



**UNIVERSIDAD
NACIONAL
DE LOJA**



Área de la energía, las industrias y los recursos naturales no renovables

CARRERA DE INGENIERÍA EN SISTEMAS

**“DESARROLLO DE UN AGENTE INTELIGENTE
PARA DISPOSITIVOS MÓVILES ANDROID QUE
PERMITA LLEVAR LOS INGRESOS Y EGRESOS
ECONÓMICOS DE UNA PERSONA”**

Tesis de Grado previo a la
obtención del título de
Ingeniería en Sistemas

Autor:

John Patricio Solano Cabrera

Tutor:

Ing. Henry Patricio Paz Arias.

Loja-Ecuador

2015

CERTIFICACIÓN DEL DIRECTOR

Mg. Sc. Henry Patricio Paz Arias

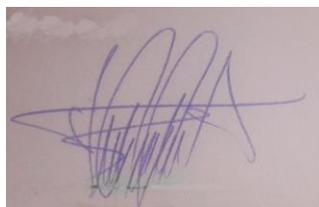
DOCENTE DE LA CARRERA DE INGENIERÍA EN SISTEMAS, DEL ÁREA DE LA ENERGÍA, LAS INDUSTRIAS Y LOS RECURSOS NATURALES NO RENOVABLES DE LA UNIVERSIDAD NACIONAL DE LOJA.

CERTIFICA:

Haber asesorado y revisado detenida y minuciosamente durante todo su desarrollo, el Proyecto de Fin de Carrera titulado: “**DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS MÓVILES ANDROID QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS ECONÓMICOS DE UNA PERSONA.**” . Realizado por el postulante John Patricio Solano Cabrera.

Por lo tanto, autorizo proseguir los trámites legales pertinentes para su presentación y defensa.

Jueves 18 de Diciembre de 2014



Mg.Sc. Henry Patricio Paz Arias

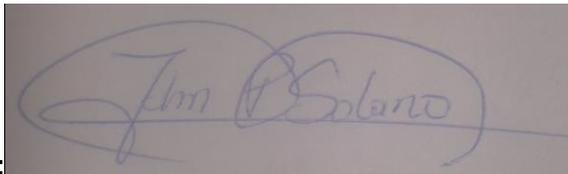
DIRECTOR DEL PFC

AUTORÍA

Yo **JOHN PATRICIO SOLANO CABRERA** declaro ser autor del presente trabajo de tesis y eximo expresamente a la Universidad Nacional de Loja y sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi tesis en el Repositorio Institucional- Biblioteca Virtual.

Firma:

A rectangular box containing a handwritten signature in blue ink. The signature is cursive and appears to read "John Patricio Solano".

Cédula: 1104268899

Fecha: 20/07/2015

CARTA DE AUTORIZACIÓN

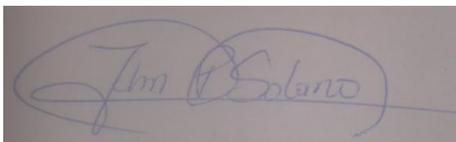
Yo **JOHN PATRICIO SOLANO CABRERA**, declaro ser autor de la tesis titulada: **“DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS MÓVILES ANDROID QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS ECONÓMICOS DE UNA PERSONA”**, como requisito para optar al grado de: **INGENIERO EN SISTEMAS**; autorizo al Sistema de Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional:

Los usuarios pueden consultar el contenido de este trabajo de RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja a los veinte días del mes de julio del dos mil quince.

Firma:



Autor: John Patricio Solano Cabrera

Cédula: 1104268899

Dirección: Loja (Daniel alvarez: Av. Benjamin Carrión y Francisco de miranda)

Correo Electrónico: jpsolanoc@unl.edu.ec

Teléfono: 072587392

Celular: 0992118974

DATOS COMPLEMENTARIOS

Director de Tesis: Ing. Henry Patricio Paz Arias, Mg.Sc.

Tribunal de Grado: Ing. Luis Roberto Jácome Galarza, Mg. Sc.

Ing. Mario Andrés Palma Jaramillo, Mg. Sc.

Ing. Carlos Miguel Jaramillo Castro. Mg. Sc.

AGRADECIMIENTOS

Mi más profundo y sincero agradecimiento a dios por guiar y bendecir todo esfuerzo y dedicación en mi vida; A la Universidad Nacional de Loja, al Área de la energía, las industrias y los recursos naturales no renovables, a la carrera de Ingeniería en Sistemas, por darme la oportunidad de formarme como profesional; Al coordinador y distinguidos catedráticos por guiarme y compartir sus sabias enseñanzas en el transcurso de mi formación académica.

De manera especial al Mg. Sc. Henry Patricio Paz Arias quien con sus valiosos conocimientos y dedicación supo guiarme en el desarrollo del presente trabajo de titulación.

John Patricio Solano Cabrera.

DEDICATORIA

El presente trabajo de titulación está dedicado, en especial a Dios por darme la vida y salud para luchar en el camino del saber, a mis abnegados padres, Martha Judit Cabrera y José Manuel Solano mis tíos que con sus consejos han sabido orientarme por el camino del bien a mi hijo, Anthony Josué Solano que ha sido fuente de inspiración para poder culminar mis estudios.

A mis queridos hermanas Alexandra, gracias por haber fomentado en mí el deseo de superación y triunfo en la vida.

John Patricio Solano Cabrera

A. TÍTULO

“DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS MÓVILES ANDROID QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS ECONÓMICOS DE UNA PERSONA”

B. RESUMEN

En la actualidad muchos de nosotros nos vemos envueltos en dilemas económicos ya que no contamos con una cultura de ahorro, esto produce que los egresos sean mayores que los ingresos, por tal motivo desestabiliza la situación económica dentro de las familias, se debe mencionar además que las personas que llevan un cuidado personal en sus gastos consumen tiempo en realizarlo.

El rol de las nuevas tecnologías es ayudar y brindar un buen vivir para todas las personas, es por eso que este trabajo de titulación está orientado a dar solución y ayudar a controlar de forma eficiente nuestros ingresos y egresos económicos sugiriendo ideas de cómo ahorrar y aportando con análisis de datos que ayuden de manifiesto para saber las prioridades de los gastos.

Este trabajo de titulación se centra en agentes inteligentes en dispositivos móviles android con uso de la librería JADE-LEAP desarrollada en Java, y la implementación de algoritmos que ayuden a evaluar de una manera eficiente el presupuesto de una persona.

Para el desarrollo de la aplicación se utilizó la metodología XP Programación Extrema y para el desarrollo del agente se hizo uso de la metodología GAIA para agentes inteligentes deliberativos.

SUMMARY

Nowadays many of us are involved in economic dilemmas because we don't have a culture of savings, this causes that the expenses are greater than the income, therefore it destabilizes the economic situation within families, also should be mentioned that people who take personal care in their spending consume time doing this.

The role of new technologies is to help and provide a good life for all people that is why the current degree work is aimed at solving and help to control in an efficient way our income and expenses suggesting ideas on how to save and contributing to analyze data that helps to reveal the priorities of expenses.

The current degree work focuses on intelligent agents on Android mobile devices that use JADE-LEAP library developed in Java, and the implementation of algorithms that help to evaluate efficiently the budget of a person.

For the development of this application was used the XP Extreme Programming Methodology and for the development of the agent was used the Methodology GAIA for deliberative intelligent agents.

ÍNDICE DE CONTENIDOS

CERTIFICACIÓN DEL DIRECTOR	II
AUTORÍA	III
CARTA DE AUTORIZACIÓN.....	IV
AGRADECIMIENTOS	V
DEDICATORIA	VI
A. TÍTULO	VII
B. RESUMEN.....	VIII
SUMMARY	IX
ÍNDICE DE CONTENIDOS.....	X
ÍNDICE DE FIGURAS	XVI
ÍNDICE DE TABLAS	XX
C. INTRODUCCIÓN	1
D. REVISIÓN DE LITERATURA	3
CAPÍTULO 1. ECONOMÍA.....	3
1. INTRODUCCIÓN ECONOMÍA.....	3
1.1. LA ACTIVIDAD ECONÓMICA	3
1.1.1. ECONOMIZAR.....	3
1.1.2. OPTIMIZAR.....	3
1.1.3. LA RIQUEZA	4
1.1.4. LOS GASTOS FIJOS	5
1.1.5. LOS GASTOS VARIABLES	5
1.1.6. LOS GASTOS ESENCIALES	5
1.1.7. LOS GASTOS OPCIONALES	5

1.1.8.	INGRESO MARGINAL	6
1.1.9.	INGRESO MEDIO.....	7
1.1.10.	INGRESO DEL PRODUCTO MARGINAL	7
1.1.11.	CONSUMO RESPONSABLE E INTELIGENTE.....	8
1.1.12.	CONTROLAR EL GASTO	8
1.1.13.	AYUDA DE UN PROFESIONAL	8
1.1.14.	INVERTIR Y MEJORAR NUESTRO AHORRO	8
CAPÍTULO 2.	AGENTES	9
2.	INTRODUCCIÓN AGENTES.....	9
2.1.	CARACTERÍSTICAS DE LOS AGENTES.	10
2.1.1.	ADAPTABILIDAD	11
2.1.2.	FORMAS POSIBLES DE COMUNICACIÓN:.....	11
2.2.	CLASIFICACIÓN DE LOS AGENTES INTELIGENTES.	12
2.2.1.	AGENTES DE REFLEJOS SIMPLES.....	12
2.2.2.	AGENTES BIEN INFORMADOS DE TODO LO QUE PASA.....	13
2.2.3.	AGENTES BASADOS EN METAS	14
2.2.4.	AGENTES BASADOS EN LA UTILIDAD	15
2.2.5.	AGENTES DE INTERFAZ	16
2.2.6.	AGENTES DE INFORMACIÓN.	16
2.2.6.1.	AGENTES CONSEJEROS.....	17
2.2.6.2.	AGENTES DE NAVEGACIÓN.....	17
2.2.6.3.	AGENTES DE MONITOREO.....	17
2.2.6.4.	AGENTES RECOMENDACIÓN	17
2.3.	TIPOS DE AMBIENTES AI.....	17
2.3.1.	ACCESIBLES Y NO ACCESIBLES.	17

2.3.2.	DETERMINISTA Y NO DETERMINISTA	18
2.3.3.	EPISÓDICOS Y NO EPISÓDICOS.....	18
2.3.4.	ESTÁTICA Y DINÁMICA	18
2.3.5.	DISCRETOS Y CONTINUOS.....	19
2.4.	METODOLOGÍAS ORIENTADAS A AGENTES.....	19
2.4.1.	METODOLOGÍA TROPOS.....	20
2.4.1.1.	REQUERIMIENTOS TEMPRANOS	20
2.4.1.2.	REQUERIMIENTOS TARDÍOS.....	20
2.4.1.3.	DISEÑO DE ARQUITECTURA	21
2.4.1.4.	DISEÑO DETALLADO	21
2.4.1.5.	IMPLEMENTACIÓN.....	21
2.4.2.	METODOLOGÍA INGENIAS.	21
2.4.3.	METODOLOGÍA GAIA.....	22
2.4.3.1.	INTRODUCCIÓN.	22
2.4.3.2.	DESCRIPCIÓN.	23
2.4.4.	PROCESOS PARA HACER LOS MODELOS.	25
2.4.4.1.	PARA HACER MODELOS EN GAIA REQUIERE DE UN CONJUNTO DE ETAPAS QUE SE MENCIONA A CONTINUACIÓN.	25
CAPÍTULO 3. ANDROID		26
3.	INTRODUCCIÓN.	26
3.1.	HISTORIA DE ANDROID.....	26
3.2.	ESTRUCTURA DE ANDROID	27
3.2.1.	CAPA DEL KERNEL (ROJA).....	28
3.2.2.	CAPA LIBRERÍAS (VERDE).....	28
3.2.3.	ANDROID RUNTIME (CAPA AMARILLA)	28

3.2.4.	APLICACIÓN FRAMEWORK (CAPA AZUL)	29
3.2.4.1.	APPLICATIONS (CAPA AZUL ÚLTIMA).....	29
3.3.	METODOLOGÍAS ÁGILES PARA DESARROLLO DE APLICACIONES MÓVILES.	30
3.3.1.	METODOLOGÍA SCRUM	30
3.3.2.	TEST DRIVEN DEVELOPMENT (TDD).....	31
3.3.3.	MOBILE-D.....	32
3.3.4.	PROGRAMACIÓN EXTREMA (XP).....	33
3.3.4.1.	INTRODUCCIÓN.	33
3.3.4.2.	HISTORIA	33
3.3.4.3.	OBJETIVOS.	34
E.	MATERIALES Y MÉTODOS.....	38
F.	RESULTADOS	43
1.	FASE UNO: ANÁLISIS DE REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES.	43
1.1.	ANÁLISIS DE REQUERIMIENTOS NO FUNCIONALES	43
1.1.1.	ANÁLISIS DE REQUERIMIENTOS FUNCIONALES.....	43
1.1.2.	REQUERIMIENTOS DE SOFTWARE.....	44
1.1.3.	REQUERIMIENTOS DE HARDWARE.	45
1.2.	DEFINICIÓN DE LAS ENTRADAS, ROLES, PROTOTIPOS ASOCIADOS PARA EL AGENTE. 45	
1.2.1.	MODELO DE ROL DEL AGENTE	45
1.2.2.	DEFINICIÓN DE LOS PROTOCOLOS.....	47
1.2.3.	MODELO DE SERVICIOS.....	48
1.3.	ANÁLISIS DE DATOS QUE VAN A SER ALMACENADOS SQLITE.	49
2.	FASE DOS: DISEÑO PARA LA APLICACIÓN Y EL AGENTE DELIBERATIVO.....	50
2.1.	PROTOTIPOS DE PANTALLA.....	50

2.2.	MODELO DE LA BASE DE DATOS.....	52
2.3.	DISEÑO DE CREENCIAS DESEOS E INTENCIONES DEL AGENTE DELIBERATIVO.....	53
2.3.1.	MODELO DEL AGENTE.....	53
2.3.2.	DISEÑO DE INTERACCIÓN.....	54
2.3.3.	ARQUITECTURA DEL AGENTE.	55
3.	FASE TRES: DESARROLLO DE APLICACIÓN E INTEGRACIÓN CON EL AGENTE.	56
3.1.	CODIFICACIÓN DE LA APLICACIÓN ANDROID.	56
3.1.1.	DETERMINACIÓN DE PAQUETES USADOS.	56
3.2.	IMPLEMENTACIÓN DEL AGENTE DELIBERATIVO PARA ANDROID.....	58
3.2.1.	INSTALACIÓN DE JADE-LEAP PARA ANDROID.....	58
3.2.2.	LEVANTAMIENTO DEL CONTENEDOR DE AGENTES.....	60
3.2.2.1.	LEVANTAMIENTO DEL CONTENEDOR PRINCIPAL.....	60
3.2.2.2.	LEVANTAMIENTO DEL CONTENEDOR SECUNDARIO.	64
3.2.3.	LEVANTAMIENTO DEL AGENTE DELIBERATIVO CONTABLE.	65
3.2.4.	FUNCIONAMIENTO DEL AGENTE.....	66
3.2.4.1.	IMPLEMENTACIÓN DE PERMISOS EN ANDROIDMANIFEST.....	66
3.2.4.2.	LEVANTAMIENTO DE SERVICIOS PARA EL CONTENEDOR DE AGENTES.....	66
3.2.5.	DEFINICIÓN DE LA CLASE AGENTE CONTABLE.	67
3.2.6.	DEFINICIÓN DE LAS DIFERENTES ACCIONES QUE REALIZARA EL AGENTE.	68
4.	FASE CUATRO: PRUEBAS DE FUNCIONALIDAD Y DE FACTIBILIDAD.	72
4.1.	PRUEBAS DE CÓDIGO.....	72
4.2.	PRUEBAS DE CARGA	76
4.3.	EJECUCIÓN EN UN EMULADOR APROPIADO.....	80
4.3.1.	PANTALLA DE BIENVENIDA.....	81
4.3.2.	PANTALLA DE ACTIVIDADES DE LA APLICACIÓN.	81

4.3.3.	LISTA DE INGRESOS Y LISTA DE EGRESOS	82
4.3.4.	AGREGAR NUEVOS ÍTEM A LA LISTA DE INGRESOS Y EGRESOS.....	82
4.3.5.	EJECUCIÓN DEL AGENTE	83
4.4.	PRUEBAS DE COMPATIBILIDAD.....	84
4.5.	EJECUCIÓN DE APLICACIÓN EN UN DISPOSITIVO FÍSICO QUE TENGA PLATAFORMA ANDROID.....	86
4.5.1.	PLAY STORE.....	87
4.5.2.	ESTADÍSTICAS PLAY STORE.....	88
4.5.3.	INSTALACIONES ACTUALES POR DISPOSITIVO.	88
G.	DISCUSIÓN	90
1.	DESARROLLO DE LA PROPUESTA ALTERNATIVA.....	90
2.	VALORACIÓN TÉCNICA – ECONÓMICA- SOCIAL.....	91
2.1.	VALORACIÓN TÉCNICA ECONÓMICA.....	91
2.2.	VALORACIÓN SOCIAL.....	94
H.	CONCLUSIONES.....	95
I.	RECOMENDACIONES	96
J.	BIBLIOGRAFÍA.....	97
K.	ANEXOS.....	100
	Anexo 01: Certificacion de traducción.....	100
	Anexo 02: Licencia Creative Commons del Proyecto.....	101
	Anexo 03: Artículo Científico.....	102

ÍNDICE DE FIGURAS

<i>Figura 1. Contabilidad y ahorro</i>	8
<i>Figura 2. Agentes inteligentes</i>	10
<i>Figura 3. Agentes de reflejo simple</i>	13
<i>Figura 4. Agentes bien informados</i>	14
<i>Figura 5. Agentes basados en metas</i>	15
<i>Figura 6. Agentes basados en la utilidad</i>	16
<i>Figura 7. AI Accesibles y no accesibles</i>	17
<i>Figura 8. AI Deterministas y no deterministas</i>	18
<i>Figura 9. AI episódicos y no episódicos</i>	18
<i>Figura 10. AI Estáticos y dinámicos</i>	18
<i>Figura 11. AI discretos y continuos</i>	19
<i>Figura 12. Metodologías Orientadas a Agentes Inteligentes</i>	19
<i>Figura 13. Acciones de la metodología GAIA</i>	23
<i>Figura 14. Definición de etapas GAIA</i>	24
<i>Figura 15. Estructura del SO Android</i>	27
<i>Figura 16. Metodología SCRUM</i>	31
<i>Figura 17. Metodología TDD</i>	32
<i>Figura 18. Pantalla entrada al sistema</i>	50
<i>Figura 19. Pantalla actividades</i>	50
<i>Figura 20. Pantalla de lista de Ingresos y Egresos</i>	51
<i>Figura 21. Pantalla de registro de Ingresos y Egresos</i>	51
<i>Figura 22. Pantalla de PRONÓSTICO</i>	52
<i>Figura 23. Modelo de la base de datos</i>	52
<i>Figura 24. Modelo del Agente Inteligente</i>	53

<i>Figura 25. Diseño de Interacción del Agente</i>	<i>54</i>
<i>Figura 26.Arquitectura del agente contable</i>	<i>55</i>
<i>Figura 27. Eclipse Luna</i>	<i>56</i>
<i>Figura 28. Determinación de paquetes entorno de desarrollo.....</i>	<i>57</i>
<i>Figura 29.Paquetes desglosados Contabilidad Inteligente.....</i>	<i>58</i>
<i>Figura 30. Página oficial JADE.....</i>	<i>59</i>
<i>Figura 31. Entorno de descargas add-ons.....</i>	<i>59</i>
<i>Figura 32. Implementación de la librería JADEANDROID.....</i>	<i>60</i>
<i>Figura 33. Entorno de descarga Jade</i>	<i>61</i>
<i>Figura 34.configuración del PATH en MAC OS YOSEMITE.....</i>	<i>62</i>
<i>Figura 35. Entorno Grafico JADE contenedor principal</i>	<i>63</i>
<i>Figura 36. Levantamiento del contenedor principal</i>	<i>63</i>
<i>Figura 37. Levantamiento del contenedor secundario.....</i>	<i>64</i>
<i>Figura 38. Puesta en marcha del contenedor.....</i>	<i>65</i>
<i>Figura 39. RuntimeCallBack para agentes Android.....</i>	<i>65</i>
<i>Figura 40. Pase de mensajes entre usuario y el agente.....</i>	<i>66</i>
<i>Figura 41. Permisos en ANDROIDMANIFEST</i>	<i>66</i>
<i>Figura 42. MircroRunTimeService servicio de jade android</i>	<i>67</i>
<i>Figura 43. Implementación del agente deliberativo</i>	<i>68</i>
<i>Figura 44. Acción nombre del agente.....</i>	<i>69</i>
<i>Figura 45. Acción gastos diario.</i>	<i>69</i>
<i>Figura 46. Acción como ahorro.....</i>	<i>70</i>
<i>Figura 47.Acción retorna categorías</i>	<i>70</i>
<i>Figura 48.Acción evaluación de gastos.....</i>	<i>70</i>
<i>Figura 49.Acción fin del presupuesto</i>	<i>71</i>

<i>Figura 50 Prueba Main Activiti</i>	72
<i>Figura 51 Prueba Actividades</i>	73
<i>Figura 52 Prueba Lista Ingresos</i>	74
<i>Figura 53 Prueba Lista Egresos</i>	74
<i>Figura 54 Prueba Ingreso Categoría</i>	75
<i>Figura 55 Prueba Egreso Categoría</i>	75
<i>Figura 56 Prueba Configuración Agente</i>	76
<i>Figura 57 Contenedor de agentes</i>	76
<i>Figura 58 Desarrollo de aplicación para pruebas de carga</i>	77
<i>Figura 59 . Comparativa de comportamientos</i>	78
<i>Figura 60 Estado de memoria y procesador prueba normal</i>	78
<i>Figura 61 ESTADO DE MEMORIA Y PROCESADOR PRUEBA SATURADO</i>	79
<i>Figura 62 Prueba con resultados saturación</i>	79
<i>Figura 63 Prueba Resultados saturado</i>	80
<i>Figura 64.Emulador Android</i>	80
<i>Figura 65Emulador pantalla principal</i>	81
<i>Figura 66.Emulador pantalla acciones</i>	81
<i>Figura 67.Emulador pantalla de ingresos y menú</i>	82
<i>Figura 68.Emulador pantalla insertar ingreso</i>	83
<i>Figura 69. Emulador conexión con el contenedor principal</i>	83
<i>Figura 70. Emulador puesta en marcha del agente</i>	84
<i>Figura 71. Pruebas Apkudo</i>	85
<i>Figura 72. Puesta en marcha del análisis Apkudo</i>	85
<i>Figura 73. Resultado de pruebas de la aplicación</i>	86
<i>Figura 74 Teléfono de prueba</i>	87

<i>Figura 75 Análisis Contabilidad Inteligente Play Store</i>	<i>87</i>
<i>Figura 76 Estadísticas Play Store</i>	<i>88</i>
<i>Figura 77 Instalación de dispositivos actual.....</i>	<i>89</i>

ÍNDICE DE TABLAS

TABLA I. REQUERIMIENTOS FUNCIONALES.....	43
TABLA II. REQUERIMIENTOS DE SOFTWARE.....	44
TABLA III. REQUERIMIENTO DE HARDWARE.....	45
TABLA IV. MODELO DE ROL DEL AGENTE.....	46
TABLA V. DEFINICIÓN DE PROTOCOLOS	47
TABLA VI. MODELO DE SERVICIOS.....	48
TABLA VII PRUEBA MAIN ACTIVITY.....	72
TABLA VIII PRUEBA ACTIVIDADES	73
TABLA IX PRUEBA LISTA DE INGRESOS.....	74
TABLA X PRUEBA LISTA DE EGRESOS	74
TABLA XI PRUEBA DE INGRESO CATEGORIA.....	75
TABLA XII PRUEBA DE EGRESO CATEGORIA	75
TABLA XIII PRUEBA CONFIGURACIÓN DEL AGENTE	76
TABLA XIV NÚMERO DE PETICIONES.....	77
TABLA XV ESTADÍSTICAS DE INSTALACIÓN	88
TABLA XVI. RECURSOS HUMANOS.....	92
TABLA XVII. RECURSOS TÉCNICOS Y TECNOLÓGICOS	92
TABLA XVIII. RECURSOS MATERIALES	93
TABLA XIX. PRESUPUESTO TOTAL.....	94

C. INTRODUCCIÓN

La universidad nacional de Loja dentro de sus objetivos tiene como prioridad la formación profesional con el fin de brindar soluciones a los problemas de la realidad que está sometida nuestra sociedad, y es indiscutible que las nuevas tecnologías están al servicio de brindar soluciones; mejorando el estatus de vida de cada persona.

El aumento de la pobreza y del subdesarrollo se debe especial mente a la falta de dinero o de un presupuesto ajustado a las necesidades de cada persona para su subsistencia, esto ha contribuido a que las nuevas tecnologías estén orientadas a solucionar esta clase de problemas, y estos adelantos son innegables ya que ahora contamos con diferentes software que ayudan de alguna manera solucionar este percance, pero hay que tomar en cuenta que para dispositivos móviles es escasa la implementación de esta clase de sistema, es allí donde surge la idea de crear una aplicación que ayude a llevar nuestros ingresos y egresos económicos de una forma sencilla y portable desde nuestro dispositivos móviles android, además se ha hecho uso de agentes inteligentes que son parte de la inteligencia artificial, esto ayuda a que la aplicación sea sostenible y robusta.[1]

Con esta perspectiva el presente trabajo de investigación se compone de tres capítulos en los cuales se abarca la información relevante para la realización del mismo que se relatan a continuación.

La economía personal es una tarea tediosa y que cuenta con algunos parámetros de evaluación, que si no se es un experto en la materia se dificulta llevar el control. Las variables que intervienen son ingresos, egresos, categorías, fecha; que a la final son parámetros que ayudan a tener una descripción y procedes a su análisis.

Los agentes inteligentes proponen una alternativa eficaz a la hora de programar ya que son entidades, estas ayudan a que el la aplicación sea incremental y sostenible, como ejemplo esta JADE que es una plataforma desarrollada en JAVA, haciendo énfasis en un biblioteca llamado JADE-LEAP que es para desarrollar aplicaciones en dispositivos android.

Hay que tomar en cuenta que el SO (Sistema operativo) Android es uno de los más populares del mercado, y su desarrollo se realiza bajo los conceptos de java, existen varias metodologías para implementación de aplicaciones pero en este trabajo se ha realizado con XP (Programación Extrema).[2]

La metodología utilizada para establecer el camino a seguir con el fin de lograr conocimientos seguros, confiables y demostrables; y las técnicas aplicadas en el transcurso del proyecto se resumen en la fase de materiales y métodos.

En la fase de resultados se hace énfasis a la solución y sus diferentes etapas del desarrollo para su puesta en marcha y funcionamiento.

Se referencia la fase de discusión, si se han cumplido con los objetivos planteados en este proceso investigativo tanto el objetivo general como los específicos y su evaluación del desarrollo.

Finalmente se pone de manifiesto las conclusiones determinadas mediante el estudio de los resultados obtenidos, las respectivas recomendaciones o sugerencias para el posterior mejoramiento y sostenibilidad de la aplicación creada; así como también las diferentes bibliografías que hacen referencia a los contenidos abordados y los anexos en los que constan los recursos complementarios utilizados dentro de este trabajo de titulación.

D. REVISIÓN DE LITERATURA

Capítulo 1. Economía

1. Introducción Economía.

La economía es el estudio de cómo la gente en cada país o grupo de países utilizan o administran sus recursos limitados con el objeto de producir bienes y servicios y distribuirlos para su consumo entre los miembros de la sociedad de modo que satisfagan sus necesidades.

El propósito de la economía es bosquejar un proyecto de prosperidad para los individuos y para la sociedad. Para ello hace el análisis a soluciones de eficiencia en la producción, distribución, oferta y la demanda de bienes y servicios.

Debido a que la escasez nos lleva a economizar y a elegir el máximo beneficio al mínimo costo, a veces se conoce a la economía como la ciencia de la elección, que predice en qué forma los cambios de circunstancias afectan las elecciones de las personas. Esta comprensión exige contar con teorías que expliquen el funcionamiento de los fenómenos económicos, y para ello hay que recurrir a la abstracción.

1.1. La actividad económica

Es el uso de recursos que hace la gente para afrontar la escasez y generar riqueza.

1.1.1. Economizar

Es hacer el mejor uso de los recursos disponibles, limitados frente a las necesidades.

1.1.2. Optimizar

Es evaluar el mejor uso de los recursos y procurar el máximo beneficio o el mínimo costo, dentro de los límites de lo posible.

Fenómenos como el desempleo y la inflación son referente común en las intervenciones económicas. En los estratos socioeconómicos altos la mayoría de la gente tiene diferentes tipos de necesidades satisfechas y se trabaja para vivir aún mejor.

1.1.3. La riqueza

Se define aquí en función del logro de bienes y servicios que cubren necesidades inmediatas y futuras. En los estratos socioeconómicos bajos la insatisfacción es alta y se trabaja para sobrevivir. Los diferentes estándares de vida son una expresión de la cantidad, calidad y variedad de los bienes y servicios.[3]

- **Los gastos económicos.**

A cualquier empresa, industrial, comercial o de prestación de servicios– para funcionar normalmente le resulta ineludible adquirir ciertos bienes y servicios como son: mano de obra, energía eléctrica, teléfono, etc.

La empresa realiza un gasto cuando obtiene una contraprestación real del exterior, es decir, cuando percibe alguno de estos bienes o servicios. Así, incurre en un gasto cuando disfruta de la mano de obra de sus trabajadores, cuando utiliza el suministro eléctrico para mantener operativas sus instalaciones, etc.

El hecho de que la entidad utilice ciertos servicios le obliga a pagarlos. Ahora bien, el pago de éstos se produce en la fecha en la que sale dinero de tesorería para remunerarlos. Por ello, es sumamente importante distinguir los conceptos de gasto y de pago. Se incurre en un gasto al disfrutar de cierto servicio; se efectúa el pago del mismo en la fecha convenida, en el instante en que reciba el bien o servicio o en fechas posteriores, puesto que el aplazamiento del pago es una práctica bastante extendida en el mundo empresarial.

Por lo tanto, la realización de un gasto por parte de la empresa lleva de la mano una disminución del patrimonio empresarial. Al percibir la contraprestación real, que generalmente no integra su patrimonio, paga o aplaza su pago, minorando su tesorería o contrayendo una obligación de pago, que integrará el pasivo.[4]

Tipos de gastos.

En las finanzas, antes de comenzar a recortar su presupuesto, es útil profundizar su análisis financiero y categorizar los gastos. Una manera de hacerlo es someterlos a dos filtros. El primero separa los gastos fijos de los variables, y el segundo separa las necesidades de los deseos.

1.1.4. Los gastos fijos

Son los que no se modifican de un mes a otro, como el pago de la hipoteca o el alquiler, las primas del seguro y la membresía del gimnasio.

1.1.5. Los gastos variables

Por el contrario, varían de un mes a otro e incluyen lo que gasta en alimentos, transporte, gastos médicos y entretenimiento. El segundo filtro diferencia los gastos esenciales de los opcionales, o no esenciales. En términos más informales, discrimina necesidades y deseos.

1.1.6. Los gastos esenciales

Son aquellos de los que, literalmente, no puede prescindir, como alimentos, vestimenta y vivienda. Si bien puede tomar medidas para mantener los gastos en estas áreas bajo control, por ejemplo, no salir a comer con frecuencia y optar por una casa o apartamento más pequeño que el que le gustaría, no puede reducir estos gastos por completo.

1.1.7. Los gastos opcionales

Por otra parte, generalmente son “deseables”, como un nuevo teléfono celular o un par de zapatos, pero no son necesarios. Si bien esta categoría probablemente sea la más fácil de recortar en su presupuesto, no es necesario que se prive de todo lo incluido en este grupo para marcar una diferencia positiva en sus hábitos de consumo.

Al categorizar sus gastos, tendrá un panorama más claro de dónde gasta menos y dónde gasta más. También puede usar su índice de gastos como guía para aclarar aún más las áreas en las que podría aplicar alguna reducción. Un índice

de gastos identifica el porcentaje de sus ingresos totales que destina a pagar un gasto específico.

Ejemplo de Gastos.

Por ejemplo, si su ingreso es de US\$60,000 y el año anterior gastó US\$20,000 en vivienda, divida lo que gastó en vivienda por sus ingresos totales para obtener el índice de gastos. En este ejemplo sería el 33%. Dado que la pauta estándar para gastos de vivienda es el 28% de los ingresos, quizá desee reducir los costos de vivienda y liberar dinero para otros objetivos, como su jubilación. Si esto no fuera posible, ya sea porque no puede o no quiere mudarse, deberá encontrar otras áreas en las que recortar gastos[5]

Ingresos

Los ingresos, en términos económicos, hacen referencia a todas las entradas económicas que recibe una persona, una familia, una empresa, una organización, un gobierno, etc.

El tipo de ingreso que recibe una persona o una empresa u organización depende del tipo de actividad que realice (un trabajo, un negocio, una venta, etc.). El ingreso es una remuneración que se obtiene por realizar dicha actividad.

Cuando una empresa vende su producción o sus servicios a un cliente, el valor de la compra, pagada por el cliente, es el ingreso percibido por la empresa. Estos ingresos pueden ser utilizados por las empresas para pagar los salarios de los trabajadores, pagar los bienes que utilizaron para la producción (insumos), pagar los créditos que hayan obtenido, dar utilidades a los dueños de la empresa, ahorrar, realizar nuevas inversiones, etc.

Tipos de Ingresos.

Cuando el ingreso proviene de actividades productivas, se puede clasificar en varios tipos.

1.1.8. Ingreso marginal

Generado por el aumento de la producción en una unidad.

1.1.9. Ingreso medio.

Ingreso que se obtiene, en promedio, por cada unidad de producto vendida; es decir, es el ingreso total dividido en el total de unidades vendidas.

1.1.10. Ingreso del producto marginal

Ingreso generado por la utilización de una unidad adicional de algún factor de producción (trabajo, capital), por ejemplo, la utilización de un trabajador más, etc.

En general, las personas, las familias, las empresas, etc., buscan aumentar sus ingresos. Si éstos se elevan, su consumo y su ahorro pueden aumentar, llevando, en muchos casos, a un mejor nivel de vida y de bienestar.[6]

- **Ejemplo de ingreso**

El ingreso que recibe un trabajador asalariado por su trabajo es el salario. Si ésta es la única actividad remunerada que hizo la persona durante un periodo, el salario será su ingreso total. Por el contrario, si esta persona, además de su salario, arrienda un apartamento de su propiedad a un amigo, el dinero que le paga el amigo por el arriendo también es un ingreso. En este último caso, el salario más el dinero del arrendamiento constituyen el ingreso total. Los ingresos pueden ser utilizados para satisfacer las necesidades.[6]

- **Ahorro personal.**

Cuidar nuestro ahorro personal es fundamental para la economía familiar ya que si no conseguimos ahorrar una suma de dinero durante el año no podremos hacer frente a los gastos que conlleva la hipoteca del hogar, el coche, los seguros, los recibos de luz y de agua y la alimentación. Si además queremos disfrutar de las vacaciones y concedernos algún capricho es absolutamente necesario saber cómo potenciar el ahorro personal.

Pautas para realizar un ahorro.

Se deben realizar o espetar pautas para conseguir un propósito y en especial cuando se decide hacer un ahorro significativo.



FIGURA 1. CONTABILIDAD Y AHORRO

1.1.11. Consumo responsable e inteligente

Evitando prácticas que generen gastos innecesarios. Evaluar nuestros hábitos y cuantificar qué coste suponen en nuestra economía familiar nos pueden ayudar a cambiar y reducir el consumo. Reutilizar y reciclar tanto ropa como muebles y objetos puede ayudarnos a reducir el gasto en nuestro hogar, con ellos podemos crear decoración y muebles para nuestra casa como los muebles hechos con palets.

1.1.12. Controlar el gasto

Mes a mes planificando previamente en qué vamos a utilizar nuestro dinero. De este modo podemos conocer si hemos realizado gastos extras que no habíamos contemplado en un principio y saber qué podemos hacer para reducirlos.

1.1.13. Ayuda de un profesional

En el caso de que nuestra situación sea muy complicada para que estudie profundamente nuestra situación económica y nos marque unas pautas de comportamiento y de ahorro que habremos de seguir fielmente para mejorar nuestra economía.

1.1.14. Invertir y mejorar nuestro ahorro

Contratando un producto financiero sin asumir riesgos innecesarios, tender a la diversificación y crear una cartera de inversiones estables que nos permitan una liquidez constante de nuestros ahorros y no tomar decisiones precipitadas.[7]

Capítulo 2. Agentes

2. Introducción Agentes.

"Los agentes constituyen el próximo avance más significativo en el desarrollo de sistemas y pueden ser considerados como la nueva revolución en el software" (Dr. Nicholas Jennings).

Frente a una afirmación de este tipo, surgen obligatoriamente ciertas preguntas de carácter general: ¿en qué consiste este nuevo paradigma?, ¿qué es un agente?, ¿cómo se caracteriza un agente?, ¿qué tipos existen de ellos?

Podemos definir al Agente Inteligente como una entidad software que, basándose en su propio conocimiento, realiza un conjunto de operaciones destinadas a satisfacer las necesidades de un usuario o de otro programa, ya sea por iniciativa propia o porque alguno de éstos se lo requiere.

Todos los agentes inteligentes son programas, pero no todos los programas son agentes inteligentes. Los agentes en sí mismos pueden ser considerados como entidades individuales (partes de programa que tienen control sobre sus propias vidas y movimientos). Continuamente están realizando procesos que les indican qué hacer y cómo. Además, se comunican con otros agentes para resolver de forma adecuada su trabajo.

Un agente está caracterizado por una serie de calificativos, los cuales vienen a denotar ciertas propiedades a cumplir por el agente, tales como autonomía, sociabilidad, racionalidad, reactividad, pro-actividad, adaptabilidad, movilidad, benevolencia, entre otros. No existe un consenso sobre el grado de importancia de cada una de estas propiedades para un agente. Sin embargo, se puede afirmar que estas propiedades son las que distinguen a los agentes de muchos programas.

Además, los agentes inteligentes se clasifican por estar basados en mapeo de percepciones de acciones o según el tipo de aplicación. Éstos últimos pueden ser Aplicaciones de interfaz, de sistemas, de concejos, de navegación, de monitoreo, de recomendación, y de recuperación de información.[8]

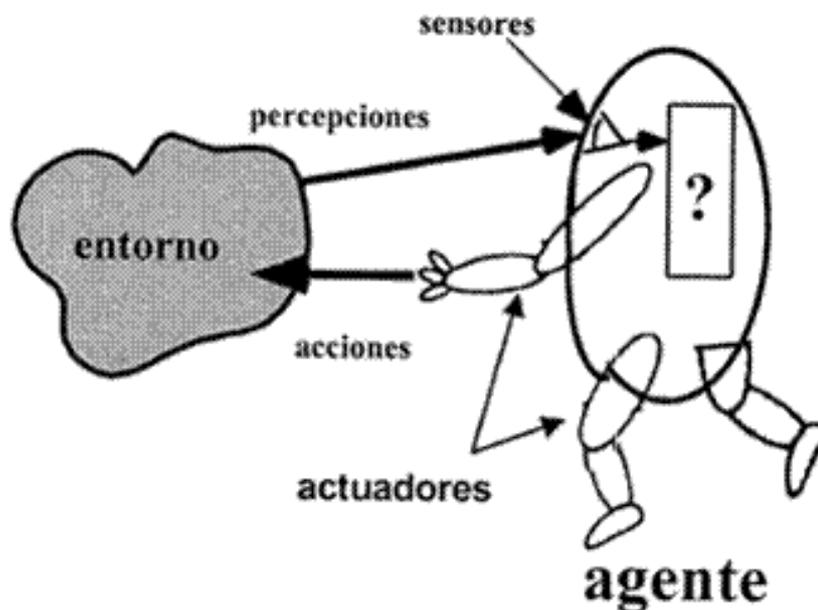


FIGURA 2. AGENTES INTELIGENTES

2.1. Características de los agentes.

Las características que se mencionaran son fundamentales, pero a su vez los agentes inteligentes pueden presentar otras peculiaridades tales como, cooperación (capacidad de interactuar con otros agentes), movilidad (de un sistema a otro para acceder a recursos remotos o reunirse con otros agentes), benevolencia (disponibilidad para ayudar a otros agentes cuando no entre en conflicto con sus propios objetivos).

Un agente inteligente está diseñado especialmente para prestar servicio al usuario con el que se comunica a través de consultas, y para ello posee al menos unos de los siguientes elementos:

Capacidad de proceso: es capaz de descomponer las consultas en subconsultas y asociar los términos resultantes de esta operación con otros afines.

Conocimiento del entorno donde se mueve: un agente debe saber en todo momento a qué información acceder o a qué otro agente dirigirse para obtener esta información.

Autonomía

El agente inteligente actúa sin ningún tipo de intervención humana directa y tiene control de sus propios actos.

Capacidad de reacción o reactividad

Los agentes pueden detectar cambios en su entorno y reaccionar ante ellos.

2.1.1. Adaptabilidad

Es una consecuencia directa de la característica de reacción, ya que los cambios producidos por el entorno provocan que los agentes se estén continuamente adaptando a dichos cambios.

Comunicación o sociabilidad

El agente es capaz de comunicarse por medio de un lenguaje común con otros agentes e incluso con las personas.

Para un SMA la coordinación es indispensable y esta no se puede realizar mientras no exista comunicación.

2.1.2. Formas posibles de comunicación:

Sin comunicación. Los agentes pueden interactuar sin comunicarse, infiriendo las intenciones de otros agentes. Esto puede deberse a la imposibilidad técnica o al deseo de una mayor autonomía por parte de los agentes.

Comunicación primitiva. Se restringe la comunicación a un conjunto de señales con interpretación fija.

Arquitecturas de pizarra. Los agentes se comunican mediante un esquema que permite publicar mensajes al resto de agentes.

Paso de mensajes. Los agentes dirigen mensajes a otros agentes, identificados por una dirección, nombre o alias.

Comunicación de alto nivel. Consiste en posibilitar la comunicación en el nivel simbólico que requieren los agentes deliberativos o intencionales para comunicar sus deseos, intenciones y objetivos. Para ello se definen actos comunicativos y conversaciones que incluyen una descripción semántica convenida de antemano. El lenguaje KQML6 aplica la teoría de actos de habla, y los

estándares de comunicación de agentes de la FIPA. Así mismo, FIPA – ACL es otro lenguaje de comunicación de actos del habla, mejorando los estándares de comunicaciones de KQML.

Interacciones Hombre- Máquina. La comunicación entre un agente artificial y un usuario se realiza mediante dos estrategias, básicamente. Convertir los mensajes del usuario a un lenguaje de comunicación de agentes y aprovechar la tecnología Multiagentes para simplificar la comunicación con el usuario.

Los mensajes FIPA ACL están compuestos de propiedades variables según el contexto. La única propiedad obligatoria es performative (tipo de mensaje) aunque en general se suelen enviar también los campos sender, receiver, y content.[9]

Iniciativa o pro-actividad

El agente tiene un propósito u objetivo determinado y emprende las acciones necesarias hasta conseguirlo.

Continuidad temporal

los agentes no sólo realizan ejecuciones en un momento determinado sino que, desde su creación, pasan a un estado de espera hasta cualquier evento provocado por otro agente o usuario, o cualquier cambio producido en el entorno les haga reaccionar.[10]

2.2. Clasificación de los Agentes Inteligentes.

2.2.1. Agentes de Reflejos Simples

El recurso de utilizar una tabla de consulta explícita está fuera de toda consideración. Pero es posible resumir fragmentos de la tabla, prestamos atención a ciertas asociaciones entre entradas/ salidas que se producen con frecuencia. Se realiza un proceso como respuesta a la entrada visual y se establece la condición “el auto de adelante está frenado”, esto activa la conexión ya definida en el programa del agente y la acción correspondiente “frenar”. A esta conexión se la llama Regla de Condición – Acción, se escribe: Si el auto de adelante está frenado entonces empieza a frenar.

Las reglas condición – acción permitan al agente establecer la conexión entre percepciones y acciones. Los rectángulos se usan para indicar el estado interno en un momento dado del proceso de decisión del agente. Los óvalos representan la información de base utilizada en el proceso.

Un agente reflejo simple actúa encontrando una regla cuya condición coincida con la situación actual (definida por la percepción) y efectuando la acción que corresponda a tal regla.

Sólo funcionará si se toma la decisión adecuada con base en la percepción de un momento dado.[11]

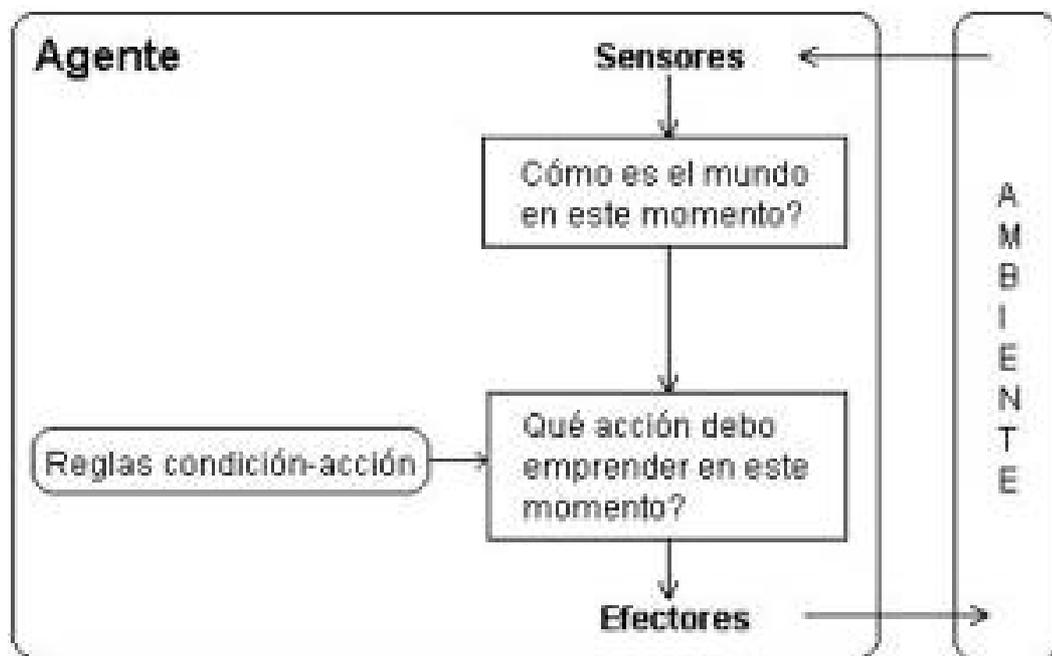


FIGURA 3. AGENTES DE REFLEJO SIMPLE

2.2.2. Agentes bien Informados de todo lo que pasa.

El problema es que los sensores no informan acerca del estado total del mundo. El agente necesita actualizar información en el estado interno que le permita discernir entre estados del mundo que generan la misma entrada de percepciones pero que son distintos. Para cada uno de los estados, se necesitan acciones distintas. El estado interno, le da al agente las condiciones de optar por una acción.

La actualización de esta información sobre el estado interno, exige dos conocimientos en el programa de agente. Primero, se necesita información de cómo está evolucionando el mundo, independientemente del agente. Segundo, se debe informar como las acciones del agente afectan al mundo.

Se combina las percepciones prevalecientes con el estado interno anterior para generar la descripción actualizada del estado prevaleciente.

En el programa se crea una nueva descripción del estado interno.[11]

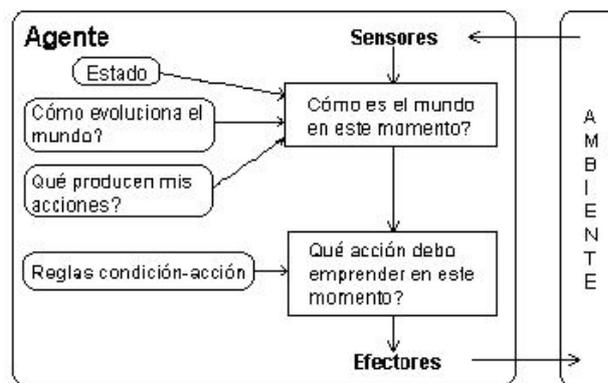


FIGURA 4. AGENTES BIEN INFORMADOS

2.2.3. Agentes Basados en Metas

Para decidir que hay que hacer, además de una descripción del estado prevaleciente, también se requiere información sobre su meta, información que detalla las situaciones deseables.

La búsqueda y la planificación, son los sub campos de la inteligencia artificial que se ocupa de encontrar las secuencias de acciones que permiten alcanzar las metas de un agente.

Aquí en la toma de decisiones se toma en cuenta el futuro. En el agente reflejo ésta información no se utiliza puesto que el diseñador calcula previamente la acción correcta. Un agente basado en metas, razonaría que sí en el auto de adelante se prenden las luces, debe disminuir su velocidad. Considerando la forma como normalmente evolucionan las cosas en el mundo, la única acción permitida para alcanzar la meta, es frenar.

El agente basado en metas es menos eficiente, pero es más flexible.[11]

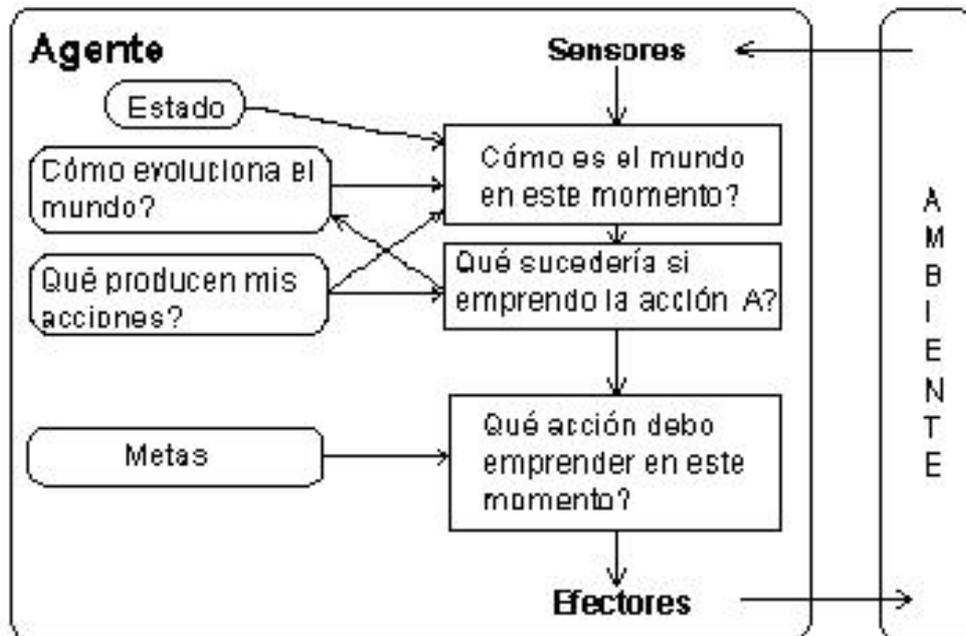


FIGURA 5. AGENTES BASADOS EN METAS

2.2.4. Agentes Basados en la Utilidad

Si se prefiere un estado del mundo por otro, entonces ese estado ofrece mayor utilidad al agente. La utilidad es una función que correlaciona un estado y un número real que corresponde al grado de satisfacción.

La utilidad permite la toma de decisiones racionales cuando la meta tiene problemas. Primero, cuando el logro de alguna meta implica un conflicto y sólo alguna de ellas se puede obtener, la utilidad definirá cual es el compromiso adecuado. Segundo, cuando son varias las metas que el agente podría desear obtener, pero no existe la certeza de lograr ninguna de ellas, la utilidad pondera la posibilidad de tener éxito considerando la importancia de las diferentes metas.[11]

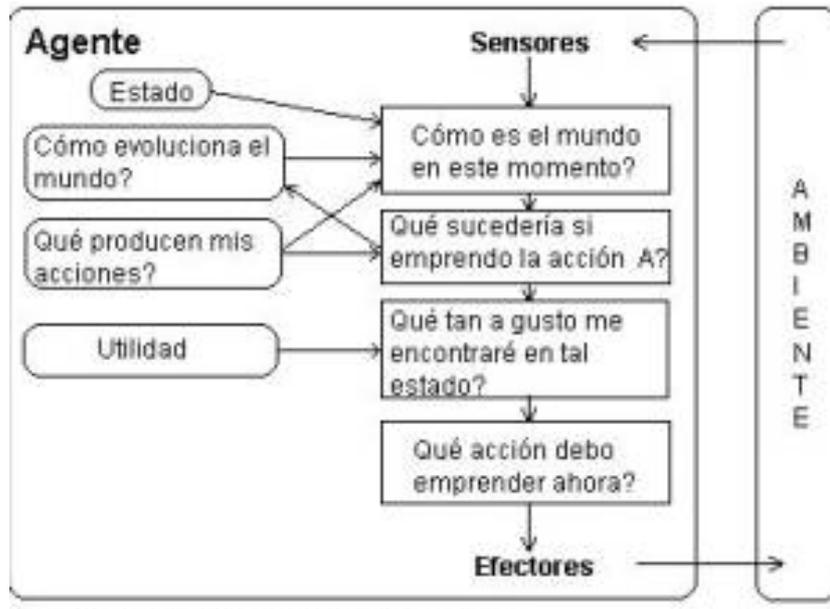


FIGURA 6. AGENTES BASADOS EN LA UTILIDAD

2.2.5. Agentes de interfaz

Se caracteriza por su capacidad de hacer comprensible las interfaces. También denominados asistentes personales, tienen como objetivo simplificar las tareas rutinarias que realiza un usuario, por ejemplo detectar que una noticia pueda ser importante para un usuario y comunicárselo.[12]

Agentes móviles.

Esto quiere decir que el agente puede transitar entre varias máquinas para utilizar recursos de los cuales no dispone en su propia máquina o simplemente para evitar una sobrecarga de comunicación. En este tipo el principal problema es la seguridad. Pretende facilitarle al usuario el acceso a la información personal cuando usa dispositivos móviles y una conexión de red estable, como redes inalámbricas.[12]

2.2.6. Agentes de información.

Su objetivo es recolectar información a través de la red, indexarla y ofrecerla al usuario cuando realiza una consulta, Estos Agentes han tenido gran acogida debido a las grandes cantidades de información que circulan hoy por la red.[12]

2.2.6.1. Agentes Consejeros

Este tipo de agente da consejos al usuario, una herramienta y un sistema de diagnóstico o ayuda. [13]

2.2.6.2. Agentes de Navegación

Estos agentes son utilizados para navegar en la red, su funcionamiento principal es recordar sitios y direcciones de interés para el usuario.[14]

2.2.6.3. Agentes de monitoreo

Estos agentes proporcionan información de manera eficaz y oportuna para el usuario, en el momento que ocurre un evento.[13]

2.2.6.4. Agentes recomendación

Este agente posee una base de datos con información acerca de un tópico de interés para un grupo, al hacer las recomendaciones se basan en analogías con otros usuarios de perfil similar.[13]

2.3. Tipos de Ambientes AI.

Al determinar los agentes inteligentes tenemos que tener claro la definición de los ambientes que intervienen dentro de los mismos y a continuación tenemos un enfoque global de cada uno de ellos.

2.3.1. Accesibles y no accesibles.

Si los sensores detectan todos los aspectos necesarios para elegir una acción. Ejemplo: 8 puzzle (accesible) Juego de cartas (No accesible).

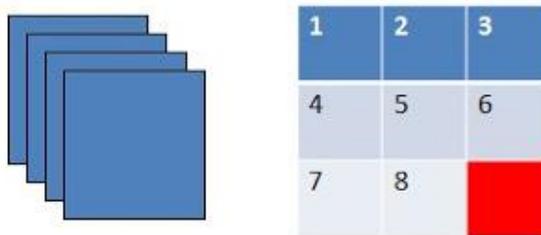


FIGURA 7. AI ACCESIBLES Y NO ACCESIBLES

2.3.2. Determinista y no determinista

Si el estado siguiente del ambiente se puede determinar completamente con el estado actual y las acciones escogidas.

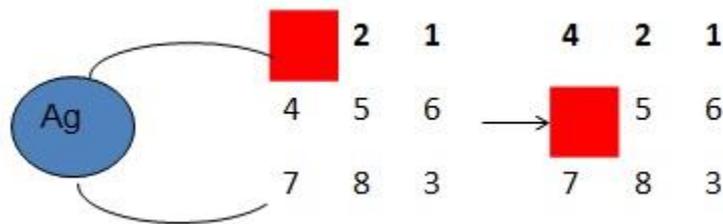


FIGURA 8. AI DETERMINISTAS Y NO DETERMINISTAS

2.3.3. Episódicos y no episódicos

La experiencia del agente se divide en episodios.

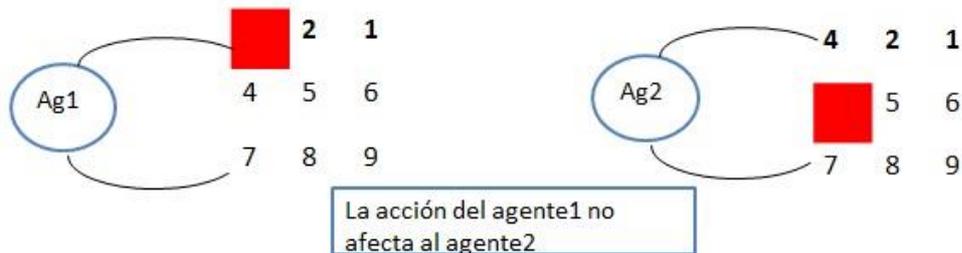


FIGURA 9. AI EPISÓDICOS Y NO EPISÓDICOS

2.3.4. Estática y dinámica

Si es que el medio ambiente no cambia cuando el agente está deliberando.

Ejemplo:

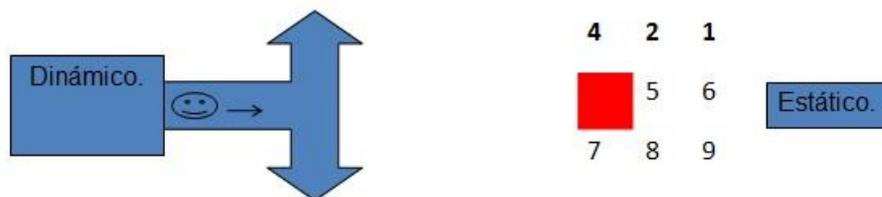


FIGURA 10. AI ESTÁTICOS Y DINÁMICOS

2.3.5. Discretos y continuos

Discretos si es que existe una cantidad limitada de percepciones y acciones distintas y distinguibles.

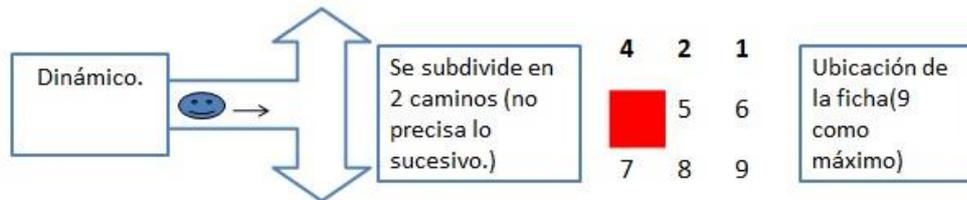


FIGURA 11. AI DISCRETOS Y CONTINUOS

2.4. Metodologías orientadas a agentes.

Existe varias metodologías que permiten la creación de agentes cada una de ellas se acopla dependiendo el problema a resolver se mencionara algunas de las más utilizadas e importante a más de esto, en esta sección haremos énfasis a la metodología GAIA la cual será implementada para la creación del agente inteligente deliberativo para llevar los ingresos y egresos contable de una persona.

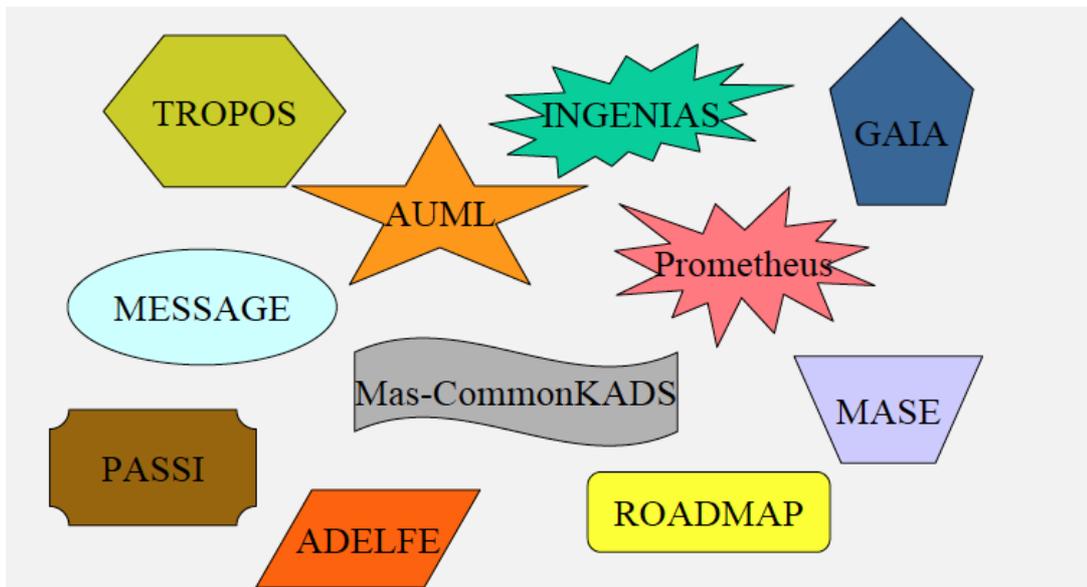


FIGURA 12. METODOLOGÍAS ORIENTADAS A AGENTES INTELIGENTES

2.4.1. Metodología Tropos.

Tropos es una metodología que está siendo desarrollada por un grupo de investigación conformado por personas pertenecientes a diferentes universidades del mundo, y que tiene su principal centro en la Universidad de Trento en Italia. Esta metodología de desarrollo de sistemas, toma como base tecnologías de Sistemas Multiagentes para definir la fase de implementación; además, se propone cubrir todo el ciclo de desarrollo de vida y está centrada en el modelamiento temprano de requisitos. Los conceptos relacionados con esta actividad, son la base de la metodología y la estructura de trabajo para desarrollo que proponen. Tropos adopta el marco de trabajo planteado por el cual ofrece las nociones de actor, meta y dependencia de actores, utilizándolas para modelar los requerimientos y el diseño de la arquitectura.

Una de las principales características de esta metodología se encuentra en el manejo de requerimientos, dado que los actores y las metas son usados como fundamentos para el modelamiento y el análisis en todas las fases del desarrollo. De esta manera, se manejan cinco fases para el desarrollo de software, las cuales son:

2.4.1.1. Requerimientos tempranos

En esta fase se entiende el problema mediante el estudio de una organización existente. El resultado de esta fase es un modelo organizacional que incluye a los actores relevantes y sus respectivas dependencias.

2.4.1.2. Requerimientos tardíos

En esta fase el futuro sistema se describe en su entorno operacional junto con las funciones relevantes y cualidades. Esta descripción modela el sistema como un conjunto de actores que así mismo tienen dependencias con actores de su entorno. Estas dependencias definen los requerimientos funcionales y no-funcionales del sistema.

2.4.1.3. Diseño de arquitectura

En esta fase se define la arquitectura global del sistema en términos de subsistemas interconectados por medio de flujos de datos y de control. En este marco de trabajo, los subsistemas son representados como actores y flujo de datos o control y las interconexiones son representados como dependencias del sistema o del actor.

2.4.1.4. Diseño detallado

En esta fase se define cada componente de la arquitectura con mayor detalle en cuanto a entradas, salidas, controles y otra información relevante. En este marco de trabajo se adoptan algunos elementos de AUML.

2.4.1.5. Implementación

Esta es la fase en la cual se desarrolla la implementación del sistema de manera consistente con el diseño detallado. Se utiliza una plataforma para programación de agentes basada en la arquitectura BDI (Beliefs-Desires-Intentions).[15]

2.4.2. Metodología Ingenias.

La metodología INGENIAS (Gómez, Pavón) propone un proceso de desarrollo basado en el Proceso Unificado de Desarrollo de Software (Unified Software Development Process o RUP) para desarrollar SMA2. Durante este proceso se generan una serie de modelos que se detallan a continuación:

- **Modelo de organización.** Describe la estructura del SMA, los roles y los flujos de trabajo (workflows).
- **Modelo de entorno.** Describe las entidades y relaciones con el entorno del SMA.
- **Modelo de agente.** Describe los agentes del sistema sus responsabilidades, control y estado mental.
- **Modelo de objetivos y tareas.** Identifica los objetivos y tareas generales del sistema, los cuales pueden asignarse a agentes.
- **Modelo de interacción.** Describe las interacciones existentes entre agentes.

Estos modelos se especifican en los flujos de análisis y diseño, los cuales se dividen a su vez en tres fases, inicio, elaboración y construcción, de acuerdo al modelo iterativo RUP. La fase de análisis- inicio identifica los componentes que utilizan los agentes para interactuar con el entorno y especifica los casos de uso más relevantes. Los componentes se traducen en aplicaciones que especifican las acciones que pueden ejecutar las percepciones de los agentes. La fase de diseño- inicio consiste en elaborar un prototipo del sistema que muestra como interactúa el usuario con la aplicación y qué tipo de resultados se esperan. En la fase de análisis- elaboración se analizan los casos de uso, obtenidos en la etapa anterior, para generar la arquitectura del sistema. Para ello se refinan los casos de uso y se asocian con modelos de interacciones. La fase de diseño- elaboración aumenta el nivel de detalle de la especificación determinando cómo se llevan a cabo los casos de uso identificados. Los resultados que se obtienen se centran en refinar los flujos de trabajo y las tareas asociadas, las interacciones, cómo es el control del agente y cómo se satisfacen los objetivos del sistema. El modelo de organización se refina asociando tareas a los flujos de trabajo. El modelo de interacción se detalla añadiendo relaciones con los flujos de trabajo. Tras desarrollar los casos de uso necesarios para determinar la arquitectura del sistema, se pasa a la fase de análisis- construcción para identificar los casos de uso restantes sin modificar sustancialmente la visión del sistema que se tiene hasta ahora del sistema. En la fase de diseño -construcción se detallan las relaciones sociales, principalmente las relaciones de subordinación y las relaciones cliente-servidor.[16]

2.4.3. Metodología GAIA.

GAIA tiene la capacidad de implantar desde la solución más sencilla hasta aquellas que contengan un alto grado de complejidad, así como el análisis, diseño y la liberación del mismo.

2.4.3.1. Introducción.

GAIA es una metodología para el diseño de sistemas basados en agentes cuyo objetivo es obtener un sistema que maximice alguna medida de calidad global.

En GAIA se entiende que el objetivo del análisis es conseguir comprender el sistema y su estructura sin referenciar ningún aspecto de implementación.

Esta metodología sólo buscar especificar cómo una sociedad de agentes colabora para alcanzar los objetivos del sistema, y qué se requiere de cada uno para lograr esto último. La principal crítica que se puede hacer a GAIA es que se queda a un nivel de abstracción demasiado alto.

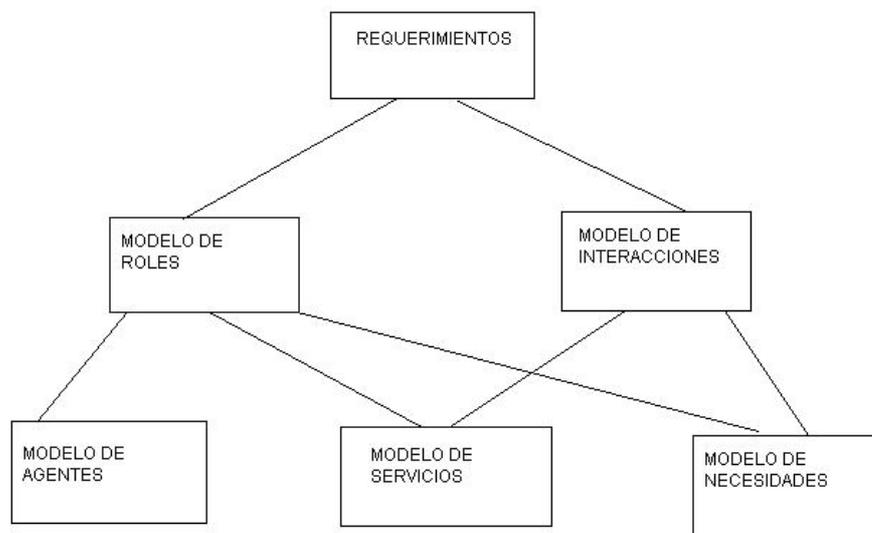


FIGURA 13. ACCIONES DE LA METODOLOGÍA GAIA

2.4.3.2. Descripción.

En GAIA se habla de organización que es una colección de roles, los cuales mantienen ciertas relaciones con otros e interacciones con otros roles. Los roles agrupan cuatro aspectos: responsabilidades del agente, los recursos que se le permite utilizar, las tareas asociadas e interacciones.

GAIA propone trabajar inicialmente con un análisis a alto nivel donde se usan dos modelos, el modelo de roles para identificar los roles clave en el sistema junto con sus propiedades definitorias y el modelo de interacciones. Luego vendría el diseño a alto nivel. El objetivo de este diseño es generar tres modelos: el modelo de agentes (tipos), modelo de servicios (funciones), y un modelo de conocidos (enlaces).

Los principales conceptos que aparecen en la metodología se dividen en dos: abstractos y concretos.

Las entidades abstractas son aquellas que son empleadas durante el análisis para la conceptualización del sistema.

Las entidades concretas son empleadas en el proceso de diseño.

La entidad más abstracta de un sistema en la jerarquía de conceptos que se presenta es el sistema relacionado con la idea de “sociedad” u “organización”

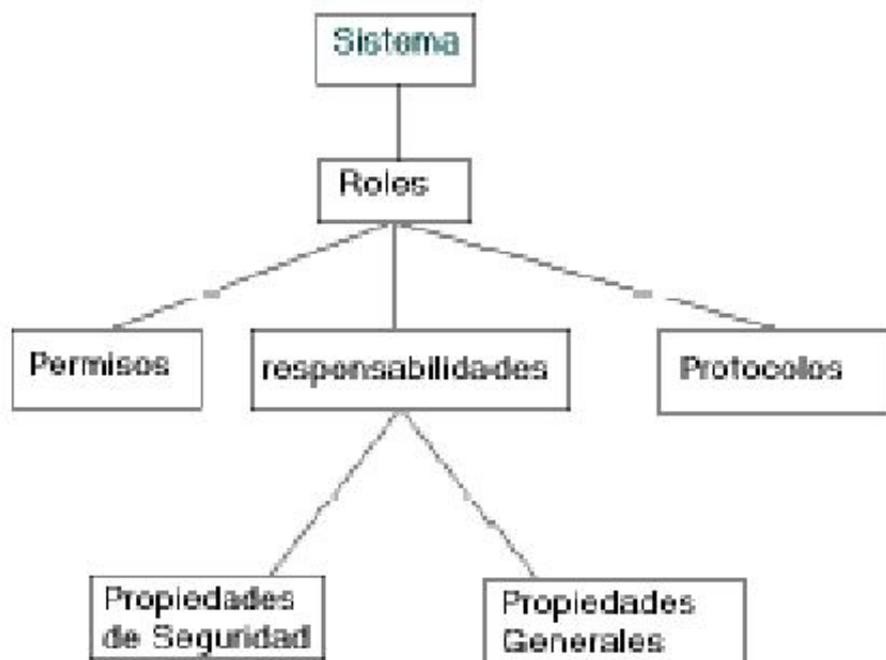


FIGURA 14. DEFINICIÓN DE ETAPAS GAIA

Un rol es asociado a:

Un conjunto de permisos: derechos asociados al rol, identifican los recursos que están disponibles al rol para realizar sus responsabilidades.

Un conjunto de protocolos: los cuales definen la manera de interactuar del rol con otros roles.

Las responsabilidades: determinan la funcionalidad, podemos ver dos tipos:

- Propiedades de viveza: describen aquellos estados de los asuntos que un agente debe efectuar. Decir que “algo será hecho”
- Propiedades de seguridad: son invariantes.
- Condiciones de seguridad.

2.4.4. Procesos para hacer los modelos.

Los modelos se determinan en fases que determinan de forma clara cada una de sus etapas, para ayudar a la mejor comprensión de su implementación.

2.4.4.1. Para hacer modelos en GAIA requiere de un conjunto de etapas que se menciona a continuación.

El proceso de análisis

En resumen el proceso de análisis se compone de:

- a) Identificar los roles del sistema
- b) Para cada papel identificar y documentar los protocolos asociados
- c) Empleando el modelo de protocolos como base, elaborar con más detalle el modelo de roles.

El proceso de diseño.

En resumen el proceso de diseño se compone de las siguientes fases:

- a) Creación de un modelo de agente, agregando roles a los tipos de agentes y documentando las instancias de cada tipo de agente.
- b) Desarrollo de un modelo de servicios, examinando protocolos y propiedades de viveza y seguridad.
- c) Desarrollo de un modelo de conocimiento, por medio del modelo de interacción y el modelo de agente.[17]

Capítulo 3. Android

3. Introducción.

Android es la plataforma libre desarrollada por Google, ampliamente utilizada en multitud de dispositivos como móviles, tabletas, TV o sistemas empujados. Su expansión ha sido espectacular, superando en la actualidad al S.O. Microsoft Windows.[18]

La evolución de la tecnología va a paso veloz, Android es de las tecnologías que esta alcanzado a todos por el simple motivo de que se encuentra en los móviles. Android es un sistema operativo basado en Linux. La diferencia principal es que tiene módulos que responden a la pantalla táctil, eventos nativos del móvil. Se desarrolló por una compañía llamada Android, Inc. En 2005 Google adquiere la empresa para seguir trabajando en el mismo proyecto que después conociera la luz como un S.O. para móviles denominado finalmente como Android.[19]

3.1. Historia de android.

Android tiene una característica peculiar: las versiones tienen nombre de postres en inglés y cada versión que cambia, continúa de forma incremental en el alfabeto, es decir que si el primer nombre inicio con A, el siguiente con B, el siguiente C y así sucesivamente; ya veremos que sucede cuando lleguen a la Z.

Hasta el día de hoy, que comienzo a escribir el manual Android para Desarrolloweb.com, tenemos la versión 4.4 KitKat.

Demos un repaso a las Versiones.

- **Versión 1.0 Apple Pie** - Salió en septiembre del 2008.
- **Versión 1.1 Banana Bread** - Salió en febrero 2009.
- **Versión 1.5 Cup Cake** - Salió en abril 2009
- **Versión 1.6 Donut** - Salió en septiembre 2009
- **Versión 2.0 Eclair** - Salió en octubre 2009
- **Versión 2.2 Froyo** - Salió en mayo 2010
- **Versión 2.3 Gingerbread** - Salió en diciembre 2010

- **Versión 3.0 Honeycomb** - Salió en febrero 2011
- **Versión 4 .0 Ice Cream Sandwich** - Salió en octubre 2011
- **Versión 4.1 Jelly Bean** - Salió en julio 2012
- **Versión 4.4 KitKat** - Salió en octubre 2013 [19]

3.2. Estructura de Android

Android utiliza como base el kernel de Linux pero los dos sistemas no son lo mismo, Android no cuenta con un sistema nativo de ventanas de Linux ni tiene soporte para glibc (librería estándar de C); tampoco es posible utilizar la mayoría de aplicaciones de GNU de Linux.

Además de todo lo ya implementado en el kernel de Linux, Androide agrega algunas cosas específicas para plataformas móviles como la comunicación entre procesos (lograda a través del binder), la forma de manejar la memoria compartida (ashmem) y la administración de energía (con wakelocks).

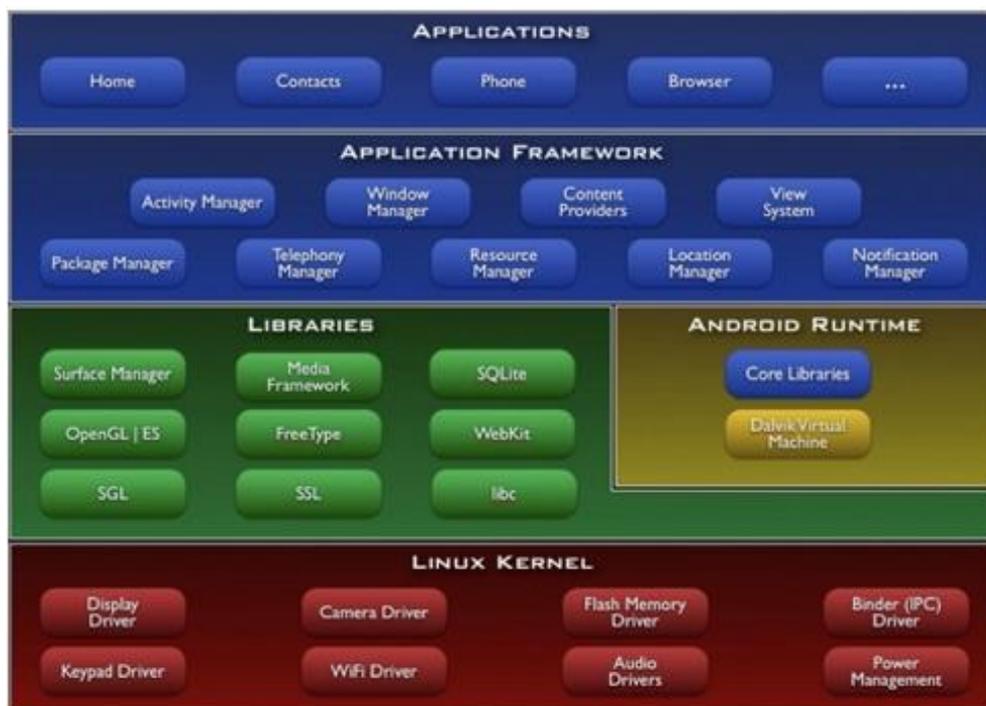


FIGURA 15. ESTRUCTURA DEL SO ANDROID

3.2.1. Capa del Kernel (Roja)

Aquí tenemos el corazón de Android: el manejo de memoria, procesos, *drivers*, etc. Aquí es donde se da la comunicación con el *hardware*. Esto nos sirve para no estar peleando con los fabricantes de cada móvil, nos ayuda a solo usar la “cámara” y no tener que saber cómo funciona la cámara del fabricante X, fabricante Y; solamente hacemos lo que nos interesa, que sería usar la cámara y listo. Además de eso, aquí se administran los recursos del celular, memoria, energía.

3.2.2. Capa Librerías (Verde)

Esta capa tiene las librerías nativas de Android, están escritas en C o C++ y tienen tareas específicas.

- **Surface manager:** Gestión del acceso a la pantalla.
- **Media Framework:** Reproducción de imágenes, audio y vídeo.
- **SQLite:** BD base de datos.
- **Webkit,** Navegador.
- **SGL:** Gráficos 2D.
- **OpenGL:** Gráficos 3D.
- **Freetype:** Renderizar vectores o imágenes.

3.2.3. Android Runtime (Capa Amarilla)

Esta capa amarilla no se considera al 100% una capa. Lo que es muy importante comentar es que aquí se encuentra Dalvik, la máquina virtual de Android, que no es lo misma que la Java Virtual Machine. Esto quiere decir que cuando compilamos en Java lo que se genera solamente va a funcionar en la JVM, porque Dalvik es una máquina virtual, pero de Android, así que el ByteCode que genera Java es inservible para Dalvik.

Algunas de las características de Dalvik son:

- Trabaja en entorno con restricción de memoria y procesador.
- Ejecuta el formato `.dex`.
- Convierte `.class` en `.dx`.

3.2.4. Aplicación Framework (Capa azul)

Esta capa es la es más visible para el desarrollador, ya que la mayoría de los componentes que forman parte del desarrollo los vamos a encontrar aquí.

- **Activity Manager**- Administra las actividades de nuestra aplicación y el ciclo de vida.
- **Windows Manager**- Administra lo que se muestra en la pantalla.
- **Content Provider**-. Administra dependiendo de cómo le indiquemos algunos contenidos, puede ser información que necesitamos la encapsule para enviar o compartir.
- **View**- Las vistas de elementos que son parte de la interfaz gráfica, como los mapas, cuadros de texto, etc.
- **Notification Manager**- Administra las notificaciones.
- **Package Manger**- Administra los paquetes y nos permite el uso de archivos en otros paquetes.
- **Telephony Manager**- Administra lo que tiene que ver con la telefonía, llamadas, mensajes.
- **Resource Manager**- Administra recursos de la aplicación, como los xml, imágenes, sonido.
- **Location Manager**- Gestiona la posición geográfica.
- **Sensor Manager**- Gestiona los sensores que tenga el dispositivo.
- **Cámara**- Administra la cámara.
- **Multimedia**- Administra lo referente a audio, video y fotos.

3.2.4.1. Applications (Capa Azul última)

Aquí tenemos las aplicaciones que vienen en el dispositivo, por ejemplo: el gestor de correos, los mensajes, el *market*, etc.[19]

3.3. Metodologías ágiles para desarrollo de aplicaciones móviles.

Las metodologías en general se clasifican según su enfoque y características esenciales, las más recientes, que se fueron gestando a finales del siglo pasado y que se han comenzado a manifestar desde principios del actual, se han denominado “metodologías ágiles” y surgen como una alternativa a las tradicionales.[20]

3.3.1. Metodología Scrum

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.[21]

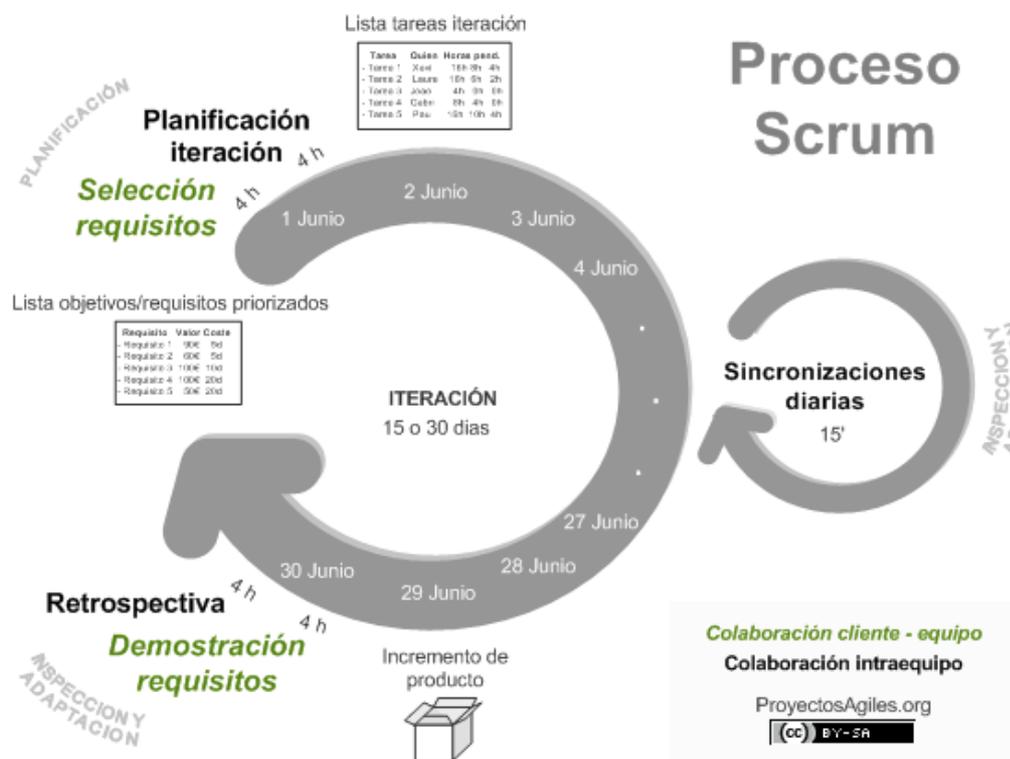


FIGURA 16. METODOLOGÍA SCRUM

3.3.2. Test Driven Development (TDD)

TDD o Test Driven Development es una práctica de programación que consiste en escribir primero las pruebas (generalmente unitarias), después escribir el código fuente que pase la prueba satisfactoriamente y, por último, refactorizar el código escrito. Con esta práctica se consigue entre otras cosas: un código más robusto, más seguro, más mantenibilidad y una mayor rapidez en el desarrollo.

TDD Fue creado por Kent Beck (quien también inventó Extreme Programming y JUnit), y en esencia, es un proceso a seguir, lo cual ya lo hace diferente a un simple enfoque de pruebas primero.

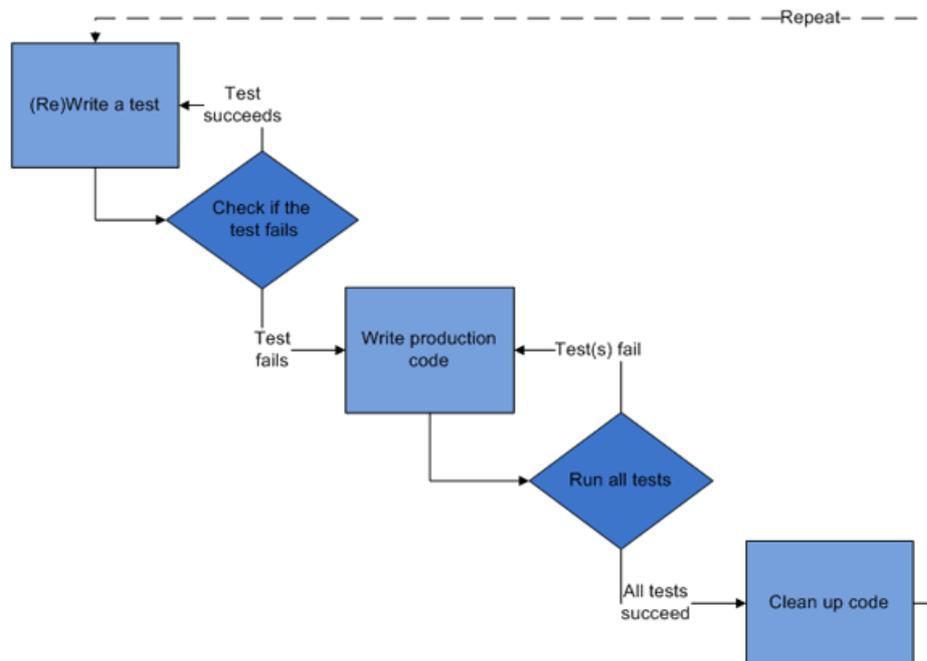


FIGURA 17. METODOLOGÍA TDD

Este ciclo también se lo conoce como rojo (hacer que la prueba falle), verde (hacer que la prueba pase) y refactor. Aunque al principio pueda parecer muy parecido a un enfoque de probar primero, al combinarlo con prácticas de desarrollo ágil, TDD toma un enfoque mucho más amplio, y cambia su atención de las pruebas al diseño. TDD está mucho más relacionado con el diseño emergente que con las pruebas, de hecho, que TDD genere una gran cantidad de pruebas es un efecto secundario positivo, pero no es su propósito final. [22]

3.3.3. Mobile-D

Se podría pensar que Mobile-D es una creación un tanto antigua, ya que se desarrolló como parte de un proyecto finlandés, ICAROS, allá por 2004. Sin embargo, creemos que vale la pena mencionarlo por dos razones. Primera: fue creado mediante un proyecto de cooperación muy estrecha con la industria. El grueso del trabajo fue realizado por los investigadores del VTT. Aun así la

metodología de diseño se elaboró con una participación importante de las empresas de TI finlandesas.

Segundo, Mobile-D es una mezcla de muchas técnicas, los investigadores no dudaron en echar mano de las prácticas habituales de desarrollo software. Pero, al mismo tiempo, consiguieron crear una contribución original para el nuevo escenario del desarrollo de aplicaciones para sistemas móviles. Creemos que este ejemplo ilustra perfectamente cómo se pueden usar conjuntamente diferentes metodologías y técnicas en el contexto del desarrollo ágil.[23]

3.3.4. Programación Extrema (XP).

3.3.4.1. Introducción.

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.[24]

3.3.4.2. Historia

La programación extrema o Xtreme Programming (XP) es un enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre la materia, *Extreme Programming Explained: Embrace Change* (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor

y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.[24]

¿Qué es programación extrema o xp?

- ✓ Metodología liviana de desarrollo de software
- ✓ Conjunto de prácticas y reglas empleadas para desarrollar software
- ✓ Basada en diferentes ideas acerca de cómo enfrentar ambientes muy cambiantes
- ✓ Originada en el proyecto C3 para Chrysler
- ✓ En vez de planificar, analizar y diseñar para el futuro distante, hacer todo esto un poco cada vez, a través de todo el proceso de desarrollo

3.3.4.3. Objetivos.

- ✓ Establecer las mejores prácticas de Ingeniería de Software en los desarrollo de proyectos.
- ✓ Mejorar la productividad de los proyectos.
- ✓ Garantizar la Calidad del Software desarrollando, haciendo que este supere las expectativas del cliente.

Contexto XP.

- ✓ Cliente bien definido
- ✓ Los requisitos pueden (y van a) cambiar
- ✓ Grupo pequeño y muy integrado (máximo 12 personas)
- ✓ Equipo con formación elevada y capacidad de aprender

Características XP.

- ✓ Metodología basada en prueba y error
- ✓ Fundamentada en Valores y Prácticas

✓ Expresada en forma de 12 Prácticas–Conjunto completo–Se soportan unas a otras–Son conocidas desde hace tiempo. La novedad es juntarlas

Valores XP.

✓ Simplicidad XP propone el principio de hacer la cosa más simple que pueda funcionar, en relación al proceso y la codificación. Es mejor hacer hoy algo simple, que hacerlo complicado y probablemente nunca usarlo mañana.

✓ Comunicación Algunos problemas en los proyectos tienen origen en que alguien no dijo algo importante en algún momento. XP hace casi imposible la falta de comunicación.

✓ Realimentación Retroalimentación concreta y frecuente del cliente, del equipo y de los usuarios finales da una mayor oportunidad de dirigir el esfuerzo eficientemente.

✓ Coraje El coraje (valor) existe en el contexto de los otros 3 valores. (si funciona...mejóralo)

El estilo programación extrema XP

✓ Está orientada hacia quien produce y usa el software

✓ Reduce el costo del cambio en todas las etapas del ciclo de vida del sistema.

✓ Combina las que han demostrado ser las mejores prácticas para desarrollar software, y las lleva al extremo.

Prácticas básicas de la programación extrema

Para que todo esto funcione, la programación extrema se basa en doce "prácticas básicas" que deben seguirse al pie de la letra.

- **Equipo completo:** Forman parte del equipo todas las personas que tienen algo que ver con el proyecto, incluido el cliente y el responsable del proyecto.
- **Planificación:** Se hacen las historias de usuario y se planifica en qué orden se van a hacer y las mini-versiones. La planificación se revisa continuamente.

- **Test del cliente:** El cliente, con la ayuda de los desarrolladores, propone sus propias pruebas para validar las mini-versiones.
- **Versiones pequeñas:** Las mini-versiones deben ser lo suficientemente pequeñas como para poder hacer una cada pocas semanas. Deben ser versiones que ofrezcan algo útil al usuario final y no trozos de código que no pueda ver funcionando.
- **Diseño simple:** Hacer siempre lo mínimo imprescindible de la forma más sencilla posible. Mantener siempre sencillo el código.
- **Pareja de programadores:** Los programadores trabajan por parejas (dos delante del mismo ordenador) y se intercambian las parejas con frecuencia (un cambio diario).
- **Desarrollo guiado por las pruebas automáticas:** Se deben realizar programas de prueba automática y deben ejecutarse con mucha frecuencia. Cuantas más pruebas se hagan, mejor.
- **Integración continua:** Deben tenerse siempre un ejecutable del proyecto que funcione y en cuanto se tenga una nueva pequeña funcionalidad, debe recompilarse y probarse. Es un error mantener una versión congelada dos meses mientras se hacen mejoras y luego integrarlas todas de golpe. Cuando falle algo, no se sabe qué es lo que falla de todo lo que hemos metido.
- **El código es de todos:** Cualquiera puede y debe tocar y conocer cualquier parte del código. Para eso se hacen las pruebas automáticas.
- **Normas de codificación:** Debe haber un estilo común de codificación (no importa cuál), de forma que parezca que ha sido realizado por una única persona.
- **Metáforas:** Hay que buscar unas frases o nombres que definan cómo funcionan las distintas partes del programa, de forma que sólo con los nombres se pueda uno hacer una idea de qué es lo que hace cada parte del programa. Un ejemplo claro es el "recolector de basura" de java. Ayuda a que todos los programadores (y el cliente) sepan de qué estamos hablando y que no haya mal entendidos.
- **Ritmo sostenible:** Se debe trabajar a un ritmo que se pueda mantener indefinidamente. Esto quiere decir que no debe haber días muertos en

que no se sabe qué hacer y que no se deben hacer un exceso de horas otros días. Al tener claro semana a semana lo que debe hacerse, hay que trabajar duro en ello para conseguir el objetivo cercano de terminar una historia de usuario o mini-versión.[24]

E.MATERIALES Y MÉTODOS

La implementación del agente inteligente para llevar la economía personal de una persona se realizó a través del entorno de programación android, además haciendo uso de la librería Jade LEAP para el agente y SQLite para la persistencia de datos, integrando todo en Eclipse para su desarrollo.

1. METODOLOGÍA DEL DESARROLLO DEL SISTEMA BASADO EN CONOCIMIENTOS.

Una vez analizadas las metodologías que sirvieron tanto para realizar el análisis como para el desarrollo (Ver sección Revisión Literaria – Capitulo II y Capitulo III), donde se encuentra documentada las diferentes etapas que intervienen en el sistema como los agentes y el desarrollo en android de donde se toma las partes relevantes que son XP (Programación extrema) y GAIA para el desarrollo del sistema.

La metodología de programación extrema sirvió para definir las tareas importantes y relevantes del sistema que intervienen dentro del desarrollo del mismo y que ayudan a agilizar el proceso del mismo.

Además la metodología GAIA para agentes es la que ayuda de manera eficiente a llevar las fases de desarrollo del mismo e identificar las etapas que se encuentran dentro de la misma.

- **Fase I: Identificación de Tareas.**

En esta primera fase se definió algunas estrategias y se determinó algunas necesidades primordiales que servirán para realización del sistema.

Etapas 1: Plan de requisitos y Adquisición de Conocimientos.

En esta fase de recolección de información relevante que servirán de punto de partida para la realización del sistema, mediante la recopilación de esta información teórica (Ver Sección Revisión de Literatura – Capítulo I: Economía, Capitulo II: Agentes Inteligentes, Capitulo III: Android).

Etapa 2: Evaluación y Selección de Tareas

En esta etapa en base a la información obtenida se escogió una metodología tanto de programación como para la realización de agentes inteligentes, dando como mejor alternativa la metodología GAIA para agentes y la metodología XP para la programación ya que permitieron el desarrollo ágil del App y este mismo pueda ser desarrollado de manera eficiente.

Una vez realizada la selección de la metodología se pudo determinar que el proyecto cumple con las características de fiabilidad y factibilidad en su desarrollo.

Etapa 3: Definición de las características de la tarea.

En esta etapa se definió tareas que ayuden a determinar los requerimientos funcionales y no funcionales (Ver Sección Resultados-Aparado 1. Análisis de requerimientos). Esto permitió identificar características iniciales para el sistema y documentar de manera idónea cada una de sus partes, en esta parte de igual forma se definió la parte en la cual interviene el agente inteligente documentado sus etapas iniciales (Ver Sección Resultados Apartado 1.3. Definición de las entradas, roles, prototipos asociados para el agente).

- **Fase II: Desarrollo del sistema**

En esta etapa se desarrolló los diferentes prototipos de carácter arquitectónico, que permite definir y refinar las especificaciones o requerimientos del sistema esto se realizó en forma gradual hasta conseguir tener de forma clara todos los prototipos que intervienen en el App.

Etapa 1: Conceptualización de la solución.

Una vez realizada la conceptualización de los requerimientos en la fase anterior se procedió a determinar los diferentes diagramas para de forma clara y concisa que ayudaran al entendimiento del sistema y a su comprensión tanto en la parte del sistema como en la parte del desarrollo

del agente contable. (Ver Sección Resultados Apartado 2.1 Diseño para la aplicación y el agente deliberativo). Los diagramas ayudaron a la comprensión clara y concisa, de la creación del sistema.

Etapa 2: Desarrollo de la aplicación

Una vez obtenida una idea clara de los diferentes factores que intervienen el sistema, se procedió a desarrollar el sistema escribiendo código y definiendo las diferentes actividades que se realizaran dentro del App. (Ver Sección Resultados Apartado 3. Desarrollo de aplicación e integración con el agente.) En donde se codificó la aplicación, además se determinó los paquetes, la implementación del agente, y las diferentes instalaciones previas a su desarrollo, además en esta sección se procedió a levantar el agente contable en la parte del dispositivo como en el contenedor principal. (Ver Sección Resultados Apartado 3.2.2 Levantamiento de los contenedores de agentes)

Una vez realizado estos requerimientos previos se pudo proceder a su implementación eficaz del sistema hasta tener el producto tangible que es el App que se encuentra en producción.

- **Fase III: Ejecución de pruebas del sistema y de aceptación.**

Para esta fase se realizó un análisis sobre algunas herramientas que se encuentran disponibles en el internet para realizar pruebas de aceptabilidad y de funcionalidad.

Etapa 1: Pruebas de código y de carga.

Las pruebas de código se realizaron con la herramienta de pruebas JUnit 3.0 que proporciona un ambiente propicio para realizar las pruebas. (Ver Sección Resultados 4.1. Pruebas de código).

Las pruebas de carga se realizaron levantando agentes de forma consecutiva en el servidor desarrollando una pequeña aplicación en java para que levante de forma concurrencia un número determinado de agentes en el servidor y ver el comportamiento del mismo a la hora de ejecutar las pruebas. (Ver Sección Resultados 4.1. Pruebas de carga).

Etapa 2: Selección de herramientas.

Una vez terminada la decodificación se procedió a realizar las pruebas, comenzando por seleccionar herramientas que permitan hacer pruebas de funcionalidad y factibilidad de la App, en esto se determinó que Apkudo una plataforma para realizar pruebas de ejecución de la aplicación. Además Play Store es una herramienta que permito ver el número de dispositivos que se encuentra corriendo la aplicación. (Ver Sección Resultados Fase cuatro: Pruebas de funcionalidad y de factibilidad. Apartado 4.4 Pruebas de compatibilidad).

2. MÉTODOS DE INVESTIGACIÓN.

- **Método deductivo:** Es utilizada para organizar la información de una forma racional, interrelacionada e interactivamente partiendo de lo general (efectos) buscan la particularidad o la clase del problema, partiendo de que los usuarios no tienen una cultura de ahorro y dando una solución siguiendo los siguientes lineamientos.
 - Lluvia de ideas para tesis.
 - Realización.
 - Solución planteada de la tesis.
 - Desarrollo del anteproyecto.
- **Método documental:** La investigación basada en documentos se dedica a reunir seleccionar y analizar datos que están en forma de documentos producidos por la sociedad para estudiar un fenómeno determinado. Es decir, al que se puede acudir como fuentes o referencias en cualquier momento o lugar sin que se altere su naturaleza o sentido, para que aporte información o rinda cuenta de una realidad o acontecimiento. Con este método se analizó la información escrita sobre el tema de objeto de estudio, de esta forma se realizó en base a la revisión de (documentos, revistas universitarias, artículos científicos, Tesis doctorales, Libros, etc.).
- **Método experimental:** En la investigación del enfoque experimental el investigador manipula una o más variables de estudio para controlar el aumento o disminución de esas variables y su efecto en las conductas observadas. Dicho de otra forma, un experimento consiste en hacer un

cambio en el valor de una variables (variables independientes) y observar su efecto en otra variable (variables dependientes). Esto se lleva a cabo en condiciones rigurosamente controladas, con el fin de describir de que causa se produce situación o acontecimiento particular.

Esto llevo a construir un banco de pruebas que ayudan a sostener el sistema y hacerlo más estable variando las diferentes entradas que podría tener el sistema.

- **Método Inductivo:** Este método fue utilizado partiendo de lo particular del proyecto o causas del mismo hacia la generalidad efectos determinados en el método deductivo. Por tal motivo se considera pertinente el proyecto “DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS MÓVILES ANDROID QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS ECONÓMICOS DE UNA PERSONA”; posicionando al proyecto como una solución en el ámbito social-económico.

3. TÉCNICAS DE RECOLECCION DE INFORMACIÓN.

Durante el desarrollo del presente trabajo de titulación fue necesario utilizar técnicas de recolección de información que sea relevante y consistente que permita afianzar los conocimientos, consultar con expertos para determinar bien los parámetros que intervienen en el presente sistema.

a. Técnica de entrevista.

Con el objetivo de mejorar y aligerar el desarrollo del sistema se procedió a realizar entrevistas a expertos que saben acerca de los procesos contables y que su objetividad en los términos contables sirvieron de ayuda para mejorar el proceso de formulación de requerimientos.

F. RESULTADOS

A continuación se detallan las fases realizadas correspondientes a la metodología mencionada en la sección (E. Materiales y Métodos).

1. Fase uno: Análisis de requerimientos funcionales y no funcionales.

Se determina las entradas y la funcionalidad que tendrá la aplicación, en esta fase se describen las entradas, comportamientos y salidas del sistema.

1.1. Análisis de requerimientos no funcionales

Se hace referencia a las características del sistema que se aplicaran de manera general a todo el sistema.

1.1.1. Análisis de requerimientos funcionales.

Para esta primera fase se realizó el análisis de las entradas de la aplicación las diferentes operaciones que intervienen dentro del sistema.

TABLA I. REQUERIMIENTOS FUNCIONALES

Requerimiento	Descripción del requerimiento.
RF1	El sistema permitirá al usuario agregar ingresos y egresos
RF2	El sistema permitirá al usuario editar ingresos y egresos
RF 3	El sistema permitirá al usuario eliminar ingresos y egresos
RF4	El sistema permitirá al usuario visualizar información de cada ingresos y egresos
RF5	El sistema mostrara al usuario dentro de egresos, descripción, monto, fecha, la periodicidad y categoría.
RF6	El sistema mostrara dentro de ingresos, descripción, monto, fecha, ahorro.

RF7	El sistema definirá automáticamente la fecha de ingreso de un valor.
RF8	El sistema verificara que los datos ingresados sean correctos
RF9	El sistema mediante el agente deliberativo permitirá evaluar los datos y emitir alertas de gastos en su evaluador.
RF10	El sistema mediante el agente deliberativo mostrara las descripción de la categoría de los egresos y como debe ser su gasto.
RF11	El sistema realizara estimación del fin del presupuesto y del gasto diario.

1.1.2. Requerimientos de software.

Se ha evaluado un conjunto de herramientas las cuales servirán para la realización del proyecto y que tendrán una disposición permanente en todo el proceso.

TABLA II. REQUERIMIENTOS DE SOFTWARE

Requerimiento	Descripción del requerimiento.
RS1 Paquete ADT para Android <ul style="list-style-type: none"> • Eclipse + ADT plugin • Android SDK herramientas. • Android plataforma – tool. • La ultima plataforma de android.4w • Imagen sistema. Android para emular • SQLite nativa de Android 	Para el desarrollo de la aplicación también se hará uso del paquete ADT para Android ya que es uno de los sistemas que estará usando. Este paquete incluye Eclipse y ADT plugin. Android SDK, Herramientas, Android Platform-tool. La ultima plataforma Android e imagen sistema. Para emulador.
RS2 Paquete de JADE <ul style="list-style-type: none"> • JADE-LEAP 	Para poder integrar el agente deliberativo al sistema hará uso de la librería Jade-Leap desarrollada en java para el manejo del agente.

1.1.3. Requerimientos de hardware.

Se lista todas las herramientas físicas utilizadas dentro del proyecto.

TABLA III. REQUERIMIENTO DE HARDWARE

Requerimiento	Descripción del requerimiento.
RH1 Computadora laptop	Para el desarrollo de la aplicación móvil se requerirá de una computadora, se a elegido laptop por su portabilidad y movilidad.
RH2 Celular con S.O. Android	Debido a que la aplicación a desarrollar será implementada para dispositivos móviles, se requiere de un celular para realizar las debidas pruebas de funcionamiento.

1.2. Definición de las entradas, roles, prototipos asociados para el agente.

Empecemos definiendo los roles de los dos agentes del caso de estudio: Primero llenamos en una plantilla el esquema de rol del Agente Contable, indicando una descripción de lo que hace y también indicando los protocolos, actividades, permisos y responsabilidades tipo su ciclo de vida y tipo muerte.

1.2.1. Modelo de Rol del Agente

Un rol necesita un conjunto de Permisos que identifique aquellos recursos que puede acceder para poder cumplir con sus responsabilidades. Estos recursos son generalmente fuentes de información en las que puede leer como (Base de Datos), hacer modificaciones y enriquecer bajo ciertas circunstancias.

TABLA IV. MODELO DE ROL DEL AGENTE

Esquema de rol	Agente Contable
<p>Descripción.</p> <p>Este rol involucra que el agente contable mediante los datos de entrada como ingresos y egresos pueda evaluar estos y determinar si es preciso emitir una alerta al usuario para que tenga en cuenta los gastos excesivos que existen en el proceso de egresos de esta misma forma emitirá una alerta del valor diario que puede tener una persona contara con evaluador de porcentajes de egresos excesivos y este evaluara el fin del presupuesto.</p>	
<p>Protocolos y actividades</p> <p>Actividades.</p> <ul style="list-style-type: none"> • Vigilar.- Consiste en observar los ingresos y egresos definidos como entradas del usuario y que para el agente servirán como los creencias. • Evaluar.- Se realizara un diagnóstico de los ingresos y egresos dentro de los parámetros (Ahorro, Monto de Ingresos y Monto de Egresos) para emitir un proceso teniendo especial cuidado en procurar el ahorro deseado por el usuario, en esta parte se evidencia los deseos que cumplirá el agente. <p>Protocolo.</p> <ul style="list-style-type: none"> • Procede.- En base a la evaluación de los diferentes parámetros se realizara. <ul style="list-style-type: none"> ✓ Calculo de valor de gasto diario. ✓ Calcular fin del presupuesto. ✓ Calcular gastos excesivos. ✓ Calcular porcentaje de gastos. ✓ Emisión de alertas para el usuario. 	
<p>Permisos</p> <p>Leer ingresos.</p> <p>Leer egresos.</p>	
<p>Responsabilidades.</p> <p>Agente Inteligente (Vigilar, Evaluar, Proceder)</p>	

1.2.2. Definición de los protocolos.

Son las interacciones entre los roles, básicamente es una actividad que necesita una interacción entre el rol.

TABLA V. DEFINICIÓN DE PROTOCOLOS

Protocolo: Procede
Nombre: Agente Contable
Calcular valor de gasto diario <p>Se evalúa el gasto en base a sus ingresos y egresos, estos una vez evaluados permitirán mostrar el valor de gasto diario para el usuario teniendo en cuenta el porcentaje de ahorro que se desea obtener el usuario.</p>
Calcular fin del presupuesto. <p>En base a la entrada del usuario de los ingreso estos se evaluarán para definir la terminación del mismo teniendo en cuenta valores del supervivencia básicos dentro de un rango de presupuestos.</p>
Calcular gastos excesivos. <p>En base a los pesos que tendrá cada categoría se evaluará que gastos son excesivos e innecesarios para el usuario pueda tomar en cuenta y modificar su gasto.</p>
Calcular porcentaje de gastos. <p>En base a los gastos ingresados por el usuario el agente podrá calcular el porcentaje de cada gasto y su influencia en el presupuesto total.</p>
Emisión de alertas <p>En base a los cálculos realizados por el agente este mismo emitirá alertas y avisos al usuario para que pueda modificar y obtener el ahorro deseado.</p>

1.2.3. Modelo de servicios.

Se identifica las entradas, salidas su precondición y post-condición dentro de su uso en el sistema.

TABLA VI. MODELO DE SERVICIOS

Servicio	Entrada	Salidas	Precondición	Post-Condición
Vigilar	Ingresos Monto Ahorro Fecha Egreso Monto. Categorías.	Notifica a procede	Ingresos \neq null Monto > 0 Ahorro > 0 Fecha \neq null Egresos \neq null Monto > 0 Categorías \neq null	Datos Insuficientes para su evaluación
Evaluar	Ingresos Monto Ahorro Fecha Egreso Monto. Categorías.	Notificar a procede	Ingresos \neq null Monto > 0 Ahorro > 0 Fecha \neq null Egresos \neq null Monto > 0 Categorías \neq null Vigilar = true	No se puede evaluar sin datos, se debe vigilar antes de poder evaluar.
Procede	Notificación es entrantes de Vigilar y	Gasto Diario Ahorro total Fin Presupuesto	Vigilar = true Evaluar = true	Se debe vigilar y evaluar para proceder

	evaluar	Gastos Excesivos Porcentaje de Gastos Alerta Usuario		
--	---------	--	--	--

1.3. Análisis de datos que van a ser almacenados SQLite.

Los datos definidos de entrada son:

- **Ingresos.**

- Descripción: Se deberá definir una representación válida para el ingreso.
- Monto: Este deberá ser el valor real (12.60).
- Fecha: Debe ser la fecha en la cual se obtuvo el dinero.
- Ahorro: debe ser un valor entero el cual puede ir desde 0 a 100 por ciento.

- **Egresos.**

- Descripción: Se deberá definir una representación válida para el egreso o gasto.
- Monto: Este deberá ser el valor real (12,60).
- Fecha: Debe ser la fecha en la cual se realizó el gasto.
- Periodicidad: El sistema mostrara una lista en la cual el usuario escogerá una periodicidad dependiendo de cada que tiempo se realiza ese gasto.
- Categoría: El sistema mostrara un grupo de palabra que describa bien la categoría a la cual pertenece el egreso este puede ser de la siguiente forma, Vivienda.

2. Fase dos: Diseño para la aplicación y el agente deliberativo.

2.1. Prototipos de pantalla

En esta parte se definió la diferente interfaz del usuario con las cuales interactuara el usuario en la aplicación.

- **Pantalla de entrada al sistema:** Mostrara un mensaje de bienvenida al usuario indicándole la usabilidad del sistema.



FIGURA 18. PANTALLA ENTRADA AL SISTEMA

- **Pantalla de Actividades:** Se determina las diferentes actividades con las que cuenta el sistema, se considerara como un menú de entrada al sistema.



FIGURA 19. PANTALLA ACTIVIDADES

- **Pantalla de lista de ingresos y egresos:** Aquí se determinara una lista completa tanto de ingresos como de egresos en pantallas diferentes pero con las mismas características, la cual contara con un menú que permitirá al usuario ya sea agregar, editar, eliminar y ver información e los ítems.



FIGURA 20 PANTALLA DE LISTA DE INGRESOS Y EGRESOS

- **Pantalla para registrar los ingresos y egresos:** Se determinara los diferentes valores que tendrá cada registro tanto de ingreso como de egreso respectivamente.



FIGURA 21. PANTALLA DE REGISTRO DE INGRESOS Y EGRESOS

- **Pantalla de pronóstico:** Se definirá las diferentes opciones que tendrá el agente y la implementación que ayudara al usuario al manejo del mismo.



FIGURA 22. PANTALLA DE PRONÓSTICO

2.2. Modelo de la base de datos.

Se definió un mini modelo de la base de datos nativa de android que permitirá guardar los datos directamente en el dispositivo con soporte SQLite.

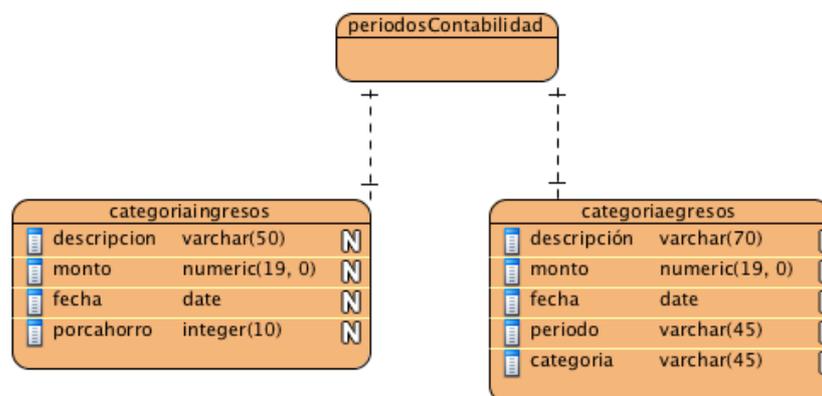


FIGURA 23. MODELO DE LA BASE DE DATOS

2.3. Diseño de creencias deseos e intenciones del agente deliberativo.

La metodología GAIA comprende varios puntos, pero para comenzar se requiere de requerimientos iniciales los cuales no abordaremos en esta sección ya que la metodología de programación extrema nos ayudó a resolver esta parte, estos son los requerimientos funcionales mencionados en la parte anterior, que en agentes se consideraran como creencias.

En esta parte se determina los diferentes diagramas que servirán de apoyo para la creación del agente contable.

2.3.1. Modelo del Agente.

Se determina las acciones o actividades que realizara el agente dentro del sistema.

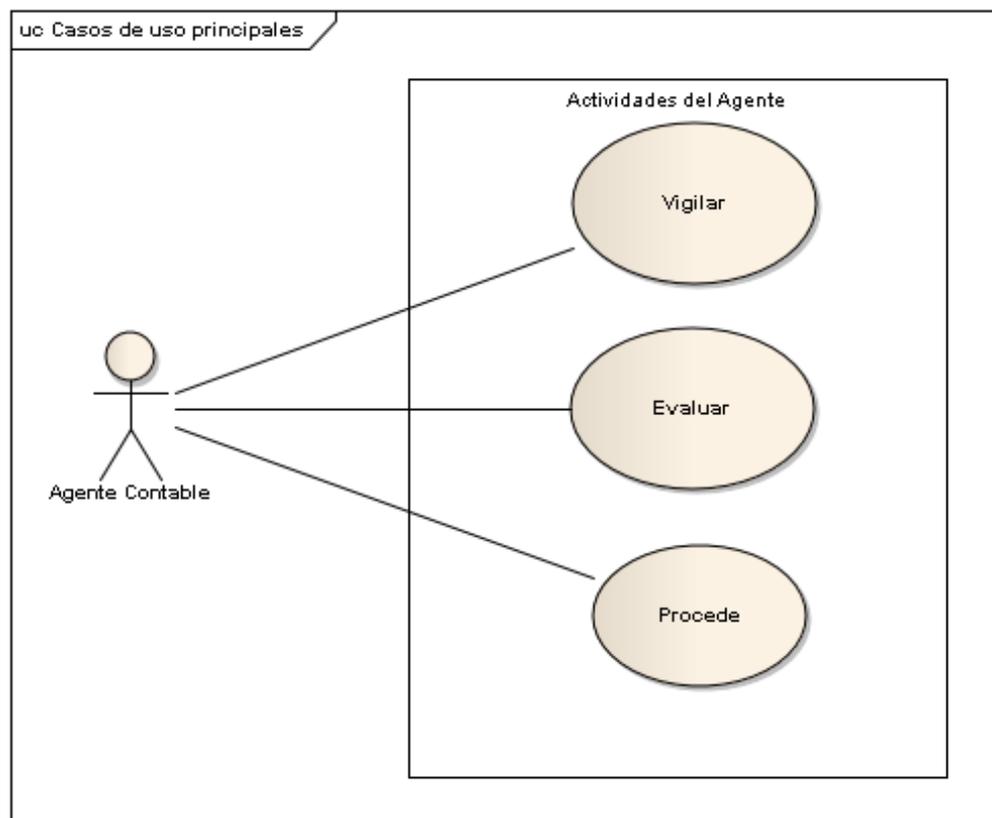


FIGURA 24. MODELO DEL AGENTE INTELIGENTE

2.3.2. Diseño de Interacción.

El modelo de interacción para el agente contable define la funcionalidad del agente y su conexión entre Creencias, Deseos e intenciones y la base de datos además su interacción con el usuario.

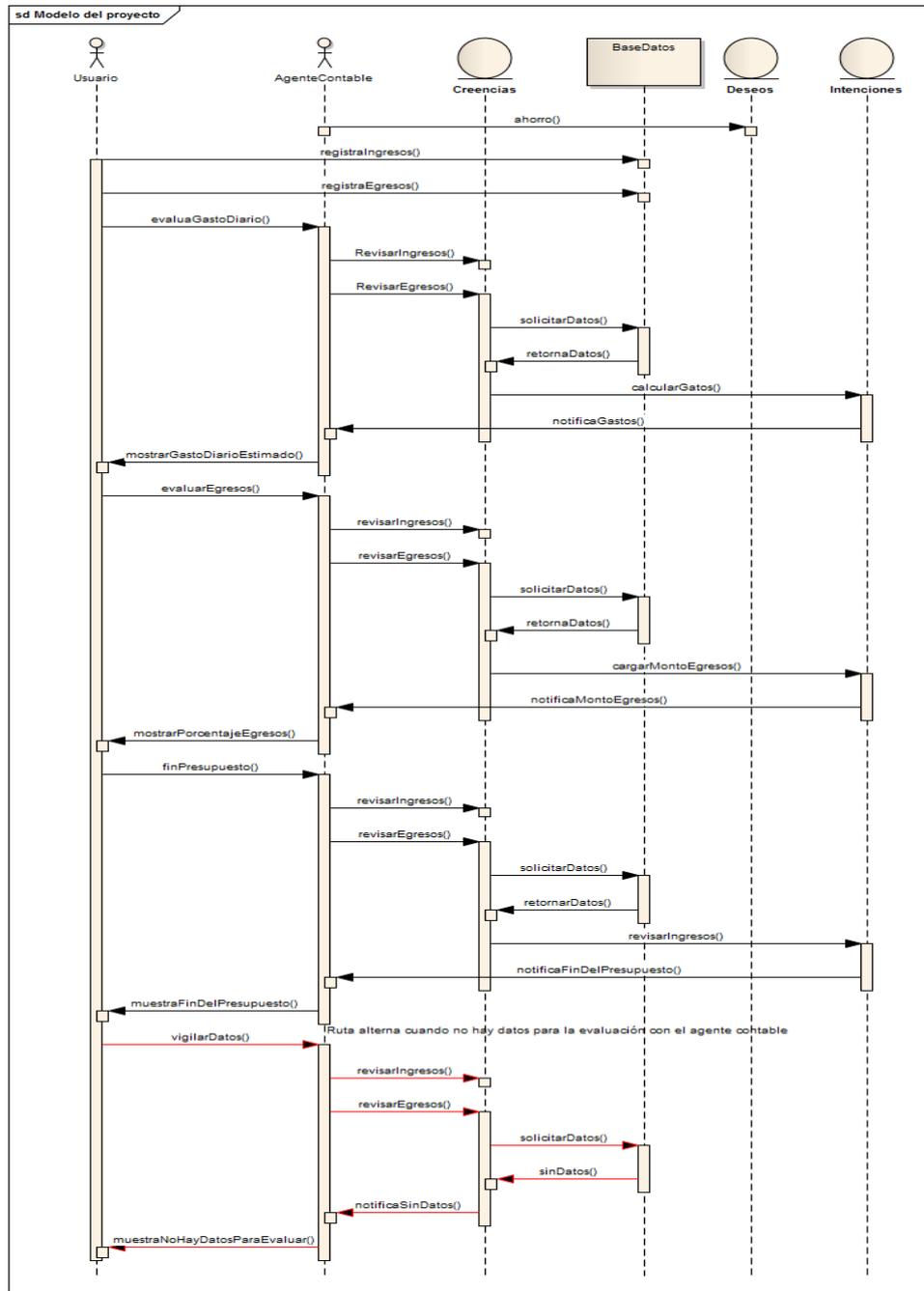


FIGURA 25. DISEÑO DE INTERACCIÓN DEL AGENTE

2.3.3. Arquitectura del agente.

La arquitectura del agente refiere a la comunicación entre los diferentes dispositivos ya que para la utilización del sistema requiere un contenedor principal el cual se encontrara en la nube.

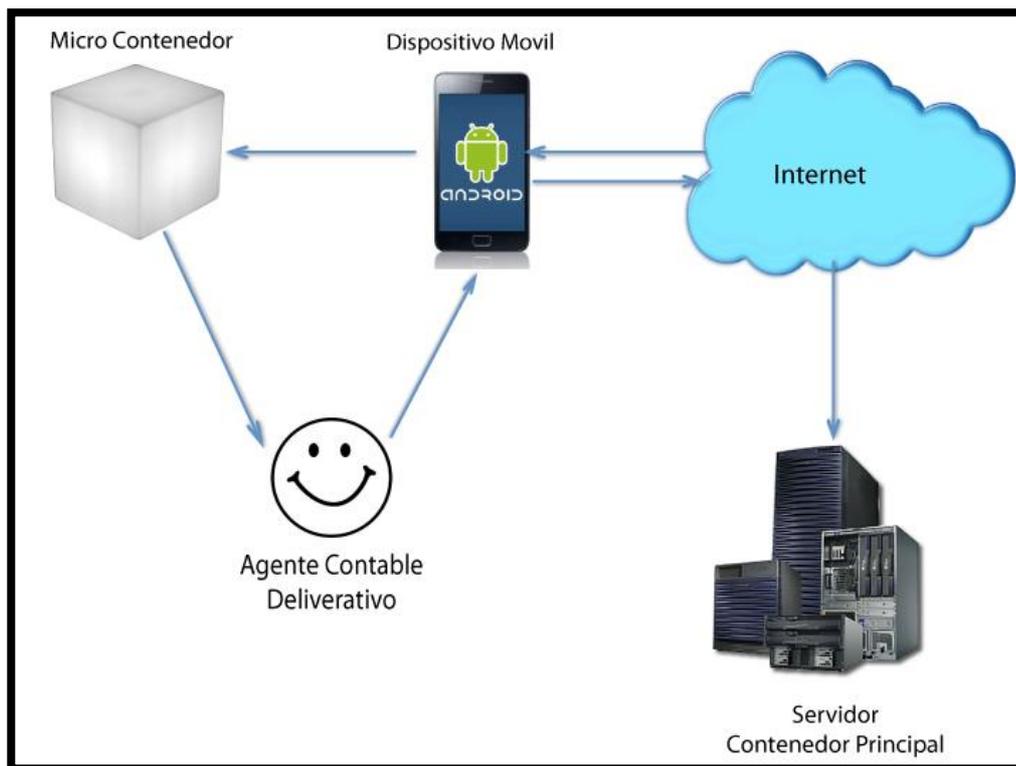


FIGURA 26. ARQUITECTURA DEL AGENTE CONTABLE

3. Fase tres: Desarrollo de aplicación e integración con el agente.

En esta parte se determina la codificación de la aplicación y su integración con el agente inteligente deliberativo.

3.1. Codificación de la aplicación Android.

El desarrollo se lo realizo utilizando el framework de desarrollo eclipse, el cual se integra de una manera eficiente con la programación orientada a dispositivos móviles android.



FIGURA 27. ECLIPSE LUNA

3.1.1. Determinación de paquetes usados.

Se determinó una serie de paquetes que ayudaron a definir un orden específico para dentro ubicar las clases.

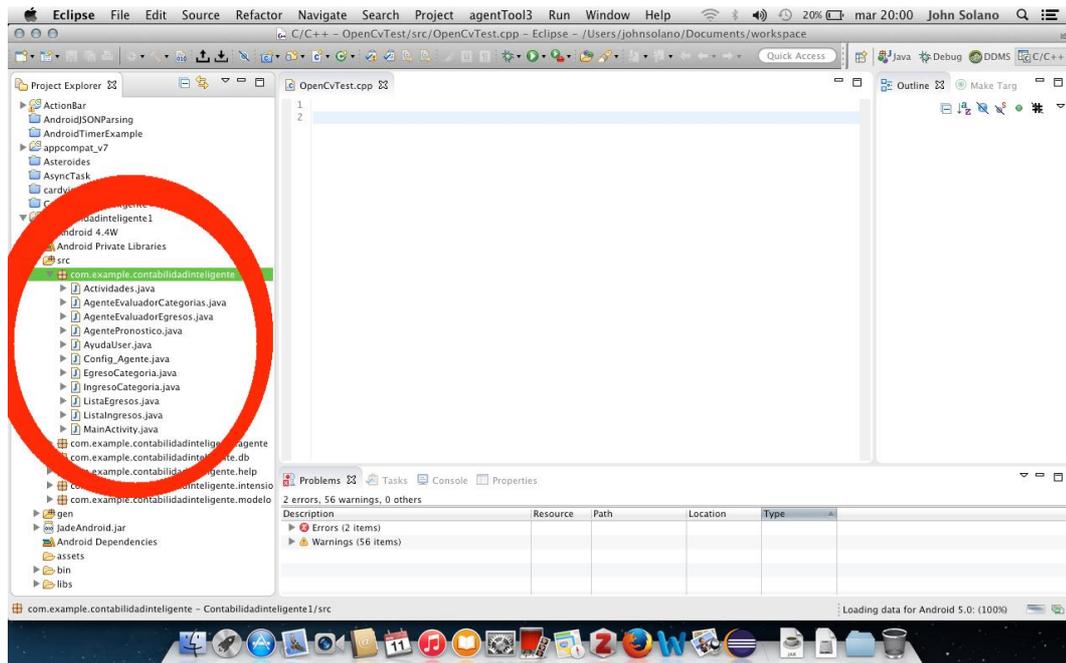


FIGURA 28. DETERMINACIÓN DE PAQUETES ENTORNO DE DESARROLLO

- En el paquete (com.example.contabilidadinteligente) se maneja toda la unión de la interfaz con el modelo de la aplicación.
- En el paquete (com.example.contabilidadinteligente.agente) en este paquete se determinó el agente contable con su interfaz es decir su conexión con la vista del usuario.
- En el paquete (com.example.contabilidadinteligente.db) en esta parte se especificó la base de datos y los diferentes métodos para su implementación como agregar, editar, eliminar registros.
- En el paquete (com.example.contabilidadinteligente.help) en este paquete se determinó las diferentes clases de ayuda a la implementación de la aplicación en esta parte se encuentran tanto nombres del agente, mensajes.
- En el paquete (com.example.contabilidadinteligente.intensiones) este paquete determina las diferentes acciones que realiza el agente dentro del sistema los cuales serán abordados más adelante con más detalle.
- En el paquete (com.example.contabilidadinteligente.modelo) se encuentra la lógica de la aplicación es decir los ingresos y egresos contables.

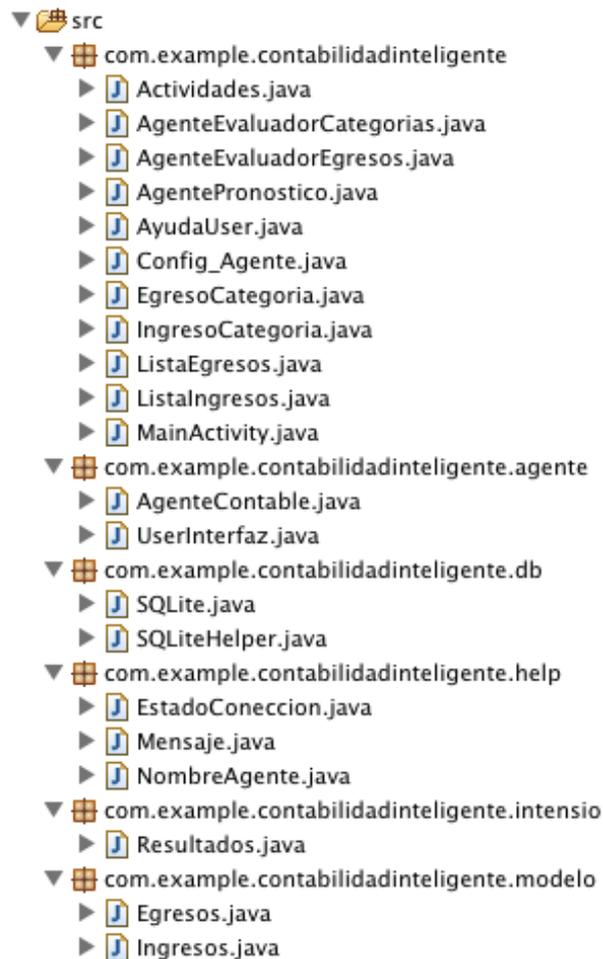


FIGURA 29.PAQUETES DESGLOSADOS CONTABILIDAD INTELIGENTE

3.2. Implementación del agente deliberativo para android.

Definimos las diferentes etapas para la puesta en marcha y unión del agente con la aplicación contable

3.2.1. Instalación de JADE-LEAP para android.

Lo primero que debemos hacer es descargar el complemento para desarrollo de agentes, desde la página oficial de jade donde a más de encontrarse este paquete existe la documentación respectiva e información relevante sobre agentes inteligentes[27].



FIGURA 30. PÁGINA OFICIAL JADE

Una vez dentro de la página oficial procedemos a ubicarnos dentro de la pestaña descargas -> add-ons donde se encuentran todas la librerías de jade en nuestro caso procedemos a descargas el soporte para android el cual es JADE-LEAP [28].

JADE add-ons tested by the JADE team

Note that we recommend to extract the files by using the 'jar -xvf' command rather than the winzip application because some incompatibilities have been reported in the past.

File	File size	Last update	Author	License	Description
Leap	12.9 MB	Jun. 2012	JADE Team	LGPL	The LEAP add-on allows deploying JADE agents on Java enabled cell-phones. Look at the LEAP User Guide for a complete description of how to use it
JADE-OSGi	123 KB	Jun. 2012	JADE Team	LGPL	The JADE-OSGi add-on allows running JADE agents in an OSGi environment. Look at the JADE-OSGi Guide for a complete description of how to use it
WSIG	15.5 MB	Dec. 2013	JADE Board	LGPL	The Web Service Integration Gateway add-on allows automatically exposing agent services registered with the DF as Web Services. Look at the WSIG guide for details
WSDC	4.3 MB	Dec. 2013	JADE Team	LGPL	The Web Service Dynamic Client add-on allows invoking web services without any previous class generation step. Look at the WSDC guide for details

FIGURA 31. ENTORNO DE DESCARGAS ADD-ONS

Luego de esto agregamos el jade android al paquete de desarrollo donde estamos realizando la aplicación, agregando el mismo como una librería.

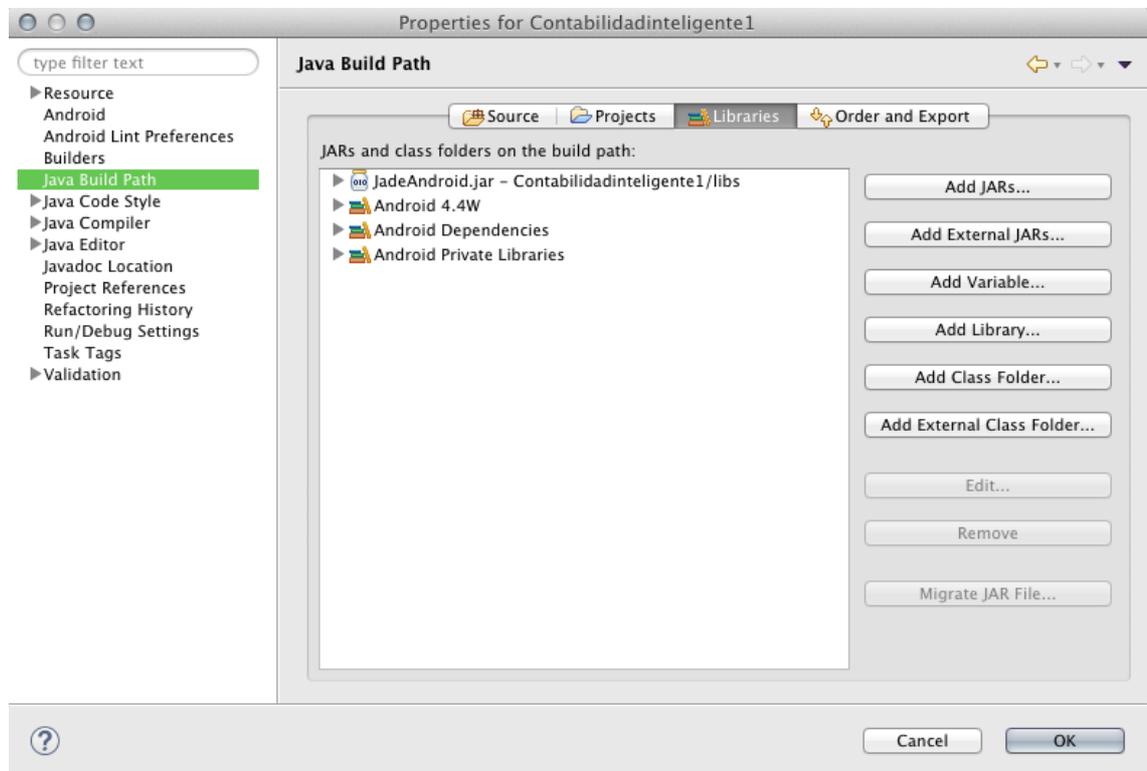


FIGURA 32. IMPLEMENTACIÓN DE LA LIBRERÍA JADEANDROID

Una vez realizado este proceso ya contamos con el soporte para utilizar agentes inteligentes dentro de nuestra aplicación.

3.2.2. Levantamiento del contenedor de agentes.

Este proceso se realiza tanto en el dispositivo como en el servidor ya que como antes se mencionó en la arquitectura de agentes este debe tener un contenedor principal.

3.2.2.1. Levantamiento del contenedor principal.

El proceso comienza primeramente descargando jade ALL de la página oficial[27].

Jade Download

From this page you can now download JADE.
 We suggest to unzip the files by using the 'jar xvf' command rather than the winzip application because some incompatibilities have been reported in the past.

Current version of JADE is 4.3.2 (28/03/2014).

Look at the [ChangeLog](#) for the list of improvements with respect to the previous versions.

Note: All the binaries (lib/.jar) have been generated by using the JDK1.6 compiler.*

JADE	File size	Description of the content
jadeAll	17.8 MB	This file contains all JADE, i.e. it is just composed of the 4 files below. If it is too large for downloading, the 4 files below might be downloaded instead.
jadeBin	2.5 MB	This file contains JADE already compiled and ready to be used, i.e. a set of JAVA archive JAR files.
jadeDoc	12.8 MB	This file contains all the JADE documentation included the Administrator's Guide and and the Programmer's Guide. NOTICE THAT all the documentation is also available on-line.
jadeSrc	2.2 MB	This file contains all the JADE source code.
jadeExamples	520 KB	This file contains the source code of the examples and a simple demo. All the examples and demo must be compiled.

FIGURA 33. ENTORNO DE DESCARGA JADE

Una vez realizado el proceso anterior procedemos a registrar en nuestras variables de entorno para poder usar desde línea de comandos desde jade, este proceso varía dependiendo el sistema operativo, aquí se adjunta un link para el uso de las variables dentro del sistema operativo Mac os Yosemite[29].



FIGURA 34. CONFIGURACIÓN DEL PATH EN MAC OS YOSEMITE

Seguidamente se procede a correr este archivo jade que es una librería de java para lo cual se debe realizar en un terminal por línea de comando de la siguiente manera.

```
java jade.Boot -gui -host 192.168.1.4 -port 1099
```

En donde

Java significa que vamos a correr un archivo con soporte java, es decir para la máquina virtual de java.

Jade.Boot es la librería java que vamos a correr para levantar el contenedor de agentes.

-gui significa que podremos visualizar la pantalla de contenedor de agente esto nos sirve para hacer un monitoreo de los agentes que están activos dentro de nuestra aplicación.

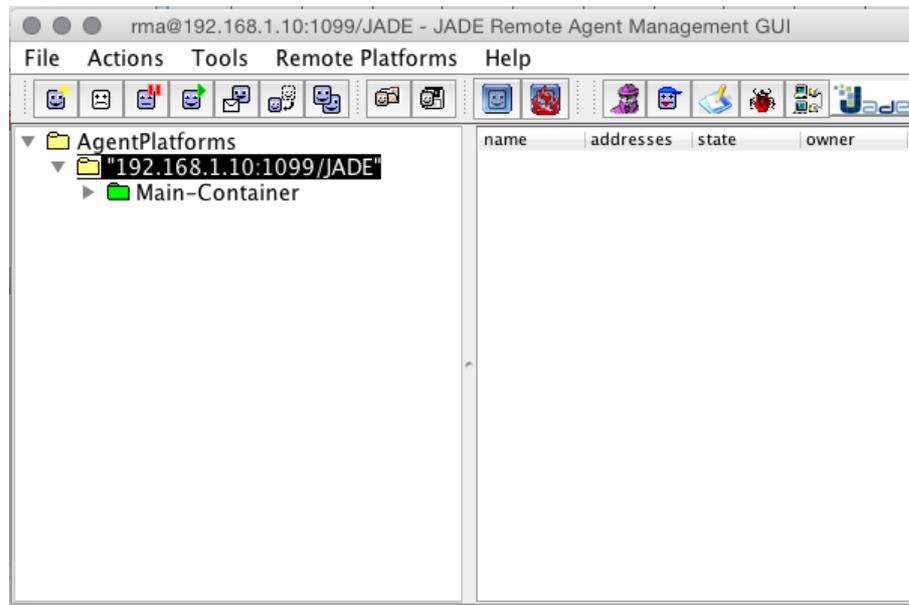


FIGURA 35. ENTORNO GRAFICO JADE CONTENEDOR PRINCIPAL

-host esta parte es imprescindible ya que nos da la opción de definir la IP en la cual queremos correr nuestros agentes es decir la IP publica de conexión de los agentes.

-port aquí definimos el puerto por el cual se realizara la conexión a nuestros agentes.

```

johnsolano -- java -- 80x37
MacBookPro-2:~ johnsolano$ java jade.Boot -host 192.168.1.10 -port 1099
nov 22, 2014 2:29:31 PM jade.Boot parseCmdLineArgs
ADVERTENCIA: WARNING: ignoring command line argument 1099 occurring after agents
specification
nov 22, 2014 2:29:32 PM jade.core.Runtime beginContainer
INFORMACION: -----
This is JADE 4.3.2 - revision 6708 of 2014/03/28 15:19:44
downloaded in Open Source, under LGPL restrictions,
at http://jade.tilab.com/
-----
nov 22, 2014 2:29:32 PM jade.imtp.leap.LEAPIMTPManager initialize
INFORMACION: Listening for intra-platform commands on address:
- jicp://192.168.1.10:1099
nov 22, 2014 2:29:34 PM jade.core.BaseService init
INFORMACION: Service jade.core.management.AgentManagement initialized
nov 22, 2014 2:29:34 PM jade.core.BaseService init
INFORMACION: Service jade.core.messaging.Messaging initialized
nov 22, 2014 2:29:34 PM jade.core.BaseService init
INFORMACION: Service jade.core.resource.ResourceManagement initialized
nov 22, 2014 2:29:34 PM jade.core.BaseService init
INFORMACION: Service jade.core.mobility.AgentMobility initialized
nov 22, 2014 2:29:34 PM jade.core.BaseService init
INFORMACION: Service jade.core.event.Notification initialized
nov 22, 2014 2:29:34 PM jade.mtp.http.HTTPServer <init>
INFORMACION: HTTP-MTP Using XML parser com.sun.org.apache.xerces.internal.jaxp.S
AXParserImpl$JAXPSAXParser
nov 22, 2014 2:29:34 PM jade.core.messaging.MessagingService boot
INFORMACION: MTP addresses:
http://192.168.1.10:7778/acc
nov 22, 2014 2:29:35 PM jade.core.AgentContainerImpl startBootstrapAgents
ADVERTENCIA: Cannot create an agent with no name. Class was -port
nov 22, 2014 2:29:35 PM jade.core.AgentContainerImpl joinPlatform
INFORMACION: -----
Agent container Main-Container@192.168.1.10 is ready.
-----

```

FIGURA 36. LEVANTAMIENTO DEL CONTENEDOR PRINCIPAL

3.2.2.2. Levantamiento del contenedor secundario.

Este contenedor se debe implementar directamente en el dispositivo es decir este será el encargado de emitir la alerta que está activo el contenedor de agente móviles, por tanto debe activarse de igual manera en el contenedor principal.

Este requiere un proceso de desarrollo, una vez instalado el soporte, se procede a definir el nombre del agente más conocido como nickname, hots de destino y el puerto a conectarse el agente que son los datos necesarios.

```
public void levantarContenedorContable(final String nickname, final String host,
    final String port,
    final RuntimeCallback<AgentController> agentStartupCallback) {
    final Properties profile = new Properties();
    profile.setProperty(Profile.MAIN_HOST, host);
    profile.setProperty(Profile.MAIN_PORT, port);
    profile.setProperty(Profile.MAIN, Boolean.FALSE.toString());
    profile.setProperty(Profile.JVM, Profile.ANDROID);

    if (AndroidHelper.isEmulator()) {
        profile.setProperty(Profile.LOCAL_HOST, AndroidHelper.LOOPBACK);
    } else {
        profile.setProperty(Profile.LOCAL_HOST,
            AndroidHelper.getLocalIpAddress());
    }
    profile.setProperty(Profile.LOCAL_PORT, "2000");

    if (microRuntimeServiceBinder == null) {
        serviceConnection = new ServiceConnection() {
            public void onServiceConnected(ComponentName className,
                IBinder service) {
                microRuntimeServiceBinder = (MicroRuntimeServiceBinder) service;
                startContainer(nickname, profile, agentStartupCallback);
            };

            public void onServiceDisconnected(ComponentName className) {
                microRuntimeServiceBinder = null;
            }
        };
        bindService(new Intent(getApplicationContext(),
            MicroRuntimeService.class), serviceConnection,
            Context.BIND_AUTO_CREATE);
    } else {
        startContainer(nickname, profile, agentStartupCallback);
    }
}
```

FIGURA 37. LEVANTAMIENTO DEL CONTENEDOR SECUNDARIO

Seguidamente se procede a levantar el contenedor de agente móvil que está definido como MicroRuntime.

```

private void startContainer(final String nickname, Properties profile,
    final RuntimeCallback<AgentController> agentStartupCallback) {
    if (!MicroRuntime.isRunning()) {
        microRuntimeServiceBinder.startAgentContainer(profile,
            new RuntimeCallback<Void>() {
                @Override
                public void onSuccess(Void thisIsNull) {
                    logger.log(Level.INFO,
                        "Se abre el contenedor de agente...");
                    startAgent(nickname, agentStartupCallback);
                }

                @Override
                public void onFailure(Throwable throwable) {

                    if (pDialog.isShowing()){
                        pDialog.dismiss();
                    }
                    logger.log(Level.SEVERE,
                        "Failed to start the container...");
                }
            });
    } else {
        startAgent(nickname, agentStartupCallback);
    }
}

```

FIGURA 38. PUESTA EN MARCHA DEL CONTENEDOR

3.2.3. Levantamiento del agente deliberativo contable.

Una vez realizados los procesos antes mencionados para el levantamiento del contenedor móvil se procede a levantar el agente inteligente el cual cuenta con el nombre asignado y RuntimeCallBack es el encargado de poner el agente en el contenedor activo.

```

private void startContainer(final String nickname, Properties profile,
    final RuntimeCallback<AgentController> agentStartupCallback) {
    if (!MicroRuntime.isRunning()) {
        microRuntimeServiceBinder.startAgentContainer(profile,
            new RuntimeCallback<Void>() {
                @Override
                public void onSuccess(Void thisIsNull) {
                    logger.log(Level.INFO,
                        "Se abre el contenedor de agente...");
                    startAgent(nickname, agentStartupCallback);
                }

                @Override
                public void onFailure(Throwable throwable) {

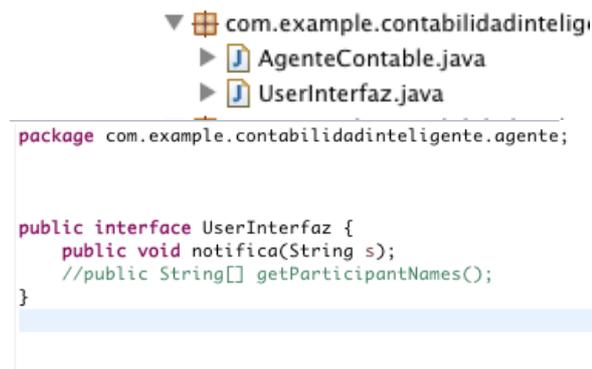
                    if (pDialog.isShowing()){
                        pDialog.dismiss();
                    }
                    logger.log(Level.SEVERE,
                        "Failed to start the container...");
                }
            });
    } else {
        startAgent(nickname, agentStartupCallback);
    }
}

```

FIGURA 39. RUNTIMECALLBACK PARA AGENTES ANDROID

3.2.4. Funcionamiento del agente.

Para el correcto funcionamiento del agente en la plataforma android se debe definir una interfaz que servirá de comunicación entre el usuario y el agente, donde el usuario será el que le da las ordenes al agente esto gracias al paso de mensajes que es implementado tanto en android como en JADE.



```
com.example.contabilidadintelig
├── AgenteContable.java
└── UserInterfaz.java

package com.example.contabilidadinteligente.agente;

public interface UserInterfaz {
    public void notifica(String s);
    //public String[] getParticipantNames();
}
```

FIGURA 40. PASE DE MENSAJES ENTRE USUARIO Y EL AGENTE

3.2.4.1. Implementación de permisos en ANDROIDMANIFEST

Estos permisos se deben establecer para el correcto funcionamiento, primeramente se debe establecer permisos de conexión a internet y de acceso a la tarjeta de red.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

FIGURA 41. PERMISOS EN ANDROIDMANIFEST

3.2.4.2. Levantamiento de servicios para el contenedor de agentes

Se debe realizar el levantamiento jade.android.MicroRuntimeService el cual es un servicio que permite trabajar con el contenedor de android jade-Leap, esto se realiza igualmente dentro del ANDROIDMANIFEST como se puede observar en la Figura 42.

```

<application
    android:allowBackup="true"
    android:icon="@drawable/icono"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <service android:name="jade.android.MicroRuntimeService" />

```

FIGURA 42. MIRCORUNTIMESERVICE SERVICIO DE JADE ANDROID

3.2.5. Definición de la clase agente contable.

Dentro de esta clase se define el comportamiento del agente y sus diferentes interacciones con el usuario esta clase se extiende de la clase Agent de JADE-LEAP además se implementa la interfaz de Usuario.

Como esta clase extiende de Agent se puede utilizar todos los métodos de agentes como:

Setup() que sirve para inicializar el agente en esta parte se establece todos los parámetros de inicio del agente entre estos la interfaz, los comportamientos y el lanzamiento de las diferentes orden del agente.

TakeDown() que es muerte del agente una vez que realice sus actividades se puede definir, que queremos que haga antes de la muerte del agente.

De igual forma se define los comportamientos que tendrá el agente para el propósito del agente contable se definió los siguientes comportamientos:

- AGENTEDELIBERATIVO.- Es el comportamiento de lanzamiento de la primera orden del agente en esta se define el nombre del agente y algún saludo de bienvenida.
- EVALUACATEGORIAS.- Este comportamiento define el evaluador de categorías las acciones que realiza en esta parte se detallaran más adelante.
- COMOAHORRAR.- Este comportamiento da las pautas para que el usuario pueda ahorrar una meta planteada con anterioridad.
- EVALUAGASTOS.- El comportamiento evalúa los gastos que realiza el usuario del total de su presupuesto.

- FINPRESUPUESTO.- Este determina una estimación acercada del final del presupuesto total del usuario.

Notifica.- este método extiende de la interfaz y es la comunicación entre el usuario y el agente ayuda además a que el agente se pueda actualizar en cualquier momento que es usuario necesite del mismo.

```
public class AgenteContable extends Agent implements UserInterfaz{
    private Resultados resultados;
    private Context context;
    private static final long serialVersionUID = -4870059023120650732L;
    private Logger logger = Logger.getJADELogger(this.getClass().getName());
    public void setup() {
        Object[] args = getArguments();
        if (args != null && args.length > 0) {
            if (args[0] instanceof Context) {
                context = (Context) args[0];
            }
        }
        registerO2AInterface(UserInterfaz.class, this);
        resultados = new Resultados(context);
        resultados.getAlertas(context);
        Intent abrirPronostico = new Intent();
        abrirPronostico.setAction("jade.conta.pronostico");
        abrirPronostico.putExtra("alertas", resultados.getAlertas());
        abrirPronostico.putExtra("sugerencias", resultados.getSugerencias());
        abrirPronostico.putExtra("color", resultados.getColor());
        logger.log(Level.INFO, "Sending broadcast " + abrirPronostico.getAction());
        context.sendBroadcast(abrirPronostico);
    }
}
```

FIGURA 43. IMPLEMENTACIÓN DEL AGENTE DELIBERATIVO

3.2.6. Definición de las diferentes acciones que realizara el agente.

Se determina las actividades que realiza el agente para satisfacer las necesidades contables propuestas para lograr el deseo, es decir el ahorro.

Entre las cuales tenemos.

Nombre del agente. El cual retorna el nombre del agente que esta online.

```

public void nombre_Agente(Context c, String a) {
    Intent emision = new Intent();
    emision.setAction("jade.conta.name");
    emision.putExtra("name", "Mi nombre es" + a);
    logger.log(Level.INFO, "enviando: " + emision.getAction());
    c.sendBroadcast(emision);
}

```

FIGURA 44. ACCIÓN NOMBRE DEL AGENTE

Gastos diarios, determina una revisión tanto de ingresos como de egresos que se encuentran en la base de datos para seguidamente ser evaluador y poder definir cuánto dinero puede gastar aproximadamente el usuario diariamente.

```

public void gasto_Diario(Context c) {
    MDB = new SQLite(c);
    getEvaluarFinPresupuesto();
    if (MDB.recuperarIngresos().size() == 0) {
        Intent emision = new Intent();
        emision.setAction("jade.conta.gasto");
        emision.putExtra(
            "gasto_diario",
            "El valor de gasto diario no se puede evaluar por "
            + "que usted no tiene ingresos, trate de buscar una fuente de ingresos ");
        logger.log(Level.INFO, "enviando: " + emision.getAction());
        c.sendBroadcast(emision);
    } else {
        double montoIngresos = 0;
        double montoEgresos = 0;
        double montoAhorro = 0;
        double ahorroTotal = 0;
        double vGasto = 0;
        for (int i = 0; i < MDB.recuperarIngresos().size(); i++) {
            montoIngresos = montoIngresos
                + MDB.recuperarIngresos().get(i).getMonto();
        }
        if (MDB.recuperarEgresos().size() == 0) {
            montoEgresos = 0;
        } else {
            for (int i = 0; i < MDB.recuperarEgresos().size(); i++) {
                montoEgresos = montoEgresos
                    + MDB.recuperarEgresos().get(i).getMonto();
            }
        }
        for (int i = 0; i < MDB.recuperarIngresos().size(); i++) {
            montoAhorro = montoAhorro
                + MDB.recuperarIngresos().get(i).getAhorro();
        }
        ahorroTotal = montoIngresos * (montoAhorro / 100);
        vGasto = (montoIngresos - montoEgresos - ahorroTotal) / dias;
        formatoAhorro = new Formatter();
        formatoGasto = new Formatter();
        formatoAhorro.format("%-6.2f", ahorroTotal);
        formatoGasto.format("%-6.2f", vGasto);
        Intent emision = new Intent();
    }
}

```

FIGURA 45. ACCIÓN GASTOS DIARIO.

Ideas de ahorro. Devuelve algunas ideas de cómo el usuario pudiese ahorrar.

```

public void comoAhorrar(Context context) {
    Intent comoAhorra = new Intent();
    comoAhorra.setAction("jade.conta.comoahorro");
    comoAhorra.putExtra("ideasAhorro", IDEAS_AHORRO);
    logger.log(Level.INFO, "Sending broadcast " + comoAhorra.getAction());
    context.sendBroadcast(comoAhorra);
}

```

FIGURA 46. ACCIÓN COMO AHORRO

Retorno de categorías de egreso. Esto nos retorna todas las categorías de los egresos disponibles en el sistema.

```

public void retornaCategoriaEgreso(Context c) {
    Intent evaluaCategorias = new Intent();
    evaluaCategorias.setAction("jade.conta.categorias");
    evaluaCategorias.putExtra("listaCategorias", LISTA_CATEGORIAS);
    evaluaCategorias.putExtra("listaPesos", PESO_CATEGORIAS);
    logger.log(Level.INFO,
        "Sending broadcast " + evaluaCategorias.getAction());
    c.sendBroadcast(evaluaCategorias);
}

```

FIGURA 47. ACCIÓN RETORNA CATEGORÍAS

Evaluación de gastos. Evalúa los egresos del usuario y devuelve los porcentajes de gastos del usuario sobre el ingreso total del mismo.

```

public void evaluacionGasto(Context c) {
    MDB = new SQLite(c);
    final double porcentajeTotal = 100;
    double totalIngresos = 0;
    int resultado[] = new int[MDB.recuperarEgresos().size()];
    String nameEgreso[] = new String[MDB.recuperarEgresos().size()];

    for (int i = 0; i < MDB.recuperarIngresos().size(); i++) {
        totalIngresos = totalIngresos
            + MDB.recuperarIngresos().get(i).getMonto();
    }
    Log.d("total Ingreso: ", String.valueOf(totalIngresos));

    for (int i = 0; i < MDB.recuperarEgresos().size(); i++) {
        resultado[i] = (int) ((porcentajeTotal * MDB.recuperarEgresos()
            .get(i).getMonto()) / totalIngresos);
        nameEgreso[i] = MDB.recuperarEgresos().get(i).getTipo();
    }

    Intent evaluaEgresos = new Intent();
    evaluaEgresos.setAction("jade.conta.evaluaGastos");
    evaluaEgresos.putExtra("nombreResultados", nameEgreso);
    evaluaEgresos.putExtra("listaResultados", resultado);
    logger.log(Level.INFO, "Sending broadcast " + evaluaEgresos.getAction());
    c.sendBroadcast(evaluaEgresos);
}

```

FIGURA 48. ACCIÓN EVALUACIÓN DE GASTOS

Alertar. Se terminó algunas alertas que emitirá el agente después de hacer la evaluación de los ingresos y egresos al sistema, así mismo estas servirán para determinar la finalización del presupuesto.

```
public void finPresupuesto(Context c) {
    getFinPresupuesto();
    Intent evaluaFinPresupuesto = new Intent();
    evaluaFinPresupuesto.setAction("jade.conta.finPresupuesto");
    evaluaFinPresupuesto.putExtra("finPresupuesto", finPresupuesto);
    logger.log(Level.INFO,
        "Sending broadcast " + evaluaFinPresupuesto.getAction());
    c.sendBroadcast(evaluaFinPresupuesto);
}
```

FIGURA 49. ACCIÓN FIN DEL PRESUPUESTO

Una vez realizado todo este proceso se puede determinar que se cumplió con la fase de desarrollo y se procede a la fase de pruebas para ver la funcionalidad y utilizad de los agenten inteligentes dentro de la plataforma android.

4. Fase cuatro: Pruebas de funcionalidad y de factibilidad.

En esta fase se realizó pruebas código, carga con el servidor y de funcionalidad tanto en dispositivos emulados como en dispositivo físico con lo cual determinaremos la funcionalidad de la aplicación puesta en marcha.

4.1. Pruebas de Código

Para Las pruebas de código se ha realizado con la herramienta JUnit 3.0 de java, seguidamente se listara las pruebas de código realizadas a las clases activity y principales del proyecto contabilidad inteligente.

TABLA VII PRUEBA MAIN ACTIVITY

Clase: MainActivity	Descripción: esta clase es donde se presenta el mensaje de bienvenida.
Pruebas: Se ha realizado pruebas de inicialización, SQLITE, Shareference.	Resultados: testInicio()

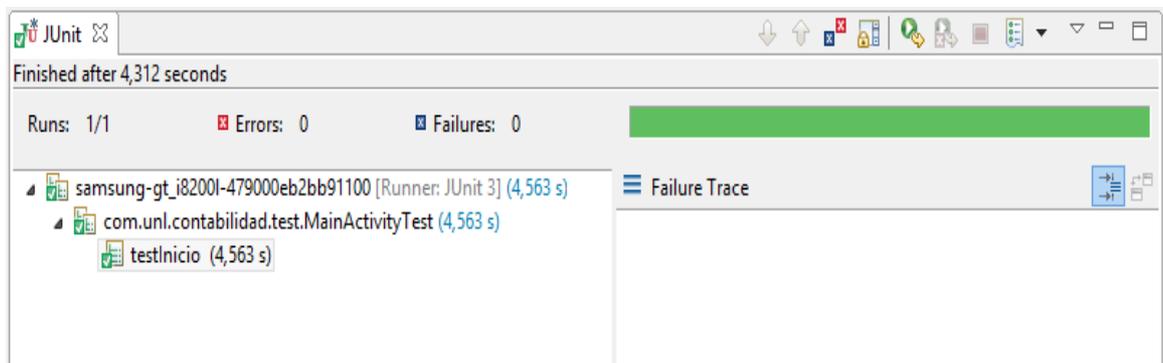


FIGURA 50 PRUEBA MAINACTIVITY

TABLA VIII PRUEBA ACTIVIDADES

Clase: Actividades	Descripción: En esta clase es donde se encuentran todas las actividades contables, además aquí podemos apreciar el levantamiento del contenedor de agentes tanto en el dispositivo como en el contenedor principal.
Pruebas: Se ha realizado pruebas de inicialización de variables y de que no existan datos nulos	Resultados: testObjetosNulos()

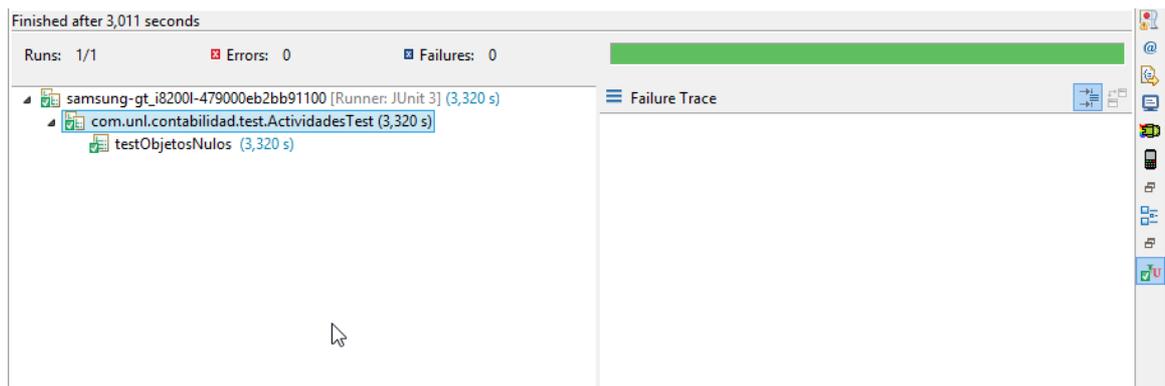


FIGURA 51 PRUEBA ACTIVIDADES

TABLA IX PRUEBA LISTA DE INGRESOS

Clase: ListaIngresos	Descripción: En esta clase se puede apreciar la lista de todos los ingresos que se han registrado en el sistema.
Pruebas: Se ha realizado pruebas de inicialización, SQLITE y Lista vacía o nula	Resultados: testObjetosInicializacion()

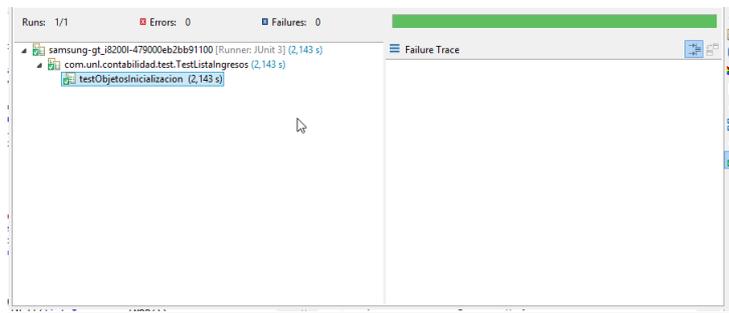


FIGURA 52 PRUEBA LISTA INGRESOS

TABLA X PRUEBA LISTA DE EGRESOS

Clase: ListaEgresos	Descripción: En esta clase se puede apreciar la lista de todos los egresos que se han registrado en el sistema.
Pruebas: Se ha realizado pruebas de inicialización, SQLITE y Lista vacía o nula	Resultados: testInicio()

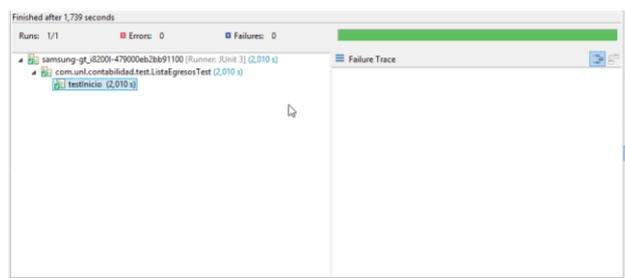


FIGURA 53 PRUEBA LISTA EGRESOS

TABLA XI PRUEBA DE INGRESO CATEGORIA

Clase: IngresoCategoria	Descripción: En esta clase se puede apreciar el formulario para realizar el registro de los ingresos contables.
Pruebas: Se ha realizado pruebas de inicialización, SQLITE y datos nulos.	Resultados: testObjetosinicializacion ()

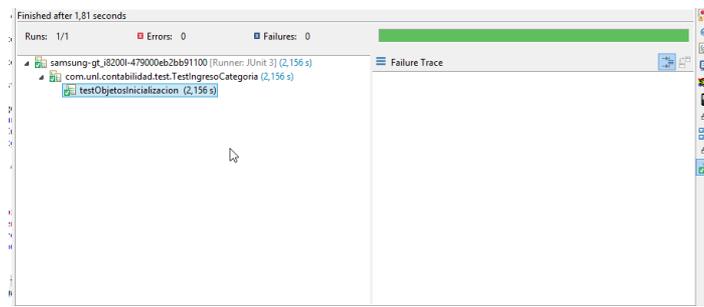


FIGURA 54 PRUEBA INGRESO CATEGORÍA

TABLA XII PRUEBA DE EGRESO CATEGORIA

Clase: EgresoCategoria	Descripción: En esta clase se puede apreciar el formulario para realizar el registro de los egresos contables
Pruebas: Se ha realizado pruebas de inicialización, SQLITE, y datos nulos.	Resultados: testObjetosinicializacion ()

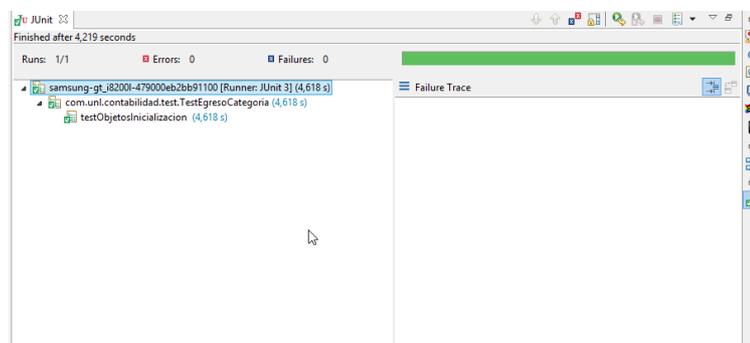


FIGURA 55 PRUEBA EGRESO CATEGORÍA

TABLA XIII PRUEBA CONFIGURACIÓN DEL AGENTE

Clase: Config_Agente	Descripción: Esta clase sirve para configurar el agente, su ip y su puerto de enlace al contenedor principal.
Pruebas: Se ha realizado pruebas de inicialización y Sharepreference.	Resultados: <u>testObjetosinicializacion ()</u>

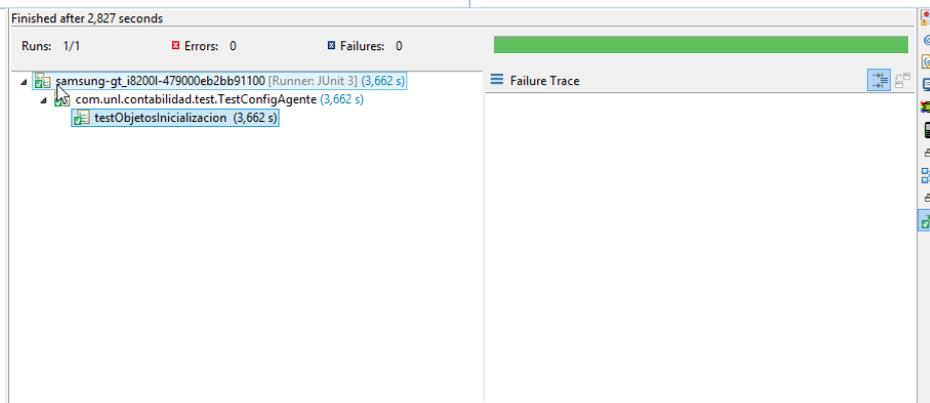


FIGURA 56 PRUEBA CONFIGURACIÓN AGENTE

4.2. Pruebas de Carga

Para las pruebas de carga se realizaron, levantando agentes de forma consecutiva hasta ver el soporte de la pc que funcionara como contenedor principal, el cual será el encargado de contener todos los agentes.

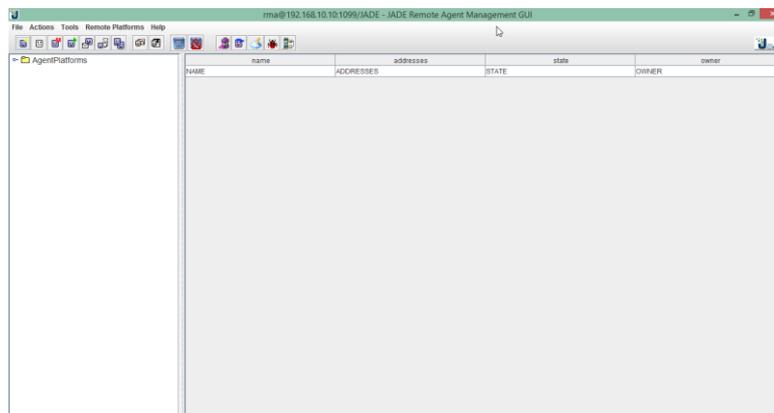


FIGURA 57 CONTENEDOR DE AGENTES

Se desarrolló una aplicación en el java que con la ayuda de la biblioteca JADE.

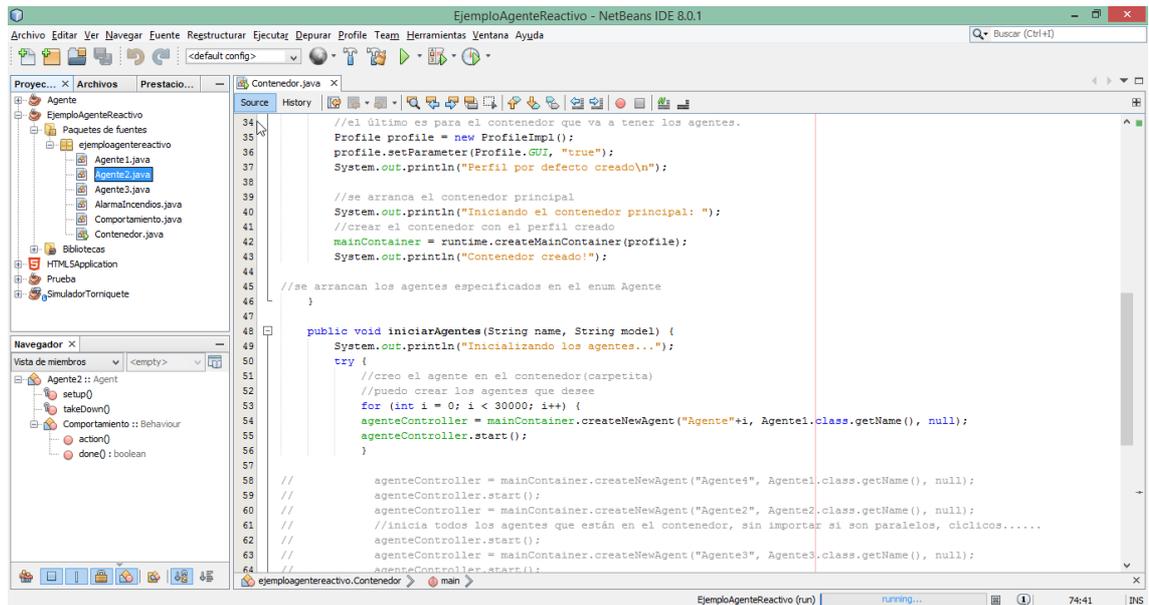


FIGURA 58 DESARROLLO DE APLICACIÓN PARA PRUEBAS DE CARGA

Una vez codificado el agente y el contenedor de agentes se procedió a realizar las pruebas, las pruebas se realizaron en un computador CORE I3 con RAM de 8 GB y un disco de 500GB.

Se realizaron tres secciones de prueba dando como resultado los siguientes datos.

Comportamientos en base a número de agentes contenidos dentro del contenedor principal.

TABLA XIV NÚMERO DE PETICIONES

Normal	10000
Saturado	45000
Deficiente	60000

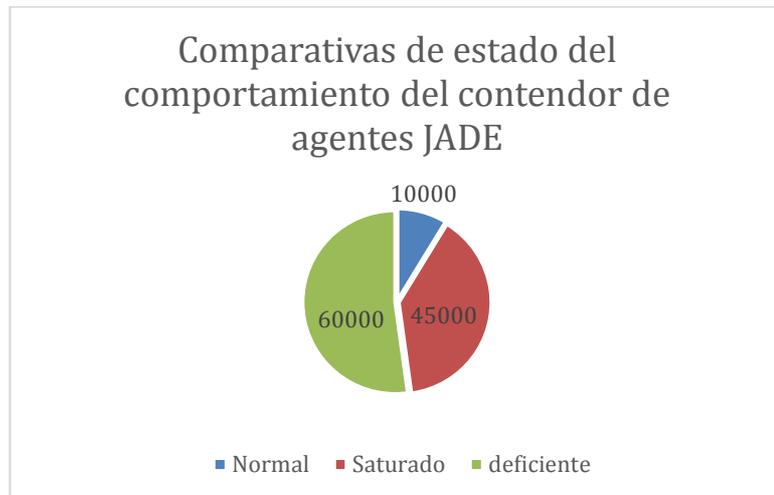


FIGURA 59 . COMPARATIVA DE COMPORTAMIENTOS

Con el comportamiento normal la computadora no se saturó, y su funcionamiento fue el adecuado esto se pudo apreciar en el administrador de tareas de Windows.

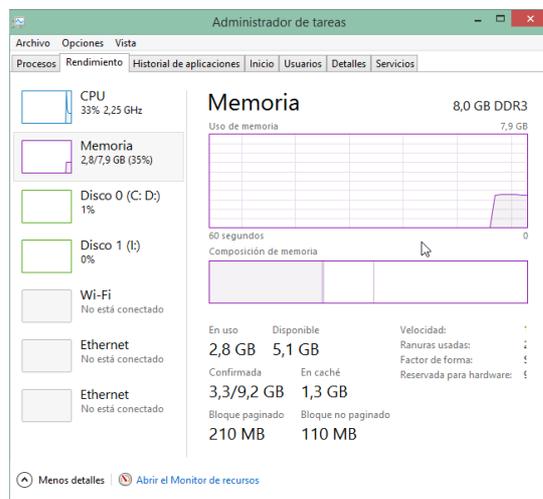


FIGURA 60 ESTADO DE MEMORIA Y PROCESADOR PRUEBA NORMAL

En el comportamiento saturado se realizó con el levantamiento de 30000 agentes dentro del contenedor principal, dando como resultado de igual manera aceptable ya que el funcionamiento del computador fue normal. Lo que sí se pudo apreciar es que ocupó más recursos del computador.

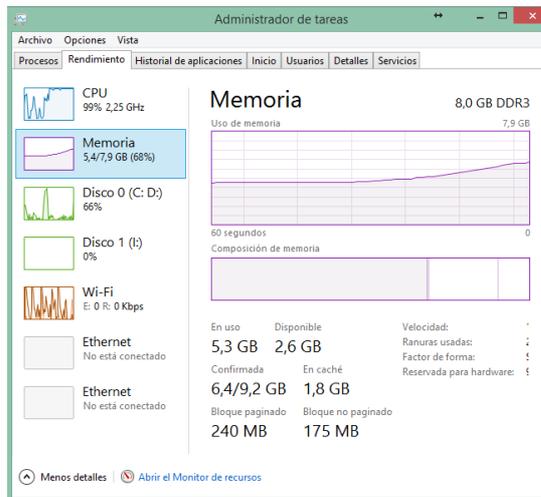


FIGURA 61 ESTADO DE MEMORIA Y PROCESADOR PRUEBA SATURADO

El comportamiento deficiente se realizó con el levantamiento de 60000 agentes dentro del contenedor principal, una vez levantado el contenedor se pudo apreciar que una vez que llego a crear los 47000 agentes la computadora dejo de funcionar, es decir que su funcionamiento ya no fue el adecuado y se tuvo que proceder a reiniciar el computador ya que el mismo no respondía.

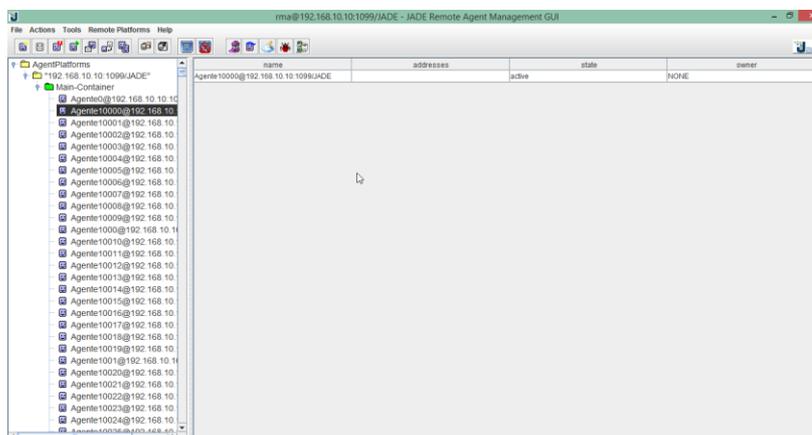


FIGURA 62 PRUEBA CON RESULTADOS SATURACIÓN

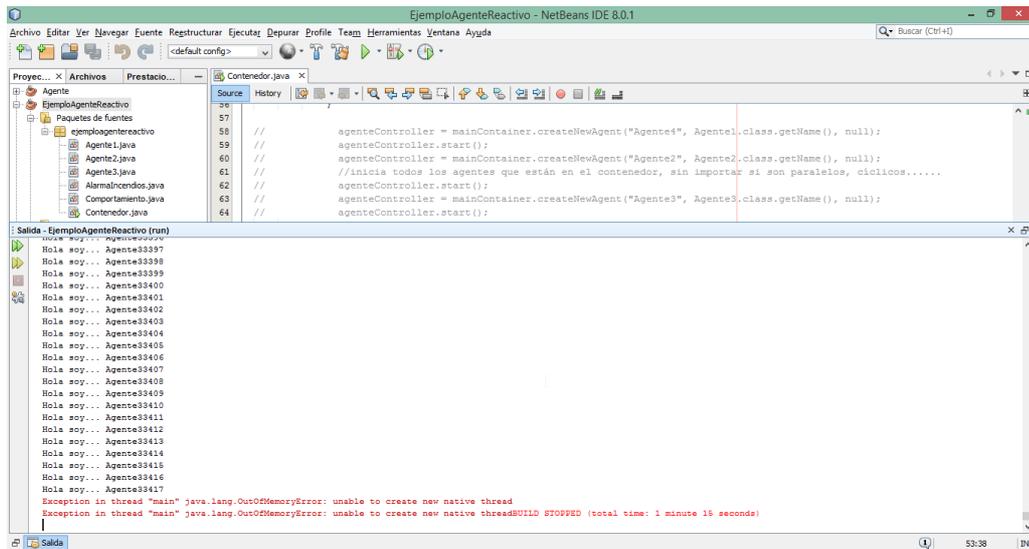


FIGURA 63 PRUEBA RESULTADOS SATURADO

4.3. Ejecución en un emulador apropiado.

La funcionalidad de la aplicación en dispositivos emulados fue aceptable ya que la aplicación se ejecutó de una manera apropiada.

Versión de android: 4.0

Procesador de emulador: ARM (armeabi-v7a)



FIGURA 64.EMULADOR ANDROID

4.3.1. Pantalla de bienvenida.

Se puede evidenciar la pantalla de bienvenida esta da un perspectiva de la aplicación.



FIGURA 65 EMULADOR PANTALLA PRINCIPAL

4.3.2. Pantalla de actividades de la aplicación.

Aquí se encuentran todas las actividades con las que cuenta la aplicación, el usuario es el encargado de realizar sus ingresos y egresos, para que el agente inteligente ayude a mitigar sus gastos y poder obtener un ahorro significativo de sus ingresos económicos.



FIGURA 66. EMULADOR PANTALLA ACCIONES

4.3.3. Lista de ingresos y lista de egresos

Se muestra la lista completa de todos los ingresos y egresos económicos del usuario estos son obtenidos desde la base de datos para su visualización así como para su persistencia, Además esta cuenta con un menú para que el usuario pueda agregar, eliminar, modificar y visualizar la información de cada uno de estos.

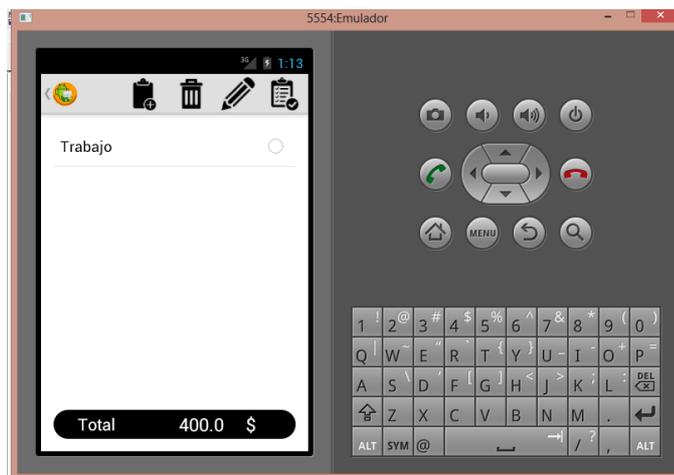


FIGURA 67. EMULADOR PANTALLA DE INGRESOS Y MENÚ

4.3.4. Agregar nuevos ítem a la lista de ingresos y egresos

La aplicación cuenta con formularios para ingresos y egresos económicos los cuales son persuasivos ya que informan al usuario de lo que tiene que ingresar al sistema y que información es de relevancia para su uso posterior.



FIGURA 68. EMULADOR PANTALLA INSERTAR INGRESO

4.3.5. Ejecución del agente

El agente inteligente presenta un mensaje al iniciar ya que este debe conectar con el servidor principal de agentes el cual va a alojar a todas las conexiones del agente.



FIGURA 69. EMULADOR CONEXIÓN CON EL CONTENEDOR PRINCIPAL

Seguidamente el agente una vez conectado con el contenedor principal procede a mostrar toda su estructura.



FIGURA 70. EMULADOR PUESTA EN MARCHA DEL AGENTE

Se puede apreciar que el agente cumple con todos los requerimientos expuestos al inicio de este proyecto y que cada uno de ellos se ejecuta mediante el agente inteligente deliberativo.

4.4. Pruebas de Compatibilidad.

Para esta fase de pruebas se ha hecho uso de la plataforma web apkudo que proporciona una solución robusta que permite conocer de forma acertada en que plataformas de android correrá de manera adecuada la aplicación.



FIGURA 71. PRUEBAS APKUDO

Se procede a realizar el registro en dicha página con esto se nos activa la opción para cargar nuestro apk y probar la funcionalidad en las diferentes plataformas con las que cuenta android.

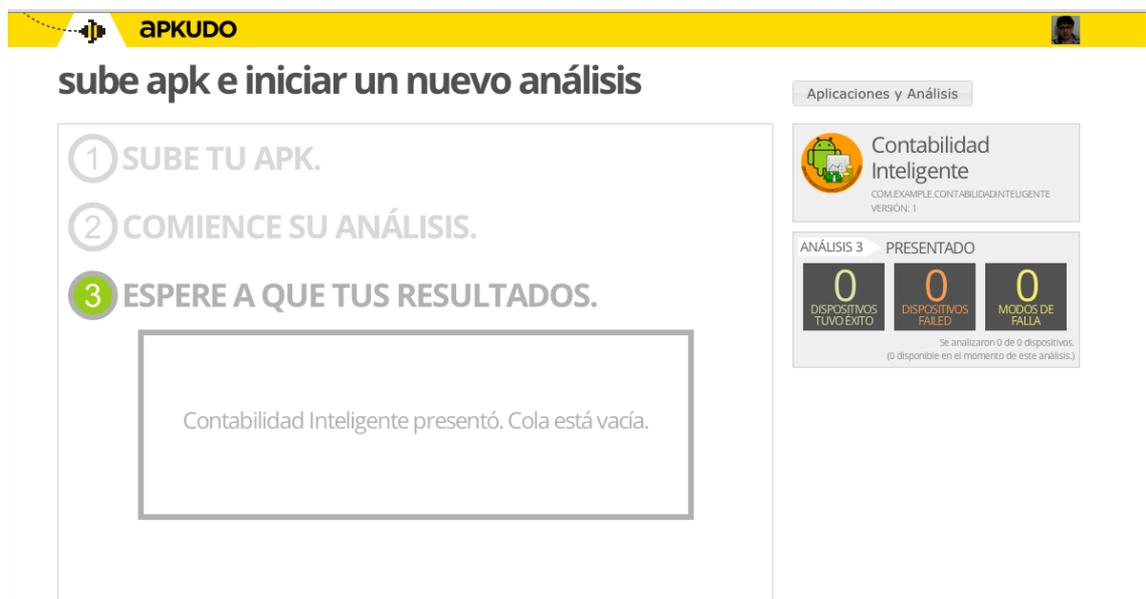


FIGURA 72. PUESTA EN MARCHA DEL ANÁLISIS APKUDO

Una vez analizada la aplicación podemos darnos cuenta en cuantos dispositivos puede ser ejecutada de manera apropiada.



FIGURA 73. RESULTADO DE PRUEBAS DE LA APLICACIÓN

Como se puede observar en la imagen anterior la aplicación tuvo éxito ejecutándose en 24 dispositivos de 74 probados esto se debe a que el soporte tanto de agentes como de la base de datos y las nuevas tecnologías nos obligan a siempre orientarnos a lo moderno es por eso que para la ejecución correcta de esta aplicación se solicita que el usuario tenga por lo menos un celular con la versión de android mínima de 3.0

4.5. Ejecución de aplicación en un dispositivo físico que tenga plataforma Android.

Las pruebas realizadas en los dispositivos físicos dieron como resultado, éxito en él ya que su ejecución fue satisfactoria.

Las pruebas se realizaron en un dispositivo con las siguientes características.

Numero de modelo: Samsung GT-I8200L

Versión de android: 4.2.2 JELLY BEAN



FIGURA 74 TELÉFONO DE PRUEBA

4.5.1. Play Store.

CONTABILIDAD INTELIGENTE
com.unl.contabilidadinteligente Ver en Google Play Store
PUBLICADA Anular publicación de aplicación Enviar actualización

Estadísticas
Valoraciones y opiniones
Errores y ANRs
Sugerencias de optimización

APK
Ficha de Play Store
Precio y distribución
Productos integrados en la aplicación
Servicios y APIs

FICHA DE PLAY STORE

INFORMACIÓN DEL PRODUCTO Para publicar la aplicación, debes rellenar los campos marcados con *

Español (Latinoamérica) – es-419 Gestionar traducciones

Título*
Español (Latinoamérica) – es-419
24 de 30 caracteres

Descripción breve*
Español (Latinoamérica) – es-419
80 de 80 caracteres

Descripción completa*
Español (Latinoamérica) – es-419

FIGURA 75 ANÁLISIS CONTABILIDAD INTELIGENTE PLAY STORE

La aplicación fue subida con éxito al repositorio principal de aplicaciones Play store de google.

Seguidamente se procedió a realizar las correcciones respectivas para subir la aplicación ya que requiere de algunas tareas a realizar para su correcta puesta en marcha dentro de la plataforma google Play.

- Obtener una llave de la aplicación.
- Definir algunos diseños para su presentación
- Determinar nombres a los paquetes de forma apropiada.

4.5.2. Estadísticas Play Store



FIGURA 76 ESTADÍSTICAS PLAY STORE

Dentro de Play store contamos con un espacio donde podemos observar algunas estadísticas como por ejemplo número de descargas y número de dispositivos en los cuales está corriendo y está siendo utilizado.

TABLA XV ESTADÍSTICAS DE INSTALACIÓN

Número de instalaciones	Superior a 50
Numero de dispositivos que está corriendo	Superior a 11

4.5.3. Instalaciones actuales por dispositivo.

En la gráfica posterior se muestra, en que versiones de android ha sido instalada la aplicación contabilidad inteligente.



FIGURA 77 INSTALACIÓN DE DISPOSITIVOS ACTUAL

Dando como resultado que la mayoría de usuarios está instalando la aplicación es en dispositivos con android 4.4

G. DISCUSIÓN

1. Desarrollo de la propuesta alternativa

Para llevar a cabo la realización del presente proyecto de titulación se evaluaron cada uno de los objetivos específicos planteados, indicando la forma en la que se fueron cumpliendo.

Los cuales se detallan a continuación.

- **Objetivo Específico 1:** Analizar los requerimientos de ingresos y egresos que intervienen dentro de un control de gastos.

En esta primera etapa se realizó énfasis en encontrar todas las variables que intervienen dentro de una contabilidad diaria de cada persona (ver Sección Revisión Literaria –Capítulo I). Se cumplió con el objetivo planteado, buscando e interpretando la información relevante, además esto permitió adquirir los conocimientos necesarios, también se hizo énfasis en la búsqueda de información relevante en la WEB documentando todo en la sección (Bibliografía), el desarrollo de esta primera etapa se encuentra (ver Sección Resultados- Análisis de requerimientos funcionales y no funcionales).

- **Objetivo específico 2:** Desarrollar una aplicación para dispositivos móviles Android que ayude a solucionar el problema de control de gastos.

El cumplimiento de este objetivo se realizó en base al desarrollo de la aplicación en la plataforma android dentro del entorno de desarrollo Eclipse, esta parte se manejó de igual forma en base a consultas (ver Sección Revisión Literaria –Capítulo III). Se cumplió el objetivo en base a entrega de resultados en formato de versiones con la ayuda de la metodología de la Programación Extrema (ver Sección Resultados - Análisis).

- **Objetivo específico 3:** Diseñar e implementar un agente inteligente deliberativo que nos permita hacer estimaciones en base a tiempo y dinero en la duración de nuestro presupuesto

Una vez desarrollado el objetivo 2, se procedió a unir el agente al

software primeramente haciendo énfasis a diseño con la metodología GAIA (ver Sección Revisión Literaria-Capitulo II Metodología GAIA) se cumplió con este objetivo determinando cada una de las etapas con las que cuenta esta metodología (ver Sección Resultado Análisis y diseño para la implementación de la Metodología GAIA de agentes Inteligentes).

2. VALORACIÓN TÉCNICA – ECONÓMICA- SOCIAL

Luego de la construcción e implementación de la aplicación es indispensable realizar la valoración técnica, económica y ambiental del mismo, permitiendo así determinar que se desarrolló de manera satisfactoria y factible con todos los recursos necesarios por tal razón se ha evidenciado que la aplicación cumple con los objetivos planteados al inicio del trabajo del titulación.

2.1. Valoración técnica económica.

Los costes principales del presente proyecto son establecidos por los gastos de trabajo de desarrollo del sistema. La cuenta de gastos del trabajo para el tiempo invertido en el proyecto es por e ingeniero del conocimiento (autor del trabajo de titulación). La opción de recursos técnicos y tecnológicos para desarrollo de la aplicación se basa en la naturaleza del problema y las facilidades del computador y/o recursos de desarrollador.

Dentro de las herramientas empleadas desde el punto de vista tecnológico, es fundamental el uso Eclipse para el desarrollo el cual permite potenciar el desarrollo del mismo y para el desarrollo del agente se ha hecho uso de la biblioteca JADE-LEAP que facilita la implementación del agente en la plataforma android.

De igual importancia, se requiere una base de datos SQLite que sirve para guardar los datos de ingresos/egresos esta base de datos es nativa de Android.

Desde el ámbito económico para el desarrollo e implementación se contó con los suficientes recursos humanos económicos que permitieron el éxito del trabajo de titulación cumpliendo con sus objetivos.

RECURSOS HUMANOS

TABLA XVI. RECURSOS HUMANOS

Descripción	Cantidad	Nº de Horas	Valor unitario	Valor Total
<ul style="list-style-type: none"> Investigador del proyecto de titulación ✓ John Solano Cabrera 	1	440	\$ 3,30	\$ 1452,00
<ul style="list-style-type: none"> Director del trabajo 	1	--	--	--
			SUBTOTAL	\$ 1452,00

RECURSOS TÉCNICOS Y TECNOLÓGICOS

TABLA XVII. RECURSOS TÉCNICOS Y TECNOLÓGICOS

Descripción	Cantidad	Valor unitario	Valor Total
HARDWARE			
<ul style="list-style-type: none"> Computadora MAC BOOK PRO 	1	\$ 1200,00	\$ 1200,00
<ul style="list-style-type: none"> Impresora Canon MP230 	1	\$ 60,00	\$ 60,00
<ul style="list-style-type: none"> Memoria Flash 	1	\$ 15,00	\$ 15,00
<ul style="list-style-type: none"> Celular Galaxi S3 Mini 	1	\$ 260,00	\$ 260,00
SOFTWARE			

• Plataforma Java	1	--	--
• JADE-LEAP	1	--	--
• Eclipse LUNA	1	--	--
• SQLite	1	--	--
• Enterprise 8	1	--	--
COMUNICACIÓN			
• Internet	1000	\$ 0,50	\$ 500,00
		SUBTOTAL	\$ 2035,00

RECURSOS MATERIALES

TABLA XVIII. RECURSOS MATERIALES

Descripción	Cantidad	Valor unitario	Valor Total
• Resma de Hojas A4	5	\$ 5,00	\$ 25,00
• Cartuchos	2	\$ 30,00	\$ 60,00
• Empastados	4	\$ 15,00	\$ 60,00
• Carpetas	4	\$ 0,80	\$ 3,20
• Perforadora	1	\$ 3,00	\$ 3,00
• Grapadora	1	\$ 3,00	\$ 3,00
• CDs (Diseño Impresión)	4	\$ 5,00	\$ 20,00
• Fotocopias	1000	\$ 0,02	\$ 20,00
		SUBTOTAL	\$ 194,20

TABLA XIX. PRESUPUESTO TOTAL

Recursos	Subtotales
Recursos Humanos	\$ 1452,00
Recursos Técnicos y tecnológicos	\$ 2035,00
Recursos Materiales	\$ 194,20
Subtotal	\$ 3681,00
Imprevistos 10%	\$ 368,12
Total	4049,12

2.2. Valoración Social

Ya que este proyecto está orientado a dar solución a los problemas financieros personales en lo que es el ahorro, se puede decir que se cumple con esta valoración ya que orienta al usuario para que tenga una mentalidad de ahorro y pueda ser partícipe del buen vivir.

H. CONCLUSIONES

- La aplicación de la metodología de programación extrema (XP) permitió documentar las partes relevantes que ayudan a determinar las entradas tanto de la aplicación como definir las creencias con las que trabaja el agente contable.
- El uso del entorno de desarrollo eclipse permitió el manejo objetivo y eficaz en la creación de la aplicación para los dispositivos móviles android.
- La librería JADE-LEAP de java en android para la implementación del agente inteligente permitió agilizar el desarrollo del mismo.
- La metodología GAIA para agentes inteligentes permitió tener un enfoque claro de las funciones que realiza el agente dentro del sistema además definir todas las propiedades necesarias para su funcionamiento.

I. RECOMENDACIONES

- Al emprender en un proyecto hay que revisar la documentación ya que hay algunos temas que no se encuentran aún disponibles y esto demora la ejecución del mismo.
- Para la ejecución de trabajos futuros sería positivo implementar la comunicación entre agentes móviles para de esta manera tener monitoreado el sistema desde cualquier lugar.
- Trabajar con agentes inteligentes permite a los desarrolladores tener control sobre el software y este pueda ser escalable o modificable de una forma óptima.

J. BIBLIOGRAFÍA

- [1] *Conspiración oculta: gran debacle de la crisis económica inminente en 2014* (documental). 2014 [Online]. Available: https://www.youtube.com/watch?v=sR5WxdTEPFk&feature=youtube_gdata_player. [Accessed: 23-Oct-2014]
- [2] “Metodología agil para el diseño de aplicaciones multimedias moviles,” 12:40:17 UTC [Online]. Available: <http://es.slideshare.net/cristopherf/metodologia-agil-para-el-diseo-de-aplicaciones-multimedias-moviles>. [Accessed: 23-Oct-2014]
- [3] K. S. Bose and R. H. Sarma, “Introduccion a la economia,” *Biochem. Biophys. Res. Commun.*, vol. 66, no. 4, pp. 1173–1179, Oct. 2012.
- [4] “2. Concepto de gastos e ingresos | Libro de contabilidad,” *contabilidad.tk*. [Online]. Available: <http://www.contabilidad.tk/concepto-de-gastos-e-ingresos-13.htm>. [Accessed: 20-May-2014]
- [5] “Conozca los diferentes tipos de gastos,” *Pymex*. [Online]. Available: <http://pymex.pe/finanzas/finanzas-y-contabilidad/conozca-los-diferentes-tipos-de-gastos>. [Accessed: 20-May-2014]
- [6] K. Moroi and T. Sato, “Ingresos Economicos,” *Biochem. Pharmacol.*, vol. 24, no. 16, pp. 1517–1521, Aug. 2011.
- [7] “Cómo cuidar el ahorro personal,” *Economía Simple*. [Online]. Available: <http://www.economiasimple.net/como-cuidar-el-ahorro-personal.html>. [Accessed: 20-May-2014]
- [8] Wikiversidad, “Curso de agentes Inteligentes,” *wiki*. [Online]. Available: http://es.wikiversity.org/wiki/Curso_de_Agentes_Inteligentes. [Accessed: 20-May-2014]
- [9] Henry Paz, “Tesis Multiagentes.”
- [10] “Definicion de Agentes inteligentes y Clasificación.” [Online]. Available: http://personales.upv.es/ccarrasc/doc/2003-2004/AI_Web/definicion.html. [Accessed: 20-May-2014]

- [11] Angel Fire, "Ejemplo de Agente segun su clasificación." [Online]. Available: <http://www.angelfire.com/falcon/miqueleiz/09tiposdeagentes.htm>. [Accessed: 20-May-2014]
- [12] leonardobernalzamora, "Agentes," 22:38:31 UTC [Online]. Available: <http://www.slideshare.net/leonardobernalzamora/agentes#>. [Accessed: 29-May-2014]
- [13] Javier Ballesteros, "Agentes Inteligente," *weebly*. [Online]. Available: http://javierballesteros-uptc.weebly.com/uploads/1/9/6/8/1968265/agentes_inteligentes.pdf. [Accessed: 20-May-2014]
- [14] Fernández Ramírez, "5. Agentes Tipos." [Online]. Available: <http://ict.udlap.mx/people/lulu/documento/capitulo5.html>. [Accessed: 20-May-2014]
- [15] Youngblood G, "Web Hunting: Design of a Simple Intelligent Web Search Agent." [Online]. Available: <http://www.acm.org/crossroads/xrds5-4/webhunting.html>. [Accessed: 20-May-2014]
- [16] Tomás de Teresa Trancón, "Un sistema de control inteligente de entrada/salida de humanos a un recinto siguiendo la metodología VigilAgent," UNIVERSIDAD DE CASTILLA-LA MANCHA, España, 2010 [Online]. Available: <http://deteresa.com/wp-content/uploads/2014/05/pfc.pdf>. [Accessed: 20-May-2014]
- [17] "Metodologia GAIA," *galeon.com*. [Online]. Available: <http://metodologiadeagentes.galeon.com/productos1098420.html>. [Accessed: 20-May-2014]
- [18] "Curso Android: Introducción a la programación," *Android: Introducción a la programación*. [Online]. Available: <http://cursoprogramacionandroid.upvx.es/>. [Accessed: 29-May-2014]
- [19] "Introducción a Android," *DesarrolloWeb.com*. [Online]. Available: <http://www.desarrolloweb.com/articulos/introduccion-android.html>. [Accessed: 29-May-2014]

- [20] Yohn daniel Amaya Balaguera, "Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles." 23-Jul-2013 [Online]. Available: http://www.uelbosque.edu.co/sites/default/files/publicaciones/revistas/revista_tecnologia/volumen12_numero2/12Articulo_Rev-Tec-Num-2.pdf. [Accessed: 29-May-2014]
- [21] Proyectos Ágiles, "Proyecto Ágiles SCRUM." [Online]. Available: <http://www.proyectosagiles.org/que-es-scrum>. [Accessed: 29-May-2014]
- [22] J. I. H. Roldán, "TDD como metodología de diseño de software," *Paradigma*. [Online]. Available: <http://www.paradigmatecnologico.com/blog/tdd-como-metodologia-de-diseno-de-software/>. [Accessed: 29-May-2014]
- [23] "Metodología de desarrollo ágil para sistemas móviles," *Introducción al desarrollo con Android y el iPhone*. [Online]. Available: http://www.adamwesterski.com/wp-content/files/docsCursos/Agile_doc_TemasAnv.pdf
- [24] "XP - Extreme Programming Ingeniería de Software." [Online]. Available: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programming.html. [Accessed: 29-May-2014]
- [25] "Pruebas de caja blanca y negra," 01:36:02 UTC [Online]. Available: <http://es.slideshare.net/rinconsete/pruebas-de-caja-blanca-y-negra>. [Accessed: 29-Sep-2014]
- [26] "Definición GAIA - ¿qué es GAIA? - Jojooa - tecnología, marketing y crm." [Online]. Available: <https://sites.google.com/site/jojooa/inteligencia-artificial/definicion-gaia-que-es-gaia>. [Accessed: 29-Sep-2014]
- [27] "Jade Site | Java Agent Development Framework." [Online]. Available: <http://jade.tilab.com/>. [Accessed: 22-Nov-2014]
- [28] "Add-Ons | Jade Site." [Online]. Available: <http://jade.tilab.com/download/add-ons/>. [Accessed: 22-Nov-2014]
- [29] "La variable de entorno PATH en Mac, desmitificada," *Bonillaware*. [Online]. Available: <http://www.bonillaware.com/configurar-variable-entorno-path-mac>. [Accessed: 22-Nov-2014]

K. ANEXOS

Anexo 1: Certificación de Traducción del Resumen.



Sr. Carlos Velastegui
PROFESOR DEL INSTITUTO
"FINE-TUNED ENGLISH"

CERTIFICA:

Que el documento aquí compuesto es fiel traducción del idioma español al idioma inglés del resumen para el trabajo de titulación denominado: *"DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS MÓVILES ANDROID, QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS ECONÓMICOS DE UNA PERSONA"*, del alumno JOHN PATRICIO SOLANO CABRERA, egresado de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.

Lo certifica en honor a la verdad y autoriza al interesado hacer uso del presente en lo que a sus intereses convenga.

Loja, 13 de mayo de 2015


Sr. Carlos Velastegui
PROFESOR DE F.T.E.



Anexo 2: Licencia Creative Commons del Proyecto.



DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS
MÓVILES ANDROID QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS
ECONÓMICOS DE UNA PERSONA por John Patricio Solano Cabrera se distribuye
bajo una [Licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0
Internacional](https://creativecommons.org/licenses/by-nc-sa/4.0/).

Anexo 3: Artículo Científico.

DESARROLLO DE UN AGENTE INTELIGENTE PARA DISPOSITIVOS MÓVILES ANDROID QUE PERMITA LLEVAR LOS INGRESOS Y EGRESOS ECONÓMICOS DE UNA PERSONA

*John Solano Cabrera*¹

¹ Universidad Nacional de Loja, Ecuador, jpsolanoc@unl.edu.ec

RESUMEN

En nuestros tiempo la economía personal es un dilema, ya que llevar los ingresos y egresos de una forma clara y concisa toma tiempo esto produce que las personas no lleven tales registros y se produzca un defisis en la economía ya que gastamos más dinero del que obtenemos. Es por eso que las nuevas tecnologías están orientadas a solucionar dichos dilemas, Los dispositivos móviles en estos últimos tiempos han tenido una gran acogida en el mercado y se orientan a dar soluciones de bolsillo es decir que se puedan llevar a todas partes y permitan solucionar problemas al instante, ANDROID es una buena alternativa, es por eso que el desarrollo de este trabajo se utilizó el mismo y la metodología XP programación extrema para su implementación; además se ha hecho uso de Inteligencia Artificial mediante Agentes Inteligentes aquí intervino la librería JADE-LEAP desarrollada en java y su implementación se realizó mediante la metodología GAIA para agentes inteligentes; la aplicación una vez terminada ayudara no solo a llevar los ingresos y egresos de una forma ordenada si no que contara con un pronosticador que guiara al usuario para sobrellevar su economía y motivar al usuario para que obtenga un porcentaje de ahorro de sus ingresos.

Palabras Clave: Android, Inteligencia Artificial, Agentes Inteligentes, JADE-LEAP, GAIA.

ABSTRACT

In our personal time economy is a dilemma as to bring revenues and expenditures in a clear and concise manner that takes time produces people with no such records and

defisis occurring in the economy and we spend more money than we get . That's why new technologies are aimed at solving these dilemmas, Mobile devices in recent times have had an immeasurable received in the market and aim to provide solutions pocket ie that can be carried everywhere and enable troubleshooting instantly, ANDROID is a good alternative, that is why the development of this work it was used and XP extreme programming methodology for implementation; further to made use of Artificial Intelligence through Intelligent Agents here intervened library developed in java and its implementation was performed by the GAIA methodology for intelligent agents JADE-LEAP; the application once completed will help not only bring income and expenses in an orderly way to tell if a predictor to guide the user to cope with its economy and motivate the user to get a save percentage of their income.Traducción al inglés del resumen presentado.

KeyWords: Android, Artificial Intelligence, Intelligent Agents, JADE-LEAP, GAIA.

1. INTRODUCCIÓN

EL presente trabajo trata de solucionar una de los problemas que aqueja a muchas personas de nivel medio y bajo la cual es llevar una economía acorde a los ingresos que tenemos disponibles,[1] las personas no está acostumbrada a llevar los ingresos y egresos de forma ordenada por falta de tiempo o por olvido de anotar a los gastos, esto produce endeudamiento incremental ya que mes a mes se produce un nuevo gasto extra y pago de deudas del mes anterior y más ahora se ve reflejado en el dinero plástico que son las tarjetas de crédito que permiten hacer comprar y diferirlas pero sin un control adecuado que permita determinar la cantidad la cual se puede gastar. Sin embargo este problema se puede resolver ya que las nuevas tecnologías están orientadas a dar soluciones de forma ágil a esta clase de problemáticas y más aún ahora con la aparición de los dispositivos de bolsillo que permiten llevar a todas partes nuestros datos y estar disponibles a toda hora.

2. METODOLOGÍA

2.1 Programación extrema

La metodología de programación extrema o metodología ágil se utilizó para la implementación y desarrollo de la aplicación ya que esta no requiere mucha documentación y se puede realizar enfocándose más a la programación, el trabajo

estrechamente con el usuario y entrega de versiones. **[1]**

Los pasos para esta metodología son.

- Fase de exploración

En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología **[2]**

- Fase de planeamiento

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días. Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos. Por otra parte, el equipo de desarrollo mantiene un registro de la “velocidad” de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración. La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar. Al planificar según alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación. **[2]**

- Fase de producción

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase. Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento). **[2]**

- Fase de mantenimiento

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura. **[2]**

- Fase de muerte de la aplicación

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto

requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. [2]

Entre los diferentes puntos dentro de esta metodología se abarca todo el desarrollo de esta aplicación desde sus requerimientos iniciales hasta prototipos utilizados para su implementación en la plataforma ANDROID.

2.2 GAIA

La metodología GAIA como lo propone Wooldridge, Jennings y Kinny (2000) permite ir sistemáticamente desde los requerimientos hasta un diseño lo suficientemente detallado que puede ser implementado directamente. El análisis y el diseño se pueden ver como el proceso de desarrollo incremental que conlleva a modelos más detallados del sistema que se va a construir. Los principales conceptos de gaia se pueden dividir en dos categorías: abstractas y concretas. Las entidades abstractas son las que durante el análisis, conceptualizan el sistema pero no necesariamente tienen una realización directa en el sistema. En contraste las entidades concretas son usadas en el proceso de diseño y típicamente están más relacionadas con el sistema en tiempo de ejecución. [3]

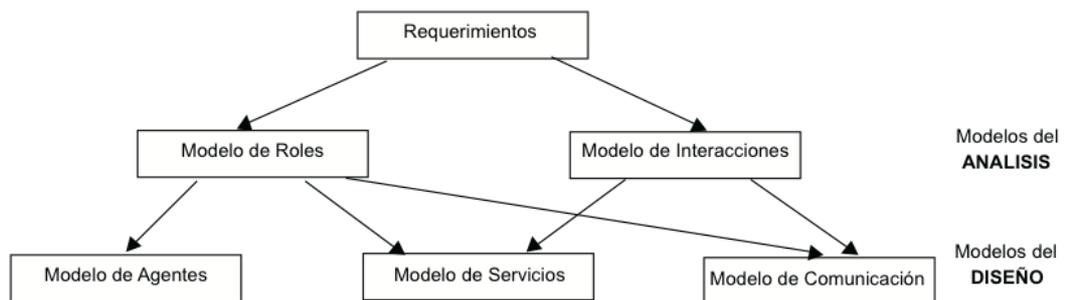


FIG. 1 MODELOS DE LA METODOLOGÍA GAIA

Estos son las diferentes interacciones que se deben realizar para la realización del agente deliberativo que funcionara conjuntamente con la aplicación móvil. [4]

3. MATERIALES Y MÉTODOS

La implementación del agente inteligente para llevar la economía personal de una persona se realizó a través del entorno de programación android, además haciendo uso de la librería Jade LEAP para el agente y SQLite para la persistencia de datos la cual se encuentra integrada en los dispositivos, integrando todo en Eclipse para su desarrollo.

A más de ello sitios web dedicados al tema de agentes inteligentes en dispositivos móviles, las fuentes de información a consultar se basaron en artículos científicos, tesis doctorales, sitios especializados y oficiales de la temática a buscar además sin olvidar las entrevistas realizadas en la parte contable a personas expertas en la temática.

Con respecto a los métodos, para cumplir con los objetivos planteados se hizo énfasis a la investigación objetiva, ordenada pero sobre todo alcanzable de acuerdo a los parámetros propuestos en la realización del trabajo presente trabajo.

4. DISEÑO E IMPLEMENTACIÓN

4.1 REQUERIMIENTOS FUNCIONALES

Se determina las entradas y la funcionalidad que tendrá la aplicación, en esta fase se describen las entradas, comportamientos y salidas del sistema.

Requerimiento	Descripción del requerimiento.
RF1	El sistema permitirá al usuario agregar ingresos y egresos
RF2	El sistema permitirá al usuario editar ingresos y egresos
RF 3	El sistema permitirá al usuario eliminar ingresos y egresos
RF 4	El sistema permitirá al usuario visualizar información de cada ingresos y egresos
RF 5	El sistema mostrara al usuario dentro de egresos, descripción, monto, fecha, la periodicidad y categoría.
RF 6	El sistema mostrara dentro de ingresos, descripción, monto, fecha, ahorro.
RF 7	El sistema definirá automáticamente la fecha de ingreso de un valor.
RF 8	El sistema verificara que los datos ingresados sean correctos
RF 9	El sistema mediante el agente deliberativo permitirá evaluar los datos y emitir alertas de gastos en su evaluador.

RF 13	El sistema mediante el agente deliberativo mostrara las descripción de la categoría de los egresos y como debe ser su gasto.
RF 14	El sistema realizara estimación del fin del presupuesto y del gasto diario.

4.2 MODELO DE ROL DEL AGENTE

Un rol necesita un conjunto de Permisos que identifique aquellos recursos que puede acceder para poder cumplir con sus responsabilidades. Estos recursos son generalmente fuentes de información en las que puede leer como (Base de Datos), hacer modificaciones y enriquecer bajo ciertas circunstancias.

Esquema de rol	Agente Contable
<p>Descripción.</p> <p>Este rol involucra que el agente contable mediante los datos de entrada como ingresos y egresos pueda evaluar estos y determinar si es preciso emitir una alerta al usuario para que tenga en cuenta los gastos excesivos que existen en el proceso de egresos de esta misma forma emitirá una alerta del valor diario que puede tener una persona contara con evaluador de porcentajes de egresos excesivos y este evaluara el fin del presupuesto.</p>	

Protocolos y actividades

Actividades.

- **Vigilar.-** Consiste en observar los ingresos y egresos definidos como entradas del usuario y que para el agente servirán como los creencias.
- **Evaluar.-** Se realizara un diagnóstico de los ingresos y egresos dentro de los parámetros (Ahorro, Monto de Ingresos y Monto de Egresos) para emitir un proceso teniendo especial cuidado en procurar el ahorro deseado por el usuario, en esta parte se evidencia los deseos que cumplirá el agente.

Protocolo.

- **Procede.-** En base a la evaluación de los diferentes parámetros se realizara.
 - ✓ Calculo de valor de gasto diario.
 - ✓ Calcular fin del presupuesto.
 - ✓ Calcular gastos excesivos.
 - ✓ Calcular porcentaje de gastos.
 - ✓ Emisión de alertas para el usuario.

Permisos

Leer ingresos.

Leer egresos.

Responsabilidades.

Agente Inteligente (Vigilar, Evaluar, Proceder)

4.3 DEFINICIÓN DE PROTOCOLOS

Son las interacciones entre los roles, básicamente es una actividad que necesita una interacción entre el rol.

Protocolo: Procede
Nombre: Agente Contable
Calcular valor de gasto diario Se evalúa el gasto en base a sus ingresos y egresos, estos una vez evaluados permitirán mostrar el valor de gasto diario para el usuario teniendo en cuenta el porcentaje de ahorro que se desea obtener el usuario.
Calcular fin del presupuesto. En base a la entrada del usuario de los ingreso estos se evaluaran para definir la terminación del mismo teniendo en cuenta valores del supervivencia básicos dentro de un rango de presupuestos.
Calcular gastos excesivos. En base a los pesos que tendrá cada categoría se evaluara que gastos son excesivos e innecesarios para el usuario pueda tomar en cuenta y modificar su gasto.
Calcular porcentaje de gastos. En base a los gastos ingresados por el usuario el agente podrá calcular el porcentaje de cada gasto y su influencia en el presupuesto total.
Emisión de alertas En base a los cálculos realizados por el agente este mismo emitirá alertas y avisos al usuario para que pueda modificar y obtener el ahorro deseado.

4.4 Modelo de servicios.

Se identifica las entradas, salidas su precondición y post-condición dentro de su uso en el sistema.

Servicio	Entrada	Salidas	Precondición	Post-Condición
Vigilar	Ingresos Monto Ahorro Fecha Egreso Monto. Categorías.	Notifica a procede	Ingresos \neq null Monto > 0 Ahorro > 0 Fecha \neq null Egresos \neq null Monto > 0 Categorías \neq null	Datos Insuficientes para su evaluación
Evaluar	Ingresos Monto Ahorro Fecha Egreso Monto. Categorías.	Notificar a procede	Ingresos \neq null Monto > 0 Ahorro > 0 Fecha \neq null Egresos \neq null Monto > 0 Categorías \neq null Vigilar = true	No se puede evaluar sin datos, se debe vigilar antes de poder evaluar.
Procede	Notificaciones entrantes de Vigilar y evaluar	Gasto Diario Ahorro total Fin Presupuesto Gastos Excesivos	Vigilar = true Evaluar = true	Se debe vigilar y evaluar para proceder

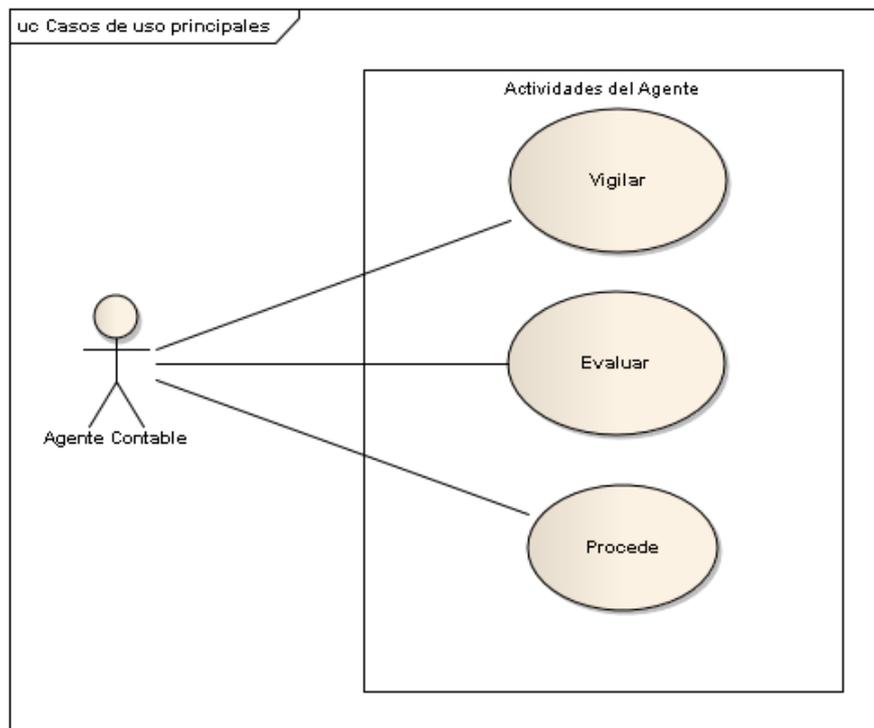
		Porcentaje de Gastos Alerta Usuario		
--	--	---	--	--

4.5 Diseño de creencias deseos e intenciones del agente deliberativo.

La metodología GAIA comprende varios puntos, pero para comenzar se requiere de requerimientos iniciales los cuales no abordaremos en esta sección ya que la metodología de programación extrema nos ayudó a resolver esta parte, estos son los requerimientos funcionales mencionados en la parte anterior, que en agentes se consideraran como creencias.

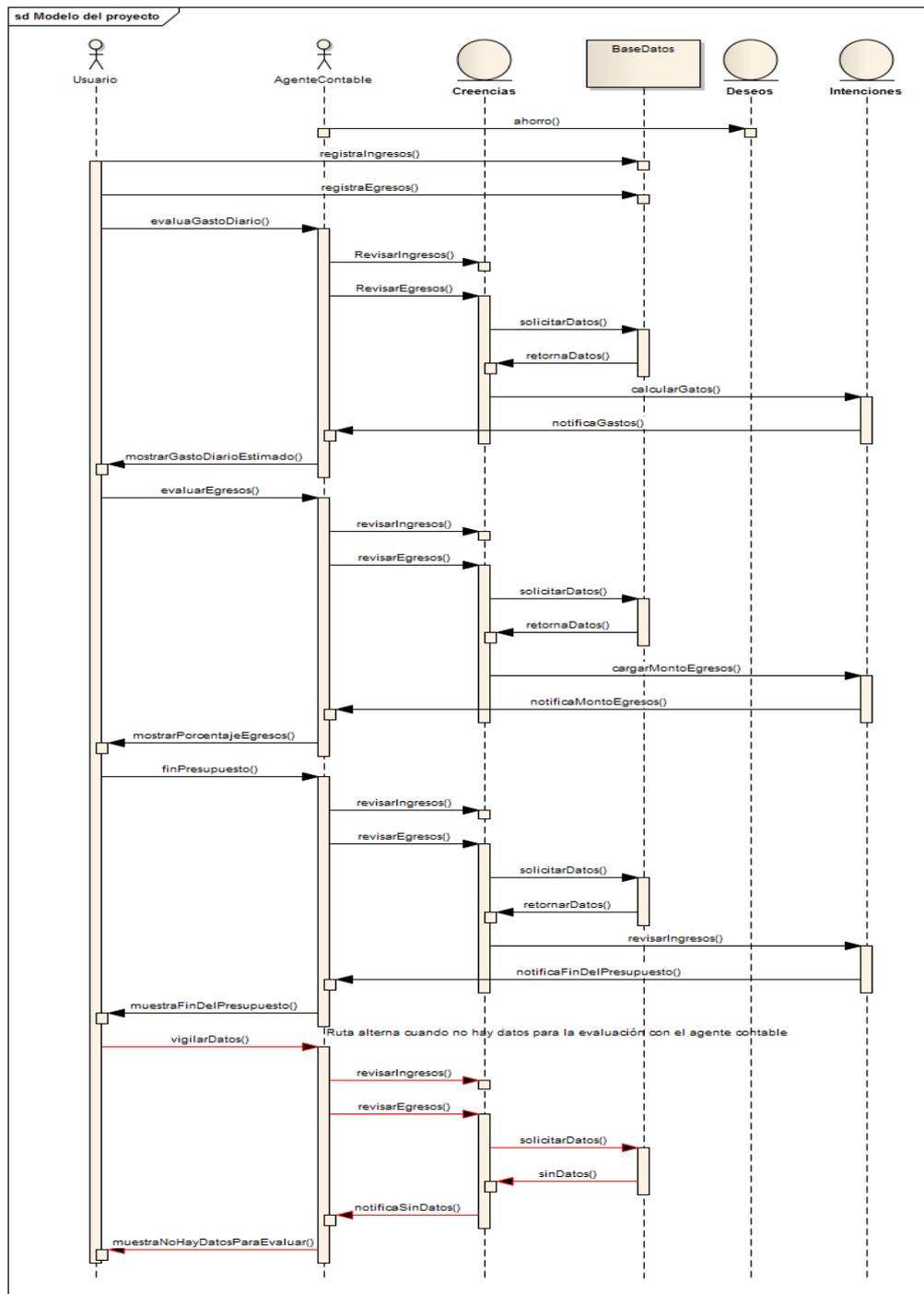
4.5.1 Modelo del Agente.

Se determina las acciones o actividades que realizara el agente dentro del sistema.



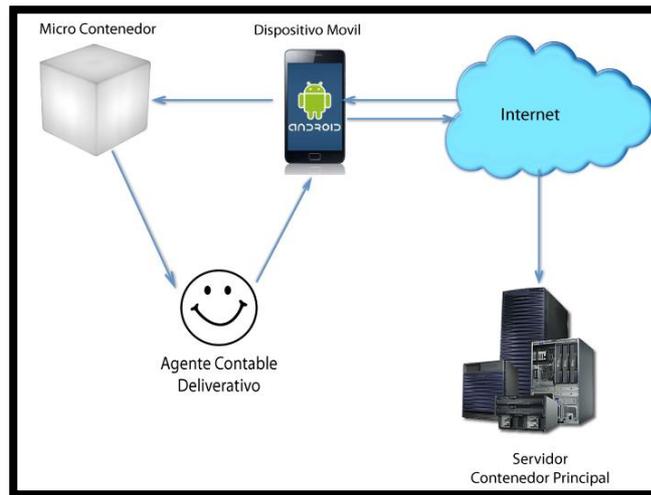
4.5.2 Diseño de Interacción.

El modelo de interacción para el agente contable define la funcionalidad del agente y su conexión entre Creencias, Deseos e intenciones y la base de datos además su interacción con el usuario.



4.5.3 Arquitectura del agente.

La arquitectura del agente refiere a la comunicación entre los diferentes dispositivos ya que para la utilización del sistema requiere un contenedor principal el cual se encontrara en la nube.



5. RESULTADOS

5.1 PRUEBAS EN APKUDO

Para esta fase de pruebas se ha hecho uso de la plataforma web apkudo que proporciona una solución robusta que permite conocer de forma acertada en que plataformas de android correrá de manera adecuada la aplicación.

La imagen muestra la interfaz de usuario de la plataforma web APKUDO. El título principal es 'analysis run results'. A la izquierda hay una tabla con los siguientes datos:

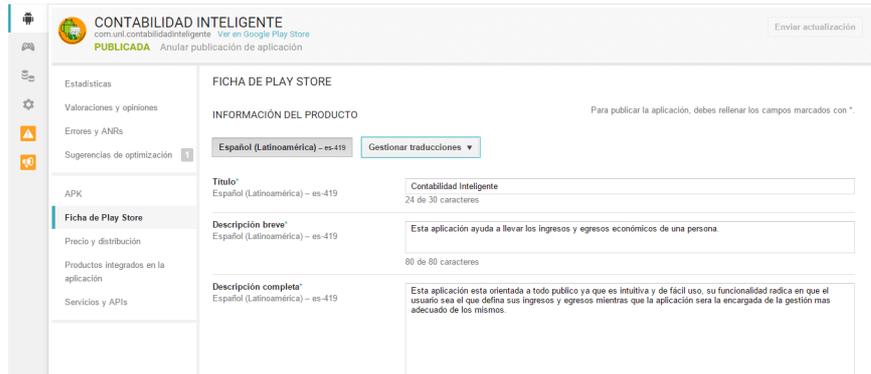
Manufacturer	Android	State	Name	Model	Res
Blu	2.3.5	Completed	Studio 5.3	DS10	Unava
Coolpad	2.3.7	Completed	Quattro 4G	S860E	Unava
HTC	2.3.4	Completed	G2	PC10100	Failure:
	2.3.6	Completed	Nexus One	P999100	Unava
	4.0.3	Completed	One X LTE	P983100	Failure:
		Completed	One V	T-320e	Succ
		Completed	Evo 4G LTE	AP9325CKT	Failure:
		Completed	One X	S720E	Failure:
	4.0.4	Completed	One X	S720E	Unava
		Completed	One S	HTC One S	Failure:
	4.1.1	Completed	HTC One X+	HTC One X+	Failure:

A la derecha de la tabla, hay un panel de 'Apps & Analyses' para 'Contabilidad Inteligente'. Muestra un resumen de 'ANALYSIS 3 COMPLETE' con 24 dispositivos exitosos, 27 fallidos y 5 en estado de fallo. También incluye filtros para 'Manufacturer' y 'Android Version'.

Como se puede observar en la imagen anterior la aplicación tuvo éxito ejecutándose en 24 dispositivos de 74 probados esto se debe a que el soporte tanto de agentes como de la base de datos y las nuevas tecnologías nos obligan a siempre orientarnos a lo moderno es por eso que para la ejecución correcta de

esta aplicación se solicita que el usuario tenga por lo menos un celular con la versión de android mínima de 3.0.

5.2 Play Store.



Dentro de Play store contamos con un espacio donde podemos observar algunas estadísticas como por ejemplo número de descargas y número de dispositivos en los cuales está corriendo y está siendo utilizado.

Número de instalaciones	Superior a 50
Numero de dispositivos que está corriendo	Superior a 11

En la gráfica posterior se muestra, en que versiones de android ha sido instalada la aplicación contabilidad inteligente.



6. TRABAJOS RELACIONADOS

Se puede mencionar varios trabajos relacionados con agentes inteligentes como por ejemplos chat, proyecto de salud, en la parte contable son innumerables los sistemas que se han realizado para ingresos contables ejemplo Hojas de cálculo con fórmulas o Mónica sistema contable. Pero la unión de estos dos según esta

investigación es única es por tal motivo que se puede determinar que este trabajo es pertinente y de relevancia. [5]

7. CONCLUSIONES Y TRABAJO FUTURO.

La aplicación de la metodología de programación extrema (XP) permitió documentar las partes relevantes que ayudan a determinar las entradas tanto de la aplicación como definir las creencias con las que trabaja el agente contable.

El uso del entorno de desarrollo eclipse permitió el manejo objetivo y eficaz en la creación de la aplicación para los dispositivos móviles android.

La librería JADE-LEAP de java en android para la implementación del agente inteligente permitió agilizar el desarrollo del mismo.

La metodología GAIA para agentes inteligentes permitió tener un enfoque claro de las funciones que realiza el agente dentro del sistema además definir todas las propiedades necesarias para su funcionamiento.

El enfoque en trabajos futuros se lo puede definir como un mundo incremental, ya que esta aplicación hace uso de agentes inteligentes, esta puede comunicarse con otros agentes definidos en cualquier espacio de la red y entre estos podrían llegar a deliberar forma más acertada y concreta de como un usuario puede llegar a tener márgenes de ahorro significativos. Mostrando una visión se podría pensar en la autonomía del sistema es decir que los datos definidos como entradas que ahora realiza el usuario podrían ser realizadas por otros agentes así mismo plantear objetivos los cuales pueden ser cubiertos por los agentes del sistema. [6]

8. AGRADECIMIENTOS

Mi más profundo y sincero agradecimiento a dios por guiar y bendecir todo esfuerzo y dedicación en mi vida; A la Universidad Nacional de Loja, al Área de la energía las industrias y los recursos naturales no renovables, a la carrera de Ingeniería en Sistemas, por darme la oportunidad de formarme como profesional; Al coordinador y distinguidos catedráticos por guiarme y compartir sus sabias enseñanzas en el transcurso de mi formación académica.

9. REFERENCIAS BIBLIOGRÁFICAS

- P. E. (XP), « EcuRed,» 14 05 2011. [En línea]. Available:
1] http://www.ecured.cu/index.php/Programaci%C3%B3n_Extrema_%28XP%29. [Último acceso: 19 11 2014].
- M. e. C. A. S. Alejandro., «Revista digital universitaria,» 12 30 2010. [En línea].
2] Available: <http://www.revista.unam.mx/vol.3/num4/art39/>. [Último acceso: 3 12 2014].
- «Metodología GAIA,» GALEON, 11 12 2012. [En línea]. Available:
3] <http://metodologiadeagentes.galeon.com/productos1098420.html>. [Último acceso: 7 11 2014].
- J. P. Mestras, «Agentes Inteligentes,» 01 01 2010. [En línea]. Available:
4] <http://www.fdi.ucm.es/profesor/jpavon/doctorado/arquitecturas.pdf>. [Último acceso: 9 10 2014].
- J. R. Rodríguez, «Diseño de un Agente Inteligente Web basado en técnicas de
5] Inteligencia Artificial,» Universidad de Sevilla, 28 06 2006. [En línea]. Available:
<http://www.dte.us.es/personal/jropero/Trabajo%20Investigacion2.pdf>. [Último acceso:
10 10 2014].
- «Las Herramientas del Futuro,» 24 06 2010. [En línea]. Available:
6] http://tecnologias-lasherramientasdelfuturo.blogspot.com/2010/06/agentes-inteligentes_24.html. [Último acceso: 11 10 2014].