida:
    1. Se desplaza hasta el kit.
    2. Le comunica a la enfermera que va él mismo.
    3. Toma las medicinas contenidas en la lista *medPend* y llena una nueva lista *medActualOwner*.
    4. Envía *medActualOwner* a la enfermera para que confirme que las ha tomado.
- Si el owner no está ocupado, el owner va a por la medicina activando *aPorMedicina*

Después de ejecutar *aPorMedicina,* ejecuta el plan *aMiBola* de nuevo. El plan *aPorMedicina* y sus casos serán explicados en el diagrama secundario.

***contadorCaducidad*:** este plan se ejecuta constantemente para observar el tiempo que le falta a cada pauta para que se caduque, haciendo uso de las creencias presentes en el agente sobre la caducidad de los medicamentos, reduce segundo a segundo el contador de la caducidad. La creencia *pedidoReposicion* define si el medicamento se ha pedido al auxiliar para que lo reponga. Existen dos casos a considerar:

- Si el medicamento no está en el pedido de reposición, se observa si el tiempo para que la medicina se caduque es menor a 15 segundos.
    - En caso afirmativo:
        1. Se añade la creencia *pedidoReposicion(Medicina)* junto con la medicina requerida.
        2. El owner envía al auxiliar la orden de reponer dicha medicina al mismo tiempo que seguimos reduciendo el contador de caducidad.
        3. Una vez sea repuesta, se da el aviso de nuevo al owner y se reinicia el contador al tiempo inicial.
    - En caso negativo, simplemente se continúa reduciendo el contador.
- Si el medicamento ya se ha pedido y existe *pedidoReposicion*(Medicina), simplemente se continúa reduciendo el contador.

***aMiBola*:** el propietario se mueve libremente por la casa, ejecuta el plan *sit* (definido en el esqueleto entregado), sentándose en sillas y sofás. Después de una cantidad de tiempo aleatoria, se dirigirá él mismo a por la medicina. Habiendo tomado la medicina o no, este plan vuelve a iniciarse.

En este apartado se cumplen los objetivos OBJ1, OBJ2 Y OBJ3.

A continuación se muestra el diagrama de agente secundario para el agente Owner:

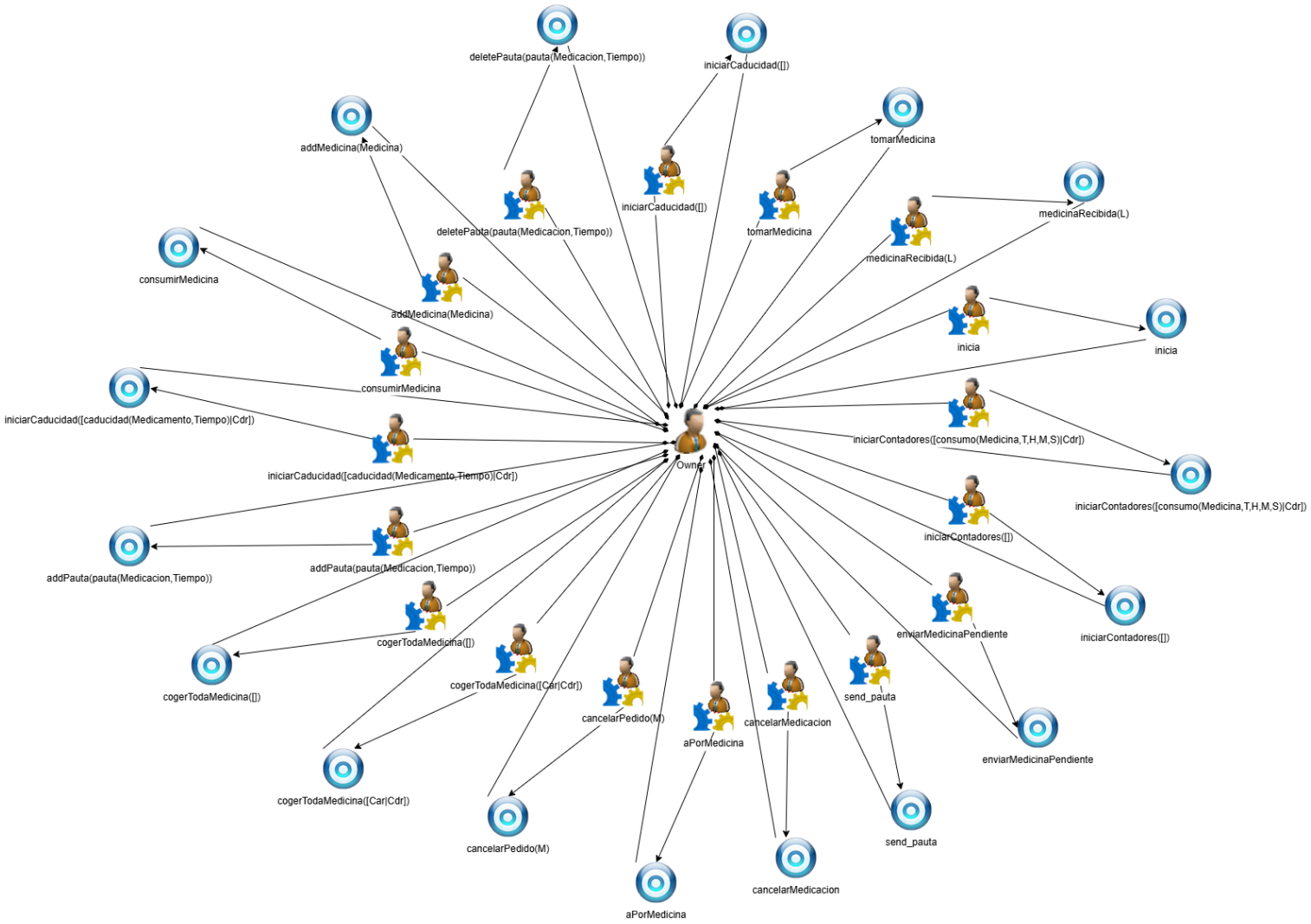

Diagrama 6. Diagrama de agente Owner secundario

aPorMedicina: este plan define las acciones que debe seguir el owner para coger la medicina él mismo, éstas son:

1. Se dirige hacia el kit de medicinas.
2. Una vez allí, se envía un mensaje *achive* a la enfermera de *cancelarMedicación*, marcándole que se ocupará él de hacerse con la medicación correspondiente.
3. Ejecuta el plan *cogerTodaMedicina* para tomar la medicina pendiente.
4. Ejecuta el plan *consumirMedicina* para tomarse la medicina.
5. Actualiza su creencia *medicPend* a una lista vacía y la comparte con la enfermera ejecutando *enviarMedicinaPendiente*.

inicia: marca el inicio de ejecución del agente. Crea creencias de consumo a partir de las pautas. Envía caducidades y consumos a auxiliar y enfermera. Inicia contadores de consumo y caducidad y manda comenzar los contadores de caducidad y existencias a la enfermera y auxiliar.

***sendPauta*:** envía consumo y caducidades a enfermera y auxiliar y ordena que inicien su ejecución.

***iniciarCaducidad*:** por cada medicina ejecuta el plan *contadorCaducidad*.

***contadorCaducidad*:** lleva la cuenta de la caducidad de los medicamentos. Cuando un contador de caducidad baja de 15, ordena al auxiliar que lo reponga.

***addPauta*:** añade una pauta nueva e informa de la misma la enfermera.

deletePauta: elimina una pauta y ordena a la enfermera que haga lo propio.

***cancelarMedicacion*:** este plan se ejecuta cuando la enfermera llega antes al kit. El owner deja de ir a por la medicina y ejecuta el plan *aMiBola*.

enviarMedicinaPendiente: envía la lista *medicPend* a la enfermera.

***cogerTodaMedicina*:** llena la lista *medicActualOwner* elemento a elemento, mostrando las medicinas que toma el owner.

***consumirMedicina*:** vacía la lista *medicActualOwner,* elemento a elemento mostrando las medicinas que está consumiendo el owner.

***medicinaRecibida*:** actualiza la lista medicPend con la lista *L*. Este plan es activado por un *achieve* a la enfermera cuando ésta ha cogido la medicina.

***addMedicina*:** añade la medicina Medicina a la lista *medicPend*.

***at*:** comprueba si el agente está en el destino, si no es así ejecuta *go*.

go: avanza hacia el destino, comportamiento del movimiento está en explicado en Introducción.

En este apartado se cumple el objetivo OBJ4.

Enfermera

A continuación, se muestra el diagrama de agente principal para el agente enfermera:

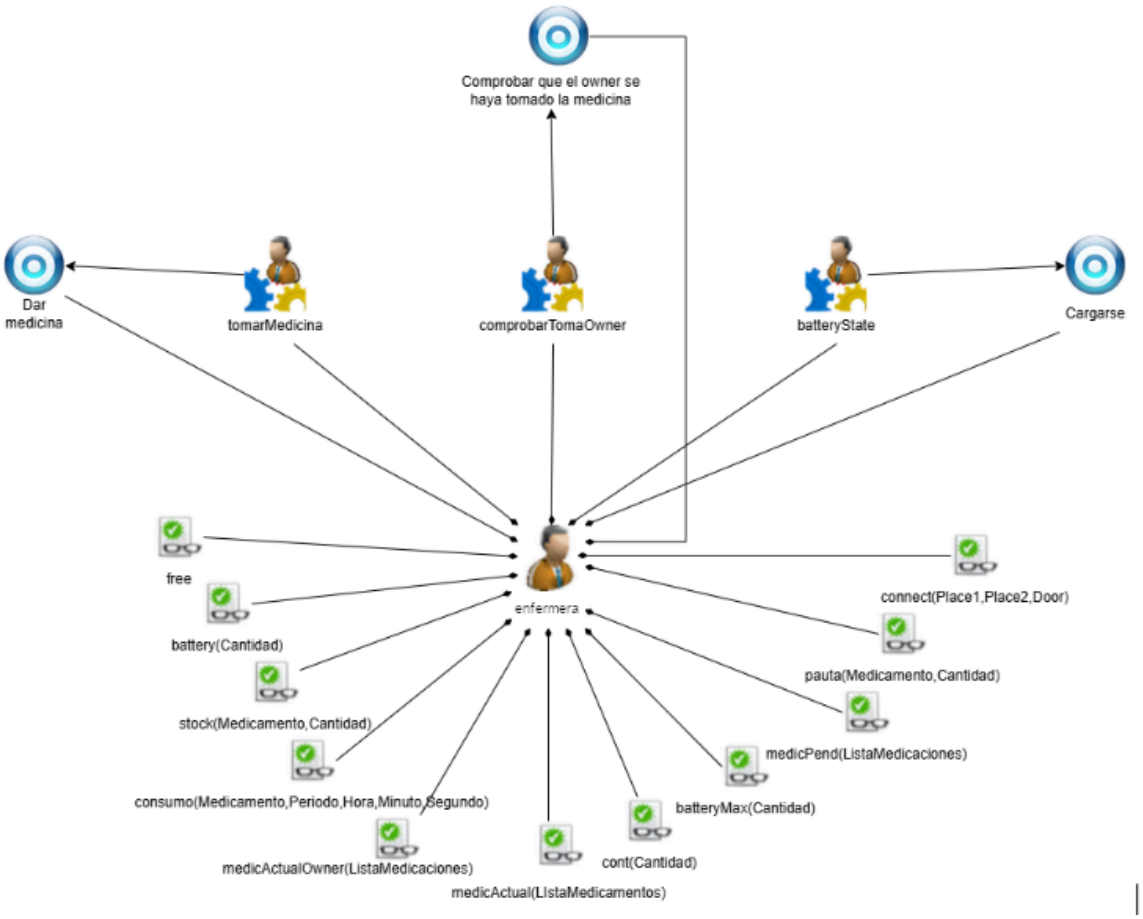

Diagrama 7. Diagrama de agente Enfermera

En el diagrama observamos que, aparte de las creencias presentes en el *environment*, la enfermera contiene también las siguientes:

- *free*: indica que la enfermera no está ocupada.
- *battery(Cantidad):* contiene la información relativa al nivel de batería restante.
- *batteryMax(Cantidad):* guarda el nivel máximo de la batería.
- *stock(Medicina,Cantidad):* contiene el último stock de medicinas existentes, la última vez que la enfermera comprobó el stock. Puede no corresponder con el actual.

- *consumo(Medicamento,Periodo,Hora,Minuto,Segundo):* indica a la enfermera la hora exacta a la que debe suministrar el medicamento referido al propietario.

- *medicActualOwner(ListaMedicamentos):* le permite consultar qué medicamentos han sido tomados por el propietario por su cuenta.

- *medicActual(ListaMedicamentos):* guarda una lista de los medicamentos que lleva la enfermera.

- *cont(Cantidad):* número de veces que la enfermera ha salido de la zona de carga antes que la batería alcance el nivel máximo.

- *medicPend(ListaMedicamentos):* contiene los medicamentos que deben suministrarse al propietario en el momento actual.

- *pauta(Medicamento,Periodo):* creencia inicial sobre la preescripción de la medicación.

- *connect(Place1,Place2,Door):* guarda qué puertas conectan las distintas habitaciones.

El conjunto de todas estas creencias permite llevar a cabo las tareas a realizar por el robot:

***tomarMedicina*:** este plan se está ejecutando constantemente para comprobar si el Owner debe tomarse el medicamento según sus pautas. Como hemos visto, la enfermera tiene creencias *consumo(Medicamento,Periodo,Hora,Minuto,Segundo)*, las cuales marcan el momento en el que el owner debe tomarse la medicación. Siete segundos antes de la hora marcada va a por la medicación. Se va con suficiente margen de tiempo para suministrar medicación a tiempo.
Existen dos casos a considerar:

- Cuando el owner debe tomar el medicamento en 7s: Una vez se llega a este momento, el medicamento se incluye en la lista *medicPend* y pasa a ejecutarse el plan *aPorMedicina*.
- Si no es así: el plan se mantiene vivo y seguimos esperando.

El plan *aPorMedicina* y sus distintos casos se explicarán en el diagrama secundario.

***comprobarTomaOwner*:** se activa cuando el owner ha sido el que ha ido a por la medicación. Éste envía *cancelarMedicacion* a la enfermera y, por extensión se ejecuta el plan *comprobarTomaOwner*. Cuando esto pasa la enfermera debe comprobar que se ha tomado las medicinas correspondientes (el owner al coger las medicinas, informa a la enfermera sobre la medicación que se va a tomar). Los casos que consideramos son los siguientes:

- Si está libre, la enfermera se dirige hacia el kit. Cuando llega, llama al plan *comprobarStock*, que será el encargado de comparar el stock presente en la base de creencias de la enfermera con el stock que le ofrece el entorno para determinar si el owner ha tomado la medicina.
- Si no está libre, mantiene vivo el plan y espera.

El plan *comprobarStock* y sus distintos casos se explicarán en el diagrama secundario.

***batteryState*:** comprueba constantemente el nivel de batería disponible. La enfermera consume batería al realizar ciertas acciones, en concreto:

- Movimiento por el hogar: consumo de 1 unidad de batería.
- Acciones como coger, dejar medicinas consumen 2 unidades de batería.

Este plan contiene los siguientes casos:

- La enfermera dispone de menos de unidades de batería y está libre:
    1. Se dirige hacia la zona *waitCharger*. En esta posición comprobará si el auxiliar está haciendo uso de la zona de carga.
        a. Si el auxiliar lo está utilizando, espera a que lo libere.
        b. Si está libre se coloca sobre él.
    2. Una vez sobre el cargador, se mantiene allí durante 3 segundos (1 hora en nuestro mundo.
    3. Finalmente, abandona el cargador y se desplaza a la zona *afterChargerRobot*. Esta zona existe para que si la enfermera utiliza la zona de carga y no tiene nada más que hacer, no permanezca ocupando el cargador en caso de que sea requerido por el auxiliar.
- La enfermera dispone de menos de 100 unidades y no está libre, espera 1 segundo y vuelve a comprobarlo.
- En el caso de que no tenga un nivel de batería de menos de 100, simplemente se mantiene vivo el plan.

En este apartado se cumplen los objetivos OBJ2, OBJ4 OBJ7, OBJ10, OBJ11.

A continuación se muestra el diagrama de agente secundario para el agente enfermera:

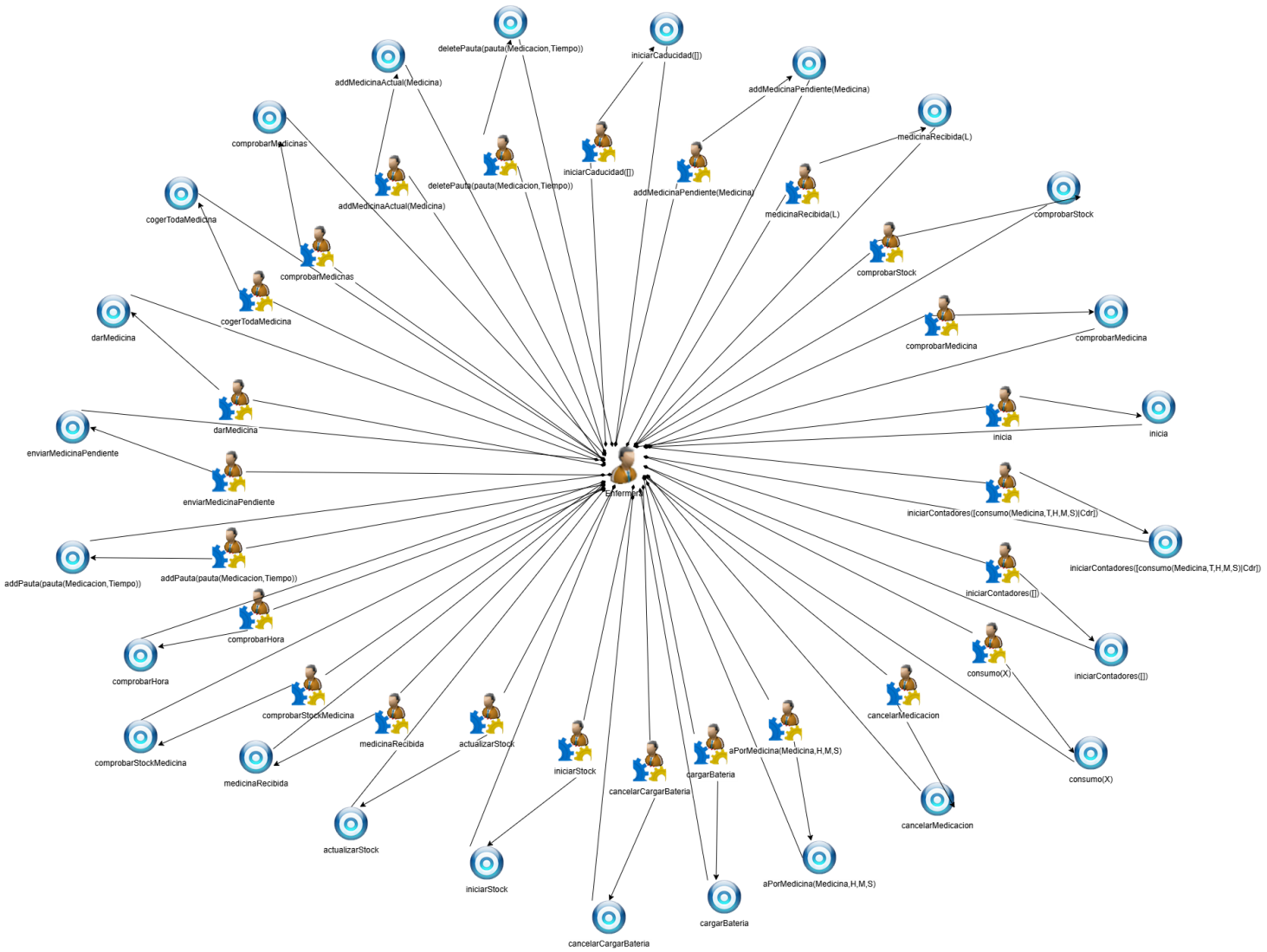

Diagrama 8. Diagrama de agente Enfermera secundario

aPorMedicina: este plan define las acciones que debe tomar la enfermera para coger la medicina y suministrarla al owner. Los casos a considerar son:

- Si la enfermera está libre realiza las acciones principales:
    1. Va hacia el kit de medicinas
    2. Una vez allí, comunica al owner que ha llegado ella primero.
    3. Ejecuta el plan *cogerMedicina* y toma toda la medicina, actualiza su lista de *medicPend* y se la envía al owner.
    4. Ejecuta el plan comprobar hora, el cual se desgrana en: dirigirse hacia el propietario y esperar a la hora exacta en la que tiene que ser suministrada la medicina.
    5. Suministra la medicina.

- Si no está libre y está ejecutando el plan *comprobarTomaOwner,* se llama al plan *aPorMedicina* para ejecutarlo cuando termine de comprobar.
- Si no está libre, asumimos que está llevando medicina al owner y simplemente mostramos que se ha añadido el medicamento a lista *medicPend*.

Las creencias *consumo(Medicamento,Periodo,Hora,Minuto,Segundo)* son actualizadas y preparadas para la siguiente suministración.

***comprobarStock*:** comprueba si el owner se ha tomado la medicación. Existen dos casos:

- No se dispone de *medicActualOwner:* esperamos a que el owner comunique qué medicación se ha tomado.
- Se dispone de *medicActualOwner*: en este caso recuperamos el stock que realmente existe en el entorno. Por cada medicina que tenemos en *medicActualOwner*, comparamos si el stock real es menor que la creencia que tenia la enfermera. Si es así, el owner ha tomado la medicación. Por el contrario, se avisa de que no se ha tomado la medicina.

***inicia*:** marca el inicio de ejecución del agente. Añade las creencias y ejecuta todos los planes necesarios para la ejecución del agente.

***iniciarContadores*:** añade creencias de consumo a la enfermera.

***addPauta*:** añade dinámicamente una pauta definida por el usuario.

***deletePauta*:** elimina dinámicamente una pauta.

***iniciarStock*:** se añaden creencias de stock de medicinas.

***consumo*:** reduce la batería del agente.

***comprobarCargadorLibre*:** comprueba si el auxiliar está en el punto de carga.

***cargarBateria*:** toma 3 segundos para cargar la batería al completo.

***cancelarCargarBateria*:** cancela la carga y aumenta su contador. Cuando esta situación se da 3 veces, afecta a la salud de la batería reduciendo su carga máxima.

***actualizarStock*:** se actualiza la creencia de *stockActual* de la enfermera.

***comprobarHora*:** la enfermera se desplaza hacia el owner. Mientras no sea la hora perfecta para tomarse la medicina, la enfermera sigue al owner.

***enviarMedicinaPendiente*:** informa al owner sobre la medicación pendiente actual que debe tomar.

***darMedicina*:** da la medicina al owner.

***addMedicinaPendiente*:** añade la medicina a la lista de medicación pendiente.

***addMedicinaActual*:** añade la medicina a la lista de medicación que ya lleva consigo el robot.

***cogerTodaMedicina*:** coge las medicinas del kit.

***cancelarMedicacion*:** cancela el *aPorMedicina* y va a comprobar las medicinas.

***comprobarMedicinas*:** por cada medicina que ha cogido el owner comprobamos la reducción en el stock.

***comprobarMedicina*:** asi el stock ha bajado, significa que el owner se lo ha tomado. Si no es así, se avisa que no se ha tomado la medicación.

***comprobarStockMedicina*:** nos avisa si el owner no se ha tomado la medicación.

***medicinaRecibida*:** actualiza la lista de medicación pendiente cuando el owner informa sobre las medicinas que ha tomado.

***at*:** comprueba si el agente está en el destino, si no es así ejecuta *go*.

***go*:** avanza hacia el destino. El comportamiento del movimiento está explicado en la Introducción.

Cabe destacar que los planes que gastan energía en la enfermera disponen de planes alternativos en caso de que la batería se agote.

Auxiliar

A continuación, se muestra el diagrama de agente principal para el agente Auxiliar:

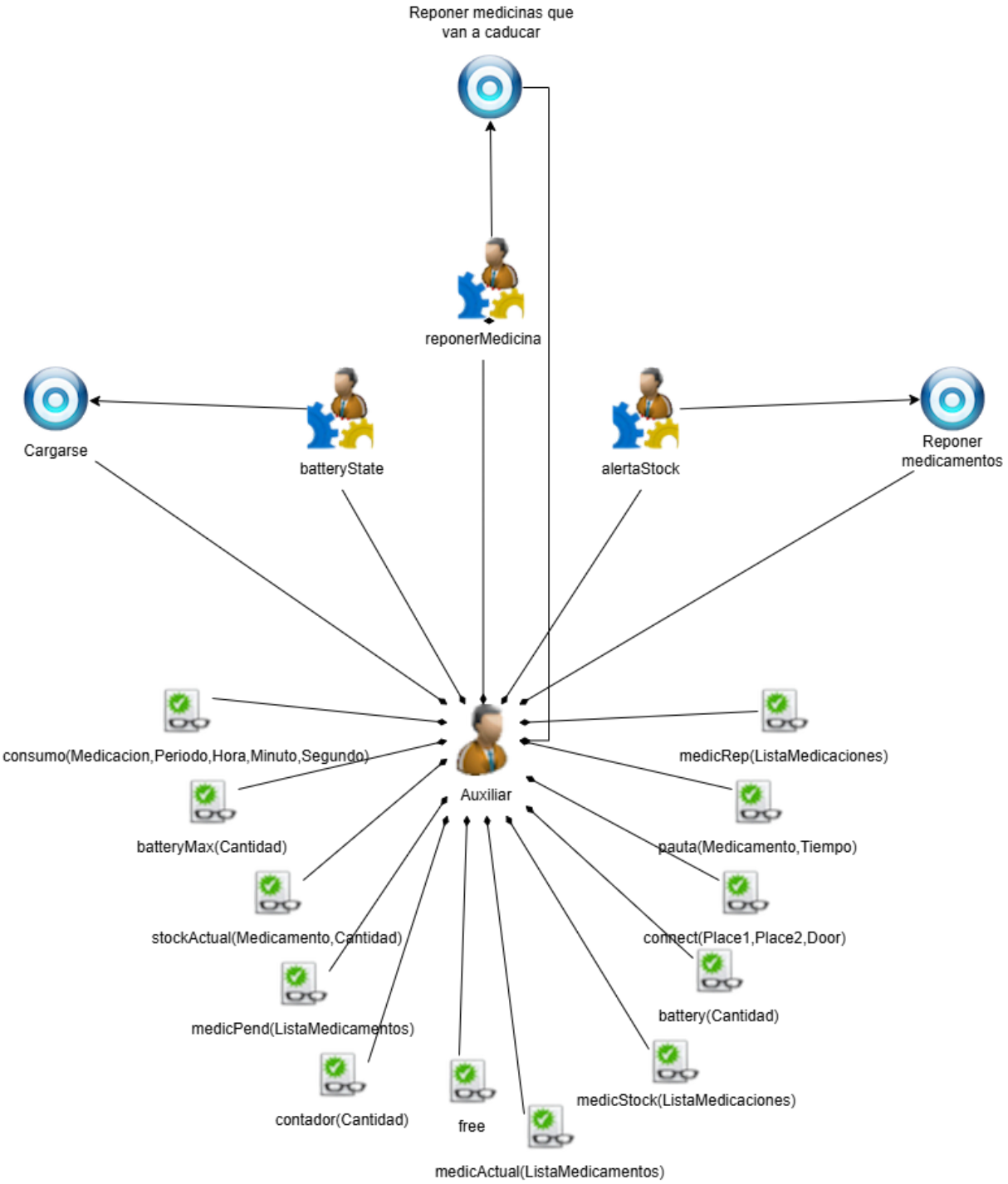

Diagrama 9. Diagrama de agente Auxiliar

En el diagrama observamos que, aparte de las creencias presentes en el *environment*, el auxiliar contiene también las siguientes:

- *free*: indica que el auxiliar no está ocupado.
- *battery(Cantidad):* contiene la información relativa al nivel de batería restante.
- *batteryMax(Cantidad):* guarda el nivel máximo de la batería.
- *consumo(Medicamento,Periodo,Hora,Minuto,Segundo):* indica al auxiliar la hora exacta a la que la enfermera debe suministrar el medicamento referido al propietario.
- *medicActual(ListaMedicamentos):* guarda una lista de los medicamentos que lleva el auxiliar.
- *contador(Cantidad):* número de veces que el auxiliar ha salido de la zona de carga antes de cargarse al máximo.
- *medicPend(ListaMedicamentos):* contiene los medicamentos que deben suministrarse al propietario en el momento actual.
- *pauta(Medicamento,Periodo):* creencia inicial sobre la preescripción de la medicación.
- *connect(Place1,Place2,Door):* guarda qué puertas conectan las distintas habitaciones.
- *stockActual(Medicamento,Cantidad):* indica el número de medicamentos disponibles actualmente.
- *medicStock(ListaMedicaciones):* contiene una lista de los medicamentos que necesitan reponer existencias.
- *medicRep(ListaMedicaciones):* contiene una lista de los medicamentos caducados que deben cambiarse.

El conjunto de todas estas creencias permite llevar a cabo las tareas a realizar por el auxiliar:

***reponerMedicina*:** este plan es utilizado para la reposición de medicinas por motivo de caducidad. El que activa este plan es el owner, el cual manda al auxiliar que una medicina se está caducando.
En cuanto al owner, utilizando la creencia sobre la caducidad de las medicinas, cuando el contador de alguna de ellas baje de 15s, comunica al auxiliar qué medicina que está próxima a la caducidad.
Se tienen en cuenta los siguientes casos:

- Si la lista *medicRep* está vacía, se añade Medicina a la lista *medicRep* y se empieza a ejecutar el plan *reponerMedicinas,* el cual de forma resumida:
    1. Se desplaza hasta el delivery.
    2. Coge las medicinas de la lista *medicRep*.
    3. Las deja en el kit y tira las que van a caducar.
- Si la lista *medicRep* no está vacía, se añade la Medicina a la lista *medicRep*.

El plan *reponerMedicinas* se explica con más profundidad en el diagrama secundario.

De esta manera, en ningún caso se suministrará un medicamento caducado al disponer de tiempo suficiente para realizar el cambio antes de que eso ocurra.

***alertaStock*:** plan que controla el stock de cada uno de los medicamentos de forma constante. La lista *medicStock* se va llenando con los medicamentos que tienen poco stock. Se consideran dos casos para este plan:

- *medicStock* este vacía: significa que, de momento, ninguna medicina necesita ser repuesta todavía. Se recupera el stock actual y se ejecuta el plan *recorrerStock* que, resumidamente:
    1. Recorre todos los medicamentos.
    2. Si algún medicamento tiene un stock menor o igual a dos unidades, se añade a la lista *medicStock*.
    3. Cuando ha terminado, si hay alguna medicina dentro de *medicStock*, se llama al plan *hayQueRecoger*, que se encargará de reponer medicinas.
- Si no está vacío, se mantiene vivo el plan porque se asume que el auxiliar está yendo a reponer medicinas.

El plan *hayQueRecoger* se explica con más profundidad en el diagrama secundario

***batteryState*:** el auxiliar dispone del mismo plan que la enfermera por lo que no sé tratará en detalle y sólo se mencionará que, en el caso de este agente, recibirá la orden de cargarse cuando la batería sea menor a 75 unidades.

En este apartado, se cumplen los objetivos OBJ5, OBJ8, OBJ9, OBJ10, OBJ11.

Seguidamente, se muestra el diagrama de agente secundario para el agente Auxiliar:

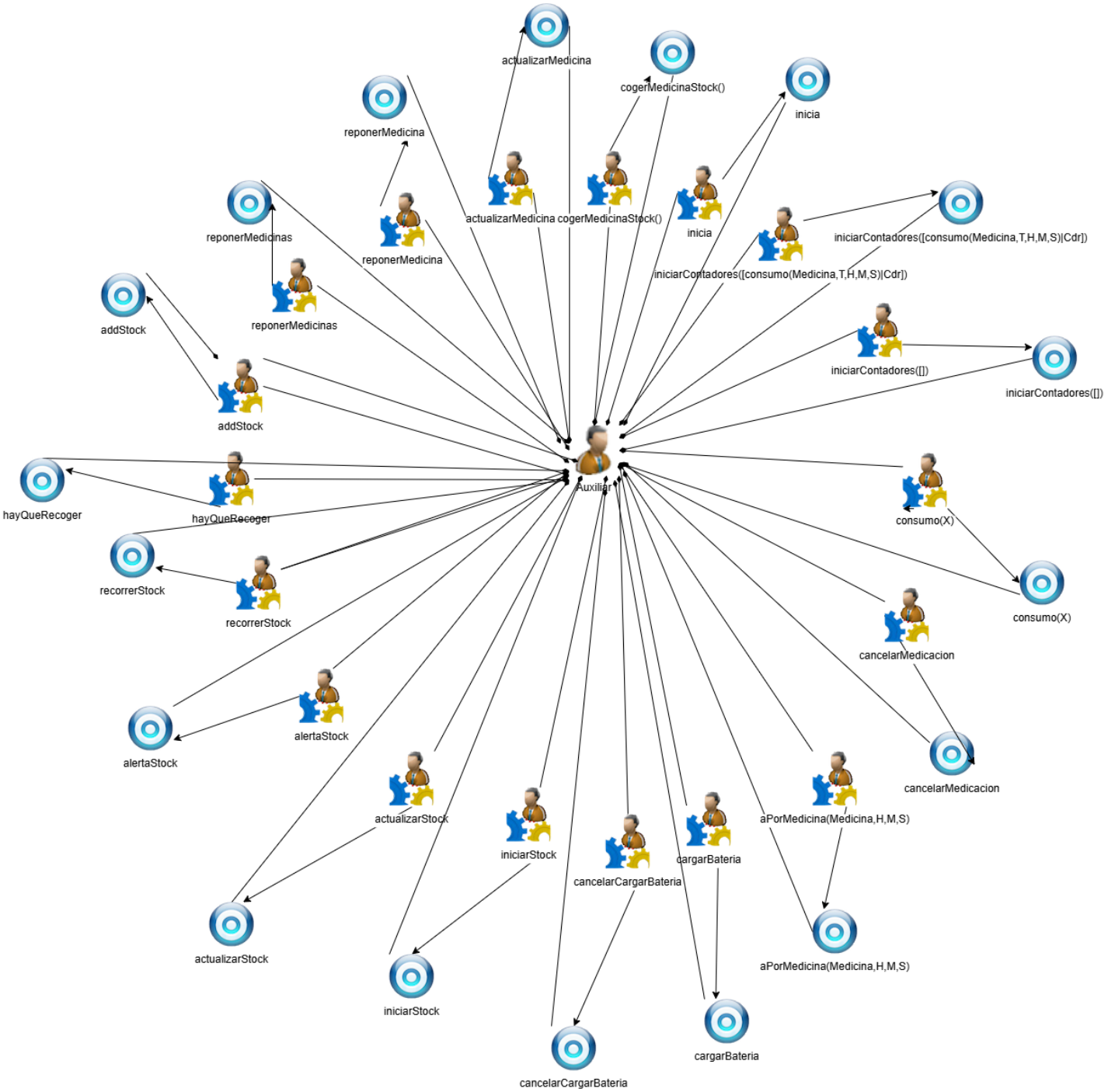

Diagrama 10. Diagrama de agente Auxiliar secundario

A continuación se explican los planes secundarios presentes dentro del auxiliar:

***hayQueRecoger*:** plan encargado de ejecutar las acciones para reponer el stock en el kit. Se consideran varios casos:

- Si el auxiliar está libre, se ejecutan las acciones principales:
    1. Se dirige hacia la zona *delivery*.
    2. Coge las medicinas de la lista *medicStock* ejecutando *cogerMedicinaStock*.
    3. Se dirige hacia el kit de medicinas.
    4. Una vez allí, añade a kit las medicinas que ha recogido.
    5. Actualiza sus creencias de stock.
- Si no está libre espera y mantiene vivo el plan.
- Si *medicStock* esta vacío, es decir, no hay ninguna medicina por recoger, el auxiliar pasa a estar libre.

***inicia*:** marca el inicio de ejecución del agente. Añade las creencias y ejecuta todos los planes necesarios para la ejecución del agente.

***inicarStock*:** obtiene el stock actual y actualiza la creencia *stockActual*.
consumo: Reduce la batería del agente.

***comprobarCargadorLibre*:** verifica si el cargador está siendo usado por la enfermera.

***cargarBateria*:** toma 3 segundos para cargar la batería al completo.

***cancelarCargarBateria*:** cancela la carga y aumenta su contador. Cuando esta situación se da 3 veces, afecta a la salud de la batería reduciendo su carga máxima.

***actualizarStock*:** actualiza *stockActual* con la información del entorno.

***recorrerStock*:** añade medicamentos con un stock por debajo del umbral a la lista *medicStock*.

***cogerMedicinaStock*:** recoge medicamentos desde el delivery y los registra en *medicActual*.

***addStock*:** añade medicamentos al kit desde *medicActual*.

***reponerMedicinas*:** recolecta y repone medicinas con fecha de caducidad próxima.

***cogerMedicina*:** toma las medicinas que van a caducar.

***actualizarMedicina*:** reemplaza las medicinas caducadas y actualiza las creencias en los agentes.

***at*:** comprueba si el agente está en el destino, si no es así ejecuta *go*.

***go*:** avanza hacia el destino. El comportamiento del movimiento está explicado en Introducción, Movimiento de los agentes y desplazamiento por el hogar.

Cabe destacar que los planes que gastan energía en auxiliar disponen de planes alternativos en caso de que la batería se agote.

Interacción entre agentes

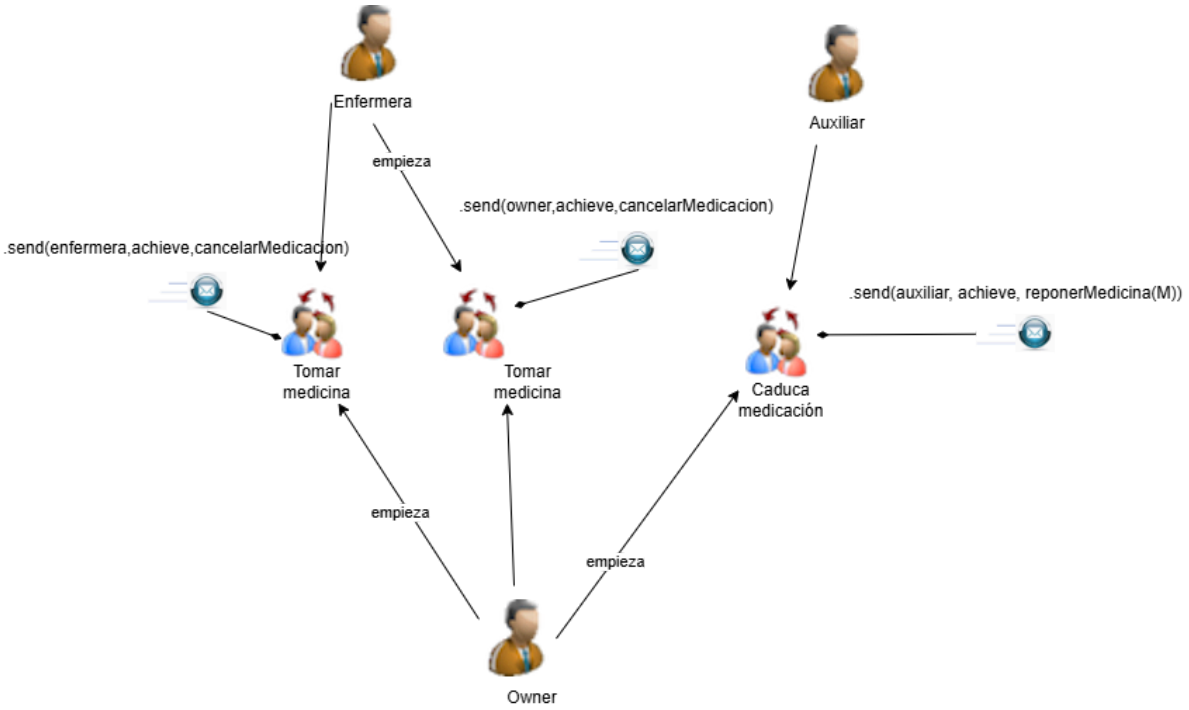

Diagrama 11. Diagrama de interacción

En este diagrama se describe el funcionamiento principal y la comunicación entre agentes para llevar a cabo las tareas. Como podemos observar, tanto owner y la enfermera pueden empezar el objetivo de tomar medicina. Si es el owner quien comienza, envía un mensaje a la enfermera para que no lo haga y viceversa. También, el owner avisa al auxiliar cuando un medicamento va a caducar para que lo tire y lo reponga.

Tabla de trazabilidad

Una tabla de trazabilidad es una tabla que permite rastrear y vincular los requisitos de un proyecto con sus correspondientes elementos de diseño y desarrollo Sirve para asegurarse de que todos los objetivos del proyecto han sido correctamente implementados y verificados.
Con el fin de sintetizar los objetivos a cumplir por el proyecto, se incluye una tabla de trazabilidad. A continuación, se proporciona un listado de objetivos establecidos y alcanzados en la memoria, que han sido referenciados a lo largo de la misma.

ÍNDICE	**OBJETIVO**
OBJ1	El agente owner tiene necesidad de tomar sus medicaciones pautadas (deberéis elegir al menos 5 medicamentos y la pauta de toma de cada uno de ellas). El agente owner puede moverse libremente por la casa y descansar en zonas adecuadas para ello (silla, sofá, cama, ...)
OBJ2	La medicación puede ser proporcionada por el agente enfermera o por el propio agente owner. Para que la enfermera pueda servir los medicamentos al owner, este debe indicar la pauta de las medicaciones a la enfermera nada más crearse.
OBJ3	En caso de haber tomado la medicación, el owner debe indicarlo a la enfermera.
OBJ4	Con el paso del tiempo, el owner modificará la pauta de sus medicaciones y añadirá y/o eliminará algunas de ellas. Las medicaciones deberán guardarse en sitios adecuados y accesibles tanto para el owner como para la enfermera.
OBJ5	La medicación estará disponible en cantidad siempre suficiente para poder ser administrada.
OBJ6	El agente enfermera debe poder moverse libremente por la casa sorteando los objetos que en ella se encuentra, de una habitación a otra, buscando al owner para entregarle la medicación cuando proceda.
OBJ7	Si la enfermera recibe indicación de que alguna medicación se ha tomado, debe comprobar que ha sido así.
OBJ8	Se incorpora un nuevo agente (auxiliar) para ayudar a la enfermera en algunas tareas.

OBJ9	Las medicinas tienen fecha de caducidad y un límite que cuando se alcanza obliga a reponer/cambiar las medicinas (auxiliar) para reponerlas el agente deberá ir a una zona de entrega, escogida a tal efecto, y "*recoger*" las nuevas dosis de medicina entregadas.
OBJ10	Los agentes auxiliar y enfermera (como autómatas que son) precisan energía para desplazarse y realizar el resto de acciones de las que son capaces. Inicialmente la cantidad de energía de la que disponen coincide con el número de celdas de vuestro entorno (*Casa*) y se decrementará en una unidad con cada acción que realizan en el entorno, entre ellos o con el owner.
OBJ11	Para realizar la recarga los robots contarán con un único punto de recarga (*incluir en el entorno*) y para recargarse de manera completa consumirán el equivalente a una hora de tiempo. Pueden optar por estar menos tiempo en el punto de recarga, pero la realización de esa opción en más de 3 ocasiones reducirá la carga máxima de energía en un 5% para ese agente.