

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
FACULTAD DE INGENIERÍA
MAESTRÍA Y DOCTORADO EN CIENCIAS E INGENIERÍA



**“Componente Consciente del Contexto para desarrollar Sistemas de
Inteligencia Ambiental para Apoyar la Vida Independiente de los Adultos
Mayores”.**

T E S I S

que presenta para obtener el grado de MAESTRO EN CIENCIAS

DIANA SALDAÑA JIMÉNEZ

DIRECTOR DE TESIS:

MARCELA DEYANIRA RODRIGUEZ URREA

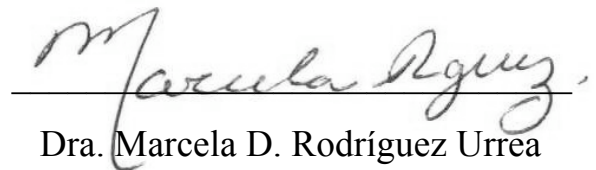
MEXICALI, B. C.

NOVIEMBRE DEL 2010.

RESUMEN de la tesis de **Diana Saldaña Jiménez**, presentada como requisito parcial para la obtención del grado de MAESTRO EN CIENCIAS en el campo de conocimientos de COMPUTACIÓN. Mexicali, Baja California, Noviembre 2010.

**COMPONENTE CONSCIENTE DEL CONTEXTO PARA
DESARROLLAR SISTEMAS DE INTELIGENCIA AMBIENTAL
PARA APOYAR LA VIDA INDEPENDIENTE DE LOS ADULTOS
MAYORES**

Resumen aprobado por:



Dra. Marcela D. Rodríguez Urrea

Directora de Tesis

El desarrollo de Sistemas de Inteligencia Ambiental (por su siglas en ingles: AmI) presenta algunos retos al momento de su construcción y diseño, tal como la captura, representación e inferencia del contexto de los usuarios.

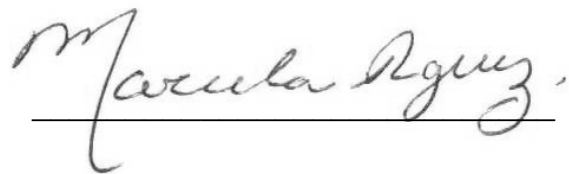
El enfoque de esta tesis radica en facilitar el desarrollo del Sistema Consciente del Contexto de los sistemas AmI dirigidos especialmente a apoyar las actividades de vida diaria de los adultos mayores. En particular, se desarrolló un componente de software basado en agentes inteligentes llamado *ELDeR*. El componente *ELDeR* encapsula la funcionalidad de la representación e inferencia del contexto, que se realiza mediante su *Ontología y Base de Reglas CARE*. Lo anterior facilita implementar sistemas AmI para prevenir riesgos en las actividades de vida diaria del adulto mayor y por consiguiente mejorar su vida independiente.

Palabras clave: Ontología, AmI, Componente de Software y Agentes.

ABSTRACT of the thesis presented by **Diana Saldaña Jiménez** as a partial requirement to obtain the MASTER OF SCIENCE degree in the field of COMPUTER. Mexicali, Baja California, Mexico, November 2010.

**CONTEXT-AWARE COMPONENT TO DEVELOP AMBIENT
INTELLIGENCE SYSTEMS TO SUPPORT THE INDEPENDENT
LIVING OF ELDERS**

Approved by:

A handwritten signature in dark ink, reading "Marcela Rodríguez Urrea", is written over a horizontal line.

Dr. Marcela D. Rodríguez Urrea

Thesis Advisor

The development of Ambient Intelligence (AmI) systems presents some challenges for their development, like the capture, representation and inference of the user's context.

The approach of this thesis is to provide facilities to implement the Context-Aware sub-system of an AmI system that supports elder people to carry out their activities of daily living. In particular, a software component based on intelligent agents was developed called *ELDeR*. The *ELDeR* component encapsulates functionalities for representing and inferring the users' context through the CARE ontology and rules base. Thus, the ELDeR Component facilitates to implement AmI systems that prevent risks in the elder's activities, and therefore improve their independent living.

Keywords: Ontology, Ambient Intelligence, software component and agents.

DEDICATORIA

A mis padres, mi hermana y mis amigos.

AGRADECIMIENTOS

Existen muchas personas a las que quisiera agradecer por apoyarme durante estos dos años de estudio y algunos durante mas tiempo.

Primeramente, me gustaría agradecer a mi asesora la Dra. Marcela por guiarme y aconsejarme en el desarrollo de mi trabajo. Sus consejos y paciencia siempre serán recordados con mucho cariño, ya que me han alentado a seguir adelante y siempre dar lo mejor de mí, no nada mas en lo profesional si no también como persona.

A mi comité de tesis, integrado por el Doctor Gabriel López y la Maestra Maria Luisa González. Muchas gracias por sus sugerencias y sus interesantes preguntas, las cuales enriquecieron mi trabajo y me motivaron a seguir investigando.

Durante mi estancia en la Facultad, conocí a compañeros de cubículo, que ahora tengo la fortuna de llamarlos amigos, Daniela Martínez y Juan Pablo García. Y mis amigos desde hace tiempo Rubén Roa, Ulises García, Josué Pizano y Selene Solis. Muchas gracias a todos ustedes, por sus palabras de aliento y su tiempo invertido en escucharme para darme ideas que considero “geniales”. A mi maestra Cecilia Curlango, gracias por su ayuda, platicas y sobre todo consejos ya que me han ayudado a tomar decisiones importantes en mi vida.

Finalmente, pero no menos importante, quisiera reiterar mi eterno agradecimiento a mis padres, ya que con su apoyo sincero e incondicional me encuentro en donde quiero estar. A mi hermana, que siempre me motiva para seguir adelante y llena de alegría todos los días de mi vida.

Gracias a CONACYT, por brindarme una beca durante mis estudios de posgrado.

Tabla de contenido

Capítulo I.....	1
Introducción	1
I.1 Contexto del Problema	2
I.1.1 Envejecimiento Poblacional	2
I.1.2 Actividades de la Vida Diaria	3
I.1.3 Clasificación de las actividades de la vida diaria	3
I.2 Antecedentes y Conceptos Relevantes.....	4
I.2.1 Sistemas de Apoyo a las Actividades de la Vida Diaria del Adulto Mayor	4
I.2.2 Cómputo Consciente del Contexto	5
I.2.3 Definición de Contexto	6
I.2.4 Retos para crear Sistemas Concientes del Contexto	6
I.3 Planteamiento del Problema	7
I.4 Objetivos.....	8
I.5 Metodología.....	8
I.6 Organización de la Tesis	10
Capítulo II.....	11
Trabajo Relacionado.....	11
II.1 SOCAM.....	11
II.1.1 Arquitectura.....	11
II.1.2 Ontología.....	12
II.2 CoBrA.....	14
II.2.1 Arquitectura.....	14
II.2.2 Ontología.....	14
II.3 CoDAMoS.....	16
II.3.1 Arquitectura.....	16
II.3.2 Ontología.....	16
II.4 Red Bayesiana para tomar decisiones con base al contexto del Adulto Mayor	17
II.4.1 Arquitectura.....	17
II.4.2 Representación del Contexto	18
II.5 Sistema mySAM.....	19
II.5.1 Arquitectura.....	19
II.5.2 Representación del Contexto	20
II.6 Middleware Gaia.....	21
II.6.1 Arquitectura.....	21
II.6.2 Representación del contexto	22
II.7 Ontología COSAR.....	24
II.7.1 Arquitectura.....	24
II.7.2 Ontología.....	24
II.8 Sistema I.L.S.A.	25
II.8.1 Arquitectura.....	25
II.8.2 Ontología.....	26
II.9 Framework CASIS	27
II.9.1 Arquitectura.....	27
II.9.2 Representación del Contexto	28
II.10 Discusión	29

Capítulo III	32
Análisis de Escenarios de Sistemas de Inteligencia Ambiental	32
III.1 Escenarios de AmI para apoyar ABVD y AIVD	32
III.1.1. Evitando riesgos de quemaduras y sofocación al bañarse	32
III.1.2 Apoyo al cumplimiento de la Medicación mediante Recordatorios Oportunos	34
III.1.3 Motivando el Cumplimiento de la Medicación	36
III.2 Requerimientos del Componente para desarrollar Sistemas AmI.....	37
III.2.1 Proveer un modelo de representación del contexto primario y secundario de los adultos mayores ejecutando sus Actividades de Vida Diaria	37
III.2.2 Proveer de mecanismos que facilite adaptar el comportamiento del sistema AmI al perfil, preferencias y necesidades de los usuarios para realizar sus Actividades de Vida Diaria	38
III.2.3 Desarrollar un componente que encapsule la funcionalidad descrita anteriormente.....	39
III.2.4 Facilitar que el desarrollador pueda especializar y extender la funcionalidad del componente para implementar aplicaciones AmI.	41
III.3 Discusión	43
Capítulo IV.....	44
Diseño del Componente ELDeR.....	44
IV.1 Middleware SALSA	44
IV.2 Arquitectura del Componente ELDeR	47
IV.2.1 Reglas CARE.....	48
IV.2.2 Ontología CARE.....	48
IV.2.3 Puertos.....	50
IV.2.4 Agentes del <i>Componente ELDeR</i>	51
IV.3 Implementación del Componente ELDeR	52
IV.3.1 Implementación de la base de <i>Reglas CARE</i> con el lenguaje JESS	52
IV.3.2 Implementación de la <i>Ontología CARE</i> con el lenguaje OWL-DL	52
IV.3.3 Agentes del Componente ELDeR programados con lenguaje JAVA utilizando el <i>middleware SALSA</i>	53
IV.4 Discusión.....	54
Capítulo V Uso del Componente ELDeR para implementar aplicaciones de Inteligencia Ambiental	55
V.1 Implementación del Sistema de Ambiente Inteligente CARE-Me	55
V.1.1 Arquitectura del Sistema CARE-Me	55
V.1.2 Ontología del Sistema <i>CARE-Me</i>	58
V.1.3 Base de Reglas del Sistema <i>CARE-Me</i>	60
V.2 Discusión	61
Capítulo VI.....	63
Evaluación del Componente ELDeR.....	63
VI.1 Protocolo de Evaluación.....	63
VI.1.1 Perfil del participante	64
VI.1.2 Descripción de las etapas de la evaluación	64
VI.2 Resultados de la Evaluación.....	76
VI.2.1 Resultados de la Etapa 1. Homogenización de los conocimientos	80
VI.2.2 Resultados de la Etapa 2. Evaluación del componente para diseñar sistemas AmI.....	81
VI.2.3 Resultados de la Etapa 3. Evaluación del componente para implementar sistemas AmI	81

VI.3 Discusión.....	85
Capítulo VII.....	86
Conclusiones y Trabajo Futuro.....	86
VII.1 Contribuciones.....	87
VII.2 Trabajo Futuro.....	89
Referencias	90
Apéndice A.....	96
Apéndice B.	99
Apéndice C.....	104
Apéndice D.....	107
Apéndice E.	114
Apéndice F.	119
Apéndice G.....	122
Apéndice H.....	127

Índice de Figuras

Figura 1. Metodología	9
Figura 2. Arquitectura de SOCAM (<i>Gu, 2005</i>)	12
Figura 3. Red bayesiana para las actividades del hogar (<i>Osmani, 2009</i>)	18
Figura 4. Arquitectura de Gaia (<i>Guarino, 1998</i>)	22
Figura 5. Escenario de Uso	33
Figura 6. Escenario de uso de la Sra. Ana	35
Figura 7. Escenario de uso de la Sra. María	36
Figura 8. Arquitectura del middleware SALSA (<i>Rodríguez, 2005</i>)	45
Figura 9. Arquitectura del <i>Componente ELDeR</i>	47
Figura 10. <i>Ontología CARE</i>	49
Figura 11. Funcionamiento del Componente ELDeR	52
Figura 12. Arquitectura del Sistema CARE-Me	57
Figura 13. Funcionamiento del Sistema CARE-Me desarrollado con el Componente ELDeR	58
Figura 14. Instancia de la Ontología CARE para representar contexto durante la medicación del adulto mayor	59
Figura 15. Instancia de la Ontología CARE cuando el adulto mayor se retira a dormir	59
Figura 16. Solución del Ejercicio 1	68
Figura 17. Solución del Ejercicio 2	69
Figura 18. Solución del Ejercicio 3	70
Figura 19. Solución del Ejercicio 4	71
Figura 20. Solución del Ejercicio 5	72

Índice de Tablas

Tabla 1. Reglas que definen el comportamiento del Adulto Mayor	60
Tabla 2. Reglas que rigen el comportamiento del ambiente inteligente	61
Tabla 3. Agenda del proceso de evaluación	65
Tabla 4. Plantillas de código para la práctica de implementación	74
Tabla 5. Resultados de las tres etapas de la evaluación	79
Tabla 6. Nomenclatura de los errores presentados en los exámenes aplicados	80
Tabla 7. Resultados de la encuesta final de la evaluación	83
Tabla 8. Resultados de la pregunta no. 10 de la encuesta final de la evaluación	84

Capítulo I

Introducción

El desarrollo de la tecnología, la innovación en proyectos y su aplicación en la vida diaria, han dado como resultado comodidad, funcionalidad y condiciones apropiadas que facilitan las actividades del ser humano en el siglo XXI. Los recursos tecnológicos que están al alcance y que son omnipresentes en la cotidianidad, fortalecen el intercambio humano-tecnología de los servicios que se obtienen en cualquier lugar y a cualquier hora.

El enfoque de esta tesis radica en los Sistemas Conscientes del Contexto dirigidos especialmente a apoyar a los adultos mayores y sus actividades de vida diaria.

En particular, se propone proveer un *software* que facilite el desarrollo de sistemas de inteligencia ambiental para mejorar la vida independiente de los adultos mayores. En el contenido de esta investigación se presentan el objetivo general y objetivos particulares, el planteamiento del problema y la metodología seguida para la realización de esta tesis. Un análisis del trabajo relacionado a este, una descripción detallada de la arquitectura de la propuesta tecnológica y los resultados de la evaluación aplicada.

I.1 Contexto del Problema

I.1.1 Envejecimiento Poblacional

En México y en el mundo se ha experimentado un proceso acelerado de transición demográfica. A partir de los años 30's inició el descenso de la mortalidad y trajo consigo un periodo caracterizado por un elevado crecimiento demográfico (CONAPO, 2008).

Diversos factores han contribuido al envejecimiento poblacional como la mayor cobertura de servicios de salud pública y los avances médicos para sanar y curar enfermedades características del adulto mayor. El envejecimiento de la población implicará una mayor demanda de servicios de salud, pues en este grupo de edad se presentan mayores tasas de morbilidad y necesidades de atención médica que en el resto de la población. Al mismo tiempo, los padecimientos de la población en edades avanzadas tienden a concentrarse en males crónico-degenerativos (CONAPO, 2008).

Además de lo anterior, se incrementará la demanda de servicios de retiro, asilos y centros de día. Sin embargo, se ha identificado que al ingresar el adulto mayor en esas instituciones, ellos experimentan sentimientos negativos como: abandono, tristeza, falta de intimidad, problemas con la convivencia con extraños y una vez en la institución, existe un incremento de la dependencia, disminución de la capacidad de autocontrol, disminución del contacto con la comunidad y con la sociedad. Para poder evitar las consecuencias debidas al ingreso a estas instituciones, la Organización Mundial de la Salud (OMS) propuso una iniciativa cuyo objetivo principal es lograr la integración familiar y social de las personas de edad avanzada (OMS, 2008). Esto ha obligado a organismos internacionales a diseñar políticas sociales que garanticen calidad de vida a los adultos mayores y el respeto por sus derechos en todos los ámbitos. Tal cual asume el lema de dicha iniciativa propuesta por la OMS, la cual dice:

“Envejecer en casa para mantener al mayor en su entorno familiar como medida óptima para la promoción de la salud y la prevención de enfermedades” (OMS, 2008).

Para apoyar el envejecimiento en el hogar, se requiere que los adultos mayores realicen sus actividades de vida diaria (bañarse, cocinar, comer, etc) de forma independiente y exitosa. Es por eso que existen aparatos que se pueden instalar en los

hogares para auxiliarlos en la realización de dichas actividades, tales como: andaderas, soportes para el baño y diversas soluciones tecnológicas de información y comunicaciones (TIC), cuyos ejemplos varían desde sistemas basados en Web, robots y sistemas empotrados. A continuación se describe que son las Actividades de Vida Diaria y posteriormente se presenta el enfoque tecnológico que nos interesa explorar en esta tesis para apoyar la vida independiente del adulto mayor, el cual es permitir la integración de la Inteligencia Ambiental en el hogar.

I.1.2 Actividades de la Vida Diaria

Las actividades que los seres humanos hacemos cotidianamente, como asearnos, vestirnos, alimentarnos o desplazarnos de un lugar a otro, han sido objeto de interés y atención constante. No fue hasta 1978 cuando la Asociación Americana de Terapia Ocupacional (*AOTA, por sus siglas en inglés*) elaboró la primera definición de Actividades de la Vida Diaria (AVD). En ella se define como: *“Los componentes de la actividad cotidiana comprendidos en las actividades de auto-cuidado, trabajo y juego/ocio”* (Moruno, 2008).

El término más actual de AVD es: *“El conjunto de actividades cuya realización es necesaria para el auto cuidado personal y el mantenimiento de una vida independiente”* (Moruno, 2008). Además de que estas reflejan las tendencias en función de la pertenencia a determinados grupos sociales.

Las actividades de vida diaria tienen diferentes características y grados de complejidad para desarrollarse por lo que para su estudio, de acuerdo con Moruno (2008), se clasifican en Actividades: *Básicas, Instrumentales y Avanzadas* de vida diaria.

I.1.3 Clasificación de las actividades de la vida diaria

Actividades Básicas de la Vida Diaria (ABVD). Conjunto de actividades primarias de las personas encaminadas a su *auto cuidado y movilidad*, que la dotan de autonomía e independencia elementales, que le permiten vivir sin precisar ayuda continua de otros. Por ejemplo: bañarse, control de esfínteres, comer, entre otros (Moruno, 2008).

Actividades Instrumentales de la Vida Diaria (AIVD). Son las tareas que le permiten a una persona adaptarse a su entorno y mantener su independencia en la comunidad. Por

ejemplo: Uso de dispositivos de comunicación, cuidado de otros, compras, preparación de alimentos, etcétera (Moruno, 2008).

Actividades Avanzadas de la Vida Diaria (AAVD). Es un conjunto de actividades segregadas de las Actividades de Vida Diaria Instrumental, que se caracterizan por ser conductas elaboradas, de control del medio físico y del entorno social, que le permiten al individuo desarrollar un rol social, mantener una buena salud mental y disfrutar de una excelente calidad de vida. Por ejemplo: viajar, trabajar, hacer deporte, etcétera (Moruno, 2008).

I.2 Antecedentes y Conceptos Relevantes

I.2.1 Sistemas de Apoyo a las Actividades de la Vida Diaria del Adulto Mayor

El envejecimiento, en la mayoría de los casos viene acompañado por varios problemas de salud relacionados con la edad, que en varios casos decrementa la calidad de vida del adulto mayor. Para mejorarla se propone una nueva tendencia llamada Inteligencia Ambiental (AmI, del inglés: *Ambient Intelligence*), que se ha utilizado para darle apoyo al adulto mayor en su hogar. Así pues, las tecnologías AmI en el hogar del adulto mayor podran proveer las facilidades para apoyar su autonomía, y en caso de algún posible riesgo debería generar diferentes tipos de alertas dependiendo de la gravedad de la situación. Por ejemplo, alertando al adulto en caso de un posible riesgo, o a sus cuidadores en caso de una emergencia. Los sistemas de Inteligencia Ambiental deben incluir:

Tecnología Penetrante. Se utiliza para detectar el comportamiento y la interacción de los adultos mayores con los objetos de su casa. Uno de los elementos más importantes es el sensado de las acciones y el comportamiento de los adultos mayores, ya que de esta manera se pueden reconocer sus actividades de la vida diaria (Rodríguez, 2005).

Técnicas de Interacción Humano-Computadora. Estas se utilizan para proveer asistencia directa al adulto mayor, por ejemplo para recordarle como realizar una actividad apropiadamente y libre de riesgos, o para notificar a los cuidadores el estado actual de la persona adulta a su cuidado (Kimmel, 2007).

Sistemas Concientes del Contexto. Monitorean las actividades de la vida diaria y reconocen cuando los adultos mayores están en una situación de riesgo. Algunos de estos

sistemas utilizan agentes como una herramienta que facilita el diseño e implementación de los sistemas de ambientes inteligentes. Los agentes proveen las características de inteligencia y conciencia del contexto requeridos para inferir las actividades de los adultos mayores y predecir si hay un riesgo (Rodríguez, 2005).

Así, los sistemas AmI podrían ayudar al adulto mayor a realizar actividades por sí solos, prolongando su estilo de vida independiente. En esta tesis se abordan algunos de los retos para crear Sistemas Concientes del Contexto, los cuales forman parte del Sistema de Inteligencia Ambiental en el hogar del adulto mayor. Específicamente, se desarrollarán mecanismos de representación y diseminación del contexto de los adultos mayores. Estos mecanismos facilitarán la inferencia de riesgos para que el sistema AmI tome la acción apropiada y prevenga un posible riesgo para el adulto mayor o notifique a sus cuidadores o familiares si se considera necesario.

I.2.2 Cómputo Conciente del Contexto

El cómputo conciente del contexto fue discutido por primera vez por Schilit y Theimer en 1994 (Schilit, 1994). Definieron que es un software que se adapta a la ubicación, las personas y objetos, así como los cambios en los objetos a través del tiempo. El término *conciente del contexto* se volvió una especie de sinónimo de otros términos como: adaptativo (Brown, 1996), reactivo (Cooperstock, 1995), responsivo (Elrod, 1993), situado (Hull, 1997), sensible al contexto (Rekimoto, 1998) y de ambiente directo (Flickas, 1997). Definiciones previas de cómputo conciente de contexto recaen en dos categorías que son: usar el contexto y adaptar el contexto. La primera significa que los dispositivos computacionales detecten, sensen, interpreten y respondan a aspectos del ambiente del usuario. La segunda significa que sufren cambios dinámicos o que se adaptan al comportamiento con base en el contexto de la aplicación y del usuario. Posteriormente fue definido por Anind Dey en el año 2000 (Dey, 1999), quién define a un sistema conciente de contexto como las aplicaciones que tienen un cambio dinámico o que adaptan su comportamiento con base en el contexto de la aplicación del usuario (Weiser, 1993). La definición más reciente de Cómputo Conciente del Contexto establece que:

“Un sistema es conciente de contexto si se usa el contexto para proveer información relevante y/o servicios para el usuario, donde la relevancia depende de la tarea del usuario” (Dey, 1999).

I.2.3 Definición de Contexto

Según Anind K. Dey, la interacción del individuo con otros de su misma especie, es debido a que comparten ideas, lenguaje, el mundo que los rodea, como funcionan las cosas para resolver situaciones de la vida diaria y además la disponibilidad de usar información implícita para incrementar la conversación entre ellos. Bajo el anterior entendimiento el ser humano está usando su contexto, es decir:

“El contexto es cualquier información que puede ser usada para caracterizar la situación de la entidad. Una entidad es una persona, lugar u objeto que se considera relevante para la interacción entre un usuario y una aplicación, incluyendo el usuario y las aplicaciones (Dey, 1999)”.

El contexto se puede clasificar en dos tipos, el contexto primario, el cual se caracteriza por ser el que se encuentra de manera inmediata a través de los sentidos humanos, y el contexto secundario, que se infiere a través del contexto primario.

I.2.4 Retos para crear Sistemas Concientes del Contexto

Existen varios retos para las aplicaciones concientes de contexto, ya que es difícil trabajar con el contexto por distintas razones (Dey, 1999). Capturar el contexto primario, por ejemplo, requiere el uso de sensores y dispositivos computacionales. Posteriormente, el contexto debe ser abstraído para que tenga sentido para la aplicación, por ejemplo, el identificador (ID) de un usuario móvil debe convertirse en el nombre del usuario y/o en el rol que juega en el ambiente. Finalmente, otros retos a enfrentar son: representar y diseminar la información contextual capturada en ambientes computacionales dinámicos (Rodríguez, 2005), los cuales se abordan en esta tesis.

Existen varios tipos de contexto, que en la práctica unos son más importantes que otros para una aplicación en particular. Algunos de ellos son ubicación, identidad, actividad y la hora. Estos son información contextual de tipo primario, que sirven para caracterizar una situación determinada. Estos tipos de contexto no solo responden a las preguntas de

quién, qué, cuándo y dónde, y actúan como índices en otras fuentes de información contextual conocido como contexto de segundo nivel o contexto secundario. Los elementos del contexto secundario comparten una característica en común la cual es que se puede indexar por el contexto primario ya que sus atributos son la entidad para el contexto primario (Dey, 1999).

Para apoyar al adulto mayor a realizar sus actividades de forma independiente en el hogar, se propone el enfoque de los sistemas de inteligencia ambiental (AmI), el cual converge en cuatro áreas de la computación; cómputo ubicuo, el cual se representa por computadoras o sensores empotradas en el ambiente y sensan las actividades realizadas por el adulto mayor. Los Sistemas Conscientes del Contexto y los Sistemas Inteligentes, representan el contexto que se monitorea e infiere un riesgo respectivamente, es ahí donde actúan las técnicas de Interacción Humano-Computadora (IHC), las cuales son las encargadas de notificar dicho riesgo y así apoyar la independencia del adulto mayor.

El enfoque de esta tesis radica en los Sistemas Conscientes del Contexto y los Sistemas Inteligentes, por lo tanto se propone el diseño y desarrollo de un componente de software mediante agentes autónomos que comuniquen los servicios que se ofrecen, una ontología que represente las actividades del adulto mayor y una base reglas que infiera un posible riesgo.

I.3 Planteamiento del Problema

En la actualidad existen diversas plataformas o tecnologías que asisten a los adultos mayores en sus actividades de vida diaria. Sin embargo, para implementar sistemas AmI se enfrentan retos tal como los mencionados en la sección 1.2.4. Es por eso que en esta tesis se plantea un componente de *software* el cual representa el contexto sensado y lo disemina entre los subsistemas del sistema AmI que lo requiera procesar para tomar una decisión del tipo de asistencia que requiere el adulto mayor con base a su contexto.

I.4 Objetivos

El objetivo general de esta tesis es:

“Desarrollar un componente basado en agentes de software y en una ontología, el cual facilite la implementación de sistemas de Inteligencia Ambiental que apoyen a los adultos mayores a realizar sus actividades de la vida diaria libre de riesgos”.

Al componente propuesto, se le puso por título ELDeR, cuyo nombre se obtuvo retomando las siglas de la frase en inglés: ***“Enabling Living inDependently of Risks”***.

A partir de este objetivo general, se derivan los siguientes objetivos particulares:

- Identificar las Actividades de la Vida Diaria que el componente propuesto apoyaría.
- Diseñar e implementar el componente mediante herramientas y lenguajes estandarizados de dominio público.
- Conocer la facilidad del uso del componente para crear sistemas de inteligencia ambiental que apoyen las Actividades de la Vida Diaria de los adultos mayores.
- Conocer la flexibilidad de la ontología para adaptarse a diferentes escenarios en los que se requiera inferir riesgos asociados a actividades de vida diaria.

I.5 Metodología

La metodología que se siguió para el desarrollo del *Componente ELDeR*, se presenta en la Figura 1, y consistió de las siguientes etapas:

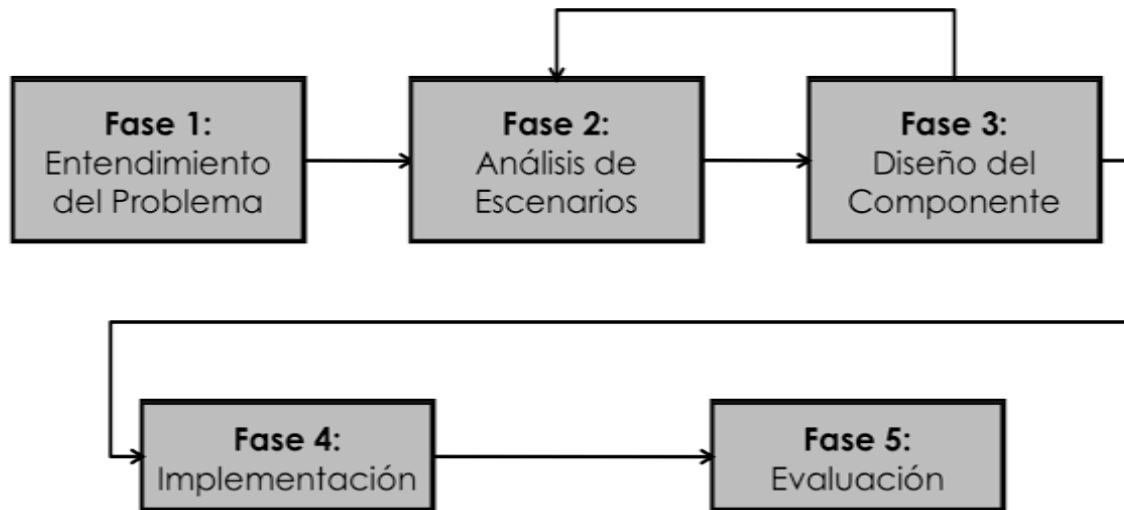


Figura 1. Metodología

Fase 1: Entendimiento del Problema. Consistió básicamente en obtener información de la literatura médica sobre AVD, e identificar el trabajo relacionado sobre *middlewares* o infraestructuras para el desarrollo de sistemas conscientes del contexto que utilicen ontologías para representar el contexto, así como técnicas para inferir contexto secundario.

Fase 2: Análisis de Escenarios. Se analizaron las aplicaciones de inteligencia ambiental para apoyar la vida independiente del adulto mayor, las cuales desarrollaron estudiantes del posgrado MyDCI; es decir, estas aplicaciones se tomaron como caso de estudio para identificar las características de las actividades de vida diaria que apoyaban estas aplicaciones y que ayudarían a identificar los conceptos y relaciones de la ontología. Adicionalmente, el análisis de estos escenarios permitió establecer los requerimientos del componente a desarrollar. Ya que se observó las características comunes entre cada escenario, que se abstraieron para ser planteados como requerimientos.

Fase 3: Diseño del Componente. A partir de los análisis de los diferentes escenarios que se estudiaron se realizó el diseño de la arquitectura del componente propuesto.

Fase 4: Implementación. Al terminar el diseño arquitectónico se desarrolló la etapa de la implementación de la propuesta tecnológica, utilizando SALSA para implementar los agentes, y lenguajes de programación como JAVA y JESS para el desarrollo de la

comunicación y el razonamiento del componente y lenguajes como OWL para la representación del contexto.

Fase 5: Evaluación. Por último se evaluó la facilidad de uso del *Componente ELDeR* para el desarrollo de sistemas de inteligencia ambiental. En esta evaluación participaron ocho personas con perfil de Ingeniero en Computación durante tres sesiones que comprendieron: pláticas introductorias al problema y las tecnologías a utilizarse, y exámenes para evaluar la comprensión de la información proporcionada a los estudiantes y su desempeño utilizando el componente.

I.6 Organización de la Tesis

En el siguiente capítulo, se describe el trabajo relacionado con esta propuesta tecnológica, los cuales son infraestructuras que facilitan el desarrollo de sistemas conscientes del contexto, y en particular, de aquellos que están hechos para realizar sistemas Aml que ayuden a los adultos mayores. En el Capítulo III, se presenta un análisis de los diferentes escenarios que se estudiaron para llevar a cabo el diseño de esta propuesta. En el Capítulo IV, se incluye una descripción del diseño de la arquitectura de la propuesta tecnológica, que permite el desarrollo de sistemas de ambientes inteligentes, debido a que toma decisiones a partir de un contexto capturado. El Capítulo V describe el ejemplo de una implementación simulada de un sistema de inteligencia ambiental utilizando el *Componente ELDeR*. Para después continuar en el Capítulo VI, que explica la evaluación realizada para determinar si la implementación es fácil de usar para los programadores y si les gustaría utilizarlo en futuras aplicaciones de sistemas de inteligencia ambiental. Finalizando con el Capítulo VII que contiene las conclusiones y el trabajo futuro de esta investigación.

Capítulo II

Trabajo Relacionado

En este capítulo se analiza el trabajo relacionado a infraestructuras de desarrollo para sistemas conscientes del contexto de sistemas Aml, y a aplicaciones desarrolladas para apoyar la vida independiente del adulto mayor. Finalmente se cierra con una breve análisis de los trabajos presentados.

II.1 SOCAM

La construcción de los servicios conscientes de contexto es una tarea compleja debido a la falta de un adecuado soporte en el ambiente de tecnología penetrante (*Kimel, 2007*). Para abordar este problema se propuso el *middleware Service-Oriented Context-Aware Middleware* (SOCAM), el cual es una arquitectura para la construcción y el diseño rápido de prototipos de servicios conscientes de contexto.

II.1.1 Arquitectura

La idea principal de SOCAM es proveer una infraestructura que soporte la construcción de ambientes inteligentes. Este *middleware* transfiere el contexto del usuario y lo transforma a un espacio semántico que puede ser compartido y accedido por los servicios conscientes del contexto (Ver Figura 2). SOCAM consiste en los siguientes sub-sistemas:

- **Proovedor contextual.** Sirve para la abstracción del contexto desde fuentes heterogéneas (externas o internas) y las convierten en representaciones en el lenguaje ontológico OWL (*Ontology Web Language*) (*OWL, 2008*), así que el contexto puede ser compartida y reutilizada por los componentes de servicio. Es decir, separa el contexto sensado del contexto que es inferido por SOCAM. Todo *proveedor del contexto* necesita registrarse en un registro de servicio usando el mecanismo de *Servicio de localización* para que otros componentes del sistema puedan encontrarlos.
- **Interprete contextual.** Este componente es responsable de procesar el contexto usando un razonamiento lógico. Las tareas del procesamiento contextual puede

incluir definir el contexto primario del secundario (Dey, 2001), mantener la consistencia del contexto, y resolver conflictos contextuales.

- **Base de datos contextual.** Guarda el contexto en ontologías y el contexto pasado del usuario para crear un subdominio.
- **Servicios conscientes del contexto.** Usan los diferentes niveles contextuales y adaptan el comportamiento del contexto actual.
- **Servicio de localización.** Provee un mecanismo donde el *proveedor del contexto* y el *interprete del contexto* pueden advertir su presencia y pueden detectar la ubicación del usuario.

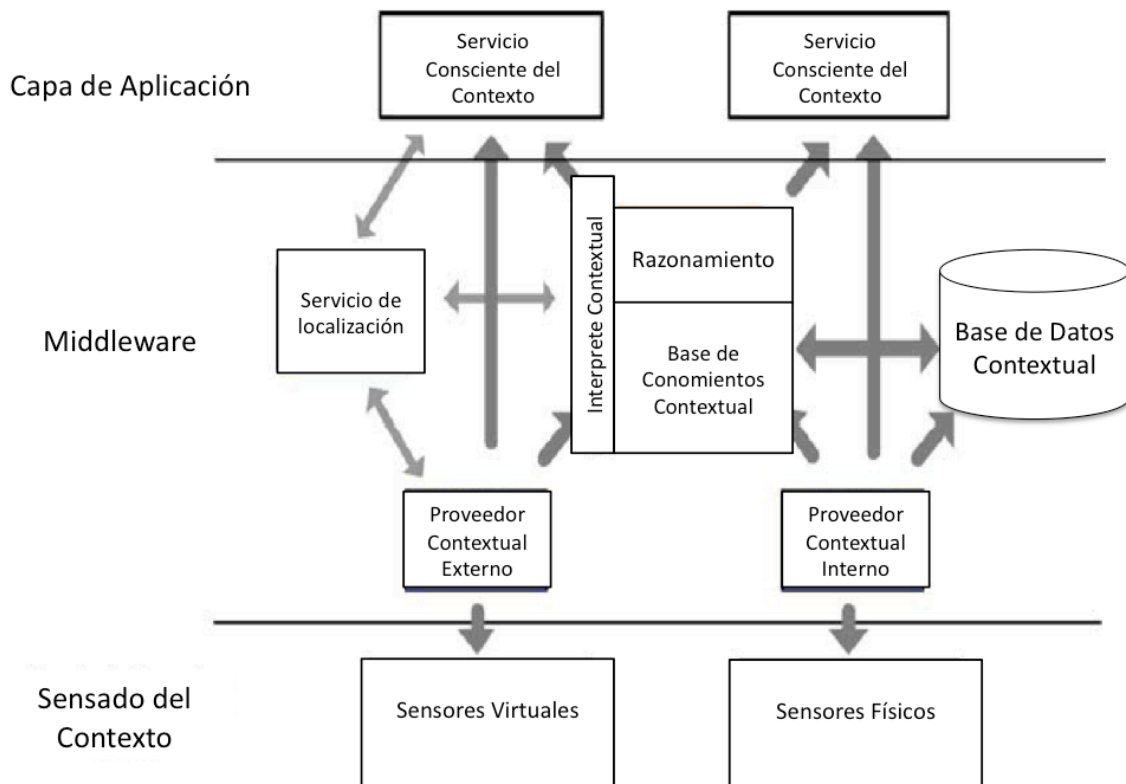


Figura 2. Arquitectura de SOCAM (Gu, 2005)

II.1.2 Ontología

Para facilitar el modelado de contexto en diferentes dominios, SOCAM propone y define una ontología de alto nivel (*Generalized Ontology in Upper Level*) que incluye conceptos generales y propone ontologías de dominio específico (*Domain-Specific Ontologies*) que

incluye los conceptos propios del dominio modelado. La ontología de alto nivel captura el contexto general permitiendo definir y compartir conceptos que pertenecen a diferentes dominios. La ontología generalizada consiste en 14 clases y seis propiedades o relaciones principales. Esta ontología define los conceptos básicos de la persona, su ubicación, su entidad computacional y la actividad que esta realizando (*Gu, 2005*).

II.1.3 Razonamiento

Se propone un modelo común para la representación de contexto incierto usando Redes Bayesianas como medio de razonamiento. Este modelo se extiende del modelo básico de la ontología (*Gu, 2005*).

La estructura y los predicados contextuales están descritos en la ontología la cual incluye las descripciones de las clases, propiedades y sus instancias. La ontología está escrita en OWL con tripletas RDF.

II.2 CoBrA

Uno de los retos en el área de agentes inteligentes es definir una arquitectura que soporte una comunidad extensa de agentes conscientes del contexto en ambientes inteligentes. Las investigaciones realizadas incluyen el modelado y razonamiento del contexto, conocimiento compartido y arquitectura propuesta para abordar lo anterior es CoBrA (del inglés *Context Broker Architecture*). El cual es un *middleware* con una arquitectura basada en agentes para desarrollar sistemas conscientes de contexto en ambientes inteligentes tales como casas, vehículos, oficinas y salas de juntas (Chen, 2004).

II.2.1 Arquitectura

El núcleo de CoBrA es un servidor especializado y es llamado *broker contextual*, el cual consiste de los siguiente componentes: Primero, una base de conocimiento del contexto. Segundo, tiene un mecanismo de razonamiento basado en reglas que definen como deducir el conocimiento a través de una ontología basada en OWL. Tercero, tiene un módulo de adquisición de contexto, el cual tiene como propósito principal el crear una librería de procedimientos para adquirir información contextual. Finalmente, tiene un módulo de gestión de políticas, el cual comparte la información del usuario con un agente, este módulo determina si el agente tiene el derecho de recibir la información, el *broker* comparte la información solo si el usuario predefinió las políticas que lo permiten (Chen, 2004).

II.2.2 Ontología

CoBrA difiere de otros *frameworks* que usan lenguajes de Web Semántica para expresar el contexto por medio de ontologías. CoBrA utiliza un grupo de ontologías llamada *Ontología Cobra*, que importa otras ontologías de alto nivel de la ontología SOUPA (Chen, 2004). La *Ontología Cobra*, importa vocabularios para expresar tiempo, espacio, políticas, redes sociales, acciones, ubicación de contexto, documentos y eventos.

La *Ontología Cobra* tiene dos principales elementos, los cuales son:

- **Ontología del contexto:** La ontología provee un conjunto de términos para representar el conocimiento.
- **Máquina de inferencia:** Tiene un sistema de lógica para razonar acerca de la información contextual de la ontología.

La Ontología Cobra se conforma de cuatro conceptos, los cuales son los que se muestran a continuación:

1. Los conceptos que definen lugares físicos y sus relaciones (por ejemplo; salones, edificios, campus universitarios)
2. Los conceptos que definen agentes (por ejemplo; agentes humanos y agentes de software)
3. Los conceptos que describen la ubicación del contexto (por ejemplo; cocina, auto, universidad)
4. Los conceptos que describen las actividades incluyendo los roles de los usuarios

II.3 CoDAMoS

El *Context-Driven Adaptation of Mobile Services* (CoDAMoS) es una arquitectura para resolver un conjunto de retos claves en el área de los Ambientes Inteligentes, donde los dispositivos personales móviles forman una extensión de cada ambiente del usuario, utilizando servicios móviles adaptados a el usuario y a su contexto.

II.3.1 Arquitectura

El núcleo del proyecto consiste en 5 paquetes de trabajo, que aborda la generación de la implementación del manejo del contexto, modelado del funcionamiento, integración del servicio, interfaces de usuario conscientes del contexto y código de movilidad (*CoDAMoS*, 2009).

II.3.2 Ontología

CoDAMoS propone una ontología extremadamente general para modelar contexto (*Bolchini, 2007*). La cual utiliza dos capas de representación de contexto utilizada en PACE (*Henricksen, 2005*), el cual es un *middleware* para sistemas conscientes de contexto que describe el contexto en términos de fina granularidad y descripción detallada.

El contexto se representa mediante una ontología la cual representa los conceptos y la posibilidad de expresarlos con diferente nivel de granularidad (*Bolchini, 2007*). La ontología representa las actividades de los usuarios, sus roles, tipos de usuarios, los dispositivos móviles que utiliza y la relación que existe entre ellos.

II.4 Red Bayesiana para tomar decisiones con base al contexto del Adulto Mayor

II.4.1 Arquitectura

El apoyo que brinda esta aplicación es que proporciona recordatorios para facilitar la vida independiente de los adultos mayores con síntomas asociados al declive cognoscitivo. Para entregar recordatorios, el sistema no solo considera un plan predefinido, sino que toma en cuenta el contexto actual del adulto mayor y las actividades que realiza frecuentemente (*Osmani, 2009*). Una de las posibles razones de que los adultos mayores dejen de tomar en cuenta estos recordatorios es porque quizá las alertas son imprudentes o innecesarias. Sin embargo, es una necesidad monitorear las actividades de los adultos mayores.

El sistema que se propone monitorea actividades a través del uso de dos componentes llamados *Red de Objetos* y *Módulo de Decisión*. El primero es una red de los objetos de uso diario que proveen una plataforma de monitoreo y procesamiento de información como un resultado de las acciones del usuario. Las acciones del usuario son típicamente descritas en términos de objetos que el usuario pueda manipular. El *Módulo de Decisión* infiere las actividades del usuario basadas en los eventos filtrados a través de la *Red de Objetos*. Estos eventos son alimentados a un repositorio predefinido de patrones de actividades del usuario conocidos como *Mapa de Actividades*. El *Mapa de Actividades* es un repositorio que guarda actividades que el usuario pueda realizar. La estructura interna del *Mapa de Actividades* corresponde a un grafo acíclico dirigido donde cada arco tiene asignado un valor de probabilidad. Esto esencialmente forma una red bayesiana. Un ejemplo del grafo se presenta en la Figura 3.

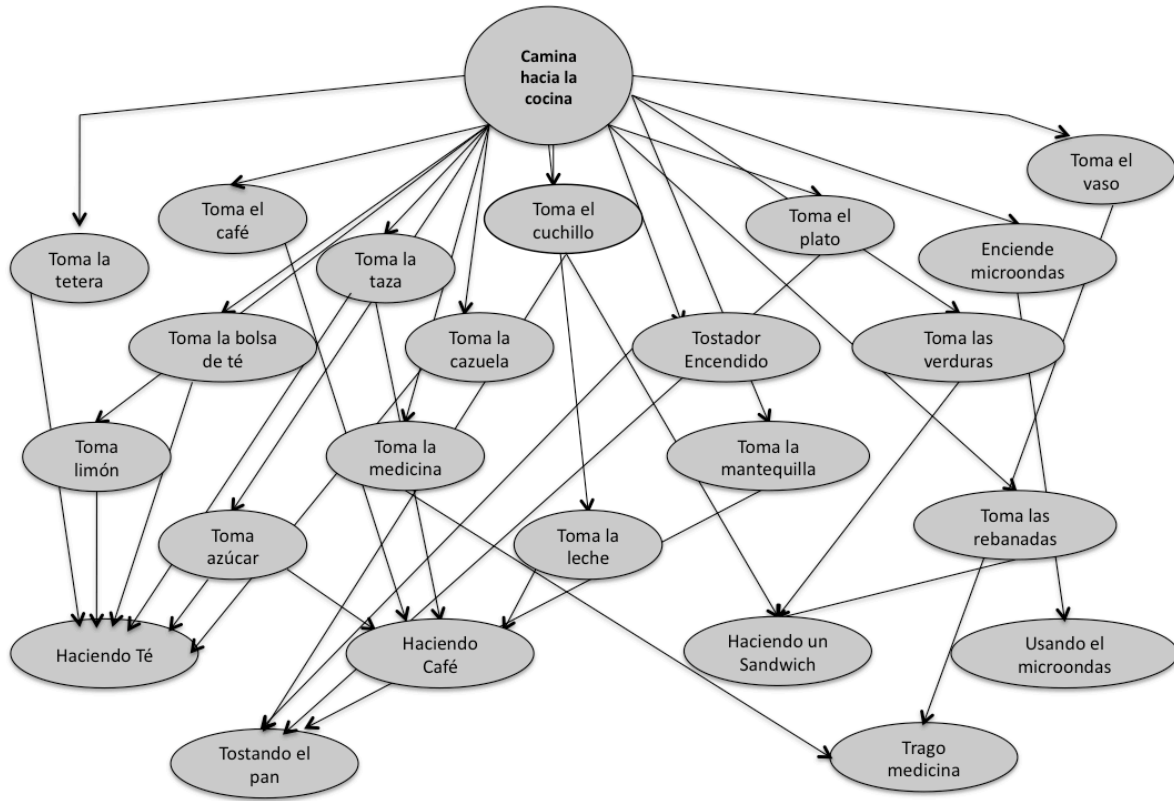


Figura 3. Red bayesiana para las actividades del hogar (Osmani, 2009)

II.4.2 Representación del Contexto

La Figura 3 muestra que cada actividad, por ejemplo *haciendo un sandwich*, tiene un conjunto de causas asociadas con la actividad que impacta el proceso de reconocimiento de la actividad cada uno de esas causales o causas tiene un peso específico en la actividad, representada a través de los valores de probabilidad condicional (las cuales son extraídas mediante una búsqueda en Google (Google, 2009)). Una vez que la actividad es reconocida, el sistema usa esta información para decidir como dar un aviso al adulto mayor (Osmani, 2009).

II.5 Sistema mySAM

El sistema *mySAM* (*my Smart Agenda Manager*) se construyó como un Sistema Multi Agentes (*MAS por sus siglas en inglés*), el cual está compuesto por sociedades heterogéneas de agentes que comparten la información del contexto sentido, modelan y razonan dicho conocimiento y también aprenden a discernir entre el contexto relevante y el no relevante para tomar decisiones acertadas en un futuro. Usualmente, el contexto relevante es asumido y los agentes tienen que aprender cómo adaptarse a él.

El sistema que se diseñó, para abordar lo anterior, es el que está compuesto por los agentes de *mySAM* y pueden ser ejecutados en PC o dispositivos móviles del tipo PDA (*Personal Digital Assistant*). Los agentes de *mySAM* asisten al usuario mediante negociaciones con ellos usando la información contextual para aceptar o rechazar una propuesta de encuentro hecho por otro usuario (*Bucur, 2005*).

II.5.1 Arquitectura

Se propuso una arquitectura en capas, el cual está estructurado de la siguiente manera:

- Capa 0: Captura el contexto mediante sensores
- Capa 1: Capa que gestiona y maneja el contexto sentido
- Capa 2: Esta capa está compuesta por los *Administradores del Contexto*
- Capa 3: Capa dedicada a los agentes encargados del razonamiento del contexto
- Capa 4: Nivel de aplicación del sistema

La capa dos está compuesta por lo que se llama *Administrador del Contexto*, uno por cada grupo de agentes en cada capa. La tarea de estos es manejar la información contextual disponible en cada área de los correspondientes grupos de agentes. Los agentes que operan en la tercera capa, usan una retroalimentación de la información contextual que los ayuda a decidir y a resolver problemas.

Para asegurar un común entendimiento, el *Administrador del Contexto* usa una representación ontológica, la cual mejora sus capacidades basándose en otras experiencias. Los agentes pueden compartir con otros agentes la manera de cómo resolvieron problemas similares, ya que no importa a qué grupo o capa pertenezcan ya que tienen una manera específica de preguntarse e identificarse unos a otros acerca de la información contextual.

El propósito de esta aplicación es facilitar la tarea de los diseñadores de sistemas AmI al diseñar las ontologías y definir sus atributos.

II.5.2 Representación del Contexto

El propósito es representar el contexto de una manera general y manejable para todas las aplicaciones que necesiten representar un razonamiento acerca de él. El enfoque principal es tener una representación genérica y compleja para el *atributo contextual*, la representación que asegurará la interoperabilidad, pero también la posibilidad de representar los atributos contextuales que no importan. Para esto se definen algunas ontologías para la representación del contexto usando el lenguaje OWL (*Bucur, 2005*).

II.6 Middleware Gaia

Es un *middleware* que promueve el uso de los agentes de software en el desarrollo de AmI. Esta basado en un modelo contextual de predicados y soporta el uso de mecanismos de razonamiento diferentes como la lógica de primer orden y la lógica temporal de los agentes.

Gaia usa ontologías para definir la semántica de varios contextos, estas definen la estructura y las propiedades de la información contextual. Permiten a los agentes tener un común entendimiento entre ellos y genera un rápido prototipo de agentes conscientes del contexto para la construcción de AmI. Tiene un razonamiento para asegurar la interoperabilidad entre agentes a través del uso de ontologías.

II.6.1 Arquitectura

El *middleware* Gaia provee diferentes formas en lo que los agentes adquieren varios tipos de información contextual y razonan acerca de eso. Un diagrama de la infraestructura de Gaia se presenta en la Figura 4, la cual contiene los siguientes elementos.

- **Proveedor Contextual.** Son sensores u otras formas de fuentes contextuales.
- **Sintetizador Contextual.** Este recibe el contexto sentido del Proveedor Contextual, para deducir o abstraer contexto de alto nivel a través de un simple dato capturado y proveen el contexto deducido a otros agentes.
- **Consumidor Contextual.** Estos obtienen información a través de los dos anteriores, después razonan el contexto actual y lo adaptan su comportamiento acorde a dicho contexto.
- **Buscador.** Permite que los agentes encuentren a los apropiados Proveedores Contextuales. Existe uno por cada AmI.
- **Historial del Contexto.** Almacena el contexto pasado en una base de datos.
- **Servidor de Ontologías.** Mantiene las ontologías que describen la información contextual.

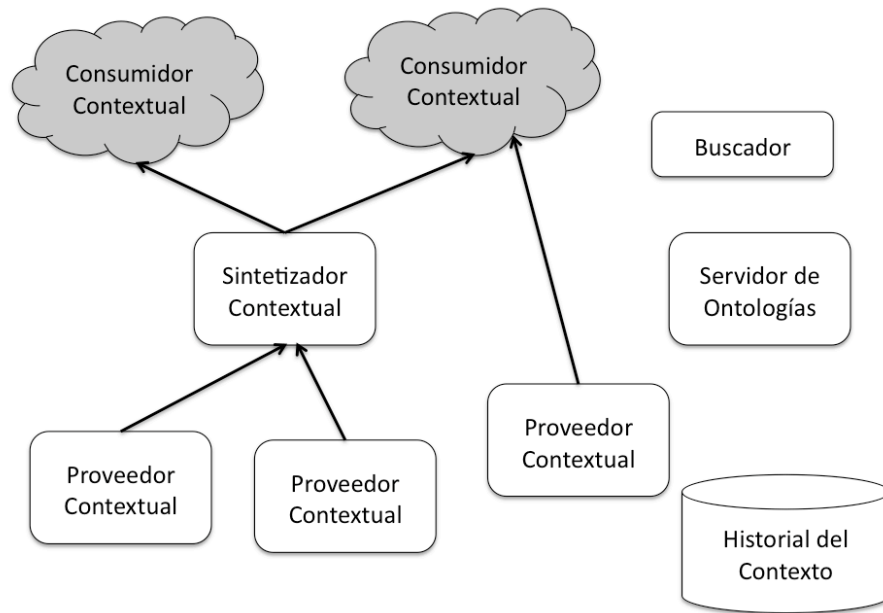


Figura 4. Arquitectura de Gaia (Guarino, 1998)

II.6.2 Representación del contexto

Para representar el contexto, Gaia utiliza predicados con lógica de primer orden que siguen la convención de que el nombre del predicado es el tipo de contexto que está descrito (como la ubicación, temperatura o el tiempo). Esta convención permite tener una representación simple y uniforme para diferentes clases de contexto. Además permite describir fácilmente los diferentes contextos en una ontología, esto se hace posible mediante operadores relacionales como “-“ y “<” que argumenten el predicado.

La estructura de los predicados contextuales es especificada en una ontología (Guarino, 1998). Cada tipo de contexto corresponde a una clase en la ontología. Esta ontología define varios tipos de contextos así como de argumentos que se permitan en los predicados.

La función principal de la ontología es revisar la validez de los predicados y también hacer más fácil su escritura, ya que se conoce la estructura del predicado y la clase de valores que debe de tomar los argumentos. También permite la interoperabilidad de los sistemas Aml porque hace posible la traducción entre términos usados en ontologías de este mismo ambiente.

Además de que permite la descripción del contexto de un sistema que se programa independientemente con otro lenguaje, sistema operativo o *middleware*. El modelo del predicado del contexto es también genérica y permite diferentes mecanismos de razonamientos. Por ejemplo, es posible escribir reglas usando predicados de contexto que describen la aplicación usando diferentes lógicas de programación. Es posible desarrollar otras clases de inferencia basadas en los predicados utilizando Redes Bayesianas, Redes Neuronales y otros razonamientos (*Ranganathan, 2003*).

II.7 Ontología COSAR

La forma mas común para el reconocimiento de actividades es mediante métodos de aprendizaje estadísticos. El modelo obtenido es utilizado para clasificar nuevas instancias de actividades, la cuales están caracterizadas por la duración, persona y otras. Cada una de estas instancias se representan mediante un vector llamado *feature*, en el cual cada característica (*feature*) corresponde a una medida o valor (propiamente estadístico extraído de un sensor o un conjunto de sensores durante la duración de la actividad instanciada).

II.7.1 Arquitectura

La solución que se propone en COSAR (*Combining Ontological Reasoning and Statistical Activity Recognition*), para poder reconocer las actividades esta basado en una ontología (*Horrocks, 2003*), la cual representa las características mencionadas anteriormente, así como las relaciones entre ellas. Esta ontología llamada COSAR-ONT, incluye 30 símbolos de ubicación dentro de un mismo entorno y 35 actividades.

II.7.2 Ontología

Para poder explicar su lógica se seguirá el siguiente ejemplo: Considerando la actividad *Brushing Teeth* (lavarse los dientes), infiere automáticamente que esta actividad puede realizarse en un lugar donde exista un *sink* (lavamanos) (las cuales están definidas en la subclase *WaterFixture* de la COSAR-ONT). Ahora se consideran dos ubicaciones simbólicas, llamadas *RestRoom* (baño) y *Living Room* (sala), acordando que *RestRoom* (baño) es un cuarto que contiene un lavamanos, mientras que la sala es un cuarto que no contiene un lavamanos. Dado estos resultados las definiciones ontológicas posibles se derivan a través del razonamiento de COSAR-ONT. Esto se puede deducir porque la ontología tiene una parte llamada *ABox*, donde se crea una instancia de la actividad identificando la actividad actual e instanciando el lugar donde está la persona.

II.8 Sistema I.L.S.A.

El sistema I.L.S.A. (de sus siglas en inglés: *Independent LifeStyle Assistant*), tiene como propósito monitorear y apoyar a los adultos mayores a prolongar su estancia en su propio hogar el mayor tiempo posible y así reducir la probabilidad de que requieran una persona encargada de cuidarlos dentro de su hogar. I.L.S.A. (Haigh, 2003) es un sistema multi-agentes que incorpora un modelo unificado de sensado, modelado de situaciones o contexto, planeaciones, respuestas en tiempo real y una máquina de aprendizaje. Su objetivo principal es integrar un conjunto de dispositivos individuales (sensores) y dependiendo de los datos monitoreados por ellos crear un razonamiento coherente, inteligente y útil con la finalidad de que el adulto mayor disfrute su vida dentro de su hogar de la mejor manera posible e independientemente.

II.8.1 Arquitectura

El sistema que se implementó para I.L.S.A. tiene las siguientes características:

- **Monitoreo Pasivo:** Monitoreo de las Actividades de Vida Diaria como la medicación y movilidad.
- **SopORTE cognitivo:** Recordadores para el adulto mayor de fecha/hora del día para que se tomara su medicamento.
- **Alertas y Notificaciones:** Para que los cuidadores supieran si el adulto mayor estaba bien o mal (notificaciones vía telefónica).
- **Reportes:** Sumario de las actividades del adulto mayor.
- **Acceso Remoto a la información:** Vía Internet o telefónica.
- **Control:** Modos de apagado/encendido.

I.L.S.A. se desarrolló fácil de configurar y fácil de actualizar. Su arquitectura se basa en agentes que proveen modularidad, distribución, funcionalidad y capacidad dinámica de descubrimiento. También se desarrolló una ontología explícita. Se desarrolló en JADE (JADE, 2008) como plataforma de desarrollo de agentes.

Los agentes del sistema incluyen los controladores de dispositivos, agentes de dominio, *response planners* y de gestión. Los agentes creados para el cliente (Adulto Mayor) son *Medication*, *Mobility*, *Modes (on/off)*, *Reminders*, *ResponseCoordinator*,

Machine Learning. La interfaz de usuario esta equipada con los Honeywell Web Pads (*Honeywell Web Pads, 2008*) con acceso a Internet inalámbrico. A través de estas, los adultos mayores pueden desplegar recordatorios, calendarios de medicación y estatus, resumen de la movilidad y controles de encendido/apagado e información acerca de sus cuidadores.

Los cuidadores pueden acceder a los datos que genera I.L.S.A. acerca de los adultos mayores y los miembros de su familia desde su servidor y con la conexión *ISP Web Connection*. La interfaz permite ver y conocer las alertas que se han generado y el estatus general de las AVD realizadas por el adulto mayor a lo largo del día, historiales de la medicación y su movilidad, además de que permite personalizar las alertas.

II.8.2 Ontología

Se implementó la *Consolidated Home Ontology in Protégé* (CHOP) (*Myers, 2003*), la cual sirve para dos principales propósitos; primero, como vocabulario común entre los conceptos creados por I.L.S.A. y sus relaciones. Segundo, es una interfaz de comunicación entre los agentes creados para este sistema.

CHOP es una ontología que contiene mas de 800 conceptos distintos para poder representar lo anterior. Contiene varios conceptos, entre ellos están la configuración de agentes, los estados del ambiente y los estatus de comunicación. CHOP representa una forma ontológica para desarrollar conocimiento reusable y para la asistencia de sistemas en el hogar.

II.9 Framework CASIS

II.9.1 Arquitectura

El *framework* CASIS (que por sus siglas en inglés: *Context-Aware Service Integration System*) es un sistema orientado a servicios y administrado por eventos integrando a su vez los servicios de *web* y la plataforma *OSGi*. Provee una plataforma de servicios para el cuidado de la salud y permite a los cuidadores o familiares del adulto mayor monitorear sus actividades remotamente, ya que la residencia del anciano será un “ambiente inteligente”. Dicho ambiente interactúa con el adulto mayor a través de una variedad de aplicaciones que permiten la representación del contexto, esto gracias a sensores y cámaras empotradas en el hogar, después la persona recibe mensajes por medio de bocinas, la televisión, así como sus dispositivos móviles.

CASIS explora como las tecnologías pueden aumentar la calidad de vida de los adultos mayores al vivir en un ambiente inteligente. CASIS provee los servicios de Mobiliario Inteligente (*Smart furniture*), Servicios de Información Consciente del Contexto (*Context-Aware Information Services*) y Servicios de Salud (*Healthcare Services*). Su arquitectura esta desarrollada en una plataforma multi-agente orientada a los servicios conscientes del contexto con hardware heterogéneos e integrando componentes de software con servicios web. La arquitectura de CASIS consiste en un Broker de Eventos Contextuales de contexto que almacena dicha información en un repositorio para después enviarlo al visualizador contextual el cual le da un formato especial al contexto. Los sensores están conectados al Agente de Dispositivos. Los Eventos generados por el Agente de Dispositivos o por el Agente de Inferencia se envía al Broker de Eventos Contextuales por Servicios Web. Estos intercambian mensajes para la interacción con CASIS y agentes de software externos. Así las personas externas del hogar del adulto mayor podrán cuidar de él accedando a la información contextual acorde a sus roles y responsabilidades mediante un navegador y esta acción es administrada por el Componente de Control de Accesos (*Wan-rong, 2006*).

II.9.2 Representación del Contexto

Para la representación contextual CASIS utiliza el lenguaje RDF y XML, el primero lo utiliza para una relación lógica. El segundo para una especificación de la sintaxis. Como se muestra en el siguiente ejemplo:

```
<context>  
<subject>Mr.Chang</subject>  
<predicate>activity</predicate>  
<object>sleeping</object>  
</context>
```

...

```
<context>  
<subject>telephone</subject>  
<predicate>state</predicate>  
<object>auto-answer</object>  
</context>
```

II.10 Discusión

Los trabajos relacionados a esta propuesta tienen ciertas similitudes, pero también algunas diferencias con el enfoque general de esta investigación, las cuales se resaltan a continuación.

SOCAM provee una infraestructura que soporta la construcción de ambientes inteligentes (Gu, 2005). Propone y define una ontología de alto nivel que incluye conceptos generales (tales como, la Entidad Computacional, la Actividad, la Ubicación, la Persona y el Horario de sus actividades), y además propone ontologías de dominio específico que incluye los conceptos propios de este (tales como, el tipo de Habitación, Corredor o la entrada. El jardín o el estacionamiento). Para ilustrar la utilidad de su propuesta, los autores presentan el diseño de una ontología vehicular que se enlaza con la ontología de alto nivel. CoDAMoS es una arquitectura para que los dispositivos personales móviles sean una extensión del ambiente del usuario (CoDAMoS, 2009). Similar a SOCAM, CoDAMoS incluye una ontología considerada de alto nivel para modelar contexto del usuario, el ambiente y la plataforma del sistema. Entre los conceptos que permite modelar CoDAMoS están el Usuario, el Rol de este, su Perfil, la Tarea que realiza, el Servicio que esta utilizando, la Plataforma y el Software de este Servicio. Otra arquitectura que de acuerdo a los autores incluye una ontología general para administrar contexto, es el sistema mySAM, el cual propone una arquitectura donde existe una separación entre la distinción del contexto relevante y la toma de decisiones del agente. El propósito es representar el contexto de una manera general y manejable para todas las aplicaciones que necesiten representar un razonamiento acerca de él en unas ontologías. COSAR propone reconocer las actividades mediante una ontología que representa ubicaciones, rutas de comunicación, actividades, artefactos, personas y tiempo así como las relaciones entre ellas.

Como trabajo relacionado, también se identificaron arquitecturas que permiten desarrollar aplicaciones Aml para un dominio específico. Estas arquitecturas no utilizan las ontologías para representar contexto, sino para propósitos diferentes tal como se explica a continuación. CoBrA es un *middleware* con una arquitectura basada en agentes para desarrollar sistemas conscientes de contexto en ambientes inteligentes en oficinas y salas de juntas. La *Ontología Cobra* incluye una ontología, que es el vocabulario que permite

expresar tiempo, espacio, políticas, redes sociales, acciones, ubicación de contexto, documentos y eventos del sistema. En este caso, la ontología no permite plasmar las características generales o de alto nivel de cualquier ambiente de inteligencia ambiental, sino contiene conceptos y relaciones que caracterizan al dominio particular de oficinas, el donde los usuarios se reúnen para tomar decisiones mediante la interacción y el intercambio de documentos.

Similar a CoBrA, el sistema I.L.S.A. se propuso para facilitar el desarrollo de aplicaciones AmI para un dominio específico. En este caso, I.L.S.A. es una arquitectura que facilita el desarrollo de sistemas AmI para apoyar la independencia del adulto mayor. Al momento de realizar el estudio de trabajo relacionado para proponer esta tesis, se identificó que este era de los pocos trabajos propuestos para este dominio y por sus similitudes con la propuesta de esta tesis se decidió analizarla. El sistema I.L.S.A. tiene como propósito monitorear y apoyar a los adultos mayores a prolongar su estancia en su propio hogar el mayor tiempo posible y así reducir la probabilidad de que requieran una persona encargada de cuidarlos dentro de su hogar. Como trabajo relacionado, fue de utilidad analizar aquellas propuestas que aunque no presentaban arquitecturas que faciliten desarrollar sistemas AmI, presentaban aplicaciones AmI que incluían mecanismos de modelado para el contexto de los usuarios. Durante el desarrollo de esta tesis, se identificó el trabajo presentado por Osmani (*Osmani, 2009*), en el que se propone una red bayesiana en el dominio del adulto mayor. Este proporciona recordatorios para facilitar la vida independiente de las personas con síntomas asociados al declive cognitivo formando un mapa de actividades que corresponde a un grafo acíclico dirigido donde cada arco tiene asignado un valor de probabilidad. Gaia es un *middleware* que está basado en un modelo contextual de predicados y soporta el uso de mecanismos de razonamiento diferentes como la lógica de primer orden y la lógica temporal de los agentes. Utiliza ontologías para definir la semántica de varios contextos, estas definen la estructura y las propiedades de la información contextual. El *framework* CASIS provee una plataforma de servicios para el cuidado de la salud y permite a los cuidadores o familiares del adulto mayor monitorear sus actividades remotamente. El sistema interactúa con el adulto mayor a través de una variedad de aplicaciones que permiten la representación del contexto, esto gracias a

sensores y cámaras empotradas en el hogar, después la persona recibe mensajes por medio de bocinas, la televisión, así como sus dispositivos móviles.

Las ontologías incluidas en las arquitecturas explicadas anteriormente tal como SOCAM, CoBrA, CoDAMoS, mySAM, Gaia, COSAR, I.L.S.A y CASIS tienen como propósito la representación de contexto de los usuarios, pero ninguna de ellas representa el contexto generado en el dominio que interesa abordar en esta tesis: adultos mayores realizando sus Actividades de la Vida Diaria (AVD) en el hogar. Sin embargo, otros trabajos, tal como el presentado por *Osmani, et al (2009)*, identifica la necesidad de inferir con base al contexto generado por el adulto mayor en su hogar, para lo que propone el uso de redes bayesianas, sin embargo en este trabajo se utiliza este mecanismo de razonamiento y no de representación de contexto. Similarmente, las arquitecturas I.L.S.A., CASIS, CoBrA, CoDAMoS y Gaia cuentan con mecanismos de razonamiento para inferir contexto secundario con base a sus ontologías, pero ninguno de estas profundiza en cuanto a la inferencia de riesgos en las actividades de los usuarios.

A partir del trabajo presentado en esta sección, se identifica la necesidad de desarrollar ontologías de dominio específico (también llamadas de bajo nivel), lo cual no es una tarea sencilla ya que se requiere de un análisis profundo para identificar aquellos conceptos y relaciones que caracterizan a la información de contexto de los adultos mayores en el hogar. El propósito de esta tesis es crear un componente de *software* basado en agentes que contenga una ontología lo suficientemente flexible para permitir a los desarrolladores cambiar fácilmente las condiciones contextuales para inferir riesgos en las actividades de vida diaria del adulto mayor, dentro de su hogar. Además de encapsular también mecanismos que permitan inferir posibles riesgos en las actividades y solicitar historiales contextuales.

Siguiendo la metodología propuesta, se continúa con el análisis de escenarios de sistemas de inteligencia ambiental para abstraer los requerimientos del componente en el contexto que rodea al adulto mayor.

Capítulo III

Análisis de Escenarios de Sistemas de Inteligencia Ambiental

III.1 Escenarios de AmI para apoyar ABVD y AIVD

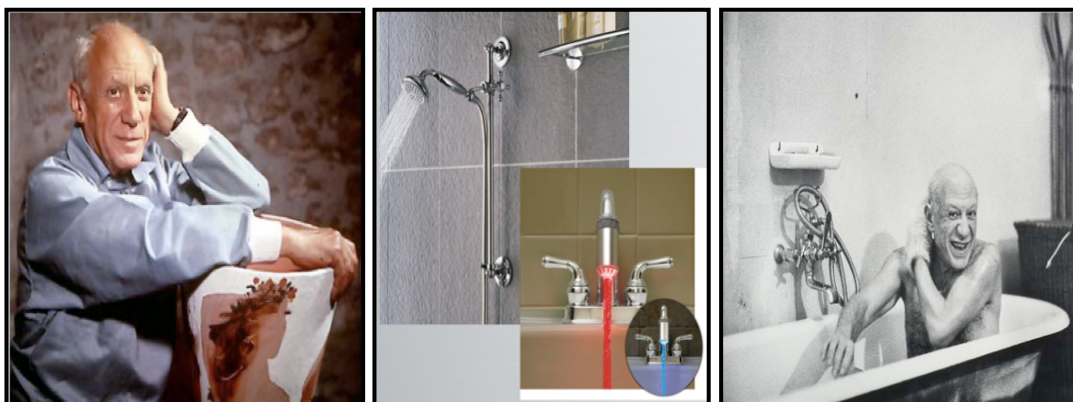
III.1.1. Evitando riesgos de quemaduras y sofocación al bañarse

Se exploraron diferentes escenarios de uso que posteriormente se analizaron. Estos se propusieron para apoyar las ABVD y AIVD ya que de acuerdo con *Lawton (1990)* estas son las actividades que las personas requieren realizar con autonomía para vivir de forma independiente en su hogar o comunidad. Específicamente, las actividades que se analizaron y se propusieron apoyar con tecnología AmI son: bañarse y medicarse.

Los adultos mayores son susceptibles a quemarse con el agua cuando preparan su baño, por lo que es necesario dotar su ambiente de tecnología que les faciliten realizar dicha actividad (*Capetuti, 2007*). En *Rodríguez, et al (2008)* se presentó el diseño e implementación de un sistema propuesto para evitar los riesgos que puede enfrentar un adulto mayor por bañarse con agua caliente. Este sistema consistió de componentes de hardware y software que permitieron iluminar el agua que sale del grifo cuando su temperatura es mayor a 54 grados centígrados. A continuación se presenta el escenario de uso de este sistema.

III.1.1.1 Escenario de Uso

“Pablo es un adulto mayor de 70 años (Figura 5a), por las noches prepara su baño. Por su edad su piel se volvió más delgada, haciéndolo propenso a mayores daños. Él gira la perilla de las llaves de la regadera, ésta tiene un dispositivo integrado que le notifica cuando el agua está a una temperatura adecuada (Figura 5b). El agua empieza a salir caliente, entonces la regadera se torna de un color rojo para que Pablo no meta sus manos. Pablo regula la temperatura del agua y la regadera se torna de un color azul indicándole que puede bañarse (Figura 5c) (García, 2009)”.



5a) Adulto Mayor Pablo

5b) Regadera

5c) Pablo bañándose

Figura 5. Escenario de Uso

III.1.1.2 Discusión

Los aspectos para el diseño del Componente identificados en el escenario anterior son los siguientes:

- a) **Información del contexto que requiere ser representado.** El contexto identificado en este escenario es primeramente la ubicación del adulto mayor la cual es el baño, la acción que ejecutó al abrir la llave de la regadera, la temperatura del agua y la hora en la que se realizó dicha actividad.
- b) **Inferencia del riesgo.** El contexto secundario a inferir, será si la temperatura del agua es la adecuada para que el adulto mayor se pueda bañar. Si la temperatura rebasa los 54 grados centígrados, entonces el agua de la regadera se tornará de un color rojo. Si por el contrario la temperatura del agua es por debajo de los 54 grados, entonces el agua de la regadera se tornará de un color azul.
- c) **Preferencias y necesidades del adulto mayor.** Debido a que cada adulto mayor es diferente el sistema debe de tener la capacidad de ajustarse a las preferencias del adulto mayor y la tolerancia de su piel y de su sistema respiratorio para soportar la temperatura del agua y sus consecuencias.

III.1.2 Apoyo al cumplimiento de la Medicación mediante Recordatorios Oportunos

Otra actividad de vida diaria, del tipo instrumental, que se considera crítica para que el adulto mayor viva con independencia es la de administrar su medicación (*García, 2010*). Se estima que del 26% al 59% de los adultos mayores tienen problemas con la medicación, ya sea que se les olvida medicarse, toman medicamentos caducos, no surten sus recetas, entre otras (*Orwing, 2006*). Es por esto que es necesario proveer al adulto mayor de tecnología que asesore esta importante actividad que los ayuda a mantener una mejor calidad de vida. En *Espinoza, et al (2009)* y *García, et al (2010)* se presentan sistemas AmI que fueron diseñados para asistir al adulto mayor a medicarse. En la Figura 6 se muestran imágenes que presentan estos sistemas, los cuales se llaman REMIND-Me y GUIDE-Me. El primer sistema (REMIND-Me), consiste de un dispositivo vestible que provee recordatorios auditivos al adulto mayor para indicarle que se medique. El segundo sistema (GUIDE-Me), son dispositivos que se anexan a los contenedores de medicamentos y que se enfatizan para indicarle al adulto mayor cuales medicamentos debe de tomar, así como le provee información relevante a su medicación, tal como la dosis y padecimiento que se aborda con el medicamento. Para más información de estos sistemas se pueden consultar los artículos (*Espinoza, 2009*) (*García, 2010*), respectivamente. A continuación, se presenta el escenario de uso de estos sistemas, el cual fue analizado para identificar sus características o aspectos de consciencia de contexto que se abordan con tales sistemas.

III.1.2.1 Escenario de Uso

“Son las 8:00 PM y la Sra. Ana entra a la cocina para prepararse su cena (Figura 6a), pero no recuerda que es hora de su última medicación. Mientras prepara su comida, el sistema AmI en su hogar, infiere que existe una alta probabilidad de que olvide medicarse, entonces la pulsera vestible (REMIND-Me) emite una notificación auditiva para recordarle que debe medicarse (Figura 6b). Posteriormente se acerca a su repositorio de medicamentos (GUIDE-Me), el cual le presenta información sobre los medicamentos a tomar, dosis y el padecimiento para lo cual fueron prescrito (Figura 6c)”.



Figura 6a) Sra. Ana



Figura 6b) Notificación Auditiva



Figura 6c) Dispensador

Figura 6. Escenario de uso de la Sra. Ana

III.1.2.2 Discusión

Los aspectos para el diseño del Componente identificados en el escenario anterior son los siguientes:

- a) **Información del contexto que requiere ser representado.** El contexto primario identificado en este escenario es la hora en la que entró a la cocina, la ubicación ya que es importante saber si el adulto mayor está en el lugar donde debe medicarse. También las acciones realizadas por el adulto mayor desde que entró a la cocina.
- b) **Inferencia del riesgo.** El contexto secundario a inferir es si es su hora de medicación la hora de medicación y la actividad que esta haciendo en ese momento. Si no está realizando la actividad de medicarse, entonces se genera un recordatorio auditivo para indicarle que debe medicarse.
- c) **Preferencias y necesidades del adulto mayor.** Cada persona tiene su propia prescripción médica, sus propias actividades y rutinas. Por lo tanto, el sistema tiene la facilidad de adaptarse al entorno y circunstancias de la vida diaria del adulto mayor.

III.1.3 Motivando el Cumplimiento de la Medicación

III.1.3.1 Escenario de Uso

“Hoy es domingo por la noche, y como es costumbre, la Sra. María está en su sala disfrutando de su café y leyendo un libro (Figura 7a). De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción medica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida (Figura 7b). Así que la Sra. María se dirige a la cocina donde están sus medicamentos, toma un vaso de agua y se acerca a su dispensador para tomar su prescripción médica (Figura 7c). El sistema esta consciente de que María cumplió correctamente con la medicación del día de hoy, por lo que el perico que representa la semana actual aumenta de tamaño. Adicionalmente, cómo la semana ha terminado, aparece un nuevo perico para representar la semana que empieza. Posteriormente, María se retira a su cuarto para prepararse a dormir” (García, 2010).



7a) Sra. María



7b) Visualizador Ambiental



7c) Dispensador

Figura 7. Escenario de uso de la Sra. María

III.1.3.2 Discusión

Los aspectos para el diseño del componente identificados en el escenario anterior son los siguientes:

- a) **Información del contexto que requiere ser representado.** El contexto primario identificado en este escenario es la hora del día, la ubicación del adulto mayor y la dosis de medicamento que tomo del dispensador.
- b) **Inferencia del riesgo.** El contexto secundario a inferir es si es la hora de medicación para el adulto mayor. Si se cumple esta condición, entonces se notifica mediante un visualizador ambiental (Figura 7b) que tiene unos periquitos que emiten sonido y le recuerdan a la persona que debe medicarse.
- c) **Preferencias y necesidades del adulto mayor.** Cada persona tiene su propia prescripción médica, sus propias actividades y rutinas. Por lo tanto, el sistema tiene la facilidad de que el desarrollador de AmI pueda adecuar preferencias y necesidades de cada adulto mayor, para así adaptarse a su entorno.

III.2 Requerimientos del Componente para desarrollar Sistemas AmI

III.2.1 Proveer un modelo de representación del contexto primario y secundario de los adultos mayores ejecutando sus Actividades de Vida Diaria

En los Sistemas Conscientes del Contexto, la representación del contexto es importante para desarrollar una aplicación AmI porque permite a una entidad de *software* determinar las acciones, que son consecuencia del contexto representado, mediante un razonamiento inteligente y así ejecutar una acción oportuna para apoyar, asesorar o notificar algún riesgo en una AVD.

Para resolver el reto de la representación contextual en los sistemas conscientes del contexto, se proponen varias técnicas. Algunas de ellas, comúnmente utilizadas son el lenguaje *XML* (del inglés: *eXtensible Markup Language*), las ontologías y la lógica de predicados.

Para propósitos de esta tesis se ha seleccionado a las ontologías como técnica de representación del contexto, por considerarlas una herramienta que ayuda a plasmar el ambiente de una manera específica.

El término ontología viene del campo de la filosofía que concierne el estudio del ser o de la existencia. En filosofía, se puede hablar acerca de una ontología como una teoría de la naturaleza de la existencia. En computación y en ciencias de la información, la ontología es un término técnico que denota un artefacto diseñado con el propósito de modelar el conocimiento acerca de algún dominio real o imaginario.

El término fue adoptado por el área de conocimientos de la Inteligencia Artificial, quienes reconocen la aplicabilidad del trabajo en la lógica matemática (McCarthy, 1980). Por consiguiente, una definición mas adecuada es la que menciona que: “*Una ontología define los conceptos, relaciones y otras distinciones que son relevantes para modelar un dominio*” (Gruber, 2008). Las especificaciones son definiciones de vocabularios que representan el ambiente, los cuales proveen significados para un vocabulario formal, constante y de uso coherente.

Sin embargo la definición mas utilizada de ontología es la dada por Tom Gruber, la cual indica que: “*Es una especificación explícita de una conceptualización*” (Gruber, 1993) esto explica que una ontología es una descripción de los conceptos y de sus relaciones que puedan existir por un agente o una comunidad de agentes. Esta definición es consistente con el uso de la ontología como un conjunto de definiciones conceptuales en general. Una ontología explica la “existencia sistemática” del ambiente.

III.2.2 Proveer de mecanismos que facilite adaptar el comportamiento del sistema AmI al perfil, preferencias y necesidades de los usuarios para realizar sus Actividades de Vida Diaria

La representación del contexto por sí sola no ofrece la inteligencia necesaria para inferir o decidir las acciones que debe realizar un Sistema Consciente del Contexto, es decir dotarlo de inteligencia, se requieren de técnicas de la Inteligencia Artificial donde se aplican mecanismos de manipulación del conocimiento. Los sistemas basados en reglas incorporan una significativa cantidad de conocimientos expertos acerca de un dominio bien delimitado y restringido, y pueden usar técnicas de razonamiento simbólico para resolver problemas en dicho dominio (Pajares, 2006). Por lo tanto, un sistema basado en reglas es básicamente un programa con conocimiento que incorpora de forma computacional la información de un experto humano para realizar una *tarea X* en un *dominio Y*, de forma que para esa tarea, la

eficiencia que subyace del programa es comparable a la que muestra el experto humano (Pajares, 2006).

III.2.3 Desarrollar un componente que encapsule la funcionalidad descrita anteriormente

Se propuso desarrollar un componente de *software* porque la filosofía de la teoría de componentes parte de la idea principal de que la industria desarrolla actualmente muchos componentes de *software* para poder adaptarlos a diversas áreas, además muchos desarrolladores pueden acceder a ellos y reutilizarlos fácilmente. Bajo esta filosofía, se producen aplicaciones de software (componentes) para ser ensambladas con otras, fabricadas por diferentes personas y así desarrollar un nuevo producto (Ahmed, 2007). Partiendo de esta premisa, se observa que en los AmI no es la excepción, ya que día con día se desarrollan mas aplicaciones que pueden ser reutilizadas por otras personas que programan ambientes inteligentes. Sin embargo, hasta nuestro conocimiento no existen componentes destinados a facilitar el desarrollo de sistemas AmI para apoyar las actividades de vida diaria de los adultos mayores. Y en particular, no existen componentes que aborden los retos relacionados con la representación e inferencia de contexto de los usuarios.

Los componentes de software emergieron como un elemento principal en el desarrollo de sistemas complejos de software. Szyperski (2002) lo define como:

“Un componente de software es una unidad de composición con interfaces especificadas y dependencias explícitas. Un componente de software puede ser ejecutado independientemente y es sujeto de composición con terceros”.

Otra definición de componente es la siguiente:

“Un componente es una unidad de desempeño independiente que interactúa con su ambiente a través de interfaces bien definidas las cuales encapsulan su implementación (Griss, 2001)”.

Algunas de las características de los componentes de software son las que se presentan a continuación:

- Es una unidad de composición de aplicaciones de software.
- Posee un conjunto de interfaces.
- Satisface un conjunto de requisitos.
- Puede ser desarrollado o adquirido, para incorporarlo a un sistema.
- Puede estar compuesto por otros componentes.
- Permite el desarrollo de aplicaciones a un mayor nivel de abstracción.

Las ventajas del uso de los componentes de *software* son varias, entre las más importantes se encuentran la calidad, debido a que varias personas están involucradas en el desarrollo y enriquecen el producto final. Otra de las ventajas es que se reducen los riesgos y el tiempo para el desarrollo de la aplicación. La facilidad de desarrollo y la flexibilidad son otras de las ventajas que tiene el uso de componentes ya que por su misma portabilidad son fáciles de integrar, desarrollar y son flexibles para conectar con otros componentes de sistemas distintos.

Un componente de software está formado por:

- **Interfaz.** Es una definición de una colección de uno o más métodos, y cero o más atributos. Idealmente se define como un conjunto de comportamientos. Están conectadas a un puerto, el cual es una característica de clasificación que especifica una interacción distinta entre los clasificadores y su ambiente. Los *puertos* son descritos por pequeños cuadros en los lados de los clasificadores.
- **Receptáculos.** Conexiones orientadas a establecer contacto entre una interfaz. Representan las dependencias con otros componentes.
- **Atributos.** Nombre del componente, tipo, valor inicial y otras propiedades.

El uso de componentes se justifica porque son útiles y unifican marcos de trabajo, proveen una estructura y cierta semántica. Si el sistema no está basado en componentes o no tiene una arquitectura identificable, los otros disponibles para conectarse con el se harán menos valiosos y en algunos casos tal vez más raros o difíciles de implementar conforme a la interfaz del componente (*Hopkins, 2000*). Con respecto a la calidad, el usar componentes mantiene al diseño e implementación de esta arquitectura estable o independiente, ya que si existe algún error en algún componente, este se puede modificar y lo harán poco a poco

susceptibles a errores o corrupciones entre ellos. Además de que se puede aumentar la granularidad; detallándolo grado por grado hasta que resulte ser específico para la solución del problema.

III.2.4 Facilitar que el desarrollador pueda especializar y extender la funcionalidad del componente para implementar aplicaciones AmI.

Para la implementación del componente se propuso utilizar Agentes de *Software*, ya que estos ofrecen características de autonomía, reactividad, colaboración y adaptación, las cuales son características que se comparten con los AmI (Rodríguez, 2005). Es decir, los sistemas AmI están formados de componentes o sub-sistemas que autónomamente deciden como actuar y cuales servicios proveerles a los usuarios con base a su contexto. Para lograr lo anterior, los componentes requieren ser reactivos al contexto de los usuarios y tener capacidades para comunicarse o colaborar con otros componentes del ambiente. Uno de los usos de los Agentes de *Software* es para comunicar al componente con dispositivos externos que le permitan funcionar completamente. Finalmente, estos componentes requieren adaptar la información y servicios, que presentan al usuario, de acuerdo a su perfil, preferencia y necesidades.

Existen varias definiciones de Agente de *Software*, entre las que están: “*Una entidad de software que funciona continuamente y autónomamente en un ambiente en particular, comúnmente habitado por otros agentes y procesos*” (Shoham, 1997).

“*Un agente es una entidad de software que actúa y se comporta dependiendo de una tarea en particular que haya sido delegada a el. Para hacer esto, un agente puede inferir las preferencias de los usuarios y/o sus necesidades tomando en cuenta sus peculiaridades y la situación en la que se encuentra cada usuario*” (Bradshaw, 1997).

En otra opinión se dice que agente, “*Es un sistema computacional al cual se le añaden las propiedades de conocimiento, creencias, deseos, intenciones u obligaciones*” (Wooldridge, 1995).

Sin embargo, una de las definiciones más utilizada es aquella que define a un Agente de Software por los atributos que pueda poseer (*Wooldridge, 1995*):

- **Autonomía.** Los agentes encapsulan algún estado que no es accesible para otros agentes, y toman decisiones acerca de lo que esté ocurriendo en dicho estado sin la intervención directa de humanos u otros agentes.
- **Reactividad.** Los agentes están situados en un ambiente (el cual puede ser el mundo real, el Internet, avatares o combinados), el cual es percibido (mediante sensores) y son capaces de responder a tiempo según los cambios que ocurran en el ambiente sentido.
- **Pro-actividad.** Los agentes no son actúan de una forma simple ni responden a su ambiente de la misma manera, ellos son capaces cumplir con su objetivo tomando la iniciativa por si solos.
- **Habilidad Social.** Los agentes interactúan con otros agentes (y posiblemente humanos) mediante un lenguaje de comunicación entre agentes (*Genesereth, 1994*) y típicamente tienen la habilidad de comprometerse con otras actividades sociales (con otros agentes) para cumplir su objetivo para el que fue creado.

Con base a esta última definición y considerando que un sistema de Inteligencia Ambiental está caracterizado por ser autónomo, reactivo, distribuido, colaborativo y adaptativo a los artefactos y al contexto que es sentido, se propuso desarrollar el Componente como un sistema multi-agentes. En esta tesis, se parte de la premisa de que los agentes son un enfoque apropiado para la construcción de AmI, ya que permite abordar algunos de los retos para desarrollar estos sistemas (*Rodríguez, 2005*). Uno de estos retos, es proporcionar servicios conscientes del contexto que facilite inferir riesgos en las actividades del Adulto Mayor en el hogar.

Los AmI tienen la habilidad de aprender y adaptarse con el tiempo a los caracteres de las personas y a sus perfiles, así que siempre están en constante crecimiento y aprendizaje a la par con los humanos o usuarios del sistema.

III.3 Discusión

Se propone un componente de *software* que encapsule las funciones de representación del contexto del adulto mayor de sus actividades diarias. Con ayuda de esta información los mecanismos de inferencia responderán a dichos eventos y generarán un resultado, el cual será una notificación para el adulto mayor. Además, dicho componente sea capaz de adaptarse al comportamiento de cada usuario, ya que cada uno de ellos tiene un diferente perfil. Finalmente, se pretende que el componente facilite al desarrollador de sistemas AmI el poder especializar y extender la funcionalidad hasta donde el crea conveniente. La arquitectura de este componente se presenta de una manera detallada en el capítulo siguiente con el fin de facilitar el entendimiento para su futura implementación.

Capítulo IV

Diseño del Componente ELDeR

En este capítulo se presenta la arquitectura y el diseño del componente desarrollado, el cual lleva por nombre *Componente ELDeR* (de las siglas en inglés: *Enabling Living InDependently of Risks*). También se presenta una descripción de los fundamentos teóricos utilizados para el desarrollo de esta aplicación la cual tiene como propósito facilitar el desarrollo de Sistemas AmI para apoyar las Actividades de Vida Diaria del Adulto Mayor. El *Componente ELDeR* captura el contexto adquirido por dispositivos o métodos desarrollados por el programador, lo representa, disemina e infiere el tipo de acción que conviene hacer, dependiendo de la situación enfrentada por el Adulto Mayor. En esta subsección se describirá el funcionamiento del *Componente ELDeR* y de los agentes que lo implementan. El *Componente ELDeR* extiende los servicios del *middleware* SALSA para facilitar el desarrollo de sistemas AmI basados en agentes autónomos. Antes de explicar el diseño e implementación del *Componente ELDeR*, en la siguiente sección se presenta el *middleware* SALSA.

IV.1 Middleware SALSA

Para el desarrollo del trabajo de esta tesis se utilizó SALSA (de las siglas en inglés: *Simple Agent Library for Smart Ambients*), el cual es un *middleware* creado para el manejo de agentes y provee características específicas para facilitar la creación de aplicaciones de cómputo ubicuo basadas en agentes (Rodríguez, 2005). En la Figura 8, se presenta la arquitectura de SALSA.

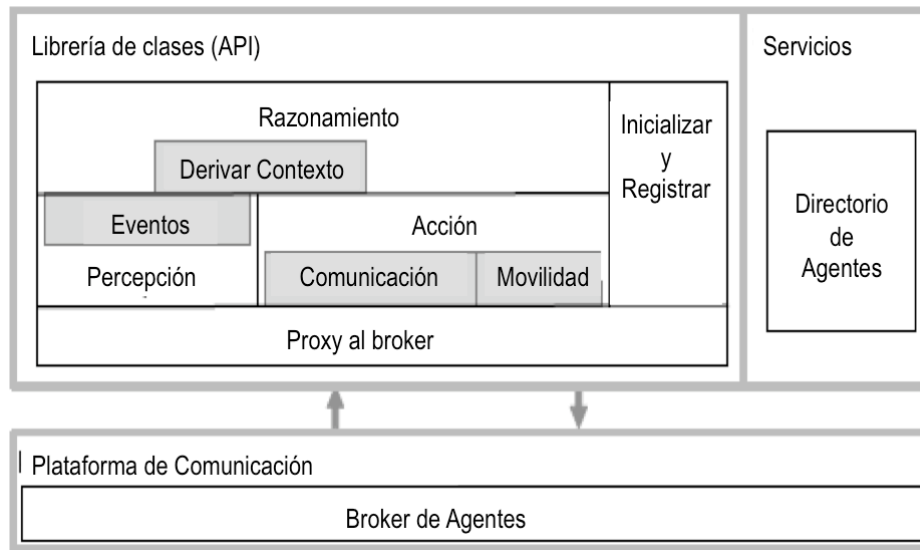


Figura 8. Arquitectura del middleware SALSA (Rodríguez, 2005)

- Plataforma de comunicación.** Un *Broker* de Agentes es el canal de comunicación entre agentes y usuarios. La implementación del *Broker* es un servidor de mensajería instantánea basado en el protocolo *Jabber* (Jabber, 2009), el cual fue extendido para crear el protocolo de comunicación de SALSA. Este protocolo consiste de un lenguaje que permite a los agentes intercambiar diferentes tipos de objetos entre agentes, (tal como información de contexto, solicitudes de servicio) entre agentes y usuarios (tal como eventos generados por la acciones de los usuarios) y entre agentes y servicios. Esta información es enviada o recibida por el agente a través de un proxy al Broker, el cual es parte de los agentes de SALSA. Los agentes son implementados mediante la librería de clases de SALSA (API, del inglés: *Application Programming Interface*).
- API de SALSA.** El elemento principal de SALSA es un conjunto de clases (API) que facilita la implementación de los componentes de los agentes, tal como la percepción, el cual es el *Proxy al Broker*, el razonamiento y la acción. A través del *Proxy al Broker*, el componente de percepción recibe información y genera eventos para comunicar esta información al componente de razonamiento para que sea analizada o procesada y decida qué acción ejecutar. Este componente puede ser fácilmente modificado por el programador para implementar o modificar el

algoritmo de razonamiento que implemente la funcionalidad deseada del agente. Este razonamiento puede consistir de un simple conjunto de reglas o de un algoritmo más complejo, tal como una red neuronal. Como parte de las acciones, el agente puede requerir comunicarse con otros agentes o usuarios, o actualizar su componente de razonamiento adquiriendo el código del algoritmo de razonamiento de otro agente residiendo en un servicio. Finalmente, el API proporciona un conjunto de clases que permiten al agente registrarse en uno o más Directorios de Agentes y solicitar información de otros agentes con lo que requiera interactuar.

- **Servicios.** SALSA provee de un Directorio de Agentes para que los agentes registren sus servicios y realicen búsquedas de otros agentes. Un agente puede buscar información de otro agente con el cual requiera interactuar buscando por el servicio que ofrece. La implementación del Directorio de Agentes consiste de un servidor LDAP (*Lightweight Directory Access Protocol*) y un agente actuando como *proxy* al Directorio de Agentes (*AD-proxy agent*). Las solicitudes de información o de registro de agentes son atendidas por un agente *Proxy* al Directorio de Agentes.

El ciclo de vida de un agente en SALSA incluye los siguientes estados:

- **Activado.** Este es el estado inicial para cuando un agente es creado. Es un super estado que contiene diferentes sub-estados internamente.
- **Percibiendo.** Es un sub-estado inicial en el que el agente adquiere información contextual del ambiente en diferentes formas.
- **Razonando.** El agente evalúa la información percibida para lo que tal vez requiere aplicar un simple o complejo algoritmo de razonamiento para interpretar o transformar esta información en contexto secundario.
- **Ejecutando.** Basado en los resultados del componente de razonamiento, el agente ejecuta una acción, que puede involucrar comunicarse con otros agentes, mueven los agentes a otras plataformas como continuar percibiendo información o terminar un proceso de ejecución.
- **Comunicando.** El agente interactúa con uno o más agentes para proveer la información que sea necesaria para alcanzar los objetivos.

- **Suspendido.** Un agente en este estado está con vida pero no está realizando ninguna actividad. Está esperando a ser reactivado.
- **Desactivado.** Si el agente tiene una meta, primero entra a este estado y después es terminado.

IV.2 Arquitectura del Componente ELDeR

Tal como se presenta en la Figura 9, el *Componente ELDeR* consiste de varios agentes, una ontología y una base de reglas que proveen la siguiente funcionalidad. El *Componente ELDeR* extiende los servicios del *middleware* SALSA para facilitar el desarrollo de sistemas AmI basados en agentes autónomos. El *Componente ELDeR* percibe el contexto del adulto mayor (Agente Consciente del Contexto) al realizar sus AVD y modela el contexto del usuario (Agente proxy-ELDeR) para analizar si requieren asistencia al realizar sus AVD. Para modelar el contexto del adulto mayor se realiza al generar instancias de la *Ontología CARE* (del inglés: *Context Aware Representation*) (García, 2010) la cual contiene conceptos y relaciones que permiten representar las características principales de las actividades, tal como acciones ejecutadas, objetos utilizados, lugar donde se ejecuta. La base de *Reglas CARE* facilita que los desarrolladores incluyan las acciones de asistencia que el sistema AmI debe proporcionar con base al comportamiento esperado del adulto mayor.

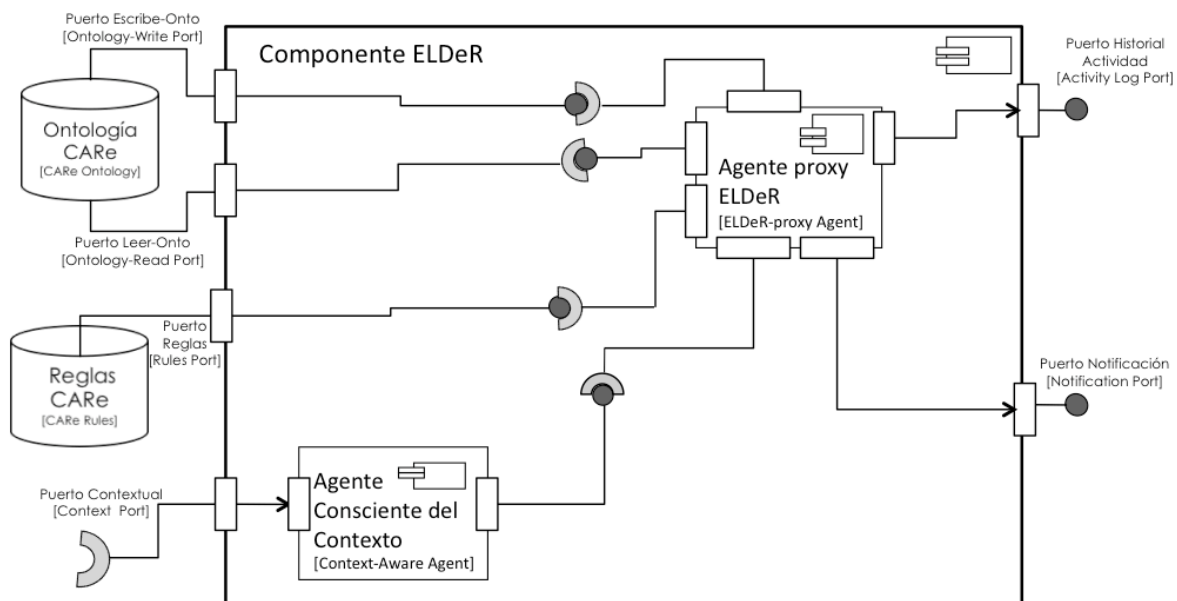


Figura 9. Arquitectura del *Componente ELDeR*

A continuación se presenta una descripción de los puertos y agentes que forman la arquitectura del *Componente ELDeR*.

IV.2.1 Reglas CARE

El *Componente ELDeR* funciona con dos tipos de reglas, las que definen el comportamiento del adulto mayor y del ambiente inteligente.

-Reglas que definen el comportamiento esperado del adulto mayor. Estas reglas activan a los agentes del componente para decidir si una actividad genera un riesgo o necesita algún tipo de asesoría. La decisión se realiza tomando en cuenta lo que está representado en la *Ontología CARE*.

-Reglas que rigen el comportamiento del Ambiente Inteligente. Estos tipos de reglas proveen una representación de como los sistemas esperan que el adulto mayor se comporte de una manera acorde con sus necesidades particulares para realizar sus actividades de vida diaria.

El contenido de la base de *Reglas CARE*, será definido por los desarrolladores para que la actividad que se quiera apoyar, pueda ser soportada por el ambiente inteligente. Se tiene un repositorio para los dos tipos de reglas que permiten a los desarrolladores adaptar fácilmente la aplicación para cualquier persona de la tercera edad.

IV.2.2 Ontología CARE

La ontología es la técnica de representación de contexto que se utiliza en este trabajo, la cual es la encargada de representar las Actividades de Vida Diaria del adulto mayor. La *Ontología CARE* (del inglés: *Context Aware Representation*), se muestra en la Figura 10.

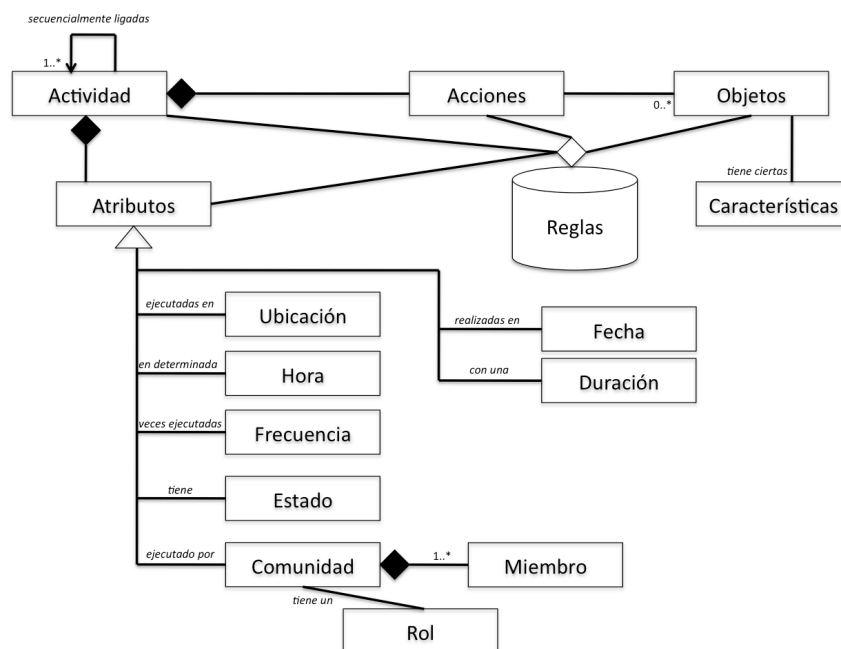


Figura 10. Ontología CARE

El diseño de la *Ontología CARE* describe que una *Actividad* está ligada a otras actividades, por ejemplo, cuando se ingieren alimentos después le sigue lavarse los dientes. Una *Actividad* esta compuesta de *Atributos*, donde los atributos pueden ser:

- la *Ubicación* donde se realiza la actividad (tal como medicarse en la cocina, o en la recámara),
- la *Hora* en que se realiza la actividad (la hora en la que el adulto mayor se medica),
- la *Frecuencia* o número de veces que realiza una actividad durante el día (por ejemplo, comer tres veces al día)
- el *Estado* de la actividad que indica si se está ejecutando, suspendida por el momento o terminada. Por ejemplo, el adulto mayor mientras cocina, podría interrumpir esta actividad (por lo que queda suspendida) para medicarse (actividad ejecutándose).
- la *Comunidad*, indica la(s) persona(s) que intervienen durante la ejecución de la actividad. Por ejemplo, el adulto mayor para adquirir sus medicamentos requiere del apoyo del hijo quien lo lleva a la farmacia, tal como se indica en los resultados del caso de estudio reportado en *García (2010)*. Así, una comunidad está compuesta por *Miembros*, y cada miembro tiene un *Rol* en específico (doctor, familia, adulto mayor, cuidador, etc).

- la *Fecha* en que se realiza la actividad.
- la *Duración* de la actividad ejecutada. Es decir, es el tiempo que transcurre entre el estado de una actividad que inicia el estado de ejecución y cuando se detecta el estado de terminada.

A su vez, una *Actividad* está compuesta por *Acciones* (tal como, la actividad cocinar un sándwich podría involucrar varias acciones como tomar un cuchillo, tomar el pan u otra acción), estas acciones se realizan utilizando *Objetos* (por ejemplo, el cuchillo, el pan). Estos objetos pueden tener ciertas *Características*, por ejemplo el *Objeto* medicina, su principal característica a registrarse en la ontología es la dosis que es tomada por el adulto mayor.

IV.2.3 Puertos

El componente incluye cinco puertos que actúan como un puente (de entrada o de salida) de datos entre los agentes internos y los agentes conectados por el exterior. Los puertos que incluye son los siguientes:

- **Puerto Contextual (Context Port)**

Este puerto de entrada, recibe información de contexto generada por la interacción del adulto mayor con los objetos que utiliza mientras realiza una actividad de vida diaria, y por su movilidad dentro de su hogar. Para lograr lo anterior, a este puerto se conecta un agente que comunica esta información al componente, ya que tiene los mecanismos de razonamiento adecuados para obtener o determinar dicha información. El diseño de este agente esta fuera del alcance de esta tesis, sin embargo su diseño podría basarse en algoritmos como los que presenta (*García, 2010*); tal como la hora en la que realizó la actividad, lugar en donde realizó la actividad, la duración de la actividad, la frecuencia con la que tomó un objeto, el objeto o herramienta que tomó y otras características adicionales que el desarrollador del sistema AmI debe especificar.

- **Puerto Historial de Actividad (Activity Log Port)**

Este puerto de salida, envía a los agentes externos del *Componente ELDeR* el historial de una determinada actividad que realizó el Adulto Mayor en la fecha solicitada.

- **Puerto Notificación (Notification Port)**

Este puerto de salida, envía a los agentes externos del *Componente ELDeR*, el tipo de notificación o riesgo inferido a partir del contexto primario capturado y de las reglas que rigen el comportamiento del sistema y de las reglas que especifican como se espera se comporte el adulto mayor.

IV.2.4 Agentes del *Componente ELDeR*

El componente cuenta con dos agentes internos que trabajando en conjunto pueden inferir riesgos en las actividades realizadas por el adulto mayor dentro del hogar. A continuación se presenta una descripción de su funcionamiento.

- **Agente Consciente del Contexto (Context-Aware Agent)**

Este agente es el encargado de recibir del Puerto Contextual la información capturada por los agentes externos al *Componente ELDeR* y de enviársela al Agente Proxy en un formato entendible para él.

- **Agente Proxy - ELDeR (ELDeR-proxy Agent)**

Este agente es el encargado de escribir y leer en la *Ontología CARE* y consultar la base de reglas programadas en JESS para inferir el contexto secundario del Adulto Mayor. El contexto inferido lo envía al *Puerto Notificación*, en donde un agente recibe la información y decide como actuar de acuerdo con la lógica del sistema AmI implementado. Adicionalmente, el *Agente Proxy-ELDeR* atiende las solicitudes de otros agentes (externos al componente) para consultar la *Ontología CARE*, es decir, genera historiales de las actividades del adulto mayor, lo cual envía por el *Puerto Historial Actividad*.

El funcionamiento interno del *Componente ELDeR* se ilustra en la Figura 11, donde se muestran los estados por los que pasa el Componente desde que percibe información de contexto hasta que infiere un riesgo o situación anormal enfrentada por el adulto mayor. Como primera etapa, percibe el contexto del adulto mayor, para después actualizarlo en la *Ontología CARE* representando la actividad que esta realizando, con estos datos se evalúan las *Reglas CARE* programadas por el desarrollador y si alguna de ellas se cumple, entonces se notifica al adulto mayor algún tipo de aviso y después continua recibiendo el contexto de

la persona, si no se cumple ninguna regla continua percibiendo el contexto para volver a empezar con el ciclo.

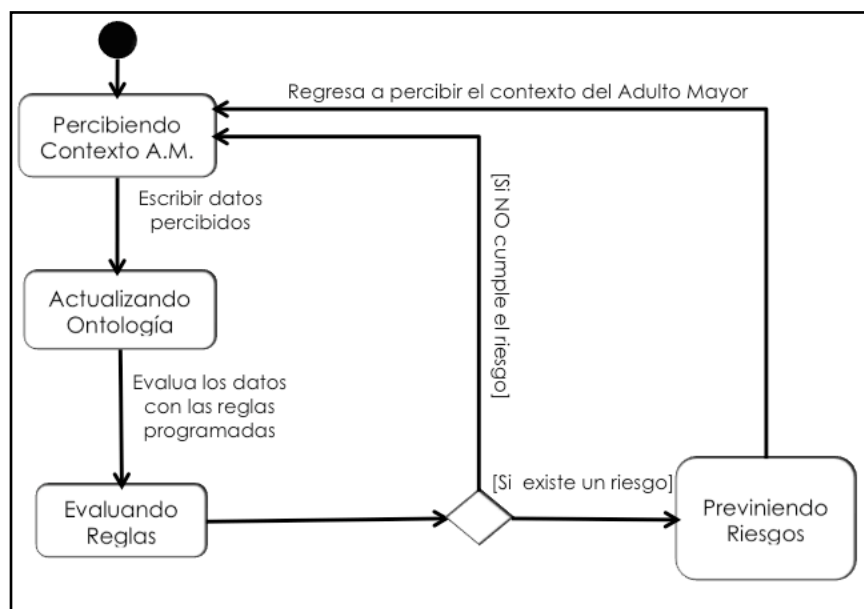


Figura 11. Funcionamiento del Componente ELDeR

IV.3 Implementación del Componente ELDeR

IV.3.1 Implementación de la base de *Reglas CARE* con el lenguaje JESS

En la implementación de la base de *Reglas CARE*, se utiliza el lenguaje de programación llamado JESS (*Java Expert System Shell*) (JESS, 2009). El cual es un motor de reglas para la plataforma JAVA, se extiende del lenguaje CLIPS, desarrollado por *Ernest Friedman-Hill* de *Sandia National Labs* en 1995.

JESS funciona como cualquier sistema experto basado en reglas. Contiene una memoria donde se almacena la base de conocimientos de hechos y reglas (la cual contiene los predicados que indican el funcionamiento del programa), una máquina de inferencia y una motor de ejecución.

IV.3.2 Implementación de la *Ontología CARE* con el lenguaje OWL-DL

OWL es un lenguaje semántico para publicar y compartir ontologías en el *World Wide Web* (OWL, 2009). OWL es desarrollado como un vocabulario que se extiende del lenguaje RDF

(*Resource Description Framework*) (RDF, 2009) y se deriva del lenguaje ontológico DAML+OIL (DAML, 2009).

El lenguaje OWL se divide en tres sublenguajes, OWL Full, OWL DL, OWL Lite. Los primeros dos soportan el mismo conjunto de constructores y la única diferencia entre ellos radica en las restricciones del uso de algunas características. OWL Full permite mezclar OWL con el esquema RDF para permitir una mayor libertad de expresión. Sin embargo, OWL DL posee mecanismos de razonamiento que son compatibles con las herramientas constructoras de ontologías. Por último el lenguaje OWL Lite solo soporta un subconjunto de constructores que se utiliza para aplicaciones básicas.

El lenguaje utilizado para la programación de la *Ontología CARE* es el Lenguaje OWL DL, ya que por sus características que permiten una mayor expresividad para la representación de dominios específicos y sus mecanismos de razonamiento son compatibles con la herramienta de programación, *Protégé* (Protégé, 2009).

IV.3.3 Agentes del Componente ELDeR programados con lenguaje JAVA utilizando el *middleware* SALSA

Para la programación de los agentes internos y externos del *Componente ELDeR*, se utiliza el lenguaje JAVA el cual es el lenguaje para programar los agentes SALSA.

El *middleware* SALSA facilita desarrollar agentes autónomos como las entidades de software principales de los sistemas de inteligencia ambiental (Rodríguez, 2008).

IV.4 Discusión

En el Capítulo IV se presentó la arquitectura de la propuesta tecnológica, al cual le llamaremos *Componente ELDeR* que fue diseñado para facilitar el desarrollo de sistemas de inteligencia ambiental para apoyar las actividades de vida diaria del adulto mayor. El componente esta formado por tres principales conceptos: Agentes de *Software*, Ontología y una Base de Reglas.

Los agentes de software se utilizan para comunicar al componente las actividades que realiza el adulto mayor en su hogar. También informan al adulto mayor las notificaciones de riesgo en la actividad que realiza, las cuales se infirieron en el componente. La ontología se utiliza para la representación del contexto del adulto mayor, por ejemplo las acciones que realizó, su ubicación, la hora en que sucedieron, etc. La base de reglas fue diseñada para plasmar el comportamiento ideal del usuario y de las acciones que rigen el comportamiento del sistema AmI. En el capítulo siguiente, se presenta la implementación de un sistema de inteligencia ambiental utilizando el *Componente ELDeR* y explicando detalladamente el funcionamiento de cada concepto expresado en el diseño.

Capítulo V

Uso del Componente ELDeR para implementar aplicaciones de Inteligencia Ambiental

Para ilustrar como el *Componente ELDeR* facilita implementar un sistema AmI para apoyar la vida independiente del adulto mayor, se retoma uno de los escenarios presentados en el Capítulo III sección 1.3.1 y se explica como se implementa el sistema ilustrado en tal escenario.

V.1 Implementación del Sistema de Ambiente Inteligente CARE-Me

En este caso es el Sistema *CARE-Me* (del inglés: *Context Aware Reminder of the Medication*) que se reporta en (García, 2010), el cual se explica nuevamente en el siguiente escenario:

“Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que su prescripción médica indica que en la noche debe de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida. La Sra. Ana se dirige a su cuarto porque ya se terminó el día. El sistema esta consciente de que Ana cumplió correctamente con la medicación del día de hoy, por lo que el perico que representa la semana actual aumenta de tamaño. Adicionalmente, como la semana ha terminado, aparece un nuevo perico para representar la semana que empieza”.

V.1.1 Arquitectura del Sistema CARE-Me

Para implementar el sistema CARE-Me, se diseñaron los agentes que se comunicarán con el Componente ELDeR para proveerle del contexto del adulto mayor y los que permiten asistir al adulto mayor para que realice su actividad de vida diaria libre de riesgos. En la Figura 12, se muestra que el sistema CARE-Me incluye al *Componente ELDeR* y los siguientes agentes SALSA que proven o reciben información del componente:

- **Agente Adulto Mayor.** Este agente envía el contexto del adulto mayor al componente. El Agente Adulto Mayor podría obtener esta información de contexto a través de diferentes sensores y/o otros agentes actuando como proxies a sensores. Cabe resaltar, que la funcionalidad de cómo se estima el contexto del adulto mayor (p. ej. Actividad realizada, objetos utilizados, localización, etc), no es objetivo de esta tesis.
- **Agente Riesgo.** Este agente recibe desde el *Componente ELDeR* un mensaje SALSA notificando el tipo de riesgo que enfrenta el adulto mayor. Con base a este mensaje, el agente decide las acciones a ejecutar para asistir o ayudar al adulto mayor a realizar su actividad de vida diaria libre de riesgos. El *Componente* determina cuando comunicarse con el agente de riesgo cuando se cumple una condición de la Base de *Reglas CARE* indicando que el adulto mayor requiere asistencia. Por consiguiente, se le envía la información contextual al Agente Contextual y este se encarga de ordenar el mensaje para enviarsela al Agente Proxy, el cual escribe en la *Ontología CARE* los datos del contexto y evalúa las *Reglas CARE*.
- **Agente proxy-Visualizador Ambiental.** Este agente es el que provee de la interfaz de usuario para asistir al adulto mayor con su medicación. El tipo de interfaz de usuario que provee, se conoce como visualizador ambiental. En este caso, esta interfaz no solo presenta recordatorios para medicarse, sino que provee una representación del cumplimiento de la medicación del adulto mayor. Para lograr lo anterior, este agente recibe un mensaje SALSA del Agente Riesgo, que contiene información sobre el cumplimiento de la medicación del adulto mayor, lo cual se refleja en el visualizador ambiental *CARE-Me* tal como se menciona en el escenario anterior.

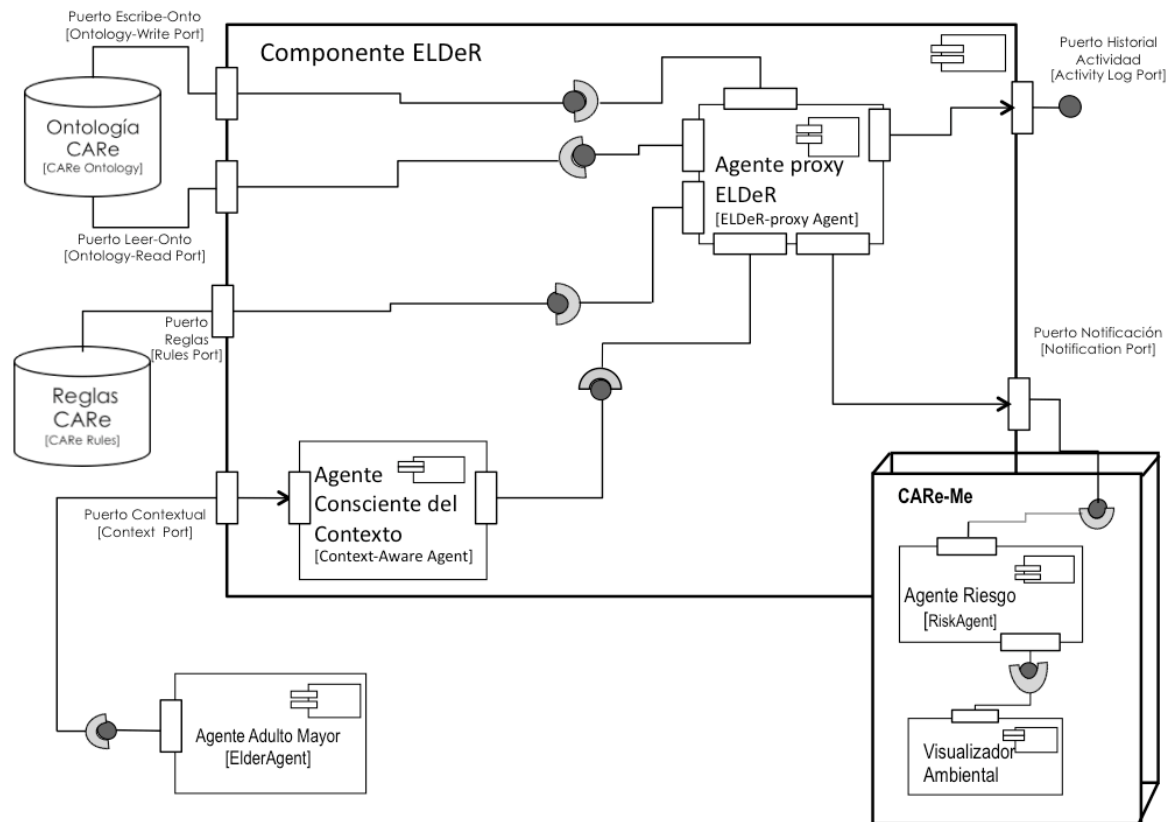


Figura 12. Arquitectura del Sistema CARE-Me

Para explicar la comunicación de los agentes anteriores con el *Componente ELDeR*, se describe a continuación el diagrama de secuencias en la Figura 13. Este muestra que el *Agente Adulto Mayor*, mediante el método de SALSA *sendNotificationInfo()*, le envía la información contextual y las rutas de los archivos al *Agente Consciente del Contexto*, este a su vez mediante el mismo método de *sendNotificationInfo()* le envía el contexto recibido y las rutas de los archivos en un formato adecuado al *Agente Proxy-ELDeR* actualice la *Ontología CARE* con el método *updateOntology()* y evalúa las *Reglas CARE* con el método *verifyRules()*. En este caso, como se cumple una de las reglas de la Base de *Reglas CARE*, entonces el *Agente Proxy-ELDeR* notifica (mediante el método *sendNotificationInfo()*) al *Agente Riesgo*, que se debe recordar al adulto mayor medicarse. El *Agente Riesgo* crea una instancia de la clase *Recordar*, la cual contiene la funcionalidad para mostrar a los pericos cantando para recordar al adulto mayor que debe de medicarse (para una mayor descripción de los métodos y clases utilizadas, véase Apéndice H).

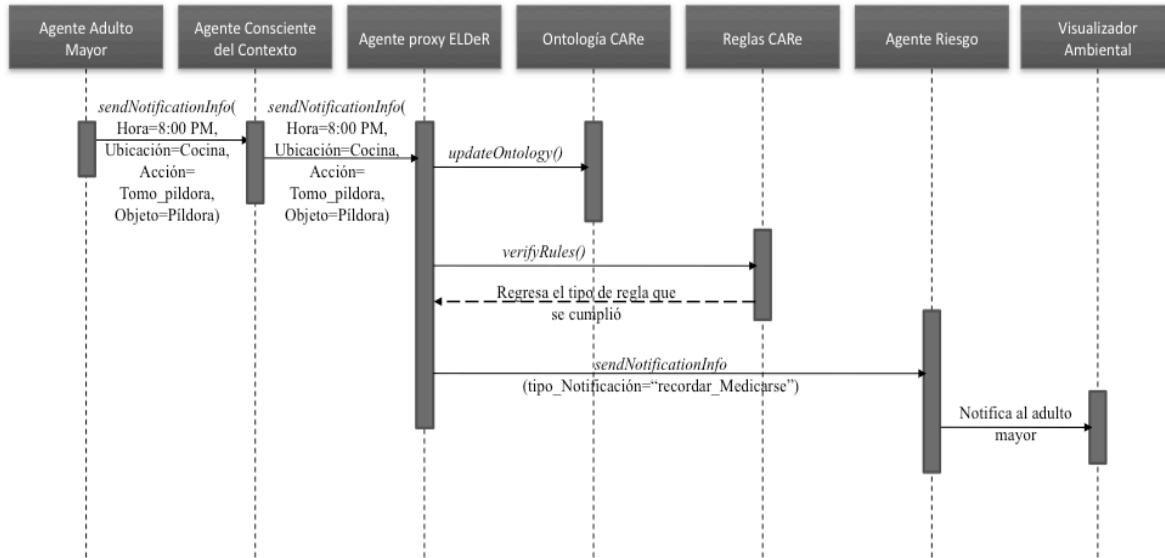


Figura 13. Funcionamiento del Sistema CARE-Me desarrollado con el Componente ELDeR

V.1.2 Ontología del Sistema *CARE-Me*

Para explicar como la *Ontología CARE-Me* permite representar el contexto del adulto mayor mientras se medica, en la Figura 14 se muestra una instancia de la ontología en que la Señora Ana toma sus pastillas. Teniendo como *Actividad* la de Medicarse, los Atributos que se presentan en dicha actividad son la *Fecha* (Domingo), *Ubicación* (Cocina) y la *Hora* (Noche). Al medicarse se presentan diferentes *Acciones*, como tomar el vaso y las píldoras, que son los *Objetos* o *Herramientas* que se utiliza para completar la actividad. Por último esos *Objetos* tienen ciertas *Características*, que en el caso de la medicación son las dosis de Glibenclamida, Pravastatina y Enalapril.

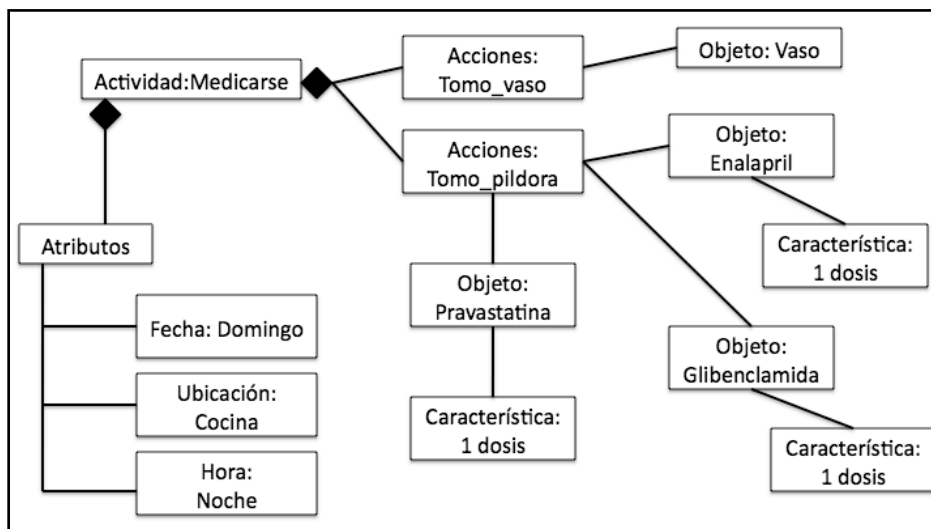


Figura 14. Instancia de la Ontología CARE para representar contexto durante la medicación del adulto mayor

En la Figura 15, se muestra la representación ontológica del momento en el que la Sra. Ana se dirige a su recámara para dormir. En este caso la *Actividad* es dirigirse a su habitación para después descansar. Los *Atributos* que se identifican es la de la *Fecha* (Domingo), *Ubicación* (Recamara) y *Hora* (Noche). Las *Acciones* detectadas durante la actividad es la de caminar.

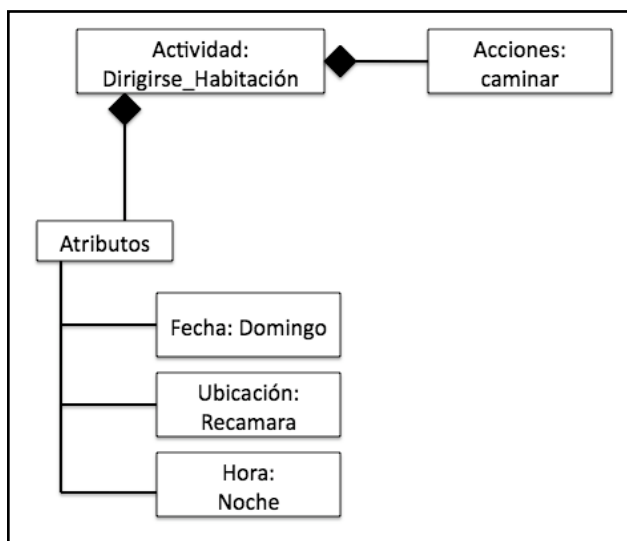


Figura 15. Instancia de la Ontología CARE cuando el adulto mayor se retira a dormir

V.1.3 Base de Reglas del Sistema *CARe-Me*

Las reglas identificadas para implementar el sistema *CARe-Me* se presentan en la Tabla 1, la cual describe las reglas que definen el comportamiento del adulto mayor, y la Tabla 2, la cual presenta las reglas que rigen el comportamiento del sistema *CARe-Me*. En la primera columna de cada tabla se muestra el diseño de las reglas y la segunda columna muestra las reglas programadas en lenguaje JESS.

Reglas que definen el comportamiento del adulto mayor identificadas en el escenario	Reglas que definen el comportamiento del adulto mayor programadas en JESS
Si hora == noche, entonces hora = hora de medicarse, pastilla = pravastatina, dosis =1	<pre>(defrule prescripcionNochePravastatina (Plantilla {Hora == noche}) => (assert (Ubicacion cocina) (Objetos Pravastatina) (Caracteristicas 1)) (bind ?Frecuencia 3))</pre>
Si hora == noche, entonces hora = hora de medicarse, pastilla = glibenclamida, dosis =1	<pre>(defrule prescripcionNoche Glibenclamida(Plantilla {Hora==noche}) => (assert (Ubicacion cocina) (Objetos Glibenclamida) (Caracteristicas 1)) (bind ?Frecuencia 3))</pre>
Si hora == noche, entonces hora = hora de medicarse, pastilla = enalapril, dosis =1	<pre>(defrule prescripcionNocheEnalapril(Plantilla {Hora == noche}) => (assert (Ubicacion cocina) (Objetos Enalapril) (Caracteristicas 1)) (bind ?Frecuencia 3))</pre>

Tabla 1. Reglas que definen el comportamiento del Adulto Mayor

Reglas que rigen el comportamiento del ambiente inteligente identificadas en el escenario	Reglas que rigen el comportamiento del ambiente inteligente programadas en JESS
Si hora == hora _medicacion entonces recordarMedicarse	((defrule recordatorioCARE-Me(Plantilla {horaMedicacion == mañana horaMedicacion == tarde horaMedicacion == noche}) => (assert (agenteRiesgo riskagent@localhost) (accion recordar)))
Si cumplió con medicación diaria, entonces perico crece	((defrule pericoCrece (Plantilla (Frecuencia ?Frecuencia)) => (assert (agenteRiesgo riskagent@localhost) (accion crecerPerico)))
Si es el ultimo día de medicación, entonces perico que representa la semana actual vuela al tronco y nace uno nuevo.	((defrule obtenerFecha (Plantilla { Fecha==06-06-2010}) => ((bind ?fecha 06-06-2010)) (defrule vuelaPerico (Plantilla (Fecha ?fecha)) => (assert (agenteRiesgo riskagent@localhost) (accion volar)))

Tabla 2. Reglas que rigen el comportamiento del ambiente inteligente

V.2 Discusión

Para ilustrar el uso y comportamiento del *Componente ELDeR* al desarrollar y ejecutar sistemas Aml, se explicó mediante la implementación del sistema CARE-Me. El cual tiene como propósito recordar al adulto mayor cuando es la hora de su medicación y le muestra una representación del cumplimiento de ésta para motivarlo a continuar la actividad de un modo correcto en el futuro. Para implementar la funcionalidad anterior, el *Componente ELDeR* se encarga de recibir la información contextual, representarla en la *Ontología CARE* y diseminarla a los agentes que requieren procesarla para inferir un posible riesgo con base al contenido de la Base de *Reglas CARE*. Si existe un riesgo, entonces se envía una notificación a los agentes del visualizador ambiental para que éste le recuerde que es hora de medicarse.

Con el propósito de obtener evidencia empírica sobre la facilidad de uso del componente, se realizó una evaluación en la que participaron estudiantes de la Facultad de Ingeniería, la cual se presenta en el Capítulo siguiente.

Capítulo VI

Evaluación del Componente ELDeR

Para conocer la facilidad de uso del *Componente ELDeR* al implementar sistemas AmI, se realizó una evaluación para obtener la percepción de facilidad de uso y utilidad de ocho (8) personas (estudiantes y profesores) de la Facultad de Ingeniería cursando la materia de Cómputo Móvil y Ubicuo. El protocolo de la evaluación consistió en presentaciones de los conceptos teóricos y prácticos necesarios para utilizar el Componente ELDeR, y de ejercicios de diseño e implementación de un Sistema AmI. Se aplicaron dos exámenes escritos, uno de opción múltiple y otro de respuesta abierta, finalizando con una encuesta basada en el instrumento SUS (*System Usability Scale*), el cual permite medir la percepción de facilidad de uso y utilidad de un sistema. La evaluación consistió de tres sesiones de trabajo, durante las cuales se registraron los avances y el desarrollo de cada persona con relación a su conocimiento previo y el que se adquirió después de las sesiones de trabajo.

VI.1 Protocolo de Evaluación

El objetivo general de la evaluación fue:

“Identificar si el Componente ELDeR, propuesto en esta investigación, es fácil de usar y de comprender, para desarrollar aplicaciones de Inteligencia Ambiental”.

De este objetivo se generaron las siguientes hipótesis.

Hipótesis 1:

“El Componente ELDeR es útil y fácil de usar para el diseño e implementación de la funcionalidad consciente del contexto de los sistemas AmI”.

Hipótesis 2:

“La ontología CARE es fácil de extender para representar Actividades de la Vida Diaria”.

Hipótesis 3:

“La base de reglas CARE facilitan el diseño y la implementación del comportamiento del sistema AmI con base al comportamiento esperado del adulto mayor”.

VI.1.1 Perfil del participante

Durante la evaluación participaron ocho estudiantes (cinco estudiantes de licenciatura en Ingeniería en Computación, dos estudiantes de Maestría y un Maestro en Ingeniería recién egresado del posgrado), los cuales tenían el siguiente perfil, requerido para participar en la evaluación:

- Haber cursado o estar cursando la materia de cómputo ubicuo.
- Manejo básico del lenguaje de programación JAVA.
- Habilidades para la programación orientada a objetos.

VI.1.2 Descripción de las etapas de la evaluación

El proceso seguido para realizar la evaluación consistió de tres etapas, las cuales se describen en la Tabla 3. La primera etapa tuvo como propósito homogenizar los conocimientos de los participantes conocimientos acerca del Cómputo Ubicuo, Sistemas Conscientes del Contexto, Sistemas Inteligentes e Interacción Humano-Computadora. Esta etapa finalizó con un cuestionario de conocimientos teóricos. La segunda etapa consistió de una presentación del diseño de la arquitectura del componente, donde se explicó como está formado y las partes que los desarrolladores deben implementar. Similarmente, esta etapa también finalizó con un cuestionario para evaluar la comprensión de los estudiantes en cuanto a diseñar el comportamiento de un sistema AmI utilizando el *Componente ELDeR* y agentes. La última de las etapas consistió en un ejercicio de implementación en el que, el participante debía completar el código del agente que provee el contexto al *Componente ELDeR* y del agente encargado de recibir la notificación del riesgo inferido por el componente.

AGENDA	
Tiempo Requerido	Descripción de las etapas de la evaluación
	Etapa 1. Homogenización de conocimientos (31 de Mayo, 2010)
35 minutos	Presentación introductoria al cómputo ubicuo, sistemas conscientes del contexto, técnicas de representación del contexto y agentes de <i>software</i>
30 minutos	Aplicación de un cuestionario
	Etapa 2. Evaluación del componente para diseñar sistemas AmI (2 de Junio, 2010)
30 minutos	Presentación introductoria a la arquitectura del componente ELDeR
60 minutos	Práctica de diseño de la arquitectura del componente mediante un escenario de uso propuesto
	Etapa 3. Evaluación del componente para implementar sistemas AmI (9 de Junio, 2010)
120 minutos	Experimento de programación

Tabla 3. Agenda del proceso de evaluación

A continuación se describen detalladamente las tres etapas de evaluación.

VI.1.2.1 Descripción de la Etapa 1. Homogenización de conocimientos

Consistió en una exposición ante el grupo, en el que se les explicó que es el Cómputo Ubicuo y los Sistemas AmI donde se explicaron conceptos tales como (Ver Apéndice D):

- a. Definición de Ambiente Inteligente.** Es la convergencia de cuatro áreas de la computación; Cómputo Ubicuo, Sistemas Conscientes del Contexto, Sistemas Inteligentes e Interacción Humano-Computadora. Siendo los Sistemas Conscientes del Contexto el área en el que se enfoca el trabajo de investigación.
- b. Definición de Sistemas Conscientes del Contexto.** Un sistema es consciente de contexto si se utiliza el contexto para proveer información relevante y/o servicios para el usuario, donde la relevancia depende de la tarea del usuario.
- c. Definición de Contexto y Tipos de Contexto.** El contexto es cualquier información que puede ser usada para caracterizar la situación de la entidad. Una entidad es una persona, lugar u objeto que se considera relevante para mejorar la

interacción entre un usuario y una aplicación. Existen dos tipos de contexto: Contexto Primario y Contexto Secundario.

- d. Técnicas de representación del contexto.** Existen muchas técnicas de representación del contexto, en esta ocasión se tomarán como base la lógica de predicados que es una técnica que permite inferir con reglas determinadas por el programador lo que hace cumplir esa regla correctamente. Algunos ejemplos de lenguajes de programación que utilizan esta técnica son PROLOG, LISP y JESS. Otra técnica que se aplica para la representación del contexto es el lenguaje XML que por sus siglas en inglés significa *eXtensible Markup Language*, el cual permite crear al programador sus propias etiquetas para establecer la definición, transmisión, validación e interpretación de datos entre aplicaciones y organizaciones. Un ejemplo de la sintaxis (orden que sigue el lenguaje XML para tener un significado) de este lenguaje es el siguiente, donde las etiquetas están delimitadas por los signos de corchetes angulares (<>) y en medio de ellas está la información y/o aplicación del contexto.

```
<Actividad>Tomar_ducha</Actividad>
<Ubicación>baño</Ubicación>
```

La tercera técnica utilizada es la Ontología que es la rectora en el presente trabajo de investigación para la representación del contexto del Adulto Mayor. La ontología es una conceptualización del dominio específico del contexto que se desea representar de la manera más detallada posible en una interacción de significados. Un ejemplo de una ontología donde el dominio específico es la actividad que realiza el adulto mayor y sus relaciones como la ubicación, herramientas que se utilizan para realizar la actividad, características relacionadas a la herramienta utilizada y el posible riesgo al que se expone la persona por diversas causas.

- e. Definición de middleware SALSA.** *Middleware* es lo que conecta, lo que une dos aplicaciones distintas o distribuidas para formar una herramienta completa. *Middleware* es una aplicación que conecta componentes de software para lograr una completa comunicación entre ellos. El middleware que se utiliza en este trabajo es el llamado SALSA (por sus siglas en inglés *Simple Agent Library for Smart Ambients*), el cual se selecciona debido a que facilita la implementación y evolución

progresiva de Sistemas de Cómputo Ubicuo mediante el uso de agentes. SALSA se utiliza como herramienta para programar el *Componente ELDeR* porque proporciona una plataforma de comunicación entre agentes, que tienen las características de autonomía, colaboración, reactividad, comunicación y que tienen como ciclo de vida la de percibir información, razonar con base a esta y actuar.

Después de la presentación de los subtemas anteriores, se aplicó un cuestionario para evaluar si los participantes comprendieron exitosamente la teoría. El cuestionario tuvo un total de 10 preguntas de opción múltiple con un valor cada una de un punto. Se dividió en tres partes, el primer subtema: una introducción a los AmI, seguidamente de las técnicas de representación del contexto y finalmente fue una plática acerca de la definición de agentes de software (Ver Apéndice A).

VI.1.2.2 Descripción de la Etapa 2. Evaluación del componente para diseñar sistemas AmI

La segunda etapa consistió de una exposición (la cual se presenta en el Apéndice E) acerca de la arquitectura del *Componente ELDeR*, es decir se explicó como está organizado y su funcionamiento.

El ejercicio de diseño que los participantes realizaron después de la exposición se presenta en el Apéndice B. Este ejercicio, consistió en darles a los participantes un escenario de uso de un sistema AmI que apoya la actividad de medicación de un adulto mayor. El sistema que ilustra este escenario es el sistema *CARe-Me (Context Aware Reminder Medication)*, el cual se presentó en el Capítulo III.

Con base en el escenario anterior, se le solicitó a los participantes que utilizaran la *Ontología CARe* para representar el contexto del adulto mayor y que identificaran los dos tipos de reglas que permiten al *Componente ELDeR* inferir riesgos enfrentados por el adulto mayor. Para realizar lo anterior, se le pidió a los participantes que realizaran lo siguiente:

El primer ejercicio consistió en identificar el contexto del adulto mayor subrayando en el texto con pluma negra, y posteriormente representarlo en la *Ontología CARe*. La Figura 16, presenta la solución a este ejercicio de diseño, tal como se esperaba que se realizara. Este ejercicio tuvo un valor de 10 puntos. El criterio para evaluar si se realizó

correctamente, fue que por cada error de omisión de un concepto en la *Ontología CARE* se sancionaba quitando medio punto, y si se omitía una relación se quitaba un cuarto de punto.

Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción medica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.

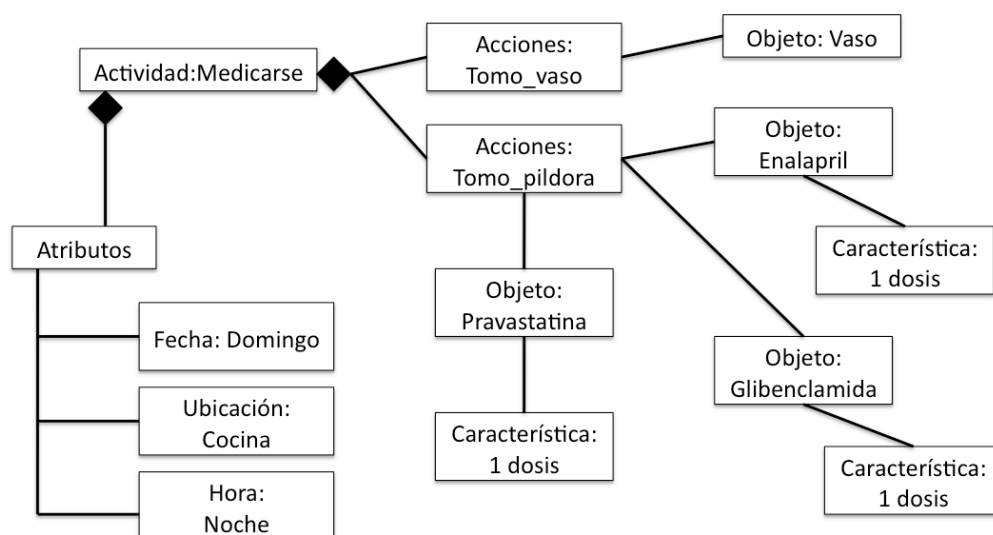


Figura 16. Solución del Ejercicio 1

El segundo ejercicio solicitaba a los estudiantes que identificaran los dos tipos de reglas que debía contener la Base de Reglas CARE. Este tenía un valor de 10 puntos y el criterio para evaluarlo fue el siguiente:

- Formato correcto de la expresión de regla, es decir que tenga las expresiones “Si” y “entonces” (1 punto).
- Primera parte de la regla, la cual debe contener la variable tiempo en ella (3 puntos).
- Segunda parte de la regla, la cual debe expresar el tipo de medicamento que el adulto mayor debe tomarse (6 puntos).

Si alguna de estas particularidades no se cumplía se sancionaba quitando los respectivos puntos de cada una. Para identificar las reglas, se le solicitó al participante que subrayará con pluma azul la información necesaria para obtener las reglas que definen el comportamiento esperado del adulto mayor. La Figura 17 presenta como se esperaba que este ejercicio se resolviera.

Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción médica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.

Las reglas que definen el comportamiento del adulto mayor, se expresan en las siguientes frases:

Si hora == noche, entonces hora = hora de medicarse, pastilla = pravastatina, dosis =1

Si hora == noche, entonces hora = hora de medicarse, pastilla = glibenclamida, dosis =1

Si hora == noche, entonces hora = hora de medicarse, pastilla = enalapril, dosis =1

Figura 17. Solución del Ejercicio 2

El tercer ejercicio que el estudiante realizó tuvo un valor de 10 puntos y se evaluó de la siguiente manera:

- Formato de expresión de regla, es decir que tenga las expresiones “Si” y “entonces” (1 punto).
- Primera parte de la regla, la cual debe contener la variable de la hora de medicación (4.5 puntos).
- Segunda parte de la regla, la cual debe expresar la acción que debe notificar el sistema que en este caso es “recordar” (4.5 puntos).

El ejercicio consistió en subrayar con pluma roja la información necesaria para obtener las reglas que definen el comportamiento del sistema. El resultado que se esperaba obtener de este ejercicio, se presenta en la Figura 18.

Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción medica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.

La regla que define el comportamiento del Ambiente Inteligente se expresa en la siguiente frase:

Si hora == hora _medicacion entonces recordarMedicarse

Figura 18. Solución del Ejercicio 3

El cuarto ejercicio con valor de 10 puntos, consistió en identificar el contexto del adulto mayor de la segunda parte del escenario de uso del sistema CARE-Me presentado en el Capítulo III. Los participantes debían subrayar el contexto del adulto mayor en el texto del segundo escenario con pluma negra, para después representarlo en la *Ontología CARE*, si se omitía un concepto en la ontología se restaba medio punto y si se omitía una relación se restaba un cuarto de punto. En la Figura 19 se presenta tal como los participantes debían identificar el contexto en el escenario.

La Sra Ana se dirige a su cuarto porque ya se terminó el día. El sistema esta consciente de que Ana cumplió correctamente con la medicación del día de hoy, por lo que el perico que representa la semana actual aumenta de tamaño. Adicionalmente, como la semana ha terminado, aparece un nuevo perico para representar la semana que empieza.

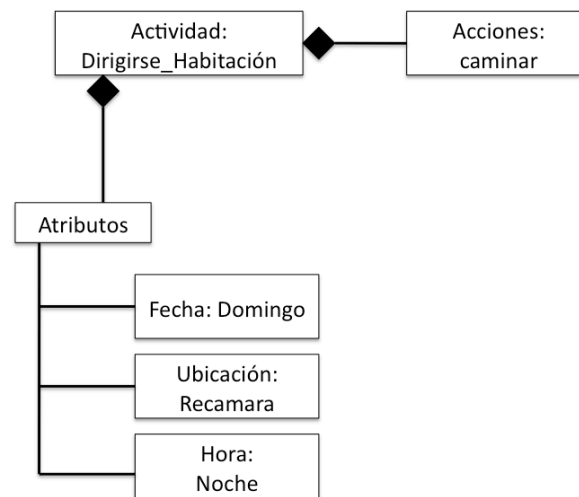


Figura 19. Solución del Ejercicio 4

El quinto ejercicio con valor de 10 puntos, consistió en identificar la información que define el comportamiento del sistema, y subrayar esta información con pluma roja. Posteriormente, el participante debía escribir la regla. Este ejercicio se evaluó con base al siguiente criterio:

- Formato de expresión de regla, es decir que tenga las expresiones “Si” y “entonces” (1 punto).
- Primera parte de la regla, la cual debe expresar si la medicación del adulto mayor fue la correcta y si es el ultimo día de la semana (4.5 puntos).
- Segunda parte de la regla, la cual debe contener la acción del sistema que en este caso es que el perico del cuadro crezca un nivel y que vuele al arbol para que nazca uno nuevo (4.5 puntos).

La Figura 20 muestra la solución a este ejercicio.

La Sra Ana se dirige a su cuarto porque ya se terminó el día. El sistema esta consciente de que Ana cumplió correctamente con la medicación del día de hoy, por lo que el perico que representa la semana actual aumenta de tamaño. Adicionalmente, como la semana ha terminado, aparece un nuevo perico para representar la semana que empieza.

Las reglas que definen el comportamiento del Ambiente Inteligente se expresa en la siguiente frase:

Si cumplió con medicación diaria, entonces perico crece.

Si es el ultimo día de medicación, entonces perico que representa la semana actual vuela al tronco y nace uno nuevo.

Figura 20. Solución del Ejercicio 5

VI.1.2.3 Descripción de la Etapa 3. Evaluación del componente para implementar sistemas AmI

En esta tercera etapa y última etapa de la evaluación, se empezó con una presentación que explicaba a los participantes la solución del examen de diseño (Ver Apéndice F), esto con el propósito de aclarar el enfoque para implementar aplicaciones con el *Componente ELDeR*. Como paso seguido, se explicó como utilizar el *Componente ELDeR* para desarrollar un sistema AmI. Es decir, el ejercicio de implementación consistió de una explicación de cómo implementar la aplicación AmI, y posteriormente los participantes realizaron una práctica de implementación guiada por el instructor, por lo que los estudiantes tenían que seguir las instrucciones para implementar el sistema. El Apéndice G presenta la informaciónn presentada a los estudiantes durante esta sesión, la cual tuvo la siguiente organización.

Se condujo una evaluación informal y guiada para la práctica de implementación. A través de un ejercicio de programación, el cual consistió en programar los agentes *Elder* y *Risk* para el simular el funcionamiento completo del *Componente ELDeR*. Se tuvo que programar el agente que simula el contexto recibido por sensores que detectan las

actividades del adulto mayor en el ambiente (*ElderAgent*) y el agente que recibe el tipo de notificación que se infiere dependiendo de lo que se decida en las reglas programadas en JESS (*RiskAgent*). En la Tabla 4 se describe los tres ejercicios que realizaron los participantes para programar los agentes *Elder* y *Risk*.

En el Ejercicio 1, el participante tuvo que escribir la simulación de las acciones que estaba haciendo el adulto mayor para poder activar la regla que le recuerda que es su hora de medicación. En la segunda columna de la Tabla 4 aparecen los valores que el desarrollador agregó al agente.

En el Ejercicio 2, el participante tuvo que escribir en el agente *Risk*, la llamada al método que permite la acción de recordar la medicación al adulto mayor, esto se hace mediante la clase *PericoCanta*.

En el Ejercicio 3, el participante escribió parte del razonamiento del agente *Risk* donde se definieron el nombre de las clases de acciones que puede realizar el agente según la regla que se haya cumplido.

Ejercicio 1. Agente Elder	
Plantilla de la Acción del agente Elder	Plantilla llenada por los participantes
<pre> public class ElderAction extends Action{ public Object execute() { ContextualInformation ci = new ContextualInformation(); ci.setActivity(""); ci.setTime(""); ci.setCommunity(""); ci.setDate(""); ci.setDuration(""); ci.setFeatures(""); ci.setFrequency(""); ci.setMembers(""); ci.setRole(""); ci.setStatus(""); ci.setLocation(""); ci.setActions(""); ci.setTypeofADL(""); ci.setTools(""); ci.setOntologyPath(""); ci.setParticularRulesPath(""); ci.setGeneralRulesPath(""); this.sendNotificationInfo("contextawareagent@localhost", ci.xmlMessageFactory(), " ", ci.requestPaths()); return null; } </pre>	<pre> public class ElderAction extends Action{ public Object execute() { ContextualInformation ci = new ContextualInformation(); ci.setActivity("Leyendo"); ci.setTime("noon"); ci.setCommunity("Ninguna"); ci.setDate("09-06-2010"); ci.setDuration("ninguna"); ci.setFeatures("ninguna"); ci.setFrequency("ninguna"); ci.setMembers("Adulto_Mayor"); ci.setRole("Adulto_Mayor"); ci.setStatus("Executing"); ci.setLocation("Sala"); ci.setActions("Leyendo_libro"); ci.setTypeofADL("Instrumental"); ci.setTools("Libro"); ci.setOntologyPath("RUTA ARCHIVO DE LA ONTOLOGIA"); ci.setParticularRulesPath("RUTA ARCHIVO DE LAS REGLAS QUE EXPRESAN EL COMPORTAMIENTO IDEAL DEL ADULTO MAYOR"); </pre>

<pre> public class ElderAction extends Action{ public Object execute() { ContextualInformation ci = new ContextualInformation(); ci.setActivity(""); ci.setTime(""); ci.setCommunity(""); ci.setDate(""); ci.setDuration(""); ci.setFeatures(""); ci.setFrequency(""); ci.setMembers(""); ci.setRole(""); ci.setStatus(""); ci.setLocation(""); ci.setActions(""); ci.setTypeofADL(""); ci.setTools(""); ci.setOntologyPath(""); ci.setParticularRulesPath(""); ci.setGeneralRulesPath(""); this.sendNotificationInfo("contextawareagent@localhost", ci.xmlMessageFactory(), " ", ci.requestPaths()); return null; } </pre>	<pre> public class ElderAction extends Action{ public Object execute() { ContextualInformation ci = new ContextualInformation(); ci.setActivity("Leyendo"); ci.setTime("noon"); ci.setCommunity("Ninguna"); ci.setDate("09-06-2010"); ci.setDuration("ninguna"); ci.setFeatures("ninguna"); ci.setFrequency("ninguna"); ci.setMembers("Adulto_Mayor"); ci.setRole("Adulto_Mayor"); ci.setStatus("Executing"); ci.setLocation("Sala"); ci.setActions("Leyendo_libro"); ci.setTypeofADL("Instrumental"); ci.setTools("Libro"); ci.setOntologyPath("RUTA ARCHIVO DE LA ONTOLOGIA"); ci.setParticularRulesPath("RUTA ARCHIVO DE LAS REGLAS QUE EXPRESAN EL COMPORTAMIENTO IDEAL DEL ADULTO MAYOR"); ci.setGeneralRulesPath("RUTA ARCHIVO DE LAS REGLAS QUE RIGEN EL AmI"); this.sendNotificationInfo("contextawareagent@localhost", ci.xmlMessageFactory(), " ", ci.requestPaths()); return null; } </pre>
Ejercicio 2. Acción Agente Risk	
Plantilla de la acción del agente Risk	Plantilla llenada por los participantes
<pre> public class RiskActionCreceer extends Action{ public Object execute() { </pre>	<pre> public class RiskActionCreceer extends Action{ public Object execute() { </pre>

Tabla 4. Plantillas de código para la práctica de implementación

Para evaluar la facilidad de uso del componente se aplica una encuesta, donde se tomaron ejemplos de los cuestionarios utilizados por el *System Usability Scale* (SUS) (Bangor, 2008) el cual fue desarrollado en 1986 como parte de la introducción de la ingeniería de usabilidad a los sistemas de oficina de *Digital Equipment Co. Ltd*, su propósito era proporcionar un test fácil de completar, de puntuar y que permitiera establecer comparaciones cruzadas entre productos. Ha sido usado extensivamente en evaluaciones de proyectos resultando como simple y fiable. La encuesta se les entregó a los participantes al terminar con la evaluación de la implementación. El formato completo del cuestionario se muestra en el Apéndice C. En la Tabla 7 aparecen los resultados de esta encuesta, donde se les dio a los participantes la oportunidad de elegir una de cinco opciones que variaban del completamente en desacuerdo, en desacuerdo, neutro, de acuerdo y completamente de acuerdo. A los cuales se les dieron los valores de (-2), (-1), (0), (1) y (2) respectivamente, para elaborar una apreciación de los resultados en escala Likert.

VI.2 Resultados de la Evaluación

Etapas	No. Participante	Calificación	Tipo de Error	Comentarios del participante	Observaciones
Etapas 1. Homogenización de conocimientos	1	9	E.Agente	<i>Ninguno</i>	<i>Ninguna</i>
	2	10	Ninguno	<i>Ninguno</i>	<i>Ninguna</i>
	3	10	Ninguno	<i>Ninguno</i>	<i>Ninguna</i>
	4	10	Ninguno	<i>Ninguno</i>	<i>Ninguna</i>
	5	10	Ninguno	<i>Ninguno</i>	<i>Ninguna</i>
	6	9	E.Ont.	<i>Ninguno</i>	<i>Ninguna</i>
	7	9	E.Ont.	<i>Ninguno</i>	<i>Ninguna</i>
	8	9	E.Contexto	<i>Ninguno</i>	<i>Ninguna</i>
Etapas 2. Evaluación del componente para	1	46.25	F.Concepto F.Relación	<i>“Tengo dudas con que poner en la ontología y las reglas”</i>	<i>Confunde la información de las reglas y la ontología</i> <i>Se confunde de donde tomar las variables para instanciar la ontología</i>
	2	50	E.Rep.Sist.	<i>“¿Subrayo todo lo que es contexto o nada mas lo que lo orillo a medicarse?”</i>	<i>Ninguna</i>

diseñar sistemas AmI	2	50	E.Rep.Sist.	<i>“¿Que es instanciar?”</i>	<i>Ninguna</i>
	3	48.5	F.Form.Regla E.Rep.Sist. F.Concepto	<i>“¿Hay que subrayar con respecto a la ontología?”</i>	<i>Dudas sobre hasta donde se tenia que subrayar</i>
	4	45	F.Relación F.Concepto F.Form.Regla	<i>“¿Hay que subrayar con respecto a la ontología?”</i>	<i>Dudas respecto a los colores con los que tenia que subrayar</i>
	5	50	E.Rep.Sist.	<i>Ninguno</i>	<i>Comentarios acerca de la redacción de los ejercicios</i>
	6	27	F.Concepto F.Relación F.Form. Regla F.Tiempo	<i>“¿Hay que subrayar con respecto a la ontología?”</i> <i>“¿Hay que identificar conforme se vio en las diapositivas?”</i> <i>“¿Le hago así y lo vuelvo a hacer?”</i> <i>“¿Cual es la acción a representarse?”</i> <i>“¿Desde donde empezar a representarse?”</i>	<i>A este participante se le tuvo que volver a explicar la mayoría de los conceptos y explicarle las instrucciones paso a paso</i>
	7	38.25	F.Concepto F.Relación F.Form.Regla	<i>“¿Hay que subrayar con respecto a la ontología?”</i> <i>“¿Hay que identificar conforme se vio en las diapositivas?”</i>	<i>Ninguna</i>

			F.Tiempo	<i>“¿Le hago así y lo vuelvo a hacer?”</i> <i>“¿Cual es la acción a representarse?”</i> <i>“¿Desde donde empezar a representarse?”</i>	
	8	46	F.Relación F.Concepto F.Tiempo	<i>“¿Cual es la acción a representarse?”</i> <i>“¿Desde donde empezar a representarse?”</i>	Ninguna
Etapas 3. Evaluación del componente para implementar sistemas AmI	1	NO	N.Clasif.	<i>“¿Qué es lo que voy a poner en el agente Elder?”</i> <i>“No entiendo la sintaxis de JESS”</i>	Ninguna
	2	NO	N.Clasif.	<i>“¿Cómo inicializo los agentes?”</i> <i>“No sale el cuadro de los pericos, se cicla”</i>	Ninguna
	3	NO	N.Clasif.	Ninguno	Dudas respecto a como dar de alta los agentes
	4	NO	N.Clasif.	Ninguno	Dudas respecto a como dar de alta los agentes
	5	NO	N.Clasif.	Ninguno	Dudas respecto a como dar de alta los agentes Como se instala el servidor Openfire

	6	NO	N.Clasif.	<i>Ninguno</i>	<i>Dudas respecto a como dar de alta los agentes</i>
	7	NO	N.Clasif.	<i>Ninguno</i>	<i>Dudas de cómo instalar el JDK 1.4, los jar de JESS y dar de alta los agentes</i>
	8	NO	N.Clasif.	<i>Ninguno</i>	<i>Dudas respecto a como dar de alta los agentes</i>

Tabla 5. Resultados de las tres etapas de la evaluación

NOMENCLATURA	
Clave	Descripción
E.Ont.	<i>Error Ontología pregunta 7 primer examen</i>
E.Agente.	<i>Error Agente pregunta 10 primer examen</i>
E.Contexto	<i>Error contexto pregunta 5 primer examen</i>
N.Clasif.	<i>No hay Clasificación</i>
E.Rep.Sist.	<i>Error de la Representación del Sistema en la Ontología. Este error consiste en que se representa las acciones del sistema y no las del adulto mayor en la ontología</i>
F.Tiempo	<i>Error de que Falto el Tiempo en la Regla del ejercicios numero dos del examen de diseño</i>
F.Relación	<i>Falta Relación en la ontología</i>
F.Concepto	<i>Falta Concepto en la ontología</i>
F.Form.Regla	<i>No tiene Formato de Regla</i>

Tabla 6. Nomenclatura de los errores presentados en los exámenes aplicados

VI.2.1 Resultados de la Etapa 1. Homogenización de los conocimientos

Como se muestra en la Tabla 5 los resultados del primer examen teórico descrito en la sección 1.2.1, fueron los siguientes:

Cuatro (4) de los ocho (8) participantes obtuvieron la calificación máxima, que es de diez (10) puntos, y los otros cuatro (4) obtuvieron nueve (9) puntos. Los errores mas comunes fueron los relacionados a la definición de ontología, contexto y ciclo de vida en los agentes de *software*. Esto indica que los participantes obtuvieron un resultado satisfactorio, además proyecta que los conocimientos aprendidos en la exposición se comprendieron. Sin embargo, para continuar con la etapa siguiente de la evaluación, se explicaron las preguntas que no fueron correctamente respondidas.

VI.2.2 Resultados de la Etapa 2. Evaluación del componente para diseñar sistemas Aml

En esta siguiente etapa los resultados del examen, que se explicó en la sección 1.2.2 y que se resumen en la Tabla 5, muestran que solo dos (2) participantes obtuvieron la puntuación máxima, el cual es de cincuenta (50) puntos, cinco (5) de ellos obtuvieron una puntuación en el rango de los cuarenta (40) a los cincuenta (50) puntos, los cuales cometieron algunos errores comunes como; el olvidar incluir relaciones entre los conceptos de la ontología (cinco personas cometieron ese error), olvidar representar los conceptos en la ontología (seis personas cometieron ese error), representaron en la ontología las acciones del sistema y no las del adulto mayor (tres personas cometieron ese error), cuatro personas no siguieron el formato de escribir como una regla y tres participantes olvidar incluir el tiempo en la regla que define el comportamiento del adulto mayor. En resumen:

- El error más común en el ejercicio de diseño fue que a los participantes no estructuraron bien la representación ontológica del contexto del adulto mayor ya que seis (6) participantes olvidaron conceptos o relaciones en la instancia de la ontología.
- Tres (3) de los participantes confunde la parte de la representación en la ontología, ya que no solo expresan el contexto del Adulto Mayor, sino también el del sistema (lo cual es un error, solamente se debe expresar el contexto del A.M.).

VI.2.3 Resultados de la Etapa 3. Evaluación del componente para implementar sistemas Aml

En la etapa final de esta evaluación se aplicó una práctica de laboratorio guiada, en la que los participantes tenían que programar agentes de *software* que se comunicaran con el *Componente ELDeR*, el ejercicio aplicado se explica en la sección 1.2.3 de este capítulo. El propósito de este ejercicio fue conocer la percepción de facilidad de uso y utilidad del Componente ELDeR. Para lograr lo anterior, al finalizar el ejercicio de programación, los estudiantes contestaron una encuesta de opinión a los participantes, descrita anteriormente. Dichos resultados se plasman en la Tabla 7 y Tabla 8.

Algunos de los resultados que se obtuvieron de la encuesta, utilizando el instrumento SUS, son los siguientes:

- Cuatro (4) de los participantes concluyen que lo más difícil fue implementar la comunicación entre los agentes con el *Componente ELDeR* y los otros cuatro (4) considera que lo más difícil fue la sintaxis en JESS.
- Siete (7) participantes estuvieron entre de acuerdo y completamente de acuerdo de que la explicación dada fue clara.
- Tres (3) participantes estuvieron completamente de acuerdo en utilizar el Componente ELDeR en el futuro, cuatro (4) estuvieron de acuerdo y uno (1) en posición neutra.
- Cinco (5) personas estuvieron de acuerdo en que se les hizo fácil programar, una (1) persona estuvo completamente de acuerdo y dos (2) en término neutro.
- Seis (6) personas decidieron que necesitan conocimientos previos antes de utilizar el *Componente ELDeR*, una (1) esta en desacuerdo y una (1) esta completamente de acuerdo. Entre los conocimientos que se eligieron como previos están los agentes de software y la sintaxis del lenguaje JESS.

Algunos de los resultados que se obtuvieron de la encuesta, de la pregunta número diez que fue agregada y no es propia del instrumento SUS, fueron las siguientes:

- Todos los participantes concluyeron que lo mas sencillo de realizar fue la identificación del contexto, la identificación de las reglas que rigen el comportamiento del Sistema AmI e identificar las reglas que definen el comportamiento esperado del usuario o del Adulto Mayor.
- Los dos tipos de reglas que utiliza el componente fueron fáciles de identificar y de diseñar, sin embargo al momento de implementarlas con el lenguaje JESS tuvieron dificultades.

<i>Enunciados</i>	<i>Número de Personas que eligieron de estas opciones</i>					
	<i>Completamente en desacuerdo (-2)</i>	<i>En desacuerdo (-1)</i>	<i>Posición neutra (0)</i>	<i>De acuerdo (1)</i>	<i>Completamente de acuerdo (2)</i>	<i>Promedio</i>
<i>La explicación dada para utilizar el componente ELDeR fue clara.</i>	0	0	1	5	2	1.125
<i>Si requiriera implementar en el futuro una aplicación AmI, consideraría utilizar el componente ELDeR.</i>	0	0	1	4	3	1.25
<i>Encontré que implementar una aplicación AmI con el componente ELDeR es complejo.</i>	0	2	5	1	0	-0.125
<i>Creo que necesitaría del apoyo de un experto para programar AmI con el componente ELDeR.</i>	1	2	3	2	0	-0.25
<i>Imagino que la mayoría de los desarrolladores con una experiencia en programación como la mía, podrán aprender pronto a usar el componente ELDeR.</i>	0	0	1	5	2	1.125
<i>Me sentí confiado cuando comencé a implementar la aplicación de CARE-Me.</i>	0	1	5	2	0	0.125
<i>Me pareció que había demasiadas inconsistencias en la implementación.</i>	1	3	3	1	0	-0.5
<i>Considero que aun necesito aprender otros conocimientos antes de poder utilizar el componente ELDeR para implementar sistemas AmI.</i>	0	1	0	6	1	0.875
<i>Fue fácil de programar una aplicación AmI utilizando el componente ELDeR.</i>	0	0	2	5	1	0.875

Tabla 7. Resultados de la encuesta final de la evaluación

	<i>Número de Personas que eligieron de estas opciones</i>								
<i>Enunciados</i>	1	2	3	4	5	6	7	8	9
<i>Identificar el contexto</i>	5	1	2	0	0	0	0	0	0
<i>Identificar las reglas que rigen el comportamiento del Sistema AmI</i>	3	4	1	0	0	0	0	0	0
<i>Identificar las reglas que definen el comportamiento esperado del usuario o del Adulto Mayor</i>	1	3	3	1	0	0	0	0	0
<i>Instanciar la ontología</i>	1	0	0	4	1	0	2	0	0
<i>Implementar las reglas con el lenguaje JESS</i>	0	0	2	1	1	2	0	1	1
<i>Implementar la acción de los agentes SALSA</i>	0	1	0	1	2	3	0	1	0
<i>Implementar el razonamiento de los agentes SALSA</i>	1	0	0	0	0	3	4	0	0
<i>Implementar la comunicación de los agentes con el componente ELDeR</i>	1	0	0	0	0	0	1	3	3
<i>Especificar el contexto en el agente ElderAgent para comunicarlo con el componente ELDeR</i>	0	1	0	0	3	0	0	2	2

Tabla 8. Resultados de la pregunta no. 10 de la encuesta final de la evaluación

VI.3 Discusión

La etapa de evaluación sirvió para conocer la facilidad de uso que tiene el *Componente ELDeR* al utilizarse para el desarrollo de AmI para apoyar las AVD del adulto mayor. Algunos de los puntos sobresalientes de esta evaluación fueron los siguientes:

Considerando las hipótesis:

H1. *“El Componente ELDeR es útil y fácil de usar para el diseño e implementación de la funcionalidad consciente del contexto de los sistemas AmI”.*

H2. *“La Ontología CArE es fácil de extender para representar AVD”.*

H3. *“La base de reglas CArE facilitan el diseño y la implementación del comportamiento del sistema AmI con base al comportamiento esperado del adulto mayor”.*

- Se concluye, para H2 y H3, que el diseño de la ontología y las reglas fue considerado fácil. Sin embargo, algunos de los participantes tuvieron problemas para distinguir el contexto del adulto mayor de las acciones que realizaría el sistema.
- Para los participantes de la evaluación fue más fácil diseñar las reglas e identificarlas que programarlas en el lenguaje JESS.
- A partir del análisis de los resultados obtenidos de los ejercicios de programación del sistema AmI y la encuesta con el instrumento SUS, se llegó a la conclusión de que el *Componente ELDeR* es fácil de usar, comprobando la hipótesis H1.
- La comunicación de los agentes no es sencilla de entender, por lo que se requiere que los programadores realicen varios ejercicios de programación.
- La hipótesis H1, se complementa con que los participantes consideraron que el *Componente ELDeR* es útil al componente, ya que están dispuestos a utilizarlo en un futuro.

Capítulo VII

Conclusiones y Trabajo Futuro

Para el desarrollo de sistemas AmI que apoyen al adulto mayor durante sus actividades de vida diaria, se presentan varios retos entre los que están:

- La representación del contexto del adulto mayor
- Inferir riesgos a partir del contexto representado
- Crear infraestructuras que apoyen al desarrollo de sistemas AmI que apoye al adulto mayor a realizar sus actividades de vida diaria (AVD) con independencia.

Para abordar estos retos, se diseñó e implementó un componente de *software*, llamado *Componente ELDeR*, que encapsula agentes autónomos que tienen la funcionalidad para representar el contexto del adulto mayor realizando sus AVD e inferir la situación de riesgo a la que se enfrenta el adulto mayor. El *Componente ELDeR* contiene una Ontología para representar el contexto y una Base de Reglas para inferir el tipo de asistencia requerida por el adulto mayor. Además, el diseño de la Base de Reglas, facilita a los desarrolladores adaptar la aplicación AmI, implementada con el *Componente ELDeR*, a las necesidades y preferencias del usuario (adulto mayor que usará la aplicación). Lo anterior se logra cuando el desarrollador del sistema AmI especializa las reglas que rigen el comportamiento esperado del adulto mayor.

Para ilustrar el uso y comportamiento del *Componente ELDeR*, se presentó la implementación del sistema CARE-Me, el cual asiste al adulto mayor para que se medique adecuadamente. Aun cuando lo anterior se presentó como evidencia de que el *Componente ELDeR* provee de mecanismos que facilitan implementar sistemas AmI, es importante conocer el punto de vista de programadores, y en particular es importante identificar si consideran que es fácil de usar. Por lo que la etapa final de esta tesis, consistió de una evaluación con 8 programadores que usaron el *Componente ELDeR* para diseñar un sistema AmI e implementarlo mediante ejercicios de programación diseñados para realizarse como una práctica de laboratorio de un curso de Cómputo Móvil y Ubicuo impartido en la Facultad de Ingeniería.

En la etapa de evaluación se concluye que el diseño de la ontología y las reglas fue considerado fácil. Sin embargo, durante la etapa de implementación, los participantes enfrentaron algunas dificultades, es particular para implementar las reglas con el lenguaje JESS. Aun así, los resultados de los ejercicios aplicados y de la encuesta final fueron que el *Componente ELDeR* es fácil de usar, pero requiere extenderse para proveer mecanismos que ayuden a programar las reglas y ejercicios previos para la comprensión de la filosofía de los agentes de software. Como conclusión final, se obtuvo que los participantes consideraron útil al componente, ya que estarían dispuestos a utilizarlo en un futuro para el desarrollo de aplicaciones AmI.

VII.1 Contribuciones

Como contribuciones de este trabajo de investigación se presentan las siguientes:

- El diseño de una representación ontológica para las actividades de vida diaria del adulto mayor en el hogar. Esta ontología permite plasmar el contexto generado por la persona mientras realiza sus actividades. Los conceptos a representar son: Actividad, Acciones, Objetos, Características, Ubicación, Hora, Frecuencia, Estado, Comunidad, Fecha, Duración. El diseño de una Base de Reglas que modelan el comportamiento esperado del sistema AmI, así como el del adulto mayor. Por ejemplo, una actividad esperada a realizar por el adulto mayor es medicarse según su prescripción médica y si no lo cumple, el ambiente le tendrá que recordar que se tome sus medicinas correctamente.
- La evaluación de la facilidad de uso del sistema AmI se realizó en tres fases. La primera consistió en una presentación introductoria a los conceptos de cómputo ubicuo, sistemas conscientes del contexto, técnicas de representación contextual y agentes de software para finalizar con un cuestionario de opción múltiple que permitió conocer la comprensión de la teoría. La siguiente etapa, fue una presentación donde se explicaba la arquitectura del *Componente ELDeR* para después proseguir con la aplicación de un ejercicio de diseño. Finalmente, en la tercera etapa se elaboró una práctica guiada de laboratorio donde los participantes programaron una aplicación y concluyeron contestando una encuesta de opinión

donde se pretendía evaluar la facilidad de uso del *Componente ELDeR*. Los resultados de esta evaluación se consideran los siguientes:

- El diseño de la ontología y las reglas fue considerado fácil.
- Para los participantes de la evaluación fue más fácil diseñar las reglas e identificarlas que programarlas en el lenguaje JESS.
- El *Componente ELDeR* es fácil de usar, pero requiere extenderse para proveer mecanismos que ayuden a programar las reglas.
- La comunicación de los agentes no es sencilla de entender, por lo que se requiere que los programadores realicen varios ejercicios de programación.
- Los participantes consideraron que el *Componente ELDeR* es útil al componente, ya que están dispuestos a utilizarlo en un futuro.
- Durante el desarrollo del proyecto se generaron las siguientes publicaciones que plasman los resultados parciales:
 - Juan P. García-Vazquez. Marcela D. Rodríguez. Angel Andrade. Diana Saldaña-Jiménez. (2009). *Ambient Information Systems for Supporting Ageing in Place*. Presentado en: The 2009 Sixth International Conference on Information Technology: New Generations. Páginas 1232-1237.
 - Diana Saldaña-Jiménez. Marcela D. Rodríguez. Juan Pablo García-Vázquez. (2009). *ELDeR: An Ontology for Enabling Living inDependently of Risks*. Springer Berlin/Heidelberg. Volumen 5872/2009. On the Move to Meaningful Internet Systems: OTM 2009 Workshops. Páginas 622-627.
 - Diana Saldaña-Jiménez. Marcela D. Rodríguez. Adan-Noe Espinoza. Juan P. Garcia-Vazquez. (2009). *A context-aware component for identifying risks associated to elders' activities of daily living*. Pervasive Computing Technologies for Healthcare, 2009. Pervasive Health 2009. 3rd International Conference On. London UK. Páginas 1-4.
 - Antonio Sánchez. Ricardo Tercero. Diana Saldaña-Jiménez. Lisa Ball. *SDBI: An Ontology Based Smart Home Lab Environment*. MICAI 2009. Workshop on Intelligent Learning Environments (WILE 2009). Guanajuato, Mexico.

- Finalmente se obtuvo una publicación en revista que plasma parcialmente el diseño final del componente y un artículo con el resultado obtenido de la evaluación respectivamente:
 - Juan Pablo, García-Vázquez., Marcela D. Rodríguez., Mónica E. Tentori., Diana Saldaña-Jiménez. (2010). *An agent-based Architecture for Developing Activity-Aware Systems for Assisting Elderly*. Aceptado para J. Of Universal Computer Science, JUCS 2010.
 - Marcela D. Rodríguez., Jesus Favela., Diana Saldaña. (2010). *Students' User Experience with the SALSA Architecture for Developing Ubiquitous Computing Applications*. UbiComp '10. ACM.

VII.2 Trabajo Futuro

Como trabajo futuro se propone implementar una interfaz gráfica que permita la programación de reglas del lenguaje JESS, para facilitar la aplicación del ambiente inteligente en el hogar del adulto mayor.

Otra de las propuestas, es dotar al *Componente ELDeR* de la capacidad de razonamiento para adaptarse automáticamente al usuario. Explorando algoritmos de inteligencia artificial que permitan realizar lo anterior.

Finalmente, se propone extender la funcionalidad del *Componente ELDeR* para que permita generar historias de la actividad del adulto mayor con base al contenido de la *Ontología CARE*. Esta funcionalidad podría ser útil para implementar aplicaciones AmI que con base al historial de actividades se generen diagnósticos en la salud del adulto mayor. Un historial de la actividad plasma las acciones pasadas del anciano dentro de su hogar y por consiguiente se podría inferir la existencia de un deterioro cognitivo o motriz. Por ejemplo, un sistema AmI que determine si los patrones de sueño del adulto mayor se han alterado, lo que podría indicar que está deprimido o con algún otro problema de salud.

Referencias

- (Ahmed, 2007) Ahmed Elfatraty. (2007). *Dealing with change: Components Versus Services*. Communications of the ACM, Volumen 50. Páginas 35-39.
- (Amandi, 2001) Amandi, Analía. (2001). *Desarrollo de Sistemas Multi-Agentes*. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. Número 13. Páginas 33-35.
- (Bangor, 2008) Bangor, Aaron., Kortum, Philip T., Miller, James T. (2008). An Empirical Evaluation of the System Usability Scale. *International Journal of Human Computer Interaction*. Volumen 26, Número 6. Páginas 574-594.
- (Bolchini, 2007) Bolchini, Cristiana. Curino, Carlo A. Quintarelli, Elisa. Schereiber, Fabio A. Tanca, Letizia. (2007). *A Data-oriented Survey of Context Models*. Aparece en ACM, Volumen 36. Páginas 19-26.
- (Bradshaw, 1997) Bradshaw, J. M. (Ed.). (1997). *Software agents*. Cambridge, MA: MIT Press. Laurel, B. Interface agents: Metaphors with character. In Bradshaw, J. M.
- (Brown, 1996) Brown, M. (1996). *Supporting User Mobility*. International Federation for Information Processing.
- (Bucur, 2005) Bucur, Oana., Boissier. (2005). Representing Context in an Agent Architecture for Context-Based Decision Making. In *Context Representation and Reasoning Satellite Workshop at CONTEXT*.
- (Capetuti, 2007) Capetuti, Liz., Capezuti, Elizabeth A., Siegler, Eugenia L., Siegler, Mathy., Mezey, D. (2007). *The encyclopedia of elder care; The comprehensive Resource on Geriatric and Social Care*. Aparece en: Springer Publishing Company, 860 páginas.
- (Chen, 2004) Chen, H. Finin, T. Anupam Joshi, Kagal, L. Perich, F. and Dipanjan Chakraborty. (2004). *Intelligent Agents Meet the Semantic Web in Smart Spaces*. Aparece en: *Internet Computing*, IEEE. Páginas 69-79.
- (CoDAMoS, 2009) Context Driven Adaptation of Mobile Services. Extraído de <http://distrinet.cs.kuleuven.be/projects/CoDAMoS/> el 14 de Agosto del 2009.
- (CONAPO, 2008) Consejo Nacional de Población Secretaría de Gobernación. *El Envejecimiento de la Población en México*. Extraído de <http://www.conapo.gob.mx> el 23 de Octubre del 2008.

(Cooperstock, 1995) Cooperstock, J., Tanikoshi, K., Beirne, G., Narine, T., Buxton, W. (1995). *Evolution of a Reactive Environment*. Presentado en: Evolution of a Reactive Environment CHI '95. Páginas 170-177.

(DAML, 2009) DARPA Agent Markup Language. Extraído de <http://www.daml.org/> el 29 de Octubre del 2009.

(Dey, 1999) Dey Anind K. and Abowd Gregory. (1999). *Towards a Better Understanding of Context and Context – Awareness*. Panel en el: 1st International Symposium on Handheld and Ubiquitous Computing (HUC '99). Páginas 304-307.

(Dey, 2001) Dey, A.K. (2001). *Understanding and Using Context*. Aparece en: Personal and Ubiquitous Computing. Springer-Verlag. Páginas 4-7.

(Elrod, 1993) Elrod, S., Hall, G., Costanza, R., Dixon, M., des Rivieres, J. (1993). *Responsive Office Environments*. CACM. Volumen 36. Páginas 84-85.

(Espinoza, 2009) Adán-Noé Espinoza, Juan-Pablo García-Vázquez, Marcela D. Rodríguez, Ángel G. Andrade, Carmen García-Pena. *Enhancing a Wearable Help Button to Support the Medication Adherence of Older Adults*. LA-WEB/CLIHIC 2009. Páginas 3-7.

(Fickas, 1997) Fickas, S., Korteum, G., Segall, Z. (1997). *Software Organization for Dynamic and Adaptable Wearable Systems*. Presentado en: 1st International Symposium on Wearable Computers. Páginas 56-63.

(Garate, 2005) A. Garate, I. Lucas, N. Herrastini and A. Lopez. (2005). *Ambient intelligence as paradigma of a full automation process at home in a real application*. Aparece en: IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA '05. Páginas 475-479.

(García, 2009) García-Vázquez, Juan Pablo.; Rodríguez, M.D.; Andrade, A.G.; Saldaña, Diana.; Mercado, F.; Ruelas, E. (2009). *Ambient Information Systems for Supporting Ageing in Place*. Aparece en: Information Technology: New Generations, '09. Páginas 1232-1237.

(García, 2010) Juan Pablo, García-Vázquez., Marcela D. Rodríguez., Mónica E. Tentori., Diana Saldaña-Jiménez. (2010). *An agent-based Architecture for Developing Activity-Aware Systems for Assisting Elderly*. Aceptado para J. Of Universal Computer Science, JUCS 2010.

(García, 2010) Juan Pablo, García Vázquez, Marcela D. Rodríguez., Angel G. Andrade. (2010). *Design Dimensions of Ambient Information Systems to Assist elderly with their Activities of Daily Living*. Aparece en: ACM 978-1-60558-843-8. UbiComp '10.

(Genesereth, 1994) M.R. Genesereth and S.P. Ketchpel. (1994). *Software agents*. Communications of the ACM, Volumen 37. Páginas 48-53.

(Google, 2009) Google. (2009). Extraído de <http://www.google.com>

(Griss, 2001) Griss, Martin L. & Pour, Gilda. (2001). *Accelerating Development with Agents Components*. Aparece en: IEEE Computer, Volumen 34. Páginas 37-43.

(Gruber, 1993) Thomas R. Gruber. (1993). *Toward principles for the design of ontologies used for knowledge sharing*. Publicada en el: International Journal of Human-Computer Studies, Volume 43. Páginas 907-928.

(Gruber, 2008) Tom Gruber. (2008). *Ontology*. Publicada en: The Encyclopedia of Database Systems, Ling Liu and M. Tamer Özsu (Eds.), Springer-Verlag.

(Gu, 2005) Gu, T., Keng Pung, H., Qing Zhang, Da. (2005). *A service-oriented middleware for building context-aware services*. Journal of Network and Computer Applications, Volumen 28. Páginas 1-18.

(Guarino, 1998) Guarino N. (1998). *Formal Ontology in Information Systems*. Presentado en FOIS'98. Páginas 6-8.

(Haigh, 2003) Zita Haigh, K. Kiff, L. et al. (2003). *The Independent LifeStyle Assitant (I.L.S.A.): Lessons Learned*. Honeywell Laboratories.

(Harmelon, 2001) Harmelon, F., et al. (2001). *Reference Description of the DAML+OIL ontology markup language*. Extraído en: <http://www.daml.org/2001/03/reference.html>

(Henricksen, 2005) K. Henricksen, J. Indulska, T. McFadden, and S. Balasubramaniam. (2005). *Middleware for distributed context-aware systems*. Por aparecer en: of On the Move to Meaningful Internet Systems, LNCS 3670. Páginas 846-863.

(Honeywell Web Pads, 2008) Honeywell Web Pads. Extraído de http://www.smecc.org/honeywell_webpad_hold.htm

(Hopkins, 2000) Hopkins, Jon. (2000). *Component Primer*. Communications of the ACM. Volumen 43, Número 10. Páginas 27-30.

(Horrocks, 2003) Horrocks, I., Patel-Schneider, P.F., van Harmelen, F. (2003). *SHIQ y RDF a OWL: The making of a Web Ontology Language*. Journal of Web Semantics. Volumen 1. Páginas 7-26.

(Hull, 1997) Hull, R., Neaves, P., Bedford-Roberts, J. (1997). *Towards Situated Computing*. Aparece en: 1st International Symposium on Wearable Computer. Páginas 146-153.

(Jabber, 2009) Jabber. Extraído de <http://www.jabber.com> el 12 de Septiembre del 2009.

(JADE, 2008) JADE. Extraído de <http://jade.tilab.com/> el 23 de Septiembre del 2008.

(JESS, 2009) Java Expert Shell System. Extraído de <http://www.jessrules.com/> el 12 de Septiembre del 2009.

(Kimel, 2007) Kimel J et al. (2007). *Exploring the Nuances of Murphy's Law Long-term Deployments of Pervasive Technology into the Homes of Older Adults*. ACM Interaction. Páginas 38-41.

(Lawton, 1990) Lawton, M.P. (1990). *Aging and performance on home tasks*. Human Factors, Volumen 32. Páginas 527-537.

(McCarthy, 1980) McCarthy, J. (1980). *Circumscription--A Form of Non-Monotonic Reasoning*. Artificial Intelligence. Volumen 13. Páginas 27-39.

(Moruno, 2008) Moruno Miralles, Pedro; Romero Ayuso, Dulce María. *Actividades de la Vida Diaria*. Editorial Masson. Extraído de <http://books.google.com/books?id=WF6nls1pVg4C&printsec=frontcover&dq=Actividades+de+la+vida+diaria&hl=es#PPPI,M1> el 20 de Agosto del 2008.

(Myers, 2003) Myers, Janet. & Phelps, John. (2003). *Consolidated Home Ontology in Protégé*. Honeywell Laboratories.

(OMS, 2008) Organización Mundial de la Salud. (1984). *Informe Técnico No.706*. Extraído de <http://200.10.250.206/bvsacd/eco/038910.pdf> el 15 de Noviembre del 2008.

(Orwing, 2006) Orwing, D., Brandt, N., Gruber-Baldini, A.L. (2006). *Medication Management Assessment for Older Adults in the Community, The Gerontologist*. Páginas 661-668.

(Osmani, 2009) Osmani, Venet. Zhang, Daqing. & Balasubramaniam, Sasitharan. (2009). *Human activity recognition supporting context-appropriate reminders for elderly*. PERVASENSE '09.

(OWL, 2008) Web Ontology Language. Extraído de <http://www.w3.org/TR/owl-features/> el 3 Noviembre del 2008.

(Pajares, 2006) Pajares Martinsanz, Gonzalo. & Santos Peñas, Matilde. (2006). *Inteligencia Artificial e Ingeniería del Conocimiento*. Editorial Alfaomega-RaMa. México, DF.

(Protégé, 2009) Protégé. Extraído de <http://protege.stanford.edu/> el 2 de Marzo del 2009.

(Raian, 2008) Raian, Ali. Sameh Abdel-Naby. Antonio Maña, Antonio Muñoz. & Paolo Girogini. (2008). *Agent Oriented AmI Engineering*. Developing Ambient Intellicenge Book. Springer Paris. Páginas 166-179.

(Ranganathan, 2003) Anand Ranganathan, Roy H. Campbell. (2003). *A Middleware for Context-Aware Agents in Ubiquitous Computing Environments*. Presentado en ACM/IFIP/USENIX International Middleware Conference, 2003, Río de Janeiro, Brasil.

(RDF, 2009) Resource Description Framework. Extraído de <http://www.w3.org/RDF/> el 5 de Marzo del 2009.

(Rekimoto, 1998) Rekimoto, J., Ayatsuka, Y., Hayashi, K. (1998). *Augment-able Reality: Situated Communication through Physical and Digital Spaces*. 2nd International Symposium on Wearable Computers. Páginas 68-75.

(Riboni, 2009) Riboni, Daniele., and Bettini, Claudio. (2009). *Context-Aware Activity Recognition through a Combination of Ontological of Ontological and Statistical Reasoning*. UIC'09.

(Rodríguez, 2005) Rodríguez MD, Favela J, Preciado A, Vizcaino A. (2005). *Agent-based ambient intelligence for healthcare*. AI Communications, IOS Press. Volumen 18. Páginas 201-216.

(Rodríguez, 2005) Rodríguez, M.D. (2005). *Agentes Autónomos en Ambientes de Cómputo Colaborativos Ubicuos*. Tesis Doctoral CICESE.

(Rodríguez, 2008) Rodríguez, M.D., Favela, J. (2008). *Evaluation of an Agent Framework for the Development of Ambient Computing*. OTM Workshops, LNCS 5333, Springer. Páginas 374-383.

(Rodríguez, 2008) Marcela D. Rodríguez., Cecilia Curlango., Juan Pablo García. (2008). *An agent-based component for identifying elders' at home risks through ontologies*. Presentado en: 3rd Symposium of Ubiquitous Computing and Ambient Intelligence.

(Schilit, 1994) Schilit, B., Theimer, M. (2004). *Disseminating Active Map Information to Mobile Hosts*. IEEE Network, Volumen 8. Páginas 22-32.

(Shoham, 1997) Y. Shoham. (1997). *Agent Oriented Programming: a survey*. En: Software Agents, J.M.

(Szyperski, 2002) Szyperski, Clemens., Gruntz, Dominik., Murer, Stephan. (2002). *Component Software Beyond Object Oriented Programming*. Addison-Wesley/ACM Press.

(Wan-rong, 2006) Wan-rong, Jih., Jane, Yung-jen Hsu., Tse-Ming Tsai. (2006). *Context-Aware Service Integration for Elderly Care in A Smart Environment*. En: AAAI 2006 Workshop: Modeling and Retrieval of Context Retrieval of Context. Páginas 44-48.

(Weiser, 1993) Weiser Mark. (1993). Ubiquitous Computing. Aparece en: IEEE Hot Topics. Extraído de: <http://www.ubiq.com/hipertext/weiser/UbiCompHotTopics.html>

(Wooldridge, 1995) M. Wooldridge and N.R. Jeanings. (1995). *Intelligent agents: Theory and practice*. The Knowledge Engineering Review. Voumen 10. Páginas 115-152.

Apéndice A.

Examen de Conocimientos Básicos.

Universidad Autónoma de Baja California

Facultad de Ingeniería, Mexicali

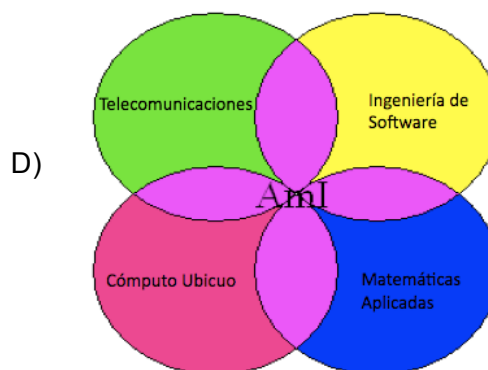
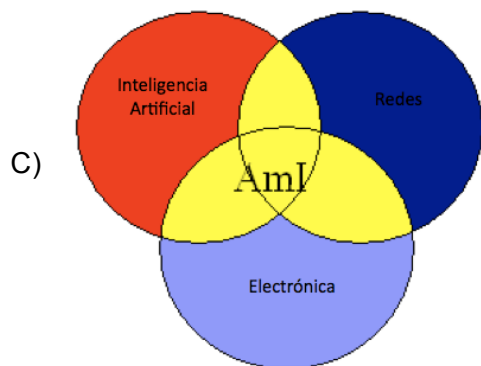
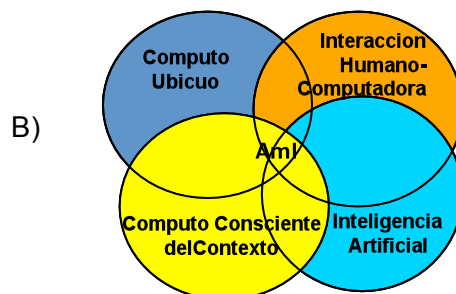
Examen de Conocimientos



Nombre: _____

Instrucciones: Marca la opción correcta que complete los siguientes enunciados.

1.- De acuerdo al artículo de *Shadbot, 2003*, las áreas de la computación que convergen para desarrollar los Ambientes Inteligentes son:



2.- La definición de un Sistema Consciente de Contexto es aquel que:

A) solo toma en cuenta el humor y estado del ánimo del usuario para proveerle información.

B) toma en cuenta el contexto del usuario para proveer información relevante y/o servicios para el usuario en el momento y lugar oportuno.

C) presenta información y servicios cuando el usuarios los solicita.

D) toma en cuenta el contexto de los usuarios para monitorear el desempeño de la computadora.

3.- ¿Cuál de las siguientes definiciones es la que mejor describe y abarca lo que qué es contexto?

A) Es cualquier información que es útil para mejorar la interacción del usuario y la aplicación. Y puede ser identificado contestando las 5 W's.

B) Es la localización exacta del usuario.

C) Se refiere solamente a los objetos que rodea al usuario.

C) Es el humor y estado emocional del usuario al momento de usar una computadora.

4.- ¿Cuáles son los tipos de contexto que se pueden identificar?

A) Contexto fijo y variable.

B) Contexto superior e inferior.

C) Contexto primario y secundario.

D) Contexto fácil y difícil.

5.- Tener un mecanismo para representar contexto es importante porque:

A) Ayuda a que el contexto sea fácilmente diseminado por los diferentes subsistemas o componentes del Ambiente Inteligente, y facilita la inferencia de contexto secundario.

B) Los usuarios pueden mantener una base de datos con información sobre las actividades realizadas diariamente, por si acaso se les olvida lo que hicieron el día de hoy.

C) Facilita a que el sistema capture el contexto con sensores especializados.

D) Facilita a que el programador implemente el sistema en un lenguaje de alto nivel.

6.- Algunos de los tecnologías que se han utilizado para representar contexto son:

A) Java, .Net, y C++.

B) XML, ontologías, y predicados de primer orden.

C) HTML, bases de datos y UML.

D) JESS, PROLOG y Java.

7.- Desde el punto de vista de la informática, una ontología es:

A) Un algoritmo para representar contexto.

B) Un componente inteligente que ayuda a inferir contexto.

C) Una base de datos que describe conceptos del ambiente en un dominio específico, y las relaciones de estos conceptos.

D) Un lenguaje de alto nivel para crear aplicaciones de computo ubicuo.

8. Algunos usos de la ontologías son:

A) Permitir desarrollar sistemas fácil y rápidamente.

B) Permitir la interoperabilidad entre sistemas y crear diccionarios para la Web Semántica.

C) Implementar paginas Web.

D) Desarrollar bases de datos.

9.- Un Agente de Software es una entidad de software que tiene las siguientes características:

A) autonomía, reactividad, proactividad, movilidad y colaboración.

B) inteligencia, ubicuidad, consciencia del contexto.

C) hablan, entienden, y aprenden.

D) sincronización, coordinación y centralización.

10.- El ciclo de vida de un Agente implementado con SALSA incluye los siguientes estados:

A) nace, crece, reproduce y muere.

B) despierta, procesa y muere.

C) percibe información, razona y actúa.

D) captura información, la almacena y la procesa.

S U E R T E!! 😊

Apéndice B.

Examen de diseño que evalúa el Componente ELDeR para diseñar sistemas AmI.

Universidad Autónoma de Baja California
Facultad de Ingeniería
Examen de Diseño



Nombre: _____

Instrucciones: Suponga que usted fue contratado para desarrollar el sistema CARE-Me que se describe en el siguiente escenario. Para lograr lo anterior usted utilizará el componente ELDeR. Siguiendo un enfoque de la Ingeniería de Software, el primer paso a realizar es el diseño y posteriormente la implementación. Por lo que en los siguientes ejercicios usted empezará a diseñar el sistema CARE-Me.

Escenario:

1a) Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro, tal como se muestra en la figura 1. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción medica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos (como se muestra en figura 2) y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.

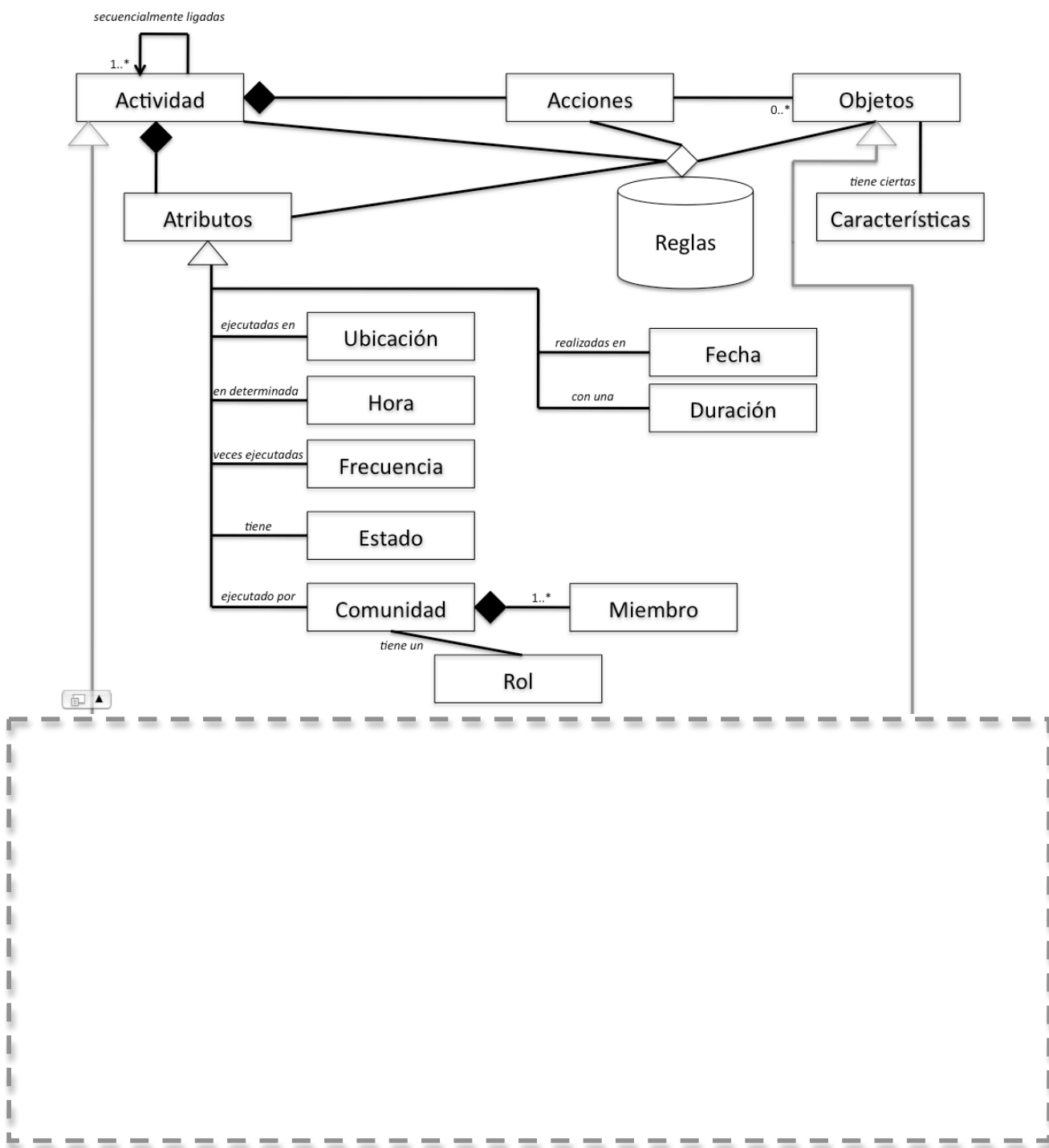


Figura 1

Figura 2

Ejercicio 1. A partir del escenario anterior realice lo siguiente (10 puntos):

- Identifique el contexto del adulto mayor y subráyelo en el texto del escenario con pluma negra desde que el adulto mayor entre en la cocina.
- Instancie la ontología CARE en el espacio correspondiente de la siguiente figura, para representar el contexto identificado.



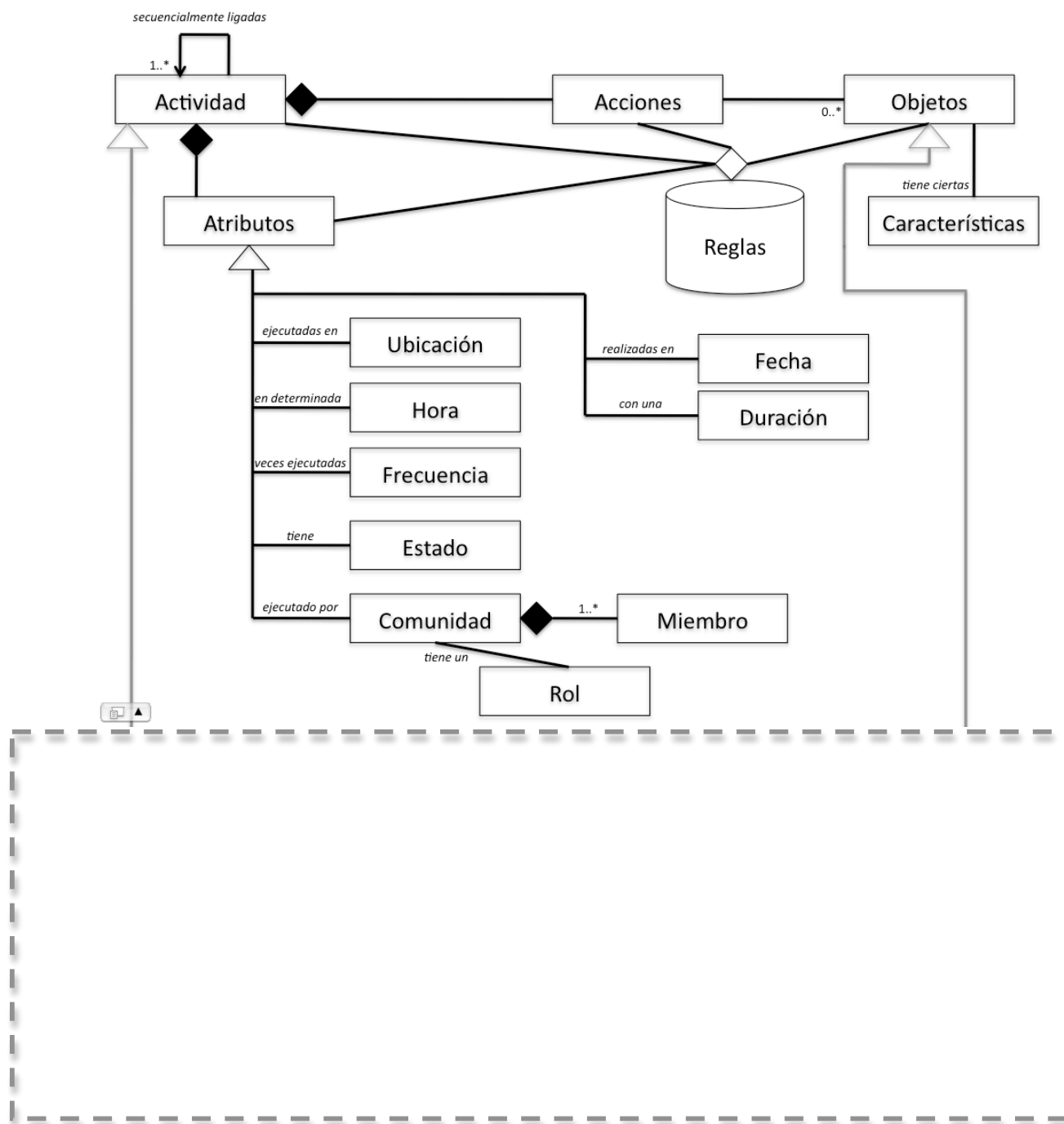
Ejercicio 2. Ahora con pluma azul subraye la información necesaria para obtener las reglas que definen el comportamiento esperado del adulto mayor. Y exprese en el siguiente espacio, con sus propias palabras y expresiones, las reglas identificadas (10 puntos).

Ejercicio 3. Ahora con pluma roja subraye la información necesaria para obtener las reglas que definen el comportamiento del Sistema. Y exprese en el siguiente espacio, con sus propias palabras y expresiones, las reglas identificadas (10 puntos).

Ejercicio 4. Lea el siguiente escenario y siga las instrucciones (10 puntos).

1b) La Sra Ana se dirige a su cuarto porque ya se terminó el día. El sistema esta consciente de que Ana cumplió correctamente con la medicación del día de hoy, por lo que el perico que representa la semana actual aumenta de tamaño. Adicionalmente, como la semana ha terminado, aparece un nuevo perico para representar la semana que empieza.

- Identifique el contexto del adulto mayor y subráyelo en el texto del escenario con pluma negra.
- Instancie la ontología CARE en el espacio correspondiente de la siguiente figura, para representar el contexto identificado.



Ejercicio 5. Ahora con pluma roja subraye la información necesaria para obtener las reglas que definen el comportamiento del Sistema. Y exprese con sus propias palabras las reglas identificadas (10 puntos).

Apéndice C.

Encuesta de opinión del *Componente ELDeR*.

Universidad Autónoma de Baja California

Facultad de Ingeniería

Encuesta de Opinión



Nombre: _____

Instrucciones: De las siguientes afirmaciones, marque la opción que considere conveniente, considerando que 1 significa que usted está **completamente en desacuerdo**, 2 que usted está en **desacuerdo**, 3 si su **posición es neutra**, 4 que usted está **de acuerdo** y el 5 que usted está **completamente de acuerdo**.

1.- La explicación dada para utilizar el componente ELDeR fue clara.

1) 2) 3) 4) 5)

2.- Si requiriera implementar en el futuro una aplicación Aml, consideraría utilizar el componente ELDeR.

1) 2) 3) 4) 5)

3.- Encontré que implementar una aplicación Aml con el componente ELDeR es complejo.

1) 2) 3) 4) 5)

4.- Creo que necesitaría del apoyo de un experto para programar Aml con el componente ELDeR.

1) 2) 3) 4) 5)

5.- Imagino que la mayoría de los desarrolladores con una experiencia en programación como la mía, podrán aprender pronto a usar el componente ELDeR.

1) 2) 3) 4) 5)

6.- Me sentí confiado cuando comencé a implementar la aplicación de CARE-Me.

- 1) 😂 2) 😐 3) 🤔 4) 😊 5) 🤗

7.- Me pareció que había demasiadas inconsistencias en la implementación.

- 1) 😂 2) 😐 3) 🤔 4) 😊 5) 🤗

En caso de haber identificado inconsistencias, por favor explique cuales.

8.- Considero que aun necesito aprender otros conocimientos antes de poder utilizar el componente ELDeR para implementar sistemas Aml.

- 1) 😂 2) 😐 3) 🤔 4) 😊 5) 🤗

En caso de que considere que requiere otros conocimientos para utilizar el componente, mencione cuales.

9.- Fue fácil de programar una aplicación Aml utilizando el componente ELDeR.

- 1) 😂 2) 😐 3) 🤔 4) 😊 5) 🤗

Explique su respuesta:

10.- Numere de lo mas fácil a lo mas difícil las etapas necesarias para implementar una aplicación Aml con el componente ELDeR. Considerando que el numero 1 es el mas fácil, 2 menos fácil, y así sucesivamente.

____ Identificar el contexto

____ Identificar las reglas que rigen el comportamiento del Sistema Aml

____ Identificar las reglas que definen el comportamiento esperado del usuario o del Adulto Mayor

____ Instanciar la ontología

____ Implementar las reglas con el lenguaje JESS

____ Implementar la acción de los agentes SALSA

____ Implementar el razonamiento de los agentes SALSA

____ Implementar la comunicación de los agentes con el componente ELDeR

____ Especificar el contexto en el agente ElderAgent para comunicarlo con el componente ELDeR

Agregue una explicación detallada de la etapa que le fue mas difícil realizar.

Apéndice D.

Presentación de los conceptos teóricos necesarios para la comprensión del *Componente ELDeR*.

3

CONTENIDO

- Introducción a los Ambientes Inteligentes
- Sistemas Conscientes del Contexto
- Retos de los Sistemas Conscientes del Contexto
- Representación del Contexto
- Ontologías

3

4

Ejemplo de un Ambiente Inteligente



4



Universidad Autónoma de Baja California

Tutorial Teórico

Presentado por: Diana Saldana Jimenez

México, B.C., a 31 de Mayo del 2010.

5

Introducción a los Ambientes Inteligentes

- Problema: Falta de adherencia del adulto mayor a la medicación.
- Solución: Que el hogar del adulto mayor este adecuado con tecnologías que le ayuden a realizar exitosamente la actividad de medicarse.

5

Definición de Ambiente Inteligente

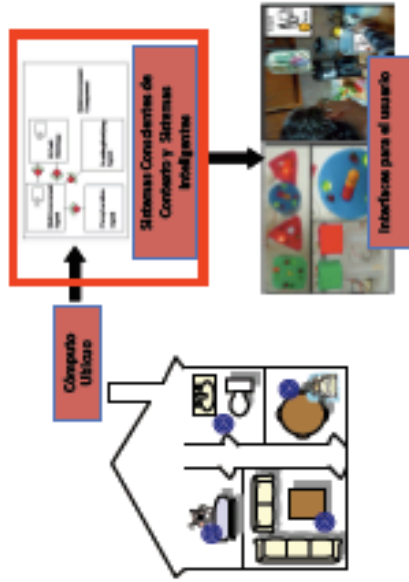
- El concepto de Ambiente Inteligente (Ami) envuelve la convergencia de algunas áreas computacionales.
 - Cómputo Ubicuo
 - Sistemas Inteligentes
 - Sistemas Conscientes del Contexto
 - Interfaces para el usuario (Interacción Humano-Computadora)

30/03/10

Diana Sotillo, PhD, UNAM

6

Ami en el hogar del Adulto Mayor



30/03/10

Diana Sotillo, PhD, UNAM

8

Ami en el hogar del Adulto Mayor Shadbot, 2003



30/03/10

Diana Sotillo, PhD, UNAM

7

Sistemas Conscientes del Contexto

- "Un software que se adapta a la ubicación, las personas y objetos, así como los cambios en los objetos a través del tiempo". [Schilit, B., Theimer, M., 1994]
- "Un sistema es consciente de contexto si se usa el contexto para proveer información relevante y/o servicios para el usuario, donde la relevancia depende de la tarea del usuario". [Dey Anind K. and Abowd Gregory]

30/03/10

Diana Sotillo, PhD, UNAM

9

Definición de contexto

- "El contexto es cualquier información que puede ser usada para caracterizar la situación de la entidad. Una entidad es una persona, lugar u objeto que se considera relevante para mejorar la interacción entre un usuario y una aplicación". [Dey Anind K. and Abowd Gregory]

30/03/20

Clara Sotillo, Anaïs, UAMC

9

Five W's

- Who's?
- What's?
- Where's?
- When's?
- Why's?

30/03/20

Clara Sotillo, Anaïs, UAMC

10

Tipos de Contexto

- **Contexto Primario:** Se define como contexto primario al que se encuentra de manera inmediata a través de los sentidos humanos.
- **Contexto Secundario:** Se define como aquel que se infiere a través del contexto primario.

30/03/20

Clara Sotillo, Anaïs, UAMC

11

Ejemplo de contexto primario y secundario

Contexto Primario
Temperatura

Contexto Secundario

Dependiendo de la temperatura del agua el visualizador indica al adulto mayor el puede o no puede bañarse.

30/03/20

Clara Sotillo, Anaïs, UAMC

12

EVALUACIÓN PARTE 1

30/03/20

Olivero García, Andrea, UAMC

35

Retos de los Sistemas Conscientes del Contexto

- Representar el contexto capturado, interpretarlo y diseminar la información contextual entre los subsistemas de Aml.
- Inferir el contexto secundario a partir del contexto primario capturado.
- Crear infraestructuras para facilitar el desarrollo de Sistemas Conscientes del Contexto que asistan al adulto mayor en sus AVD.

30/03/20

Olivero García, Andrea, UAMC

36

Retos de los Sistemas Conscientes del Contexto

- Representar el contexto capturado, interpretarlo y diseminar la información contextual entre los subsistemas de Aml.
- Inferir el contexto secundario a partir del contexto primario capturado.
- Crear infraestructuras para facilitar el desarrollo de Sistemas Conscientes del Contexto que asistan al adulto mayor en sus AVD.

30/03/20

Olivero García, Andrea, UAMC

39

¿Por qué es importante la representación del contexto?

- La representación contextual es fundamentalmente una manera de permitir a una entidad, determinar consecuencias mediante el razonamiento en lugar de la acción.
- La representación es el concepto fundamental del razonamiento inteligente.
- Permitir generar una base de conocimientos adquiridas a través del tiempo.

30/03/20

Olivero García, Andrea, UAMC

39

Técnicas para la representación del contexto

- Algunas técnicas de representación contexto son:
 - Lógica de predicados
 - XML (eXtensible Markup Language)
 - Ontologías

16/03/20

David Saldaña Encinas, UMBC

17

Lógica de predicados

- Introduce el concepto de cuantificadores, lo que va a permitir la referencia a un grupo o grupos de objetos. Los dos tipos de cuantificadores que admite la lógica de predicados son:
 - Cuantificador existencial $\exists$.
 - Cuantificador universal $\forall$.
- Con ayuda de estos cuantificadores se realizan las técnicas de resolución y unificación que forman la base de lenguajes como **PROLOG** y **LISP**.

16/03/20

David Saldaña Encinas, UMBC

18

Ejemplo de PROLOG

```
Action(openFaucet)
Features(temp,54)
Location(shower)

Rule(Action,[Feature, Value] Location,
Time, Risk, NotifyingAgent):- Action =
openFaucet, Features=temp=>54,
Location=shower, Risk=scalding,
NotifyingAgent=scaldingNotifyingAg
```

16/03/20

David Saldaña Encinas, UMBC

19

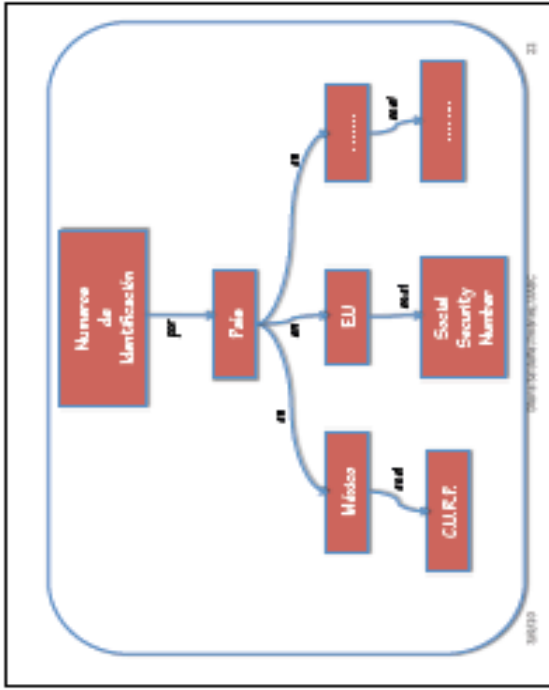
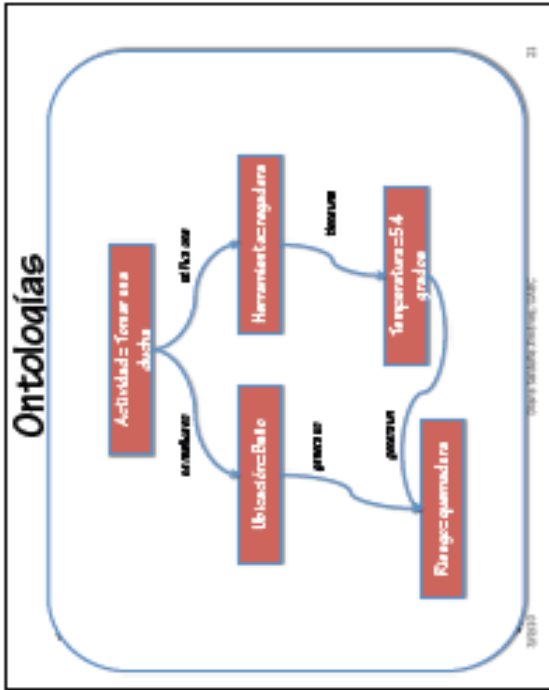
eXtensible Markup Language

```
<Actividad>Tomar_ducha</Actividad>
  <Ubicación>baño</Ubicación>
  <Herramienta>regadera</Herramienta>
<Características>54_gradosC</Características>
  <Persona>Adulto_Mayor</Persona>
<Notificación>ScaldingAgent</Notificación>
```

16/03/20

David Saldaña Encinas, UMBC

20



Ejemplos de Ontologías utilizadas en la Web

- En <http://www.snomed.org/> se puede encontrar una ontología médica (SNOMED).
- WordNet <http://wordnet.princeton.edu/perl/webwn> es una ontología lingüística en línea que organiza los nombres, verbos y adjetivos del idioma inglés en grupos de sinónimos.
- En <http://www.unspsc.org/> se describe una ontología de carácter general desarrollada por las Naciones Unidas (vocabulario para servicios y productos).

Definición Ontología

- "Una ontología es una **especificación explícita** de una **conceptualización**". [Thomas Gruber, 1993]
- "Una ontología es una base de datos que **describe** los **conceptos del ambiente** que nos rodea o de un **dominio en específico** y algunas de sus propiedades y relaciones". [Weigand, 1997]

Ontología Pizzas



SPR13

Onto y Semántica: Introducción, UAMC

25

Ontología News Paper (1)

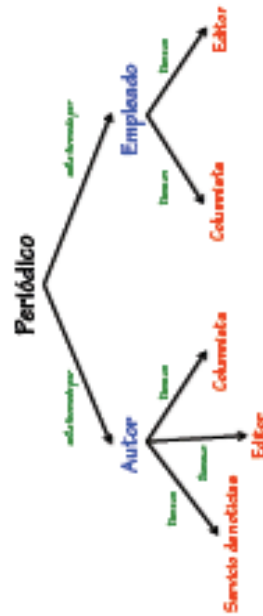
- ¿Quién es responsable de cada sección del periódico?
- ¿Qué es el contenido de cada artículo en una sección y quién es su autor?
- ¿Quién realiza el reporte al autor?
- ¿Cual es el costo por cada sección?

SPR13

Onto y Semántica: Introducción, UAMC

26

Ontología News Paper (2)



SPR13

Onto y Semántica: Introducción, UAMC

27

EVALUACIÓN PARTE 2

Apéndice E.

Presentación del tutorial de diseño de la arquitectura del Componente ELDeR.

CONTENIDO

- Retos de los Sistemas Conscientes del Contexto
- Componente consciente del contexto (ELDeR)
- Ontología CARE
- Reglas
- Evaluación

3

Diana Saldaña Jiménez, UAMC

Retos de los Sistemas Conscientes del Contexto

- Representar el contexto capturado, Interpretarlo y diseminar la información contextual entre los subsistemas de Aml.
- Inferir el contexto secundario a partir del contexto primario capturado.
- Crear infraestructuras para facilitar el desarrollo de Sistemas Conscientes del Contexto que asistan al adulto mayor en sus AVD.

4

Diana Saldaña Jiménez, UAMC



Universidad Autónoma de Baja California

Tutorial Diseño

Presentado por: Diana Saldaña Jiménez

Medicali, B.C., a 7 de Junio del 2010.

5

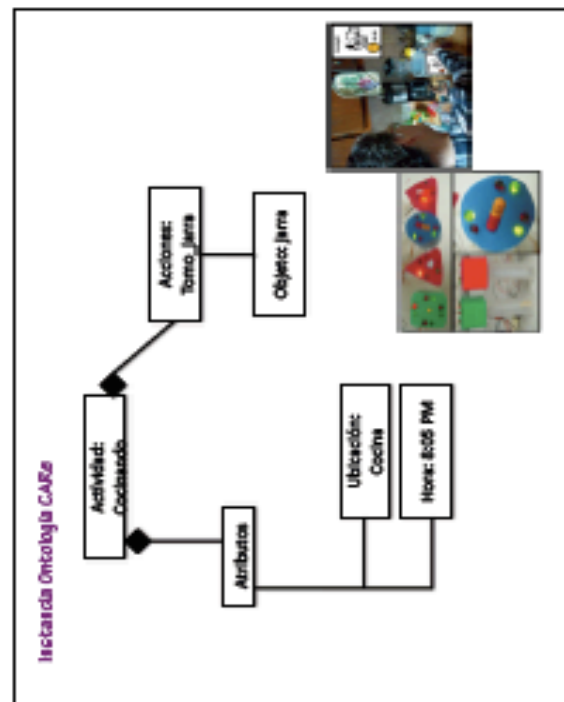
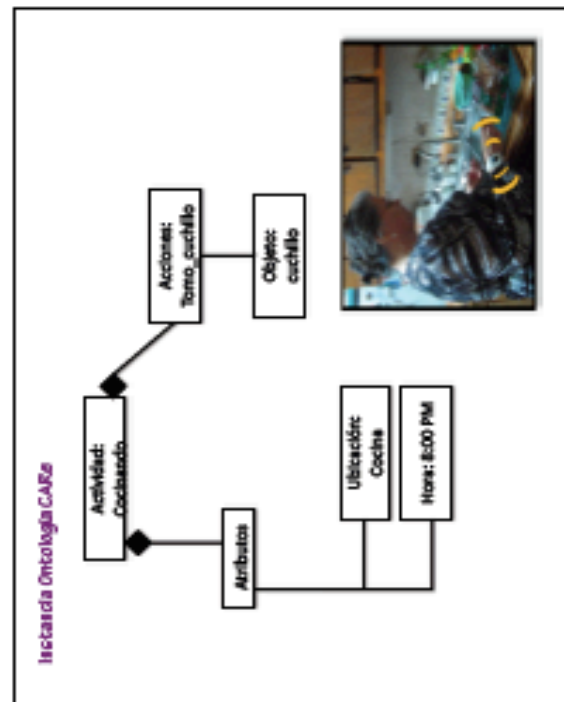
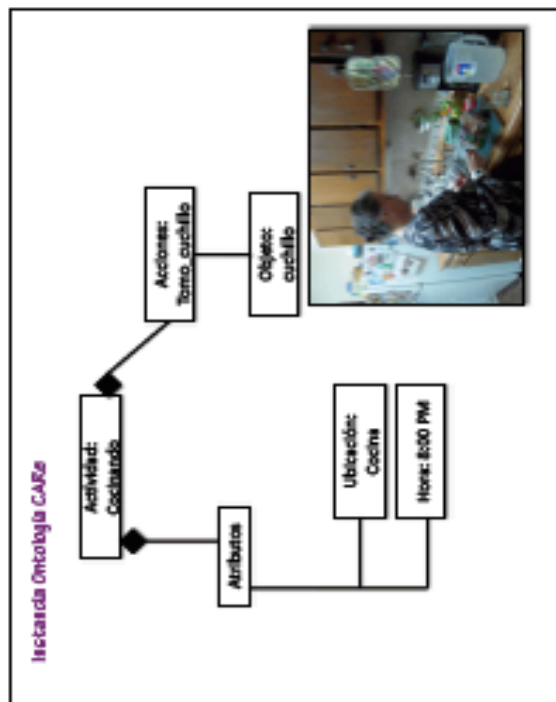
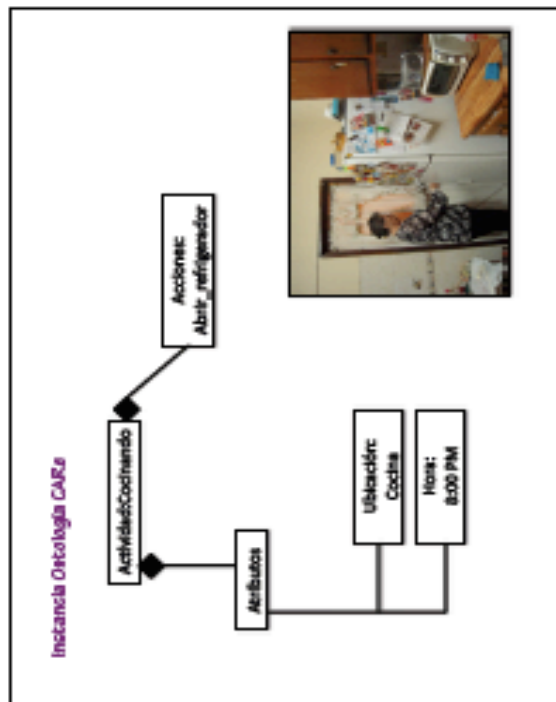
Diana Saldaña Jiménez, UAMC

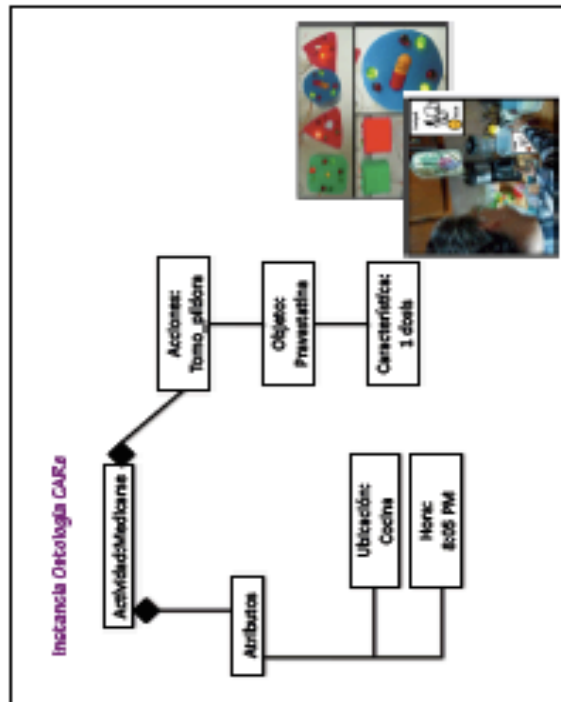
Ejemplo de un Ambiente Inteligente



6

Diana Saldaña Jiménez, UAMC





Tipos de Reglas en el Componente

- Donde unas definen el **comportamiento esperado** del **adulto mayor**.
 - Si la hora de la Medicación == nocturno y ubicación==cocina, entonces acción=tomar Pravastatina,dosis=1 pastilla
- Y las otras rigen el **comportamiento del Ambiente Inteligente**.
 - Si la hora actual==hora de la Medicación y ubicación== lugar Medicacion, entonces el sistema debe recordar mediante la pulsera que debe tomarse su medicamento.

Retos de los Sistemas Conscientes del Contexto

- Representar el contexto capturado, Interpretarlo y diseminar la información contextual entre los subsistemas de Aml.
- Inferir el contexto secundario a partir del contexto primario capturado.
- Crear Infraestructuras para facilitar el desarrollo de Sistemas Conscientes del Contexto que asistan al adulto mayor en sus AVD.

Otro ejemplo

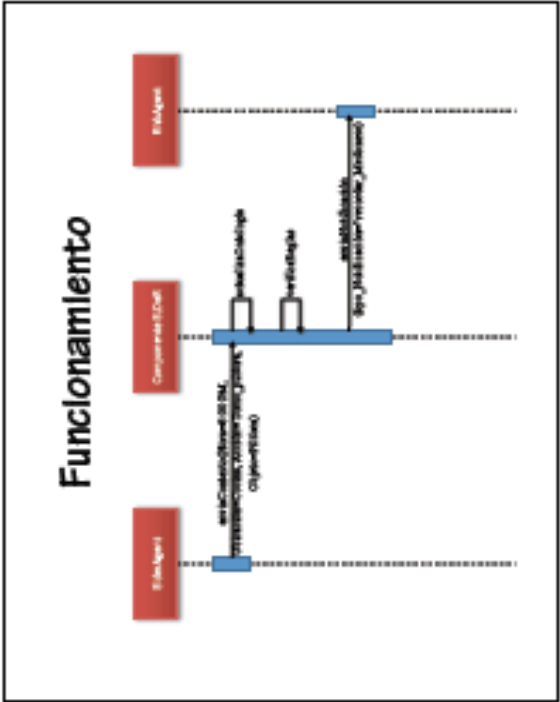
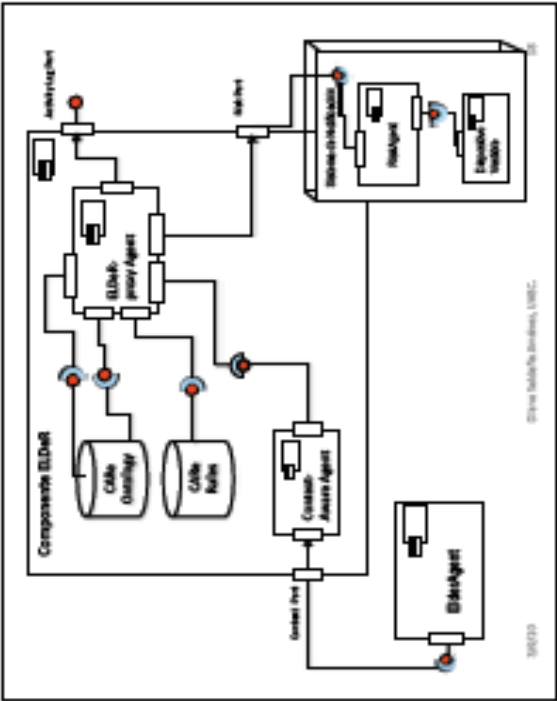
- Donde unas definen el comportamiento esperado del adulto mayor.
 - Si el horario de la Medicación == nocturno y ubicación==cocina y objeto == jarra, entonces acción=tomarJarra
- Y las otras rigen el comportamiento del Ambiente Inteligente.
 - Si la hora actual==hora de la Medicación y acción == tomarJarra, entonces el sistema debe asesorar al adulto mayor a tomar su medicamento.

30/03/20

Operatividad de Redes, UABC

37

AGENTES



30/03/20

Operatividad de Redes, UABC

40

EJERCICIO

Motivación del AmI

30/03/20

Diana Espinosa, Bachelors, UAMC

23

Escenario a evaluar



30/03/20

Diana Espinosa, Bachelors, UAMC

22

Escenario a evaluar (2)



30/03/20

Diana Espinosa, Bachelors, UAMC

25

Objetivo de la Evaluación

- Diseñar un escenario de Ambiente Inteligente, utilizando al Componente ELDeR como herramienta para representar contexto, inferir contexto secundario y notificar acciones al adulto mayor.
 - Realizar las instancias posibles en la Ontología CARE en las acciones que realiza el Adulto Mayor en el escenario.
 - Diseñar las reglas generales y particulares en el ambiente.
 - Decidir que tipos de agentes de comunicación utiliza el componente para poder ejecutar el escenario dado.

30/03/20

Diana Espinosa, Bachelors, UAMC

24

Apéndice F.

Presentación de las respuestas del examen de diseño.



Universidad Autónoma de Baja California

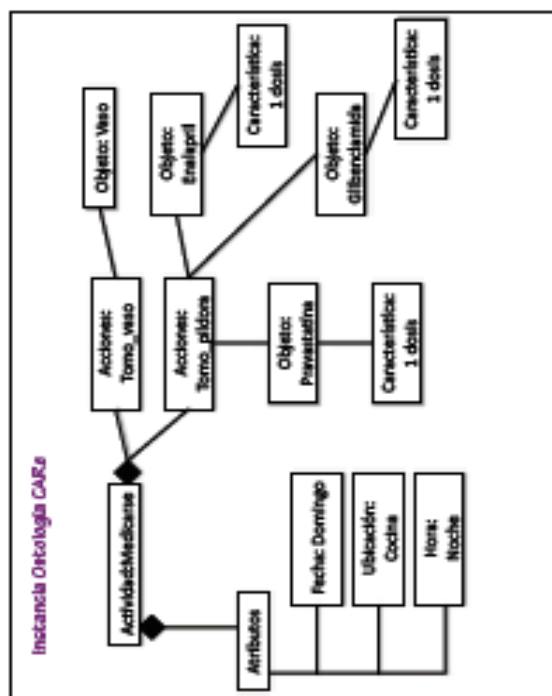
Respuestas Práctica Diseño

Presentado por: Diana Saldana Jiménez

México, B.C., a 7 de Junio del 2010.

Escenario 1a) Subrayando el contexto del adulto mayor...

- Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción médica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.



Escenario 1a) Subrayando el contexto del adulto mayor...

- Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro, tal como se muestra en la figura 1. De repente escucha cantar a los periquitos del sistema CARE-Me, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción médica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos (como se muestra en figura 2) y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.

Escenario 1a) Subrayando el contexto del adulto mayor...

- Hoy es domingo por la noche, y como es costumbre, la Sra. Ana está en su sala disfrutando de su café y leyendo un libro, tal como se muestra en la figura 1. De repente escucha **cantar a los pajaritos del sistema CAPE-Mé, lo cual le recuerda que necesita tomarse su medicina, ya que la prescripción médica indica que es momento de tomar una dosis de Pravastatina, una dosis de Enalapril y una dosis de Glibenclamida. Así que la Sra. Ana se dirige a la cocina donde están sus medicamentos. Cuando Ana entra a la cocina decide primero servirse un vaso de agua y se acerca a su dispensador de medicamentos (como se muestra en la figura 2) y toma una dosis de Pravastatina, otra de Enalapril y otra de Glibenclamida.**

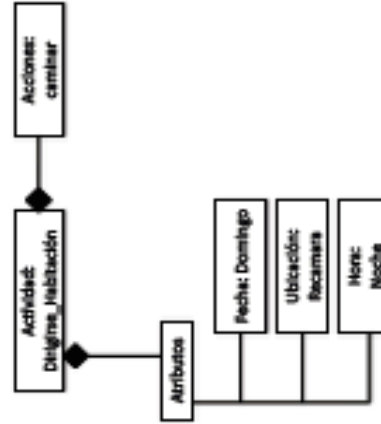
Regla Expresada

- Si hora == noche, entonces hora = hora de medicarse, pastilla = pravastatina, dosis = 1
- Si hora == noche, entonces hora = hora de medicarse, pastilla = glibenclamida, dosis = 1
- Si hora == noche, entonces hora = hora de medicarse, pastilla = enalapril, dosis = 1
- Si hora == hora _medicacion entonces recordarMedicarse

Escenario 1b) Subrayando el contexto del adulto mayor...

- La Sra Ana se dirige a su cuarto porque ya se terminó el día. El sistema está consciente de que Ana cumplió correctamente con la medicación del día de hoy, por lo que el perro que representa la semana actual aumenta de tamaño. Adicionalmente, como la semana ha terminado, aparece un nuevo perro para representar la semana que empieza.

Instancia Ontología CAPE



Escenario 1b) Subrayando el contexto del adulto mayor...

- La Sra Ana se dirige a su cuarto porque ya se terminó el día. El sistema esta consciente de que Ana cumplió correctamente con la medicación del día de hoy, por lo que el perico que representa la semana actual aumenta de tamaño. Adicionalmente, como la semana ha terminado, aparece un nuevo perico para representar la semana que empieza.

Regla expresada

- Si cumplió con medicación diaria, entonces perico crece.
- Si es el ultimo día de medicación, entonces perico que representa la semana actual vuela al tronco y nace uno nuevo.

Apéndice G.

Presentación del tutorial de implementación del Componente ELDeR.



Universidad Autónoma de Baja California

Tutorial Implementación

Presentado por: Diana Saldaña Jiménez

México, B.C., a 7 de Junio del 2010.

Retos de los Sistemas Conscientes del Contexto

- Representar el contexto capturado, interpretarlo y diseminar la información contextual entre los subsistemas de Aml.
- Inferir el contexto secundario a partir del contexto primario capturado.
- Crear Infraestructuras para facilitar el desarrollo de Sistemas Conscientes del Contexto que asistan al adulto mayor en sus AVD.

30/03/10 Diana Saldaña Jiménez, UABC 3

REGLAS EN JESS



Universidad Autónoma de Baja California

Tutorial Implementación

Presentado por: Diana Saldaña Jiménez

México, B.C., a 7 de Junio del 2010.

Pasos para utilizar el componente ELDeR

- 1.Implementar un agente que perciba el contexto sensed y lo envíe al componente.
- 2.Implementar las reglas.
- 3.Implementar un agente que asiste al adulto mayor de acuerdo a la información recibida del componente.

30/03/10 Diana Saldaña Jiménez, UABC 3



Universidad Autónoma de Baja California

Tutorial Implementación

Presentado por: Diana Saldaña Jiménez

México, B.C., a 7 de Junio del 2010.

Pasos para utilizar el componente ELDeR

- 1.Implementar un agente que perciba el contexto sensed y lo envíe al componente.
- 2.Implementar las reglas.
- 3.Implementar un agente que asiste al adulto mayor de acuerdo a la información recibida del componente.

30/03/10 Diana Saldaña Jiménez, UABC 3

Comando "Defemplate"

- Se utiliza para definir la estructura de un hecho (fact).

```
(defemplate Template
(slot Activity)
(slot Actions)
(slot Frequency)
(slot Duration))
```

Comando "Assert"

- ```
(assert (Template (Activity
Medicarse) (Actions tomar_pildora)
(Frequency 3) (solution No.2)))
```
- ```
(assert (Template (Activity caminar)
(Time morning) (Duration
12_minutos)))
```
- ```
(assert (nombre Diana) (edad 24)
(escolaridad licenciatura))
```

## Reglas en JESS

- ... están en la base de conocimiento del sistema
- ... se cumplen solo una vez
- ... utilizan las constantes de patrones para identificar la similitud de los hechos con las reglas
- ... son más rápidos que los enunciados de IF-THEN

## Sintaxis de las Reglas

- Las reglas se leen de izquierda a derecha
- El lado izquierdo contienen hechos que concuerdan con ciertos patrones
- El lado derecho contiene llamadas a funciones

## Comando "defrule"

- (defrule report-error (error-is-present) => (printout t "There is an error" crlf))

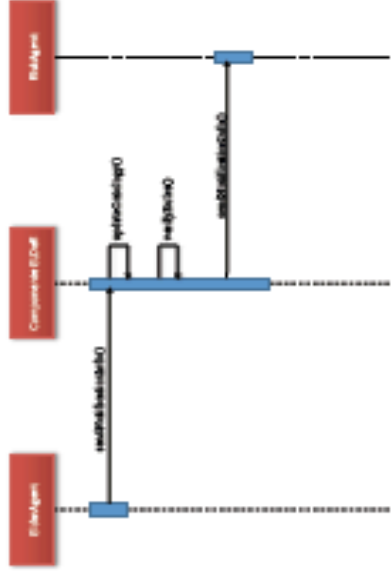
## Reglas en JESS del ejercicio anterior

- (defrule getPrescriptionMoon (Template {Time == noon}) => (assert (Location kitchen) (Tools Glibesclanida) (Features 1) (Frequency 3) (medicationTime noon) (healthProblem diabetes)) (bind ?done 3))
- (defrule provideReminderCARE-Me (Template {medicationTime == morning || medicationTime == afternoon || medicationTime == noon}) => (assert (agentRisk riskagent@localhost) (action recordar)))

## Regla en JESS

- (defrule growParrot (Template {Frequency ?dose}) => (assert (agentRisk riskagent@localhost) (action crecer)))
- (defrule getDate (Template {Date == 06-06-2010}) => (assert {Date 07-06-2010}) (bind ?lastDate 06-06-2010))
- (defrule flyParrot (Template {Date ?lastDate}) => (assert (agentRisk riskagent@localhost) (action volar)))

## Funcionamiento





## Acción RiskAgent

Código Java se ejecuta la acción que reducen este agente, si se cumple una condición.

público clase RiskAction extiende Action {

public Object execute() {

```
PeriodoCorta pc = new PeriodoCorta();
try {
 pc.execute();
} catch (MalformacionExcepcion ex) {
 ex.printStackTrace();
}
return null;
}
}
```

## Cuerpo agente RiskAgent

Código necesario para iniciar el agente. Dónde se especifica el servidor, que tipo de cliente utiliza y los parámetros de comunicación.

público clase RiskAgent extiende Agent {

public Client get

```
public RiskAgent() {
 attributes.putAttribute("riskAgentLocalhost");
 jmsnew SubscriberClient("localhost", "5222", "RiskAgent", "1234");
 RiskAgentEncoding encoding = new RiskAgentEncoding();
 activate(encoding, get);
}
// public static void main(String[] args) {
// TODO Auto-generated method stub
//
// RiskAgent ra = new RiskAgent();
//
// }
```

## Apéndice H.

Esta sección presenta las clases y métodos utilizados para la implementación del *Componente ELDeR*. Incluye la parte de los agentes del componente y la otra es la parte que se programo para poder comunicarse con la ontología y las reglas.

contextAwareAgent

### Class ContextAwareAction

```
java.lang.Object
|
+--SALSA.Action
|
+--contextAwareAgent.ContextAwareAction
```

---

```
public class ContextAwareAction
extends SALSA.Action
```

---

#### Constructor Summary

[ContextAwareAction](#)(java.lang.Object ob1, java.lang.Object ob2, java.lang.Object ob3, java.lang.Object ob4)

#### Method Summary

|                  |                            |
|------------------|----------------------------|
| java.lang.Object | <a href="#">execute</a> () |
|------------------|----------------------------|

#### Methods inherited from class SALSA.Action

getComponent, sendCommandRequest, sendContextualMsg, sendDataSensor, sendDataSensor, sendMessage, sendMessage, sendNotificationInfo, sendPresence, sendRequest, sendResponse

### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### ContextAwareAction

```
public ContextAwareAction(java.lang.Object ob1,
 java.lang.Object ob2,
 java.lang.Object ob3,
 java.lang.Object ob4)
```

## Method Detail

### execute

```
public java.lang.Object execute()
```

#### Specified by:

execute in class SALSA.Action

contextAwareAgent

## Class ContextAwareAgent

```
java.lang.Object
|
+--SALSA.Agent
|
+--contextAwareAgent.ContextAwareAgent
```

---

```
public class ContextAwareAgent
extends SALSA.Agent
```

---

### Field Summary

#### Fields inherited from class SALSA.Agent

acting, attributes, jc, passivePerception, reasoning

### Constructor Summary

[ContextAwareAgent](#)()

### Method Summary

|             |                                                |
|-------------|------------------------------------------------|
| static void | <a href="#">main</a> (java.lang.String[] args) |
|-------------|------------------------------------------------|

#### Methods inherited from class SALSA.Agent

activate, activate, activate, setAttributes



### Methods inherited from class `java.lang.Object`

`clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`, `wait`, `wait`

## Constructor Detail

### `ContextAwareAgent`

```
public ContextAwareAgent()
```

## Method Detail

### `main`

```
public static void main(java.lang.String[] args)
```

contextAwareAgent

## Class ContextAwareReasoning

```
java.lang.Object
|
+--SALSA.Reasoning
|
+--contextAwareAgent.ContextAwareReasoning
```

---

```
public class ContextAwareReasoning
extends SALSA.Reasoning
```

---

### Field Summary

#### Fields inherited from class SALSA.Reasoning

acting

### Constructor Summary

[ContextAwareReasoning](#) ()

### Method Summary

|                   |                                                  |
|-------------------|--------------------------------------------------|
| protected<br>void | <a href="#">think</a> (java.util.EventObject ev) |
|-------------------|--------------------------------------------------|

#### Methods inherited from class SALSA.Reasoning

setActing

### Methods inherited from class `java.lang.Object`

`clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`, `wait`, `wait`

## Constructor Detail

### `ContextAwareReasoning`

```
public ContextAwareReasoning()
```

## Method Detail

### `think`

```
protected void think(java.util.EventObject ev)
```

**Specified by:**

`think` in class `SALSA.Reasoning`

elderProxyAgent

## Class ELDeRproxyAction

```
java.lang.Object
|
+--SALSA.Action
|
+--elderProxyAgent.ELDeRproxyAction
```

public class **ELDeRproxyAction**  
extends SALSA.Action

### Constructor Summary

[ELDeRproxyAction](#)(java.lang.Object ob1, java.lang.Object ob2,  
java.lang.Object ob3, java.lang.Object ob4)

[ELDeRproxyAction](#)(java.lang.Object ob1, java.lang.Object ob2,  
java.lang.Object ob3, java.lang.Object ob4, java.lang.Object ob5)

### Method Summary

|                  |                            |
|------------------|----------------------------|
| java.lang.Object | <a href="#">execute</a> () |
|------------------|----------------------------|

#### Methods inherited from class SALSA.Action

getComponent, sendCommandRequest, sendContextualMsg, sendDataSensor,  
sendDataSensor, sendMessage, sendMessage, sendNotificationInfo,  
sendPresence, sendRequest, sendResponse

#### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString,  
wait, wait, wait

## Constructor Detail

### ELDeRproxyAction

```
public ELDeRproxyAction(java.lang.Object ob1,
 java.lang.Object ob2,
 java.lang.Object ob3,
 java.lang.Object ob4)
```

---

### ELDeRproxyAction

```
public ELDeRproxyAction(java.lang.Object ob1,
 java.lang.Object ob2,
 java.lang.Object ob3,
 java.lang.Object ob4,
 java.lang.Object ob5)
```

## Method Detail

### execute

```
public java.lang.Object execute()
```

**Specified by:**

execute in class SALSA.Action

elderProxyAgent

## Class ELDeRproxyAgent

```
java.lang.Object
|
+--SALSA.Agent
|
+--elderProxyAgent.ELDeRproxyAgent
```

---

```
public class ELDeRproxyAgent
extends SALSA.Agent
```

---

### Field Summary

#### Fields inherited from class SALSA.Agent

acting, attributes, jc, passivePerception, reasoning

### Constructor Summary

[ELDeRproxyAgent](#) ()

### Method Summary

|             |                                                |
|-------------|------------------------------------------------|
| static void | <a href="#">main</a> (java.lang.String[] args) |
|-------------|------------------------------------------------|

#### Methods inherited from class SALSA.Agent

activate, activate, activate, setAttributes

### Methods inherited from class `java.lang.Object`

`clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`, `wait`, `wait`

## Constructor Detail

### **ELDeRproxyAgent**

```
public ELDeRproxyAgent()
```

## Method Detail

### **main**

```
public static void main(java.lang.String[] args)
```

elderProxyAgent

## Class ELDeRproxyReasoning

```
java.lang.Object
|
+--SALSA.Reasoning
|
+--elderProxyAgent.ELDeRproxyReasoning
```

---

```
public class ELDeRproxyReasoning
extends SALSA.Reasoning
```

---

### Field Summary

#### Fields inherited from class SALSA.Reasoning

acting

### Constructor Summary

[ELDeRproxyReasoning](#) ()

### Method Summary

|                   |                                                  |
|-------------------|--------------------------------------------------|
| protected<br>void | <a href="#">think</a> (java.util.EventObject ev) |
|-------------------|--------------------------------------------------|

#### Methods inherited from class SALSA.Reasoning

setActing



### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### ELDeRproxyReasoning

```
public ELDeRproxyReasoning()
```

## Method Detail

### think

```
protected void think(java.util.EventObject ev)
```

**Specified by:**

think in class SALSA.Reasoning

salsa

## Class ContextualInformation

java.lang.Object

|

+-salsa.ContextualInformation

---

public class **ContextualInformation**  
extends java.lang.Object

---

### Constructor Summary

[ContextualInformation](#) ()

### Method Summary

|                  |                                               |
|------------------|-----------------------------------------------|
| java.lang.String | <a href="#"><u>getActions</u></a> ()          |
| java.lang.String | <a href="#"><u>getActivity</u></a> ()         |
| java.lang.String | <a href="#"><u>getAgentName</u></a> ()        |
| java.lang.String | <a href="#"><u>getCommunity</u></a> ()        |
| java.lang.String | <a href="#"><u>getDate</u></a> ()             |
| java.lang.String | <a href="#"><u>getDuration</u></a> ()         |
| java.lang.String | <a href="#"><u>getFeatures</u></a> ()         |
| java.lang.String | <a href="#"><u>getFrequency</u></a> ()        |
| java.lang.String | <a href="#"><u>getGeneralRulesPath</u></a> () |
| java.lang.String | <a href="#"><u>getLocation</u></a> ()         |
| java.lang.String | <a href="#"><u>getMembers</u></a> ()          |

|                  |                                                                                |
|------------------|--------------------------------------------------------------------------------|
| java.lang.String | <a href="#"><u>getOntologyPath</u></a> ()                                      |
| java.lang.String | <a href="#"><u>getParticularRulesPath</u></a> ()                               |
| java.lang.String | <a href="#"><u>getRole</u></a> ()                                              |
| java.lang.String | <a href="#"><u>getStatus</u></a> ()                                            |
| java.lang.String | <a href="#"><u>getTime</u></a> ()                                              |
| java.lang.String | <a href="#"><u>getTools</u></a> ()                                             |
| java.lang.String | <a href="#"><u>getTypeofADL</u></a> ()                                         |
| java.lang.String | <a href="#"><u>requestPaths</u></a> ()                                         |
| void             | <a href="#"><u>setActions</u></a> (java.lang.String actions)                   |
| void             | <a href="#"><u>setActivity</u></a> (java.lang.String activity)                 |
| void             | <a href="#"><u>setAgentName</u></a> (java.lang.String agentName)               |
| void             | <a href="#"><u>setCommunity</u></a> (java.lang.String community)               |
| void             | <a href="#"><u>setDate</u></a> (java.lang.String date)                         |
| void             | <a href="#"><u>setDuration</u></a> (java.lang.String duration)                 |
| void             | <a href="#"><u>setFeatures</u></a> (java.lang.String features)                 |
| void             | <a href="#"><u>setFrequency</u></a> (java.lang.String frequency)               |
| void             | <a href="#"><u>setGeneralRulesPath</u></a> (java.lang.String generalRulesPath) |
| void             | <a href="#"><u>setLocation</u></a> (java.lang.String location)                 |

|                  |                                                                                      |
|------------------|--------------------------------------------------------------------------------------|
| void             | <a href="#"><u>setMembers</u></a> (java.lang.String members)                         |
| void             | <a href="#"><u>setOntologyPath</u></a> (java.lang.String ontologyPath)               |
| void             | <a href="#"><u>setParticularRulesPath</u></a> (java.lang.String particularRulesPath) |
| void             | <a href="#"><u>setRole</u></a> (java.lang.String role)                               |
| void             | <a href="#"><u>setStatus</u></a> (java.lang.String status)                           |
| void             | <a href="#"><u>setTime</u></a> (java.lang.String time)                               |
| void             | <a href="#"><u>setTools</u></a> (java.lang.String tools)                             |
| void             | <a href="#"><u>setTypeofADL</u></a> (java.lang.String typeofADL)                     |
| java.lang.String | <a href="#"><u>xmlMessageFactory</u></a> ()                                          |

#### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### ContextualInformation

```
public ContextualInformation()
```

## Method Detail

### xmlMessageFactory

```
public java.lang.String xmlMessageFactory()
```

---

## **requestPaths**

```
public java.lang.String requestPaths()
```

---

## **getActivity**

```
public java.lang.String getActivity()
```

### **Returns:**

the activity

---

## **setActivity**

```
public void setActivity(java.lang.String activity)
```

### **Parameters:**

activity - the activity to set

---

## **getTime**

```
public java.lang.String getTime()
```

### **Returns:**

the time

---

## **setTime**

```
public void setTime(java.lang.String time)
```

### **Parameters:**

time - the time to set

---

## **getActions**

```
public java.lang.String getActions()
```

### **Returns:**

the actions

---

## **setActions**

```
public void setActions(java.lang.String actions)
```

### **Parameters:**

actions - the actions to set

---

## **getTools**

```
public java.lang.String getTools()
```

**Returns:**  
the tools

---

## **setTools**

```
public void setTools(java.lang.String tools)
```

**Parameters:**  
tools - the tools to set

---

## **getFeatures**

```
public java.lang.String getFeatures()
```

**Returns:**  
the features

---

## **setFeatures**

```
public void setFeatures(java.lang.String features)
```

**Parameters:**  
features - the features to set

---

## **getLocation**

```
public java.lang.String getLocation()
```

**Returns:**  
the location

---

## **setLocation**

```
public void setLocation(java.lang.String location)
```

**Parameters:**  
location - the location to set

---

## **getCommunity**

```
public java.lang.String getCommunity()
```

**Returns:**  
the community

---

### **setCommunity**

```
public void setCommunity(java.lang.String community)
```

**Parameters:**  
community - the community to set

---

### **getRole**

```
public java.lang.String getRole()
```

**Returns:**  
the role

---

### **setRole**

```
public void setRole(java.lang.String role)
```

**Parameters:**  
role - the role to set

---

### **getMembers**

```
public java.lang.String getMembers()
```

**Returns:**  
the members

---

### **setMembers**

```
public void setMembers(java.lang.String members)
```

**Parameters:**  
members - the members to set

---

### **getDuration**

```
public java.lang.String getDuration()
```

**Returns:**  
the duration

---

## **setDuration**

```
public void setDuration(java.lang.String duration)
```

### **Parameters:**

duration - the duration to set

---

## **getTypeofADL**

```
public java.lang.String getTypeofADL()
```

### **Returns:**

the typeofADL

---

## **setTypeofADL**

```
public void setTypeofADL(java.lang.String typeofADL)
```

### **Parameters:**

typeofADL - the typeofADL to set

---

## **getDate**

```
public java.lang.String getDate()
```

### **Returns:**

the date

---

## **setDate**

```
public void setDate(java.lang.String date)
```

### **Parameters:**

date - the date to set

---

## **getFrequency**

```
public java.lang.String getFrequency()
```

### **Returns:**

the frequency

---



## **setFrequency**

```
public void setFrequency(java.lang.String frequency)
```

### **Parameters:**

frequency - the frequency to set

---

## **getStatus**

```
public java.lang.String getStatus()
```

### **Returns:**

the status

---

## **setStatus**

```
public void setStatus(java.lang.String status)
```

### **Parameters:**

status - the status to set

---

## **getAgentName**

```
public java.lang.String getAgentName()
```

### **Returns:**

the agentName

---

## **setAgentName**

```
public void setAgentName(java.lang.String agentName)
```

### **Parameters:**

agentName - the agentName to set

---

## **getOntologyPath**

```
public java.lang.String getOntologyPath()
```

### **Returns:**

the ontologyPath

---

## **setOntologyPath**

```
public void setOntologyPath(java.lang.String ontologyPath)
```

**Parameters:**

ontologyPath - the ontologyPath to set

---

**getGeneralRulesPath**

```
public java.lang.String getGeneralRulesPath()
```

**Returns:**

the generalRulesPath

---

**setGeneralRulesPath**

```
public void setGeneralRulesPath(java.lang.String generalRulesPath)
```

**Parameters:**

generalRulesPath - the generalRulesPath to set

---

**getParticularRulesPath**

```
public java.lang.String getParticularRulesPath()
```

**Returns:**

the particularRulesPath

---

**setParticularRulesPath**

```
public void setParticularRulesPath(java.lang.String particularRulesPath)
```

**Parameters:**

particularRulesPath - the particularRulesPath to set

salsa

## Class ManageActivity

```
java.lang.Object
|
+--SALSA.Action
|
+--salsa.ManageActivity
```

---

```
public class ManageActivity
extends SALSA.Action
```

---

### Constructor Summary

[ManageActivity](#)(java.lang.String agent)

### Method Summary

|                  |                                                                                                                                       |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| java.lang.Object | <a href="#"><u>execute</u></a> ()                                                                                                     |
| void             | <a href="#"><u>sendPresenceActivityAttributes</u></a> (java.lang.String activity, java.lang.String activityAttributes)                |
| void             | <a href="#"><u>sendPresenceUsersActivity</u></a> (java.lang.String person, java.lang.String activity, java.lang.String activityState) |
| void             | <a href="#"><u>sendUpdateActivity</u></a> (java.lang.String activity, java.lang.String person)                                        |

### Methods inherited from class SALSA.Action

getComponent, sendCommandRequest, sendContextualMsg, sendDataSensor, sendDataSensor, sendMessage, sendMessage, sendNotificationInfo, sendPresence, sendRequest, sendResponse

## Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### ManageActivity

```
public ManageActivity(java.lang.String agent)
```

## Method Detail

### sendUpdateActivity

```
public void sendUpdateActivity(java.lang.String activity,
 java.lang.String person)
```

---

### sendPresenceActivityAttributes

```
public void sendPresenceActivityAttributes(java.lang.String activity,
java.lang.String activityAttributes)
```

---

### sendPresenceUsersActivity

```
public void sendPresenceUsersActivity(java.lang.String person,
 java.lang.String activity,
 java.lang.String activityState)
```

---

### execute

```
public java.lang.Object execute()
```

#### Specified by:

execute in class SALSA.Action

salsa

## Class Ontology

```
java.lang.Object
|
+--salsa.Ontology
```

---

```
public class Ontology
extends java.lang.Object
```

---

### Constructor Summary

[Ontology](#)(java.lang.String ontologyPath)

### Method Summary

|                  |                                                                                                     |
|------------------|-----------------------------------------------------------------------------------------------------|
| java.lang.String | <a href="#"><u>requestByActivity Date</u></a> (java.lang.String date,<br>java.lang.String activity) |
| java.lang.String | <a href="#"><u>requestByDate</u></a> (java.lang.String date)                                        |
| java.lang.String | <a href="#"><u>requestOntology</u></a> ()                                                           |
| void             | <a href="#"><u>updateOntology</u></a> (java.lang.String xmlMessage)                                 |
| void             | <a href="#"><u>updateRisk</u></a> (java.lang.String risk)                                           |

### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Detail

### Ontology

```
public Ontology(java.lang.String ontologyPath)
```

## Method Detail

### requestOntology

```
public java.lang.String requestOntology()
 throws java.io.IOException
 java.io.IOException
```

---

### updateOntology

```
public void updateOntology(java.lang.String xmlMessage)
 throws java.io.IOException
 java.io.IOException
```

---

### updateRisk

```
public void updateRisk(java.lang.String risk)
 throws java.io.FileNotFoundException,
 java.io.IOException
 java.io.FileNotFoundException
 java.io.IOException
```

---

### requestByDate

```
public java.lang.String requestByDate(java.lang.String date)
 throws java.io.FileNotFoundException,
 java.io.IOException
 java.io.FileNotFoundException
 java.io.IOException
```

---

### requestByActivity\_Date

```
public java.lang.String requestByActivity_Date(java.lang.String date,
 java.lang.String activity)
 throws
 java.io.FileNotFoundException,
 java.io.IOException
 java.io.FileNotFoundException
 java.io.IOException
```

salsa

## Class RulesBase

```
java.lang.Object
|
+--salsa.RulesBase
```

---

```
public class RulesBase
extends java.lang.Object
```

---

### Constructor Summary

[RulesBase](#)(java.lang.String generalRulesPath,  
java.lang.String particularRulesPath, java.lang.String xmlMessage)

### Method Summary

|                  |                                |
|------------------|--------------------------------|
| java.lang.String | <a href="#">verifyRules</a> () |
|------------------|--------------------------------|

### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

### Constructor Detail

#### RulesBase

```
public RulesBase(java.lang.String generalRulesPath,
 java.lang.String particularRulesPath,
 java.lang.String xmlMessage)
```

## Method Detail

### **verifyRules**

```
public java.lang.String verifyRules()
 throws jess.JessException
 jess.JessException
```