DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

# MODELO DE RECONOCIMIENTO DE BEBÉ A BORDO EN AUTO CERRADO Y APAGADO

# T E S I S

PRESENTADO POR:

**ING. FABIOLA SALAS DÍAZ**

COMO REQUISITO PARCIAL PARA

OBTENER EL GRADO DE:

**MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN**

**DIRECTOR DE TESIS:** M.C. ANA LUISA MILLÁN CASTRO
**CO-DIRECTOR DE TESIS:** DR. JUAN PABLO SOTO BARRERA

**HERMOSILLO, SONORA, MÉXICO** **AGOSTO DE 2017**

SECCIÓN: DIV. EST. POS. E INV.
No. OFICIO: DEPI/204/17.
ASUNTO: AUTORIZACIÓN DE IMPRESIÓN DE TESIS.

09 de Agosto de 2017

**C. FABIOLA SALAS DÍAZ,**
**P R E S E N T E.**

Por este conducto, y en virtud de haber concluido la revisión del trabajo de tesis que lleva por nombre **"MODELO DE RECONOCIMIENTO DE BEBÉ A BORDO EN AUTO CERRADO Y APAGADO"** que presenta para el examen de grado de la MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN, y habiéndola encontrado satisfactoria, nos permitimos comunicarle que se autoriza la impresión del mismo a efecto de que proceda el trámite de obtención de grado.

Deseándole éxito en su vida profesional, quedo de usted.

A T E N T A M E N T E

M.C. ANA LUISA MILLÁN CASTRO
DIRECTORA

DR. JUAN PABLO SOTO BARRERA
CO-DIRECTOR

DRA. MARÍA TRINIDAD SERNA ENCINAS
SECRETARIA

M.C. CÉSAR ENRIQUE ROSE GÓMEZ
VOCAL

M.C.O. ROSA IRENE SÁNCHEZ FERMÍN
JEFA DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

S.E.P.

INSTITUTO TECNOLÓGICO
DE HERMOSILLO
DIVISIÓN DE ESTUDIOS
DE POSGRADO

RISF/momv*

# Resumen

En los Estados Unidos, las muertes de bebés por golpes de calor asociadas a automóviles supera ya las 700 en los últimos 19 años. Esta problemática ha cobrado la muerte de algunos niños en México y en diversos países del mundo, ya que se conjuga el riesgo ambiental -un automóvil cerrado, en pocos minutos, puede convertirse en un horno-, con la incapacidad de los menores de regular la temperatura corporal. A pesar de ser un suceso recurrente, en México no hay una estadística oficial que registre este tipo de decesos.

Al estudiar los motivos que propician este tipo de incidentes, el más frecuente y alarmante es el olvido no intencional del menor, el segundo es el olvido intencional de éste. Por lo tanto, este trabajo de tesis se centra en ofrecer un modelo para detectar la presencia de un bebé en un automóvil apagado y prevenir que el conductor se aleje del vehículo sin retirar al menor. Si el menor se deja en el automóvil, se da aviso a los servicios de emergencia.

La propuesta se presenta a través del diseño de un modelo multi-agente donde colaboran cinco agentes para monitorizar las condiciones del automóvil, detectar presencia de un bebé e informar la presencia al responsable o a la entidad correspondiente si el abandono se presentase. Se utilizó para ello la metodología INGENIAS, determinando a través de los distintos modelos, los agentes, sus roles e interacciones con el entorno y la visión general a través del modelo de organización.

Finalmente se implementó el modelo utilizando la plataforma JADE para simular el comportamiento de los agentes. Analizando las interacciones resultantes, se validó la comunicación entre agentes y las secuencias de acción esperadas para cada caso definido.

# Abstract

In the United States, hyperthermia baby deaths related to vehicles exceed 700 in the past 19 years. This problem has led to the death of several children in Mexico and other countries around the world, because of the environmental risk -a closed car in a few minutes can become an oven- and also the inability of children to regulate body temperature. Despite being a recurring event, in Mexico there is no official statistics to support this information.

When studying the reasons for this type of incident, the most frequent and alarming is the unintentional left-behind, the second is the intentional left-behind of the child. This project then focuses on providing a model to detect the presence of a baby in a car with engine off and prevent the driver from leaving the vehicle without removing the child. If the child is abandoned in the car by the person carrying the child, emergency services are given notice.

The proposal is presented through the design of a multi-agent model where five agents collaborate to monitor the conditions of the car, detect the presence of a baby and report the presence to the responsible or the corresponding entity if the abandonment occurs. The INGENIAS methodology was used for this purpose, determining through the different models, the agents, their roles and interactions with the environment, and the general vision through the organizational model.

Finally, the model was implemented using the JADE platform to simulate agent behavior. Analyzing the resulting interactions, we validated the communication between agents and the expected sequences of action for each case defined.

# Agradecimientos

**A mis directores de Tesis**

A la Mtra. Ana Luisa Millán Castro por ser mi guía durante este tiempo, por su calidez como persona y por transmitirme la confianza necesaria para crecer con mi proyecto de tesis.

Al Dr. Juan Pablo Soto Barrera, por su disposición para resolver mis dudas y encauzar mis ideas con tanta claridad. A ambos mis infinitas gracias por su tiempo y su invaluable ayuda para llevar a término este trabajo.

**A mis asesores de Tesis**

Dra. María Trinidad Serna Encinas y Mtro. César Enrique Rose Gómez, por dedicar un espacio a revisar y retroalimentar este trabajo. Sobre todo, agradezco sus afectuosas atenciones que facilitaron los procesos académicos y administrativos.

**A mis compañeros**

Por ser maestros y amigos, por regalarme el ver la vida a través de sus ojos. Gracias por hacer más fácil esta travesía.

De manera especial, agradezco a mi esposo Luis Guillermo por su paciencia, comprensión y por obsequiarme alegrías cuando las situaciones me sobrepasaron.

Al Consejo Nacional de Ciencia y Tecnología por la contribución para el desarrollo de esta investigación a través de la beca 261077/582947

**A mis hijos, mi fortaleza,**

**luz de mi corazón.**

# Índice general

# Índice de figuras

# Lista de Tablas

# Capítulo 1

# Introducción

La facilidad con la que un bebé o un niño pequeño a bordo de un auto puede pasar desapercibido es inquietante. A edades tan tempranas, los pequeños concilian el sueño con rapidez, por lo que en un paseo breve en automóvil podrían dormirse y pasar inadvertidos, y así, en un descuido ser olvidado en el auto.

El dejar en situación de abandono a un bebé en un automóvil apagado, aún en un lapso corto de tiempo, representa una situación de alto riesgo. Además de lo peligroso que puede ser por su etapa de desarrollo, el riesgo ambiental es muy elevado, pues un automóvil cerrado, en pocos minutos, puede convertirse en un horno.

Con base en estas consideraciones, el proyecto partirá de una recopilación de información sobre el problema y las soluciones que se han propuesto, organizando la información en los subtemas que lo componen. Utilizando como referente los trabajos relacionados, se estudiarán mecanismos alternativos para presentar una solución que permita atacar la problemática a través de una propuesta innovadora.

## 1.1. Antecedentes

El mecanismo de regulación de temperatura en los niños pequeños, no es equiparable al de los adultos, esta característica les vuelve mucho más vulnerables en ambientes cálidos. Investigaciones al respecto establecen que la temperatura corpo-

ral de un niño es significativamente más alta que la de un adulto, basándose en las mismas condiciones ambientales para ambos. Esto nos lleva a establecer que en temperaturas moderadas a altas, los niños menores de 5 años sean considerados como población de alto riesgo de sufrir hipertermia [1].

La Real Academia de la Lengua Española define la hipertermia como "el aumento patológico de la temperatura del cuerpo" [2]. Para efectos de esta investigación, la hipertermia es la causa por la cual los niños pierden la vida cuando son olvidados por sus padres en un automóvil, en condiciones donde la temperatura se convierte en un factor crítico.

Por otra parte, si se toma en cuenta las temperaturas que pueden alcanzar los automóviles cerrados expuestos al sol, aun sin considerar la vulnerabilidad de los bebés a las altas temperaturas, su sola presencia en un automóvil con estas condiciones por un espacio corto de tiempo, se expone en la mayoría de los casos, a una muerte segura.

Un automóvil al sol puede llegar a calentarse 10°C en los primeros 10 minutos y seguir incrementándose en una media de 0.3°C por minuto, cuando la temperatura ambiente oscila entre 22°C y 35°C. En este rango de temperaturas el incremento se estabilizaría en una media de una hora alcanzando registros de hasta 23°C por encima de la temperatura ambiente [3].

Existen diferencias en los resultados entre los distintos estudios para medir el incremento de temperaturas con respecto al tiempo en un automóvil cerrado expuesto al sol, éstas se deben al hecho de que el sensor reciba directamente los rayos del sol o esté posicionado a la sombra. El 80 % del incremento en la temperatura con el sensor a la sombra ocurre en 30 minutos, contra un incremento del 75 % en los primeros cinco minutos si el sensor está posicionado al sol. Esta observación es importante puesto que, si de salvar una vida se trata, la consideración deberá ser la menos favorable para reducir los tiempos de respuesta [4].

En medios de comunicación de distintos países como España, Italia, Estados Unidos y México, se han hecho evidentes, casos de muertes por hipertermia en niños pequeños que fueron abandonados en un automóvil. Los motivos pueden ser desde el

no intencional olvido del niño, hasta el dejarlo intencionalmente dentro del vehículo. También se han registrado casos de niños que mueren al estar jugando dentro de un automóvil, sin supervisión o conocimiento de las personas mayores a cargo.

Al respecto, las estadísticas muestran que en los últimos 19 años se han presentado 700 muertes por hipertermia en niños olvidados en un auto en los Estados Unidos. En la Fig. 3.1 se puede observar la estadística anual de muertes por hipertermia para los Estados Unidos desde 1998 al presente.

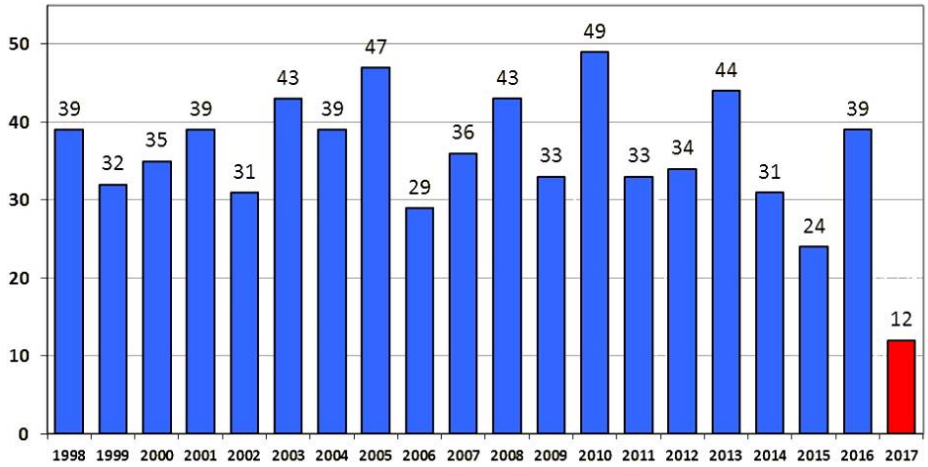



Figura 1.1: Muertes por hipertermia asociadas a automóviles en los Estados Unidos.

La estadística contempla una media anual de 39 niños, de los cuales el 74 % son menores de 2 años y el 22 % entre 3 y 5 años. Asimismo, categoriza los motivos de abandono encabezando la lista el olvido no intencional con un 54 % de los casos, niños jugando sin supervisión con un 28 %, abandono intencional 17 % y un 1 % atribuido a circunstancias desconocidas [3].

Tanto las edades de los niños pequeños como la causa del olvido por parte de sus responsables nos permiten observar que la mayoría de los casos de muerte por hipertermia se presentan en bebés que por un descuido fueron olvidados en un automóvil. El olvido no intencional, por difícil que parezca, ocurre y se han incrementado los casos justamente por regulaciones encaminadas a proteger la vida de niños y bebés.

El uso de sillas de bebé que aumentan su seguridad pero limitan su movilidad en un automóvil, pero sobre todo el hecho de llevarlos obligadamente en el asiento trasero, los vuelve propensos a ser olvidados. Estas precauciones se estandarizaron

en la década de 1990 y son de suma importancia pues protegen a los bebés de las bolsas de aire, sin embargo pueden asociarse al incremento desde esa fecha de los olvidos de pequeños en un automóvil [5].

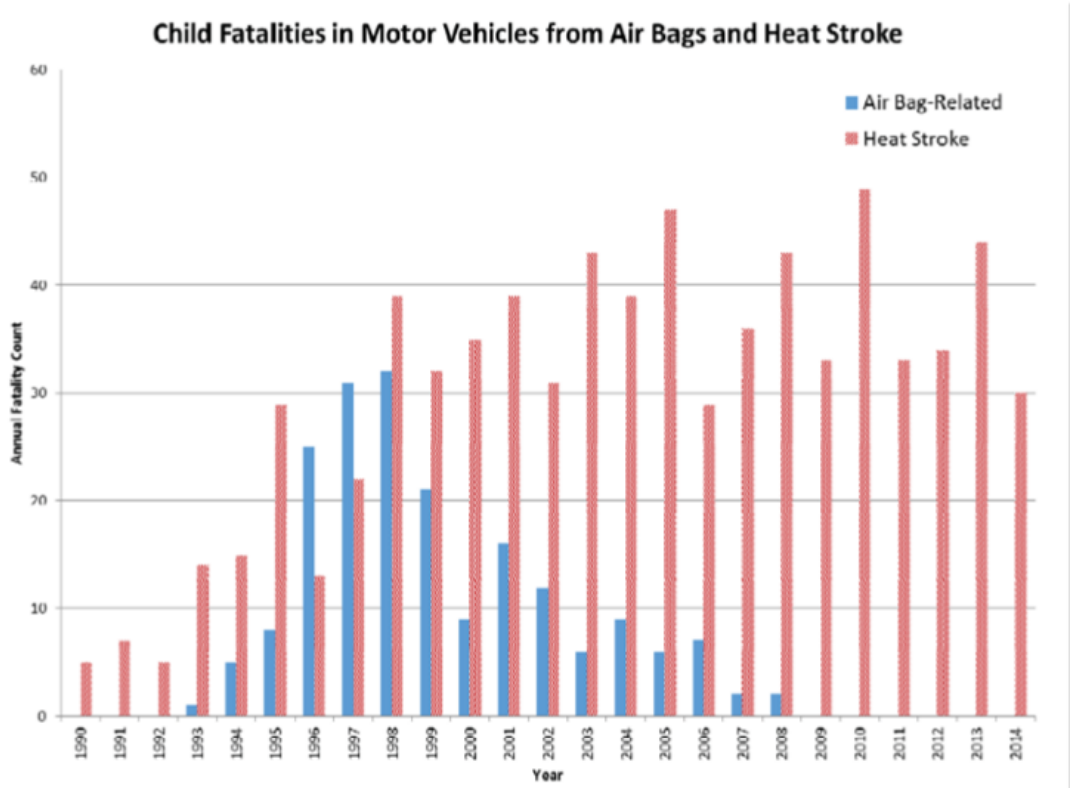



Figura 1.2: Muertes por hipertermia y por bolsa de aire asociadas a automóviles en los Estados Unidos. [6]

En este sentido, el departamento de Transporte de los Estados Unidos (NHTSA, por sus siglas en inglés), presentó una comparativa en la que se aprecia cómo las muertes de menores por bolsa de aire se redujeron a partir de la regulación que prohíbe llevar menores en el asiento delantero. Asimismo, se puede observar una tendencia al aumento en las muertes por hipertermia en este período, reforzando la idea de que llevar a los menores en el asiento trasero les vuelve más propensos a ser olvidados, (Fig. 1.2).

A pesar de lo difícil que pueda parecer olvidar a un pequeño, sobre todo si se trata de un hijo, es una situación que sucede. Al respecto, David Diamon, científico del Hospital de Veteranos en Tampa Florida, explica que las personas tienen sistemas de memoria opuestos [5]. La parte primitiva del cerebro dirige sus hábitos, permitiendo

por poner un ejemplo, conducir a casa sin pensar en las consecuencias de cada movimiento. El segundo sistema, ubicado en regiones del cerebro más avanzadas, es responsable de los planes a corto plazo, como por ejemplo comprar leche de camino a casa. El sistema de hábitos primitivos es más poderoso, pudiendo de esta forma dejar de lado los planes a corto plazo.

Por otra parte, la hipertermia se caracteriza por una temperatura superior a los 40°C, con déficits neurológicos como el delirio, convulsiones, coma y por último la muerte. Una vez que se alcanzan los 41.6°C, sobrevivir es poco probable [7]. Es importante considerar que ha habido casos de recuperación después de un cuadro de hipertermia, sin embargo es difícil y las secuelas son irreversibles.

Estudios demuestran que durante la ola de calor en Chicago en 1995, se produjeron alrededor de 60 casos de hipertermia. Al momento de presentarse dicho fenómeno, todos los afectados exhibieron disfunción en múltiples órganos, además de discapacidades neurológicas, insuficiencia renal de moderada a severa, coagulación intravascular diseminada y síndrome de afección respiratoria aguda. La tasa de mortalidad de esa muestra fue del 20 %, registrando los sobrevivientes secuelas de las que no se recuperaron y que provocaron en un lapso de un año la muerte del 28 % de los sobrevivientes [8] .

Las muertes por hipertermia originadas por las circunstancias antes expuestas, han alarmado a diversos sectores sociales de los cuales se han desprendido una variedad de recomendaciones para prevenir los casos de hipertermia dentro de los automóviles.

Asimismo, se han desarrollado distintas soluciones para atacar esta problemática. Las propuestas valoran distintos dispositivos de alerta que van desde una alerta visual o recordatorio, hasta dispositivos más completos que detectan la presencia de un bebé mediante sensores. Se han valorado otras soluciones integradas a los vehículos que por desgracia no han podido llevarse al mercado [9].

A pesar de no existir estadísticas oficiales en México respecto a muertes por hipertermia por abandono en un automóvil, los medios de comunicación nacionales han expuesto en los últimos años, al menos una muerte anual en niños de dos años

o menos, olvidados en un automóvil.

En Sonora, Baja California e incluso en la Ciudad de México donde las temperaturas no son tan extremas, se han registrado lamentables sucesos de esta índole. El más reciente en nuestro país sucedió en julio del presente año en la ciudad de Mexicali, como consecuencia de un olvido de su madre [10–12].

Estas cifras nos llevan a valorar el desarrollo de un sistema inteligente que actúe de forma preventiva al evitar cerrar el auto con un bebé. Si lo anterior no evitara el dejar al bebé en el auto, el sistema deberá identificar las condiciones expuestas anteriormente, comunicarlas a la persona o entidad pertinente y conservar de esta forma la integridad física del menor.

## 1.2. Definición del problema

Las temperaturas elevadas representan un alto riesgo en la población infantil, que incluso puede derivar en la muerte. La temperatura de un niño se incrementa de tres a cinco veces más rápido que la de un adulto, lo que constituye un mayor riesgo si consideramos un niño pequeño olvidado en un auto [13].

Aunque existen distintas razones por las que pueda ocurrir un evento de este tipo, el mayor registro de casos se debe al olvido del pequeño en el auto por parte de la persona a cargo de éste. Siguiendo este orden tendríamos a los pequeños que suben a jugar al auto y se encierran en él por accidente; finalmente aquellos casos en los que el bebé es dejado en el auto con plena consciencia por parte de sus cuidadores [3].

Esta situación no es ajena a nuestro país donde casos de esta índole han ocupado titulares en distintos medios de comunicación, con las consecuentes recomendaciones para evitar un evento de este tipo.

**Preguntas de investigación:**

- ¿Cómo detectar que hay un bebé a bordo y el auto está cerrado y apagado?
- ¿Cómo impedir el cierre de la puerta de un auto?
- ¿Cómo indicar o alarmar el peligro?
- ¿Qué dispositivos utilizar?

**Lo anterior nos lleva al siguiente planteamiento del problema:**

¿Qué funcionalidades debe tener el modelo para un sistema inteligente que impida el cierre de puerta de un automóvil apagado y que emita señales de alarma al detectar a un bebé a bordo con un registro de temperaturas elevadas?

## 1.3. Objetivos

**Objetivo general:** Modelar un sistema inteligente que impida el cierre de puerta del conductor al reconocer un bebé a bordo en un auto apagado y que emita una alarma después de algún tiempo de espera, al reconocer que el bebé aún continúa en el auto y se detectan temperaturas elevadas.

**Objetivos específicos:**

- Revisar y analizar a profundidad los temas que abarcan el marco teórico del proyecto de tesis.
- Determinar si el automóvil está apagado.
- Analizar mecanismos para impedir el cierre de puertas.
- Analizar qué tipo de sensores utilizar para detectar un bebé dentro de un auto.
- Determinar si la temperatura del auto sobrepasa los límites de tolerancia de un bebé.
- Definir protocolo de comunicación de la situación del bebé.

- Modelar un sistema de monitoreo constante que permita detectar bebés olvidados dentro de un automóvil.

## 1.4. Justificación

Modelar un sistema inteligente que evite olvidar a un bebé en un automóvil, sienta las bases para el desarrollo de un sistema que beneficia un sector vulnerable. El beneficio es tangible, ya que será capaz de comunicar una situación de alto riesgo de una persona que aún no es capaz de hacerlo por sí misma, representando un fuerte impacto social. Las aplicaciones que se le pudiera dar a dicho modelo pueden extenderse a otros sectores también vulnerables, como personas de la tercera edad, individuos en situación de discapacidad crónica o incluso mascotas, de ahí la trascendencia de este trabajo.

Finalmente, tomando en cuenta las características particulares del proyecto, su desarrollo se adhiere al área de conocimiento de sus investigadores y se cuenta con los recursos necesarios para llevarlo a cabo en tiempo y forma.

## 1.5. Limitaciones y alcances

El modelado del sistema inteligente se fundamenta en la aplicación de la técnica Poka Yoke para evitar errores inadvertidos, cuya función será imposibilitar el cierre de la puerta del conductor al detectar un bebé en el automóvil. Asimismo presentar señales de alarma al detectar la presencia de un bebé dentro del automóvil [14].

Las pruebas de funcionalidad del modelo se llevarán a cabo en el Laboratorio de Sistemas Inteligentes de la Maestría en Ciencias de la Computación, perteneciente a la División de Estudios de Posgrado e Investigación del Tecnológico Nacional de México/Instituto Tecnológico de Hermosillo.

## 1.6. Metodología

La metodología a utilizar para el desarrollo del proyecto se compone de tres fases: La primera se refiere a los fundamentos teóricos del tema, la segunda se enfoca en el análisis y el diseño del modelo del sistema inteligente y una última fase evalúa los resultados obtenidos con la implementación de una simulación que valide su funcionamiento. Una visión general de éstas se presenta a continuación:

La fase uno es una recopilación teórica de los temas derivados del proyecto, abordando en primera instancia las tendencias en cómputo pervasivo y los aportes de los sistemas multi-agentes a éstos. Acotando los resultados que se puedan obtener respecto al tema anterior, se enfoca la investigación al estudio de las arquitecturas, metodologías y plataformas de agentes. Finalmente se valoran los tipos de alarmas a utilizar para lanzar una advertencia en caso de situación de riesgo de muerte por golpe de calor en un bebé.

El hacer un análisis a profundidad de los temas que inciden en esta investigación, permitirá afinar la propuesta de solución y respaldar en todo caso el desarrollo del modelo del sistema inteligente expuesto.

La fase dos se compone de dos partes, el análisis y el diseño del modelo multi-agente. Con respecto al análisis se fundamenta en una síntesis del conocimiento obtenido acotado al ámbito del modelado del sistema inteligente, esto derivará en las especificaciones de los componentes del mismo así como la definición de las herramientas a utilizar para su integración. En la parte de diseño se modelan las especificaciones del sistema utilizando las arquitecturas 4+1 vistas e INGENIAS.

Finalmente en la fase tres o fase de implementación, se llevan a cabo las pruebas de funcionalidad. Dichas pruebas servirán para validar que se obtengan los resultados esperados; es decir, que el modelo se comporte según las expectativas definidas.

## 1.7. Organización de la tesis

El capítulo 2 presenta el estado del arte haciendo un análisis de los temas que fundamentan esta investigación, así como los trabajos relacionados para prevenir las muertes por hipertermia. Se parte de una visión general del cómputo pervasivo, para ahondar luego en los agentes software y los sistemas multi-agente, que por sus funcionalidades se utilizarán para el desarrollo del modelo.

El capítulo 3 describe el análisis y diseño del sistema utilizando para ello la metodología de INGENIAS. La representación de cada uno de los modelos que componen esta metodología presentan a detalle la arquitectura multi-agente a implementar, así como la secuencia lógica de la solución propuesta. Como parte del diseño de la solución, se incluye también un modelo de notificaciones basado en el de CANoE [15], cuyo fin es el de componer el mensaje de alerta y generar la notificación correspondiente.

La implementación del modelo multi-agente se desarrolla en el capítulo 4, exponiendo las características particulares de cada agente. En este apartado se describe la forma en la que se establecen las interacciones entre los agentes para alcanzar sus objetivos, las tareas que cada uno debe de llevar a cabo y cómo el modelo cumple con los objetivos establecidos para su operación.

Las pruebas de software se contemplan en el capítulo 5 y están dirigidas a validar la funcionalidad de la simulación. En éste se consideran los casos de prueba necesarios para cubrir las distintas posibilidades generadas por la especificación del modelo, validando de esta forma lo apropiado de los resultados obtenidos. Asimismo, en esta sección se presentan los resultados obtenidos de las interacciones entre los agentes.

Finalmente, el capítulo 6 presenta las conclusiones obtenidas durante el desarrollo de este trabajo. Plantea también los trabajos a futuro que aportarían nuevas fortalezas a lo ya propuesto.

# Capítulo 2

# Estado del Arte

## 2.1. Introducción

El presente capítulo parte de una visión generalizada sobre el cómputo pervasivo, en esta área se han conseguido importantes logros gracias a los avances tecnológicos en redes y capacidad de procesamiento, por mencionar los ejes principales de esta vertiente. Tan es así que lo que hace unos años se planteaba como tendencia, es ahora tangible y se enfrenta a nuevos retos en su evolución, el principal de ellos la seguridad.

En esta amplia gama de tecnologías que engloba el cómputo pervasivo se encuentran los agentes software. Éstos son una parte fundamental en el desarrollo de entornos inteligentes y son la base de la propuesta de esta tesis, en particular los sistemas multi-agente, por ello se considera dentro de este capítulo un apartado al respecto.

Finalmente, en la última sección, se abordarán los trabajos relacionados, clasificándolos de acuerdo al tipo de solución que ofrecen: soluciones sin capacidad de detección, soluciones integradas en el vehículo y soluciones mediante sensores en el asiento del bebé.

## 2.2. Cómputo pervasivo

A principios de la década de los 90's, Weiser, comparte su visión sobre las tendencias y la evolución tecnológica, dando lugar al concepto de cómputo ubicuo. En esencia se refiere a entornos saturados de dispositivos inteligentes interconectados, es decir medios con capacidades de cómputo y de comunicación interactuando con seres humanos de forma casi imperceptible [16].

Para ofrecer un acercamiento a lo que sería una tecnología integrada en la vida de los seres humanos y de uso imperceptible, Weiser se remonta a lo que considera la primera forma de tecnología, la escritura. La concepción y desarrollo de la misma como medio de almacenamiento y de transmisión de conocimiento, representó en su momento una barrera tecnológica. Esta tecnología de acceso a pocos se fue extendiendo con el paso del tiempo, y su uso, sus grandes ventajas en comunicación y en preservación de conocimientos, se volvió imperceptible en el desenvolvimiento diario.

La escritura, que en la actualidad se ha absorbido como propia, ha permitido interactuar de forma transparente con el entorno, de tal manera que se puedan percibir acciones o comportamientos a través de información escrita, como sucede con las señales de tránsito. De esta forma propició una distribución y prevalencia de conocimiento que sin ella no habría podido darse.

La transparencia con la que permite interactuar esta primera forma de tecnología, es a lo que se refiere el término pervasivo o ubicuo, a la integración de la tecnología en nuestras vidas de tal forma que no percibamos la interacción con ella, al mismo tiempo que vuelve nuestras vidas más fáciles de sobrellevar. La necesidad de percibir el entorno para actuar en respuesta, es lo que hace la diferencia entre el cómputo tradicional y el cómputo pervasivo.

Por definición, el término ubicuo hace alusión a la omnipresencia, a estar en todos los sitios. Entonces, como definición de cómputo ubicuo, podríamos adoptar el término que ofrece Mattern," la omnipresencia de computadores muy pequeños interconectados sin cables, que se incrustan de forma casi invisible en cualquier tipo

de objeto cotidiano" [17].

La llegada de las computadoras a los entornos personales, ha generado una sobrecarga de información en las personas. Gestionar el manejo de ellas es una tarea compleja para muchos. En este sentido, el cómputo ubicuo sugiere reducir la complejidad a través de entornos digitales interconectados que perciben, se adaptan y actúan para responder a las necesidades de los seres humanos gracias a su capacidad de inferencia.

La evolución del cómputo pervasivo se ha producido gracias al desarrollo de dos grandes áreas: los sistemas distribuidos y las redes móviles. Con el desarrollo de nuevas tecnologías en estas áreas, se han asentado las bases para ir integrando a la vida cotidiana soluciones tecnológicas más complejas en arquitectura, a la par que la interacción con ellas se vuelve cada vez más imperceptible y de mejor calidad [18].

Entre 1970 y 1990, gracias a la convergencia de las redes de datos y las PC's (del inglés personal computer, computadora personal), se crearon las bases conceptuales y algorítmicas de la computación distribuida. Fue entonces que empezó a favorecerse el desarrollo de éstos y se fueron resolviendo problemas de investigación en áreas como la comunicación remota, la tolerancia a fallos, el acceso remoto a la información, la alta disponibilidad y la seguridad distribuida.

Por otra parte, durante los primeros años de la década de los 90's, con las redes inalámbricas, surgen nuevos escenarios que confrontar en el desarrollo de sistemas distribuidos con clientes móviles. La movilidad entonces, dio pie al desarrollo de técnicas especializadas como la calidad de las redes, limitaciones de recursos locales por pesos y tamaños, la autonomía de las computadoras y la robustez de los elementos móviles [18].

Al conjugarse todos estos avances en movilidad y sistemas distribuidos, el concepto de cómputo ubicuo empieza a tomar forma. Surgen entonces nuevos espacios de crecimiento para el cómputo ubicuo, propiciando esta consolidación un modelo general de esta disciplina que clasifica e interrelaciona sus componentes en cuatro grandes ramas: dispositivos, redes, capa intermedia y aplicaciones [19].

- **Dispositivos.** Los dispositivos en un ambiente inteligente, incluyen desde los periféricos tradicionales, los PDA's (del inglés personal digital assistant, asistente digital personal), teléfonos celulares o dispositivos personales, hasta los dispositivos inteligentes con bio sensores o sensores embebidos. Una idea clave en esta rama del cómputo pervasivo es abarcar todo tipo de dispositivo, trabajando su incorporación transparente con las actividades de las personas.

- **Redes.** Los avances en la vertiente de las redes persiguen la interconexión de dispositivos con alta calidad de servicio, buscando a través de ello, integrar aplicaciones con los sistemas sociales existentes.

- **Capa Intermedia (Middleware).** La rama de capa intermedia se encarga de las interacciones entre el núcleo de red y el usuario, enmascarando la heterogeneidad y propiciando así un vínculo imperceptible para éste último.

- **Aplicaciones.** Las aplicaciones en el cómputo pervasivo son encaminadas más al medio ambiente que a un entorno web o al cómputo móvil. Explotan estas características para ofrecer un resultado que va más allá de las condiciones particulares de estas tecnologías, utilizándolas como medio para un fin que involucra la percepción del medio ambiente y la actuación en función de éste.

Conforme las tecnologías evolucionan dando paso a dispositivos más pequeños y con mejores funcionalidades, mayores velocidades de transferencia de datos, y una distribución tecnológica que facilita su adquisición, los ambientes pervasivos se consolidan y se enriquecen cada vez más.

A la par de todas las ventajas que puedan ofrecer los sistemas ubicuos, se encuentra una gran desventaja, la privacidad. Cuanto más inteligente sea un sistema, obligadamente deberá gestionar más conocimiento. En el caso del cómputo pervasivo, la inteligencia del sistema está basada en el conocimiento que obtiene de sus usuarios. Es aquí donde pueden surgir aristas que desalienten al uso de ellos.

Las capacidades de monitoreo se pueden volver en consecuencia, intrusivos para los usuarios, pues la cantidad de sensores y máquinas que fungen como observadores almacenan cada vez más aspectos del usuario y sus rutinas. En estos contextos

es difícil establecer los límites de la privacidad y para el adecuado funcionamiento de este tipo de entornos, es necesario llegar a un balance entre la protección de la privacidad y el mantener un adecuado flujo de información para dotar de inteligencia al medio.

## 2.3. Agentes software

Los agentes software constituyen un paradigma de programación para el desarrollo de aplicaciones software [20]. Éstos hoy en día, han despertado interés en muchas ramas de la computación e inteligencia artificial. Se utilizan en la implementación de una amplia variedad de aplicaciones desarrolladas para disciplinas tan dispares como la educación, el comercio electrónico o la medicina, por mencionar algunas.

Muchas son las definiciones que se han propuesto para conceptualizar lo que es un agente software, sin embargo, no se ha llegado a un consenso sobre la más adecuada. Un agente software se puede definir como un ente informático que puede percibir su entorno, tomar decisiones autónomas y actuar de una forma correcta, ya sea por iniciativa propia o porque otro agente así lo requiera. Hay sistemas que se componen de un solo agente y es suficiente, pero hay otros que necesitan dos o más, cuyos agentes están en comunicación constante para lograr un objetivo, a estos se les denomina sistemas multi-agente [20].

Algunas de las principales características de un agente son:

- **Autonomía.** Hace referencia a la capacidad del agente de tomar sus propias decisiones basándose en los cambios de su entorno y su experiencia, sin necesidad de un control exterior explícito.
- **Sociabilidad.** Se refiere a la habilidad del agente para comunicarse e intercambiar información con otros agentes y otras entidades.
- **Reactividad.** Es la capacidad de un agente para percibir los cambios en su entorno respondiendo a tiempo a los cambios detectados con el fin de alcanzar sus metas.

- **Iniciativa.** Tiene la capacidad de emprender las acciones necesarias para lograr sus objetivos.
- **Benevolencia.** Disposición del agente de cooperar con otros, mientras dicha acción no entre en conflicto con sus objetivos.
- **Racionalidad.** El agente siempre lleva a cabo la acción "correcta" de acuerdo a su entorno y experiencia.

Aunque es difícil establecer el grado de importancia que tienen cada una de estas características, al ser propias de los agentes software, hacen la diferencia entre ellos y un programa computacional. Además de estas particulares características, existen aspectos que diferencian un agente de otro. Con base a estos aspectos se establecen criterios de clasificación que según sea el caso, pueden tomar en cuenta las características individuales de cada agente, su modo de organización o su aplicación entre otras.

Considerando los aspectos individuales de un agente, su clasificación es siguiente:

- **Agentes reactivos.** Se caracterizan por realizar tareas sencillas fundamentadas en la recepción de eventos externos. El comportamiento reactivo puede ser un cambio en el estado interno del agente, o la ejecución de funciones sobre el entorno. Un agente reactivo no realiza procesos de razonamiento ni tiene mecanismos de representación del conocimiento.
- **Agentes cognitivos.** Llevan a cabo tareas complejas para las cuales realizan procesos de razonamiento, planificación y aprendizaje. Este tipo de agente se basa en un modelo computacional cuyo ciclo percibe $\rightarrow$ asimila $\rightarrow$ razona $\rightarrow$ actúa [21].

Una clasificación en función de la organización, parte de la consideración de que el agente forma parte de un conjunto o sistema multi-agente, dividiéndolos por tanto en:

- **Agentes individualistas.** Se caracterizan por no necesitar de otros agentes para lograr sus objetivos. Trabajan solos y no tienen capacidad de cooperación.

- **Agentes cooperantes.** Pueden realizar tareas solos o colaborando con otros agentes [21].

En función de la aplicación que se le dé al agente, se puede clasificar como:

- **Agente de interfaz o usuario.** Cuando el agente funciona como asistente personal y posee las características de aprendizaje y autonomía. Se utilizan para enseñar al usuario o servirle de guía en el manejo de alguna aplicación en particular.
- **Agente de búsqueda.** Construye sus acciones a partir de pequeños trozos de información. No se trata de un simple algoritmo de búsqueda y debe dar un buen resultado para el usuario.
- **Agente de monitoreo.** Su principal característica es la de ejecutarse continuamente. Está en constante vigilancia de entradas, salidas y modificaciones en un sistema, página web, base de datos o en cualquier entorno para el que se haya implementado. Su función es registrar y dar aviso de lo que pasa dentro del entorno.
- **Agente de filtrado.** Este tipo de agente trabaja en coordinación con el de monitoreo, y su función es tener la información siempre actualizada para el usuario de acuerdo a su perfil.

Es evidente que un agente, en función del contexto en el que existe, tendrá más o menos características de las mencionadas previamente. La implementación de estos aspectos particulares de los agentes se da en función de las necesidades que se requieran para su contexto.

### 2.3.1. Sistemas multi-agente

Un sistema multi-agente (SMA) se compone de dos o más agentes que se comunican entre sí. Estos agentes tienen la capacidad de actuar sobre el entorno. Cada agente tiene establecida una zona de interacción, resultando en un control mucho

más preciso de su zona y permitiendo a través de las interacciones entre ellos el control de su entorno.

En un SMA sucede lo mismo que en la sociedad, es decir, podemos encontrar una división clara en las actividades para conseguir un elevado grado de especialización y eficiencia. Esta conceptualización trasladada al caso de los agentes plantea a múltiples agentes autónomos que interactúan y se coordinan cooperando y/o compitiendo de forma adaptativa. A esta constitución de sociedades de agentes se les consideran Sistemas multi-agente.

En un SMA es importante la comunicación ya que ningún agente por sí solo conoce el entorno por completo, es por esto que para resolver el objetivo general del sistema es necesaria la interacción entre los agentes que lo conforman. Los puntos de vista de cada agente son limitados y los datos están descentralizados, por eso la comunicación es fundamental. La implementación de un SMA se lleva a cabo a través de la computación asíncrona, es decir, no todos los agentes actúan al mismo tiempo.

Los SMA están siendo utilizados cada vez más en una amplia variedad de aplicaciones entre las cuales podemos encontrar sistemas de asistencia personal, aplicaciones para el comercio electrónico, control de tráfico aéreo, reconocimiento de imágenes o sistemas complejos de misión crítica para aplicaciones de la industria entre otras [22].

### 2.3.2. Arquitecturas multi-agente

Cada agente cumple una función específica resultante de sus características particulares, éstas lo diferencian de los demás, por lo que habrá algunos cuyo comportamiento sea más complejo e inteligente que otros. Por esta razón, la arquitectura elegida dependerá de las características de los agentes, de sus tareas y de su entorno.

La arquitectura determina los mecanismos que utiliza un agente para actuar o comunicarse con el fin de que el agente exhiba una determinada conducta. La diferencia entre las arquitecturas existentes se da en la forma en la que se interco-

nectan los módulos que componen los agentes, así como también en el método de descomposición de las tareas.

En el ámbito de los agentes es posible encontrar una gran variedad de arquitecturas, mismas que se pueden clasificar como [21]:

- **Reactivas.** se caracterizan por no tener un modelo simbólico como elemento central de razonamiento y no utilizan un razonamiento simbólico complejo.
- **Deliberativas.** Utilizan modelos de representación simbólica del conocimiento, es decir, contiene un modelo simbólico del mundo, donde se decide a través de mecanismos de razonamiento lógico basados en correspondencia de patrones y manipulación simbólica.
- **Híbridas.** Se compone de una parte reactiva y otra deliberativa, reduciendo de esta forma las limitaciones que presentan las dos anteriores.

A continuación se describen algunas de las arquitecturas de agentes más utilizadas:

#### 2.3.2.1. BDI

La arquitectura BDI (Belief - Desire - Intention) tiene bases filosóficas en el entendimiento del razonamiento práctico, que se refiere a tomar decisiones momento a momento para conseguir un objetivo. Se caracteriza por el hecho de que los agentes que la implementan poseen los estados mentales creencias (belief), deseos (desire) e intenciones (intention).

Las creencias hacen referencia al conocimiento que el agente tiene del mundo. Éstas se pueden expresar con cualquier estructura de datos. En este sentido su implementación puede llevarse a cabo empleando desde un conjunto de variables hasta una base de datos utilizando un conjunto de expresiones lógicas.

Los deseos guían las motivaciones del agente, siendo sus múltiples objetivos ordenados en términos de costo o prioridad para a alcanzarlos. Su estructura es similar a la utilizada para implementar las creencias.

Las intenciones son la representación de la estrategia que está siguiendo el agente y puede que éstas sean reconsideradas. Representa la última secuencia de acciones efectuadas en el entorno [23]. La arquitectura BDI, al combinar un modelo filosófico del razonamiento humano fácil de comprender, un considerable número de implementaciones y una semántica lógica y abstracta adaptada por la comunidad científica, se ha convertido en una de las más utilizadas [24].

Según el modelo para implementar agentes BDI desarrollado por Rao y Georgeff en 1995, los agentes con esta arquitectura pueden realizar tareas de control en un entorno dinámico complejo. En general, los dominios de aplicación dinámicos en tiempo real se caracterizan por lo siguiente:

- En determinado momento pueden existir muchas formas de que el entorno varíe.
- En determinado momento pueden existir muchas acciones o procedimientos que el sistema puede ejecutar.
- En determinado momento el sistema puede tener muchos objetivos.
- El entorno sólo se percibe localmente.
- Existe un tiempo razonablemente limitado, tomando en cuenta la rapidez con la que evoluciona el entorno, en el que las acciones son llevadas a cabo.

Para que el agente actúe, debe seleccionar las acciones apropiadas de todas las opciones disponibles. Para diseñar la función de selección debe tener información sobre el estado del entorno, contenida en algún componente del estado del sistema (creencias). Además debe tener información sobre los objetivos a conseguir (deseos) en otro componente. El curso de ejecución de la acción actual también debe ser conocido (intenciones) por si el entorno varía durante la ejecución de la acción elegida, o de la función de selección [24].

#### 2.3.2.2. GRATE

La arquitectura GRATE *(Generic Rules and Agent model Tesbed Environment)* es una arquitectura en capas que se caracteriza porque el comportamiento del agente es guiado por las actitudes mentales creencias, deseos e intenciones. Los agentes se dividen en dos capas, una de dominio que resuelve los aspectos de organización, y otra de cooperación y control que asegura la coordinación de las metas individuales del agente con los demás [25].

Esta arquitectura está especialmente diseñada para construir aplicaciones en las que los agentes deben cooperar entre sí para conseguir una serie de metas. Por este motivo las dos capas de la arquitectura modelan los posibles roles que un agente puede desempeñar: como entidad individual, y como miembro de una comunidad. La característica principal que poseen los agentes implementados bajo esta arquitectura es una cantidad significativa de conocimiento de cooperación y control.

Los agentes GRATE modelan el conocimiento en tres partes:

- *El almacén de información (information store).* Comprende un repositorio de toda la información del dominio generada por el propio agente o recibida por él como resultado de un proceso de interacción con los demás agentes de la comunidad.
- *El modelo propio (self model).* Se compone del conocimiento sobre el propio agente.
- *El modelo de otros agentes (acquaitance model).*Está compuesto por las representaciones de los agentes de la comunidad con los que el agente interactúa.

El conocimiento que contienen los modelos propio y de otros agentes se puede clasificar en función del tipo de conocimiento en:

- *Conocimiento del estado.* En función del progreso de las tareas.
- *Conocimiento sobre capacidades.* Descripción de tareas.

- *Conocimiento intencional.* En función de las intenciones.
- *Conocimiento de evaluación.* De acuerdo al tiempo para completar las tareas.
- *Conocimiento del dominio.*

#### 2.3.2.3. IRMA

El modelo de creencias, deseos e intenciones (BDI) ha servido como base para el desarrollo de muchas otras arquitecturas. IRMA *(Intelligent Resource-bounded Machine Architecture)* es una arquitectura deliberativa que toma como fundamento el modelo BDI [26].

La arquitectura IRMA mantiene cuatro estructuras de datos simbólicos: una librería de planes, un conjunto de intenciones estructuradas como planes, una representación explícita del conjunto de creencias y una última estructura conteniendo el conjunto de deseos. La librería de planes contiene todos los planes sobre los cuales el agente tiene información, mientras que el conjunto de intenciones estructuradas como planes, contiene los planes del agente en el momento actual.

Los agentes creados bajo el esquema IRMA, deben realizar un conjunto de procesos desde el punto de vista funcional, estos son: un motor de razonamiento (means-end reasoner), un analizador (opportunity analyser), un proceso de filtrado (filtering process) y un proceso de deliberación (deliberation process). A continuación se presentan sus características particulares:

- *Motor de razonamiento.* La función del motor de razonamiento es proponer subplanes para completar los planes existentes y es invocado por cada uno de los planes parciales existentes. El motor de razonamiento puede proponer distintas opciones (varias y con características particulares) para conseguir el objetivo de cada plan parcial.
- *Analizador.* Este módulo propone nuevas opciones en respuesta a los cambios percibidos en el entorno. No todas las opciones se proponen por el motor de razonamiento. Las opciones que propone el analizador se generan en función

de los cambios en el entorno, por lo que no tienen que referirse necesariamente a cómo conseguir un objetivo que ya estaba planificado.

- *Proceso de filtrado.* Su función es la de filtrar las opciones que han sido propuestas por el motor de razonamiento o por el analizador. El filtrado de propuestas se realiza para hacer más eficiente el proceso de deliberación. Los planes del agente deben tener una congruencia, es decir, se deben corresponder tanto internamente como con las creencias que éste tiene. Por este motivo, en el proceso de filtrado, se descartan las opciones que no son consistentes con los planes y las creencias del agente.

- *Proceso de deliberación* Este proceso produce nuevas intenciones en función de las mejores opciones que superaron el proceso de filtrado. Las nuevas intenciones se incorporan a los planes del agente. En resumen, el proceso de deliberación es lo que se conoce como toma de decisiones en la teoría tradicional.

#### 2.3.2.4. ADEPT

La arquitectura ADEPT *(Advanced Decision Environment for Process Tasks)*, es una aproximación basada en agentes utilizada para sistemas que gestionan procesos de negocios. Se creó como un proyecto para investigar la tecnología, así como los métodos que se deben de utilizar para tratar todos los aspectos de la información. Estos aspectos comprenden la recopilación gestión, distribución y presentación, construyendo en ADEPT, para este fin, un sistema compuesto de muchos agentes autónomos [27].

Cada agente se caracteriza por los servicios que ofrece a otros agentes, de tal forma que cada agente controla, gestiona y provee un determinado número de servicios a otros agentes en la infraestructura. Para que este tipo de colaboración se dé, es necesario que los agentes se comuniquen y entablen negociaciones entre ellos.

La arquitectura de un agente ADEPT se compone de cuatro módulos, el primero de ellos contiene la información con la que cuenta el agente mientras que los otros

tres manejan las funcionalidades del agente. A continuación se describen:

- El módulo que contiene el modelo del propio agente y los modelos de los demás agentes con los que se comunica *(self and acquaitance models).* Contiene información acerca de los servicios que proporciona el agente, sobre las capacidades de negociación con otros agentes, mantiene un registro de los agentes que proporcionaron servicios en el pasado, así como de los contratos que el agente realizó con otros agentes, etc.

- El módulo de comunicación *(communication module).* Este módulo se encarga de mantener la comunicación entre el agente y su agencia local y entre el agente y la comunidad, dirigiendo los mensajes necesarios entre ellos. El establecimiento de comunicación agente - comunidad incluye planificación de servicios concurrentes, inicio de nuevos servicios, paso de información a servicios y monitorización del estado de los servicios en curso.

- El módulo de evaluación de situaciones *(situation assesment module).* Sirve de enlace entre el rol individual y el rol social del agente. Cuando este módulo identifica la necesidad de establecer un contrato con determinado agente externo para que proporcione un servicio, invoca al módulo de ejecución de servicios.

- El módulo de ejecución de servicios *(service execution module).* Es el encargado de gestionar la negociación para llevar a cabo un acuerdo.

Los módulos de comunicación, evaluación de situaciones y ejecución de servicios (módulos funcionales), hacen uso del modelo propio del agente y del modelo de los demás agentes.

Los agentes implementados bajo esta arquitectura están organizados en agencias autónomas. Un agente tiene asignadas a su cargo una serie de tareas y se puede comunicar con otros agentes de forma 'holgada', o bien de forma 'estrecha'. En la comunicación en forma holgada no existe ningún agente controlador, mientras que en la comunicación en forma estrecha, es necesaria la existencia de un agente

controlador, de forma que los agentes de fuera de la agencia tienen restringido el acceso.

En la agencia residen los agentes servidores, que se comunican de forma holgada con los demás agentes de la misma agencia, pero de forma estrecha con el agente controlador. Las agencias por lo general se organizan de forma jerárquica, de tal forma que los agentes de una agencia pueden ser controladores de otras agencias de más bajo nivel. Para dar un servicio es posible seleccionar agentes de diferentes agencias y formar una agencia virtual.

Organizar el sistema en agencias ofrece la posibilidad de agrupar a los agentes que persiguen fines comunes, por esta razón se considera la arquitectura más adecuada para desarrollar este trabajo.

### 2.3.3. Metodologías

El desarrollo de software, independientemente de sus características particulares, requiere de herramientas que faciliten la obtención del resultado final. En este sentido existen numerosas metodologías creadas para sentar las bases de un desarrollo coherente y apegado a los requerimientos. En el caso de los sistemas multi-agente se han desarrollado también distintas metodologías encaminadas a facilitar el desarrollo de agentes, tomando como base conceptos de la Ingeniería de Software Orientada a Objetos, así como de la inteligencia artificial.

El desarrollo de una metodología tiene como finalidad, ayudar a profundizar en el conocimiento de un sistema en particular, y segundo, en el diseño del mismo. Por ello, a medida que los agentes se han vuelto más comunes en el ámbito de las ciencias de la computación, se ha visto un aumento en el interés dedicado a la elaboración de metodologías para el desarrollo de sistemas basados en agentes.

Las metodologías consisten generalmente en colecciones de modelos, y asociados con estos modelos, sus líneas de seguimiento. Los modelos comienzan siendo tentativos y más bien abstractos, y conforme el proceso de análisis y diseño vaya creciendo, se vuelven más concretos, detallados, y más cercanos a la implementación. A con-

tinuación se describen algunas de las metodologías para el desarrollo de sistemas multi-agente.

#### 2.3.3.1. AAII

La metodología AAII para análisis y diseño orientado en agentes fue desarrollada como resultado de la experiencia ganada después de la creación de grandes proyectos desarrollados en el Instituto Australiano de Inteligencia Artificial (AAII por sus siglas en inglés) a lo largo de los años 90's. Tiene sus bases principalmente en las metodologías orientadas a objetos añadiendo características específicas orientadas a sistemas multi-agente. La metodología en sí, consiste en un conjunto de modelos que, cuando están completamente elaborados definen la especificación del sistema multi-agente [28].

Para el desarrollo de los modelos con base en esta metodología es necesario tomar en cuenta los siguientes aspectos:

- Identificar los roles relevantes del dominio de la aplicación, y con base en ellos desarrollar una jerarquía de clases de agente. Por ejemplo, un rol podría ser monitor de clima, con el cual, el agente i debe informar al agente j las condiciones climáticas cada cierto tiempo.

- Identificar las responsabilidades asociadas a cada rol, los servicios requeridos y los servicios proporcionados por el rol, y a partir de ellos determinar metas asociadas con cada servicio. Con respecto al ejemplo de arriba, las metas podrían ser encontrar el clima actual y comunicárselo al agente j.

- Para cada meta, determinar los planes que podrían ser usados para lograrla, y las condiciones y contexto en las que cada plan es apropiado.

- Determinar la estructura de creencias del sistema. Es necesario definir los requisitos de información para cada plan y cada meta.

#### 2.3.3.2. AUML

La metodología AUML parte de la utilización de herramientas de desarrollo ya existentes, como es el caso de UML (del inglés Unified Modeling Language, Lenguaje Unificado de Modelado), orientándolas hacia el campo de los agentes. La ventaja de UML es su gran difusión y aceptación, por lo que resulta evidente que algunas de sus herramientas pueden ser aplicadas directamente a sistemas basados en agentes adoptando algunas convenciones.

La visión que se presenta de un agente puede considerarse como el siguiente paso a partir del concepto de objeto. Actualmente se está trabajando en esa aproximación, sugiriéndose extensiones a UML para que soporte la funcionalidad adicional que aportan los agentes. Las adiciones más importantes son:

- El soporte para la expresión de procesos simultáneos de interacción (como es el caso de los mensajes de agentes), permitiendo así a UML el modelado de los protocolos de agentes conocidos como redes de comunicación.
- Una noción de 'rol' que se extienda en UML, en particular, que permita modelar agentes que jueguen muchos roles [28].

#### 2.3.3.3. INGENIAS

INGENIAS es el nombre de una metodología de ingeniería de software orientada a agentes [29], proporciona una notación visual para modelar sistemas multi-agente desde cinco puntos de vista: organización, agente, objetivos/tareas, interacciones y entorno. Los cuales se describen a continuación:

- Modelo de agente. De manera individual los agentes realizan tareas o persiguen objetivos.
- Modelo de objetivos y tareas. Identificación de objetos generales y descomposición en objetos más concretos que se pueden asignar a agentes.
- Modelo de organización. Describe el marco en el que existen los agentes.

- Modelo de interacción. Muestra las interacciones que existen entre los agentes/roles y como se producen.
- Modelo de entorno. Entidades y relaciones dentro del SMA, define los sensores y actuadores de los agentes y los recursos y aplicaciones con los que tiene que interaccionar.

Después de haber descrito las diferentes perspectivas que ofrecen las metodologías se puede decir que:

- INGENIAS ofrece mejor soporte en todas las fases del ciclo de vida de una aplicación basada en agentes incluyendo la fase de gestión.
- INGENIAS ofrece herramientas que otras metodologías no tienen. Por ejemplo, INGENIAS dispone de una herramienta de modelado visual pensada para dar soporte a las fases del ciclo de vida basada en el RUP (por sus siglas en inglés de Rational Unified Process, Proceso Racional Unificado).
- INGENIAS dispone de una herramienta de generación de código automática.

Debido a las razones ofrecidas, se decidió elegir INGENIAS como metodología de desarrollo puesto que es una de las más completas. Propone un lenguaje visual para generar los diferentes modelos o vistas, mismos que están soportados por sus correspondientes meta-modelos lo cual facilita la comprobación automática de inconsistencias en el diseño.

### 2.3.4. Plataformas

Existe en la literatura una ambigüedad al hablar de las plataformas multi-agente, ya que en unos casos se les refiere como plataformas de desarrollo de sistemas multi-agente, y en otras ocasiones como plataformas de ejecución de sistemas multi-agente, sin saber que ambas tienen su definición propia.

Cuando se habla de una plataforma de desarrollo de sistemas multi-agente, se hace referencia a una herramienta que permite el diseño y desarrollo de un sistema

formado por un grupo de agentes software. Estos agentes tendrán una arquitectura básica determinada, y podrán interactuar entre ellos para resolver los problemas para los cuales el sistema ha sido diseñado.

Por otra parte, una plataforma de ejecución de agentes debe proveer los servicios necesarios para la ejecución de un conjunto de agentes en un entorno distribuido y posiblemente heterogéneo. Debe contar con los servicios de transporte de mensajes, seguridad y directorio de agentes [30].

En el caso de las plataformas de agentes, deben poseer la capacidad de generar automáticamente el código de los agentes, el cuál proveerá su funcionalidad básica y las primitivas necesarias para la interacción con otros agentes del sistema. Además, esta herramienta debe ofrecer la posibilidad de agregar, en forma manual, código personalizado a los agentes en algún lenguaje de programación, con el fin de dotar al agente de mayor funcionalidad, inteligencia, etc. Para lograr estos objetivos, la plataforma deberá poseer un compilador para generar el código final de cada agente, incluyendo su código personalizado.

A continuación se presenta un breve repaso de algunas de las plataformas más conocidas en la literatura.

#### 2.3.4.1. JACK

JACK [31] es un entorno maduro, multi-plataforma, diseñado con el lenguaje de programación JAVA y es utilizado para la construcción, el funcionamiento y la integración de los sistemas multi-agente de calidad comercial. Está construido sobre la base lógica BDI lo cual permite a los desarrolladores gestionar la complejidad del problema. En JACK, los agentes se definen en términos de sus creencias (lo que saben y lo que saben hacer), sus deseos (cuáles son los objetivos que les gustaría lograr), y sus intenciones (las metas que se cometen en la actualidad para lograr).

Esta es una manera muy eficaz de creación de aplicaciones para entornos dinámicos y complejos, cada agente es responsable de la consecución de sus propios objetivos, reaccionando a los eventos y la comunicación con otros agentes del sistema. No

hay necesidad de programar de forma explícita las interacciones de toda la aplicación, en lugar de eso, las interacciones surgen como un subproducto de los objetivos y las capacidades de los agentes constituyentes individuales.

#### 2.3.4.2. ZEUS

ZEUS [32] es una herramienta para construir aplicaciones multi-agente colaborativas que provee un entorno integrado para el desarrollo rápido de sistemas. ZEUS define una metodología de diseño de sistemas multi-agente y lo soporta mediante un entorno visual para capturar las especificaciones de los agentes. Estas especificaciones son luego utilizadas para generar el código fuente en Java.

ZEUS es uno de los casos en los que se haba de una plataforma de desarrollo de agentes puesto que cumple con todas las características de una plataforma de desarrollo pero no es una plataforma de ejecución. Cuenta con la interfaz gráfica, es 'liviana' en ejecución, permite generar el código de los agentes y provee todos los servicios para la ejecución de los agentes. A pesar de proveer estas funcionalidades, no provee un entorno de ejecución dentro de la plataforma, la ejecución es externa a la misma.

#### 2.3.4.3. JADE

La plataforma JADE [33] fue creada completamente en Java para el desarrollo y ejecución de sistemas multi-agente. Se compone de contenedores de agentes los cuales pueden estar en una red. Los agentes viven en los contenedores que son procesos de Java que proporciona el tiempo de ejecución de JADE y todos los servicios necesarios para alojar y ejecutar a los agentes.

Además cuenta con algunas herramientas interesantes, como lo es su interfaz gráfica, la cual sirve para monitorizar y controlar la plataforma con todos sus contenedores remotos, gestiona remotamente el ciclo de vida de los agentes (crear, suspender, reactivar, matar, migrar, clonar) y lanza otras herramientas gráficas, algunas de ellas son de gran utilidad como enviar mensajes 'ad-hoc' a los agentes de

la plataforma, y observar el paso de mensajes entre los agentes durante su ejecución.

JADE se considera la mejor opción para el desarrollo de este proyecto, puesto que ofrece una plataforma para la gestión de agentes facilitando con ello la depuración del sistema al hacer pruebas por separado a cada uno de los agentes que lo componen.

## 2.4. Trabajos relacionados

Con la finalidad de abordar las soluciones propuestas para tratar la problemática de muerte por hipertermia en bebés en automóviles cerrados, se presenta a continuación una clasificación en donde se condensan las soluciones que se apeguen a cada uno de los grupos definidos. Los tres grupos son: soluciones sin capacidad de detección, soluciones basadas en el vehículo y soluciones mediante sensores en el asiento del bebé.

### 2.4.1. Soluciones sin capacidad de detección

Existen en el mercado una serie de productos que pueden clasificarse en esta categoría. Van desde tarjetas hasta brazaletes que hay que colocarse como recordatorio. La efectividad de estas soluciones es limitada, ya que la principal recomendación para su uso es volverlo una rutina y es justamente el cambio de rutina, como muestran las estadísticas de muertes por golpe de calor, lo que ocasiona estos lamentables sucesos.

### 2.4.2. Soluciones integradas en el vehículo

*Backseat minder* se fundamenta en el hecho de que colocar a un niño en la parte trasera de un automóvil, tomará más de 3 segundos. Con esta consideración la empresa CSO RADIO en NJ, USA, pone en el mercado un producto que mide el tiempo que una de las puertas traseras permaneció abierta. El sistema se activará al encender el automóvil, siempre que el tiempo de apertura supere los tres segundos.

Cuando el vehículo se apaga, el sistema emite un sonido que se apagará manualmente desde el interior del vehículo presionando un botón colocado en las puertas traseras. Si se abren las puertas traseras, pero el automóvil no se enciende en determinados minutos, el sistema no entrará en funcionamiento [9].

*Kiddie Voice Child Reminder* es un Sistema basado en el vehículo que fue anunciado en un principio para los autobuses escolares. Está integrado en la unidad de control electrónica del vehículo. El sistema se activa cada vez que el motor se apaga emitiendo un mensaje de advertencia por voz, solicitando al conductor que verifique si hay pasajeros en el vehículo. Solo puede desactivarlo el conductor desde la parte trasera del autobús, verificando durante este recorrido que el vehículo está vacío. Se lleva a cabo una segunda inspección visual al hacer el recorrido de regreso para salir del autobús [9].

El sistema *CAREseat* es una propuesta de asiento para niño modificado que se integra con el sistema electrónico del vehículo para alertar al conductor si deja un niño pequeño en el automóvil. El sistema detecta la presencia de un niño verificando si el cinturón de seguridad del asiento para niños está abrochado, si es así, cuando el conductor apaga el automóvil, saca la llave y abre su puerta se escuchará un sonido.

Si aun así el conductor cierra el automóvil sin retirar al pequeño, el claxon del vehículo sonará durante 20 segundos. Se propone como último recurso emitir una alerta a las autoridades a través de un dispositivo incluido en el vehículo, es decir, funcionará para aquellos vehículos equipados con el mismo. Esta tecnología no está en el mercado y para su operación deberá integrarse en el sistema electrónico del automóvil [9].

Otra solución basada en el vehículo es el *Life Warn System,* cuya funcionalidad se basa en el escaneo continuo dentro del vehículo (cabina y portaequipaje) en busca de dióxido de carbono ($CO_2$) exhalado. Esta solución utiliza múltiples sensores que pueden detectar humanos o mascotas olvidados. El sistema tiene la capacidad de encender el motor, quitar seguros de las puertas y abrir el portaequipajes a quien acuda a atender la emergencia. Esta solución no se ha llevado al mercado [9].

Una solución más, es la propuesta por [34], *Mechatronic system to protect child or*

*warning signal to parents.* Ésta considera un sistema mecatrónico para detectar niños atrapados en un vehículo a través de un sensor de presencia. El sistema empieza a funcionar 10 minutos después de que se ha apagado el motor y se detectan los seguros de las puertas activados. La solución propuesta al detectar presencia y temperatura es encender el aire acondicionado del auto, y 10 minutos después, si no hay respuesta, se iniciará la alarma de robo del vehículo.

*Precious cargo* es una aplicación que se conecta al automóvil a través de bluetooth, al detectar la conexión con el automóvil, pregunta al usuario si lleva 'precious cargo', si se indica que si, al apagar el automóvil la aplicación emitirá un recordatorio [35].

*Kids4kars* es una app que funciona en vehículos equipados con bluetooth. La app envía una alerta al teléfono del conductor recordándole que lleva al bebé en el automóvil cada vez que se apaga el vehículo. La app se puede configurar para varios automóviles y tiene tres modos de funcionamiento: siempre encendida, encendida por bloques de tiempo preestablecidos o sólo notificaciones. La opción notificación pregunta al conductor si quiere activar la app cada vez que el teléfono se enlaza con el automóvil [36].

### 2.4.3. Soluciones mediante sensores en el asiento del bebé

El sistema *ChildMinder Smart Pad* desarrollado por la compañía Baby Alert International, en Dallas, TX, es un Sistema de monitoreo basado en una almohadilla que detecta la presencia del bebé en el asiento para bebés y se sincroniza con un llavero que emitirá una alarma en caso de ser necesario. La almohadilla tiene distribuidos cinco sensores y se coloca debajo del cojín del asiento para el bebé para detectar la presencia del bebé a través de la presión aplicada a los sensores.

Una vez que el bebé se coloca en su asiento, el sistema empieza a monitorizarlo. Este proceso inicia cuando la almohadilla detecta al bebé en el asiento, al hacerlo emitirá un sonido indicando al adulto responsable que debe sincronizar su llavero con la unidad base. Este proceso se lleva a cabo presionando un botón en el llavero

hasta que el sonido cesa, para ello deberá estar cerca de la unidad base. A partir de este momento, si el adulto responsable se aleja más de 15 pies del vehículo dejando al bebé dentro de él, la alarma del llavero empezará a sonar. El llavero da un margen de 6 segundos para emitir la alarma una vez sobrepasada la distancia [9].

Starfish, es un sistema basado en un sensor colocado en el asiento del bebé, que se activa por el peso del bebé. Éste envía una alerta al teléfono del conductor cuando se aleja más de 6m (20ft) del vehículo. El sistema trabaja para iOS y Android y se conecta al teléfono a través de bluetooth.

Adicionalmente, la aplicación se puede configurar para notificar a contactos de emergencia definidos por el usuario, si éste no respondiera a la notificación dentro de los siguientes cinco minutos [37].

La compañía Suddenly Safe "N" Secure Systems Inc., con sede en Bensalem, PA desarrolló el Sistema de alarma *Deluxe Padded Safety Seat.* Este se compone de una almohadilla que se coloca bajo el cojín del asiento para bebé. La almohadilla tiene un sensor de presión colocado en el centro para detectar la presencia de un bebé.

En el vehículo se coloca un transmisor de radiofrecuencia que se sincroniza con un llavero manteniendo presionado un botón para este propósito mientras el llavero está al lado del transmisor. Una vez que está sincronizado, el llavero emite una fuerte alarma y vibra si el responsable del bebé se mueve fuera del rango establecido. El rango puede establecerse desde 6 hasta 50 pies [9].

*SafeBABI* es un Sistema que alerta, desde un dispositivo inalámbrico, al adulto a cargo utilizando un reloj eZ430 Chronos como receptor si el niño permanece en el vehículo. El Sistema fue desarrollado por la compañía Texas Instruments en Dallas, TX y todos los componentes para su funcionamiento son propietarios de la misma.

El dispositivo inalámbrico cuenta con un interruptor que se presiona con el peso del bebé, al estar presionado transmite continuamente este estado al reloj. Si el reloj sale del alcance del dispositivo inalámbrico y el último estado del interruptor da una lectura positiva de bebé, entonces se emitirá un sonido de alarma y en la pantalla del reloj se leerá "bebé". El alcance del dispositivo inalámbrico es de 10m. Para su adecuado funcionamiento, SafeBABI requiere de una extensa programación

e interacción con el sitio web de la compañía [9].

La NASA desarrolló una solución basada en un sensor de presión, un transmisor y un receptor a que llamó *child presence sensor.* Esta solución utiliza radiofrecuencia para notificar al adulto responsable si el bebé se encuentra en el automóvil. El sensor detecta peso a partir de 226gr. (8oz) y al hacerlo emite un código al módulo de alarma del receptor mismo que se activará sonando 10 veces si el portador del receptor se aleja demasiado del automóvil. Si el conductor no regresa al automóvil y retira al bebé, el receptor emitirá un sonido de alarma continuo después de un minuto, mismo que se apagará únicamente desde el automóvil [38].

El sistema *Forget-Me-Not* se basa también en una almohadilla con sensores de presión que se coloca bajo el asiento del bebé, al momento de alejarse del vehículo más de 20ft., si el bebé permanece en el mismo, sonará una alarma en el llavero del conductor. La alarma se puede justar a 10 o 20 ft. [9].

Una solución más que se encuentra disponible en el mercado, es *First Years' True Fit IAlert.* Es un asiento para bebé con sensores integrados de temperatura, presión y con capacidad de detectar cuando el automóvil se ha detenido. Se conecta a una aplicación móvil y emite una alerta si el bebé es olvidado en el automóvil. Su precio es de 400 USD [39].

Advanced SensorSafe Embrace, es un asiento para bebé desarrollado por Evenflo, cuya funcionalidad se basa en sincronizar un dispositivo de radiofrecuencia colocado en el clip del cinturón del bebé con un dispositivo colocado en el puerto OBT del automóvil. El dispositivo notifica que el bebé está en el automóvil tiempo antes de llegar al destino y emitirá un sonido de alarma si al llegar a destino el clip permanece abrochado [40].

### 2.4.4. Soluciones que involucran más de un sensor

Otra solución basada en sensores, es el Sistema *HALO baby seat* éste se compone de una almohadilla de presión que se activa al colocar al bebé en su asiento. La almohadilla se coloca debajo del cojín del asiento de bebé. Otra de sus funcio-

nalidades es el monitorizar la temperatura del vehículo, que al ser muy baja o muy alta emitirá una alarma al llavero del conductor. Si el conductor no respondiera a la alerta, a través de un sintetizador de voz colocado en la almohadilla se escuchará la frase "baby in danger" con la finalidad de alertar a las personas que pudieran estar alrededor.

Como proyecto por parte de la empresa desarrolladora está el mejorar las funcionalidades del sistema, añadiendo la posibilidad de llamar a una persona responsable, al 911 e interactuar con sistemas GPS. El dispositivo no se encuentra en el mercado [9].

Otra solución se propuso como proyecto de fin de carrera en la Universidad de Texas. Ésta involucra un dispositivo con sensores de movimiento e infrarrojo, con los cuales se detecta la presencia de un bebé olvidado en el automóvil. El dispositivo envía un mensaje de texto al teléfono del responsable del bebé y en caso de no haber respuesta por su parte, entonces envía una alerta a la policía [41].

*Driver's Little helper (Car Seat Monitor)* está diseñado para detectar tanto el movimiento del automóvil como la presencia del bebé en el asiento del auto. Para su funcionamiento utiliza una pequeña almohadilla con el sensor que se coloca debajo del asiento, al colocar al bebé en el asiento se comunica con el conductor a través de bluetooth a un Smartphone Android. A los 4 segundos de que el automóvil se detiene, el teléfono vibra o envía un mensaje de texto. La almohadilla detecta movimiento, temperatura y llegada a destino. Tiene la funcionalidad de alertar a un contacto en caso de emergencia. Su precio está sobre los 80 USD [42].

En la Fig. 2.1 se puede observar una clasificación de las soluciones anteriores, indicando si la solución está integrada en el vehículo, si se compone de varios sensores, si su funcionamiento es a través del asiento del bebé o si es un dispositivo de clip externo. Además de clasificarse de esta forma, se añade la siguiente información para cada solución: el tipo de sensor que utiliza (si es el caso), cuando su funcionalidad requiere del automóvil se señala la parte de ésta que se utiliza y por último el medio a través del cual se recibe la alerta.

| | DISPOSITIVO | TIPOS DE SENSORES | | | | | | | AUTOMÓVIL | | | | | ALERTA VÍA | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PESO ASIENTO | TEMPERATURA | MOVIMIENTO | INFRAROJO | SONIDO | PRESENCIA | CO2 | AC/VENTANAS | MOTOR VEHICULO | PUERTA TRASERA | CLIP / CINTURÓN | GPS | AUTO VOZ | AUTO SONIDO | AUTO ALARMA ROBO | SMARTPHONE ANDROID | SMARTPHONE IOs | LLAVERO |
| CLIP | **Small Ones Safety (SOS) [8] | | | | | | | | | | | ♦ | | | | | | | ♦ |
| | Child Minder Smart Clip System [8] | ♦ | | | | | | | | | | | | | | | | | ♦ |
| | Baby Talk GPS Child Tracker [8] | | | | | | | | | | | | ♦ | | | | ♦ | ♦ | |
| SENSOR EN EL ASIENTO DE BEBÉ | **Child Presence Sensor RF [33] | ♦ | | | | | | | | | | | | | | | | | ♦ |
| | **Forget-Me-Not Car Seat System [8] | ♦ | | | | | | | | | | | | | | | | | ♦ |
| | ChildMinder Smart Pad System [8] | ♦ | | | | | | | | | | | | | | | | | ♦ |
| | Deluxe Padded Safety Seat Alarm System RF [8] | ♦ | | | | | | | | | | | | | | | | | ♦ |
| | SafeBABI [8] | ♦ | | | | | | | | | | | | | | | | | ♦ |
| | Starfish [32] | ♦ | | | | | | | | | | | | | | | ♦ | ♦ | |
| VARIOS SENSORES | **Multi-Agent System | | ♦ | ♦ | | ♦ | | | | | | | ♦ | | | | ♦ | ♦ | |
| | **Halo Baby Seat Safety System [8] | ♦ | ♦ | | | | | | | | | | | | | | | | ♦ |
| | **UT Dallas project [36] | | | ♦ | ♦ | | | | | | | | | | | | ♦ | ♦ | |
| | Driver´s Little Helper [37] | | ♦ | ♦ | | | | | | ♦ | | | | | | | ♦ | | |
| | The First Years' True Fit IAlert car seat [34] | | ♦ | ♦ | | | | | | ♦ | | | | | | | ♦ | ♦ | |
| INTEGRADO EN EL VEHÍCULO | **CAREseat Car Seat System Vehicle-based warning [8] | | | | | | | | | | | ♦ | | | ♦ | ♦ | | | |
| | **The Life Warn System [8] | | | | | | | ♦ | | ♦ | | | | | | | | | |
| | **Mechatronic system to protect child or warning signal to parents [29] | | | | | | ♦ | | | ♦ | ♦ | | | | | | ♦ | ♦ | |
| | **Next-Generation Technologies for Preventing Accidental Death of Children | ♦ | ♦ | | | | | | ♦ | | | | | | | ♦ | ♦ | | |
| | BackSeat Minder [8] | | | | | | | | | ♦ | ♦ | | | | ♦ | | | | |
| | Kiddie Voice Child Reminder [8] | | | | | | | | | ♦ | | | | | ♦ | | | | |
| | Kars4kids safety app [31] | | | | | | | | | ♦ | | | | | | | ♦ | ♦ | |

** Solución no disponible en el mercado

Figura 2.1: Tabla comparativa de dispositivos.

Además de las soluciones presentadas previamente, existen numerosas aplicaciones basadas en notificaciones al celular, éstas no se incluyen en la comparativa puesto que no se considera una solución preventiva constante. Las soluciones que se están considerando tienen la característica de detectar presencia en el automóvil a través de un medio que permanece en el vehículo. En el caso de las soluciones basadas en notificaciones del celular, no se consideran puesto que no existe la certeza de que el teléfono esté encendido o en el automóvil todas las veces que se transporta un bebé.

# Capítulo 3

# Análisis y Diseño

## 3.1. Introducción

Este capítulo presenta el modelo de la solución propuesta, así como el análisis y diseño de la misma. Para llevarlo a cabo, se utiliza una metodología de modelado específica para sistemas multi-agente (SMA) como lo es INGENIAS [29].

Se aborda en primera instancia el modelo propuesto para evitar el olvido de bebés en los automóviles, ofreciendo una visión general del contexto de la aplicación. En el siguiente apartado se describe el modelado del sistema multi-agente que formaliza la propuesta de solución.

Finalmente, en la última sección de este capítulo, se presenta un modelo de notificaciones basado en el propuesto por S. Nava-Muñoz y A. Morán en [15]. En él se especifica la forma en la que se crean las notificaciones, la información que se envía, así como las entidades que actúan como receptores de éstas.

Al hacer uso de las especificaciones de SMA se llega a una verificación de la solución que permite identificar si existen restricciones de desarrollo al momento de consolidar lo establecido en el modelo propuesto. La utilización de esta metodología de modelado permite estructurar de forma concisa los detalles del SMA documentando y facilitando al mismo tiempo la implementación.

## 3.2. Modelo del sistema detector de bebé a bordo

El esquema general de la solución propuesta se muestra en la Fig. 3.1 y se compone de tres grupos: entorno de la aplicación, módulo de interfaz y módulo de monitoreo.

En el entorno de la aplicación se representa la interacción con el smartphone del usuario o de la dependencia de emergencias, así como la interacción con el automóvil. En este entorno se ejecutan todas las acciones, sean de prevención o de corrección, que genera la solución propuesta.

El módulo de interfaz recibe los mensajes del coordinador con el tipo de alerta que debe de enviar y a quién (usuario o línea de emergencias). Su función es la de llevar a cabo la comunicación con el smartphone del usuario o con la dependencia de emergencias.

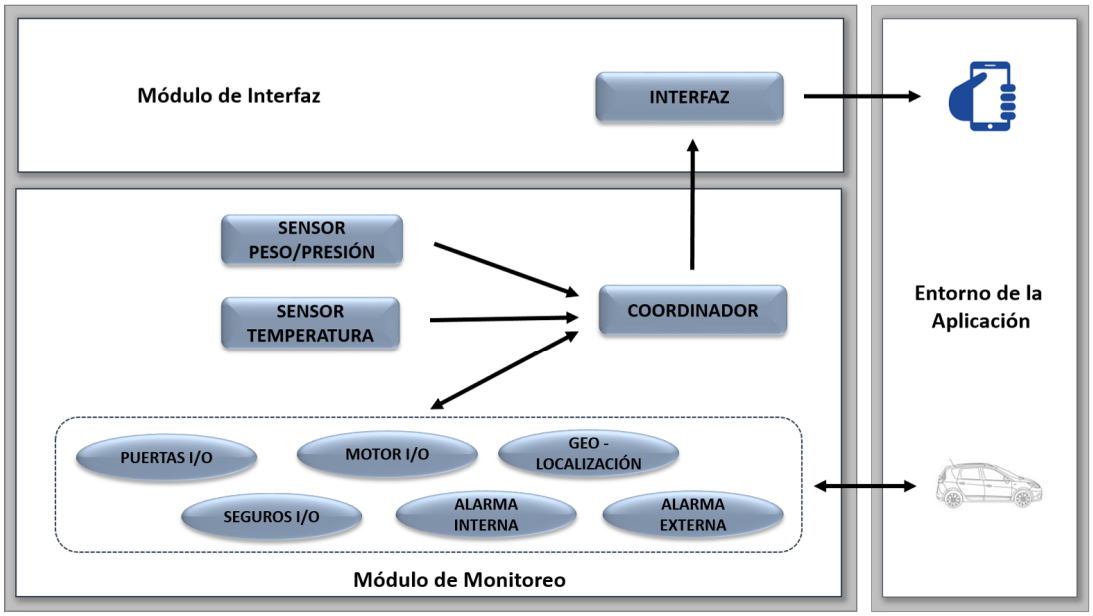



Figura 3.1: Modelo para prevenir el olvido de un bebé en un automóvil.

El tercer grupo de este modelo es el módulo de monitoreo cuya función es la de percibir el medio, en este caso el automóvil. Esta detección del medio se efectúa a través de los sensores de temperatura y peso, así como también de la información que se recibe de la unidad electrónica de control del automóvil. Como se muestra en la Fig. 3.1, la información del automóvil que se requiere para el funcionamiento

de este modelo es: el estado de las puertas, el estado del motor, los seguros del automóvil, la geolocalización y el control de las alarmas internas o externas.

Los estados que se manejan para puertas, seguros y motor son abierto/encendido o cerrado/apagado, la geolocalización informa la ubicación del vehículo, mientras que el control de las alarmas se refiere a la activación o desactivación de las mismas.

## 3.3. Arquitectura del sistema multi-agente propuesto

En esta sección se presenta el análisis y diseño del sistema multi-agente para evitar el olvido de bebés en automóviles, para hacer referencia a éste en los diagramas se utilizan las siglas SDBAB (Sistema detector de bebé a bordo). La sección de casos de uso corresponde al análisis del sistema, mientras que las subsecuentes establecen el diseño de mismo. El desarrollo del modelado se llevó a cabo con la herramienta de modelado INGENIAS [29]. Para consultar la notación de la herramienta de modelado, referirse al apéndice A.

### 3.3.1. Casos de uso

Para dar solución a la problemática planteada, se identificaron tres casos de uso principales: detectar presencia, monitorizar automóvil y enviar alertas, asociados a un actor principal definido como coordinador. La Fig. 3.2 muestra el diagrama de casos de uso.

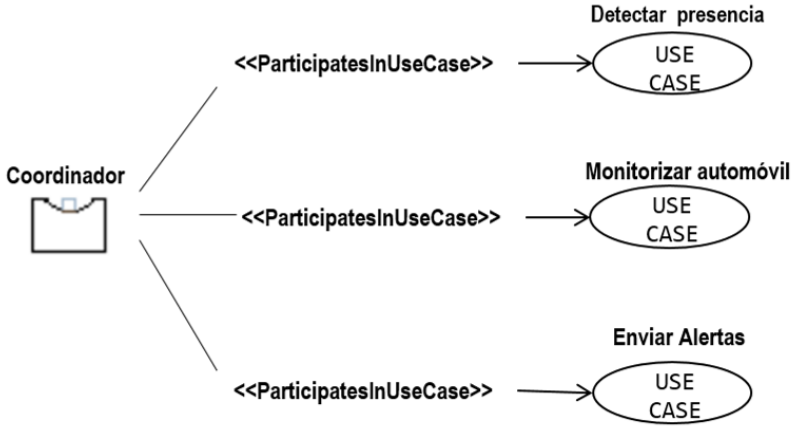



Figura 3.2: Casos de uso del sistema detector de bebé a bordo (SDBAB).

Las Tablas 3.1, 3.2 y 3.3, mostradas a continuación, contienen la especificación de los casos de uso.

| NOMBRE | Detectar presencia. |
|---|---|
| DESCRIPCIÓN | Determina si hay un bebé a bordo del automóvil. |
| ACTORES | Coordinador. |
| PRECONDICIONES | Motor del automóvil apagado. |
| FLUJO NORMAL | 1. El sensor de presión recibe una lectura $> 0$.<br>2. El coordinador recibe una lectura de peso implicando presencia de un bebé en el automóvil |
| FLUJO ALTERNATIVO | 1. No se detecta peso. |

Tabla 3.1: Caso de uso: detectar presencia.

| NOMBRE | Monitorizar automóvil. |
|---|---|
| DESCRIPCIÓN | Establece el tipo de alerta en función de lo percibido en el entorno. |
| ACTORES | Coordinador. |
| PRECONDICIONES | Motor del automóvil apagado. |
| FLUJO NORMAL | 1. El coordinador recibe las lecturas del entorno: peso, temperatura, condiciones del automóvil. En función de la combinación de estados se determina un tipo de alerta. |

Tabla 3.2: Caso de uso: monitorizar automóvil.

| NOMBRE | Enviar alertas. |
|---|---|
| DESCRIPCIÓN | Envía alertas en función de las lecturas del medio ambiente. Éstas son: activar alarma interna, enviar alerta nivel 1, enviar alerta nivel 2, alerta a policía preventiva. |
| ACTORES | Coordinador. |
| PRECONDICIONES | Motor del automóvil apagado. |
| FLUJO NORMAL | 1. En función de las condiciones monitorizadas el coordinador determina la alerta adecuada.<br>1.1 Cuando el motor está apagado, la puerta del conductor abierta y hay un registro de peso, el coordinador inicia un cronómetro con un tiempo de 3 minutos, activa la alarma interna y bloquea el seguro de las puertas.<br>1.2 El coordinador recibe un registro de peso, el motor sigue apagado y se agota el tiempo del cronómetro. Se activa la alerta nivel 1 y se inicia la medida temperatura.<br>1.3 El coordinador recibe registro de peso, el motor sigue apagado y se registra una de temperatura de riesgo (30°C o más). Se inicia alerta nivel 2, se enciende alarma externa y se envía alerta con la posición de geolocalización a policía preventiva. |

Tabla 3.3: Caso de uso: enviar alertas.

### 3.3.2. Modelo de objetivos y tareas

A partir de la definición de los casos de uso, se determinaron los objetivos que debe cumplir el sistema. En la Fig. 3.3 se exponen los tres objetivos principales del SDBAB: informar presencia del bebé, monitorizar condiciones del automóvil y enviar alerta.

- Informar presencia del bebé. Tiene como finalidad establecer a través de un sensor de peso si se está transportando un bebé en el automóvil.
- Monitorizar condiciones del automóvil. Se refiere a la vigilancia del estado de factores críticos para determinar los niveles de riesgo dentro del automóvil, así como a las acciones asociadas para prevenir o avisar el olvido de un bebé. Los factores críticos son aquéllos que se asocian al estado del motor, de la

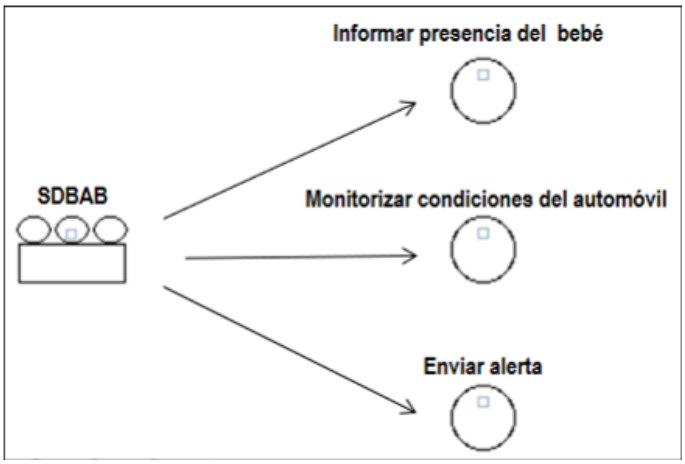



Figura 3.3: Diagrama de objetivos del sistema detector de bebé a bordo.

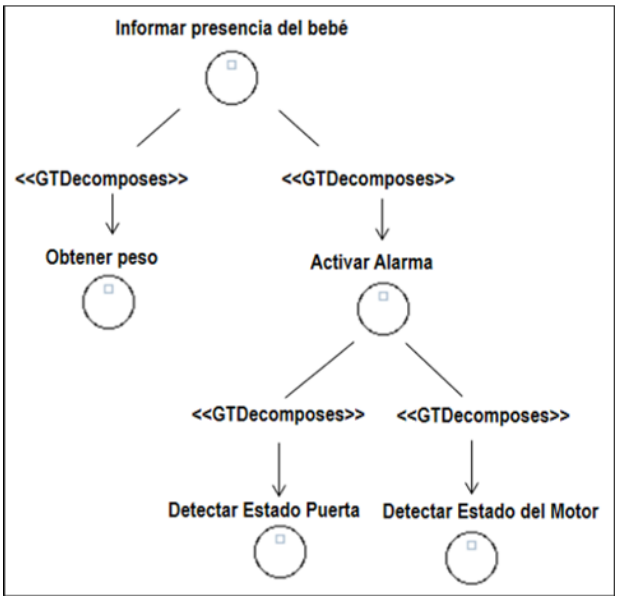



Figura 3.4: Descomposición del objetivo informar presencia del bebé.

puerta del conductor y, al incremento de la temperatura en el tiempo.

- Enviar alerta. Establece la comunicación con el usuario para avisar la presencia de un bebé en el automóvil apagado y sin conductor, en los distintos niveles de urgencia definidos.

El objetivo informar presencia del bebé, se descompone en los sub objetivos obtener peso y activar alarma. Asimismo, el sub objetivo activar alarma se descompone en detectar estado puerta y detectar estado del motor. El diagrama de descomposición de este objetivo se muestra en la Fig. 3.4.

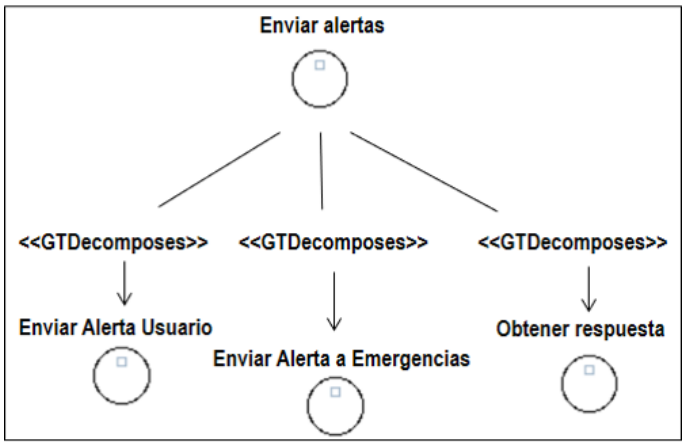



Figura 3.5: Descomposición del objetivo enviar alertas.

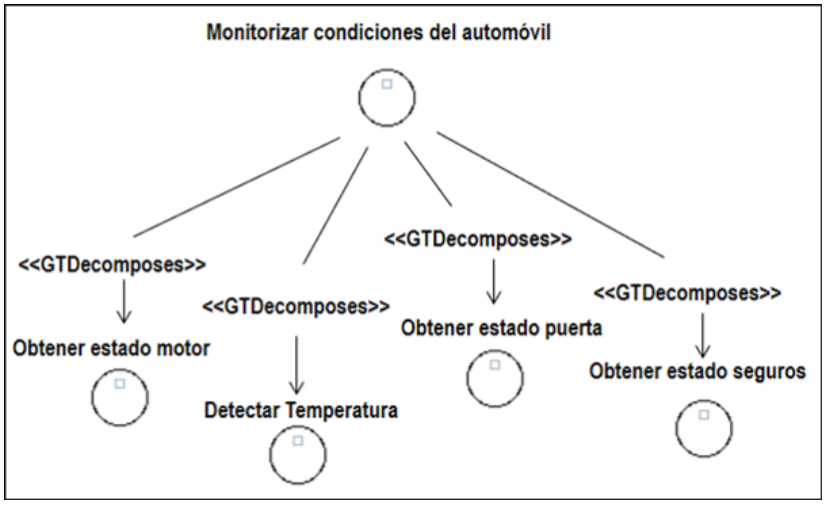



Figura 3.6: Descomposición del objetivo monitorizar condiciones del automóvil.

El objetivo enviar alertas se descompone en los sub objetivos enviar alerta usuario, enviar alerta a emergencias y obtener respuesta. El diagrama de descomposición de este objetivo se muestra en la Fig. 3.5.

Por último, el objetivo Monitorizar condiciones del automóvil se muestra en la Fig. 3.6 y se descompone en los sub objetivos obtener estado del motor, detectar temperatura, obtener estado puerta y obtener estado seguros.

### 3.3.3. Modelo de organización

El modelo de organización muestra una visión general del sistema, en él se representan los objetivos y grupos de agentes que componen el sistema. Como se observa

en la Fig. 3.7, el modelo de organización del SDBAB expone los objetivos que persigue el sistema. Como se describió en la sección previa los objetivos son tres: informar presencia de bebé, monitorizar condiciones del automóvil y enviar alertas.

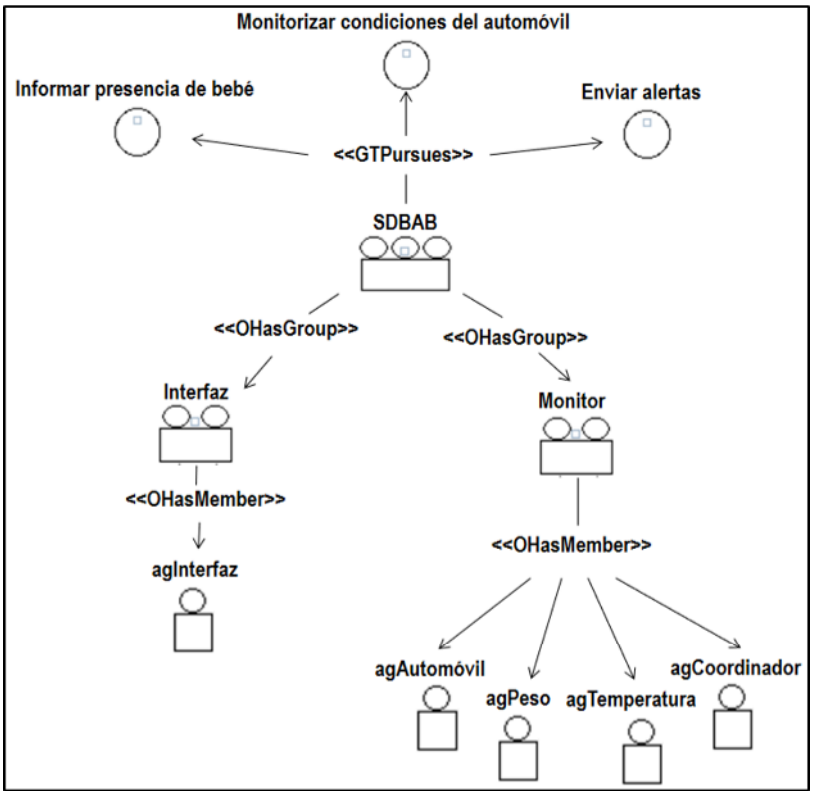



Figura 3.7: Modelo de organización.

El SDBAB se compone de dos grupos de agencias, la agencia interfaz y la agencia monitor. La agencia interfaz cumple con la función de establecer la comunicación con el exterior, bien sea el usuario o a línea de emergencia. Por otra parte, la agencia monitor se encarga de percibir las condiciones del automóvil y determinar el curso de acción del sistema.

La agencia interfaz se compone de un agente denominado agInterfaz, mientras que la agencia monitor agrupa cuatro agentes: agAutomóvil, agPeso, agTemperatura y agCoordinador. En el modelo de agentes que se presenta en la siguiente sección, se detallan las tareas asociadas a cada uno de los agentes.

### 3.3.4. Modelo de agentes

El modelo de agente muestra los roles que desempeña cada uno de ellos. Para el SDBAB se requieren cinco agentes, coordinador, interfaz, agTemperatura, agAutomóvil y agPeso, mismos que están asociados a los roles monitor, comunicador, detectorTemp, operadorAutomovil y detectorPeso respectivamente. La Fig. 3.8 muestra en el diagrama de agentes las asociaciones entre cada agente y el rol que lleva a cabo.

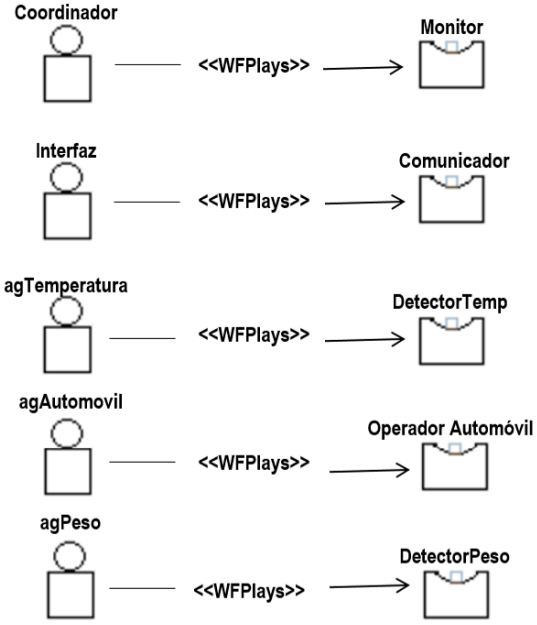



Figura 3.8: Modelo de agente.

La asociación de las tareas a cada rol se muestra en la Fig. 3.9, mientras que la descripción de cada una de ellas se presenta a continuación.

El rol Monitor se compone de las siguientes tareas:

- Evaluar riesgo. Esta tarea define el nivel de riesgo que se está manejando. Siempre que se detecte presencia de un bebé en el automóvil se genera una situación de riesgo. Los niveles de riesgo son los siguientes:

  Nivel 1, es el nivel de menor riesgo se da al apagar el motor del automóvil y abrir la puerta del conductor.

Nivel 2, se da al agotarse el intervalo de tiempo definido para retirar al bebé del automóvil.

Nivel 3, es el último nivel de riesgo y se produce cuando se detecta una temperatura de riesgo en el automóvil.

- Comunicar riesgo. Esta tarea es la responsable de enviar el tipo de alerta que comunicará la interfaz.
- Solicitar bloqueo de seguros. Envía la instrucción de impedir el cierre del automóvil.
- Solicitar inicio de alarma. Esta tarea envía al automóvil la instrucción de inicio de alarma asociada al tipo de riesgo establecido en la evaluación de riesgo.

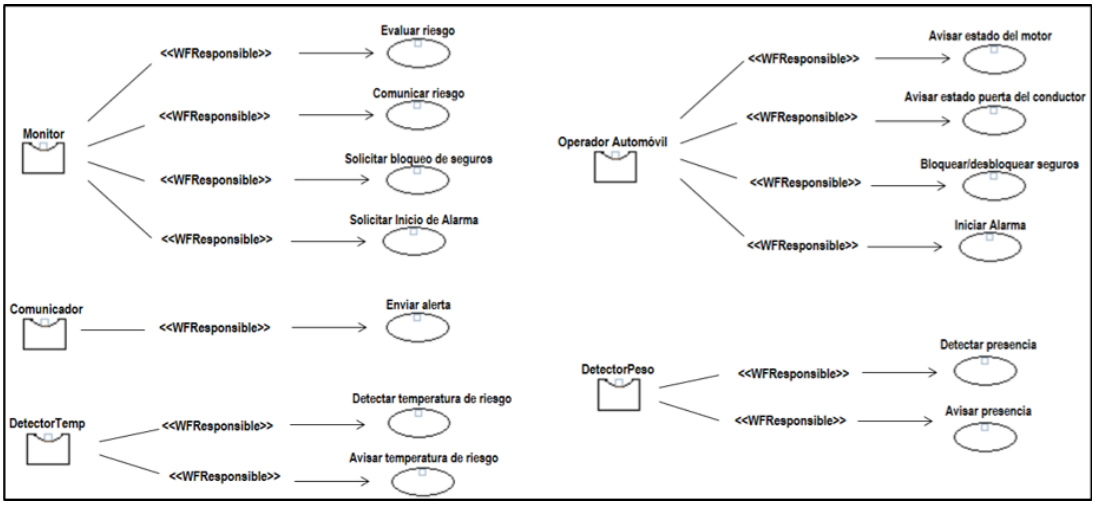



Figura 3.9: Descripción de roles del SDBAB.

El rol comunicador se compone de la tarea Enviar Alerta, que tiene como finalidad transmitir el aviso de riesgo al usuario. El rol DetectorTemp está formado por las tareas Detectar temperatura de riesgo y Avisar temperatura de riesgo descritas a continuación.

- Detectar temperatura de riesgo. Esta tarea es responsable de medir la temperatura del automóvil y detectar cuando se alcanza una temperatura de riesgo.

- Avisar temperatura de riesgo. Su función es informar que se ha identificado una temperatura de riesgo.

Otro de los roles que se manejan en el SDBAB es el de Operador Automóvil, éste se compone de cuatro tareas, mismas que se detallan a continuación:

- Avisar estado del motor, cuya función es la de notificar cuando el motor está apagado o se ha vuelto a encender.
- Avisar estado puerta conductor, informa si la puerta del conductor está abierta.
- Bloquear/desbloquear seguros. Esta tarea ejecuta el bloqueo o desbloqueo de seguros para impedir el cierre del vehículo o devolver el control del cierre al automóvil.
- Iniciar Alarma. Su función es iniciar la alarma que se le requiera, ésta puede ser alarma interna o alarma externa.

Finalmente, el rol Detector Peso, está conformado por dos tareas, Detectar presencia y Avisar presencia, su descripción es la siguiente:

- Detectar presencia. Su finalidad es la de comprobar si hay peso en la silla de bebé. Cuando se detecta peso, se asume que hay un bebé en el automóvil.
- Avisar presencia. Esta tarea es la encargada de comunicar cuando se ha detectado presencia.

### 3.3.5. Modelo de interacciones

El modelo de interacciones se establece con el fin de conocer la colaboración entre los roles que componen un sistema. En él se representa el intercambio de conocimiento entre los agentes, puesto que cada rol es ejecutado por un agente. En la Fig. 3.10 se puede observar el diagrama de interacciones. Como se observa,

en él interactúan los cinco roles definidos previamente, a través de dos principales interacciones que son la base del funcionamiento del SDBAB.

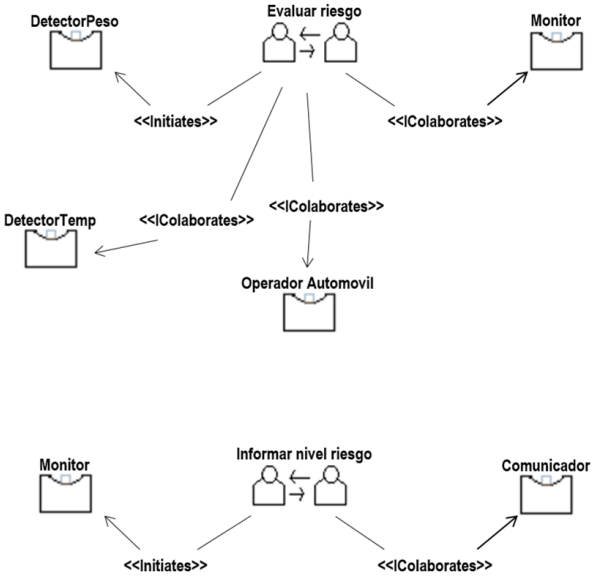



Figura 3.10: Modelo de interacciones.

La primera interacción es Evaluar riesgo, en ella colaboran los siguientes roles: Detector Peso, Detector Temp, Operador automóvil y Monitor. Al colaborar entre ellos se pueden establecer los diversos tipos de riesgo que maneja el SDBAB. Previamente, en el modelo de agentes se describieron cada uno de ellos.

La segunda interacción busca con la colaboración entre agentes llegar a informar el nivel de riesgo, como su nombre lo indica. En ella colaboran dos roles, el Monitor y el Comunicador, quienes al interactuar, cumplirán con la función de alertar al usuario o a los servicios de emergencia sobre el olvido del bebé en el automóvil.

### 3.3.6. Modelo del entorno

El modelo del entorno que se muestra en la Fig. 3.11, muestra la relación que existe entre las tareas definidas en el SDBAB y los recursos que utiliza el mismo para ofrecer su funcionalidad.

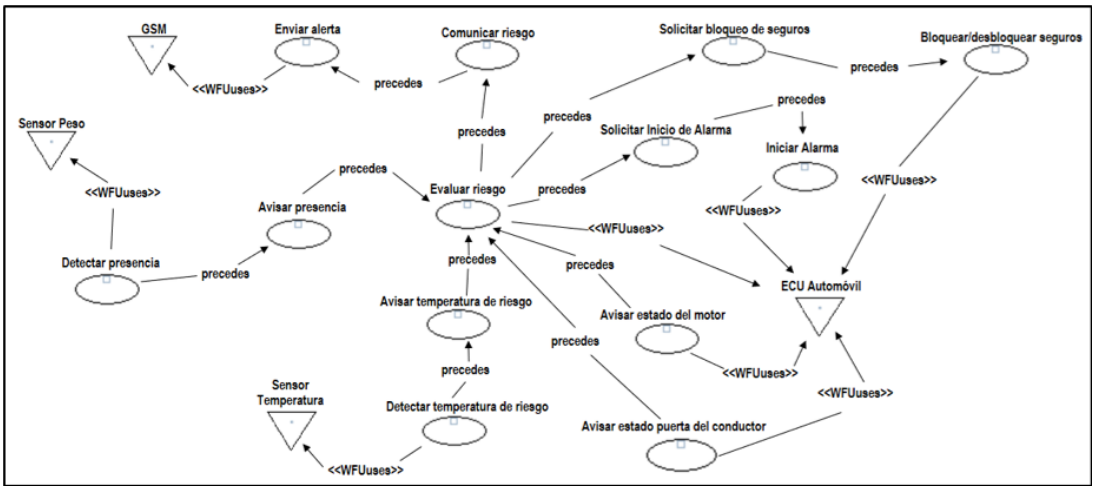



Figura 3.11: Modelo del entorno.

Los recursos identificados para cumplir con las tareas definidas son:

- Sensor Peso. Un sensor de peso colocado en la silla de bebé para detectar presencia.
- Sensor Temperatura. Un sensor de temperatura para evaluar esta condición ambiental y establecer el riesgo de hipertermia.
- GSM. Un medio de comunicación a la red celular para alertar al usuario y al servicio de emergencias.
- ECU Automóvil. El acceso a la unidad de control electrónica del automóvil para establecer cuando el automóvil está apagado y sin conductor, así como el acceso a manipular el estado de los seguros, las alarmas del automóvil y obtener la ubicación del automóvil.

## 3.4. Modelo de notificaciones

En situaciones de riesgo como la que se plantea, es importante percibir el entorno para definir el tipo de alerta que se generará. Con la finalidad de establecerlas, se adaptó el modelo de notificaciones CANoE (Context-Aware Notifications for Critical Environments) [15]. Este modelo se creó para manejar notificaciones cuando

se trabaja con ambientes críticos y permite crear la alerta adecuada a través de la adquisición del contexto, la prioridad que tendrá el mensaje y el establecimiento del mecanismo de notificación.

Al adaptar este modelo en el SDBAB, se estableció como entorno a monitorizar el Automóvil, donde se identifican al emisor del mensaje y al receptor. Se define como emisor al bebé a bordo del automóvil, puesto que es quien puede entrar en situación de riesgo. El receptor en este caso sería el conductor del automóvil, quien funge en este caso como el responsable del bebé que está siendo transportado. Por otra parte, se identificaron como receptores fuera del automóvil, el mismo conductor, los servicios de emergencia y las personas cercanas al automóvil que escuchen la alarma de robo del mismo. En la Fig. 3.12 que se muestra a continuación, se muestra el modelo de notificaciones.

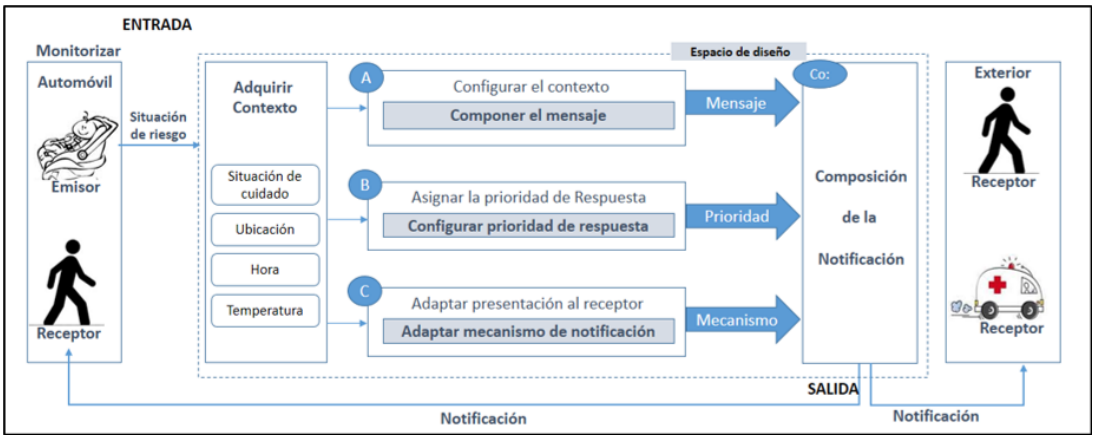



Figura 3.12: Modelo de notificaciones.

Al producirse una situación de riesgo, se genera una notificación atendiendo tres apartados. Primero, la composición del mensaje en función del contexto que se percibió en ese momento, es decir, la situación de cuidado en la que se encuentra el menor, su ubicación, la temperatura del automóvil y la hora a la que se registra el suceso (A). Enseguida se deberá asignar la prioridad de respuesta al mensaje, ésta se omitiría para la primera notificación, se asignaría una prioridad alta para la primera notificación al *smartphone* y una prioridad de extrema urgencia para la segunda notificación al *smartphone*, así como a los servicios de emergencia (B).

El tercer apartado determina el mecanismo para transmitir la notificación, es

decir, la forma en la que se presentará el mensaje al receptor. En este caso se ofrecen tres medios de exposición del mensaje: la alarma interna del automóvil, el *smartphone* y la alarma externa o alarma de robo. Cuando el conductor apaga el motor y abre la puerta del automóvil, se envía la primera notificación audible, es decir la activación de la alarma interna. En este caso no se envía el contexto al receptor pues se asume que el bebé aún no está en situación de riesgo (C).

Cuando han transcurrido tres minutos y el bebé continúa en el automóvil, se envía la primera notificación al smartphone del conductor del auto con la prioridad asociada descrita anteriormente. Finalmente, al registrarse una temperatura de riesgo, se enviarían las notificaciones al conductor del automóvil a través del *smartphone* del mismo, así como al servicio de emergencias. De la misma manera, de forma audible se notificaría en este caso a las personas cercanas al automóvil a través de la alarma de robo del automóvil.

# Capítulo 4

# Implementación

## 4.1. Introducción

En este capítulo se presenta el desarrollo del modelo multi-agente con base en lo establecido en la sección anterior. En él se describen los comportamientos añadidos a cada agente, así como las acciones que efectúan para cumplir con sus objetivos. Se presentan los mensajes que intercambian entre agentes para su adecuada operación, evidenciando de esta forma la comunicación con los demás agentes involucrados en el desarrollo.

La implementación del modelo se llevó a cabo a través de una simulación utilizando la librería de JADE versión 4.4.0, sobre el IDE de IntelliJ IDEA versión 2016.1.48. Los parámetros iniciales para la construcción de los agentes se presentan en la primera parte del capítulo. La construcción de cada uno de los agentes se detalla en la segunda sección.

## 4.2. Parámetros iniciales

La validación del modelo propuesto se desarrolló en el IDE de IntelliJ IDEA versión 2016.1.48, a través de una simulación utilizando la librería de JADE específica para sistemas multi-agente. La librería incluye una interfaz gráfica y un analizador

de mensajes.

Para lanzar la interfaz gráfica con la simulación, se especifica en la clase main la opción “-gui”, mientras que el analizador de mensajes se ejecuta con la instrucción sniffer:jade.tools.sniffer.Sniffer. Se añadirán como parámetros los agentes que se van a monitorizar por medio de las herramientas proporcionadas por la librería (Fig. 4.1).

```
    jade.Boot.main(new String[] {
            "-gui",
            "sniffer:jade.tools.sniffer.Sniffer;AgAutomovil:" + AgAutomovil.class.getName() +
                    ";AgCoordinador:" + AgCoordinador.class.getName() + ";AgPeso:" + AgPeso.class.getName() +
                    ";AgInterfaz:" + AgInterfaz.class.getName() + ";AgTemperatura:" +
                    AgTemperatura.class.getName()
    });
}
```

Figura 4.1: Parámetros para ejecutar la interfaz gráfica de JADE.

Al lanzarse la interfaz gráfica se muestra del lado izquierdo de la interfaz, la plataforma de agentes que se está utilizando, en este caso se observa la dirección IP y el puerto 192.168.3.73:1099, (Fig. 4.2). En esta plataforma se crean los contenedores donde residen los agentes, para efectos de la simulación, se utiliza solo el contenedor por defecto que es el contenedor principal o ‘Main-Container’, en la Fig. 4.2 se muestran resaltados los agentes en ejecución necesarios para este modelo: AgAutomovil, AgCoordinador, AgInterfaz, AgTemperatura y AgPeso.

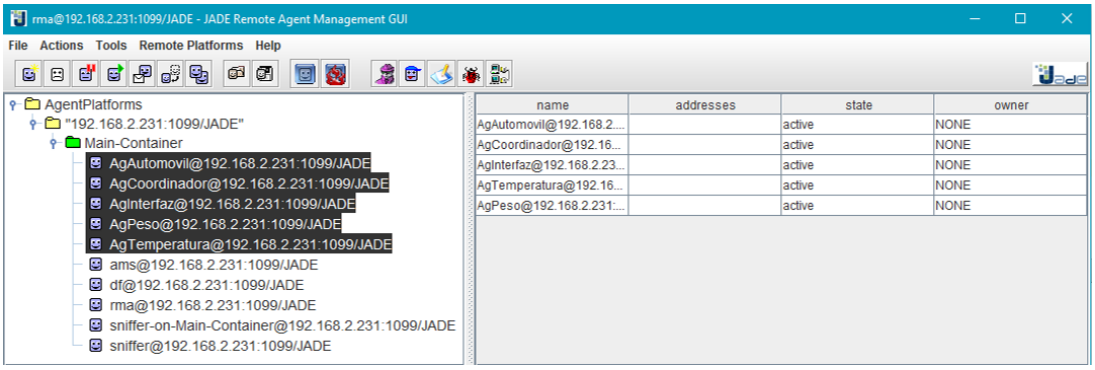



Figura 4.2: Interfaz gráfica de JADE.

Además de los agentes que se crean para el funcionamiento del modelo, se crean por defecto varios agentes que coordinan la plataforma y sus interacciones, éstos viven en el contenedor principal y sus funciones son las siguientes:

- AMS. (Agent Management System), agente que además de supervisar la plataforma, provee el servicio de administración de agentes, así como el de páginas blancas. Es el encargado de asignar un identificador a cada agente, a este identificador se le conoce como AID o Agent Identifier.

- DF. (Directoy Facilitator), agente que provee el servicio de páginas amarillas. Lo utilizan los agentes para registrar sus servicios o buscar los servicios que ofrecen los demás agentes.

- RMA. (Remote Monitoring Agent), agente que provee una interfaz visual de administración de la plataforma de agentes. Se suscribe al AMS para ser notificado de todos los eventos a todos los niveles de la plataforma.

- Sniffer. (Sniffer Agent), agente utilizado para documentar conversaciones entre los agentes, al igual que el RMA se suscribe al AMS para ser notificado de todos los eventos. A través de su interfaz gráfica permite depurar las actividades de los agentes y visualizar los mensajes específicos que se intercambian entre uno o un grupo de agentes.

Se parte de los tres posibles casos modelados en el capítulo previo, éstos son:

- Caso 1 – no se olvida al bebé.

- Caso 2 – se olvida al bebé, pero se retira del automóvil antes de presentar riesgo de hipertermia.

- Caso 3 – en el que se olvida al bebé y se notifica al servicio de emergencias además del responsable.

En estos tres casos, el denominador común - mismo que se considera el punto de partida de la simulación- son los parámetros iniciales:

- se detecta un bebé a bordo de un automóvil.

- el motor de éste está apagado.

- se abre la puerta del conductor.

En la simulación, estos valores los solicita el agente AgCoordinador que funge como coordinador, a los agentes correspondientes AgPeso y AgAutomovil respectivamente, obteniendo una respuesta arbitraria con los valores indispensables para dar inicio al proceso.

Cada agente se implementa como una clase de Java. La implementación de los agentes inteligentes, sus interacciones, tareas y comportamientos se describen a continuación.

## 4.3. Construcción de agentes

A todos los agentes se les ha añadido un tiempo de espera de 3 segundos para iniciar su ejecución con la finalidad de cargar las herramientas gráficas de monitorización y análisis de paquetes.

### 4.3.1. AgAutomovil

Al agente AgAutomovil, se le ha añadido un comportamiento cíclico con la finalidad de simular la monitorización constante de las funcionalidades del automóvil consideradas en este proyecto. Un comportamiento cíclico, en este caso, se refiere a que se repetirá constantemente durante la vida del agente. Al igual que se hace con los demás agentes, al crearse el agente, se muestra un mensaje de inicialización del mismo (Fig. 4.3).

```
public void setup() {
    doWait(3000);
    System.out.println("Agente >> " + getLocalName() + " iniciado.");
    addBehaviour(new CyclicBehaviour(this) {
```

Figura 4.3: Comportamiento cíclico del agente AgAutomovil.

A través del comportamiento cíclico del agente, se consigue que éste permanezca a la escucha de las necesidades del coordinador, de tal forma que ejecute las instrucciones pertinentes según sea el caso. Las acciones que ejecuta este agente son:

activar la alarma interna, detener la alarma interna, impedir el cierre de puertas, permitir el cierre de puertas, iniciar la alarma externa y detener la alarma externa.

A continuación (Fig. 4.4) se presentan las acciones que se ejecutan en función de las necesidades que define el agente coordinador y que se reciben como un mensaje ACL de tipo INFORM.

```
} else if (coorMsg != null && coorMsg.getSender().getLocalName().equals("AgCoordinador") &&
           coorMsg.getPerformative() == ACLMessage.INFORM) {
   if (coorMsg.getContent().equals("Active intern alarm")) {
       setInterAlarmState(true);
       System.out.println("AgAutomovil activó alarma interna \n");
   } else if (coorMsg.getContent().equals("Unlock the doors")) {
       unlock();
       System.out.println("AgAutomovil desactivó los seguros del auto \n");
   } else if (coorMsg.getContent().equals("Stop intern alarm")) {
       setInterAlarmState(false);
       System.out.println("AgAutomovil desactivó la alarma interna \n");
   } else if (coorMsg.getContent().equals("Lock the car")) {
       lock();
       System.out.println("AgAutomovil bloqueó los seguros \n");
   } else if (coorMsg.getContent().equals("Start car alarm")) {
       alarm = true;
       System.out.println("AgAutomovil inicia alarma  ** Alarma externa ACTIVA ** \n");
   } else if (coorMsg.getContent().equals("Stop car alarm")) {
       alarm = true;
       System.out.println("AgAutomovil desactivó alarma externa");
   }
}
```

Figura 4.4: Acciones que se ejecutan por instrucción de AgCoordinador.

Otras de las funciones de este agente, son percibir los estados de la puerta del conductor, del motor y de la ubicación del automóvil. La definición de éstas se presenta a continuación (Fig. 4.5).

```
// Envia mensaje con el estado de la puerta del auto a AgCoordinador
private void sendDoorStateMsg() {
    ACLMessage doorStateMsg = new ACLMessage(ACLMessage.INFORM);
    doorStateMsg.addReceiver(new AID("AgCoordinador", false));
    doorStateMsg.setContent(String.valueOf(doorState));
    send(doorStateMsg);
}
// Envia mensaje con el estado del motor del auto a AgCoordinador
private void sendEngineStateMsg() {
    ACLMessage infoResp = new ACLMessage(ACLMessage.INFORM);
    infoResp.addReceiver(new AID("AgCoordinador", false));
    infoResp.setContent(Boolean.toString(engineState));
    send(infoResp);
}
// Envia mensaje con la ubicación del auto a AgCoordinador
private void sendLocation() {
    ACLMessage infoResp = new ACLMessage(ACLMessage.INFORM);
    infoResp.addReceiver(new AID("AgCoordinador", false));
    infoResp.setContent("Calle 1 y S.Campoy, Hillo. Son.");
    send(infoResp);
}
```

Figura 4.5: Envío de información del auto a AgCoordinador.

Como se puede observar, para comunicar estos datos, se definieron tres métodos: sendDoorStateMsg(), sendEngineStateMsg() y sendLocation(), cuya función es la de establecer la comunicación con el agente AgCoordinador que funge como coordinador en la simulación.

La comunicación se establece a través de un mensaje ACL de tipo INFORM y recoge en los dos primeros métodos, el valor de las variables doorState y engineState respectivamente. Para comunicar la ubicación del automóvil, sin embargo, se crea el contenido del mensaje arbitrariamente, puesto que no influye en el desarrollo de la simulación.

### 4.3.2. AgInterfaz

El agente AgInterfaz es el responsable de establecer la comunicación con el usuario a través de los distintos medios. Para implementarlo, se le añadió un comportamiento cíclico que permita estar a la espera de las indicaciones del agente AgCoordinador.

Las instrucciones se reciben por medio de un mensaje ACL en el que se obtiene el tipo de alerta que debe transmitir InterfazAgent. En los dos casos se compone el mensaje de acuerdo a lo establecido en el modelo de notificaciones (adaptación al modelo CANoE), difiriendo los mismos en el nivel de alerta que se le envía al usuario

y en los receptores del mensaje.

```
addBehaviour((CyclicBehaviour) () → {
        ACLMessage coorMsg = receive();
        if (coorMsg != null && coorMsg.getSender().getLocalName().equals("AgCoordinador") &&
                coorMsg.getPerformative() == ACLMessage.INFORM) {
            if (coorMsg.getContent().equals("Registrar presencia")) {
                System.out.println(getLocalName() +" >> Actualizando registro de presencia \n");
            } else if (coorMsg.getContent().equals("Alert level 1")) {
                message = true;
                composeMessage();
            } else if (coorMsg.getContent().equals("Alert level 2")) {
                message = false;
                composeMessage();
            }
        }
});
```

Figura 4.6: Acciones efectuadas por InterfazAgent.

Los tres casos posibles que se simulan, se notifican a AgInterfaz para emitir en los casos de riesgo, las alertas pertinentes. Se notifica también a este agente cada vez que se detecta un bebé a bordo del auto con el fin de alimentar un registro de presencia.

En el código anterior (Fig. 4.6) se puede observar el comportamiento cíclico añadido a InterfazAgent, así como la llamada al método composeMessage(). Este método obtiene a través del agente AgCoordinador la temperatura, ubicación y nivel de urgencia de la notificación y añade también la hora del suceso para armar el mensaje de notificación. El método composeMessage() es el que construye y envía las notificaciones a los receptores establecidos del dispositivo.

### 4.3.3. AgTemperatura

AgTemperatura es el agente responsable de monitorizar la temperatura del automóvil. Al igual que los demás agentes, al crearlo se muestra un mensaje de inicialización y se le añade un tiempo de espera para que tanto el analizador de paquetes como la interfaz gráfica puedan cargarse adecuadamente y registrar las comunicaciones del agente.

Para simular su actuación se le definió como un agente que se comporta cíclicamente. Realiza una acción repetitiva principal que es la de escuchar los requerimien-

tos de información del agente AgCoordinador y enviarle información en respuesta, (Fig. 4.7).

```
addBehaviour((CyclicBehaviour) () -> {
        ACLMessage tempResp = blockingReceive(requestTemplate);
        if (tempResp != null && tempResp.getSender().getLocalName().equals("AgCoordinador")) {
            temperature = new Random().nextInt((35 - 25) + 1) + 25;
            ACLMessage temp = new ACLMessage(ACLMessage.INFORM);
            temp.addReceiver(new AID("AgCoordinador", false));
            temp.setContent(Integer.toString(temperature));
            send(temp);
        }
});
```

Figura 4.7: Acciones efectuadas por AgTemperatura.

Se obtiene aleatoriamente el valor de la temperatura, mismo que se envía a AgCoordinador para que determine el riesgo de hipertermia. Este valor generado con la función Random().nextInt(), representa la temperatura en grados centígrados y es un entero aleatorio en el rango [25-35]. Se definió este rango para obtener valores en los que la temperatura no representa un riesgo de hipertermia [25-29] y valores que permitan simular un riesgo de hipertermia [30-35]. La comunicación se establece a través de un mensaje de tipo INFORM conforme a las especificaciones de comunicaciones entre agentes.

### 4.3.4. AgPeso

Al igual que los agentes anteriormente descritos, el agente AgPeso se comporta también de forma cíclica. Este comportamiento permite responder en cualquier momento a la petición de AgCoordinador del peso registrado en la silla del bebé. La Fig. 4.8 muestra el comportamiento de este agente.

```
addBehaviour(new CyclicBehaviour(this) {
    public void action() {
        ACLMessage wpReq = blockingReceive(requestTemplate);
        if (wpReq != null && wpReq.getSender().getLocalName().equals("AgCoordinador") &&
                wpReq.getPerformative() == ACLMessage.REQUEST) {
            ACLMessage w = new ACLMessage(ACLMessage.INFORM);
            w.addReceiver(new AID("AgCoordinador", false));
            if (wpReq.getContent().equals("Give me the weight")) {
                weight = (float) (Math.floor((new Random().nextFloat()*(15 - 5) + 5)*100)/100);
                w.setContent(Float.toString(weight));
            } else if (wpReq.getContent().equals("Give me the weight 1")) {
                weight = (float) (Math.floor((new Random().nextFloat()*(15 - 10) + 10)*100)/100);
                w.setContent(Float.toString(weight));
            }
            send(w);
        }
    }
});
```

Figura 4.8: Acciones efectuadas por AgPeso.

La petición la recibe AgPeso como un mensaje ACL de tipo REQUEST, en el que se pueden presentar dos situaciones:

- Una de ellas se da la primera vez que la simulación requiere detectar presencia para poder generar uno de los tres casos, en este caso se recibe el mensaje "Give me the weight 1", al cual el agente responderá con un valor arbitrario mayor a 10.
- La otra genera un valor aleatorio que permite simular si el bebé se retira al término del trayecto o si se olvida en el auto y se retira a la primera o a la segunda alerta.

El agente AgPeso, responde a este requerimiento con un mensaje de tipo INFORM cuyo contenido está en función de las situaciones anteriores. El contenido es un número aleatorio generado por medio de la función Random().nextFloat(). Para el primer caso se establece un rango de [10-15] para generarlo, el cual forzosamente devolverá un valor que el AgCoordinador interpretará como presencia en el automóvil.

Para el segundo caso se estableció un rango de [5-15], que genera valores que el AgCoordinador podrá interpretar como presencia o ausencia de bebé. Los valores [10-15] son los que se interpretan como presencia, mientras que los que se generan en el rango [5-9] se interpretan como ausencia de bebé.

### 4.3.5. AgCoordinador

El agente AgCoordinador es el que coordina todas las acciones e inicia las conversaciones con los demás agentes para arrancar la simulación. Sus acciones están implementadas a través de un comportamiento de máquina de estados finitos, que permite descomponer las acciones de este agente en secciones mucho más manejables.

La máquina de estados finitos se crea a partir de la clase FSMBehaviour que provee métodos para representar sub-comportamientos como estados de la máquina. Para añadir comportamientos al objeto máquina de estados finitos (FSM), se utiliza el método registerState().

Se definieron 15 estados para representar los comportamientos de la FSM cada uno de los cuales recibe como parámetros las tareas que lleva a cabo el estado, así como el nombre con el que se identifica dicho estado. Para representar cada estado se utilizan las letras del alfabeto, facilitando de esta forma, el seguimiento a las secuencias de las transiciones. La Fig. 4.9 muestra la representación gráfica de la tabla de transiciones, por medio de un autómata finito.

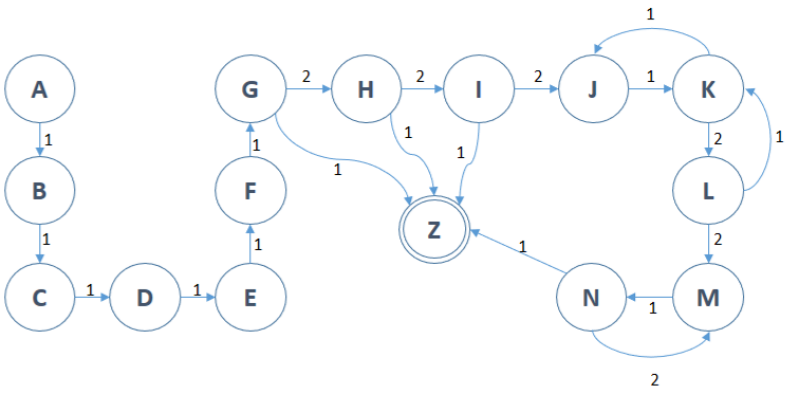



Figura 4.9: Representación del comportamiento de AgCoordinador como autómata finito.

Cada comportamiento del objeto FSM, se inicia con el método action() y finaliza con el método onEnd(), en función del valor que devuelve onEnd(), se ejecuta el siguiente estado. Se definieron transiciones por defecto con el método registerDefaultTransition(), es decir, las que al concluirse conducen siempre al mismo estado. Para aquellas transiciones que pueden arrojar distintos valores al ejecutar el método

onEnd(), se utilizó registerTransition(), indicando así los estados que le siguen.

A todos los estados que se definieron, se les trató como un comportamiento 'One ShotBehaviour', de forma que se ejecutan una sola vez y de forma ininterrumpida. El siguiente código (Fig. 4.10) muestra los cambios de estado definidos en la tabla de transiciones.

```
fsm.registerDefaultTransition("A", "B");
fsm.registerDefaultTransition("B", "C");
fsm.registerDefaultTransition("C", "D");
fsm.registerDefaultTransition("D", "E");
fsm.registerDefaultTransition("E", "F");
fsm.registerDefaultTransition("F", "G");
fsm.registerTransition("G", "H", 2);
fsm.registerTransition("G", "Z", 1);
fsm.registerTransition("H", "Z", 1);
fsm.registerTransition("H", "I", 2);
fsm.registerTransition("I", "Z", 1);
fsm.registerTransition("I", "J", 2);
fsm.registerDefaultTransition("J", "K");
fsm.registerTransition("K", "L", 2);
fsm.registerTransition("K", "J", 1);
fsm.registerTransition("L", "K", 1);
fsm.registerTransition("L", "M", 2);
fsm.registerDefaultTransition("M", "N");
fsm.registerTransition("N", "M", 2);
fsm.registerTransition("N", "Z", 1);
```

Figura 4.10: Transiciones de la máquina de estados finitos de AgCoordinador.

Los métodos que se utilizan para el desarrollo de funcionalidades de este agente son: weightReq(), engineStateMsg(), reqLocation(), lockMsg(), unlockMsg(), sendTemperature(), stopTimerMsg(), level1Alert() y level2Alert(). Los tres primeros son mensajes de solicitud de información para los agentes AgPeso y AgAutomovil. Estos tres métodos solicitan a través de un mensaje ACL de tipo REQUEST, el peso detectado, el estado del motor y la ubicación del automóvil respectivamente.

Los métodos restantes, establecen la comunicación con los demás agentes por medio de un mensaje ACL tipo INFORM, solicitando según sea el caso:

- Que se impida o se habilite el cierre del automóvil.
- Informar al AgInterfaz los datos necesarios para crear las alertas.
- Detener el cronómetro que controla la alarma interna.

- Que den inicio las alertas (uno o dos según sea el caso).

Es este agente quien, por medio de la colaboración con los demás agentes, se encarga de prevenir el olvido de un bebé en el automóvil. Esto se logra a través de la comunicación entre todos los agentes que viven en el contenedor, misma que se da ejecutando cada estado de la máquina. A continuación, se profundiza en los estados decisivos que resultan en los tres casos simulados.

El estado "E" cuyo propósito es el de responder al hecho de percibir un bebé en el automóvil, muestra los mensajes que establece AgCoordinador con los demás agentes involucrados, para prevenir que el bebé se olvide a bordo del auto. En este estado, además de la composición de los mensajes de tipo INFORM que desencadenan las acciones del mismo AgCoordinador, de AgInterfaz y de AgAutomovil, se recibe también el estado de la puerta del conductor que es el último parámetro que condiciona el inicio de este comportamiento, (Fig. 4.11).

```
fsm.registerState((OneShotBehaviour) () → {
        System.out.println("Executing behaviour " + getBehaviourName());
        ACLMessage doorStateMsg = blockingReceive(informTemplate);
        if (doorStateMsg != null && doorStateMsg.getSender().getLocalName().equals("AgAutomovil")) {
            doorState = Boolean.parseBoolean(doorStateMsg.getContent());
            // Si el peso que se recibio anteriormente es mayor o igual a 10, el motor esta
            // apagado y la puerta abierta, se informa a AgCoordinador que inicie el timer
            if (weight >= 10 && engineState == false && doorState == true) {
                System.out.println("Coordinador recibio de >> " + doorStateMsg.getSender().getName() +
                        " puerta del conductor >>  Abierta \n");
                // Se envia y recibe el mensaje para iniciar el timer
                ACLMessage startTimerMsg = new ACLMessage(ACLMessage.INFORM);
                startTimerMsg.addReceiver(new AID("AgCoordinador", false));
                startTimerMsg.setContent("Start timer");
                send(startTimerMsg);
                ACLMessage resp = blockingReceive(informTemplate);
                if (resp != null && resp.getSender().getLocalName().equals("AgCoordinador")) {
                    System.out.println("Coordinador inicio el timer");
                    timer = true;
                }
                lockMsg();          // Se envia mensaje a AgAutomovil para impedir el cierre del auto
                sendTemperature();  // Se envía temperatura para composición del registro
                // Se envia mensaje a AgInterfaz para registro de actividad
                ACLMessage recordInternAlarm = new ACLMessage(ACLMessage.INFORM);
                recordInternAlarm.setContent("Registrar presencia");
                recordInternAlarm.addReceiver(new AID("AgInterfaz", false));
                send(recordInternAlarm);
                // Se envia mensaje a AgAutomovil para que inicie la alarma interna del auto
                ACLMessage startInternAlarm = new ACLMessage(ACLMessage.INFORM);
                startInternAlarm.setContent("Active intern alarm");
                startInternAlarm.addReceiver(new AID("AgAutomovil", false));
                send(startInternAlarm);
            }
        }
}, "E");
```

Figura 4.11: Estado "E", se presentan las condiciones que originan el caso 1.

Cuando el agente AgCoordinador percibe a través de sus agentes colaboradores

que el ambiente no se ha modificado, en este caso, el motor del automóvil continúa apagado, efectúa las siguientes acciones:

- Detiene el cronómetro que maneja la alarma interna.
- Informa a través de un mensaje tipo INFORM a AgAutomovil que debe apagar la alarma interna.
- Ejecuta el método level1Alert que indica a AgInterfaz que debe crear una alerta de nivel 1 y comunicarla al usuario.
- Solicita a AgPeso detección de peso para valorar presencia.

```
fsm.registerState(new OneShotBehaviour() {
    private int exitValue;
    public void action() {
        System.out.println("Executing behaviour " + getBehaviourName());
        ACLMessage engineStateResp = blockingReceive(informTemplate);
        if (engineStateResp != null && engineStateResp.getSender().getLocalName().equals("AgAutomovil")) {
            engineState = Boolean.parseBoolean(engineStateResp.getContent());
            if (engineState == false) {
                stopTimerMsg();

                System.out.println("Coordinador recibió de >> " + engineStateResp.getSender().getName() +
                                   " estado del motor >> apagado \n");

                ACLMessage stopInternAlarm = new ACLMessage(ACLMessage.INFORM);
                stopInternAlarm.addReceiver(new AID("AgAutomovil", false));
                stopInternAlarm.setContent("Stop intern alarm");
                send(stopInternAlarm);

                level1Alert();      // Se indica a IntefazAgent que envíe el mensaje de alerta

                weightReq();        // Se verifica presencia con AgPeso
                exitValue = 2;
            } else {
                System.out.println("Coordinador recibió de >> " + engineStateResp.getSender().getName() +
                        " estado del motor >> encendido \n");
                exitValue = 1;
            }
        }
    }
    public int onEnd() { return exitValue; }
}, "H");
```

Figura 4.12: Estado “H”, se presentan las condiciones que originan el caso 2.

La Fig. 4.12 muestra la implementación del estado “H”, donde se observan las acciones descritas anteriormente. Estas acciones corresponden a las tareas que efectúa AgCoordinador cuando se presenta el caso 2 de la simulación, así como las condiciones que se dan para que esto suceda.

```
fsm.registerState(new OneShotBehaviour() {
    private int exitValue;
    public void action() {
        System.out.println("Executing behaviour " + getBehaviourName());
        ACLMessage tempResp = blockingReceive(informTemplate);
        if (tempResp != null && tempResp.getSender().getLocalName().equals("AgTemperatura")) {
            temperature = Integer.parseInt(tempResp.getContent());
            System.out.println("Coordinador recibió de >> " + tempResp.getSender().getName() +
                    " temperatura >> " + tempResp.getContent() + "°\n");

            // Si es mayor a 30 grados se solicita la ubicación
            if (temperature >= 30) {
                reqLocation();
                System.out.println("Solicitando ubicación \n");
                exitValue = 2;
            } else {
                exitValue = 1;
            }
        }
    }
    public int onEnd() { return exitValue; }
}, "K");
```

Figura 4.13: Estado “K”, se presentan las condiciones que originan el caso 3.

Al enviarse la alerta 2, el coordinador le solicita a AgTemperatura que informe la temperatura del automóvil. Cuando el agente AgTemperatura envía una temperatura mayor o igual a 30°C, AgCoordinador solicita a AgAutomovil la ubicación del automóvil a través del método reqLocation(), (Fig. 4.13). Estas condiciones son las que propician que se genere un nuevo estado en el que se desencadenan las alertas correspondientes al caso 3, estas acciones se ejecutan en estado “L”.

```
fsm.registerState(new OneShotBehaviour() {
    private int exitValue;
    public void action() {
        ACLMessage locResp = blockingReceive(informTemplate);
        if (locResp != null && locResp.getSender().getLocalName().equals("AgAutomovil")) {
            location = locResp.getContent();
            System.out.println("Ubicación recibida " + location + "\n");

                // Se envía alerta 2
                level2Alert();

                // Se envia mensaje a AgAutomovil para que inicie la alarma del auto
                ACLMessage alarmMsg = new ACLMessage(ACLMessage.INFORM);
                alarmMsg.addReceiver(new AID("AgAutomovil", false));
                alarmMsg.setContent("Start car alarm");
                send(alarmMsg);

                exitValue = 2;
            } else {
                exitValue = 1;
            }
        }
    public int onEnd() { return exitValue; }
}, "L");
```

Figura 4.14: Estado “L” acciones efectuadas para el caso 3.

Si, por otra parte, el estado “K” recibe una temperatura menor a 30°C por parte de AgTemperatura, se le remite al estado “J” donde se verifica de nuevo la presencia y se solicita a través de un mensaje ACL de tipo REQUEST al agente AgTemperatura la temperatura.

El caso 3 termina cuando después de recibir el informe por parte de AgTemperatura de una temperatura mayor a 30°C, se envían las alertas correspondientes por medio del método level2alert(). Este método a través de un mensaje ACL de tipo INFORM indica a AgInterfaz que lance las alertas correspondientes, en este caso al usuario y a los servicios de emergencia. Se envía también un mensaje a AgAutomovil para que active la alarma externa del automóvil. La implementación de este estado se observa en la Fig. 4.14.

# Capítulo 5

# Análisis de Resultados

## 5.1. Introducción

En este capítulo, se expone de forma breve lo que son las pruebas de software y se presenta el desarrollo de pruebas para validar las funcionalidades del simulador. La validación se llevó a cabo a través de casos de prueba recogidos de las particiones de una máquina de estados finitos. Con ellos se consigue probar el porcentaje de fiabilidad de la especificación del software.

En otra de las secciones se presentan los resultados obtenidos de la simulación. Éstos se reducen a los tres casos especificados con anterioridad, describiéndolos a través de la herramienta de JADE para analizar el intercambio de mensajes entre los agentes, así como de las salidas enviadas a consola. Cabe destacar que los mismos resultados obtenidos se fueron ajustando y refinando conforme al modelo propuesto, siendo por tanto una forma más de validación de la simulación.

Tanto las representaciones gráficas como los mensajes a través de consola dejan clara la forma en la que colaboran los agentes para conseguir su objetivo, que es en primera instancia, prevenir el olvido de un bebé en un automóvil y si esto no se lograse, alertar entonces del riesgo que representa el abandono del menor en un automóvil.

## 5.2. Pruebas de software

Las pruebas de software demuestran hasta qué punto las funciones del software operan ajustándose a lo especificado, de forma tal que parecen alcanzarse los requisitos de rendimiento. Existen dos enfoques básicos en las pruebas de software, las pruebas de caja negra y las pruebas de caja blanca.

Las pruebas de caja negra se enfocan en el análisis de las entradas al sistema y de los resultados obtenidos. Buscan confirmar que las entradas al sistema son adecuadas y que los resultados que se producen son correctos, manteniéndose asimismo la integridad de la información externa. A estas pruebas también se les conoce como pruebas funcionales [43].

Por otra parte, las pruebas de caja blanca, conocidas también como pruebas estructurales, se enfocan en los detalles procedimentales del software. Estas pruebas toman en cuenta los mecanismos internos del sistema o de un componente del software.

Con relación a las pruebas de caja blanca, el desarrollo de casos de pruebas que abarquen todos los posibles caminos lógicos en los que puede incurrir un programa, aun siendo pequeño, puede consumir recursos indiscriminadamente. Por este motivo, se deben elegir los casos de prueba más representativos [44].

Llevar las pruebas de software a un nivel exhaustivo es pues impráctico e innecesario, sin embargo, existen estrategias de ingeniería de software que permiten probar tanto la estructura como la funcionalidad de un sistema a niveles aceptables y con resultados representativos. Algunas de ellas incluyen los análisis de valores límite o la partición equivalente, las cuales maximizan las probabilidades de descubrir fallos con una cantidad de pruebas razonables.

## 5.3. Generación de casos de prueba

Para probar la funcionalidad de la simulación, se eligió una prueba basada en modelo (Model Based Testing). Esta prueba de caja negra, toma como base para

generar los casos de prueba, el modelo representado como la máquina de estados finitos (FSM) incluido en el capítulo anterior. De la representación de esta FSM se desprenden 6 casos de prueba con los que se valida la funcionalidad y el alcance de la aplicación.

En la Fig. 5.1 se puede observar en color verde el alcance de cada uno de los casos de prueba definidos, al validarlos se obtiene una funcionalidad del 100 % en la simulación. En este caso, el inciso 'a' corresponde al caso de prueba 1 identificado como TC1, el 'b' al segundo caso de prueba con identificador TC2, y así sucesivamente para los 6 casos que se extrajeron para probar el modelo.

Se agregó un caso más con el identificador TC0 que representa los primeros estados de la máquina, mismos que se deben cumplir para poder iniciar cualquiera de los 6 casos. Este caso de prueba se incluye como precondición de los subsecuentes y abarca del estado A al estado F, en la Fig. 5.1 se indican encerrados en línea discontinua.

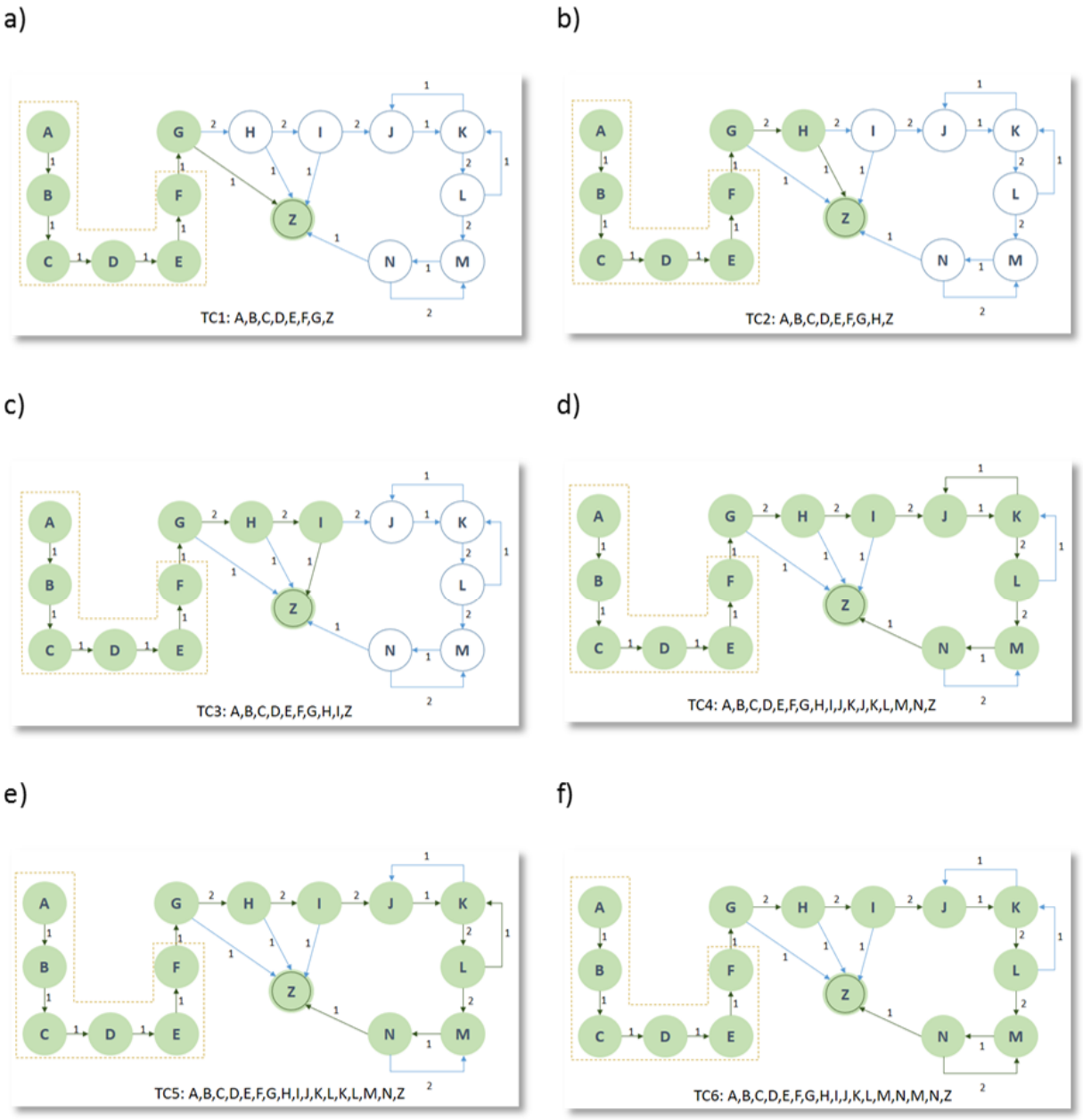



Figura 5.1: Alcance de los casos de prueba definidos. a) TC1. b) TC2. c) TC3. d) TC4. e) TC5. f) TC6.

En la Tabla 5.1 se describen los casos de prueba, los resultados esperados y obtenidos, así como el porcentaje de cobertura de cada uno de ellos. El porcentaje de cobertura se calcula dividiendo el número de estados cubiertos sobre el total de los estados de la máquina. Al recorrer todos los posibles estados y transiciones de la FSM se obtiene el 100 % de la funcionalidad del simulador, es decir, se cubren todas las especificaciones definidas evaluando la relación entradas – salidas del mismo.

| ID de Prueba | Descripción | Resultado Esperado | Resultado Obtenido |
|---|---|---|---|
| TC0 | A Solicita EstadoMotor<br>B Recibe EstadoMotor = apagado<br>C SolicitaPeso<br>D RecibePeso >= 10 y Solicita EstadoPuerta<br>E Recibe EstadoPuerta = abierto, inicia Cronómetro, envía Temperatura, Solicita RegistroActividad, solicita BloqueoSeguros y Alarma Interna<br>F SolicitaPeso | RecibePeso [5-15] | RecibePeso [5-15] **Porcentaje de cobertura = 40 %** |
| TC1 | Precondición: TC0 se ha completado exitosamente.<br>G RecibePeso<br>Z « SIMULACIÓN TERMINADA » | Peso < 10 Alarma Interna = Apagada | Peso < 10 Alarma Interna = Apagada **Porcentaje de cobertura = 14 %** |
| TC2 | Precondición: TC0 se ha completado exitosamente.<br>G RecibePeso >= 10<br>H Solicita y Recibe EstadoMotor, Detiene cronómetro, Apaga Alarma Interna<br>Z « SIMULACIÓN TERMINADA » | Seguros = desbloqueados | Seguros = desbloqueados **Porcentaje de cobertura = 20 %** |
| TC3 | Precondición: TC0 se ha completado exitosamente.<br>G RecibePeso >= 10<br>H Detiene cronómetro, Apaga Alarma Interna Solicita y Recibe EstadoMotor = Apagado, envía AlertaNivel1, SolicitaPeso<br>I RecibePeso<br>Z « SIMULACIÓN TERMINADA » | RecibePeso < 10 | RecibePeso < 10 **Porcentaje de cobertura = 27 %** |

Tabla 5.1: Especificación de los casos de prueba.

| ID de Prueba | Descripción | Resultado Esperado | Resultado Obtenido |
|---|---|---|---|
| TC4 | Precondición: TC0 se ha completado exitosamente.<br>G RecibePeso >= 10<br>H Detiene cronómetro, Apaga Alarma Interna Solicita y Recibe EstadoMotor = Apagado, envía AlertaNivel1, SolicitaPeso<br>I RecibePeso >= 10<br>J Solicita Temperatura<br>K Recibe Temperatura < 30<br>J Solicita Temperatura<br>K Recibe Temperatura >= 30, Solicita Ubicación<br>L Recibe Ubicación, Solicita Envío de Alerta 2, inicia Alarma Externa<br>M SolicitaPeso<br>N RecibePeso<br>Z « SIMULACIÓN TERMINADA » | RecibePeso < 10, Alarma Externa = Apagado, Seguros = desbloqueados | RecibePeso < 10, Alarma Externa = Apagado, Seguros = desbloqueados **Porcentaje de cobertura = 60 %** |
| TC5 | Precondición: TC0 se ha completado exitosamente.<br>G RecibePeso >= 10<br>H Detiene cronómetro, Apaga Alarma Interna Solicita y Recibe EstadoMotor = Apagado, envía AlertaNivel1, SolicitaPeso<br>I RecibePeso >= 10<br>J Solicita Temperatura<br>K Recibe Temperatura >= 30<br>L Recibe Ubicación = no recibida<br>K Solicita Ubicación | RecibePeso < 10 Alarma, Externa = Apagado, Seguros = desbloqueados | RecibePeso, Alarma Externa = Apagado, Seguros = desbloqueados **Porcentaje de cobertura = 60 %** |

Continúa Tabla 5.1: Especificación de los casos de prueba.

| ID de Prueba | Descripción | Resultado Esperado | Resultado Obtenido |
|---|---|---|---|
| Cont. TC5 | L Recibe Ubicación, Solicita Envío de Alerta 2, inicia Alarma Externa<br>M SolicitaPeso<br>N RecibePeso<br>Z « SIMULACIÓN TERMINADA » | - | - |
| TC6 | Precondición: TC0 se ha completado exitosamente.<br>G RecibePeso >= 10<br>H Detiene cronómetro, Apaga Alarma Interna Solicita y Recibe EstadoMotor = Apagado, envía AlertaNivel1, SolicitaPeso<br>I RecibePeso >= 10<br>J Solicita Temperatura<br>K Recibe Temperatura>= 30, Solicita Ubicación<br>L Recibe Ubicación, Solicita Envío de Alerta 2, inicia Alarma Externa<br>M SolicitaPeso<br>N RecibePeso<br>M SolicitaPeso<br>N RecibePeso<br>Z « SIMULACIÓN TERMINADA » | RecibePeso < 10, Alarma Externa = Apagado, Seguros = desbloqueados | RecibePeso < 10, Alarma Externa = Apagado, Seguros = desbloqueados **Porcentaje de cobertura = 60 %** |

Continúa Tabla 5.1: Especificación de los casos de prueba.

Para obtener la secuencia final de comunicación y alerta con el entorno, se utilizaron mensajes en consola, así como el analizador de paquetes de la librería de JADE. Con el uso del analizador de paquetes se depuraron las conversaciones mantenidas entre los agentes, mientras que los mensajes en consola permitieron añadir funcionalidades a la solución inicial, así como reestructurar las interacciones entre los agentes.

El establecer los valores esperados en los casos de prueba facilitó el ajuste de las interacciones entre agentes, obteniendo de esta forma una simulación acorde a lo especificado. Los resultados producidos por esta simulación se presentan a continuación.

## 5.4. Interpretación de resultados

Cada corrida de la simulación puede generar uno de los tres casos definidos en el modelo propuesto. Es evidente que pueden darse casos en los que no se esté transportando a un bebé en el automóvil, o que se transporte un bebé y sin apagar el automóvil el conductor abandone el mismo, sin embargo, para efectos del tema que nos atañe, se presentan los resultados simulando únicamente los casos previstos en los capítulos anteriores.

Cuando se detecta presencia en el automóvil, el conductor se baja del automóvil y se retira al bebé del mismo, se produce el caso 1 obteniéndose la salida de consola que muestra la Fig. 5.2. En ella se puede observar la información que recibe el AgCoordinador de los agentes AgPeso y AgAutomovil. AgCoordinador interpreta esta información como presencia de un bebé en el automóvil e indica al agente AgAutomovil que inicie la alarma interna e impida el cierre del auto bloqueando los seguros, asimismo da la instrucción al agente AgInterfaz de que actualice el registro de presencia.

Una vez más se recibe información del agente AgPeso, en este caso el AgCoordinador interpreta que no se detecta presencia en el automóvil. El siguiente paso es detener el cronómetro que inició el AgCoordinador, e indicar al AgAutomovil que detenga la alarma interna. Se muestra en la misma Fig. 5.2 el mensaje por parte del AgAutomovil cuando desactiva la alarma interna y desbloquea los seguros del automóvil. Se muestra el mensaje informando que la máquina de estados finitos ha concluido y con ello la simulación.

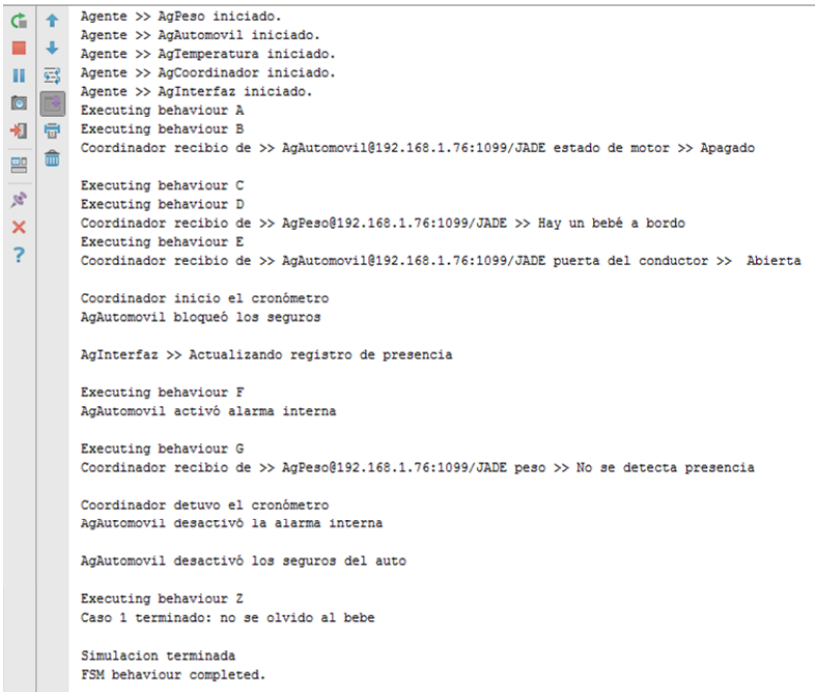



Figura 5.2: Salida en consola: secuencia caso 1.

Las interacciones entre los agentes para el caso 1 se pueden observar en la Fig. 5.3 que muestra la salida del analizador de paquetes de JADE. Se observan cinco agentes en ejecución, sin embargo, el AgTemperatura no presenta ninguna interacción con los demás agentes puesto que el bebé se retira antes de que se alcance una temperatura de riesgo.

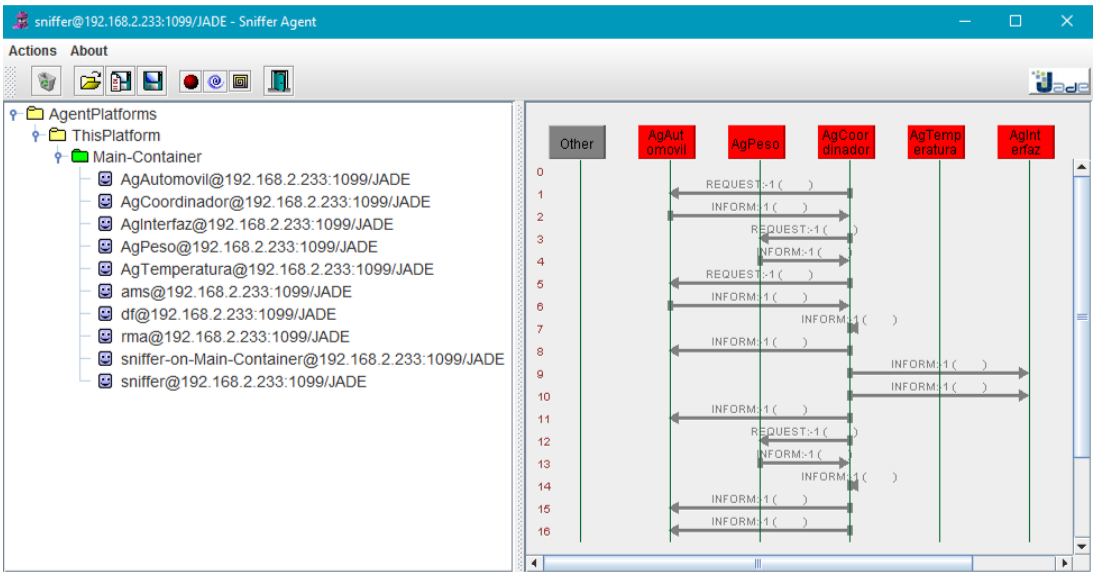



Figura 5.3: Analizador de paquetes: interacciones entre agentes, caso 1.

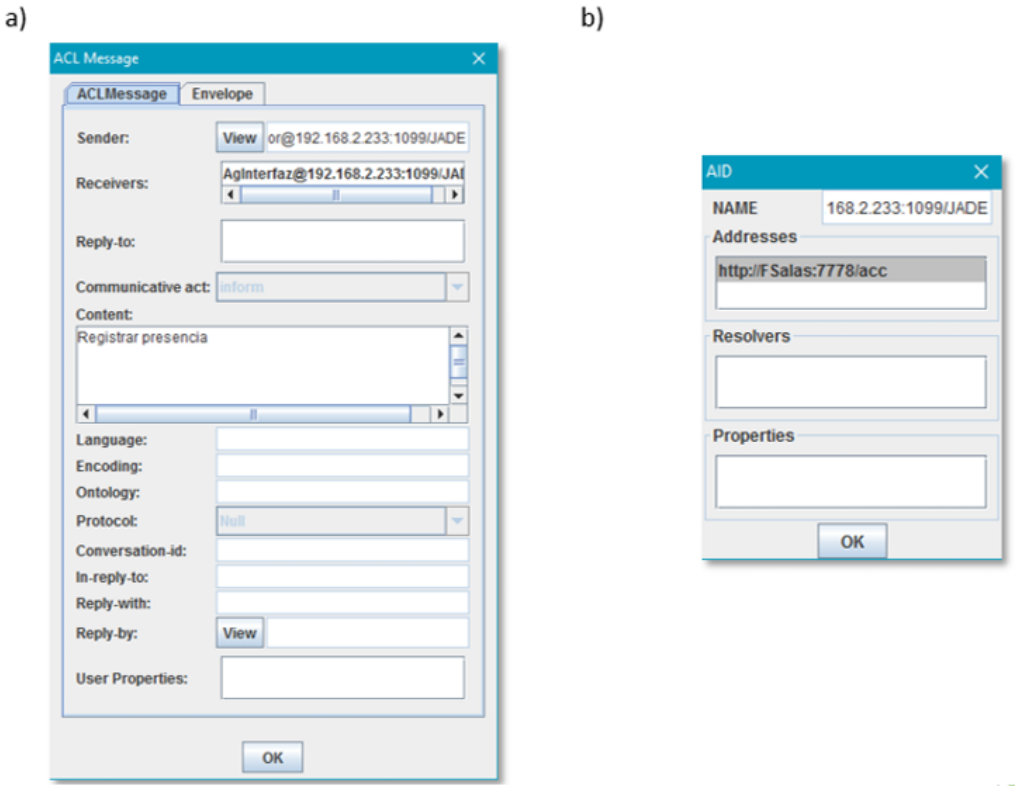



Figura 5.4: Mensaje ACL. a) Contenido. b) Identificador de Agente (AID).

La simulación desencadena la comunicación entre agentes a través de mensajes ACL (Agent Comunication Language). Estos mensajes contienen el nombre de los agentes que intervienen en la comunicación, bien sea ésta del tipo INFORM o REQUEST. En ambos casos se especifican en el contenido del mensaje, quien lo envía y quién o quiénes lo reciben, el contenido del mismo, entre otros.

La Fig. 5.4 muestra el despliegue de un mensaje ACL, en este caso el mensaje lo emite AgCoordinador y lo recibe AgInterfaz, el mensaje que se envía es 'Registrar presencia' (Fig. 5.4a). Asimismo, como parte del contenido de un mensaje ACL, se encuentra el AID o identificador de agente, éste contiene el nombre del agente y sus propiedades, mismo que se puede observar al presionar la opción View del mensaje ACL, (Fig. 5.4b).

El caso 2 se produce cuando se detecta presencia en el automóvil, el conductor se baja de éste y el bebé permanece en el automóvil a pesar de la advertencia emitida por la alarma interna. Al darse esta situación, se obtiene la salida de consola que muestra la Fig. 5.5. En ella se observa la misma secuencia de acciones descrita para el caso 1 hasta el comportamiento G, donde se sigue detectando presencia y se agota el tiempo del cronómetro, en el estado H el AgCoordinador detiene el cronómetro, recibe información del AgAutomovil indicando que no se ha encendido el motor y

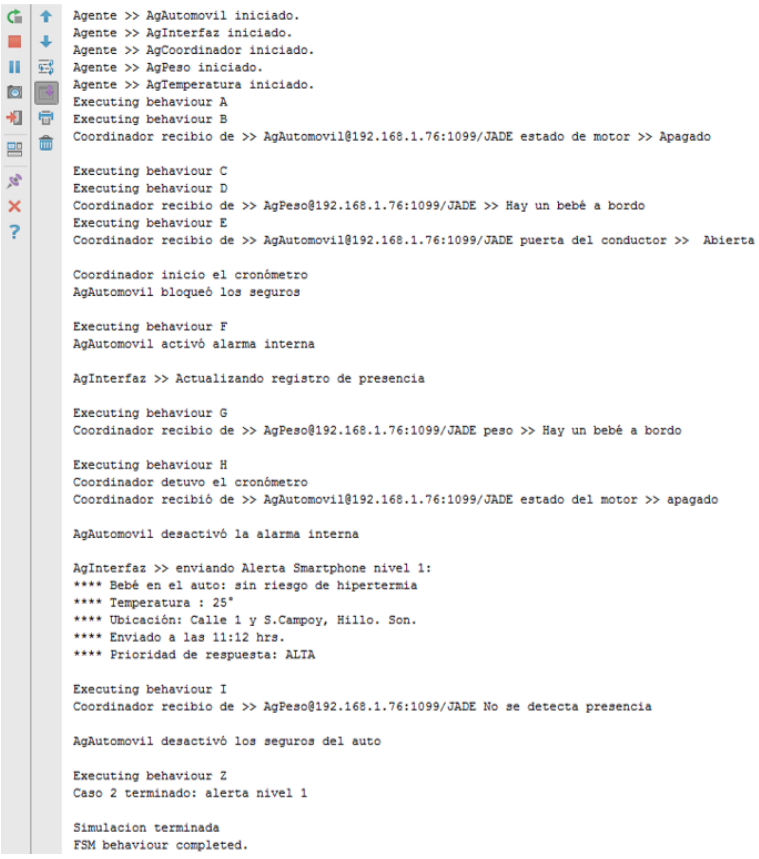



Figura 5.5: Salida de consola: secuencia caso 2.

se detiene la alarma interna para dar inicio a la primera alerta.

Se indica también al AgInterfaz que haga la composición de la alerta y la envíe al smartphone del usuario. El mensaje de alerta se muestra en consola. En el estado I se recibe la lectura del AgPeso que se interpreta en este caso como ausencia del bebé en el automóvil, indicando entonces el AgCoordinador al AgAutomovil que desactive el bloqueo de seguros del automóvil. Con esta acción concluye el caso 2, mostrándose en consola el mensaje de fin de simulación, así como el mensaje que indica que la máquina de estados finitos ha llegado a término.

La Fig. 5.6 muestra las interacciones entre los agentes para el caso 2 en el analizador de paquetes de JADE. En este caso, al igual que el anterior, se muestran cinco agentes en ejecución, y dado que el bebé se retira antes de que se alcance una temperatura de riesgo, el AgTemperatura se muestra a la espera sin ninguna interacción.

Finalmente, el caso 3 reproduce la misma secuencia del caso 2, solo que al presentarse el estado I, el AgPeso sigue detectando presencia, por lo que el AgTemperatura envía las mediciones al AgCoordinador y, al alcanzar los 30°C se solicita al AgInter-

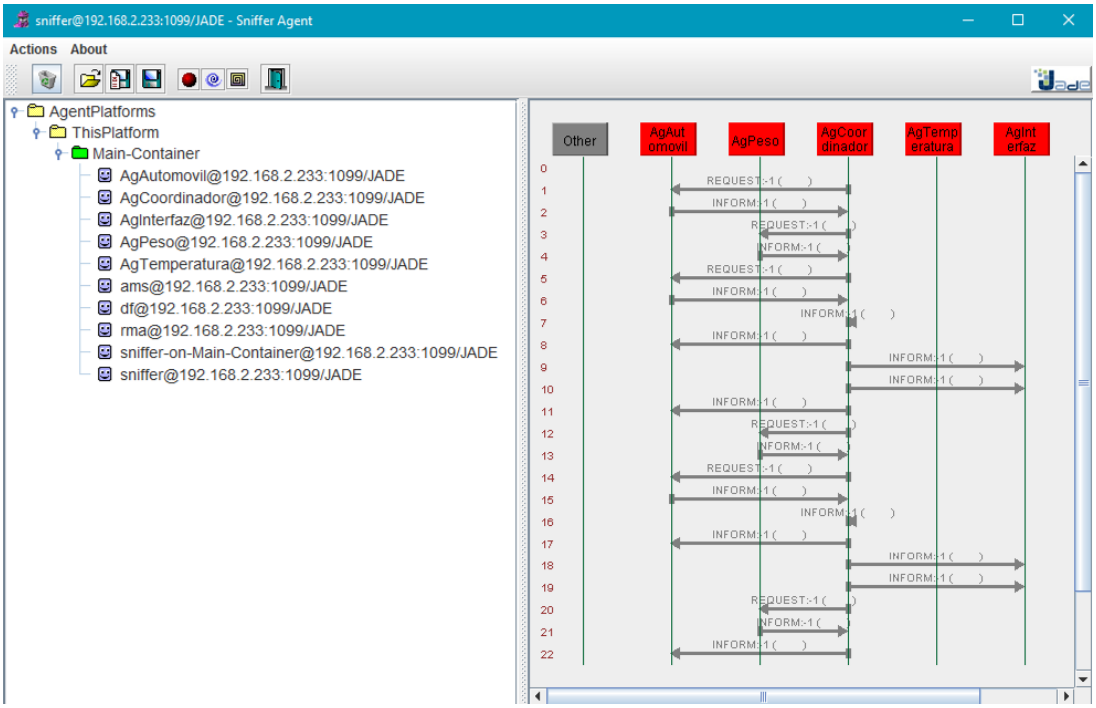
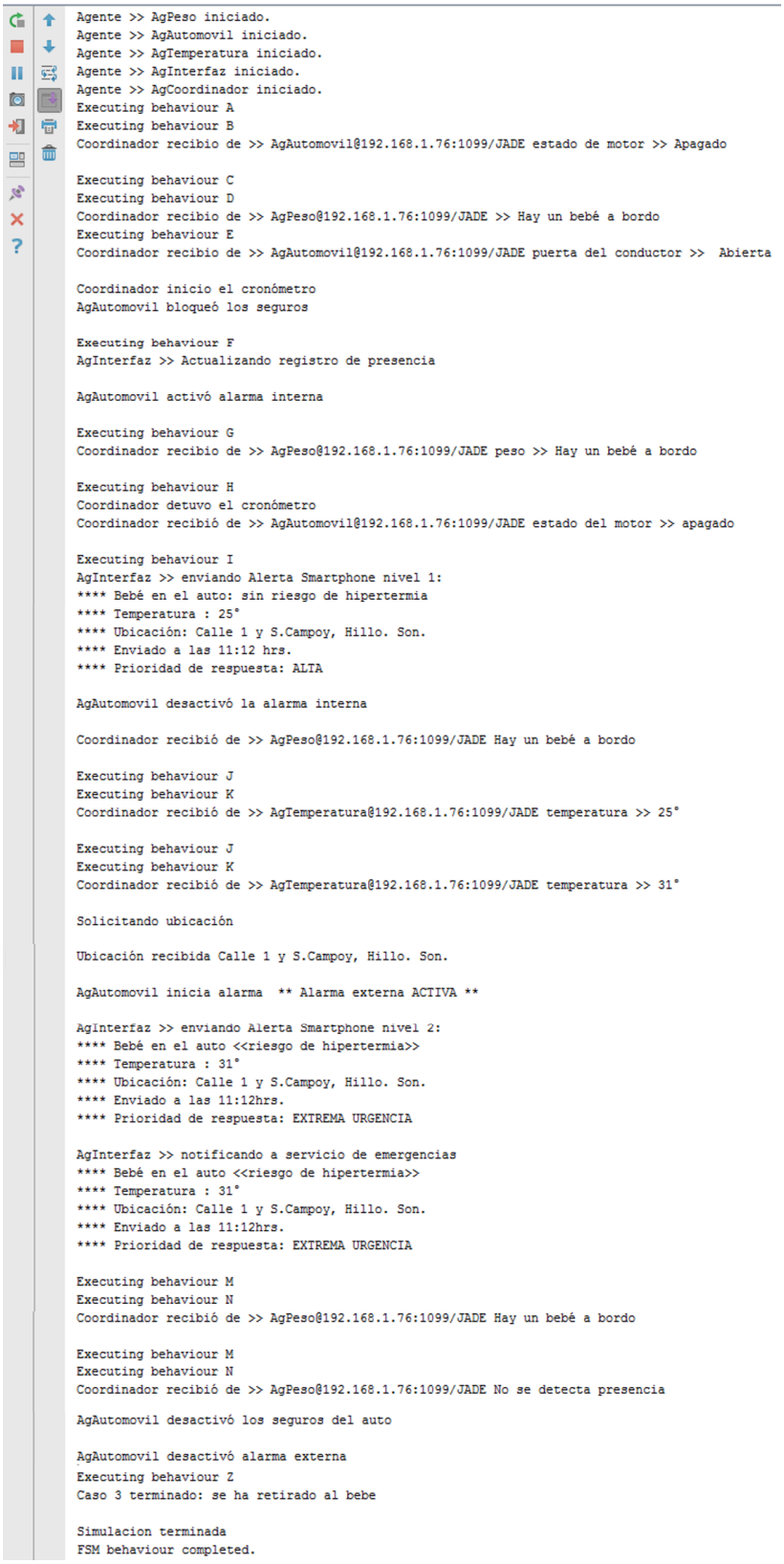



Figura 5.6: Analizador de paquetes: interacciones entre agentes, caso 2.

faz genere las alertas correspondientes. Se puede observar en consola la composición de los mensajes para el usuario y para el servicio de emergencias, así como el mensaje de que se ha activado la alarma externa.

En los estados M y N, se verifica con AgPeso si hay presencia en el automóvil y hasta que se retira el bebé del mismo, se desactiva la alarma externa y se desbloquean los seguros del auto. En este momento la máquina de estados finitos -en consecuencia la simulación- llegan a término, Fig. 5.7.

Por último, la Fig. 5.8 muestra en el analizador de paquetes de JADE las interacciones de los cinco agentes para la secuencia descrita anteriormente. En este caso a diferencia de los anteriores, se generan interacciones con el AgTemperatura, que monitoriza el clima del automóvil.



```
Agente >> AgPeso iniciado.
Agente >> AgAutomovil iniciado.
Agente >> AgTemperatura iniciado.
Agente >> AgInterfaz iniciado.
Agente >> AgCoordinador iniciado.
Executing behaviour A
Executing behaviour B
Coordinador recibio de >> AgAutomovil@192.168.1.76:1099/JADE estado de motor >> Apagado

Executing behaviour C
Executing behaviour D
Coordinador recibio de >> AgPeso@192.168.1.76:1099/JADE >> Hay un bebé a bordo
Executing behaviour E
Coordinador recibio de >> AgAutomovil@192.168.1.76:1099/JADE puerta del conductor >>  Abierta

Coordinador inicio el cronómetro
AgAutomovil bloqueó los seguros

Executing behaviour F
AgInterfaz >> Actualizando registro de presencia

AgAutomovil activó alarma interna

Executing behaviour G
Coordinador recibio de >> AgPeso@192.168.1.76:1099/JADE peso >> Hay un bebé a bordo

Executing behaviour H
Coordinador detuvo el cronómetro
Coordinador recibió de >> AgAutomovil@192.168.1.76:1099/JADE estado del motor >> apagado

Executing behaviour I
AgInterfaz >> enviando Alerta Smartphone nivel 1:
**** Bebé en el auto: sin riesgo de hipertermia
**** Temperatura : 25°
**** Ubicación: Calle 1 y S.Campoy, Hillo. Son.
**** Enviado a las 11:12 hrs.
**** Prioridad de respuesta: ALTA

AgAutomovil desactivó la alarma interna

Coordinador recibió de >> AgPeso@192.168.1.76:1099/JADE Hay un bebé a bordo

Executing behaviour J
Executing behaviour K
Coordinador recibió de >> AgTemperatura@192.168.1.76:1099/JADE temperatura >> 25°

Executing behaviour J
Executing behaviour K
Coordinador recibió de >> AgTemperatura@192.168.1.76:1099/JADE temperatura >> 31°

Solicitando ubicación

Ubicación recibida Calle 1 y S.Campoy, Hillo. Son.

AgAutomovil inicia alarma  ** Alarma externa ACTIVA **

AgInterfaz >> enviando Alerta Smartphone nivel 2:
**** Bebé en el auto <<riesgo de hipertermia>>
**** Temperatura : 31°
**** Ubicación: Calle 1 y S.Campoy, Hillo. Son.
**** Enviado a las 11:12hrs.
**** Prioridad de respuesta: EXTREMA URGENCIA

AgInterfaz >> notificando a servicio de emergencias
**** Bebé en el auto <<riesgo de hipertermia>>
**** Temperatura : 31°
**** Ubicación: Calle 1 y S.Campoy, Hillo. Son.
**** Enviado a las 11:12hrs.
**** Prioridad de respuesta: EXTREMA URGENCIA

Executing behaviour M
Executing behaviour N
Coordinador recibió de >> AgPeso@192.168.1.76:1099/JADE Hay un bebé a bordo

Executing behaviour M
Executing behaviour N
Coordinador recibió de >> AgPeso@192.168.1.76:1099/JADE No se detecta presencia

AgAutomovil desactivó los seguros del auto

AgAutomovil desactivó alarma externa
Executing behaviour Z
Caso 3 terminado: se ha retirado al bebe

Simulacion terminada
FSM behaviour completed.
```

Figura 5.7: Salida de consola: secuencia caso 3.

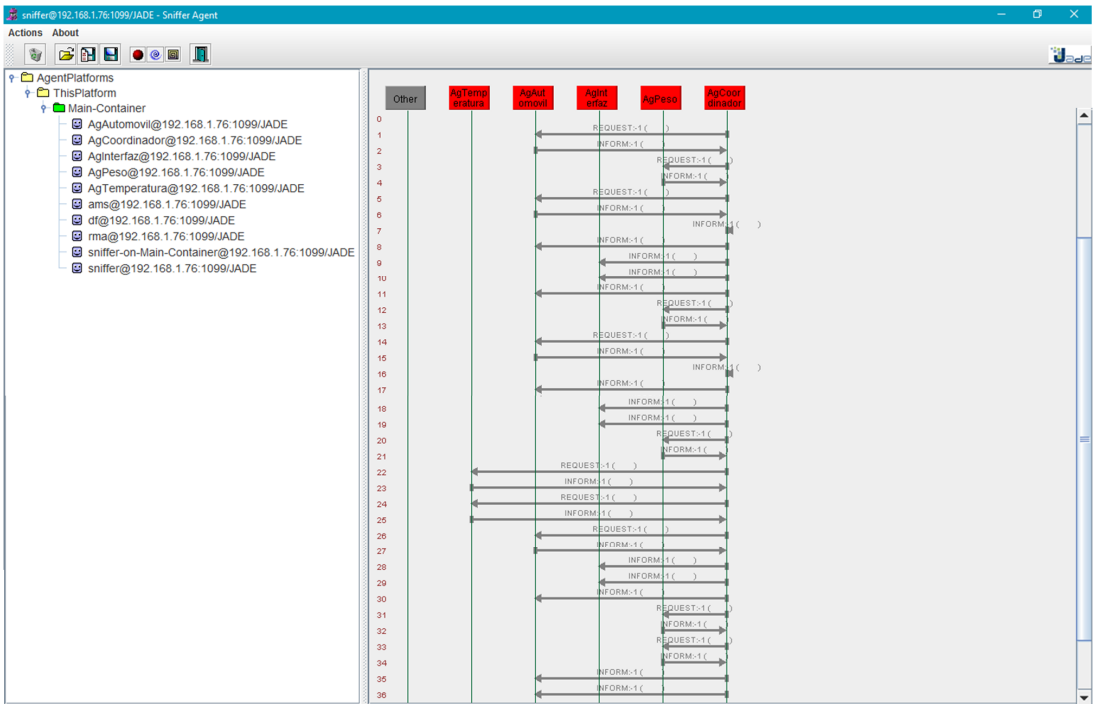



Figura 5.8: Analizador de paquetes: interacciones entre agentes, caso 3.

# Capítulo 6

# Conclusiones

Los resultados de la simulación basada en agentes que se presentó, confirman la validez del modelo propuesto. Se obtuvieron las salidas esperadas para cada uno de los casos establecidos, corroborando de esta forma que el modelo previene el olvido de bebés e intenta subsanar el abandono al alertar al usuario y a emergencias cuando se ha hecho caso omiso del aviso preventivo.

El desarrollo de un modelo que resuelva en gran medida el abandono de bebés en los automóviles a través de mecanismos imperceptibles para el usuario, representa un avance en seguridad para la población infantil, sector vulnerable, como se mencionó previamente.

## 6.1. Conclusiones

Los beneficios del cómputo pervasivo son cada vez más evidentes en ambientes cotidianos, por lo que no es de sorprender la cantidad de soluciones integradas en la vida diaria que podemos encontrar. Éstas facilitan el desenvolvimiento en los entornos y se desarrollan para campos tan diversos como agricultura, logística, salud, seguridad, manufactura entre muchos otros.

Los sistemas multi-agente como herramienta de desarrollo de sistemas inteligentes, son utilizados en la creación de ambientes pervasivos. Por tanto, esta propuesta se enfoca, a diferencia de las soluciones evaluadas, en prevenir el dejar por descuido a un bebé en un automóvil. Esta detección y las interacciones tanto con las personas como con el medio ambiente, se producen de forma imperceptible para el usuario a través de una comunicación confiable y efectiva de los agentes que componen el modelo.

La implementación del modelo representa una reducción del 54 % de las muertes por hipertermia asociadas a vehículos refiriéndose al aspecto preventivo. Esto significa que se evitarían directamente las muertes por olvido no intencional de bebés en los automóviles. El porcentaje asociado a las muertes por olvido intencional (18 %) se resuelve a través de las alertas al usuario o a protección civil, mismo que idealmente evitaría lesiones o la muerte del menor, sin embargo, esto está en función de los tiempos de respuesta del usuario o de los servicios de emergencia.

El 28 % restante de las muertes por hipertermia asociadas a vehículos, están relacionadas con niños que suben a jugar a los automóviles sin supervisión. Estas muertes no pueden ser evitadas con este modelo, ya que la detección de presencia está condicionada al uso de la silla del bebé.

## 6.2. Perspectivas

Desarrollar un dispositivo a partir del modelo es posible valiéndose de sensores de peso, temperatura y la Unidad electrónica de Control (ECU) del automóvil. La posibilidad de comunicación con el automóvil está documentada en [45], y se lleva a cabo a través de la inyección de paquetes a la ECU.

Asimismo, en el modelo se considera la generación de un registro de actividad, mismo que a futuro se podría trabajar para generar alertas en función de los cambios de ruta. Es decir, si habitualmente el vehículo que se desplaza al lugar de cuidado del menor, salta esa parada, se puede generar una alerta al dispositivo o dispositivos asociados a fin de avisar el cambio y prevenir el olvido. Lo anterior aún si el bebé se traslada en un automóvil sin el dispositivo, pero con el smartphone del conductor asociado a éste.

# Apéndice A

# Notación INGENIAS

| NOTACIÓN INGENIAS | | | |
|---|---|---|---|
|  | **Organización.** Se etiqueta con el nombre de la organización. |  | **Grupo.** Se etiqueta con el nombre del grupo. |
|  | **Agente.** Se etiqueta con el nombre del agente. |  | **Tarea.** Se etiqueta con el nombre de la tarea. |
|  | **Rol.** Se etiqueta con el nombre del rol |  | **Flujo de trabajo.** Se etiqueta con el nombre del flujo. |
|  | **Caso de uso.** Se etiqueta con el nombre del caso de uso. |  | **Objetivo.** Se etiqueta con el nombre del objetivo. |
|  | **Interacción.** Se etiqueta con el nombre de la interacción y su naturaleza, como coordinación, planificación o negociación. |  | **Recurso.** Se etiqueta con el nombre del recurso, la cantidad disponible del mismo, el límite superior e inferior admisibles. Por debajo o por encima de estos límites, el recurso se deshabilita. |

Notación utilizada para el modelado de agentes de esta investigación.

# Referencias


[1] K. Tsuzuki-Hayakawa, Y. Tochihara, and T. Ohnaka, "Thermoregulation during heat exposure of young children compared to their mothers," *European Journal of Applied Physiology and Occupational Physiology*, vol. 72, no. 1, pp. 12–17. [Online]. Available: http://dx.doi.org/10.1007/BF00964108

[2] RAE, "Hipertermia," accesed 2016-03-03. [Online]. Available: http://dle.rae.es/?id=KRjACzL

[3] J. Null, "Heatstroke Deaths of Children in Vehicles." 2015, accesed 2015-09-29. [Online]. Available: http://noheatstroke.org/

[4] C. McLaren, "Heat Stress From Enclosed Vehicles: Moderate Ambient Temperatures Cause Significant Temperature Rise in Enclosed Vehicles," *Pediatrics*, vol. 116, no. 1, pp. e109–e112, 2005. [Online]. Available: http://pediatrics.aappublications.org/cgi/doi/10.1542/peds.2004-2368

[5] CNN, "La muerte por hipertermia puede ser consecuencia de un cambio de rutina," 2010, accesed 2016-05-22. [Online]. Available: http://expansion.mx/salud/2010/07/08/la-muerte-por-hipertermia-puede-ser-consecuencia-de-un-cambio-de-rutina

[6] R. Rudd, A. Prasad, D. Weston, and K. Wiethholter, "Functional Assessment of Unattended Child Reminder Systems," no. July, 2015. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/812187_unattendedchildremindersystems.pdf

[7] A. J. Grundstein, S. V. Duzinski, D. Dolinak, J. Null, and S. S. Iyer, "Evaluating infant core temperature response in a hot car using a heat balance model," *Forensic Science, Medicine, and Pathology*, vol. 11, no. 1, pp. 13–19, 2015.

[8] J. E. Dematte, K. O'Mara, J. Buescher, C. G. Whitney, S. Forsythe, T. McNamee, R. B. Adiga, and I. M. Ndukwu, "Near-fatal heat stroke during the 1995

heat wave in Chicago," *Annals of Internal Medicine*, vol. 129, no. 3, pp. 173–181, 1998.

[9] K. Arbogast, a. Belwadi, and M. Allison, "Reducing the Potential for Heat Stroke to Children in Parked Motor Vehicles: Evaluation of Reminder Technology," no. July, pp. 1–43, 2012.

[10] Universal, "Olvidan a niña dentro de un auto, muere por calor," 2013, accesed 2015-10-07. [Online]. Available: http://www.vanguardia.com.mx/olvidananinadentrodeunautomuereporcalor-1771239.html

[11] J. Manuell, "Bebé muere por olvido en Mexicali," 2015, accesed 2015-10-17. [Online]. Available: https://www.elimparcial.com/EdicionEnLinea/Notas/Noticias/01072015/985531-Bebe-muere-por-olvido-en-Mexicali.html

[12] D. S. Dórame, "Muere bebé olvidado en un carro, a casi 50 grados de calor," accesed 2015-10-07. [Online]. Available: http://www.excelsior.com.mx/nacional/2014/05/20/960299

[13] L. Grisham, "Child deaths in hot cars: 10 key fact," 2014, accesed 2015-09-30. [Online]. Available: https://www.usatoday.com/story/news/nation-now/2014/07/02/child-deaths-hot-cars-facts/12055727/

[14] M. Vizcarra, "Conceptos de Calidad – Poka Yoke o corrección de errores," 2010, accesed 2015-11-12. [Online]. Available: http://desref.com/conceptos-de-calidad-poka-yoke-o-correcion-de-errores/

[15] S. Nava-Muñoz and A. L. Morán, "CANoE: A context-aware notification model to support the care of older adults in a nursing home," *Sensors (Switzerland)*, vol. 12, no. 9, pp. 11 477–11 504, 2012.

[16] M. Weiser, "The computer for the 21 st century," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 3, no. September, pp. 3–11, 1999. [Online]. Available: http://portal.acm.org/citation.cfm?doid=329124.329126

[17] R. Novática, "Computación ubicua," *Novática (Revista de la Asociación de Técnicos de Informática)*, no. 153, 2001.

[18] M. Satyanarayanan, "Pervasive computing: Vision and challenges," *IEEE Personal Communications*, vol. 8, no. 4, pp. 10–17, 2001.

[19] D. Saha and A. Mukherjee, "Pervasive computing: A paradigm for the 21st century," *Computer*, vol. 36, no. 3, pp. 25–31+4, 2003.

[20] P. D. Valencia, "Agentes Inteligentes : el siguiente paso en la Inteligencia Artificial," pp. 95–99, 2000.

[21] A. Mas, *Agentes software y sistemas Multi-agente: conceptos, arquitecturas y aplicaciones.* Pearson Educación, 2004.

[22] N. R. Jennings and M. Wooldridge, "Agent Technology: Foundations, Applications, and Markets," N. R. Jennings and M. J. Wooldridge, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, ch. Applicatio, pp. 3–28.

[23] A. Skarmeta, M. Pujol, and R. Rizo, *Agentes inteligentes: sistemas multiagentes y aplicaciones.*, M. d. c. y. Tecnología, Ed., 2002.

[24] A. S. Rao and M. P. Georgeff, "BDI agents: From theory to practice." *Icmas*, vol. 95, pp. 312–319, 1995.

[25] E. Jennings, N. R. in Werner and Y. Demazeau, *On being responsible. Decentralized AI 3: Proceedings of the Third European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Kaiserlauten, Germany, August 5-7, 1991.* Elsevier Science Inc., 1992.

[26] M. Bratman, D. Israel and M. Pollack, "Plans and resource-bounded practical reasoning," *Computational Intelligence*, pp. 4:349–355, 1988.

[27] J. A. Alty, D. Griffiths, N. R. Jennings, E. H. Mamdani, A. Struthers, and M. E. Wiegand, "Adept-advanced decision environment for process tasks: Overview & architecture," 1994. [Online]. Available: http://eprints.soton.ac.uk/252140/1/BCS-EXPERT-SYS94.pdf

[28] M. Wooldridge, *An Introduction to multiagent systems*, 2nd ed., 1996.

[29] J. P. Jorge J.J. Gomez-Sanz, Rubén Fuentes, "INGENIAS," accesed 2016-04-18. [Online]. Available: http://ingenias.sourceforge.net/

[30] T. J. Marchetti and A. J. García, "Una propuesta de definición para Plataformas de Desarrollo y Plataformas de Ejecución en Sistemas Multi-agente," 2002.

[31] AOS, "JACK," accesed 2016-04-18. [Online]. Available: http://agent-software.com/products/jack/

[32] J. C. Collis, D. T. Ndumu, H. S. Nwana, and L. C. Lee, "The Zeus Agent Building Toolkit," *BT Tecnology*, vol. 16, no. 3, pp. 1–9, 2001.

[33] G. D. Bellifemine F., Caire G., *Developing multi-agent systems with JADE*, Wiley, Ed., 2007.

[34] C. P. Shimpi and N. P. Kadam, "Vehicle active safety system: for children hypertermia in parked vehicle," pp. 3–5.

[35] P. Cargo, "Precious Cargo." [Online]. Available: https://play.google.com/store/apps/details?id=arya.preciouscargo

[36] Kars4kids, "Kars4Kids Safety app," accesed 2016-04-11. [Online]. Available: http://www.kars4kids.org/safety-app/#features

[37] E. Zolfagharifard, "Could this gadget prevent hot-car deaths? 'Starfish' senses if a child has been left in a vehicle and alerts a parent's phone," http://www.dailymail.co.uk/sciencetech/article-2728013/Could-device-prevent-hot-car-deaths-Starfish-senses-child-left-vehicle-alerts-parent-s-phone.html.

[38] "NASA Develops Child Car Seat Safety Device," 2002, accesed 2016-02-15. [Online]. Available: http://www.nasa.gov/centers/langley/news/releases/2002/02-008.html

[39] H. Reich, "A Car Seat That Talks," 2013, accesed 2016-02-15. [Online]. Available: http://www.jeanknowscars.com/cool-tech/car-tech-news/a-car-seat-that-talks/

[40] Evenflo, "ADVANCED SensorSafe™ Embrace™ DLX Infant Car Seat." [Online]. Available: http://www.evenflo.com/car-seats/embrace/31511754.html

[41] UT Dallas, "Engineering Graduate's Passion Project Keeps Children Safe," 2015, accesed 2016-03-15. [Online]. Available: http://www.utdallas.edu/news/2015/12/17-31834_Engineering-Graduates-Passion-Project-Keeps-Childr-_story-wide.html

[42] "Driver's Little Helper," accesed 2016-02-15. [Online]. Available: https://driverslittlehelper.com/products/drivers-little-helper

[43] L. Williams, "Testing Overview and Black-Box Testing Techniques," 2006. [Online]. Available: http://agile.csc.ncsu.edu/SEMaterials/BlackBox.pdf

[44] R. S. Pressman, *Ingeniería del software un enfoque práctico*, 5th ed. McGraw-Hill/ Interamericana de España, 2002.

[45] V. Aiello, P. N. Borazjani, E. Battista, and M. Albanese, "Next-generation technologies for preventing accidental death of children trapped in parked vehicles," *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration, IEEE IRI 2014*, pp. 508–513, 2015.