



REVECOM

Revista Venezolana de Computación

**Sociedad Venezolana
de Computación**

ISSN: 2244-7040

**Vol. 2, No. 1
Diciembre 2015**



REVECOM

Revista Venezolana de Computación

**Sociedad Venezolana
de Computación**

**Editores:
Eric Gamess, Wilmer Pereira, Yudith Cardinale**

ISSN: 2244-7040

**Vol. 2, No. 1
Diciembre 2015**

Editorial

La Arquitectura para la Organización de Sistemas

La arquitectura es una disciplina muy antigua que hace converger el arte con las técnicas de ingeniería y trata de organizar elementos dispersos de un ambiente o contexto, en un todo u objeto consistente y funcionalmente utilizable, con un propósito definido de satisfacer necesidades humanas; ha evolucionado junto con las diferentes épocas de la civilización: se pasa de la caverna, ambiente natural con un acondicionamiento mínimo, a la choza, casa de piedra, palacio, etc. Las civilizaciones pasan de estar organizadas en tribus aisladas al aglutinamiento en pueblos, ciudades, etc. Estos aglomerados han tenido sus estilos arquitectónicos particulares que han caracterizado las diferentes épocas, clásico, románico, gótico, renacentista, barroco, etc.

Un sistema computacional, usualmente llamado “sistema de software” o simplemente “software”, no escapa de tener su arquitectura particular que lo aglutina, la cual representa una abstracción de nivel más alto que el código y es una forma de organizar sus elementos o componentes que realizan funciones específicas y están relacionados entre sí; esta organización proporciona un comportamiento global al sistema. La arquitectura del sistema de software, o simplemente Arquitectura del Software (AS), no debe confundirse con la arquitectura física del computador o “hardware”, cuyo nivel de abstracción es el de microcircuitos; sobre los cuales se enlaza un sistema operativo (software) que gestiona recursos de hardware (memoria, etc.) y facilita un ambiente de desarrollo o plataforma para la ejecución de los diferentes sistemas de software en ese computador.

Con el rápido avance tecnológico de las últimas décadas, las arquitecturas de los sistemas de software también han evolucionado, al igual que las del hardware, sin embargo siempre se reconocen en ellas elementos clásicos o estilos, como son las capas o “layers”, los ductos-filtros o “pipe-filters”, las basadas en eventos o “event-driven”, para mencionar algunos; los sistemas de software se organizan en modelos cliente-servidor, distribuidos respecto a la ubicación física de computadores y conectados en red para la comunicación. Algunos hitos en la evolución de la AS son los siguientes:

- 1996 - Shaw y Garlan: definen el diseño arquitectónico del software como disciplina.
- 1996 - Gang Of Four: E. Gamma, R. Helm, R. Johnson, J. Vlissides: formulan catálogos de patrones arquitectónicos y de diseño.
- 1999 - RUP (Rational Unified Process) - G. Booch et al.: definen un método en cuya fase inicial se diseña la arquitectura “baseline” para todo el sistema.
- 2000-2002 A. van Lamswerde, I. Bosch: formulan métodos de diseño arquitectónicos basados en metas, transformaciones, atributos de calidad y reutilización de componentes.
- 2011 Pohl et al.: formulan un método para la Ingeniería de Líneas de Productos de Software (LPS), centrado en un modelo de variabilidad de una Arquitectura de Referencia (AR), a partir de la cual se derivan sistemas similares que comparten activos o “assets” reutilizables de un repositorio.

En el ciclo de vida del software, la fase de diseño de la AS es la que se denomina ahora “Diseño del Software” en el Currículo del ACM (2013) que dicta las pautas para la enseñanza de la Computación a nivel mundial. Es la fase donde se debe responder con soluciones concretas (componentes) a los requisitos funcionales (RF) y no funcionales (RNF) generados en la etapa previa de Ingeniería de Requisitos. Los RNF corresponden a exigencias de calidad, como seguridad, confiabilidad y eficiencia,

que provienen de requisitos del dominio, tales como el estilo de la arquitectura o las reglas del negocio, por ejemplo SOA (Arquitectura Orientada a Servicios)/capas para aplicaciones Web, en el dominio de los sistemas de gobierno electrónico y leyes o regulaciones, o de los propios RF, como un adecuado control de acceso al sistema; son en efecto los RNF que dirigen el proceso de diseño arquitectónico y deben ser considerados en las etapas tempranas del desarrollo, para evitar problemas de evolución del sistema. El diseño de la AS proporciona beneficios a la Ingeniería del Software, que trata de pasar de las prácticas artesanales a las prácticas industriales, tales como el manejo de la complejidad, la reutilización, la evolución, el análisis de riesgos. La AS es el puente entre los requisitos del software y el código. Una tendencia práctica en la industria del software actual es un desarrollo rápido y confiable para entregar al mercado productos de calidad a un bajo costo. Esto es posible gracias al enfoque de LPS, cuyo artefacto central es la AR para un dominio dado. Ahora bien, el diseño de una AR no es ni rápido ni trivial e involucra una inversión cuantiosa, por sus características genéricas que se basan en un estudio exhaustivo del dominio o de productos existentes y no se tienen aún métodos maduros de diseño, sino los métodos clásicos dirigidos a las arquitecturas para un solo sistema y no a una familia de sistemas.

El desarrollo de una LPS es un área de investigación abierta, en la cual se manifiestan dos tendencias: la primera y más clásica, el enfoque proactivo, descendente o “top-down” basado en un estudio exhaustivo del dominio para definir la familia de sistemas de la LPS y su AR; limitaciones: es un enfoque costoso, porque se debe estudiar la organización empresarial, su negocio y los procesos que se llevan a cabo, sensibles a ser automatizados para constituir sistemas de soporte y de ahí derivar la AR, la cual será el instrumento clave para un desarrollo rápido y confiable respecto a la calidad de los productos concretos de ahí derivados. La segunda es el enfoque extractivo, ascendente o “bottom-up”, el cual es más práctico y de bajo costo porque se basa en el estudio de productos similares existentes en un dominio dado, cuyas arquitecturas se “refactorizan” para construir la AR; las limitaciones son la falta de documentación adecuada de los productos y el análisis semántico para determinar componentes funcionalmente similares, que implica un fuerte trabajo de reingeniería. Sin embargo es un enfoque práctico, en parte automatizable y utilizado a nivel industrial. Un tercer enfoque es el reactivo, también descendente y menos costoso que el proactivo, el cual comienza con el estudio de una pequeña LPS existente y se avoca a extenderla, agregando componentes para convertirla en una arquitectura más general y que abarque más alternativas de instanciación. Estos enfoques pueden ser parcialmente combinados para de allí proponer procesos más eficaces. Lo que hay que resaltar en el diseño arquitectónico es la intervención humana, la cual juega un papel preponderante: el ingeniero del dominio o arquitecto del software, debe tener experticia en proponer soluciones arquitecturales adecuadas tecnológicamente y que respondan a los requisitos de calidad exigidos por la familia de sistemas del dominio dado; en cambio las funcionalidades son un problema resuelto porque los componentes comunes se reutilizan directamente; la AR es responsable de que los sistemas concretos de ella derivados, tengan la calidad global exigida; si la AR está bien construida, tiene esta calidad integrada, por eso su proceso de construcción es complejo y costoso. Un proceso bottom-up semiautomático de diseño de una AR, estudiando tres productos del mercado en el dominio de la salud, OpenEMR, PatientOS y Care2X, es un resultado de investigación del proyecto DISOFT del Fonacit (2013-2015), en el cual participa nuestro Laboratorio de Investigación MoST, del centro ISYS de la Escuela de Computación, Facultad de Ciencias, UCV; el proceso propuesto ha sido objeto de publicaciones nacionales e internacionales recientes y demuestra lo activo del área en cuanto a la investigación.

Dra. Francisca Losavio
Profesora de la Escuela de Computación
Universidad Central de Venezuela

Revista Venezolana de Computación

ReVeCom (Revista Venezolana de Computación) es la primera revista venezolana arbitrada, periódica, digital, orienta a la publicación de resultados de investigación en el campo de la computación. ReVeCom fue creada por la SVC (Sociedad Venezolana de Computación) y tiene entre sus objetivos hacer conocer los trabajos de alta calidad investigativa que se realizan a nivel nacional, latinoamericano e internacional. La revista permite la divulgación de artículos con aporte original en castellano o inglés.

En octubre de 2015, se celebraron conjuntamente la Tercera Conferencia Nacional de Informática, Computación y Sistemas (CoNCISa 2015) y la Tercera Escuela Venezolana de Informática (EVI 2015), en la Universidad de Carabobo, Valencia, Venezuela.

La edición de este tercer número de ReVeCom está dedicada a los mejores trabajos presentados en CoNCISa 2015. Esta edición consolida un esfuerzo grande que se ha venido haciendo en el seno de la SVC, para promover la investigación en el campo de la computación a nivel nacional, e impulsar una nueva generación académica y profesional en nuestra área de saber para el desarrollo del país.

ReVeCom es una revista abierta para una mayor difusión de los resultados de investigación. Cuenta con una página web (<http://www.svc.net.ve/revecom>), donde se encuentran los trabajos publicados e información sobre la revista. La revista promueve la pluralidad de intereses, dando cabida a la divulgación de trabajos de todos los campos del conocimiento inherentes a la computación.

Además de una selección de los mejores artículos de CoNCISa, ReVeCom también publica artículos de investigación en el campo de la computación, a través de un arbitraje por expertos del área. Por ende, se hace una invitación amplia a la comunidad informática nacional, latinoamericana e internacional, a someter sus propios trabajos para los números de ReVeCom por venir.

Directorio de la Sociedad Venezolana de Computación

Presidente:

Dr. Wilmer Pereira (Universidad Católica Andrés Bello)

Secretario:

Dr. Carlos Acosta (Universidad Central de Venezuela)

Tesorero:

Profesora Rosseline Rodríguez (Universidad Simón Bolívar)

Vicepresidente:

Dra. Yudith Cardinale (Universidad Simón Bolívar)

Coordinador de Eventos:

Dr. Leonid Tineo (Universidad Simón Bolívar)

Coordinador de Publicaciones:

Dr. Eric Gamess (Universidad Central de Venezuela)

Coordinadora de Educación e Investigación:

Dra. Judith Barrios (Universidad de Los Andes)

Edición

Comité Editorial

Director:

Dr. Eric Gamess - Universidad Central de Venezuela, Venezuela
Redes de computadores, computación de alto desempeño, simulación.

Coordinador del Comité Editorial:

Dr. Wilmer Pereira - Universidad Católica Andrés Bello, Venezuela
Inteligencia artificial, robótica autónoma, aprendizaje automatizado.

Jefe de Redacción:

Dra. Yudith Cardinale - Universidad Simón Bolívar, Venezuela
Computación paralela, computación de alto desempeño, sistemas distribuidos, computación en la nube, arquitecturas paralelas, servicios web, web semántica.

Miembros del Comité Editorial

Dr. Carlos Acosta - Universidad Central de Venezuela, Venezuela
Computación paralela, computación de alto desempeño, computación reconfigurable y FPGAs, simulación paralela y distribuida, BigData.

Dr. Andrés Arcia-Moret - Universidad de los Andes, Venezuela
Simulación de redes, protocolos de transporte, redes inalámbricas.

Dr. Ernesto Coto - The University of Sheffield, Inglaterra
Computación gráfica, visualización científica, procesamiento digital de imágenes.

Dra. Francisca Losavio - Universidad Central de Venezuela, Venezuela
Ingeniería del software, arquitecturas y calidad del software, producción industrial de software.

Dr. Francisco Luengo - Universidad del Zulia, Venezuela
Computación social, minería de texto.

Dr. Jonas Montilva - Universidad de los Andes, Venezuela
Ingeniería del software, sistemas de información.

Dra. Masun Nabhan - Universidad Simón Bolívar, Venezuela
Inteligencia artificial, minería en datos, aplicaciones de inteligencia artificial para educación y discapacitados.

Dra. Dinarle Ortega - Universidad de Carabobo, Venezuela
Ingeniería del software, arquitectura del software, arquitecturas empresariales, modelado de procesos de negocio.

Dr. David Padua - University of Illinois, USA
Compiladores, computación de alto desempeño.

Dr. Leonid Tineo - Universidad Simón Bolívar, Venezuela
Bases de datos, lógica difusa, lenguajes artificiales, minería de datos.

Tabla de Contenido

Editorial	ii
Revista Venezolana de Computación	iv
Directorio de la Sociedad Venezolana de Computación	v
Comité Editorial	vi
1. CrESTa: Ejecución Tolerante a Fallas de Servicios Web Compuestos basado en Crónicas	1-13
Marcos Grillo, Yudith Cardinale, José Aguilar	
2. Integrando el Análisis Estratégico del Negocio en el Modelado de Metas	14-25
Mayela Delgado, Alfredo Matteo, Francisca Losavio	
3. Un Marco de Referencia Metodológico para un Sistema Basado en Conocimiento Usando MDA	26-35
Pablo Valera, Dinarle Ortega, María Guevara	
4. Sistema Buscador de Contenidos Digitales de la Nube de Conocimiento del Proyecto Madre	36-47
José Aguilar, Ricardo Dos Santos	
5. Un Mecanismo de Respuestas a Consultas en Presencia de Nulos	48-59
Josué Ramírez, Leonid Tineo	
6. Un Estudio Empírico sobre el Impacto de la Mutación en la Variedad Genética y la Rata de Convergencia de un Algoritmo Genético Simple	60-69
Sandra Lima, Minoru Akiyama	
7. Shader Tool: Un Ambiente Web de Desarrollo Integrado para GLSL	70-82
Luiyit Hernández, Esmitt Ramírez	
8. DanceNet: Entorno Colaborativo para la Enseñanza de Coreografías Empleando un Kinect	83-94
Andrés Mayo, Esmitt Ramírez	
9. Aplicación para Dispositivos Móviles Android con Capacidad de Procesamiento de Consultas Realizadas sobre Documentos RDF Almacenados en la Nube	95-106
Jesús Soto, Amadís Martínez	
10. Sistema Inteligente de Reconocimiento de Estados Cerebrales	107-118
Junior Altamiranda, José Aguilar, Luis Hernández	
Índice de Autores	119

CrESTa: Ejecución Tolerante a Fallas de Servicios Web Compuestos basado en Crónicas

Marcos Grillo¹, Yudith Cardinale¹, José Aguilar²
marcos.grillor@gmail.com, yudith@ldc.usb.ve, aguilar@ula.edu.ve

¹ Departamento de Computación, Universidad Simón Bolívar, Caracas, Venezuela

² Departamento de Computación, Universidad de Los Andes, Mérida, Venezuela

Resumen: Los Servicios Web Compuestos (SWC) facilitan la definición e implementación de procesos delicados y complejos de una forma confiable y versátil. Una vez que un usuario envía un requerimiento al sistema, durante la fase de composición se realiza el descubrimiento y la composición de varios Servicios Web, que en conjunto darán respuesta a la petición del usuario. Posteriormente un motor de ejecución realizará las invocaciones apropiadas de dichos Servicios Web para finalmente retornar la respuesta al usuario. En el área de motores de ejecución de SWC se han realizado diversas propuestas que incluyen sistemas centralizados o distribuidos, que implementan mecanismos de tolerancia a fallas para garantizar el funcionamiento y consistencia del sistema. En este artículo se propone CrESTa (*ChRonical-based Execution engine for composite web Service with fault TolerAnce*), un motor de ejecución tolerante a fallas, que modela los SWC con Redes de Petri Coloreadas, usa crónicas para detectar fallas y considera parámetros de calidad de servicio. Presentamos la arquitectura de CrESTa y mostramos su funcionamiento con un caso de estudio práctico.

Palabras Clave: Composición de Servicios Web; Tolerancia a Fallas; Crónicas; Calidad de Servicio.

Abstract: Composite Web Services (CWS) facilitates complex and delicate process definition and implementation. A user might query a system for a requirement which is processed and transformed in a CWS. After the processing phase, an execution phase is needed which is executed by an engine, whose only function is to call all the Web Services involved to generate an answer that will satisfy the query. Several proposal has been done in execution engines, which includes both centralized and distributed approaches, and almost everyone implements recovery mechanisms to maintain the system stability and consistency. This article proposes CrESTa (*ChRonical-based Execution engine for composite web Service with fault TolerAnce*), a CWS execution engine with recovery mechanisms, that uses Colored Petri Networks for CWS representation, chronicles to detect failures, and considers quality assurance parameters. We present the architecture, and the execution process through a practical example.

Keywords: Composite Web Services; Recovery Mechanisms; Chronicles; Quality of Service.

I. INTRODUCCIÓN

Los Servicios Web (SW) han tenido un impacto contundente en nuestro día a día. Es muy común ver como universidades y empresas proveen servicios a través de la red para mejorar y optimizar procesos, permitiendo a las organizaciones procesar y comunicar datos sin la necesidad de conocer profundamente las tecnologías de cada organización [1]. Los SW han ganado considerable interés en investigaciones del sector académico y de la industria, dado que ofrecen un lenguaje neutral, débilmente acoplado, independiente de las plataformas y estandarizado para integrar aplicaciones y servicios dentro de organizaciones [2].

La composición de servicios web consiste en combinar SW desarrollados por diversas organizaciones que ofrecen diversas propiedades transaccionales, funcionales y calidad de ser-

vicio [3]. Diversas razones pueden influir en la necesidad de crear un Servicio Web Compuesto (SWC), tal como la creación de aplicaciones que agreguen diversos SW existentes para satisfacer requerimientos de negocio dinámicos [4]. Por ejemplo, suponga un usuario que desea comprar un boleto de avión y provee información respecto a ciertos parámetros tales como límite de precio, aerolínea favorita y no desea escalas (estos parámetros representan los requerimientos funcionales); se puede establecer un SWC que contenga diversos SW: uno para reservar el boleto, otro que contenga la conexión con su banco para el proceso de pago y otro que le envíe un correo electrónico con la confirmación de su compra. Si existen varios SW que ofrecen las mismas funcionalidades, la selección puede ser guiada por los requerimientos no funcionales del usuario (e.g., tiempo máximo de ejecución, confiabilidad de los SW); así para nuestro ejemplo, se elige el mejor entre

todos los SW de aerolíneas disponibles (el que reporte menores tiempos de ejecución y tenga mayor confiabilidad). Con este simple ejemplo se observa cómo se pueden resolver problemas complejos con una composición de SW ya disponibles [1]. El desarrollo de un SWC se puede descomponer en dos fases: composición y ejecución. Primero se debe definir la composición dinámica o estáticamente, dependiendo de los requerimientos del usuario (funcionales y no funcionales). Como resultado se obtiene un SWC conformado por varios SW, representado con una estructura como un grafo, un *workflow* o una red de Petri; esta estructura especifica un orden de ejecución particular establecido por restricciones de flujo de datos o de control. Para el ejemplo anterior, es preciso que primero se ejecute el SW que realiza la reserva del boleto aéreo, luego el SW que procesa el pago y finalmente el SW que envía el mensaje de confirmación de compra (es decir, una ejecución secuencial). Para el proceso de definición del SWC se pueden utilizar tecnologías tales como WS-BPEL¹ que es una especificación hecha bajo el precepto de modelar fácilmente procesos de negocio como un SWC o WSDL², OWL-S³ o UDDI⁴ para definir, expresar información semántica o descubrir un SW, de tal forma que se pueda elegir una composición que satisfaga los requisitos del usuario.

Luego, en la fase de ejecución, se deben invocar los SW del SWC de acuerdo a la definición realizada en la fase previa de composición. Para ejecutar un SWC es necesario un programa o un sistema que se encargue de invocar los SW de acuerdo al orden establecido en la fase de composición y se asegure de cumplir todas las restricciones que correspondan al SWC. A este tipo de programas los llamaremos *motores de ejecución*. Hay 2 esquemas ampliamente utilizados para implementar motores de ejecución: motores centralizados, en donde un coordinador se encarga de toda la ejecución, y motores distribuidos, donde la ejecución se realiza con la colaboración de diversos participantes sin un coordinador central [1]. Si hay muchos SW en la composición y el flujo de datos o de control permite ejecuciones paralelas de varios SW, se podría tomar ventaja de un modelo distribuido, ya que se podría reducir el tiempo total de ejecución del SWC, siempre y cuando el tiempo que tome la sincronización posterior de los datos sea despreciable.

Durante la ejecución de un SWC, existen diversas situaciones que pueden causar la falla de uno o más SW. En el ejemplo de la compra de boletos mencionado con anterioridad, es posible que el servicio de cobro no esté disponible al momento de

la compra, sin embargo, se podría reemplazar por otro SW de cobro que si esté funcionando, reintentar la operación o cancelar la compra completamente. Un SWC tolerante a fallas, es aquel que, aún en presencia de fallas finaliza la ejecución del SWC (e.g., reintentando la invocación, substituyendo o replicando el SW que falle) o abandona la ejecución asegurando un estado consistente del sistema (e.g., haciendo *roll-back* y compensando los SW que se ejecutaron exitosamente antes de la falla). En este sentido, la ejecución confiable de SWC se convierte en un mecanismo clave para enfrentar los retos de las aplicaciones de acceso libre y abierto en un ambiente dinámico y no confiable para asegurar consistencia del sistema aún en presencia de fallas [2].

Así la consideración del manejo de fallas debe atacarse en la fase de ejecución. Son los motores de ejecución los que deben encargarse de establecer mecanismos de recuperación ante fallas para asegurar la ejecución de la composición. En este contexto, las fallas durante la ejecución de un SWC pueden ser reparadas por recuperación hacia atrás (*Backward recovery*) o recuperación hacia adelante (*Forward recovery*). La recuperación hacia adelante implica deshacer todo el trabajo que se realizó exitosamente antes de la falla y retornar al estado inicial consistente (antes de que la ejecución del SWC comenzara), usando técnicas de *roll-back* y compensación. La recuperación hacia adelante trata de reparar la falla y continuar la ejecución; reintentar, replicación y substitución son algunas técnicas usadas. También se pueden utilizar propiedades transaccionales de los SW, que implícitamente describen su comportamiento en caso de fallas y aseguran la clásicas propiedades transaccionales ACID (*Atomicity, Consistency, Isolation, and Durability*).

Las propiedades transaccionales permiten recuperación hacia atrás y recuperación hacia adelante [5]. Estos enfoques de recuperación aseguran la propiedad todo-o-nada, cuando ocurre una falla. Sin embargo, para algunas consultas de usuarios, obtener respuestas parciales puede tener sentido y utilidad. Por lo tanto, se requieren de estrategias de recuperación alternativas que provean una propiedad intermedia en caso de fallas. La técnica de puntos de control (*checkpointing*) puede ser implementada para retornar las respuestas parciales obtenidas hasta el momento de una falla y poder reiniciar la ejecución posteriormente para finalizar con la ejecución faltante [6][7].

El motor de ejecución también debe considerar los requisitos no funcionales que se establecieron al momento de la definición del SWC. Generalmente, estos requisitos no funcionales definen la calidad de servicio solicitada por el usuario en términos de tiempo total de ejecución, costos, confiabilidad, etc. Durante la composición se seleccionan los SW de acuerdo a sus parámetros de calidad reportados previamente por estimación usando técnicas analíticas, de simulación, heurísticas o por trazas de ejecuciones previas [8][9][10]. Estas consideraciones se deben tomar en cuenta durante la ejecución del SWC, de tal forma que se asegure el cumplimiento de las restricciones impuestas por el usuario en términos de calidad de servicio. En particular, el tiempo total de la

¹Web Service Bussines Process Execution Language, 2007, <https://www.oasis-open.org/committees/download.php/23964/wsbpel-v2.0-primer.htm> tomado en Diciembre de 2014. Es una especificación de OASIS que permite modelar SWC como orquestación enfocándose en procesos de negocio.

²Web Service Description Language, 2001, <http://www.w3.org/TR/wsdl> tomado en Diciembre de 2014, se puede ver como una firma de un servicio web

³Semantic Markup for Web Services, 2004, <http://www.w3.org/Submission/OWL-S/> tomado en Diciembre de 2014, permite describir servicios web semánticos

⁴Universal Description Discovery and Integration, 2004, http://www.uddi.org/pubs/uddi_v3.htm tomado en Diciembre de 2014, es un mecanismo para registrar y localizar servicios web

ejecución y el tiempo promedio en la ejecución de un SW se pueden considerar para ofrecer una mayor compatibilidad con requisitos no funcionales preestablecidos. Estos requisitos se deben considerar incluso cuando hay una falla, ya que el objetivo primordial del motor es realizar la ejecución sin violar ningún tipo de restricciones predeterminadas. Durante la detección de fallas es necesario implementar un mecanismo que sea eficiente al tomar la decisión apropiada para realizar la reparación y que no altere significativamente los parámetros de calidad de servicio, como el tiempo de ejecución total, asegurando así la ejecución bajo los parámetros definidos.

El paradigma de crónicas ha sido usado para determinar fallas en sistemas dinámicos, permitiendo modelar las relaciones temporales entre eventos observables y describiendo los patrones de comportamiento del sistema. Una crónica es un conjunto de eventos relacionados con restricciones de tiempo que representan una interpretación de lo que está ocurriendo en el sistema estudiado en un momento dado [11]. En trabajos previos el paradigma de crónicas ha sido usado para detectar fallas en la ejecución de SWC [12].

Este trabajo propone un motor de ejecución, llamado CrESTa (*ChRonical-based Execution engine for composite web Service with fault TolerAnce*) que permite mejorar los tiempos de ejecución de un SWC utilizando un sistema distribuido en vez de uno centralizado, considerando mecanismos de tolerancia a fallas para la recuperación de la ejecución utilizando propiedades transaccionales y el tiempo de ejecución como aseguramiento de calidad de servicio. Adicionalmente se propone un modelo donde se busca evitar que los tiempos de detección de fallas desmejoren significativamente los tiempos de ejecución totales utilizando crónicas.

El trabajo está estructurado de la siguiente manera: en la Sección II se explica la representación de un SWC utilizando Redes de Petri Coloreadas y la detección de fallas utilizando crónicas; en la Sección III se describe la arquitectura de CrESTa, explicando el funcionamiento de sus componentes; la Sección IV presenta los trabajos relacionados a esta investigación; en la Sección V se da un caso de estudio donde se ejemplifica el funcionamiento de CrESTa; finalmente la Sección VI presenta nuestras conclusiones y trabajo futuro.

II. TOLERANCIA A FALLAS EN SERVICIOS WEB COMPUESTOS

Algunos estudios recientes en el área de SWC se han centrado específicamente en mecanismos de detección inmediata de fallas en sistemas distribuidos y en medidas de recuperación en presencia de éstas. A continuación se presentarán los conceptos que se utilizarán para proponer la construcción de un motor de ejecución de SWC distribuido tolerante a fallas.

A. SWC Tolerantes a Fallas Basado en Redes de Petri Coloreadas

Algunos motores de ejecución son capaces de manejar fallas, ya sea con manejo de excepciones [13][14], basado en propiedades transaccionales [15][16], usando técnicas de replicación [17][18] o por combinación de varios enfoques [4][19].

En los trabajos iniciales de soporte de tolerancia a fallas de SWC, sólo se manejaban construcciones de programación de bajo nivel como el manejo de excepciones (por ejemplo in WSBPEL). El manejo de excepciones normalmente es especificado explícitamente en tiempo de diseño, definiendo cómo son manejadas las excepciones y qué comportamiento asumen los SWC cuando es recibida una excepción. Más recientemente, la confiabilidad de SWC ha sido manejada a un nivel de abstracción más alto, es decir a nivel de la estructura del flujo de ejecución tales como *workflows*, grafos o Redes de Petri, y con tecnología de métodos independientes como propiedades transaccionales y replicación.

Cardinale et. al. en [3] explica cómo las Redes de Petri son utilizadas ampliamente en el modelaje de SW ya que capturan tanto los flujos de ejecución de los SW como su naturaleza distribuida. Adicionalmente, en el mismo trabajo se extienden las Redes de Petri Coloreadas para modelar las propiedades transaccionales de los SW involucrados en una composición.

Las propiedades transaccionales más usadas para un SW son **pivote**, **compensable** y **reintenable**. Si s es un SW, s es **pivote** (p) si una vez ejecutado sus efectos son permanentes y no se pueden deshacer semánticamente, es **compensable** (c) si es posible deshacer semánticamente la ejecución de s (ya sea por la llamada a otro servicio o al mismo con diversos parámetros, para hacer una regresión), o **reintenable** (r) si garantiza una terminación satisfactoria luego de un número finito de reintentos. Hay combinaciones posibles de propiedades, un servicio **pivote** puede ser **reintenable** (pr), así como un servicio **compensable** puede ser también **reintenable** (cr), lo que quiere decir que las propiedades transaccionales de un servicio s pertenecen al conjunto $\{p, pr, c, cr\}$.

Un SWC puede adquirir una propiedad transaccional agregada en término de las propiedades transaccionales de sus SW componentes. Así, un SWC cs es **atómico** (a) si al terminar la ejecución de todos los SW involucrados los cambios son permanentes y no se pueden deshacer semánticamente, **compensable** (c) si todos los componentes son compensables y **reintenable** (r) si todos sus componentes son reintentables. De nuevo pueden haber combinaciones de propiedades, teniendo SWC **atómicos** y **reintentables** (ar), y **compensables** y **reintentables** (cr), por lo que todas las propiedades transaccionales de cs están en el conjunto $\{a, ar, c, cr\}$

Las técnicas de recuperación soportadas por los propiedades transaccionales de SWC son:

- Recuperación hacia atrás: consiste en restaurar el estado que el sistema tenía al comienzo de la ejecución del SWC; es decir, todos los efectos producidos por el SW que falle y por los SW previamente ejecutados, son semánticamente deshechos por técnicas de *roll-back* o compesación.
- Recuperación hacia adelante: consiste en reparar la falla para permitir que el SW que falle, continúe su ejecución; reintento y sustitución son técnicas usadas para proveer recuperación hacia adelante.

Dada la gran proliferación de SW publicados en Internet, existen servicios equivalentes diseñados/desarrollados inde-

pendientemente por organizaciones diferentes, que pueden ser usados como componentes alternativos redundantes para ofrecer sistemas tolerantes a fallas. En este contexto, una réplica representa un SW funcionalmente equivalente. Las réplicas pueden ser usadas para replicación de ejecución o para sustitución permitiendo recuperación hacia adelante, sin depender de propiedades transaccionales. Estos enfoques de

recuperación aseguran la propiedad todo-o-nada, para asegurar la consistencia del sistema cuando ocurre una falla. Sin embargo, para algunas consultas de usuarios, obtener respuestas parciales puede tener sentido y utilidad. Por lo tanto, se requieren de estrategias de recuperación alternativas que provean una propiedad intermedia en caso de fallas. Cuando una falla ocurre, se generan puntos de control que contienen información de un estado avanzado de ejecución (incluyendo resultados parciales). Un SWC parcialmente ejecutado, contiene puntos de control que permite su reanudación posterior a partir de ese estado avanzado de ejecución [20][21].

Nuestro trabajo utiliza la definición de la extensión de Redes de Petri Coloreadas presentada por Cardinale et. al. en [3], llamada WSDN por sus siglas en inglés de Red de Dependencia de Servicio Web (Web Service Dependency Net). Esta extensión es una tupla (A, S, F, ξ) donde:

- A es un conjunto finito no vacío, que contiene las entradas y salidas de los SW;
- S es un conjunto finito de transiciones, cada transición es un SW;
- $F : (AxS) \cup (SxA) \rightarrow \{0, 1\}$ indica la presencia (1) o ausencia (0) de de arcos entre las entradas/salidas (A) y las transiciones (S);
- ξ es una función de coloración tal que $\xi : C_A \cup C_S$ con $C_A : A \rightarrow \Sigma_A$, $C_S : S \rightarrow \Sigma_S$ y $\Sigma_A = \{I, a, ar, c, cr\}$, $\Sigma_S = \{p, pr, a, ar, c, cr\}$, que representan las propiedades transaccionales del SWC (representado en la coloración de los parámetros de entrada) y las propiedades transaccionales de los SW (representado en la coloración de las transiciones) respectivamente.

Ilustremos con un ejemplo un SWC basado en esta definición. Supongamos que tenemos un SWC que modela una compra de uno o varios productos utilizando 2 tarjetas de débito. Como los servicios de conexión con los bancos suelen tardar, se desea crear un SWC que maneje las 2 operaciones de cobro simultáneamente. La Tabla I muestra la descripción de los SW y parámetros de este ejemplo.

Tabla I: Ejemplo de SWC

SW	Entrada	Salida	Prop. Trans.
s1	Tarjeta1, Orden de Compra	Token1	cr
s2	Tarjeta2, Orden de Compra	Token2	cr
s3	Tarjeta1, Token1	Comprobante1	pr
s4	Tarjeta2, Token2	Comprobante2	pr
s5	Comprobante1, Comprobante2, Orden de Compra	Factura, Orden de entrega	pr

El proceso sería el siguiente: los servicios $s1$ y $s2$ pertenecen a bancos distintos y se encargan de verificar si la información proporcionada de las tarjetas de débito es correcta, también generan un token en caso de que se tengan los fondos

necesarios para la operación. De ser así estos fondos son retenidos por una cantidad de tiempo definida por cada banco, este token permite identificar la transacción que reservó el dinero. Como estos servicios no generan cambios irreversibles, poseen la propiedad transaccional **compensable-reintenable**. Los servicios $s3$ y $s4$ reciben información de la tarjeta, el token que identifica la transacción y el dinero reservado en el banco; se encargan de descontar el monto correspondiente y emiten un comprobante de la transacción. Estos servicios son **pivotes-reintenable**, ya que una vez realizada la operación de cobro, ésta no se puede deshacer. Finalmente $s5$ toma los comprobantes y genera la factura de todo el proceso de compra; dado que la factura debe ser única y genera una orden de despacho, es un servicio tipo **pivote-reintenable**. Por las propiedades transaccionales de sus SW, el SWC sería *ar*, dado que todos sus componentes son reintentables. Se puede tener una representación gráfica de este SWC, tal como se muestra en la Figura 1.

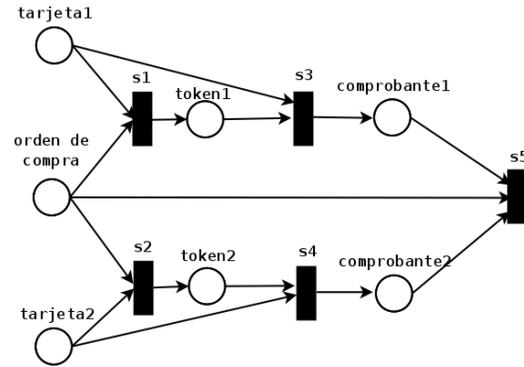


Figura 1: Ejemplo de SWC

De esta forma se puede representar de una forma robusta y sencilla todo lo necesario para la ejecución del SWC.

B. Detección de Fallas

En un escenario donde haya distintos elementos distribuidos ejecutando partes de un SWC, si uno de los elementos falla, es necesario que se active algún mecanismo de sincronización para que todas las partes se pongan de acuerdo sobre cuál va a ser el procedimiento a seguir para la recuperación de la falla. En este sentido, es necesario encontrar una forma de detectar inmediatamente la falla y disparar acciones que se encarguen de manejar la situación. Para este fin se pueden utilizar las **crónicas**.

Una crónica es un conjunto de eventos relacionados con restricciones de tiempo que representan una interpretación de lo que está ocurriendo en el sistema estudiado en un momento dado [11]. Las crónicas se basan en lógica temporal y son muy útiles para modelar sistemas dinámicos y con dependencia de ocurrencia de eventos. En [12], Vizcarrondo et. al. explican cómo cada crónica representa una situación normal o anormal, algún escenario o patrón de comportamiento. Estos eventos que conforman la crónica son observables y en el momento de su ocurrencia pueden generar acciones u otros eventos. Le Guillou et. al. en [22] definen una crónica C como un par

(E, T) donde E es el conjunto de eventos y T es el conjunto de restricciones entre los tiempos de ocurrencia de los eventos. Estas restricciones son las que permiten establecer un orden respecto a lo que debe ocurrir primero y lo que debe ocurrir después para detectar la crónica.

En [12] una crónica se representa con el siguiente modelo:

```
Cronica {
  Eventos {
    evento( $e_1, T_1$ ),
    evento( $e_2, T_2$ ),
    evento( $e_3, T_3$ )
  }
  Restricciones {
     $T_2 - T_1 < C_1$ ,
     $T_3 - T_2 < C_2$ 
  }
  acciones {
     $accion_1$ ,
     $accion_2$ 
  }
}
```

Donde:

- e_i representan los eventos para el reconocimiento de las crónicas.
- T_i son los puntos en el tiempo en la ocurrencia de los eventos.
- Restricciones: conjunto de restricciones entre los T_i .
- C_i Constantes que representan la diferencia entre la ocurrencia de dos puntos en el tiempo.
- Acciones: conjunto de acciones a ser ejecutados al momento de reconocer la crónica.

Supongamos que queremos definir una crónica que nos permita detectar la ocurrencia de un error en la ejecución de un SW. Definamos el conjunto de eventos E :

$$E = \{inicio, llamada, culminacion, error\}$$

Donde *inicio* corresponde a la inicialización del SW, *llamada* corresponde a la ejecución del SW, *culminacion* indica la terminación correcta, y *error* representa la falla en la ejecución. Definamos la crónica de la siguiente forma:

```
Cronica {
  Eventos {
    evento(inicio,  $T_1$ ),
    evento(llamada,  $T_2$ ),
    evento(error,  $T_3$ )
  }
  Restricciones {
     $T_2 - T_1 > 0$ ,
     $T_3 - T_2 > 0$ 
  }
  acciones {
    atenderFalla
  }
}
```

Esta crónica permite reconocer una ejecución fallida de un SW si ocurre un error luego de la *llamada*. T_1, T_2 y T_3 establecen el orden de los eventos, y al detectarse la crónica se llamará a *atenderFalla*. Esta acción, entonces decidirá, por ejemplo, de acuerdo a las propiedades transaccionales del SW, si el SW se puede reinvocar (si es **reintentable**) o hay que hacer *roll-back* (si es **pivote**); o simplemente informar sobre el error ocurrido.

De forma análoga, se puede definir una crónica que detecte un funcionamiento correcto de una ejecución de SW de la siguiente forma:

```
Cronica {
  Eventos {
    evento(inicio,  $T_1$ ),
    evento(llamada,  $T_2$ ),
    evento(culminacion,  $T_3$ )
  }
  Restricciones {
     $T_2 - T_1 > 0$ ,
     $T_3 - T_2 > 0$ 
  }
  acciones {
    culminacionExitosa
  }
}
```

En este caso se llama a *culminacionExitosa* al reconocer la crónica, y aunque no es común este tipo de detección, en un sistema distribuido podría servir para sincronizar toda la información entre los SW participantes, obtener el resultado final de la ejecución y entregar los parámetros de salida finales al usuario.

Vizcarrondo et. al. en [12] definen una extensión de este concepto para permitir reconocer eventos en un entorno distribuido, a esto lo llama **crónica distribuida**, y consiste en la descomposición de una crónica con eventos $E = \{E_1, \dots, E_N\}$ y restricciones T en subcrónicas, donde cada evento local pertenece al proceso o sitio de estudio. Formalmente: sea SC_i una subcrónica con eventos $E_i = \{E_{i1}, \dots, E_{iM}\}$ y restricciones T_i , donde $E_i \subset E$ y $T_i \subset T$, se tiene que para todos los procesos o sitios i se cumple que:

$$C(E, T) = UNION_{i=1 \dots N} (SC_i(E_i, T_i))$$

Donde *UNION* representa la unión de todas las subcrónicas cumpliendo las condiciones $E = \bigcup_{i=1}^N E_i$ y $T = \bigcup_{i=1}^N T_i$. Para este caso, el reconocimiento de la crónica global se puede llevar a cabo en las crónicas locales a través de **eventos enlazadores**, que una vez reconocidos comunican a los otros sitios o procesos la detección de la subcrónica.

Ilustremos con un ejemplo este concepto. Supongamos que tenemos una crónica C con los eventos: $E = \{e_1, \dots, e_7, EE_{sc1}\}$. Supongamos que esta crónica está conformada por 2 subcrónicas SC_1, SC_2 , $C = UNION(SC_1, SC_2)$ donde $E_1 = \{e_1, e_2, e_6, e_7\}$ y $E_2 = \{e_3, e_4, e_5, EE_{sc1}\}$. EE_{sc1} es el evento enlazador entre el sitio 1 y el sitio 2. La definición de estas subcrónicas se presentan en la Tabla II.

Tabla II: Definición de Subcrónicas

<pre> SC₁{ Eventos{ evento(e₁, T₁), evento(e₂, T₂), evento(e₆, T₆), evento(e₇, T₇) } Restricciones{ T₂ - T₁ > 0, T₆ - T₂ > 0, T₇ - T₆ > 0 } acciones{ emitir EE_{sc1} a SC₂ } } </pre>	<pre> SC₂{ Eventos{ evento(e₃, T₃), evento(e₄, T₄), evento(e₅, T₅), evento(EE_{sc1}, T_{sc1}) } Restricciones{ T₄ - T₃ > 0, T₅ - T₄ > 0, T_{sc1} - T₅ > 0 } acciones{ culminacionExitosa } } </pre>
---	--

En la Figura 2 se puede observar un gráfico de la crónica distribuida. La ocurrencia de los eventos sigue siendo determinada por los T_i , solo que, para este ejemplo, la subcrónica 2 (SC_2) es reconocida luego de que haya ocurrido EE_{sc1} .

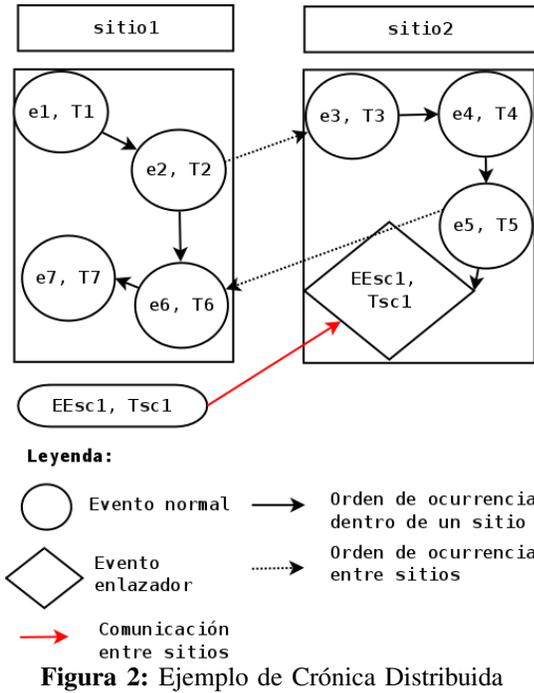


Figura 2: Ejemplo de Crónica Distribuida

Para la ejecución de SWC, se pueden crear crónicas que modelen la falla de los componentes del sistema distribuido, de tal forma que si hay un error en una parte del sistema, inmediatamente se pueda sincronizar la información necesaria y tomar decisiones inmediatas sobre qué mecanismo de recuperación se va a ejecutar. Entonces las acciones relacionadas con detección de falla (*atenderFalla*), implementarían algoritmos para decidir de acuerdo a las condiciones actuales de la ejecución (e.g., tiempos de ejecución transcurridos), a las propiedades transaccionales de los SW y a otras técnicas de recuperación disponibles, cuál es la mejor estrategia de reparación aplicable (recuperación hacia adelante o hacia atrás), manteniendo la calidad de servicio exigida inicialmente por el usuario.

III. ARQUITECTURA DE CRESTA

CrESTa (*ChRonical-based Execution engine for composite Web Service with fault TolerAnce*) es un motor de ejecución distribuido para SWC, conformado por diversas instancias que pueden ejecutarse en sitios remotos sobre Internet y se comunican para llevar a cabo su funcionalidad. En la Figura 3 se muestra una visión global de CrESTa. Cada instancia de CrESTa, que de ahora en adelante llamaremos Servicio de Ejecución (SE), recibe por HTTP⁵ las peticiones tanto de ejecución de SWC como otras necesarias dentro del sistema. Al momento de ejecutar un SWC recibe su representación como una Red de Petri Coloreada, los requisitos no funcionales de calidad de servicio, y de ser necesarias, las definiciones de crónicas distribuidas para la detección de fallas. El SE que haya recibido esta información comenzará con la ejecución y cuando sea necesario enviará esta información a otros SE para ejecutar de forma paralela el SWC. Una vez finalizado el proceso, si no hay fallas, se envía la respuesta.

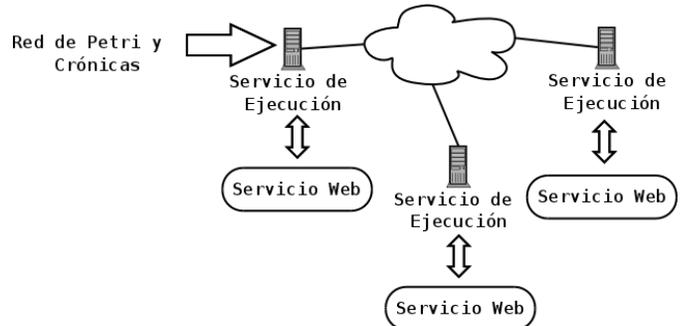


Figura 3: CrESTa: Sistema de Ejecución Distribuido, Visión Global

Cada SE va a estar compuesto por 4 módulos y tendrá una base de datos local donde se guardará información sobre la ejecución que sea necesaria por los módulos y se tenga conocimiento sobre el estado general del servicio en cualquier momento. En la Figura 4 se muestra un gráfico con los módulos propuestos.

Las operaciones que puede ejecutar un SE son las siguientes:

- 1) Descubrimiento: consiste en detectar otros SE que estén operativos, esto se hace para incrementar la escalabilidad del sistema, pues agregando más SE se podrán manejar más solicitudes de ejecución. Se tiene una lista estática con SE que estén siempre activos, y al conectarse con estos SE se envían la información de los otros SE que ya estén operativos.
- 2) Chequeo: permite detectar si un SE está operativo.
- 3) Información de carga: el SE informa al solicitante sobre su estado y cuánto trabajo tiene pendiente.
- 4) Actualizar información de servicio: permite actualizar la información que se tenga de otros SE que formen parte del sistema.
- 5) Ejecución de SWC: toma un SWC representado como una Red de Petri Coloreada y lo ejecuta paralelizando

⁵Hypertext Transfer Protocol <https://tools.ietf.org/html/rfc7230> tomado en Abril de 2015 protocolo de comunicación

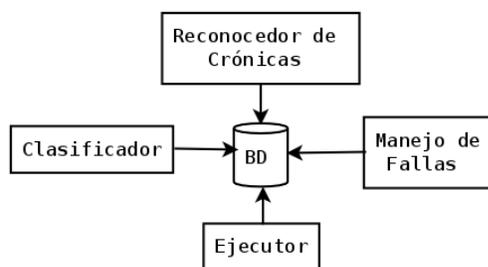


Figura 4: Arquitectura de un Servicio de Ejecución (SE) de CrESTa

peticiones cuando sea posible.

- 6) Ejecución de tolerancia a fallas: considera las propiedades transaccionales del servicio donde se generó la falla para determinar cuál es el mejor mecanismo de tolerancia a aplicar, ya sea, recuperación hacia atrás (utilizando compensaciones), o recuperación hacia adelante (utilizando sustitución o reintento del SW que haya fallado), todo dentro de los parámetros de calidad de servicio establecido por el usuario.
- 7) Sincronización: permite actualizar la información de un SWC que esté ejecutándose con la colaboración de varios SE al mismo tiempo.

Estas operaciones se desarrollan con la colaboración entre los módulos. A continuación se explica cada componente de un SE y las operaciones que realiza.

A. Módulo de Clasificación

Se encarga de recibir peticiones de servicios relacionados con los puntos 1 al 5, listados en la sección anterior. Atiende las solicitudes relacionadas a información del servicio (operaciones 1, 2, 3 y 4) y en cada caso responde inmediatamente la información solicitada. Si la petición es referente a ejecución de un SWC (operación 5), la agrega a la cola de trabajo del Módulo de Ejecución.

B. Módulo de Ejecución

Revisa la cola de trabajo tan pronto esté libre, toma una representación de SWC (que está modelado como una Red de Petri Coloreada) y comienza la ejecución desde donde se tenga indicado en la definición del SWC para realizar la operación 5. Si durante esta ejecución se detecta que es posible ejecutar SW de forma paralela, se solicita a los demás SE registrados información de carga y se determina cuál o cuáles son los mejores para ejecutar los SW paralelos, luego se instancian las crónicas correspondientes, se envía a los otros SE la ejecución del SWC (indicando uno de los SW paralelos) y por último se continúa con la llamada a alguno de los SW paralelos que no fueron delegados a otros SE. Si no hay SE disponibles, o si el mismo SE puede cumplir con la invocación de todos los SW paralelos, no se instancia ninguna crónica, dado que el mismo SE puede manejar todos los eventos que se generen a partir de la invocación de los SW paralelos (e.g., culminación exitosa, ocurrencia de una falla).

Antes de comenzar y al terminar una llamada a un SW, se establecen puntos de chequeo (*checkpoints*) que contienen la información que va a ser enviada al SW (sus parámetros de entrada) y la información devuelta (sus parámetros de salida) respectivamente. Estos puntos de chequeo mantienen la información necesaria, en caso de ser requerido que se retome posteriormente la ejecución, debido a una falla en el SWC. Adicionalmente, de ser detectado un error por el Módulo de Crónicas, se detiene la ejecución, de tal forma que se pueda sincronizar la información disponible con los otros SE involucrados en la ejecución distribuida y acordar el esquema de recuperación conveniente.

C. Módulo de Tolerancia a Fallas

Toma la información de un SWC cuya ejecución haya fallado en un momento dado, evalúa las propiedades transaccionales de los servicios ya ejecutados, considera las restricciones de calidad de servicio estimadas, decide y ejecuta el mecanismo de recuperación que corresponda, ya sea recuperación hacia atrás o hacia adelante, creación de puntos de control o simplemente devuelve la solución parcial encontrada. Esto corresponde a la operación 6 y es una adaptación al algoritmo definido por Angarita et. al. en [5].

La decisión cuál estrategia de recuperación es la apropiada depende de diversos factores, como las condiciones de ejecución actuales, las propiedades transaccionales de los SW, la calidad de servicio demandada por el usuario, la existencia de réplicas, que deben ser evaluados por este módulo para tomar la decisión apropiada de manera dinámica.

D. Módulo de Crónicas

Es una envoltura a un reconocedor de crónicas y tiene un funcionamiento similar al propuesto en [23], es decir, se tiene un analizador de crónicas que se alimenta con la información de éxito o error local (proporcionado por el Módulo de Ejecución), y recibe información de los Módulos de Crónicas de otros SE. Se encarga de avisar al Módulo de Ejecución que ocurrió un error. En la Figura 5 se muestra un ejemplo de interacciones entre distintos SE asociados a una crónica en caso de una falla. El Módulo de Ejecución es el que detecta si una llamada no fue exitosa, dado que recibe un código de respuesta por la llamada HTTP⁶, en caso de ser fallida le avisa al Módulo de Crónicas para que se disparen los eventos correspondientes en la crónica distribuida definida en ese momento.

Es posible utilizar las crónicas no sólo para la detección de fallas, sino para la sincronización de los servicios. Es decir, si se define una crónica distribuida para el caso en que todos los SW culminen exitosamente, al terminar todas las ejecuciones esta crónica se dispara y se intercambia toda la información correspondiente a la ejecución, esto corresponde a la operación 7. Posteriormente se decide cuál es el mejor SE para continuar la ejecución y se envía la petición correspondiente.

⁶Hypertext Transfer Protocol <https://tools.ietf.org/html/rfc7230> tomado en Abril de 2015 protocolo de comunicación

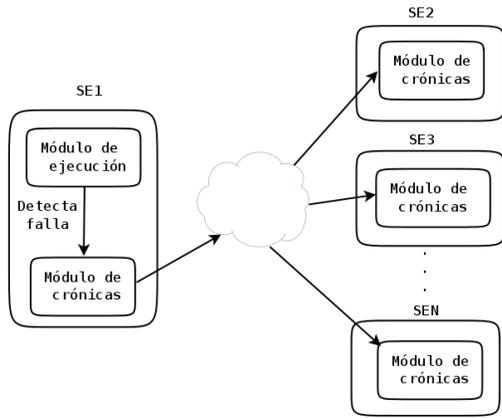


Figura 5: Interacción entre Módulos de Crónicas

IV. TRABAJOS RELACIONADOS

Los SWC han sido fuente de investigación de diversas universidades en el mundo en los últimos años. Se han hecho esfuerzos en lenguajes para la especificación (Redes de Petri Coloreadas, por ejemplo), donde se establece el qué y cuáles SW conforman el SWC, se han hecho investigaciones en técnicas de análisis, verificación y evaluación para probar y reparar errores de diseño, se han investigado técnicas para selección y especificación automática de SWC y se han propuesto diversos motores de ejecución [1].

En los siguientes puntos se estarán detallando trabajos relacionados sobre los cuales se fundamenta CrESTa, exponiendo sobre las representaciones más utilizadas para SWC, trabajos sobre calidad de servicio, el uso de crónicas en el mundo académico actual y diversas propuestas de motores de ejecución en la actualidad.

A. Representación y Calidad de Servicio de SWC

La representación de SWC se refiere a las estructuras usadas para describir el flujo de ejecución, flujo de datos y restricciones de control entre los SW que lo componen. Existen diversas especificaciones en la actualidad, tales como WS-BPEL⁷, WSCI⁸, WS-CDL⁹. Sin embargo, en [1] se propone extender las Redes de Petri Coloreadas para incorporar descripciones de propiedades no funcionales de los SW y modelar la automatización de la composición de los servicios. Este enfoque es fundamental para los mecanismos de tolerancia a fallas y por eso es utilizado en CrESTa. En [20] se propone, no sólo el uso de Redes de Petri Coloreadas sino también de ASK-Computational Tree Logic para verificar la correctitud del modelo. Sin embargo, para nuestra propuesta se asume que la representación de la composición ya es correcta.

⁷Web Service Bussines Process Execution Language, 2007, <https://www.oasis-open.org/committees/download.php/23964/wsbpel-v2.0-primer.htm> tomado en Diciembre de 2014. Es una especificación de OASIS que permite modelar SWC como orquestación enfocándose en procesos de negocio.

⁸Web Service Choreography Interface, 2002, <http://www.w3.org/TR/wsci/> tomado en Diciembre de 2014

⁹Web Service Choreography Description Language, 2005, <http://www.w3.org/TR/ws-cdl-10/> tomado en Diciembre de 2014

Si se utilizan mediciones respecto a cuánto tiempo lleva la ejecución de cada SW dentro de la composición y se compara con estimados iniciales, se puede determinar si la ejecución se encuentra dentro de los valores esperados en el marco de la calidad de servicio exigida por el usuario. Si ocurre una falla, estas mediciones permiten a los usuarios del servicio obtener respuestas en una cantidad de tiempo predeterminada, pues se puede elegir entre devolver la solución parcial obtenida, en caso que el mecanismo de tolerancia no cumpla con las especificaciones del usuario o elegir un mejor mecanismo de recuperación que permita mantener la ejecución en tiempos esperados. Considerar al menos el tiempo de ejecución como parámetro de calidad de servicio al momento de diseñar algún motor de ejecución para SWC es fundamental para el usuario. En [24] se propone una estimación de calidad de servicio en términos de parámetros no funcionales, aunque esta estimación se utiliza para elegir una composición en particular, la que mejor se adapte a los requisitos del usuario. En [25] además del tiempo de ejecución estimado de un SW y de un SWC, se considera el tiempo máximo de tolerancia para ejecución de un SWC, entre otros parámetros considerados en el modelo. En [26] se proponen estimaciones de calidad de servicio basadas en regiones.

Si la fase de composición de un SWC tomó en cuenta parámetros de calidad de servicios (requisitos no funcionales) exigidos por el usuario, es imprescindible, que el motor de ejecución del SWC sea garante de que realmente se cumplan dichos parámetros de calidad, aún en escenarios de fallas. Así que la propuesta de CrESTa, incluye la evaluación de parámetros de rendimiento para asegurar que en caso de fallas, la composición cumpla con los requisitos no funcionales del usuario.

B. Crónicas

El modelaje de sistemas reactivos (tales como sistemas operativos, protocolos, entre otros) ha llevado al estudio de la lógica temporal y de las crónicas, para tener un lenguaje mucho más expresivo y elegante para representar restricciones de tiempo y desencadenamiento de eventos. En este sentido, las crónicas permiten modelar un conjunto de eventos restringidos por relaciones temporales cualitativas o cuantitativas, como por ejemplo antes, 100 segundos antes, etc. [21]. En este campo, en [27] se plantea el uso de crónicas que sean equivalentes a Redes de Petri, de tal forma que se puedan inferir situaciones peligrosas en el funcionamiento de, por ejemplo, un aeropuerto. En [28], se extiende el concepto de crónicas para establecer el concepto de crónica distribuida, de tal forma que se puedan reconocer fallas en sistemas distribuidos.

En [27] se propone modelar Redes de Petri Coloreadas con crónicas para simular sistemas complejos y detectar actividades o analizar comportamientos.

En CrESTa, se usan las crónicas para detectar fallas y decidir el mejor esquema de recuperación, así como para modelar el comportamiento correcto de la ejecución de un SWC y lograr la sincronización requerida para respetar el flujo de ejecución

delineado en la estructura que representa el SWC (i.e., la Red de Petri Coloreada).

C. Motores de Ejecución y Frameworks de SWC

Se han hecho muchas propuestas sobre motores de ejecución para SWC que sean tolerantes a fallas. Algunos de ellos forman parte de frameworks que contienen mucho más que el motor de ejecución, pero cada uno presenta algún tipo de limitación que puede ser mejorado. En [4] se propone FACTS, que utiliza WS-BPEL¹⁰ como representación de SWC, y utiliza excepciones para detectar fallas en caso de errores en la ejecución del SWC. El mecanismo propuesto por FACTS agrega una carga extra al motor de ejecución por la evaluación de excepciones, lo cual puede ser mejorado con el uso de crónicas. En [29] se utiliza BPMN¹¹ para especificar el SWC, y propone modificar su estructura dinámicamente cuando se detecten fallas, dependiendo de configuraciones hechas por el diseñador del servicio. Estos sistemas tienen el problema de utilizar como definición una representación basada en orquestación, que hace difícil la especificación de alguna composición que no requiera un componente central. En [31] se propone reemplazar automáticamente durante la ejecución de un SWC aquellos servicios que presenten fallas dado una lista de servicios de reemplazo proporcionada por el usuario antes de la ejecución.

Utilizando crónicas se puede tener una implementación poco intrusiva y más confiable en la ejecución, pues sólo se reportarán eventos de falla y de culminación exitosa y se basará en un programa de detección de crónicas que ya ha sido probado y utilizado con anterioridad.

En [30] se propone THROWS, que es un motor de ejecución distribuido basado en listas que identifican los SWC a ser ejecutados, usando WS-SAGAS¹² para representar el SWC y los estados de los SW. Ofrece recuperación hacia adelante en caso de fallas, se concentra en el uso de CEP (Current Execution Process) y CEL (Candidate Engine List), que contiene información del progreso de la ejecución y una lista con servicios candidatos para ejecutar el SW. Este planteamiento fue mejorado posteriormente e integrado en un framework en [19], con el nombre FENECIA. Este trabajo tiene desventajas al momento de modelar el SWC. Primero no tiene la profundidad matemática que tienen las Redes de Petri Coloreadas, en segundo lugar, está aglutinando la representación del SWC junto con la información de la ejecución del mismo.

En el campo de la web semántica, en [33] se propone el

uso de una extensión de WS-BPEL¹³ llamado SPL¹⁴, que le agrega una descripción semántica utilizando RDF4S [34] a el SWC. Utiliza esta definición ontológica como un método para manejo de excepciones que integra SW, agentes y ontologías para definir procesos de negocio y darle contenido semántico. Adicionalmente, en [35] se utiliza OWL-S¹⁵ para descomponer peticiones de usuarios en SW y representarlos. En [36] se propone la generación automática de una Red de Petri Coloreada que representen la composición de un servicio web a través de un mediador que toma una solicitud y utilizando información semántica genera automáticamente la composición. Esto puede ser utilizado posteriormente para extender el trabajo realizado en este proyecto, de tal forma que se pueda tener además del motor, un sistema con definición de la composición.

Se han propuesto diversos mecanismos para manejar problemas de sincronización que se encuentran en la ejecución de SWC distribuidos. En [37] se detecta inconsistencias causadas por ciclos de dependencia y compensaciones, cambiando las dependencias de acuerdo a un conjunto de reglas que aseguran la consistencia de la ejecución. En [38] se propone un algoritmo de control de concurrencia basado en la dependencia de entrada y salida de cada SW dentro de la composición.

En [39] se propone FaCETa, donde se plantea un mecanismo para hacer recuperación hacia atrás (compensación) utilizando una Red de Petri Coloreada modificada, en particular, se invierten las aristas y se colocan SW de compensación que permitirán regresar el estado general de la composición al estado antes de la ejecución del SW original. En [5] y [25] se propone un motor de ejecución distribuido de SWC tolerante a fallas, que considera parámetros de calidad de servicio que toma un grafo que representa a un SWC como entrada, pero que considera un ambiente altamente confiable de ejecución como requisito para el sistema, se basa en el uso de MPI¹⁶ aunque se pueden utilizar tecnologías web para convertir al motor en un servicio para masificar su uso y es posible utilizar un mecanismo confiable poco intrusivo para la detección de fallas (que serían las crónicas).

Vizcarrondo et. al. en [23] proponen un middleware llamado ARMISCOM que utiliza crónicas distribuidas para detectar fallas en composición de servicios. Sin embargo, no consideran las propiedades transaccionales de los SW, no toma en cuenta la calidad de servicio para decidir dinámicamente la mejor estrategia de recuperación y es un middleware, no un motor de ejecución compuesto por múltiples instancias distribuidas, como lo que se está proponiendo en este trabajo.

En resumen, ninguno de los trabajos mencionados con anterioridad utilizan el modelo de crónicas junto con las Redes de Petri Coloreadas para la detección de fallas, se espera

¹⁰Web Service Business Process Execution Language, 2007, <https://www.oasis-open.org/committees/download.php/23964/wsbpel-v2.0-primer.htm> tomado en Diciembre de 2014. Es una especificación de OASIS que permite modelar SWC como orquestación enfocándose en procesos de negocio.

¹¹Business Process Model and Notation, 2011, <http://www.omg.org/spec/BPMN/2.0/> tomado en Abril de 2015. Es una especificación que permite modelar procesos de negocios en base a eventos, actividades y flujos, similar a un diagrama de flujos pero con artefactos adicionales.

¹²Modelo Transaccional para especificación y ejecución de composición de SW, tomado de [32] define un SWC como un conjunto de transacciones que se pueden definir recursivamente para representar la composición

¹³Web Service Business Process Execution Language, 2007, <https://www.oasis-open.org/committees/download.php/23964/wsbpel-v2.0-primer.htm> tomado en Diciembre de 2014. Es una especificación de OASIS que permite modelar SWC como orquestación enfocándose en procesos de negocio.

¹⁴Semantic Programming Language
¹⁵Semantic Markup for Web Services, 2004, <http://www.w3.org/Submission/OWL-S/> tomado en Diciembre de 2014

¹⁶Message Passing Interface, 2012, <http://www.mpi-forum.org/docs/docs.html> tomado en Diciembre de 2014

que el uso de crónicas permita una implementación segura, eficiente, confiable y poco intrusiva de un motor de ejecución, así como las Redes de Petri Coloreadas permitirán representar propiedades transaccionales de SW para el manejo de tolerancia de fallas de una forma genérica y precisa sin la necesidad de otras estructuras (como el CEP y el CEL de THROWS) para expresar el flujo de un SWC.

V. CASO DE ESTUDIO

En esta sección se ilustrará el funcionamiento y comportamiento de CrESTa, a través de un ejemplo.

Supongamos que una persona desea realizar un viaje de visita a alguna ciudad y quiere que automáticamente se cree una sugerencia de viaje incluyendo costos, fechas y sitios de estadía, considerando cierta cantidad de visitas a lugares de la ciudad. Supongamos que las restricciones del usuario son:

- Ciudad de destino
- Reputación del sitio de estadía
- Fecha aproximada de viaje y duración del mismo
- Presupuesto
- Lugares que quiere visitar
- Preferencia de transporte

Si en la ciudad hay servicios web que dan la disponibilidad de hoteles, transporte y ventas de boletos a sitios turísticos altamente deseados, es posible establecer una composición de servicios web que satisfagan las restricciones del usuario. Supongamos que los SW necesarios para cumplir la petición son los descritos en la Tabla III.

Tabla III: Servicios del Ejemplo Motivador

SW	Entrada	Salida	Prop. Trans.
hospedaje	fecha de reserva, reputación, presupuesto	fecha	cr
visita1	fecha	fecha de visita1	cr
visita2	fecha	fecha de visita2	cr
visita nocturno	fecha	fecha de visita nocturno	cr
visita3	fecha	fecha de visita3	cr
transporte	fecha de visita1, fecha de visita2, fecha de visita nocturno, fecha de visita3	boletos	cr

El servicio de hospedaje recibe como entrada una reputación que define la calidad esperada del hotel (ejemplo: 5 estrellas, 4 estrellas, 3 estrellas), el presupuesto disponible por el usuario y una fecha tentativa de reserva, luego chequea la disponibilidad en los hoteles de la ciudad y finalmente emite una fecha posible de reserva. Los servicios de visitas a sitios turísticos reciben una fecha tentativa de estadía del usuario, revisan su disponibilidad y emiten una fecha que se apegue a los parámetros recibidos. Finalmente el servicio de transporte recibe todas las fechas de visitas y determina los horarios de transporte y boletos para desplazarse entre las visitas. Como solamente se está haciendo una consulta, todos los servicios son **compensables** y es posible reintentar la solicitud, por eso todos tienen la propiedad transaccional **compensable reintentable** (cr). En la Figura 6 se muestra una Red de Petri

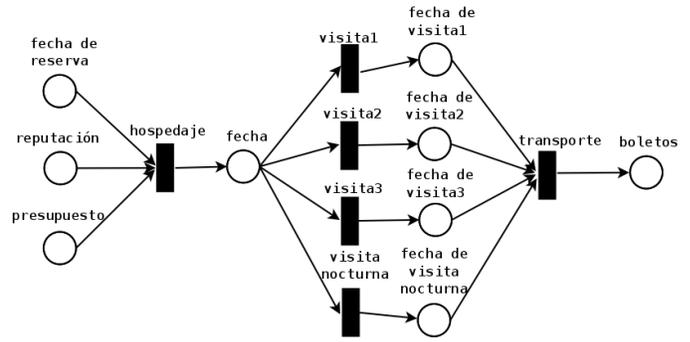


Figura 6: SWC del Caso de Estudio

Coloreada con una posible representación del SWC generado a partir de los servicios y la solicitud del usuario. La generación de esta composición no está considerada en este trabajo, se asume que ya hubo alguna herramienta que se encargó de procesar la solicitud y los requerimientos para generar correctamente el SWC.

El siguiente paso consiste en ejecutar la composición y devolver los resultados obtenidos para ser mostrados al usuario. Supongamos que hay 4 Servicios de Ejecución (SE) de CrESTa activos: SE_1 , SE_2 , SE_3 y SE_4 y que la ejecución se lleva a cabo de la siguiente manera:

- 1) SE_1 recibe la representación del SWC, la definición de crónicas distribuidas y los requisitos no funcionales. Esta información es procesada por el Módulo Clasificador, quien determina que es una solicitud de ejecución, por lo que la introduce en la cola de trabajos del Módulo de Ejecución.
- 2) Cuando el Módulo de Ejecución esté disponible, se comienza la ejecución, llamando al servicio de hospedaje.
- 3) Con el resultado del SW de hospedaje, se envía a SE_2 , SE_3 y SE_4 información para que se realice la llamada de los servicios *visita1*, *visita2* y *visita nocturna* respectivamente en paralelo, mientras que SE_1 se encarga de *visita3*. Se instancia la crónica distribuida asociada a esta ejecución paralela.
- 4) Al terminar la ejecución de los servicios mencionados, se concentran las respuestas en uno de los nodos, el último que terminó (supongamos que el SE_1), y luego se llama al servicio de transporte.
- 5) Se envían las respuestas al usuario.

Durante el proceso descrito, en los puntos 2, 3 o 4 pudo ocurrir algún error, ya sea por agotarse el tiempo de espera para respuestas del servicio (quizás el servicio esté muy ocupado) o pudo haber algún error en la conexión. Para la detección de estos errores se utilizan crónicas. A continuación se presentan crónicas para detectar errores en esos puntos.

Para el punto 2 se define la crónica $C_{hospedaje}$:

Eventos: *inicio, ejecución, error.*

Crónica:

$$C_{hospedaje} \{ \text{Eventos} \{ \text{evento}(\text{inicio}, T_1), \dots \} \}$$

```

evento(ejecucion, T2),
evento(error, T3),
}
Restricciones{
T2 - T1 > 0,
T3 - T2 > 0
}
acciones{
errorHospedaje()
}
}
    
```

Si ocurre alguno de los errores mencionados con anterioridad en SE_1 el Módulo de Ejecución lo informa al Módulo de Crónicas y éste llama a la función *errorHospedaje()* inmediatamente. Esta función se encuentra dentro del Módulo de Tolerancia a Fallas, donde se evalúan los requisitos no funcionales definidos y se decide si realizar recuperación hacia atrás o hacia adelante.

Para detectar el error en el punto 3 se deben instanciar 4 crónicas distribuidas, una por cada posible error en los SW *visita1*, *visita2*, *visita3* y *visita nocturna*. Como hay 4 SW ejecutándose en 4 *SE* entonces cada crónica debe estar compuesta por 4 subcrónicas. Ejemplificaremos la definición de la crónica distribuida para detección de error en *visita3*, que llamaremos $C_{visita3}$. Las 4 subcrónicas que la componen las llamaremos: $SC_{visita1}$, $SC_{visita2}$, $SC_{visita3}$ y SC_{visita_noc} . Definamos el conjunto de eventos $E_{visita3}$:

$$E_{visita3} = \{inicio, ejecucion, error\}$$

El conjunto de eventos enlazadores $EE_{visita3}$ viene dado por:

$$EE_{visita3} = \{EESC_{E12}^+, EESC_{E2}^-, EESC_{E13}^+, EESC_{E3}^-, EESC_{E14}^+, EESC_{E4}^-\}$$

Los eventos $EESC_{E12}^+$, $EESC_{E13}^+$, $EESC_{E14}^+$ se disparan desde el SE_1 y son los encargados de avisar a SE_2 , SE_3 y SE_4 que hubo un error en *visita3*. En la Figura 7 se puede observar la relación entre los eventos en cada uno de los servicios.

La crónica distribuida se puede ver de la siguiente forma:

```

Cvisita3{
  Eventos{
    evento(SE1.inicio, T12),
    evento(SE2.inicio, T21),
    evento(SE3.inicio, T31),
    evento(SE4.inicio, T41),
    evento(SE1.ejecucion, T12),
    evento(SE2.ejecucion, T22),
    evento(SE3.ejecucion, T32),
    evento(SE4.ejecucion, T42),
    evento(SE1.error, T13),
    evento(SE1.EESCE12+, T14),
    evento(SE2.EESCE2-, T23),
    evento(SE1.EESCE13+, T15),
    evento(SE2.EESCE3-, T33),
  }
}
    
```

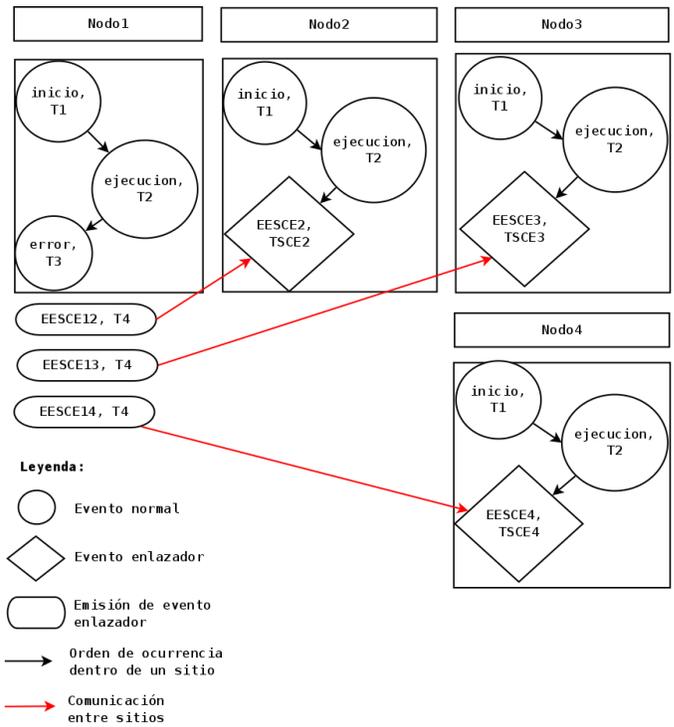


Figura 7: Crónica Distribuida para Detectar un Error en visita3

```

evento(SE1.EESCE14+, T16),
evento(SE2.EESCE4-, T43),
}
Restricciones{
T12 - T11 > 0,
T13 - T12 > 0,
T14 - T13 > 0,
T15 - T14 > 0,
T16 - T15 > 0,
T22 - T21 > 0,
T23 - T22 > 0,
T32 - T31 > 0,
T33 - T32 > 0,
T42 - T41 > 0,
T43 - T42 > 0
}
acciones{
falloDeEjecucion()
}
}
    
```

Esa secuencia de eventos vista en la crónica distribuida quiere decir que una vez iniciados los procesos de ejecución en cada *SE*, si ocurre el error en SE_1 que está ejecutando *visita3* se avisa a los demás *SE*. Una vez detectada la falla utilizando las propiedades transaccionales y considerando parámetros de calidad de servicio en cuanto a ejecución total del SWC y la ejecución estimada de cada SW, se puede establecer qué mecanismo de recuperación (hacia atrás, hacia adelante o puntos de control) se debe activar para cumplir con los requerimientos definidos por el usuario.

Para el punto 4 se define la crónica $C_{transporte}$:

Eventos: *inicio, ejecucion, error.*

Crónica:

```

Ctransporte {
  Eventos {
    evento(inicio,  $T_1$ ),
    evento(ejecucion,  $T_2$ ),
    evento(error,  $T_3$ ),
  }
  Restricciones {
     $T_2 - T_1 > 0$ ,
     $T_3 - T_2 > 0$ 
  }
  acciones {
    errorTransporte()
  }
}

```

Y se sigue el mismo procedimiento que para el punto 2.

Note que la definición de las crónicas es independiente de los mecanismos de recuperación que se puedan considerar y de los enfoques de evaluación de condiciones de ejecución y de parámetros de calidad para decidir dinámicamente cuál es el mejor esquema de recuperación (hacia adelante, hacia atrás, puntos de control). De esta manera, se pueden incorporar distintos mecanismos, enfoques y estrategias que se activen de acuerdo a las condiciones evaluadas para ejecutar algoritmos de tolerancia a fallas o implementar sincronizaciones requeridas en una ejecución exitosa del SWC.

VI. CONCLUSIONES Y TRABAJO FUTURO

Con CrESTa se busca tener un sistema robusto, tolerante a fallas y con una alta accesibilidad utilizando una arquitectura orientada a servicio para la ejecución de Servicios Web Compuestos. El uso de crónicas ayuda a este fin con la detección inmediata de fallas y se puede combinar con la evaluación de parámetros de calidad, como el tiempo de ejecución, y la consideración de las propiedades transaccionales de los servicios web para decidir dinámicamente el mejor enfoque de recuperación de fallas, sin que el usuario detecte siquiera que se presentó algún inconveniente.

Este sistema por ser distribuido, presenta un problema que un sistema centralizado no tiene y es la confidencialidad en el intercambio de información entre los SE, pues pueden estar sobre una red insegura y posiblemente se maneje información sensible al momento de contactar los SW de la composición. Actualmente se está trabajando en un esquema de seguridad que permita garantizar la confidencialidad en caso de ser solicitado y adicionalmente la autenticidad de la respuesta, para asegurar que la respuesta de CrESTa no haya sido alterada.

También se trabaja en automatizar el proceso de definición de crónicas, de tal forma que para una ejecución solamente se de la Red de Petri Coloreada y los requisitos no funcionales.

Adicionalmente se someterá el sistema a diversas pruebas de rendimiento para asegurar que el esquema de crónicas

efectivamente permite mantener los tiempos de ejecución razonables en situaciones de fallas.

REFERENCIAS

- [1] Y. Cardinale, J. El Haddad, M. Manouvrier y M. Rukoz, *Web Service Composition Based on Petri Nets: Review and Contribution*, Resource Discovery, serie Lecture Notes in Computer Science. Springer Berlin Heidelberg, vol. 8194, pp. 83–122, Enero 2013.
- [2] Q. Yu, X. Liu, A. Bouguettaya y B. Medjahed, *Deploying and Managing Web Services: Issues, Solutions, and Directions*, The VLDB Journal, vol. 17, no. 3, pp. 537–572, Mayo 2008.
- [3] Y. Cardinale, J. El Haddad, M. Manouvrier y M. Rukoz, *CPN-TWS: a Coloured Petri-net Approach for Transactional-QoS Driven Web Service Composition*, International Journal of Web and Grid Services, vol. 7, no. 1, pp. 91–115, Febrero 2011.
- [4] A. Liu, Q. Li, L. Huang y M. Xiao, *FACTS: A Framework for Fault-Tolerant Composition of Transactional Web Services*, IEEE Transactions on Services Computing, vol. 3, no. 1, pp. 46–59, Abril 2010.
- [5] R. Angarita, Y. Cardinale y M. Rukoz, *Reliable Composite Web Services Execution: Towards a Dynamic Recovery Decision*, Electronic Notes in Theoretical Computer Science, vol. 302, pp. 5–28, Febrero 2014.
- [6] R. Angarita, Y. Cardinale y M. Rukoz, *FACETA*: Checkpointing for Transactional Composite Web Service Execution Based on Petri-Nets*, Procedia Computer Science, vol. 10, pp. 874–879, 2012.
- [7] E. Sindrilaru, A. Costan y V. Cristea, *Fault Tolerance and Recovery in Grid Workflow Management Systems*, in proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 475–480, Krakow, Poland, Febrero 2010.
- [8] F. Lelli, G. Maron y S. Orlando, *Towards Response Time Estimation in Web Services*, in proceedings of the IEEE International Conference on Web Services (ICWS 2007), pp. 1138–1139, Salt Lake City, USA, Julio 2007.
- [9] M. Dharmendra y R. Mahajan, *A Novel Approach for Performance Estimation of Soap-based Web Services*, International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 2, pp. 1138–1139, Febrero 2012.
- [10] X. Zhang, Y. Yin y B. Zhang, *A Service Mining Approach for Time Estimation of Composite Web Services*, Applied Informatics and Communication, serie Communications in Computer and Information Science, vol. 228, pp. 486–492, Agosto 2011.
- [11] C. Dousson, *Extending and Unifying Chronicle Representation with Event Counters*, in proceedings of the 15th European Conference on Artificial Intelligence, pp. 257–261, Lyon, France, Julio 2002.
- [12] J. Vizcarrondo, J. Aguilar, E. Exposito y A. Subias, *Building Distributed Chronicles for Fault Diagnostic in Distributed Systems using Continuous Query Language (CQL)*, International Journal of Engineering Development and Research, vol. 3, no. 1, pp. 131–144, Enero 2015.
- [13] D. Sherman, *BPEL: Make Your Services Flow. Composing Web Services into Business Flow*, Journal in Web Services, vol. 3, no. 7, pp. 16–21, 2003.
- [14] OASIS, *Web Services Business Process Execution Language (WS-BPEL), Version 2.0* <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>, Abril 2007.
- [15] O. Bushehrian, S. Zare y N. Keihani, *A Workflow-Based Failure Recovery in Web Services Composition*, Journal of Software Engineering and Applications, vol. 5, no. 4, pp. 89–95, Febrero 2012.
- [16] J. El Haddad, M. Manouvrier y M. Rukoz, *TQoS: Transactional and QoS-Aware Selection Algorithm for Automatic Web Service Composition*, IEEE Transactions on Services Computing, vol. 3, no. 1, Abril 2010.
- [17] J. Behl, T. Distler, F. Heisig, R. Kapitzka y M. Schunter, *Providing Fault-tolerant Execution of Web-service based Workflows within Clouds*, in proceedings of the 2nd International Workshop on Cloud Computing Platforms (CloudCP), Bern, Switzerland, Abril 2012.
- [18] W. Zhou y L. Wang, *A Byzantine Fault Tolerant Protocol for Composite Web Services*, in proceedings of the International Conference on Computational Intelligence and Software Engineering (CiSE), pp. 1–4, Wuhan, China, Diciembre 2010.
- [19] N. Lakhali, T. Kobayashi y H. Yokota, *FENECIA: Failure Endurable Nested-Transaction based Execution of Composite Web Services with Incorporated State Analysis*, The VLDB Journal, vol. 18, no. 1, pp. 1–56, Enero 2009.
- [20] M. Souilah, B. Bérard y M. Boufaïda, *Analyzing Behavioral Compatibility for Web Service Choreography Using Colored Petri Nets and ASK-CTL*, in proceedings of the Sixth International Conferences on Advanced Service Computing, pp. 32–39, Venice, Italy, Mayo 2014.

- [21] J. Aguilar, *Temporal Logic from the Chronicles Paradigm: Learning and Reasoning Problems, and its Applications in Distributed Systems*, primera edición, LAP Lambert Academic Publishing, 2011.
- [22] X. Le Guillou, M. Cordier, S. Robin, L. Rozé y otros, *Chronicles for On-line Diagnosis of Distributed Systems*, in proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008), vol. 8, pp. 194–198, Patras, Greece, Mayo 2008.
- [23] J. Vizcarrondo, J. Aguilar, E. Exposito y A. Subias, *ARMISCOM: Autonomic Reflective Middleware for Management Service Composition*, in proceedings of the Global Information Infrastructure and Networking Symposium (GIIS 2012), pp. 1–8, Choroní, Venezuela, Diciembre 2012.
- [24] E. Blanco, Y. Cardinale, M. Vidal, *Aggregating Functional and Non-Functional Properties to Identify Service Compositions*, Engineering Reliable Service Oriented Architecture: Managing Complexity and Service Level Agreements, pp. 1–36, Marzo 2011.
- [25] R. Angarita, Y. Cardinale, M. Rukoz, *Dynamic Recovery Decision During Composite Web Services Execution*, in proceedings of the Fifth International Conference on Management of Emergent Digital EcoSystems (MEDES 2013), pp. 187–194, Luxembourg, Luxembourg, Octubre 2013.
- [26] W. Lo, J. Yin, S. Deng, Y. Li y Z. Wu, *Collaborative Web Service QoS Prediction with Location-based Regularization*, in proceedings of the IEEE 19th International Conference on Web Services (ICWS 2012), pp. 464–471, Honolulu, USA, Junio 2012.
- [27] C. Choppy, O. Bertrand y P. Carle, *Coloured Petri Nets for Chronicle Recognition*, Reliable Software Technologies–Ada-Europe 2009, pp. 266–281, Mayo 2009.
- [28] J. Vizcarrondo, J. Aguilar, A. Subias y E. Exposito, *Distributed Chronicles for Recognition of Failures in Web Services Composition*, in proceedings of the XXXIX Latin American Computing Conference (CLEI 2013), pp. 1–10, Naiguata, Venezuela, Octubre 2013.
- [29] K. Fang, J. Li, H. Sun, Y. Zhao y H. Zeng, *Strategy-based Two-Level Fault Handling Mechanism for Composite Service*, in proceedings of the IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS 2011), pp. 494–499, Beijing, China, Julio 2011.
- [30] N. Lakhali, T. Kobayashi y H. Yokota, *Throws: An Architecture for Highly Available Distributed Execution of Web Services Compositions*, in proceedings of the 14th International Workshop on Research Issues on Data Engineering: Web Services for e-Commerce and e-Government Applications, pp. 103–110, Boston, USA, Marzo 2004.
- [31] S. Gupta y P. Bhanodia, *A Fault Tolerant Mechanism for Composition of Web Services Using Subset Replacement*, International Journal of Advanced Research in Computer and Communication Engineering, vol. 2, no. 8, pp. 3080–3085, Agosto 2013.
- [32] N. Lakhali, Y. Kobayashi y H. Takashi, *WS-SAGAS: Transaction Model for Reliable Web Services Composition Specification and Execution*, DBSJ letters, vol. 12, no. 2, pp. 17–20, 2001.
- [33] K. Zhao, L. Zhang y S. Ying, *Ontology-Based Exception Handling for Semantic Business Process Execution*, Journal of Software, vol. 7, no. 8, pp. 1791–1798, 2012.
- [34] D. Xie, S. Ying, J. Yao y B. Xiao, *A Resource Description Framework for Service*, in proceedings of the 2007 International Conference on Wireless Communications, Networking and Mobile Computing (WiCom 2007), pp. 3351–3354, Shanghai, China, Septiembre 2007.
- [35] N. Sasikaladevi y L. Arockiam, *Extended WS-FIT Model to Enhance the Fault Tolerance in the Dynamic Composite Web Service*, in proceedings of the 3rd International Conference on Electronics Computer Technology (ICECT 2011), vol. 5, pp. 21–25, Kanyakumari, India, Abril 2011.
- [36] A. Marakhimov, J. Yim y J. Joo, *Petri Net Based Semantic Web Service Composition*, International Journal of Multimedia and Ubiquitous Engineering (IJMUE), vol. 9, no. 2, pp. 281–290, 2014.
- [37] Z. Zhao, J. Wei, L. Lin y X. Ding, *A Concurrency Control Mechanism for Composite Service Supporting User-Defined Relaxed Atomicity*, in proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference (COMPSAC 2008), pp. 275–278, Turku, Finland, Agosto 2008.
- [38] S. Sundar y R. Kanchana, *Handling Concurrency Control Problem in Web Service Compositions*, in proceedings of the Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT 2013), pp. 1–5, Tiruchengode, India, Julio 2013.
- [39] R. Angarita, Y. Cardinale y M. Rukoz, *Faceta: Backward and Forward Recovery for Execution of Transactional Composite WS*, in proceedings of the 5th International Workshop on Resource Discovery (RED 2012) at the 9th Extended Semantic Web Conference (ESWC 2012), pp. 89–103, Heraklion, Greece, Mayo 2012.

Integrando el Análisis Estratégico del Negocio en el Modelado de Metas

Mayela Delgado¹, Alfredo Matteo², Francisca Losavio²
mdelgadoh@uc.edu.ve, alfredojose.matteo@gmail.com, francislosavio@gmail.com

¹ Departamento de Computación, Universidad de Carabobo, Valencia, Venezuela

² Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: En este trabajo se propone la inclusión de conceptos básicos del análisis estratégico de la organización en la construcción de modelos de metas, con la finalidad de interrelacionar explícitamente el desarrollo de soluciones de software con las estrategias organizacionales. Esto contribuiría a clarificar el aporte que el software debe efectuar a la proposición de valor de la organización. Para modelar la alineación software-estrategia se plantea una extensión al Lenguaje de Requisitos Orientado a Metas (GRL, por sus siglas en inglés). GRL es una notación gráfica establecida como un estándar dentro del campo de la Ingeniería de Requisitos. La extensión de GRL, denominada GRL-SA (del inglés, GRL focused on Strategic Alignment), comprende: (1) la incorporación de los conceptos fundamentales para definir la dirección estratégica de una organización (visión, misión, cursos de acción) y (2) la ubicación de cada meta en el nivel de abstracción correspondiente según su alcance. El uso de los niveles de abstracción apoya el razonamiento requerido para la descomposición de metas y el análisis de contribución entre las metas de un nivel a otro, ya sea por refinamiento o por generalización. Para validar la propuesta realizada se presenta la aplicación de GRL-SA en un caso de estudio.

Palabras Clave: Análisis Estratégico; Alineación Estratégica; GRL; Ingeniería de Requisitos Orientada a Metas.

Abstract: This work proposes to include basic concepts about organizational strategic analysis into goal models construction, in order to link explicitly software solutions development and organizational strategies. This should contribute to explain how software helps to organization's value proposition. For modelling the software-strategy alignment it is suggested to extend the Goal-oriented Requirements Language (GRL). GRL is a graphical notation considered as a standard into the Requirements Engineering field. The GRL extension, named GRL-SA (GRL focused on Strategic Alignment), comprises: (1) the inclusion of fundamental concepts relative to strategic management (vision, mission, courses of action) and (2) the assignment of an abstraction level to each goal considering its scope. The use of abstraction levels supports the rationale required for goals refinement and the contribution analysis between goals of one level to another, either by refinement or by generalization. In order to validate this proposal, GRL-SA was applied to a case study.

Keywords: Strategic Analysis; Strategic Alignment; GRL; Goal-oriented Requirement Engineering.

I. INTRODUCCIÓN

La Ingeniería de Requisitos (IR) orientada a metas ha proporcionado aportes significativos al desarrollo de software [1-7]. Se fundamenta en la comprensión de las intenciones de las partes interesadas (del inglés, stakeholders) sin hacer referencia a los objetivos, propiedades o uso del sistema/software a ser desarrollado [7]. El término meta se refiere a un fin general que debe ser alcanzado y es de naturaleza cualitativa [8]. Las metas proporcionan directrices para enfocar las acciones a ejecutar. Las metas de una solución de software contribuyen a determinar su valor y justifican su desarrollo [6][7]. Las técnicas de la IR orientada a metas están enfocadas en el análisis de la organización para derivar las metas de un software y, cuando aplique, del sistema que lo

comprende [1][4][5][6]; estas metas conducen a la obtención del conjunto de requisitos que el software debe satisfacer.

El estudio presentado en [9] indica que un porcentaje importante de soluciones de software no satisfacen las necesidades de la organización porque no han sido dirigidas desde etapas tempranas del ciclo de desarrollo de software hacia el logro de las metas organizacionales. Al respecto, en esta investigación se propone la integración del modelado de metas aplicado en la IR y el análisis estratégico [10] de la organización como un medio para asegurar la alineación de las metas de un sistema/software y las metas organizacionales; de esta manera, los proyectos de desarrollo de software estarían vinculados, explícitamente, a las metas y estrategias organizacionales, y el rol y alcance del producto final se

establecerían en función de la contribución de valor que debe efectuar para que la organización alcance fines específicos.

Para la realización de la integración del modelado de metas y el análisis estratégico se propone una extensión del Lenguaje de Requisitos Orientado a Metas (en inglés GRL, Goal-oriented Requirements Language) para (1) incluir en el modelado de metas conceptos relacionados con el análisis estratégico de la organización y (2) asociar las metas expresadas en diferentes niveles de abstracción con los niveles organizacionales de dirección y gestión. GRL es una notación visual enfocada al modelado de intenciones, metas de negocio y requisitos no funcionales [11][12]. Forma parte de la Notación de Requisitos de Usuario (en inglés URN, User Requirements Notation) estandarizada por la International Telecommunication Union en 2008 [13]. Su selección se hizo considerando el análisis presentado en [14].

Este trabajo se estructuró de la siguiente manera. En la Sección II, se presentan los conceptos fundamentales del análisis estratégico, específicamente aquellos que determinan la dirección estratégica de una organización; también se define qué es la alineación estratégica y su importancia. La Sección III introduce los conceptos básicos de GRL y su metamodelo. En la Sección IV se explica la extensión propuesta en el metamodelo de GRL. La Sección V muestra la aplicación de la propuesta desarrollada mediante un caso de estudio. Por último, la Sección VI presenta las conclusiones sobre el trabajo realizado.

II. CONCEPTOS BÁSICOS EN EL ANÁLISIS ESTRATÉGICO. ALINEACIÓN ESTRATÉGICA

El análisis estratégico es un proceso que conduce al desarrollo de la estrategia de un negocio, puede abarcar la organización como un todo o enfocarse en las unidades organizativas dirigidas hacia un sector de negocios en particular [10].

El concepto de estrategia ha evolucionado a lo largo del tiempo, sin embargo aún no se ha establecido una definición de aceptación general. Existen variados puntos de vista que han generado propuestas de distinta naturaleza, todo ello derivado de que hay muchas formas de comprender qué es la estrategia, cómo desarrollarla y cómo aplicarla [8][15][16]. Siendo el concepto de estrategia fundamental para el análisis estratégico de la organización, aun cuando no hay un concepto derivado del consenso general, en este trabajo se va a adoptar el presentado en [8], dado que refleja las características básicas de los asuntos estratégicos:

Estrategia es la dirección y el alcance de una organización a largo plazo que permite lograr una ventaja en un entorno cambiante mediante la configuración de sus recursos y competencias, con el fin de satisfacer las expectativas de las partes interesadas.

Esta definición establece como pre-condición para que una organización formule una estrategia bien articulada el conocimiento suficiente y apropiado de: (1) la finalidad de la misma, (2) el ambiente competitivo externo, y (3) sus capacidades internas. Este conocimiento es esencial para determinar la dirección de la organización, es decir, hacia dónde enfocar sus actividades y definir el alcance de éstas, el cual establece los límites de la organización tomando en

consideración temas fundamentales como: la tecnología, la gama de productos, la cobertura geográfica, entre otros [8].

Antes de diseñar la estrategia en una organización, se deben ejecutar dos pasos esenciales: (1) concretar una visión del futuro de la organización, y (2) convertir esa visión en una misión que defina la finalidad o razón de ser de la organización [10]. La visión o intención estratégica es el estado futuro para la organización, resume los valores esenciales y aspiraciones de la misma en términos muy genéricos [8]. La misión es la finalidad que explica la existencia de la organización, está expresada en términos del ámbito de sus productos, mercados y áreas geográficas, así como del modo de lograr el desarrollo de las competencias distintivas necesarias para asegurar una ventaja competitiva sostenible [10][17]. La declaración de la misión también puede incluir los valores esenciales o filosofía de la organización. Cabe mencionar que es una práctica común que en los planes estratégicos la visión y la misión estén acompañadas de una declaración explícita de los valores esenciales, en donde se indican los principios fundamentales o doctrinas que deben seguirse en forma permanente en la organización [8][10][17].

De las declaraciones de visión y misión se derivan los objetivos organizacionales, los cuales proporcionan directrices para las decisiones y definen los logros o resultados específicos hacia los cuales la organización debe orientar su esfuerzo [8].

La estrategia define cómo se van a alcanzar los objetivos organizacionales de manera de lograr una ventaja competitiva que asegure la supervivencia de la organización. La visión y la misión sustentan el desarrollo de la estrategia [10][17][18]. La estrategia encamina la organización, por medio de decisiones consistentes, hacia el cumplimiento de lo planteado en la visión y en la misión. Debe ser comunicada e internalizada por cada miembro de la organización y servir de guía en la toma de decisiones difíciles [19]. Para que una estrategia tenga éxito se debe alinear la organización a ella, esto significa que las metas y objetivos, iniciativas estratégicas y procesos de negocios vinculados a las diferentes áreas funcionales y de soporte deben integrarse al curso de acción propuesto en la estrategia global de la organización. Esto se conoce como alineación estratégica [20][21]. La alineación estratégica de un elemento organizacional implica establecer cómo debe contribuir a la proposición de valor de la organización [22][23]. Esto aplica también a las soluciones de software, las cuales deben direccionarse hacia el logro de objetivos organizacionales específicos. Desde las etapas tempranas del ciclo de vida del software, el producto debe estar orientado hacia la contribución efectiva en la generación de valor [9].

Este trabajo plantea integrar en el modelado de metas conceptos relativos al análisis estratégico con el propósito de contribuir a la alineación estratégica del software que se desarrolla. A los conceptos básicos, que soportan la dirección estratégica: visión, misión, objetivos y estrategia, se adicionan otros conceptos relevantes en el desarrollo de esta investigación. La Tabla I presenta los conceptos utilizados en la propuesta descrita en la Sección IV.

Tabla I: Conceptos para el Análisis Estratégico de una Organización

Nombre	Definición
Visión	Define el estado futuro de una organización, resume los valores y aspiraciones de la misma en términos muy

Nombre	Definición
	genéricos [8].
Misión	Define la actividad de la organización de una manera muy genérica [24]. Es la finalidad que explica la existencia de la organización [17].
Meta	Fin general que debe ser alcanzado y es de naturaleza cualitativa [8].
Objetivo	Meta concreta de desempeño que indica hacia donde se dirige la organización [8]. Los objetivos aportan fundamentos sólidos para la formulación de una estrategia, para la ejecución de la misma y para el planeamiento de la acción [19].
Estrategia	Dirección y el alcance de una organización a largo plazo que permite lograr una ventaja en un entorno cambiante mediante la configuración de sus recursos y competencias, con el fin de satisfacer las expectativas de las partes interesadas [8].
Actor	Parte interesada (Stakeholder) activamente involucrada en una organización, sistema o asunto particular [7].
Unidad Estratégica de Negocio (UEN)	Parte de la organización para la que existe un mercado externo diferenciado de bienes y servicios que es distinto de otra unidad estratégica de negocio [8].
Curso de Acción	Representa lo que la empresa decide hacer. Hace referencia a una forma de configurar algunos aspectos de una empresa (cosas, lugares de trabajo, procesos, personas, tiempo) para canalizar los esfuerzos hacia los resultados deseados [24].
Táctica	Es un plan o conjunto de acciones dirigidas al logro de algún propósito. Las tácticas se utilizan para implementar las estrategias, comparadas con estas últimas su alcance es menor y el plazo de ejecución es más corto [24].
Iniciativa	Grupo de programas y proyectos de duración determinada que se ejecutan en paralelo a las actividades operacionales y están diseñadas para ayudar a la organización a alcanzar el rendimiento establecido [25].
Proceso	Es un conjunto de actividades en una organización con el objetivo de ejecutar un trabajo [26].
Actividad	Unidad de trabajo ejecutada como parte de una iniciativa o proceso [27].
Capacidad Estratégica	Conjunto de recursos y competencias de una organización que permiten tener el rendimiento necesario para sobrevivir y prosperar. Incluye el modo en que se utilizan y organizan esos recursos y competencias [8].
Recurso	Factor productivo o económico requerido para cumplir una actividad o como medio para emprender una iniciativa y lograr un resultado deseado [8].
Competencia	Se refiere a la forma en que una organización dispone sus recursos, estructurando actividades y procesos, para desempeñarse eficazmente [8].
Valor Esencial	Principio que guía las acciones de una organización [8]. Frecuentemente se usa el término <i>Valor Nuclear</i> (del inglés <i>Core Value</i>). Los valores esenciales prescriben el comportamiento deseado, carácter y cultura en una organización [10].

III. EL LENGUAJE DE REQUISITOS ORIENTADO A METAS (GRL, GOAL-ORIENTED REQUIREMENTS LANGUAGE)

A. Visión General

El Lenguaje de Requisitos Orientado a Metas (GRL) es un subconjunto de la Notación para Requisitos del Usuario (URN) utilizado para modelar y analizar requisitos, especialmente requisitos no funcionales y atributos de calidad mediante la elaboración de grafos de metas [11][13][28].

GRL es una notación visual que se origina de dos lenguajes de modelado orientado a metas que han tenido una marcada influencia en la IR:

- I* (se pronuncia i-estrella), el cual es un marco de trabajo para modelar y razonar acerca de ambientes

organizacionales y sus sistemas de información. I* asigna una importancia primaria a los actores sociales, atribuyéndoles intencionalidad, es decir, reconociendo que tienen metas, creencias, habilidades y compromisos [29][30].

- NFR Framework, enfocado en la integración de los requisitos no funcionales en el proceso de desarrollo de software. Estos requisitos condicionan y/o restringen el comportamiento del software y pueden afectar al sistema/software en forma global o solo a un conjunto específico de componentes, servicios y/o funciones [11][13][31].

Una de las ventajas de GRL, respecto a otras notaciones, es su integración con la notación de escenarios [11][13] y su extensibilidad [28][32][33]. Hay cuatro principales categorías de conceptos en GRL: *elementos intencionales*, *actores*, *indicadores* y *enlaces*. En la Figura 1 se muestran los símbolos utilizados para el modelado con GRL.

Los *elementos intencionales* en GRL son: *metas*, *softgoals*, *tareas*, *recursos* y *creencias*. Se denominan intencionales porque se utilizan en los modelos para explicar: (1) por qué determinados comportamientos y aspectos estructurales e informacionales fueron seleccionados para incluirlos en los requisitos del sistema, (2) cuáles alternativas fueron consideradas, (3) qué criterios fueron considerados para evaluar las alternativas, y (4) cuáles fueron las razones para la escogencia de una alternativa respecto a las otras [12][29].

Una *meta* (traducción de *goal*, denominada también *hardgoal*) es una condición o estado que un actor desearía alcanzar. Una *softgoal* es también una meta, es decir, una condición o estado que un actor desearía alcanzar, pero a diferencia del concepto previo de *meta* (*goal/hardgoal*) no existen criterios exactos para indicar si la condición es alcanzada, por lo que la evaluación de su grado de satisfacción está influenciada por la subjetividad del juicio de quien determina el logro de la *softgoal* establecida. Una *tarea* (*task*) señala la forma de hacer alguna cosa. Un *recurso* (*resource*) es una entidad física o informacional sobre la que el principal interés es conocer si está disponible. Una *creencia* (*belief*) es utilizada para justificar una decisión de diseño, sirve para reflejar en los modelos las propiedades del dominio [13][34].

Los *actores* (*actors*) son los portadores de las intenciones, son entidades activas en el sistema o su ambiente (por ejemplo, partes interesadas u otros sistemas), son quienes tienen *metas* que alcanzar, *tareas* que ejecutar, *recursos* que disponer y *softgoals* que ser satisfechas [29].

Los *indicadores* (*indicators*) son mediciones del mundo real que apoyan el razonamiento en los modelos de metas permitiendo una evaluación más precisa de la satisfacción de los *actores* [33].

Los *enlaces* (*links*) son utilizados para conectar elementos en un grafo de metas. Hay diferentes tipos de enlaces que describen diferentes relaciones intencionales y estructurales (comprenden *contribuciones*, *dependencias* y *descomposiciones*). Una *contribución* (*contribution*) describe cómo un elemento intencional o un indicador contribuye a la satisfacción de otro elemento intencional objetivo. Una *contribución* se representa como una *correlación* cuando se

refiere a efectos colaterales y no a impactos directos. Una *dependencia* (dependency) describe cómo un actor (denominado *dependier*) depende de otro actor (denominado *dependee*), el objeto de la dependencia (denominado *dependum*) es un *elemento intencional* o un *indicador*. Una *descomposición* (decomposition) permite desagregar un *elemento intencional* en sub-elementos. Los *tipos de descomposición* (decomposition types) AND (Y), IOR (O Inclusivo) y XOR (O Exclusivo) son soportados. Las descomposiciones IOR y XOR pueden mostrarse, alternativamente, como enlaces medios-fin, manteniendo el significado [12][13].

GRL soporta el razonamiento sobre el manejo de conflictos entre metas y la selección entre alternativas de diseño considerando el nivel de satisfacción alcanzado en las metas

(goals o softgoals) mediante las *estrategias GRL*. Las *estrategias GRL* parten de niveles de satisfacción iniciales (asignados a *elementos intencionales e indicadores*), los cuales se propagan por medio de los enlaces [13].

La Figura 2 muestra un modelo GRL simplificado en el que se ilustra el uso de los constructos de esta notación. El modelo representa la interacción entre los actores Aprendiz y Academia, relacionados con el análisis de metas efectuado para una academia de idiomas modernos que se ha planteado la opción de proporcionar servicios educativos en línea. En el modelo GRL se identifican las metas principales de cada actor y se estudia cómo podrían alcanzarse. Para elaborar este modelo se utilizó jUCMNav, herramienta de código abierto para editar y analizar modelos URN desarrollada en la Universidad de Ottawa, Canadá [35].

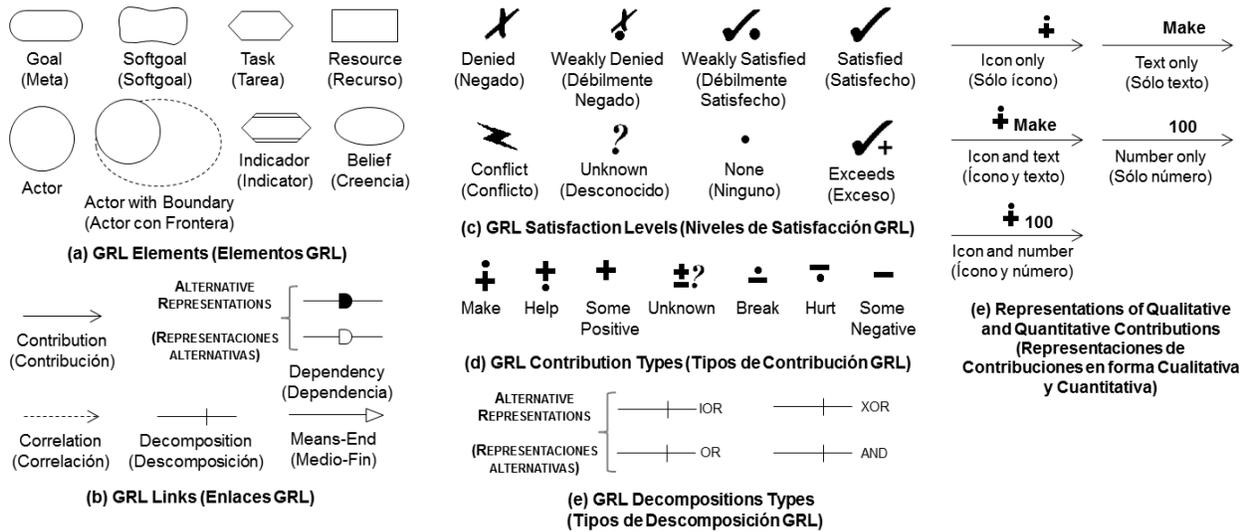


Figura 1: Símbolos de GRL [13]

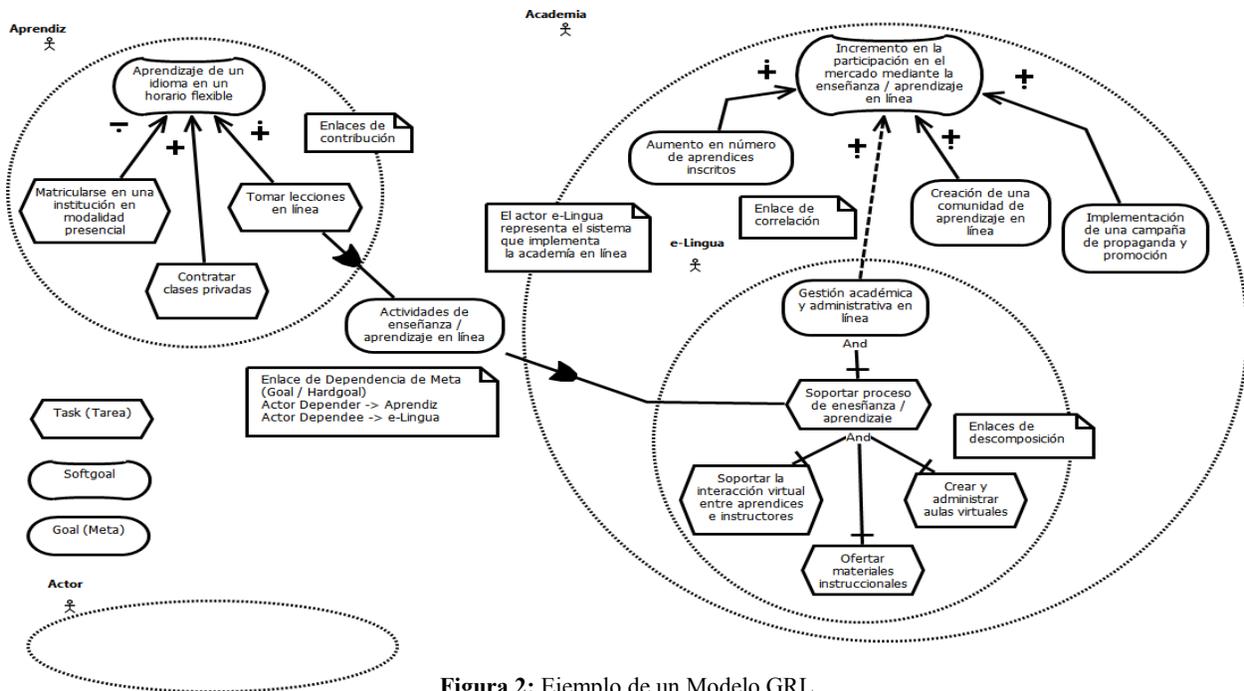


Figura 2: Ejemplo de un Modelo GRL

B. Metamodelo de GRL

La Figura 3 muestra la representación gráfica del metamodelo de los conceptos básicos de GRL, el cual constituye una parte del metamodelo de URN descrito en la Recomendación IUT-T Z.151 [13]. Estos conceptos representan la gramática abstracta del lenguaje, sin tomar en cuenta la notación. En este metamodelo se formalizan los conceptos y constructos descritos previamente. Los términos son utilizados en inglés, con el fin de asegurar el mantenimiento de la semántica original, evitando posibles ambigüedades derivadas de las traducciones.

Además de los conceptos introducidos con anterioridad, el metamodelo incluye otros componentes estructurales, los cuales se describen brevemente a continuación [13]:

- *GRLSpec*: sirve como contenedor para los elementos de la especificación GRL.

- *GRLModelElement*: es un elemento del modelo URN especializado para conceptos GRL. Cada instancia de *GRLModelElement* es única, puede contener metadatos y enlazarse a otros elementos de un modelo URN por medio de *URNLinks* (enlaces URN).
- *GRLLinkableElement*: es un elemento del modelo GRL que puede ser enlazado a otro elemento enlazable de GRL (*GRLLinkableElement*) a través de un *ElementLink* (enlace de elemento). Tiene las características comunes entre las definiciones de actores, elementos intencionales e indicadores.
- *GRLContainableElement*: es un elemento del modelo GRL que puede ser contenido en un actor.
- *ElementLink*: se utiliza para conectar dos elementos enlazables de GRL (*GRLLinkableElement*) y representa la relación intencional entre ellos.

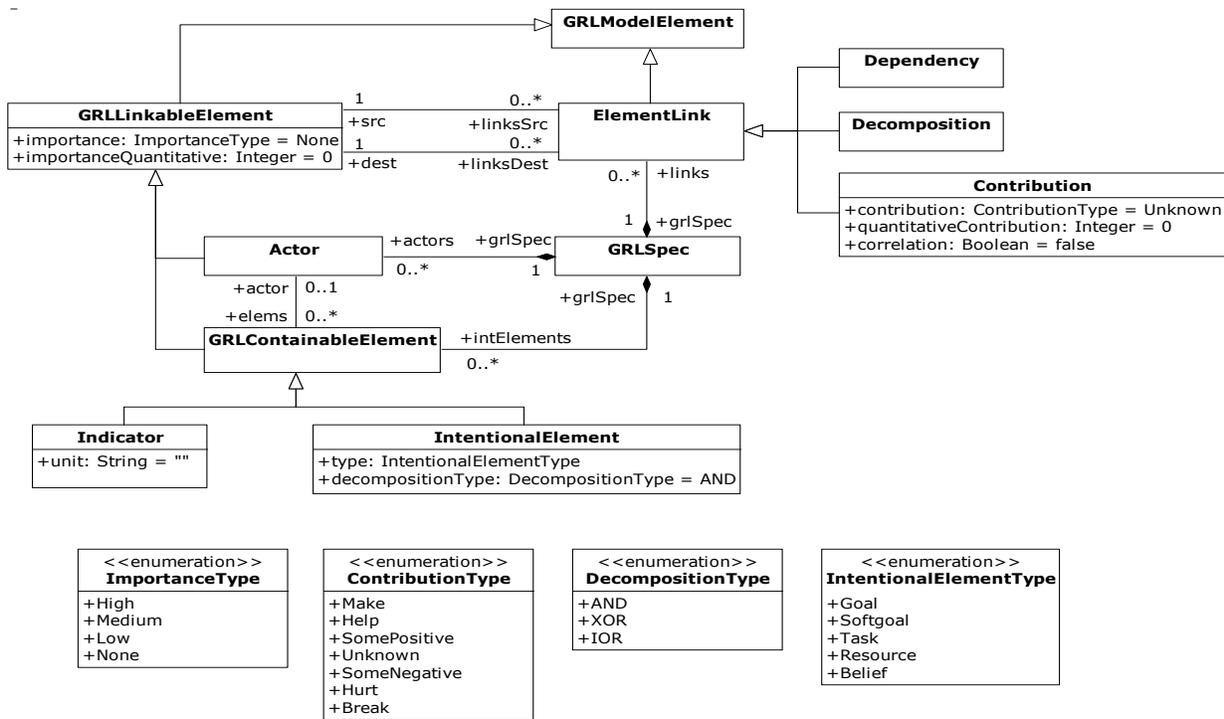


Figura 3: Metamodelo de GRL - Gramática Abstracta [13]

IV. EXTENSIÓN DEL METAMODELO DE GRL PARA MODELAR EL ANÁLISIS ESTRATÉGICO DE UNA ORGANIZACIÓN

La propuesta de extensión del metamodelo de GRL, GRL-SA (del inglés, GRL focused on Strategic Alignment), consiste fundamentalmente en: (1) incluir conceptos relacionados con el análisis estratégico de la organización, específicamente los que definen su dirección estratégica: la visión, la misión, los objetivos organizacionales y la estrategia; y (2) asociar las metas expresadas en diferentes niveles de abstracción con los niveles organizacionales de dirección y gestión.

Antes de describir en detalle el alcance de GRL-SA, es necesario destacar que los términos meta y objetivo usados en el contexto del análisis estratégico (ver definiciones en la Tabla I) serán modelados en forma indistinta, considerando que un objetivo (propósito concreto) es una sub-meta que deriva del

refinamiento de una meta (propósito general). Esto implica que ambos conceptos se representaran por medio de los elementos intencionales *meta (goal)* y *softgoal*, estableciendo el uso de un constructo u otro en base al grado de precisión con que se pueda medir su logro, tal como se plantea en GRL.

El modelado con GRL-SA está enfocado en apoyar la integración de la planificación estratégica y la ejecución operacional. Su propósito es contribuir a la alineación de los procesos organizacionales con las necesidades del negocio. Los aportes de GRL-SA están centrados en: (1) vincular las metas estratégicas con las metas operativas mediante el análisis de contribución entre metas y la descomposición de metas; (2) modelar las estrategias representando: las metas que persiguen (utilizando los elementos intencionales *metas* y *softgoals*), las acciones fundamentales que definen su alcance (haciendo uso

del elemento intencional tarea) y los medios clave para su éxito (a través de los elementos intencionales tareas y recursos); y (3) apoyar la selección de alternativas estratégicas utilizando los mecanismos que ofrece GRL, como los algoritmos de evaluación de modelos y el etiquetado [13].

GRL-SA puede ser aplicado tanto para el modelado estratégico de alto nivel, en el cual se establecen las directrices del funcionamiento de la organización, como en el modelado de asuntos más específicos, tal es el caso de los proyectos de desarrollo de software, en donde se requiere derivar de las metas del producto decisiones técnicas relacionadas con su operacionalización.

La inclusión de constructos en el metamodelo de GRL para el modelado de la visión, misión y cursos de acción como elementos del análisis estratégico (Figura 4) permite relacionar lo establecido en la dirección estratégica de una organización con las metas de un sistema/software en desarrollo, de esta manera, se brinda soporte a la alineación software-estrategia. Utilizando los enlaces establecidos en GRL se puede representar en los grafos de metas la forma en que una solución de software ayuda a implementar una determinada estrategia y cuál es su aporte en el logro de metas organizacionales específicas.

En GRL-SA se plantea representar la visión organizacional como una *softgoal*. Tal como se indicó cuando se describieron los conceptos básicos de GRL, el concepto de *softgoal* se refiere a metas cuyo logro no puede medirse de forma taxativa, es decir, no se puede establecer con precisión cuánto se ha alcanzado de la propiedad o condición deseada. Ambos conceptos, visión y *softgoal*, coinciden en que hacen referencia a estados o condiciones que se desean alcanzar en el futuro cuyo logro no se puede medir con exactitud; la visión puede ser vista como una meta deseada, de difícil modelación y realización, siendo su finalidad servir de inspiración para enfocar el esfuerzo de la organización [8][10][17][18].

Para reforzar la proposición de modelar la visión organizacional como una *softgoal* se debe considerar lo planteado en el Modelo de Motivación del Negocio o Business Motivation Model – BMM [24], especificación orientada a soportar la captura de los requisitos del negocio, en la cual se define la visión como un tipo de fin que describe una imagen global de lo que la organización quiere llegar a ser. El término fin permite asociar la visión con una condición o estado que desea alcanzar y la referencia a imagen global de una organización indica la gran amplitud o alcance del concepto, lo cual pone de manifiesto la dificultad inherente a la medición precisa de su logro.

La visión puede formularse para: (1) una corporación, cuando se hace referencia a un grupo empresarial, o bien, para una organización que abarca diversos sectores de negocio; (2) una unidad estratégica de negocio (UEN), cuando se trata de un área de negocio en particular, o bien, de una única empresa enfocada en un sector de negocios determinado; o (3) un área funcional, al aludir a una unidad organizativa interna con funciones específicas, por ejemplo: Recursos Humanos, Tecnología de la Información, Finanzas, Manufactura, entre otras.

En el metamodelo de GRL-SA (Figura 4), la visión, modelada por la clase *Vision*, se muestra como uno de los tipos de

elementos de análisis estratégico (clase *StrategicAnalysisElement*) y se indica que se representa como una *softgoal* mediante la asociación *isRepresentedAs* entre las clases *Vision* y *Softgoal*. Por medio de la enumeración *StrategicAnalysisLevelType* se especifica el alcance de la visión que se modela: Corporativo (*Corporate*), Unidad Estratégica de Negocio o Negocio (*Business*) y Área Funcional (*Functional*).

En GRL-SA se propone modelar la misión como una *tarea* (*task*). El estándar Z.151 [13] asocia el constructo *tarea* con la forma particular de hacer algo. La misión define la actividad de la organización de una manera muy genérica. La misión es una *tarea* muy compleja cuya ejecución siempre se verá como una aproximación a lo propuesto. En forma análoga a la visión, existen tres niveles en los que se puede definir la misión: Corporativo (*Corporate*), Unidad Estratégica de Negocio o Negocio (*Business*) y Área Funcional (*Functional*).

El metamodelo de GRL-SA (Figura 4) indica que la clase *Mission* (la cual se va a emplear para modelar la misión organizacional) es una especialización de la clase *StrategicAnalysisElement*. Las clases *Mission* y *Task* se relacionan mediante la asociación *isRepresentedAs*. La clase *Mission* contiene el atributo *analysisLevel* que indica el nivel al cual se define la misión, los valores que puede tomar este atributo están contenidos en la enumeración *StrategicAnalysisLevelType*. La misión hace operativa la visión, es decir, indica qué actividades se deben ejecutar para que la visión se haga realidad [24].

Los cursos de acción, los cuales definen los planes para alcanzar algún propósito, se representan con la clase *CourseOfAction*. Esta clase también es una especialización de la clase *StrategicAnalysisElement*. Agrupa las estrategias (clase *Strategy*), tácticas (clase *Tactic*) e iniciativas (clase *Initiative*). Las definiciones de táctica e iniciativa están en la Tabla I. Estos elementos utilizados en la planificación estratégica son los que permiten la especificación de las acciones a ejecutar para el logro de determinadas metas. Por analogía entre las definiciones, GRL-SA plantea representar los cursos de acción como tareas.

En el metamodelo de GRL se hace uso de la enumeración *IntentionalElementType* para definir los tipos de elementos intencionales (Figura 3). GRL-SA plantea la representación de los elementos intencionales como especializaciones para ampliar la semántica mediante: (1) el establecimiento de relaciones entre éstos y los elementos del análisis estratégico y (2) la adición de atributos. En el caso de los elementos intencionales *softgoals* y *metas* (*goals*) se agrega el atributo *levelType*, cuya función es indicar el nivel de abstracción de la meta, el cual direcciona el razonamiento relacionado con la alineación estratégica dado que permite orientar la descomposición de una meta en sub-metas y enfocar el análisis para establecer la contribución a metas de un nivel de abstracción superior.

En [14] se identificaron cinco niveles de abstracción para las metas (Figura 5), estos son: *Corporativo*, *Negocio*, *Operacional*, *Sistema* y *Software*.

El nivel *Corporativo* se refiere a las metas relacionadas con la gestión global de una organización y/o las áreas de negocio en las que se participa o se desea participar. Las metas

corporativas pueden estar vinculadas a las expectativas de los propietarios, los accionistas y la Bolsa [8]. Asimismo también abordan tópicos asociados a la gestión de la cartera de inversiones o negocios. El nivel de *Negocio* incluye las metas relativas al aprovechamiento de las capacidades estratégicas (Tabla I) y a la forma en que compite o se espera competir en áreas específicas. El nivel *Operacional* comprende las metas vinculadas a la gestión de la organización, pueden estar

asociadas al diseño de los procesos y/o a la definición de roles de los actores organizacionales. Las metas del nivel *Sistema* establecen el comportamiento del sistema y sus interacciones. En el nivel de *Software*, las metas especifican el funcionamiento del producto de software y la forma en que se espera ejecute los servicios requeridos.

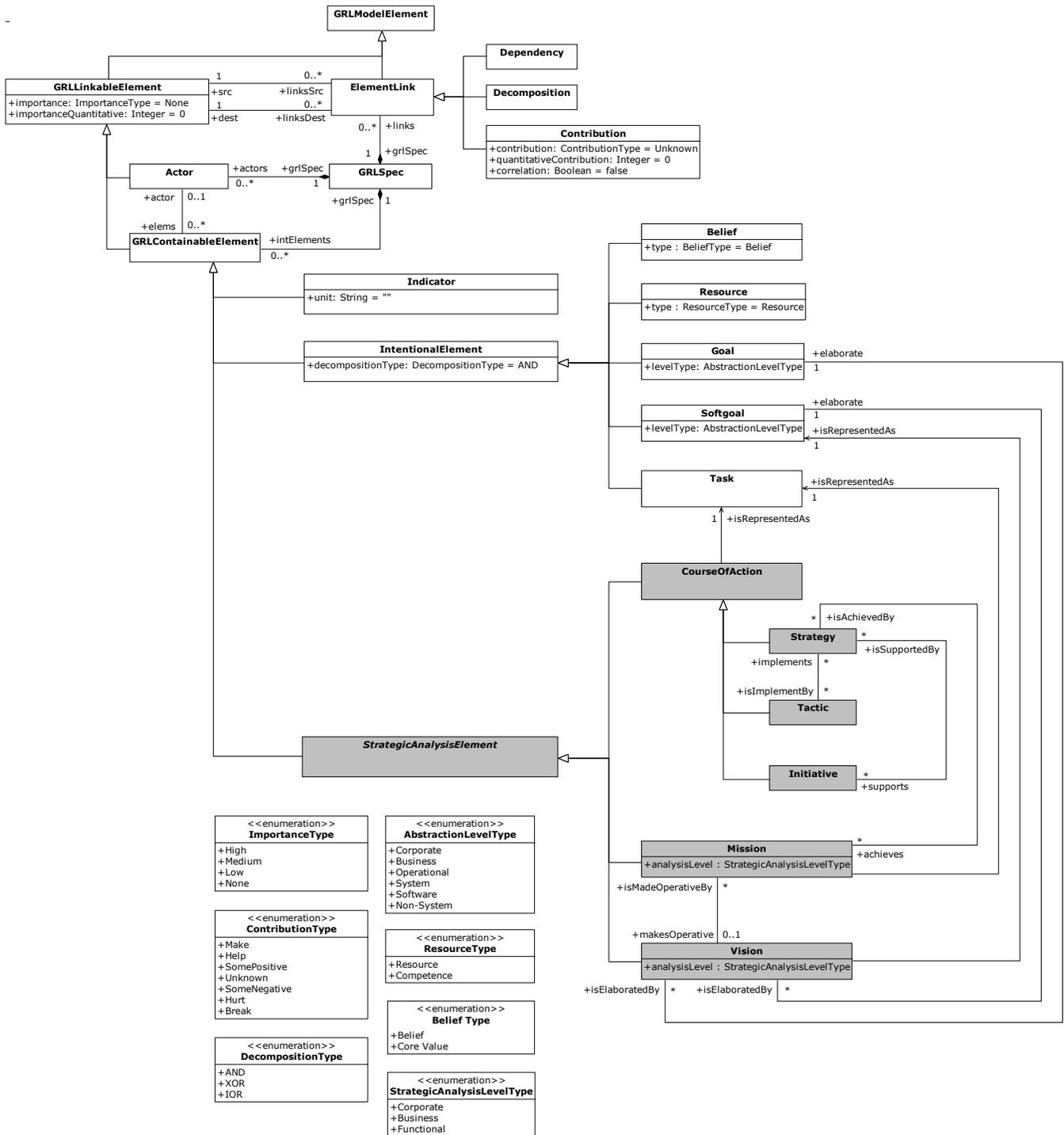


Figura 4: Metamodelo de GRL-SA

Los niveles *Corporativo*, *Negocio* y *Operacional* están enfocados en la dirección y gestión de la organización. Del

refinamiento de las metas ubicadas en estos niveles de abstracción se derivan no sólo metas dirigidas hacia un sistema

basado en software, sino otras metas organizacionales relacionadas a funciones y/o procesos que no están vinculados al sistema en desarrollo. La enumeración *AbstractionLevelType* contiene los tipos de metas según su nivel de abstracción. Cuando la meta deriva del nivel *Operacional* pero no corresponde a un sistema basado en software, se utiliza el literal *Non_System*, sin embargo, esto no limita el tratamiento dado a la meta y puede seguirse refinando hasta obtener la concreción requerida en el análisis que se lleva a cabo.

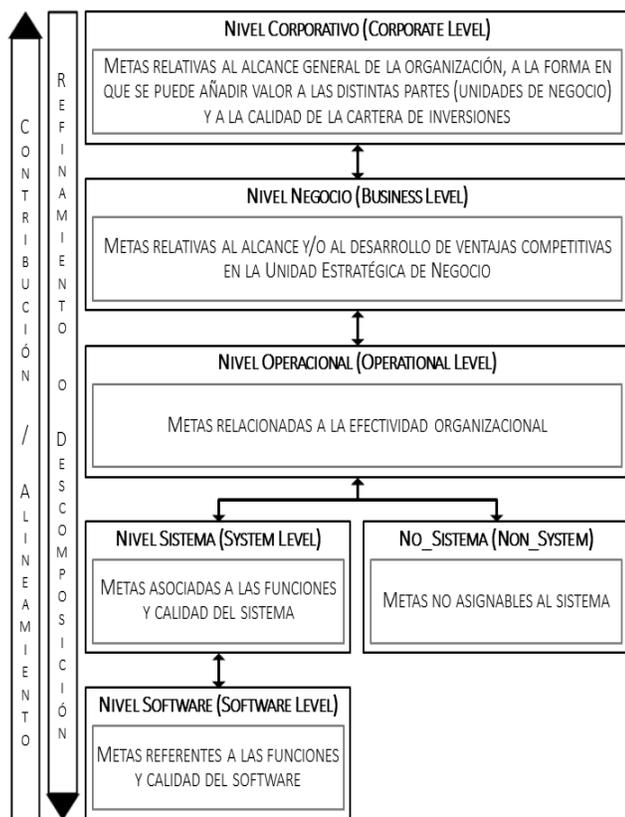


Figura 5: Niveles de Abstracción

Las estrategias que se elaboran son soportadas no solo por la visión, la misión y los objetivos organizacionales sino también por los valores esenciales (*core values*) que guían el comportamiento de la organización (Tabla I). En tal sentido, GRL-SA plantea la representación de los valores esenciales mediante el constructo *creencia* (*belief*). Una *creencia* en GRL permite modelar argumentos que justifiquen decisiones de diseño (tal como propiedades del dominio) para que en el modelo queden reflejadas las razones que apoyan las decisiones y facilitar la trazabilidad [13]. En el contexto del análisis estratégico, los valores esenciales influyen en la selección de alternativas para definir estrategias y respaldan lo establecido en la visión, la misión y los objetivos organizacionales. En la enumeración *BeliefType* se definen los literales *Belief* y *CoreValue*, para especificar el uso del constructo.

En GRL-SA se amplía el concepto del elemento intencional *recurso* (*resource*). Este constructo se podrá utilizar no sólo para modelar los recursos que puedan requerirse para satisfacer una *meta* (del tipo *goal* o del tipo *softgoal*) o la ejecución de una *tarea* (*task*), sino también las competencias (Tabla I). En el

análisis estratégico las competencias y los recursos conforman la capacidad estratégica (Tabla I) de la organización, el cual es un elemento fundamental cuando se trata de discurrir sobre cómo lograr una ventaja competitiva, factor crítico en el diseño de las estrategias. En la enumeración *ResourceType* se definen los literales: *Resource* y *Competence*, de utilidad para representar los recursos y competencias significativos en el análisis que se desarrolle.

La notación de GRL-SA se basa en la de GRL (Figura 1) e incorpora identificadores para los elementos del análisis estratégico y los elementos intencionales: *softgoal*, *meta* (*goal*) y *recurso* (*resource*), con el propósito de aportar más precisión en su representación cuando sean modelados. En la Figura 6 se muestra la notación de GRL-SA.

V. APLICACIÓN DE GRL-SA

En esta sección se ilustra el modelado basado en GRL-SA. El caso de estudio seleccionado se refiere al análisis estratégico de una institución educativa enfocada en la enseñanza de Idiomas Modernos, *Lingua For Everybody* (*LiFE*). Esta organización se propone un cambio significativo en sus operaciones incorporando la Tecnología de la Información para soportar las actividades de enseñanza y aprendizaje, y como herramienta estratégica para aumentar su participación en el mercado.

La Figura 7 exhibe un modelo de metas basado en GRL-SA. Este modelo muestra los elementos básicos de la dirección estratégica de *LiFE*. La institución concentra sus operaciones en la enseñanza de Idiomas Modernos, por lo tanto para la visión y la misión se indica en los identificadores que el nivel del análisis estratégico es *Business* (*Negocio*), correspondiente a una UEN o a una organización enfocada a un único sector de negocios (ver (a) y (b) en la Figura 7).

La visión (a) se modela utilizando el símbolo de *softgoal* acompañado del identificador apropiado, en este caso el identificador es *Business Vision*. La visión de la institución sólo puede representarse una vez. La misión (b) se modela como una *tarea* (*task*) y se le asigna el identificador correspondiente al nivel del análisis estratégico, el cual debe ser el mismo que se asignó al representar la visión. La misión se enlaza con la visión mediante una contribución (c) del tipo *Make* [13][29] lo que significa que el cumplimiento de la misión conduce a la satisfacción de la visión.

Los valores esenciales (l) y (m) de la organización son modelados como *creencias* (*beliefs*).

Al analizar la misión se deriva un conjunto de metas organizacionales de alto nivel de abstracción (nivel *Negocio*), metas estratégicas, representadas como *softgoals* – (d), (e), (f) y (g) - cuya satisfacción debe encauzar a la institución hacia el cumplimiento de su visión. A estas *softgoals* se les asigna el identificador *Business*. En el metamodelo de GRL-SA (Figura 4) se indica que las *metas* (*goals*) y las *softgoals* elaboran la visión, es decir, expresan con mayor detalle y concreción hacia dónde dirigir el esfuerzo de la organización. Las *softgoals* definidas en el nivel *Negocio* están vinculadas a la misión mediante enlaces de descomposición tipo *AND* (h), lo que indica que la misión se ejecutará a través de la satisfacción del conjunto de *softgoals* declarado.

Para establecer cómo será satisfecha una *softgoal* en particular (la cual representa un objetivo organizacional), se definen

estrategias que fijan la dirección a seguir, por ejemplo, en (i). Estas estrategias se modelan como *tareas (task)* puesto que representan acciones, aunque estén expresadas en un alto nivel de abstracción. A las estrategias se les asigna el identificador *Strategy*.

Durante el proceso de análisis estratégico, las estrategias deben ser especificadas con mayor nivel de detalle. Dependiendo de la complejidad de una estrategia ésta puede desglosarse en tácticas para facilitar su implementación. En el análisis de una

estrategia pueden modelarse metas organizacionales, las cuales representan hitos que permitirán evaluar el éxito de la estrategia. Por ejemplo, para que la estrategia “Implementar un modelo de funcionamiento de la academia en línea” (i) se considere exitosa se debe alcanzar la *softgoal* (j) “Incremento en la participación en el mercado mediante la enseñanza / aprendizaje en línea” y la *meta (goal)* (k) “Gestión académica y administrativa en línea”. Ambas son ubicadas en el nivel de abstracción *Operacional (Operational)*.

Elemento de Análisis Estratégico	Representación	Identificador	Ejemplo
Visión (Vision)		<ul style="list-style-type: none"> Corporate_Vision (Visión Corporativa) Business_Vision (Visión de UEN o Negocio) Functional_Vision (Visión de Área Funcional) 	
Misión (Mission)		<ul style="list-style-type: none"> Corporate_Mission (Misión Corporativa) Business_Mission (Misión de UEN o Negocio) Functional_Mission (Misión de Área Funcional) 	
Curso de Acción (Course of Action)		<ul style="list-style-type: none"> Strategy (Estrategia) Tactic (Táctica) Initiative (Iniciativa) 	

(a) Notación para Representar los Elementos del Análisis Estratégico en GRL-SA

Elemento Intencional	Identificador	Ejemplos
Softgoal 	<ul style="list-style-type: none"> Corporate (Nivel Corporativo) Business (Nivel de Negocio) Operational (Nivel Operacional) System (Nivel de Sistema) Software (Nivel de Software) Non-System (No asignable al sistema) ² 	
Goal (Meta) 	<ul style="list-style-type: none"> Corporate (Nivel Corporativo) Business (Nivel de Negocio) Operational (Nivel Operacional) System (Nivel de Sistema) Software (Nivel de Software) Non-System (No asignable al sistema) ² 	
Resource (Recurso) 	<ul style="list-style-type: none"> Resource (Recurso) ¹ Competence (Competencia) 	
Belief (Creencia) 	<ul style="list-style-type: none"> Belief (Creencia) ¹ CoreValue (Valor Esencial) 	

¹ EL USO DEL IDENTIFICADOR ES OPCIONAL.
² NO APLICA EL USO DEL IDENTIFICADOR.

(b) Notación para Representar los Elementos Intencionales en GRL-SA

Figura 6: Notación GRL-SA

El nivel de abstracción asociado a una meta organizacional permite focalizar el modelado y sirve para guiar su descomposición; cada nivel de abstracción tiene un enfoque específico (Figura 5) lo que debe tomarse en consideración al especificar la manera en que una meta será refinada. Cabe destacar que en GRL la descomposición de metas no solo consiste en generar un sub-conjunto de metas de menor complejidad, el enfoque principal es esbozar la forma en que se

espera lograr alguna meta en particular mediante un esquema o rutina (término propuesto en [29]) basado en la combinación de varios elementos intencionales GRL (*softgoals*, *metas*, *tareas*, *recursos*, *creencias*) interrelacionados por medio de enlaces GRL.

La *softgoal* (j) “Incremento en la participación en el mercado mediante la enseñanza / aprendizaje en línea” es refinada en el

modelo mostrado en la Figura 8. Este modelo se construyó utilizando la herramienta jUCMNav [35] considerando la extensión planteada en GRL-SA. Es importante notar que en la descomposición de una meta se pueden obtener sub-metas que, por su complejidad o alcance, estén ubicadas en el mismo nivel de abstracción de la meta padre.

En la Figura 8 se observa el modelado de cursos de acción de los tipos: tácticas e iniciativas. Las tácticas sirven para facilitar la implementación de las estrategias y las iniciativas se refieren a proyectos o programas que se ejecutan en paralelo a las operaciones con el fin de reforzar el desempeño de la

organización para el logro de determinado propósito. Para representar las tácticas y las iniciativas se utiliza el constructo *tarea (task)* de GRL.

El modelado con GRL-SA permite clarificar el rol de un sistema/software en el contexto de la estrategia de una organización, facilitando la identificación de las metas organizacionales a las que debe contribuir. Para obtener los requisitos del software [36] es necesario descomponer las metas, por medio de rutinas, hasta llegar a la obtención de tareas atómicas o simples que puedan, en caso de ser pertinente, ser asignadas a un componente de software.

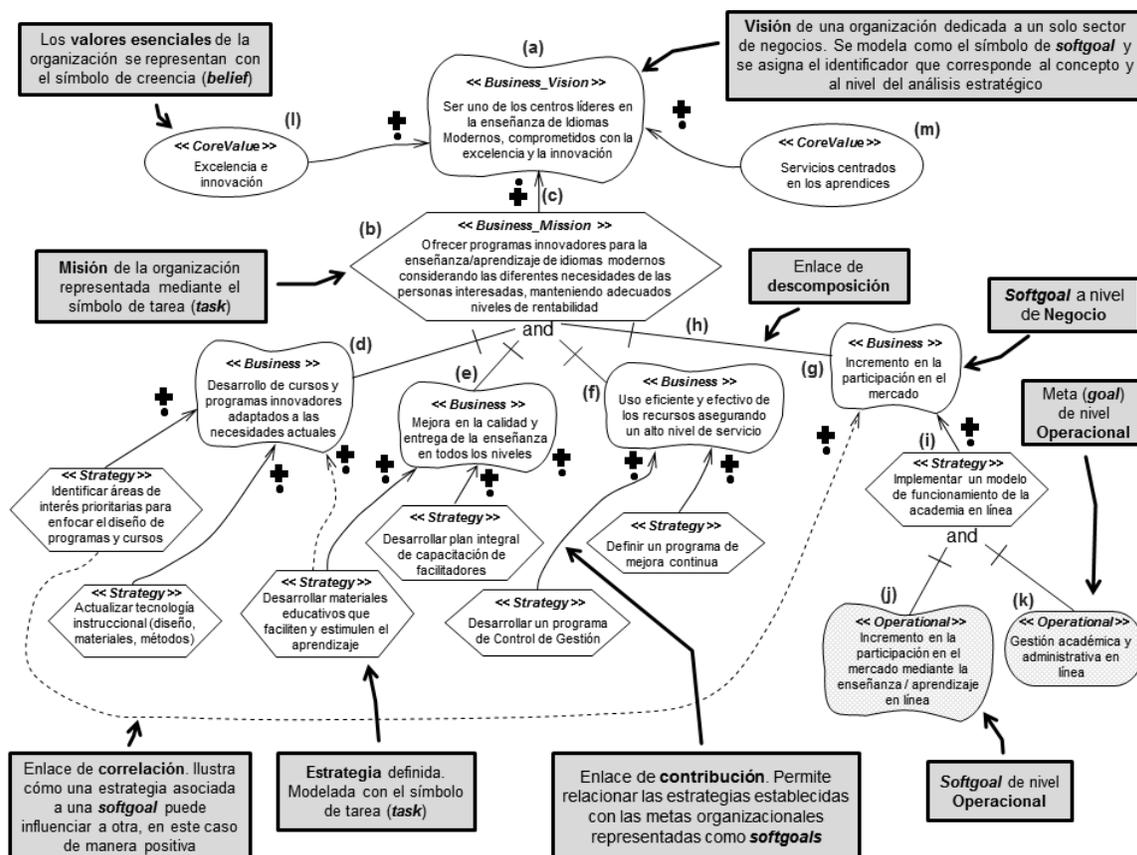


Figura 7: Modelado del Análisis Estratégico de LiFE Utilizando GRL-SA

VI. CONCLUSIONES

Los sistemas de software tienen gran relevancia dentro de las organizaciones. En muchos sectores de negocio resultan vitales para la supervivencia de las empresas. La importancia del papel del software destaca la necesidad de su alineación con la estrategia organizacional. Esta alineación debe asegurarse desde la etapa de IR, que es donde se definen las metas y el alcance del software.

GRL es una notación, utilizada en la IR, que permite el modelado organizacional, sin embargo, es poco frecuente que en los modelos se represente la vinculación entre las metas del software y la dirección estratégica de una organización debido a que, usualmente, los modelos de metas se centran en procesos relacionados a la operaciones y no incorporan los asuntos estratégicos. Al respecto, en este trabajo se propone una extensión para GRL, GRL-SA, en la cual se incluyen conceptos relacionados con el análisis estratégico de una

organización, de esta forma, en los modelos se puede enlazar el software con las metas definidas por la dirección y las estrategias organizacionales. Este enlace sirve para justificar el desarrollo del sistema/software, mejorar la comprensión de su rol como elemento organizacional (no sólo tecnológico) y validar su alineación estratégica.

GRL-SA contempla el manejo del nivel de abstracción de las metas. Este concepto ayuda a definir el nivel de complejidad y el alcance de una meta direccionando el modelado en lo referente a: (1) el refinamiento o descomposición de la meta, (2) el análisis de contribución a metas de un nivel superior, (3) la asignación de la responsabilidad de su cumplimiento, y (4) la evaluación de las capacidades estratégicas (recursos y competencias) requeridas para su satisfacción.

En GRL-SA se manejan cinco niveles de abstracción para una meta: *Corporativo, Negocio, Operacional, Sistema y Software*. Ubicar una meta en determinado nivel implica analizar su

contribución a las metas del nivel superior y guía su descomposición en sub-metas considerando las características del nivel inferior. Lo anterior ofrece un mecanismo para enfocar el análisis hacia la alineación estratégica y proporciona argumentos para justificar la existencia de la meta.

En el caso de estudio descrito se pudo observar como en el modelo de metas se representan los conceptos vinculados a la planificación estratégica de la organización y, progresivamente, se derivan las metas del sistema/software

necesario para apoyar metas organizacionales de mayor nivel de abstracción. Sin embargo, desde la perspectiva del análisis estratégico, GRL-SA sólo permite el modelado de los asuntos internos de la organización, no incluye la representación de los elementos del análisis externo, tema relevante en el proceso de planificación estratégica. Actualmente se está trabajando en la incorporación de sentencias OCL (del inglés, Object Constraint Language) que ayuden a definir reglas que orienten el modelado.

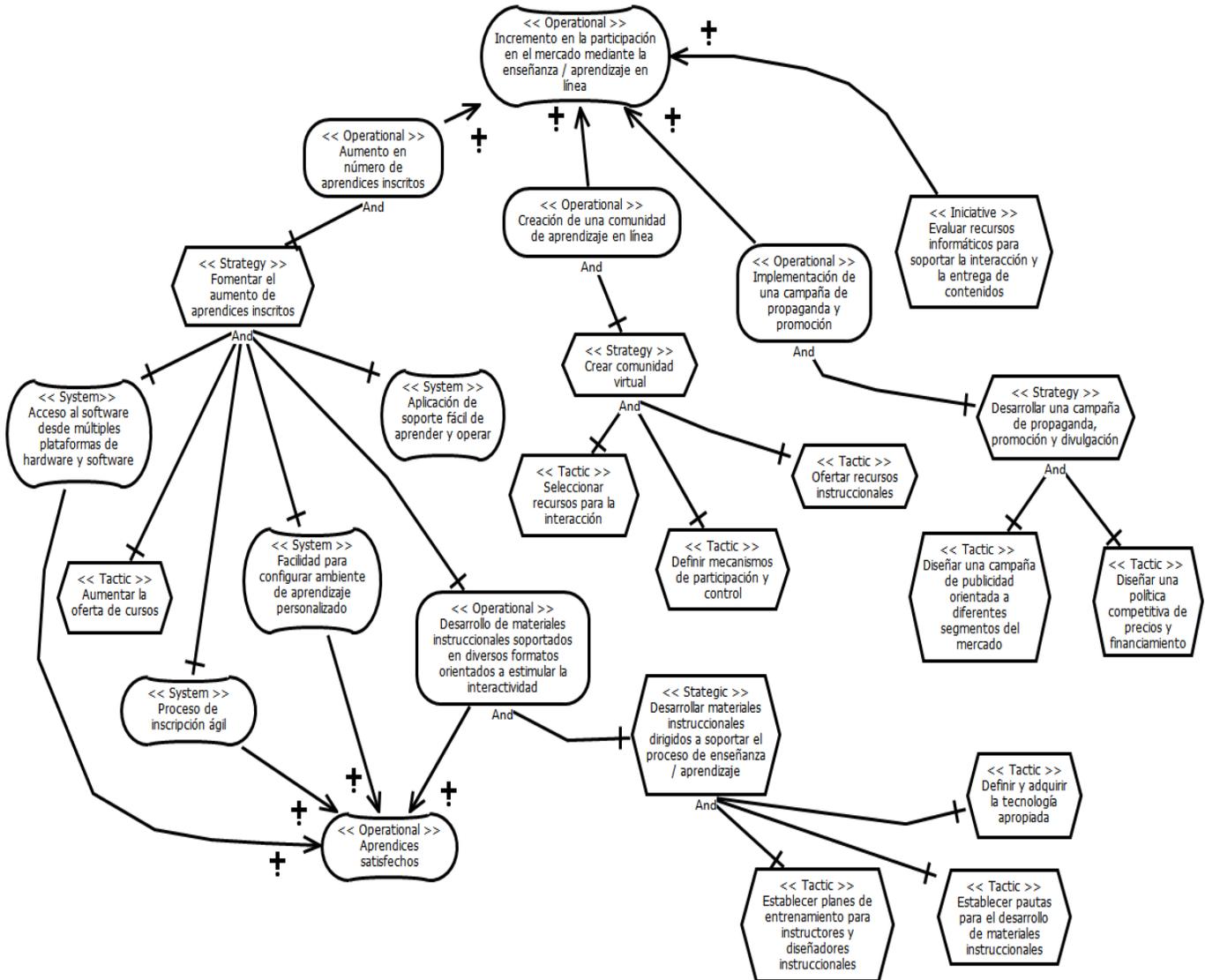


Figura 8: Refinamiento de una Softgoal Utilizando GRL-SA

REFERENCIAS

[1] A. Antón, *Goal-Based Requirements Analysis*, in proceedings of the 2nd International Conference on Requirements Engineering (ICRE 1996), Colorado Spring, USA, April 1996.

[2] S. Baïna, P. Ansias, M. Petit and A. Castiaux, *Strategic Business/IT Alignment using Goal Models*, in proceedings of the Third International Workshop on Business/IT Alignment and Interoperability (BUSITAL 2008), Held in Conjunction with CAISE 2008 Conference, Montpellier, France, June 2008.

[3] A. Dardenne, A. van Lamsweerde and S. Fickas, *Goal-Directed Requirements Acquisition*, Science of Computer Programming, vol. 20, no. 1-2, pp. 3-50, April 1993.

[4] A. van Lamsweerde, *Goal-Oriented Requirements Engineering: A Guided Tour*, in proceedings of the Fifth IEEE International Symposium on Requirements Engineering (ISRE 2001), Toronto, Canada, August 2001.

[5] A. van Lamsweerde, *Requirements Engineering: from Craft to Discipline*, in proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (SIGSOFT 2008), Atlanta, USA, November 2008.

- [6] A. van Lamsweerde, *Requirements Engineering: From System Goals to UML Models to Software Specifications*, 1st edition, John Wiley & Sons, Inc, England, 2009.
- [7] K. Pohl, *Requirements Engineering: Fundamentals, Principles, and Techniques*, 1st edition, Springer, Germany, 2010.
- [8] G. Johnson, K. Scholes and R. Whittington, *Dirección Estratégica*, Seventh edition, Pearson Educación, S. A., Spain, 2006.
- [9] S. Pass and B. Ronen, *Reducing the Software Value Gap*, Communications of the ACM, vol. 57, no. 5, pp. 80-87, May 2014.
- [10] R. Kaplan, D. Norton and E. Barrows, *Developing the Strategy: Vision, Value Gaps, and Analysis*, Balanced Scorecard Report, Harvard Business School Publishing, vol. 10, no. 1, pp. 1-16, January-February 2008.
- [11] D. Amyot, *Introduction to the User Requirements Notation: Learning by Example*, Computer Networks, vol. 42, no. 3, pp. 285-301, June 2003.
- [12] D. Amyot y G. Mussbacher, *User Requirements Notation: The First Ten Years, The Next Ten Years*, Invited paper, Journal of Software (JSW), vol. 6, no. 5, pp. 747-768, May 2011.
- [13] International Telecommunication Union - Telecommunication Standardization Sector (ITU-T), *User Requirements Notation (URN) – Language Definition, Recommendation ITU-T Z.151*, 2012.
- [14] M. Delgado, F. Losavio and A. Matteo, *Goal Oriented Techniques and Methods: Goal Refinement and Levels of Abstraction*, in proceedings of the 39th Latin American Computing Conference (CLEI 2013), Naiguata, Venezuela, October 2013.
- [15] K. Favaro, K. Rangan and E. Hirsh, *Strategy: An Executive's Definition. Strategy + Business*, Summer 2012, no. 67, May 2012.
- [16] H. Mintzberg, *The Strategy Concept I: Five Ps for Strategy*, California Management Review, vol. 20, no. 1, pp. 11-24, Fall 1987.
- [17] J. Lema, *La Guía Estratégica: El Corazón del Plan Estratégico*, Revista EIA, College of Engineering of Antioquia, Medellín, Colombia, vol. 1, no. 2, pp. 9-16, August 2004.
- [18] J. Collins and J. Porras, *Building Your Company's Vision*, Harvard Business Review, vol. 74, no. 5, September-October 1996.
- [19] D. Collis and M. Rukstad, *Can You Say What Your Strategy Is?*, Harvard Business Review, vol. 86, no. 4, pp. 98-107, April 2008.
- [20] J. Gerow, J. Thatcher and V. Grover, *Six Types of IT-Business Strategic Alignment: An Investigation of the Constructs and their Measurement*, European Journal of Information Systems, vol. 24, no. 5, pp. 465-491, 2015.
- [21] M. Rosemann and J. vom Brocke, *The Six Core Elements of Business Process Management*, Handbook on Business Process Management 1, International Handbooks on Information Systems, Springer-Verlag, Berlin, Germany, 2010. DOI: 10.1007/978-3-642-00416-2_5.
- [22] H. Armitagey and C. Scholey, *Using Strategy Maps to Drive Performance*, Management Accounting Guideline, published by The Society of Management Accountants of Canada (CMA-Canada), The American Institute of Certified Public Accountants, Inc. (AICPA) and The Chartered Institute of Management Accountants (CIMA), 2006.
- [23] R. Kaplan and D. Norton, *Strategy Maps. Executive Books Summaries*, vol. 26, no. 4 (2 parts) part 1, April 2004.
- [24] Object Management Group (OMG), *Business Motivation Model*, version 1.3, May 2015, Standard document.
- [25] R. Kaplan and D. Norton, *Integrating Strategy Planning and Operational Execution: A Six-Stage System*, Balanced Scorecard Report, Harvard Business School Publishing, vol. 10, no. 3, May- June 2008.
- [26] Object Management Group (OMG), *Business Process Model and Notation (BPMN)*, version 2.0.2, December 2013, Standard document.
- [27] International Institute of Business Analysis (IIBA), *A Guide to the Business Analysis – Body of Knowledge*, version 2.0, 2009.
- [28] International Telecommunication Union - Telecommunication Standardization Sector (ITU-T), *User Requirements Notation (URN) – Language Requirements and Framework, Recommendation ITU-T Z.150*, 2011.
- [29] E. Yu, *Modelling Strategic Relationships for Process Reengineering*, Ph.D. Dissertation, University of Toronto, Toronto, Canada, 1995.
- [30] E. Yu, *Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering*, in proceedings of the 3rd IEEE International Symposium on Requirements Engineering (ISRE 1997), Annapolis, USA, January 1997.
- [31] L. Chung, *Representation and Utilization of Non-Functional Requirements for Information System Design*, in proceedings of 3rd International Conference in Advanced Information Systems Engineering, (CAISE 1991), Trondheim, Norway, May 1991.
- [32] G. Mussbacher, *Aspect-Oriented User Requirements Notation*, Ph.D. Dissertation, University of Ottawa, Ottawa, Canada, November 2010.
- [33] A. Pourshahid, D. Amyot, L. Peyton, S. Ghanavati, P. Chen, M. Weiss and A. J. Forster, *Business Process Management with the User Requirements Notation*, Electronic Commerce Research, vol. 9, no. 4, pp. 269-316, December 2009.
- [34] L. Liu y E. Yu, *GRL - Goal-Oriented Requirement Language*, Web Site, 2000.
- [35] J. Roy, J. Kealey y D. Amyot, *Towards Integrated Tool Support for the User Requirements Notation*, in proceedings of Fifth Workshop on System Analysis and Modelling, Kaiserslautern, Germany, May 2006.
- [36] Institute of Electrical and Electronics Engineers (IEEE), *Guide to the Software Engineering Body of Knowledge – SWEBOK*. P. Bourque y R.E. Fairley, eds, 2014.

Un Marco de Referencia Metodológico para un Sistema Basado en Conocimiento Usando MDA

Pablo Valera^{1,2}, Dinarle Ortega², María Guevara²
pjvalera@yahoo.com, dinarleortega@gmail.com, mguevara@uc.edu.ve

¹ Departamento de Estudios Básicos, Universidad José Antonio Páez, Valencia, Venezuela

² Departamento de Computación, FACYT, Universidad de Carabobo, Valencia, Venezuela

Resumen: El presente trabajo muestra una revisión bibliográfica de los Sistemas Basados en Conocimiento (SBC), identificando la problemática de la ausencia de un lenguaje de representación estándar para su modelado, así como la trazabilidad clara entre sus modelos conceptuales y de diseño.

Posteriormente, se busca aplicar un enfoque de una Arquitectura Dirigida por Modelos (MDA, en inglés) a una metodología ampliamente conocida para el desarrollo de sistemas basados en conocimiento (COMMONKADS), y de esta manera generar un modelo independiente (*Platform Independent Model*, PIM) y un modelo específico de la plataforma (*Platform Specific Model*, PSM) que permitan la transformación así como la trazabilidad entre el modelo conceptual y de diseño de los SBC, para convertirlos de esta forma en los actores centrales del proceso de desarrollo. En particular, se genera un PSM para Java exhibiendo algunas transformaciones que seguidamente serán aplicadas a un caso de estudio. El PIM y el PSM obtenidos son diagramados en UML buscando unificar o proveer un lenguaje de representación alternativo para el modelado de este tipo de sistemas. Finalmente, se instancian los resultados en un caso de estudio para ser validados en el modelado de un sistema para el pronóstico de cáncer de seno.

Palabras Clave: MDA; SBC; COMMONKADS.

Abstract: This paper presents a literature review of Knowledge Based Systems (KBS), identifying the problem of the lack of a standard language representation for modeling and clear traceability between their conceptual and design models.

Subsequently, it seeks to apply an approach of Model Driven Architecture (MDA) to a widely known methodology for developing knowledge based systems (COMMONKADS), and thus generate a Platform Independent Model (PIM) and a Platform Specific Model (PSM) that allow the transformation and traceability between the conceptual and design models of the KBS, turning them this way in the central actors in the development process. In particular, a PSM is generated for Java exhibiting some transformations that will be applied to a case of study. PIM and PSM obtained are diagrammed in UML seeking to unify or provide an alternative representation language for modeling such systems. Finally, the results are instantiated in a case of study to be validated in the modeling of a breast cancer prediction system.

Keywords: MDA; KBS; COMMONKADS.

I. INTRODUCCIÓN

El quehacer económico y social está resultando cada vez más y más orientado al conocimiento, cobrando relevancia día a día el término, “sociedad de la información”. De acuerdo a Peter Druker [1] “El conocimiento es el único recurso significativo hoy en día”. Es un activo cada vez más apreciado por la sociedad contemporánea porque permite capturar la experiencia adquirida de las personas a través del tiempo, soportando de esta forma la toma de decisiones y facilitando el flujo de trabajo de cualquier empresa.

En particular, en el área de la informática se identifica al conocimiento como un elemento significativo para mejorar el

ejercicio de diversas labores, por medio de su representación y empleo para la sistematización de las tareas. Se procura crear una aplicación mecanizada que almacene, explote y actualice los conocimientos que los especialistas en un campo pueden aportar para la solución de problemas. Para lograr esto, se ha planteado la creación de “Sistemas Basados en Conocimiento” (SBC) correspondientes a un dominio específico, utilizando los aportes de la inteligencia artificial para propiciar el aprendizaje, razonamiento, representación y comunicación de éste conocimiento.

Los SBC, introducidos a principios de los años 70's como sistemas expertos desde el campo de la inteligencia artificial (IA), fueron originalmente diseñados para extraer

conocimiento de un humano experto en el dominio y codificar éste como reglas en una base de conocimiento (BC). No obstante, ha habido un cambio de paradigma en la Ingeniería del Conocimiento (IC), donde el enfoque de “extracción/transferencia” ha sido sustituido por el enfoque de “modelado”. Los modelos son usados para proveer una abstracción o simplificación acerca de la realidad y a través de los mismos, el enfoque del humano experto resolvidor del problema es modelado y usado en el desarrollo del SBC.

En la actualidad las metodologías de la IC están suficientemente maduras y ofrecen un conjunto de actividades y técnicas que abarcan el análisis y el posterior modelado conceptual del sistema a ser desarrollado de una manera adecuada [2]. Entre ellas se destacan COMMONKADS, MIKE, y PROTÉGÉ-II. De igual forma, se ha intentado también adaptar muchas técnicas de Ingeniería de Software (IS) a la IC, para mejorar aún más el desarrollo de los SBC, sin embargo, todavía queda camino por recorrer. En el caso de las metodologías mencionadas, éstas adolecen de una falta de rigor en cuanto a proveer elementos para la trazabilidad del modelado conceptual al diseño del sistema, tal y como lo propone la IS [3][4]. Adicionalmente, se aprecia que en la mayoría de los casos, varios lenguajes de modelado son aplicados en un proyecto. Por consiguiente, se experimenta la limitación de compartir y reusar el diseño de los SBC, por no haber un lenguaje de modelado ampliamente aceptado.

Esta limitación constituye una amenaza para los sistemas del futuro, debido a que serán diseñados para trabajar juntos con otras aplicaciones como parte de los sistemas de las empresas y requerirán el intercambio de reglas entre herramientas de modelado y motores de inferencia. La mayoría de estos sistemas son o serán construidos usando lenguajes de programación orientada a objeto para soportar técnicas de programación estandarizadas. Como no se conoce una forma estándar de modelar sistemas basados en conocimiento usando lenguajes de modelado de ingeniería del conocimiento, se plantea la necesidad de extender el uso de los lenguajes de modelado estandarizados de la ingeniería del software [2][3][5]. De este modo, sería posible integrar, reusar e interoperar entre los sistemas de las empresas y los motores de inferencia.

El planteamiento anterior evidencia la importancia de proponer estrategias de trabajo que potencien la reutilización a un alto nivel de abstracción. La Arquitectura Dirigida por Modelos (MDA en inglés) se ha convertido en una iniciativa para propiciar soluciones bajo este contexto. En este sentido, esta investigación propone la aplicación del enfoque MDA a una de las metodologías ampliamente conocida para el desarrollo de sistemas basados en conocimiento (COMMONKADS), y de esta manera generar un modelo independiente de la plataforma (*Platform Independent Model*, PIM) y un modelo específico de la plataforma (*Platform Specific Model*, PSM) que permitan la transformación así como la trazabilidad entre el modelo conceptual y de diseño de los SBC, para convertirlos de esta forma en los actores centrales del proceso de desarrollo, antes que el código.

El presente trabajo está estructurado de la siguiente manera: en la Sección II y III hablaremos brevemente acerca del estado del arte de los Sistemas Basados en Conocimiento y una de sus metodologías. Para posteriormente, en la Sección IV conversar

sobre MDA y luego aplicarlo en la Sección V, enfocándonos sobre una de las metodologías de SBC, donde se obtendrá un PIM y un PSM que serán validados subsiguientemente en la Sección VI. Finalmente, son presentadas nuestras conclusiones en la Sección VII.

II. LOS SISTEMAS BASADOS EN CONOCIMIENTO Y SU ENFOQUE

El *Object Management Group* (OMG) define un SBC como [6]: “Un Sistema Basado en Conocimiento, también conocido como sistema experto, es un software que tiene algún conocimiento o experticia acerca de un dominio reducido y es implementado tal que su base de conocimiento (BC) y arquitectura de control están separados. Los SBC frecuentemente incluyen procesamiento inferencial (opuesto al procesamiento algorítmico), explicación del razonamiento de los usuarios y la generación de resultados no únicos”. De la consideración de esta definición, podríamos decir que el conocimiento específico de un dominio es representado en una base de conocimiento, el cual es utilizado en el proceso de razonamiento del motor de inferencia para tomar decisiones relacionadas al dominio del problema. Esto es una característica única en un SBC, comúnmente utilizada para definirlos.

Las fases del proceso para construir un SBC en la ingeniería del conocimiento son en general: análisis de requerimientos identificando el alcance del SBC, diseño del sistema a través de la definición de las fuentes expertas en conocimiento y como representarlas, adquisición del conocimiento del experto a través de las diferentes técnicas de extracción y construcción para la base del conocimiento con las instancias del dominio, codificación, prueba, implementación y mantenimiento del sistema. Asimismo, la IS describe un conjunto de fases para el desarrollo de sistemas de software: Análisis de requisitos, Diseño del Sistema. Y, al comparar los procesos de desarrollo de la IS y la IC, se observa que la Ingeniería del Conocimiento presenta una fase adicional: “adquisición de conocimiento” Figura 1.

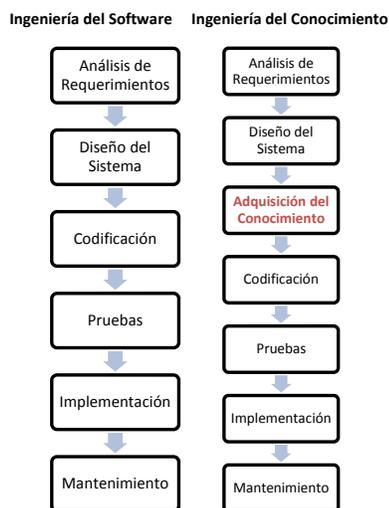


Figura 1: Comparación de los Procesos de Desarrollo para la IS y la IC [5]

La fase de adquisición de conocimiento es vital en el desarrollo del SBC debido a que el mismo es diseñado alrededor del conocimiento del experto en el dominio para la resolución de un problema en una tarea en particular, tal como un diagnóstico médico, una evaluación para un estudiante, etc. El conocimiento adquirido es entonces utilizado para poblar la base del conocimiento en forma de reglas con las cuales el sistema realizará el razonamiento.

La necesidad de administrar el conocimiento y las reglas del negocio a través de la tecnología ha causado que los SBC sean implementados en nuevos dominios y las capacidades de estos nuevos sistemas hayan sido explotadas para gestionar las competencias del ser humano. Algunos ejemplos son: diseño asistido para la arquitectura del software, gestión hospitalaria o clínica, manejo de riesgos en préstamos bancarios, asesoramiento en la bolsa de valores, entre otros. Más aún, son de gran valor ante la amenaza de rotación o cambio de personal y la pérdida del conocimiento (acumulación de hechos, eventos, procedimientos en la memoria del empleado que permiten la realización del trabajo y encontrarle sentido a los sucesos que pasan a su alrededor). Algunos autores [3][5][7], argumentan que esta “amenaza” es uno de los principales motores detrás de la aparición del enfoque en la gestión del conocimiento.

Por lo tanto, mientras los SBC tradicionales eran considerados aplicaciones estándares o tecnología de investigación en los laboratorios, hoy en día han resultado parte de los sistemas de información de las empresas, demandando aplicaciones comerciales ampliamente aceptadas por la industria. Gracias a esta alta aceptación, existe una fuerte demanda por parte de las empresas y los desarrolladores de estandarizar los lenguajes de representación de reglas y los procesos de desarrollo. La motivación detrás de todo esto son los problemas de interoperabilidad y mapeo entre los productos comerciales y las aplicaciones de las organizaciones. Estos problemas han emergido de las fusiones y adquisiciones de las compañías, legislación industrial, regulaciones de los negocios y la necesidad de intercambiar información y conocimiento entre socios.

III. MODELADO CONCEPTUAL CON COMMONKADS

Nuestro enfoque para el diseño del SBC utiliza la metodología COMMONKADS (CK). En sus inicios, KADS se centraba en el problema del cuello de botella que suponía la adquisición de conocimiento, para posteriormente convertirse en una metodología completa para el desarrollo de SBC [8]. En la actualidad, COMMONKADS, nombre que recibe la evolución de KADS, cubre la gestión del proyecto, el análisis organizacional y los aspectos relativos a las Ingenierías del Software y Conocimiento relacionados con el desarrollo del SBC. En COMMONKADS se incorporan tres ideas que han emergido, de la experiencia en la Ingeniería del Conocimiento y la Ingeniería del Software. Estas tres ideas se pueden concretar en tres conceptos: modelado, reutilización y gestión del riesgo.

El principal producto que resulta de la aplicación de COMMONKADS es el conjunto de modelos. Este conjunto de modelos se puede considerar una agrupación estructurada de conocimiento que refleja todos aquellos aspectos importantes para que el SBC tenga éxito dentro de un contexto organizacional determinado. Para reflejar los diferentes aspectos del contexto en el cual se quiere implantar el SBC, COMMONKADS ofrece seis modelos [3]: organización, tareas, agentes, comunicación, conocimiento y diseño Figura 2. Todos estos modelos están relacionados entre sí y pueden ser configurados gracias a unas plantillas que la metodología ofrece para su confección. Los tres primeros modelos describen el contexto en el cual se va a desenvolver el SBC.

Los modelos de conocimiento y agentes proporcionan los requisitos de entrada que guiarán la implementación del sistema a través del modelo de diseño. Como se puede deducir, cada una de estas agrupaciones intentan responder a cada una de las preguntas claves en el desarrollo de un SBC: ¿Es un SBC la solución idónea para el problema que se quiere resolver?, pregunta que se puede responder por la descripción del contexto dado por los tres primeros modelos; ¿Cuál es la naturaleza y la estructura tanto del conocimiento como de la comunicación utilizada?, cuya respuesta se puede encontrar en el modelo de conocimiento y en el modelo de comunicación; y finalmente, ¿Cómo debe ser implementado el conocimiento?, pregunta que intenta esclarecer el modelo del diseño.

Se hace mención especial al Modelo del Conocimiento, que está descrito en [8], el cual describe la experticia que tiene un determinado agente y que es relevante para la consecución de una determinada tarea, además de describir la estructura del mismo en función de su uso. Obviamente, este modelo se hace sin hacer referencia a aspectos de implementación. Para poder llevar a cabo el modelado de los distintos papeles que puede jugar el conocimiento, éste está distribuido en tres categorías disjuntas:

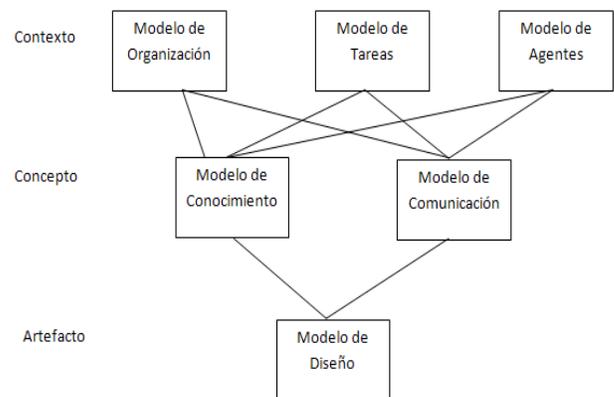


Figura 2: Modelos de la Metodología COMMONKADS [3]

- Conocimiento de tareas: Describe de una forma recursiva la descomposición de una tarea de alto nivel en varias subtareas. El conocimiento sobre una tarea se divide en dos partes: por un lado la tarea que sirve para especificar qué implica la aplicación de la misma, ya que define su objetivo en términos de los roles de entrada y de salida; por otro lado, está el método de la tarea, que define el

cómo se lleva a cabo la misma, indicando en qué subtareas se descompone y en qué orden deben de ser procesadas (control).

- Conocimiento del dominio: Especifica los hechos y asunciones que necesita el proceso de razonamiento para llevar a cabo su cometido en el dominio de la aplicación. El conocimiento puede ser estructurado en una serie de modelos del dominio que proporcionen una visión coherente de las distintas partes del mismo. Por lo tanto, en este apartado se va a especificar la forma, estructura y contenido del conocimiento relevante para la aplicación. La forma y la estructura constituyen lo que se denomina la ontología del dominio.
- Conocimiento sobre inferencias: Describe los procesos primitivos de razonamiento que tienen lugar en una aplicación, así como los roles de conocimiento que son usados por las inferencias. Obviamente, estos roles de conocimientos están relacionados con elementos del conocimiento del dominio. Hay que tener en cuenta, que las inferencias son consideradas primitivas respecto a un modelo de conocimiento determinado, ya que en otros modelos la misma inferencia puede ser una tarea descomponible.

COMMONKADS propone el lenguaje CML (*Conceptual Modelling Language*) [3], para materializar la especificación del modelo de conocimiento. Este lenguaje permite la definición de todos los elementos anteriormente citados, así como la estructuración del conocimiento en las partes anteriormente mencionadas. Además propone el uso de una notación gráfica, que permite no solo la definición de las estructuras de tareas (relación tareas-subtareas), sino que también permite la definición de la ontología, los conceptos del dominio y la definición de la dependencia de los datos entre las inferencias a través de las estructuras de inferencias. Aparte de CML, también se propone el uso de ML2 que está más orientado hacia la formalización del modelo. Esta descomposición del nivel de conocimiento en el conocimiento específico del dominio por un lado, y de las tareas y las inferencias por otro, favorece la reutilización del conocimiento en dos niveles. Al haber definido un conjunto de inferencias elementales independientes de la aplicación se permite que estas sean utilizadas en otros procesos de resolución de problemas.

El otro nivel de reutilización lo establece el conocimiento del dominio, que al ser definido independientemente de las tareas, puede ser reutilizado por otro conjunto de tareas definidas sobre el mismo dominio. Para favorecer la reutilización del conocimiento del dominio, COMMONKADS permite definir las ontologías a distintos niveles de generalización y distintos puntos de vista, lo que abre el abanico de posibles adaptaciones a otras aplicaciones. Otro de los aspectos importantes que introdujo COMMONKADS fue la definición de un marco de trabajo para la gestión y planificación del proyecto. CK define un ciclo de vida para el desarrollo del proyecto basado en un modelo en espiral como el propuesto por [9]. El modelo en espiral que plantea COMMONKADS se basa en los siguientes principios:

- La planificación del proyecto se centra principalmente en los productos y las salidas que tienen que producirse como resultado, más que un conjunto de actividades o fases.
- La planificación se realiza de una forma adaptativa a lo largo de una serie de ciclos en espiral, que están dirigidos por una valoración sistemática de los riesgos del proyecto.
- El control de calidad es una parte más de la gestión del proyecto, ya que la calidad está integrada en el desarrollo del SBC por medio de la metodología.

Estos principios están garantizados por un lado, por el conjunto de modelos, y por otro, por el ciclo de vida en espiral. Este ciclo de vida consta de cuatro fases:

- Revisión: Es el primer paso de cada ciclo y en él se revisa el estado actual del proyecto y se establecen los objetivos principales que se quieren cubrir en el ciclo en cuestión.
- Valoración de riesgos: Las líneas generales del proyecto establecidas en el paso anterior sirven de entradas para esta fase. Su función principal es la identificación y valoración de los principales obstáculos que nos podemos encontrar para la consecución exitosa del proyecto, así como las acciones que se deben tomar para minimizar dichos riesgos.
- Planificación: Una vez obtenida una visión clara de los objetivos que hay que cubrir, los riesgos que se pueden presentar y las acciones que hay que tomar, hay que realizar una planificación del trabajo a realizar. En dicha planificación, hay que establecer la distribución de la carga del trabajo en términos de qué tareas hay que realizar, una temporalización de dichas tareas, la distribución de los recursos, etc.
- Monitorización: Es la última fase del ciclo y está constituida por el desarrollo propiamente dicho. El trabajo realizado en esta fase está controlado y dirigido por el director del proyecto. Para determinar el grado de cumplimiento de los objetivos se requieren reuniones con los agentes implicados en el proyecto (usuarios, administradores, expertos,...). El resultado de dichas reuniones se utilizará como entrada del proceso de revisión del siguiente ciclo.

COMMONKADS abarca todo los aspectos del desarrollo de un SBC, proporcionando un marco de trabajo donde llevar a cabo la gestión del proyecto. Todo esto hace que COMMONKADS haya sido adoptado no solo en Europa, donde se ha convertido en un estándar de facto [3], sino que también empieza a ser utilizado en el resto del mundo.

No obstante, existe una necesidad de extender el uso de los lenguajes de modelado estandarizados de la ingeniería del software a la ingeniería del conocimiento, que permitan integrar, reusar e interoperar entre los diferentes sistemas y motores de inferencia [2]. Basado en lo anterior, se evidencia la importancia de proponer estrategias de trabajo que potencien la reutilización a un alto nivel de abstracción. La Arquitectura Dirigida por Modelos (MDA) se ha convertido en una iniciativa para propiciar soluciones bajo este contexto.

IV. ARQUITECTURA DIRIGIDA POR MODELOS (MDA)

La Arquitectura Dirigida por Modelos es un estándar del *Object Management Group* (OMG) que dirige completamente el ciclo de vida del diseño, despliegue, integración y manejo de las aplicaciones a través del uso de modelos en el desarrollo del software. MDA separa la especificación de la funcionalidad del sistema, de la especificación de la implementación sobre una plataforma de tecnología específica.

En consecuencia, MDA incentiva la especificación de un modelo independiente de la plataforma (PIM, en inglés) el cual no contiene información sobre la tecnología usada para llevar a cabo éste. Entonces, el PIM puede ser transformado a un modelo específico de la plataforma (PSM, en inglés) para incluir información acerca de la tecnología específica a utilizar para ser realizado. Posteriormente, cada PSM resultante puede ser transformado a código para ser ejecutado sobre la plataforma y obtener el producto final de software Figura 3. Sobre el tope de estos modelos, MDA también presenta un modelo computacional independiente (CIM, en inglés). Éste describe el sistema dentro de su medio ambiente (por ejemplo: dominio del negocio) y muestra lo que se espera haga el sistema sin enseñar detalles acerca de cómo será construido.

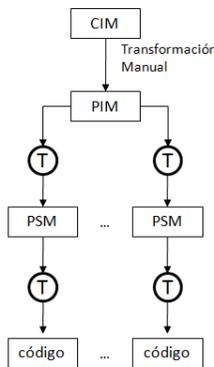


Figura 3: Marco de Trabajo de Arquitectura Dirigida por Modelos [10]

El CIM, PIM y el PSM pueden ser desarrollados usando cualquier lenguaje de especificación, pero típicamente es usado UML desde que éste es un lenguaje de modelado estándar para propósito general y al mismo tiempo puede ser extendido para definir lenguajes especializados para ciertos dominios (por ejemplo: extensión de metamodelos o perfiles).

V. MARCO DE REFERENCIA METODOLÓGICO PROPUESTO

Para el diseño del SBC en esta investigación, se propone la aplicación del marco de referencia metodológico basado en MDA de la Figura 4, obtenido de realizar una combinación entre COMMONKADS y la Arquitectura Dirigida por Modelos.

La selección de los distintos artefactos escogidos para esta propuesta, están fundamentados en unificar el lenguaje de modelado del conocimiento y proveer de transformaciones que permitan el mapeo o trazabilidad entre el diseño y la implementación del sistema.

De la Figura 4, observamos un:

A. Modelo Independiente de la Computación (CIM)

Compuesto por un diagrama de casos de uso y los 3 primeros modelos de COMMONKADS. Su objetivo al haber sido incluidos, es tener una visión global del negocio o la organización, sus procesos, tareas y recursos, que servirán de guía para generar el PIM. Para cada modelo perteneciente al CIM, se seleccionan y establecen los siguientes artefactos provenientes de COMMONKADS:

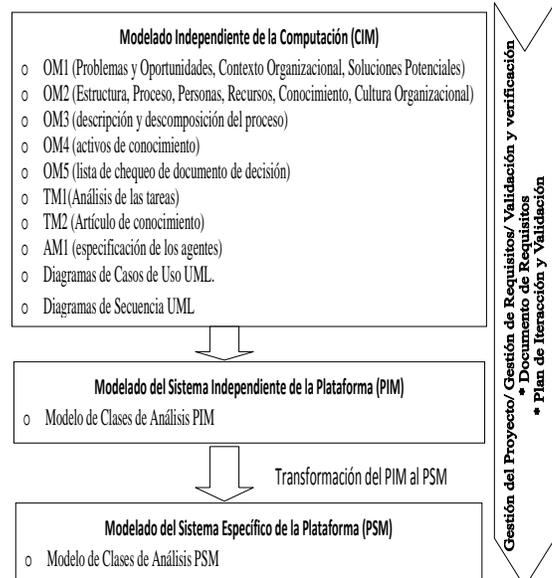


Figura 4: Marco de Referencia Metodológico Propuesto Basado en MDA para el Diseño de un SBC

Modelo de Organización:

- OM1 (problemas y oportunidades, contexto organizacional, soluciones potenciales)
- OM2 (estructura, proceso, personas, recursos, conocimiento, cultura organizacional)
- OM3 (descripción y descomposición del proceso, diagramas de actividad)
- OM4 (activos de conocimiento)
- OM5 (lista de chequeo de documento de decisión)

Modelo de Tareas:

- TM1 (análisis de las tareas)
- TM2 (artículo de conocimiento)

Modelo de Agentes:

- AM1 (especificación de los agentes)

Así mismo, de la psicología cognitiva logramos obtener, que el ser humano puede dividir sus procesos intensivos de conocimiento, en tareas de razonamiento [8]:

- Analíticas: por lo general el sistema a estudiar “pre-existe”, pero típicamente no es completamente conocido. Las tareas analíticas se pueden dividir en: clasificación, evaluación, diagnóstico, predicción, monitoreo.
- De síntesis: en estas tareas, el sistema por lo general no existe y debe ser construido. Las mismas se pueden clasificar en: síntesis, diseño, asignación, planificación, calendarización y modelación.

De esta manera, se representan a nivel general las distintas actividades que podría realizar un SBC. El mismo es diagramado en UML y puede ser visto en la Figura 5.

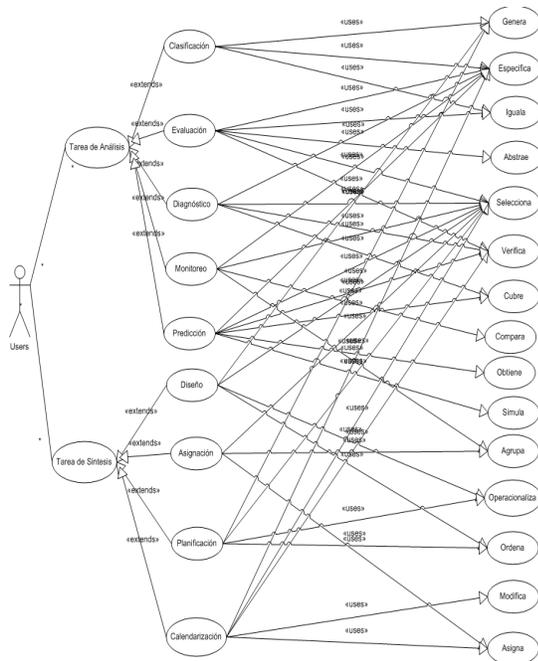


Figura 5: Diagrama de Casos de Uso General para el CIM

B. Modelo del Sistema Independiente de la Plataforma (PIM)

Para obtener el modelo independiente de la plataforma, se selecciona COMMONKADS para extraer de éste los artefactos y conceptos que nos ayudarán a la creación del modelo. Entre los conceptos más importantes identificados se mencionan:

- Tareas: representan funciones de razonamiento complejo. Una tarea define una meta de razonamiento en términos de una entrada y una salida.
- Métodos de Tarea: describe cómo una tarea puede ser realizada a través de la descomposición de sub-funciones.
- Inferencias: describe un paso de razonamiento primitivo, o lo que es igual a decir, una actividad de razonamiento simple. La realización de varias inferencias puede dar lugar al cumplimiento de una tarea.
- Roles de conocimiento: son nombres abstractos de objetos de datos que tienen una participación en el proceso de razonamiento. Por ejemplo: “hipótesis”, es un nombre

funcional para un objeto en el dominio que juega el rol de una solución candidata.

- Dominio del conocimiento: describe la información estática y los objetos del conocimiento para el dominio de la aplicación. Un dominio de conocimiento típicamente está compuesto por un esquema de dominio y una base de conocimiento.

Basándonos en todo lo anterior, se establecen un conjunto de diagramas de secuencia UML y un modelo conceptual genérico, que posteriormente podrán ser instanciados dependiendo de la tarea a realizar por el SBC, los cuales pueden ser apreciados en las Figuras 6, 7, 8, no siendo mostrados todos los artefactos por limitaciones de espacio.

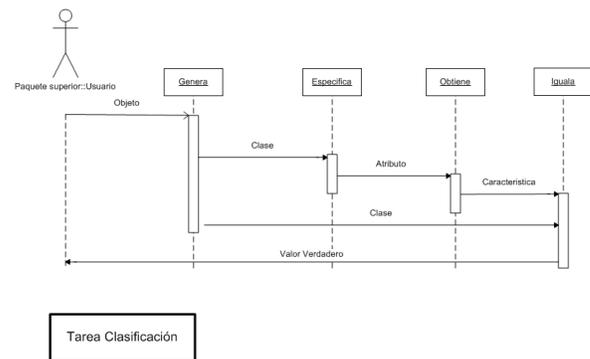


Figura 6: Diagrama de Secuencia Genérico “Clasificación” para el PIM

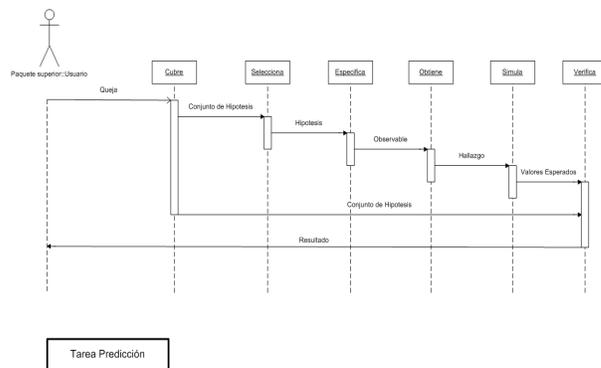


Figura 7: Diagrama de Secuencia Genérico “Predicción” para el PIM

C. Modelo del Sistema Específico de la Plataforma (PSM)

Conformado por un diagrama de clases que modela los componentes de un SBC a nivel general para la plataforma Java. Éste permitirá posteriormente instanciar los distintos elementos resultantes al modelo específico de la plataforma para un caso de estudio. El mismo, puede ser visto en la Figura 9.

Se escoge Java, debido a sus diversas ventajas, entre ellas: su soporte para múltiples plataformas, el respaldo de empresas reconocidas, arquitectura de varias capas y atributos de calidad.

TRANSFORMACIÓN: representa el mapeo o trazabilidad que permitirá convertir el PIM en un PSM. Existen diferentes enfoques para dar a un modelo la información necesaria. La OMG propone 4 enfoques para realizar esta transformación [10]. Estas son:

- Transformación Manual
- Transformación de un PIM preparado utilizando un perfil.
- Transformación usando patrones y marcas
- Transformación automática

Para este caso, fue utilizada la transformación manual, obteniendo el Mapeo de la Tabla I y Figura 10.

Tabla I: Mapeo PIM a PSM

PIM	Transformación	PSM
Tarea	T1	Class
Método de Tarea	T2	Method
Inferencia	T3	Method
Dominio de Conocimiento	T4	Package
Base de Conocimiento	T5	Method
Concepto	T6	Class
Función de Transferencia	T7	Method
Relaciones	T8	Class
Atributo de Valor	T9	Attribute
Roles de Conocimiento	T10	Method
Tipo de Reglas	T11	Method
Esquema de Dominio	T12	Class

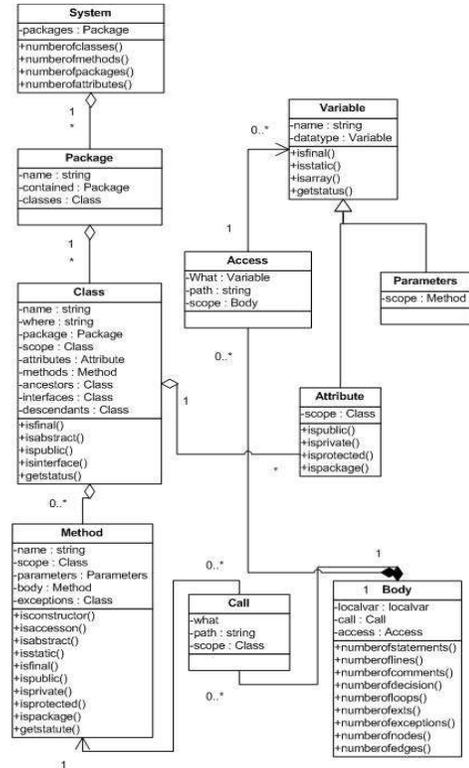


Figura 9: Diagrama de Clases General para el PSM

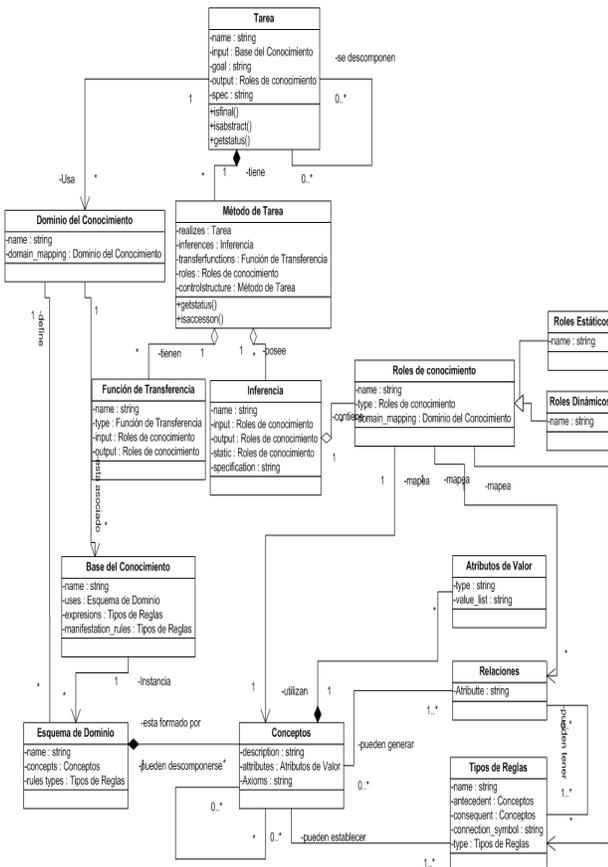


Figura 8: Modelo Conceptual General para el PIM

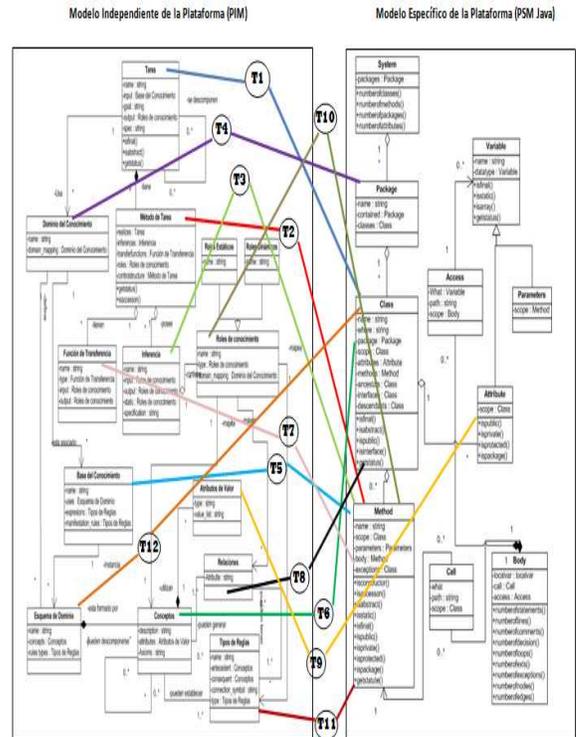


Figura 10: Transformaciones PIM a PSM

VI. INSTANCIACIÓN DEL MARCO DE REFERENCIA
 METODOLÓGICO

Después de haber definido el PIM, PSM y las transformaciones en el apartado anterior, se procede a instanciar los mismos. Todos estos orientados al marco de referencia mostrado en la Figura 4, para el modelado de un sistema que permita el pronóstico del cáncer de seno.

El proceso de gestión del paciente con cáncer de seno sigue generalmente el siguiente flujo de tareas: diagnóstico inicial, confirmación del diagnóstico, pronóstico, asignación de la terapia y monitoreo. El pronóstico como la predicción del comportamiento del tumor, es un punto crucial en la gestión del paciente. La definición de esta tarea depende principalmente si estamos buscando propiedades actuales del cáncer de seno o si deseamos identificar estados actuales no observables (por ejemplo, pasado, presente y futuro de características de crecimiento biológico)

Bajo este orden de ideas y revisando la definición de pronóstico en [11], indica que estamos tratando a nivel cognoscitivo con un problema de “análisis” con el objetivo de “predecir” un comportamiento, el cual según la Figura 5, involucra las sub-tareas de cubrir, seleccionar, especificar, obtener, simular y verificar. Basándonos en estas premisas, se procede a aplicar el marco metodológico propuesto.

A. Obteniendo El CIM Instanciado (Paso 1)

Para conseguir el CIM, se realizan todos los artefactos planteados en la sección V para el modelo organizacional, de tareas y agentes. Sin embargo, no son presentados completamente todos los artefactos por limitaciones de espacio. A continuación se muestran algunos en las Tablas II, III, IV:

Modelo Organizacional

Tabla II: Artefacto OM3 (Modelo Organizacional)

#	Tarea	Ejecutada por	Donde	Activo de Conocimiento	Intensivo	Significancia
1	Diagnóstico Inicial	Imagenólogo/ Bioanalista	Centro de Diagnóstico de Imagen	Clasificación Internacional de Enfermedades con modificación clínica	Sí	2
2	Confirmación del Diagnóstico	Médico Oncólogo/ Junta Médica	Hospital o Clínica	* Diagnóstico y Tratamiento de cáncer de seno	Sí	3
3	Pronóstico	Médico Oncólogo	Hospital o Clínica	* Guía sobre los factores de pronóstico en el cáncer de seno	Sí	4
4	Selección de la Terapia	Médico Oncólogo/ Junta Médica	Hospital o Clínica	* Guía para el tratamiento multidisciplinario en el cáncer de seno operable * Guía para Quimioterapia primaria en el cáncer de seno operable	Sí	4
5	Seguimiento	Médico Oncólogo	Hospital o Clínica	* Guía para el tratamiento multidisciplinario en el cáncer de seno operable	Sí	3

1= Menos Significativo
 5= Muy Significativo

Modelo de Tareas

Tabla III: Artefacto TM2 (Modelo de Tareas)

Nombre	Guía sobre los factores de pronóstico en el cáncer de seno	
Poseído por	Médico Oncólogo / Sistema de Conocimiento	
Usado en	3.- Pronóstico	
Dominio	Cáncer de Seno	
Naturaleza del conocimiento		Cuello de botella / Para ser mejorado
Formal, riguroso	X	
Empírico, cuantitativo		
Heurístico, reglas de oro		
Altamente especializado, específico del dominio	X	
Basado en la experiencia	X	
Basado en la acción		
Incompleto		
Incierto, podría ser incorrecto		
Rápidamente cambiante	X	X
Difícil de verificar		
Tácito, difícil de transferir		
Forma del conocimiento		
En la mente		
Papel	X	
Electrónico	X	
Habilidad de acción		
Otro		
Disponibilidad del conocimiento		
Limitaciones en tiempo	X	
Limitaciones en espacio		
Limitaciones en acceso		
Limitaciones en calidad		
Limitaciones en forma	X	X

Modelo de Agente

Tabla IV: Artefacto AM1 (Modelo de Agente)

Nombre	Médico Oncólogo
Organización	Hospital o Clínica. Juega el rol primario en el diagnóstico y pronóstico del paciente
Involucrado en	2.- Confirmación del diagnóstico 3.- Pronóstico 4.- Selección de la terapia
Se comunica con	5.- Seguimiento Base de Datos Expertos / Investigadores Junta Médica Pacientes
Conocimiento	Diagnóstico y tratamiento del cáncer de seno Factores de pronóstico en el cáncer de seno Tratamiento para el cáncer de seno Quimioterapia para el cáncer de seno
Otras competencias	Habilidad para investigar, evaluar y discernir entre distintos criterios de expertos Asegurarse de la selección de una terapia adecuada que maximice las posibilidades favorables de curación de la enfermedad, basado en un correcto diagnóstico y
Responsabilidades y limitaciones	pronóstico de la enfermedad del paciente

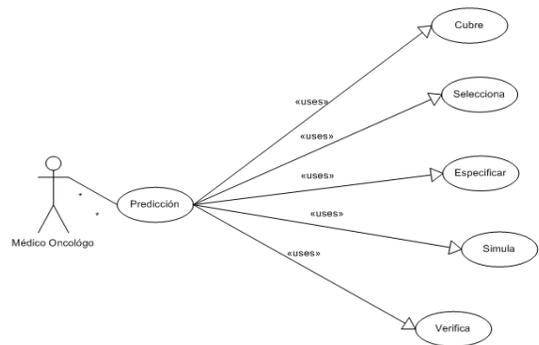


Figura 11: Transformaciones PIM a PSM

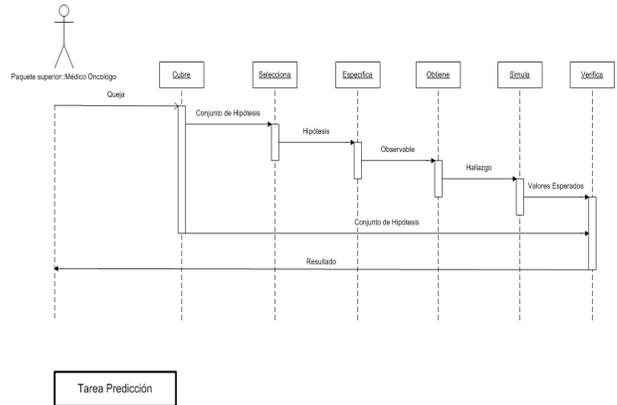


Figura 12: Diagrama de Secuencia para el "Pronóstico" del Cáncer de Seno

También es generado el diagrama de casos de uso, que resulta de seleccionar la tarea analítica "predicción" del diagrama general que se expuso en la Figura 5. El mismo puede ser apreciado en la Figura 11.

B. Obteniendo El PIM Instanciado (Paso 2)

Para obtener el PIM como se explicó previamente, se selecciona del conjunto de diagramas de secuencia mostrados en la sección V, aquella tarea que más se adapte a la actividad de conocimiento a realizar por el SBC, en nuestro caso "predicción", el cual puede ser apreciado en la Figura 12.

Adicionalmente, basados en el modelo conceptual de la Figura 8 y los factores de pronóstico de cáncer de seno [11], se obtiene el modelo conceptual del SBC para el pronóstico del cáncer de seno mostrado en la Figura 13.

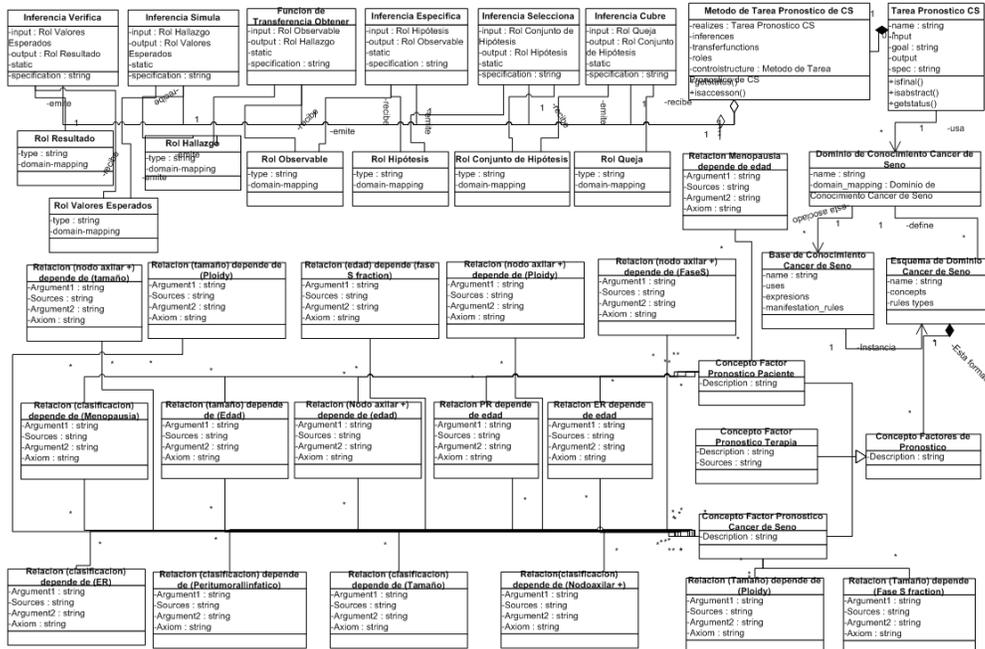


Figura 13: Modelo Conceptual del SBC para el "Pronóstico" del Cáncer de Seno

C. Obteniendo El PSM Instanciado (Paso 3)

Aplicando las transformaciones presentadas en la Figura 10 al modelo conceptual de la Figura 13 y verificando el resultado con el diagrama de clases de la Figura 9, se obtiene el modelo específico de la plataforma del Sistema Basado en Conocimiento para el pronóstico del cáncer de seno. El mismo puede ser apreciado en la Figura 14.

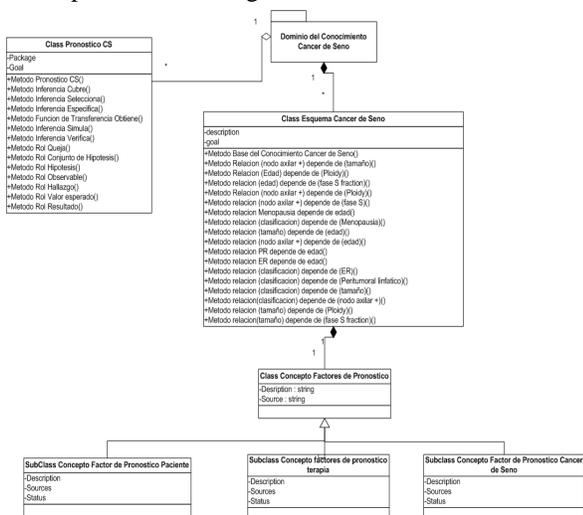


Figura 14: Diagrama de Clases del SBC para el “Pronóstico” del Cáncer de Seno

VII. CONCLUSIONES

Este trabajo ha mostrado la importancia de los sistemas basados en conocimiento y su problemática en la carencia de metodologías que establezcan un mapeo efectivo entre el modelo conceptual y el modelo de diseño.

Particularmente, se estudió COMMONKADS, de la cual se pudo observar que aunque es bastante completa y abarca la mayoría de las fases del ciclo de vida de estos sistemas, adolece de un lenguaje de representación estándar para su producto clave “el conocimiento”. Lo anterior, aunado a la ausencia de una trazabilidad clara entre sus modelos conceptuales y de diseño, genera una limitante para el desarrollo, reutilización e interoperabilidad de los mismos.

Con esta investigación, se plantea una alternativa a través de MDA buscando disminuir las carencias citadas anteriormente. El marco de referencia propuesto consigue establecer una trazabilidad entre el modelo conceptual y de diseño, así como unificar el lenguaje de modelado a través de UML, el cual es

ampliamente reconocido hoy en día por los ingenieros y desarrolladores de software, lo que permite disminuir los costos en los tiempos de aprendizaje y ganar mayor interoperabilidad entre los diversos sistemas.

Mediante el caso de estudio, se logró validar el marco de referencia propuesto, obteniendo tiempos de modelado considerablemente menores en comparación con lo que toma aplicar la metodología pura y simple, además de aprender la misma. No obstante, como resultado de nuestra experiencia, logramos validar que también se pierde un poco la riqueza semántica del modelo de conocimiento, por lo cual recomendamos la combinación de estas metodologías ya existentes con la propuesta planteada, para obtener un mejor modelado del sistema.

Se propone como trabajo futuro hacer una extensión o perfil para UML dirigido a los SBC, para evitar la pérdida del contenido semántico en el modelado.

REFERENCIAS

- [1] P. Drucker. *Los Desafíos de la Gerencia para el El Siglo XXI*. 3th Edition, Norma, Febrero 1999.
- [2] J. Cañadas, S. Túnez, J. Palma, *Aplicación de MDA a la Ingeniería del Conocimiento*, unpublished.
- [3] J. Palma, E. Paniagua, F. Martin, R. Marin, *Ingeniería del Conocimiento. de la Extracción al Modelado del Conocimiento*, Revista Iberoamericana de Inteligencia Artificial, vol. 4, no. 11, pp. 46-72, Abril 2000.
- [4] M. del Águila, J. Cañadas, J. Palma, S. Túnez, *Towards a Methodology for Hybrid Systems Software Development*, in proceeding of the XVIII International Conference on Software Engineering & Knowledge Engineering (SEKE 06), San Francisco, USA, July 2006.
- [5] M. Abdullah, *A UML Profile for Conceptual Modeling of Knowledge Base System*, Ph.D. thesis, Department of Computer Science, University of York, York, United Kingdom, April 2006.
- [6] *KBE Services for Engineering Design*, <http://www.omg.org/schedule/>.
- [7] M. Henao, *COMMONKADS-RT: Una Metodología para el Desarrollo de Sistemas Basados en el Conocimiento de Tiempo Real*, thesis Doctoral, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Valencia, España, Abril 2001.
- [8] A. Schreiber, J. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. Van del Velde, J. Wielinga, *Knowledge Engineering and Management. The COMMONKADS Methodology*, 1st Edition, MIT Press, February, 2000.
- [9] B. Boehm, *A Spiral Model of Software Development and Enhancement*, IEEE Computer, vol. 21, no. 5, pp. 61-72, May 1988.
- [10] *MDA Guide Version 1.0.1*, <http://www.omg.org/docs/omg/03-06-01.pdf>.
- [11] S. Roberto. *Using COMMONKADS to Build an Expertise Model for Breast Cancer Prognosis and Therapy*. RFC 2737, November 1995.

Sistema Buscador de Contenidos Digitales de la Nube de Conocimiento del Proyecto Madre

Jose Aguilar¹, Ricardo Dos Santos¹
aguilar@ula.ve, ricardojds@gmail.com

¹ Departamento de Sistemas Computacionales, Universidad de Los Andes, Mérida, Venezuela

Resumen: Este artículo propone un sistema buscador de contenidos digitales adaptados a los estilos de aprendizajes, utilizando tecnologías como los servicios web para lograr mayor interoperabilidad entre sistemas, y ontologías para permitir razonar sobre el conocimiento. Este sistema brinda contenidos que coincidan con las herramientas, actividades e instrumentos, según el perfil educativo de los estudiantes. Para ello, consulta a un servicio web que provee información de los paradigmas de aprendizaje, y con esta información optimiza la búsqueda de los contenidos, logrando responder a la necesidad particular de un estudiante según su estilo de aprendizaje. Además se muestra la integración de este servicio web con el Proyecto Madre, donde interactúan un conjunto de servicios que garantizan un proceso de auto-formación, que va a ser ejecutado por el estudiante guiado por los profesores.

Palabras Clave: Contenidos Digitales; Paradigmas de Aprendizaje; Arquitectura Guiada por Ontologías (ODA); Arquitectura Orientada a Servicios (SOA); Plataforma Web Educativa.

Abstract: This article proposes a searcher system of digital contents adapted to the learning styles, utilizing technologies as the web services for to achieve higher interoperability between systems, and ontologies for to permit to reason about the knowledge. This system offers contents that coincide with the tools, activities and instruments, by the educative profile of the students. For that, the system consult to a web service that provides knowledge of the learning paradigms, and with this knowledge optimizes the search of the contents, achieving to respond to the particular necessity of a student by him learning style. Further it show the integration of this web service with the Mother Project, where interacts a set of services that guarantees a process of self-formation, that will be executed for the student and guided for the teachers.

Keywords: Digital Contents; Learning Paradigms; Ontology Driven Architecture (ODA); Service Oriented Architecture (SOA); Educational Web Platform.

I. INTRODUCCIÓN

En Venezuela se viene desarrollando un constante cambio a nivel educativo inspirada en los principios y en las conclusiones de la conferencia mundial sobre la educación superior realizada por la UNESCO (Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura) en octubre de 1998 en París. Se reconoce que hay, entre otras, una gran desigualdad en la distribución del conocimiento, y por lo tanto, con las posibilidades de desarrollo en los ciudadanos, por lo que se requiere de grandes cambios con respecto a la pertinencia del conocimiento impartido, determinado por su carácter social, la inclusión de sectores tradicionalmente excluidos de los sistemas educativos, y la flexibilización de las mallas curriculares para permitir mayor fluidez y libertad al proceso de formación académica.

Los Programas Nacionales de Formación (PNF), creado en el 2008, es el primer cambio rumbo a esa igualdad y flexibilidad, que queda plasmada en el currículo que forja los procesos

de formación educativa en cuatro dimensiones principales: aprender a conocer, aprender a hacer, aprender a vivir (junto a aprender a convivir) y aprender a ser [1]. Los PNF se vienen dictando en los diferentes Institutos Universitarios de Tecnología (IUT) y los Colegios Universitarios (CU), que en el transcurso del tiempo son transformado en las Universidades Politécnicas Territoriales (UPT).

Siguiendo estos cambios, la Universidad de Los Andes (ULA), a través del Proyecto Madre, busca formar parte de esta transformación educativa, creando una carrera experimental denominada "Ingeniería en Computación e Informática", empleando el paradigma de aprendizaje basado en nubes. El paradigma lo conforman tres nubes que representan la auto-formación, los paradigmas de aprendizajes, y las fuentes de conocimiento. Las nubes permiten una dinámica muy particular, donde se puede navegar de una nube a otra, ya que las fronteras no están claramente definidas y sus elementos pueden aparecer o desaparecer.

La singularidad del paradigma basado en nubes requiere de cambios en todos los niveles de la institución, y uno de esos cambios radica en el sistema que acompañan a los profesores, estudiantes, y demás miembros de la comunidad, durante su estadía en la universidad. El sistema ya se encuentra en construcción, con desarrollos en las diferentes nubes, pero existe una necesidad latente en la nube de conocimiento, porque el actual desarrollo solo cubre objetos de aprendizajes, dejando por fuera contenidos como cursos en línea, guías, artículos y libros, entre otros.

Esta investigación soluciona la necesidad de encontrar contenidos digitales generales que necesitaba la nube de conocimiento. Dichos contenidos están regados en la inmensa Internet, permitiendo la búsqueda integrada al estilo de aprendizaje de los estudiantes, para ayudarlos en cada uno de los temas que conforman los módulos de la carrera.

Para el desarrollo de este sistema buscador de contenidos se usan ontologías, junto con el paradigma arquitectura orientada a servicios (SOA). Ambas herramientas tecnológicas permiten, entre otras cosas, la capacidad de deducir y compartir información (interoperabilidad); bases fundamentales dentro del planteamiento tecnológico educativo que propone el “Proyecto Madre”.

Este artículo está estructurado en siete secciones. La Sección I ofrece una visión del contexto general de la presente investigación. La Sección II define las bases teóricas de la propuesta, que consiste en el proyecto madre, la ontología con su capacidad de modelar el conocimiento, y por último, los contenidos digitales y sus características; La Sección III describe la construcción del sistema siguiendo la metodología para el desarrollo de aplicaciones basado en el paradigma de arquitectura orientada por ontología. La Sección IV detalla la implementación del sistema según la metodología, presentando el modelo específico de la plataforma. La Sección V muestra los casos de experimentación para probar todas las funcionalidades del sistema. La Sección VI compara otros trabajos con la presente investigación. Finalmente, la Sección VII especifica los logros y las posibles mejoras en la investigación.

II. MARCO TEÓRICO

A. Fundamentos Teóricos del Proyecto Madre

1) *Proyecto Madre*: El proyecto madre nace por la necesidad de compensar la carente calidad del proceso de aprendizaje de un estudiante, este proyecto se viene desarrollando en el Departamento de Computación de la Universidad de Los Andes. Las ideas planteadas se basan en la creación de una carrera experimental en el ámbito de las Tecnologías de la Información, utilizando el paradigma de las nubes. Basada en dicho paradigma, la misma tiene características fuera de lo convencional, como por ejemplo, una dinámica centrada en los grupos de investigación, mallas curriculares adaptables a varios perfiles de formación profesional en TICs, y un modelo pedagógico basado en el paradigma “aprender haciendo”, todo esto garantizando un proceso de auto-formación basada en recursos educativos, que va a ser ejecutado por el estudiante guiado por los profesores [2].

2) *Organización en Nubes*: El paradigma de aprendizaje basado en nubes lo conforman tres nubes que representan la auto-formación, los paradigmas de aprendizaje, y las fuentes de conocimiento, como se observa en la Figura 1; las nubes permiten una dinámica muy particular, donde se pueden navegar de una nube a otra, ya que las fronteras no están claramente definidas y sus elementos pueden aparecer o desaparecer, así como las interrelaciones entre ellos [2].

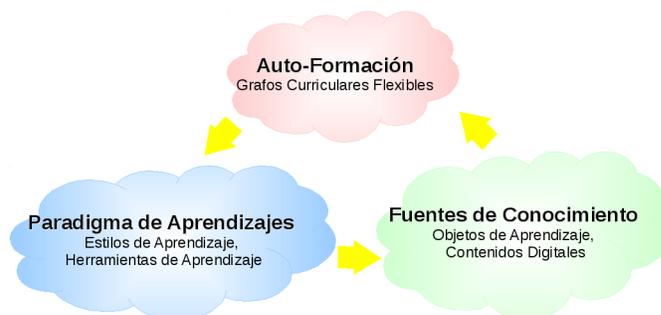


Figura 1: Paradigma de las Nubes [2]

Nube de Auto-Formación

La nube de auto-formación es el eje principal de la vida del estudiante, con toda la información proveniente de las nubes de los paradigmas de aprendizajes y de las fuentes de conocimiento. Quien entre a la nube de auto-formación debe estar consciente de los pre-requisitos exigidos por los módulos de las materias, que pueden ser cubiertos por cursos previos, pueden haber sido adquiridos por procesos de auto-formación, o eventualmente, pueden ser cubiertos durante la realización del módulo.

Esta nube permite al estudiante construir la malla curricular según su interés, logrando así conseguir competencias específicas y titulaciones en diferentes niveles: ingeniero, técnico medio, entre otros. Los perfiles de formación definidos para el recorrido del estudiante fueron inspirados en la curricula definida por la IEEE/ACM, donde se especifican 5 perfiles curriculares/profesionales, que son [3]; Ciencias de la Computación, Ingeniería del Software, Ingeniería del Computador, Sistemas de Información y Tecnologías de Información.

Nube de Paradigmas de Aprendizaje

En la nube de los paradigmas de aprendizaje es donde recae la responsabilidad de aportar los mecanismos necesarios para la auto-formación, según el estilo de aprendizaje del estudiante. Dicho estilo de aprendizaje determinará las estrategias y herramientas que mejor se adapten a la capacidad y aptitud de un estudiante particular. Esta nube guía las dinámicas de auto-formación, establece formas de acreditar cursos, posibilita maneras de plasmar los conocimientos adquiridos, espacios de intercambio, de trabajo colaborativo, de construcción colectiva del conocimiento, entre otras cosas. Permite navegar entre el mundo presencial y virtual en el proceso de aprendizaje, garantizando elementos humanísticos en el proceso de aprendizaje.

El estudiante posee un estilo de aprendizaje representado por un número de cuatro dígitos, cada dígito de este número es una

dimensión según el modelo pedagógico de Felder-Silverman. Las cuatro dimensiones en las que se clasifica a los perfiles de aprendizajes de los estudiantes son [4]: Activos o Reflexivos, Sensitivos o Intuitivos, Visuales o Verbales y Secuenciales o Globales.

Nube de Fuentes de Conocimiento

Por último, en la nube de fuentes de conocimiento confluyen las diferentes fuentes de información distribuidas a lo largo y ancho de la Internet. Ese conocimiento que se le entrega al estudiante esta adaptado a su perfil, primero basado en los requerimientos establecidos en la nube de auto-formación (por ejemplo, módulos a cursar), y segundo basado en los instrumentos, las actividades, y las herramientas especificadas en la nube de paradigma de aprendizajes para el estilo de aprendizaje del estudiante. Esta nube requerirá de herramientas para organizar, analizar, explotar, criticar, dicho conocimiento (es decir, de gestión del conocimiento), haciendo énfasis en la adquisición de conocimientos, en las destrezas para usarlo y reflexionar sobre él, y en la actitud como ciudadano para reconocerlo en el contexto social donde este inmerso.

Actualmente esta nube posee conocimientos sobre objetos de aprendizajes [5], pero esta fuente de conocimiento se queda corta con la gran gama de información que esta dispersa en la Internet, debido a esto, la presente investigación amplia de manera general la capacidad de albergar cualquier contenido digital.

B. Arquitectura Orientada por Ontologías (ODA)

La Ingeniería Dirigida por Modelos (Model Driven Engineering o MDE) es una metodología de desarrollo de software que se centra en la creación de modelos o abstracciones, para modelar y especificar sistemas (computacionales, reales, etc.)

La arquitectura dirigida por modelos (Model Driven Architecture o MDA) [6] es tal vez la más prometedora y ambiciosa propuesta de la Object Management Group (OMG), se basa en los conceptos de MDE, pero utilizando los estándares de la OMG: MOF (MetaObject Facility), XML (Metadata Interchange), OCL (Object Constraint Language), UML (Unified Modeling Language) y CWM (Common Warehouse Metamodel). En MDA se propone una forma de desarrollo del software basada en varios niveles de abstracción. Los niveles de abstracción van de lo mas abstracto a lo mas concreto, comenzando su ciclo de vida por la capa CIM y terminando en el Código.

Mientras MDA proporciona un marco de gran alcance en la Ingeniería del Software, las tecnologías semánticas, naturalmente, pueden ampliarla para permitir la representación de vocabularios inequívocos de dominio, la verificación del modelo de consistencia, y el incremento de las capacidades de expresividad en la representación de restricciones. Los Modelos semánticos u ontologías, amplían la metodología base de la OMG, dando lugar a la Arquitectura Dirigida por Ontología.

La Arquitectura Dirigida por Ontología (Ontology-Driven Architecture o ODA) [7] se basa en ontología. En pocas

palabras, ODA centra su desarrollo en las ontologías, las cuales además de describir el dominio, son capaces de representar el contenido semántico del sistema. En cambio, el MDA se centra en el desarrollo de modelos, posando en ellos toda la carga de especificación, análisis y diseño del sistema a desarrollar. El enfoque de ODA es complementario al MDA.

Una metodología para el desarrollo de aplicaciones basado en el paradigma ODA [8], consiste básicamente en las tres etapas de desarrollo: conceptualización, especificación y realización. En el desarrollo de cada una de estas etapas, se definen una serie de diagramas que ayudan a generar la ontología resultante para cada fase (ver Figura 2).



Figura 2: Fases de la Metodología [8]

Las tres fases del desarrollo metodológico son las siguientes:

- **Conceptualización o Modelo Independiente de la Computación (CIM):** En esta fase se representa de manera compacta la conceptualización del problema a través de un modelo ontológico; que usa como insumo los siguientes diagramas UML: Casos de Uso, Actividades y Secuencia. Ese modelo ontológico podría ser extendido por ontologías propias del dominio de aplicación, las cuales pueden describir estándares, normas, etc., lo que enriquece la descripción semántica de la conceptualización del problema. A ese modelo ontológico se le denomina modelo ontológico conceptual del problema.
- **Especificación o Modelo Independiente de la Plataforma (PIM):** En esta fase el objetivo fundamental es definir una ontología centrada en el diseño del sistema, sin considerar los detalles necesarios para su implementación en una plataforma determinada. Además, se debe definir la ontología que describe los paradigmas de arquitectura de software a usar para el desarrollo de la aplicación. De esta manera, la ontología de la capa PIM fusiona los aspectos específicos de diseño del sistema bajo desarrollo con los paradigmas de software a usar en su desarrollo. El diagrama UML fundamental para esta fase es el Diagrama de Clases, y la ontología producto de esta fase es el modelo ontológico de diseño del sistema.
- **Realización o Modelo Específico de la Plataforma (PSM):** En esta fase se especifica detalladamente en una ontología el proceso de implementación del sistema bajo

desarrollo, así como sus componentes, y la interacción entre ellos. Para ello, se usan como insumo los diagramas UML de Componentes y de Paquetes. Esa ontología es el modelo ontológico de implantación del sistema.

C. Contenidos Digitales

1) *Definición de Contenidos Digitales:* Según el Ministerio de Tecnologías de la Información y las Comunicaciones (Min-TIC) de Colombia se define como: “Los contenidos digitales son cualquier forma de datos o información en forma digital (es decir, un archivo electrónico)”.

Cualquier archivo electrónico como: fotos, artes gráficas, documentos de investigación, artículos, informes, estadísticas, bases de datos, planes de negocios, diseños de ingeniería, e-libros, multimedia (música y películas), objetos de aprendizajes (OA), cursos en línea, entre otros, son considerados Contenidos Digitales.

La gran ventaja que suponen los contenidos digitales es la facilidad de acceso a multitud de información o aplicaciones, sin barreras geográficas y ahorro temporal.

Por otro lado, otro beneficio de este tipo de formato es que su edición no supone unos gastos elevados. Teniendo un ordenador, una impresora y los programas informáticos apropiados, se pueden sacar informes, fotos, composiciones musicales, etc.

2) *Clasificación de los Contenidos Digitales:* Los contenidos digitales pueden ser clasificados en una variedad de maneras, por ejemplo, [9] propone la atención a diversos preceptos, entre los que destaca el de la función que desempeñan los contenidos, traducida en criterios de interés de uso, aplicación, y de producción, de los que resultan las tres categorías siguientes:

- **Informativos:** Son contenidos compuesto por información de utilidad, que explican o aclaran desde una teoría hasta datos específicos y concretos, pasando por conceptos claves. Su marco de referencia es temático. Ejemplos son: Enciclopedias, tratados, informes, artículos, noticias.
- **Apoyo Educativo:** Es una gama que incluye información que favorece la incorporación de conocimientos, pero que no tiene implícitamente una labor de instrucción. Ejemplos son: Guías de estudio, manuales, apuntes, problemarios.
- **Tratamiento Educativo:** Son contenidos que median el acceso a la información para procurar un aprendizaje formativo o informativo con un lenguaje apropiado, según el usuario al que va dirigido. Ejemplos son: tutoriales, objetos de aprendizajes, cursos en línea.

Otra clasificación interesante de los contenidos digitales es según su distribución [10]: Audio, Imagen, Vídeo, Texto y Otros.

Ontology for Media Resources 1.0 de la W3C es un estándar ontológico para la representación de Contenidos Digitales [11]

(ver Figura 3), su objetivo es proveer un conjunto de propiedades estándares, tales como: locator para la URL del contenido, title para el título del contenido, description para la descripción del contenido, agent para la información del autor/editorial, entre otros más. Asimismo, también busca proporcionar la información básica necesaria para la interoperabilidad entre los distintos tipos de formatos de metadatos y contenedores de medios disponibles en la Internet.

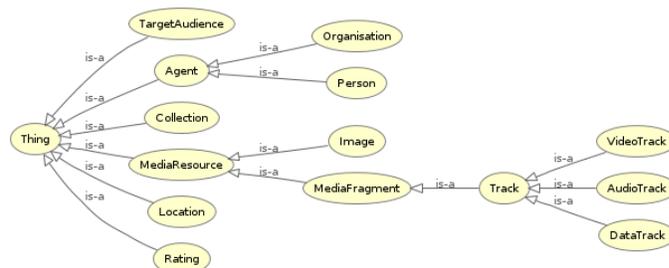


Figura 3: Ontología para Recursos de Medios de Comunicación

Las propiedades de Ontology for Media Resources se pueden dividir en las siguientes categorías: Información de identificación del recurso, Información sobre su creación, Descripción sobre el contenido, Información relacional, Derechos sobre el recurso, Información sobre su distribución, e Información técnica.

Esta ontología es especialmente útil cuando se requiere describir recursos multimedia publicados en la web que poseen distintos formatos de metadata. La especificación completa del vocabulario y los mapeos se puede encontrar en su página web [11].

Finalmente, un contenido digital de especial interés en este trabajo serán los MOOC, termino acuñado por Dave Cormier y Bryan Alexander en el año 2008. Según el artículo What is the theory that underpins our MOOCs? escrito por George Siemens, proviene del acrónimo en inglés *Massive Open Online Course*. El diccionario de Oxford lo define como: “un curso de estudio a disposición en Internet de forma gratuita a un número muy grande de personas”.

Algunas universidades de clase mundial, tales como Harvard y el MIT, ya están incursionando en los MOOC. MIT comenzó hace algunos años montando en línea y dando acceso gratuito al material de todos los cursos que ofrece. Desde hace unos años existe una alianza de las dos universidades para ofrecer varios MOOC, mediante la propuesta edX [12].

Las empresas MOOC Coursera y Udacity fueron creados por Daphne Koller, Andrew Ng, Sebastian Thrun y antiguos docentes, todos de la universidad de Stanford, en California [13].

III. DISEÑO DEL SISTEMA DE GESTIÓN DE CONTENIDOS DIGITALES

A. Arquitectura de Base del Proyecto Madre

El diseño de la arquitectura base del Proyecto Madre esta apoyada sobre dos pilares primordiales. Primero, el modelo

filosófico de las nubes descrito en el marco teórico, y segundo, la Plataforma Web esta basada en la Arquitectura Orientada a Servicio (SOA). En la Figura 4 se observan tres capas de gestión, donde cada capa administra una nube del modelo filosófico, la capa de Gestión Administrativa es la encargada de manejar la nube de Auto-Formación, la capa de Gestión Objetos Virtuales se encarga de la nube de Fuentes de Conocimiento, y por ultimo, la capa de Gestión de Paradigmas de Aprendizaje controla la nube de Paradigmas de Aprendizajes. Cada capa tiene sus propias estructuras de almacenamiento, que son ontologías específicas a cada dominio de cada capa, cuyas instancias representan la información guardada. A eso se llama en la Figura 4 Base de Datos Extendida, la cual, además, tiene la capacidad de soportar procesos de razonamiento sobre ella. En este trabajo se desarrolló la plataforma para la gestión de la nube de fuentes de conocimiento compuesta por Contenidos Digitales, que es parte de la segunda capa, el resto de capas fue, o está siendo desarrollada, en otros trabajos: la plataforma de gestión de la nube de auto-formación [14], la plataforma para la gestión de la nube de fuentes de conocimiento compuesta por objetos de aprendizaje [5], y la plataforma de gestión de la nube de paradigmas de aprendizajes [4].

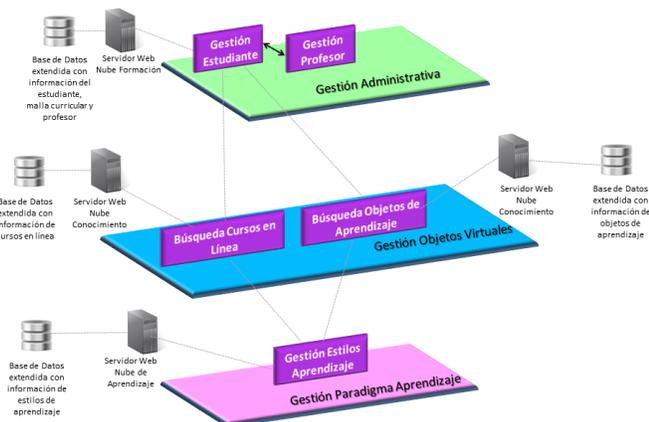


Figura 4: Arquitectura Funcional de la Plataforma Web [14]

Un ejemplo de un posible proceso de invocación entre las nubes se muestra en la Figura 5, para el caso de un estudiante que esté tomando un curso (modulo) de la malla curricular. Se inicia con la solicitud por parte de la nube de auto-formación (un estudiante ya registrado) a la nube de fuentes de conocimiento de un listado de objetos de aprendizaje o contenidos digitales para un tema en particular correspondiente al módulo y al perfil del aprendizaje del estudiante (paso 1). Ese listado debe ser compatible con la descripción del estudiante; para ello, la nube de fuentes de conocimiento realiza una consulta a la nube de paradigmas de aprendizajes de cuales herramientas, actividades, mecanismos de evaluación y tipo de obras, debe usar un estudiantes con ese perfil educativo (paso 2). Esa información es enviada por la nube de paradigmas de aprendizajes a la nube de auto-formación (paso 3). Una vez que se conoce el conjunto de características de los objetos virtuales a buscar, la nube de fuentes de conocimiento se

encarga de buscar (en su base de datos de objetos certificados o en internet) los objetos virtuales adecuados, y le entrega a la nube de auto-formación un listado ordenado por prioridad de los distintos objetos de aprendizaje o contenidos digitales asociados a ese estudiante en particular (paso 4).

Cada componente de las nubes puede estar instalado en uno o varios servidores web con el conjunto de servicios a proveer. El o los servicios se conectan a la información que ha sido almacenada en las bases de datos extendidas (u ontologías) respectivas, para poder responder a las solicitudes de otras nubes.

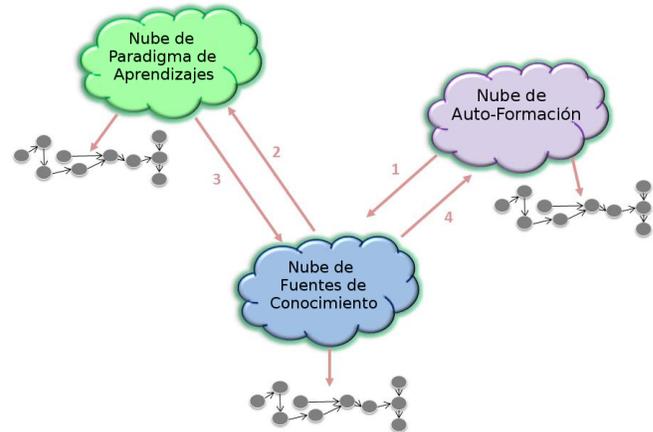


Figura 5: Proceso de Invocación entre las Nubes

B. Fase de Diseño

Esta fase está compuesta por el Modelo Independiente de la Computación (CIM) que recoge los requerimientos del sistema, y por el Modelo Independiente de la Plataforma (PIM) que detalla la arquitectura funcional, y los procesos y estructuras que lo componen.

1) *Conceptualización o Modelo Independiente de la Computación (CIM)*: En este modelo se comienza por definir los requisitos funcionales y no funcionales de la plataforma, y culmina con la ontología conceptual de la plataforma web.

Requisitos Funcionales

El conjunto de entradas, comportamientos y salidas que requiere el sistema, tanto para el cliente (sistemas externos) como para el especialista, que son los actores fundamentales, se describen a continuación:

- **Consultar:** Permite al estudiante y al especialista consultar la información de los contenidos digitales, según la necesidad que se tenga, por ejemplo, título, descripción, herramientas, actividades, entre otros. Asimismo, también debe permitir realizar consultas de los contenidos digitales que coincidan con un estilo de aprendizaje y tema específico.
- **Registrar:** Esta funcionalidad permite al especialista agregar los contenidos digitales, con la información que se tenga del contenido, por ejemplo, herramientas, actividades, instrumentos, título, entre otros.

- **Eliminar:** Permite al especialista borrar contenidos digitales que se consideren que no cumplen con las necesidades latentes.

Los procesos de consultar, registrar y eliminar en general son bastante sencillos, todos parten del mismo principio, con solo obtener el “id” del contenido digital y la necesidad específica. Hay un caso particular que es interesante describir con los diagramas de actividades, y es cuando se necesita consultar contenidos digitales para un tópico específico dependiendo del estilo de aprendizaje, para esto se cuenta con dos métodos: Consultar Interno y Consultar Externo.

Antes de describir el funcionamiento de los dos métodos, hay que explicar que es un contenido digital certificado y uno no certificado. Se dice que un contenido es certificado, si el mismo ha sido revisado y categorizado por un especialista, indicando que es un contenido apto para fortalecer o clarificar un determinado tema, y los contenidos que no pasan el proceso se considera que son contenidos digitales no certificados.

El método Consultar Interno tiene la finalidad de obtener información alojado en la ontología del sistema sobre contenidos públicos en la web, estos contenidos son certificados. En cambio, el método Consultar Externo obtiene la información sobre los contenidos de los buscadores en internet (por ejemplo, Coursera, Udacity, Miranda X, entre otros), y dichos contenidos son no certificados. La Figura 6 muestra de manera general el proceso de ambos métodos.

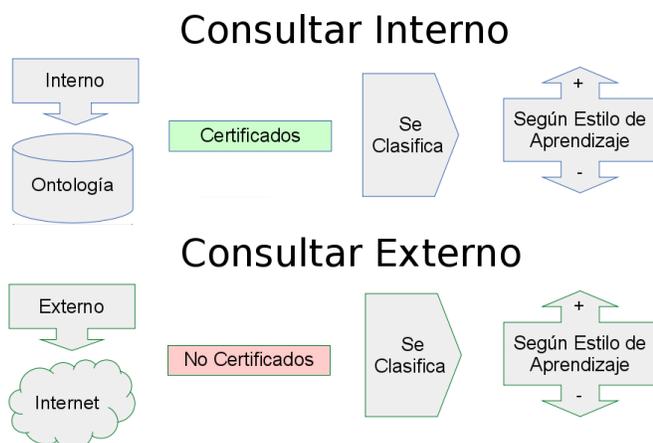


Figura 6: Métodos Consultar Interno y Externo

Es importante destacar que cada método es independiente uno del otro, es decir, el cliente puede invocar cualquiera de los dos según la necesidad que tenga. Sin embargo, la manera ideal es llamar al método Consultar Interno, y si este no retorna nada (no encontró contenidos) se llama al método Consultar Externo. El sistema no realiza las dos llamadas automáticamente, la razón de esto es que los dos métodos devuelven los contenidos de distinta forma. Por ejemplo, Consultar Interno devuelve una lista de id de los contenidos digitales, y luego con ese id se puede solicitar la información que se requiera de ese contenido, ya que está almacenado en la ontología. En cambio, Consultar Externo devuelve una lista de los contenidos con toda la información encontrada en formato de intercambio estándar JSON (JavaScript Object Notation).

Requisitos No Funcionales

Entre los requisitos no funcionales que el sistema debe cumplir, se encuentran los siguientes: Escalabilidad, Usabilidad, Confiabilidad, Seguridad e Interoperabilidad.

Ontología Conceptual

En la Figura 7 se observa la ontología de la capa CIM, dicha ontología muestra las relaciones entre los diferentes conceptos vinculados a la plataforma web y sus requerimientos.

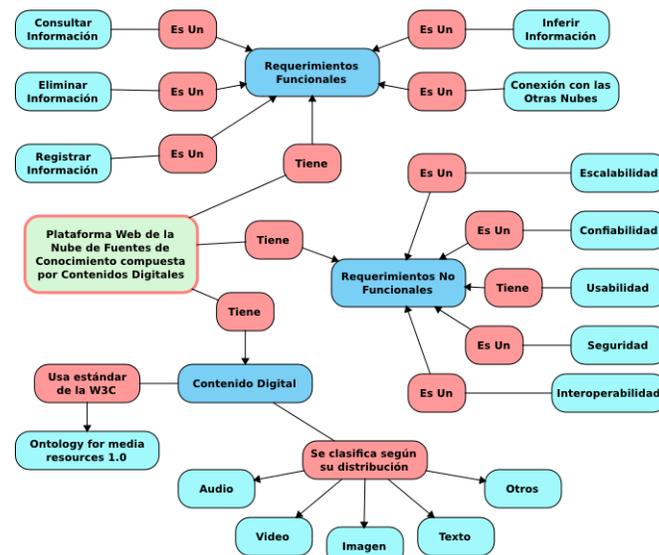


Figura 7: Ontología de Conceptualización de la Capa CIM de la Plataforma Web

Los contenidos digitales usan el estándar de la W3C Ontology for media resources 1.0 (ver Figura 7), para la base de datos que almacena la información sobre los contenidos digitales, más conceptos nuevos necesarios para ofrecer la capacidad de adaptar el contenido a un estilo de aprendizaje de un estudiante específico.

2) *Especificación o Modelo Independiente de la Plataforma (PIM):* En este modelo se describe la arquitectura funcional, sin considerar los detalles necesarios para su implementación en una plataforma determinada, acompañado de los procesos y estructuras que lo componen, para finalmente generar la ontología de diseño del sistema.

Arquitectura Funcional

La arquitectura funcional para el sistema es SOA conformada por operaciones o métodos que están disponibles en los servicios web del sistema, los cuales se muestran a continuación agrupados por funcionalidad:

- **Métodos de Consultar:** título, localizador, certificación, descripción, categoría, herramientas, instrumentos, actividades, etiquetas, autores, nombreAutor, localizadorAutor, Editoriales, nombreEditorial, localizadorEditorial, consultarInterno (consultar) y consultarExterno.
- **Métodos de Registrar:** títuloAdd, localizadorAdd, certificaciónAdd, descripciónAdd, categoríaAdd, herramientasAdd, instrumentosAdd, actividadesAdd, etique-

tasAdd, autoresAdd, nombreAutorAdd, localizadorAutorAdd, EditorialesAdd, nombreEditorialAdd, localizadorEditorialAdd.

- **Métodos de Eliminar:** eliminar.

En la Figura 8 se muestra el diagrama de clases general de la plataforma. La plataforma para la gestión de la nube de fuentes de conocimiento compuesta por contenidos digitales esta conformado por las siguientes clases:

- **ContenidoDigital:** clase principal del servicio que exhibe la estructura de los métodos.
- **OntologíaContenidoDigital:** enlace con el servicio web NubeAprendizajeWS y las clases ContenidoDigital, Ontología y Razonador.
- **NubeAprendizajeWS:** servicio responsable de entregar las herramientas, actividades e instrumentos para un estilo de aprendizaje específico.
- **Ontología:** encargado de manejar la estructura de la base de datos ontológica.
- **Razonador:** encargado de realizar las inferencias sobre la base de datos ontológica.
- **FContenidoDigital:** servicio que permite limitar las funcionalidades de la Nube de Conocimiento a solo los métodos disponible para los clientes, brindando mayor sencillez y seguridad.
- **Cliente y Especialista:** son las interfaces que consumen la información. Estas interfaces son servicios web, gracias a este tipo de arquitectura las interfaces pueden ser mejoradas o sustituidas sin afectar el funcionamiento de la plataforma.

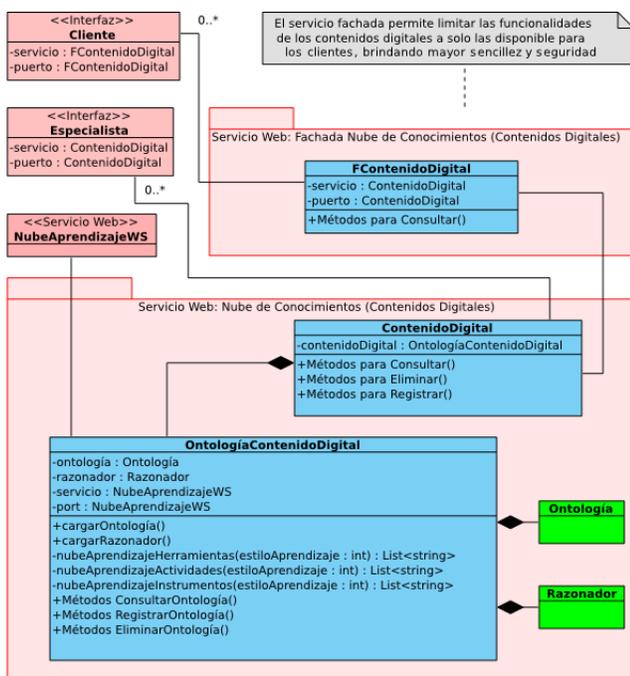


Figura 8: Diagrama de Clases de la Plataforma

Asimismo, en esta capa se deben especificar las clases propias que caracterizan la arquitectura orientada a servicios del sistema.

Ontología de Diseño

En la Figura 9 se observan las funcionalidades de la plataforma web, descritas como servicios web.

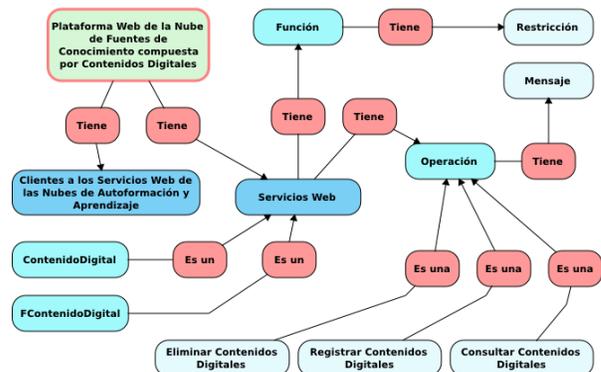


Figura 9: Ontología de Diseño de la Capa PIM de la Plataforma Web

IV. IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN DE CONTENIDOS DIGITALES

A. Arquitectura General del Sistema Web

La arquitectura del sistema se describe en la Figura 10, en ella se muestra una capa llamada servidor, contiene todos los proveedores de servicios web de cada una de las nubes, estos servicios web han sido desarrollados bajo el uso de la interfaz de programación de aplicaciones de Java (JAX-WS) (Sun Microsystems). Cada uno de estos servicios necesita conectarse a una base de datos extendida (ontología) que gestiona la información de cada nube. Esta conexión entre los servicios web en Java y las ontologías en protégé se realiza a través del OWL API, este gestiona la información haciendo uso del razonador Fact++ para el lenguaje OWL (W3C). Fact++ es un razonador de descripción lógica del tipo OWL DL. Este razonador permite validar la consistencia de las ontologías, así como hacer inferencia sobre la información almacenada.

La capa cliente aloja todas las referencias a los servicios web de la plataforma educativa, la cual permite utilizar la información y conocimiento almacenado, además de que provee la interfaz de usuario del sistema. La comunicación entre las capas cliente y servidor se hace a través de un bus de servicios, que maneja los mensajes de enlace entre las 2 capas.

B. Realización o Modelo Específico de la Plataforma (PSM)

1) *Diagrama de Componente:* La Figura 11 muestra el diagrama de componente de la plataforma, conformado por tres grandes grupos de nodos. El primer grupo lo integran los nodos que consumen los recursos ofrecidos por la nube de fuentes de conocimientos, y esta compuesta por:

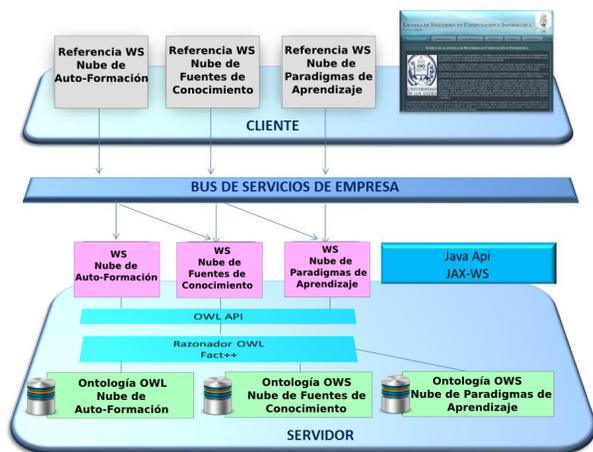


Figura 10: Arquitectura de la Plataforma Web

- **Interfaz Especialista:** responsable de registrar los contenidos digitales y probar cada funcionalidad disponible en la nube de conocimiento compuesta por contenidos digitales. Este nodo posee dos paquetes importantes: el ClienteContenidoDigital.war, provee el servicio web con las paginas (coursera.jsp y generica.jsp) disponibles para el especialista, y las paginas para las pruebas de integración del sistema; El otro paquete, es el ClienteContenidoDigitalJUnitTest.java con las pruebas unitarias automatizadas del sistema.
- **Nube de Auto-Formación:** solicita contenidos digitales específicos para el estilo de aprendizaje que tiene un estudiante particular en un tema determinado.
- **Módulo Adaptativo para el Estilo de Aprendizaje (en desarrollo):** solicita las herramientas, instrumentos y actividades de un contenido digital, para verificar y actualizar el estilo de aprendizaje asociado a un estudiante.

El segundo grupo representa las funcionalidades de la nube de fuentes de conocimiento para los contenidos digitales, constituidos por los dos nodos siguientes:

- **FContenidoDigital:** actúa como capa de seguridad, limitando los métodos disponibles en ContenidoDigital a solo los indicados en este servicio, funcionando como fachada entre el servicio de contenidos digitales y los clientes.
- **ContenidoDigital:** es el núcleo de la plataforma de contenidos digitales, está conformado por el ContenidoDigital.java, que contiene la estructura de los métodos disponibles en el servicio; OntologiaContenidoDigital.java, encargado de todo el manejo de la ontología específica del contenido digital; owlapi-distribution.jar, librería con capacidades para el manejo de la ontología; jfact.jar, razonador encargado de manejar las consultas semánticas para la ontología; y json-simple.jar, librería que permite la codificación y decodificación de textos de tipo JSON (formato ligero para el intercambio de datos).

El último grupo lo componen tres nodos muy importantes para

el funcionamiento de la nube de fuentes de conocimiento, ya que estos nodos contienen y entregan toda la información necesaria sobre los contenidos digitales. Cada nodo es detallado a continuación:

- **ContenidoDigital.owl:** ontología con la información y estructura de los contenidos digitales certificados por el especialista.
- **Repositorios (Internet):** son contenidos digitales buscados en la internet para ser ofrecidos por la nube de conocimientos como contenidos no certificados.
- **Nube de Paradigma de Aprendizajes:** servicio que permite generar listas de herramientas, instrumentos y actividades asociados a un estilo de aprendizaje.

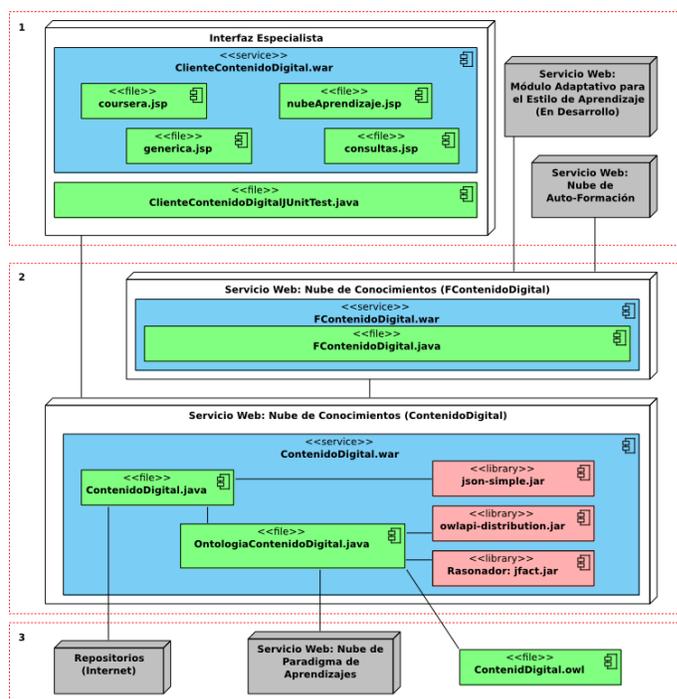


Figura 11: Diagrama de Componentes de la Plataforma

2) **Ontología de Desarrollo:** En la Figura 12 se observan las distintas tecnologías utilizadas para construir y hacer funcionar la plataforma encargada de prestar los servicios relacionados con la nube de conocimientos compuestas de contenidos digitales.

V. FUNCIONAMIENTO Y PRUEBAS DEL SISTEMA

A. Prueba de Integración con la Nube de Paradigmas de Aprendizaje

Esta prueba permite comprobar la interacción entre el servicio web que provee contenidos digitales y la nube de paradigmas de aprendizaje, la cual se realiza consultando al método nubeAprendizaje disponible en el servicio web de los contenidos digitales. El método solo recibe como parámetro el estilo de aprendizaje, como se observa en la Figura 13, y este método invoca a los cuatro métodos que provee el servicio

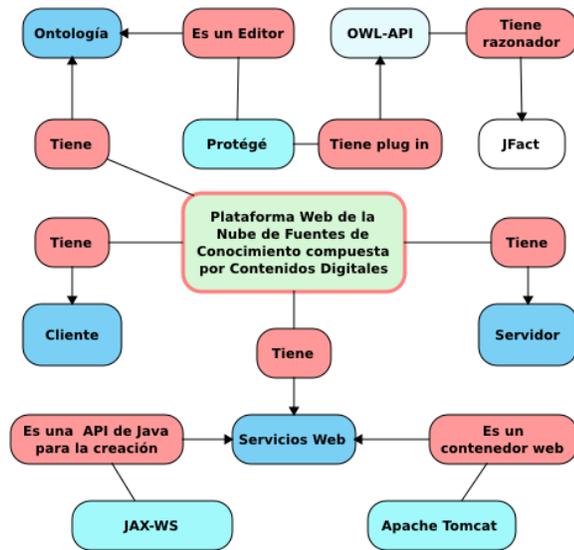


Figura 12: Ontología de Desarrollo de la Capa PSM de la Plataforma

de paradigma de aprendizajes, que son los siguientes: verParadigma, verActividades, verHerramientas y verInstrumentos. El resultado que se obtiene de la prueba es toda la información relacionada con un estilo de aprendizajes.

```

List<String> nubeAprendizaje(int estiloAprendizaje) throws Exception {
    List<String> listaAprendizaje = new ArrayList<String>();

    NubeAprendizajeWS_Service service = new NubeAprendizajeWS_Service();
    NubeAprendizajeWS_port = service.getNubeAprendizajeWSPort();

    listaAprendizaje.add(port.verParadigma(estiloAprendizaje));
    listaAprendizaje.add(port.verActividades(estiloAprendizaje));
    listaAprendizaje.add(port.verHerramientas(estiloAprendizaje));
    listaAprendizaje.add(port.verInstrumentos(estiloAprendizaje));

    return listaAprendizaje;
}
    
```

Figura 13: Método nubeAprendizaje del Servicio Web del Sistema

La Figura 14 muestra el resultado de la consulta al método nubeAprendizaje, pasando como parámetro el estilo de aprendizaje 4534, que corresponde a un Paradigma Activo. Como se lee en la Figura 14, este retornar el paradigma Activo con sus herramientas, actividades e instrumentos, quedando adicionalmente verificada la comunicación entre los componentes del sistema.

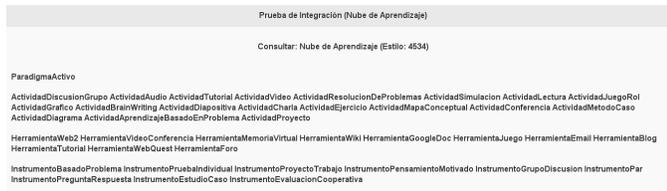


Figura 14: Prueba de Integración con la Nube de Paradigmas de Aprendizaje

B. Prueba de Integración con la Capa de Seguridad

El objetivo de esta prueba es verificar la integración de la capa de seguridad que provee el servicio web FContenidoDigital

al servicio web ContenidoDigital de la nube de fuentes de conocimiento. El proceso que se sigue es invocar los métodos disponibles en FContenidoDigital, que solo contiene métodos de consulta, y este a su vez, invocan los métodos del servicio web ContenidoDigital correspondientes. El resultado que se debe obtener es poder consultar los contenidos digitales de la nube de fuentes de conocimiento. En la Figura 15 se observa como se consume el servicio web FContenidoDigital, y la invocación del método consultar con los parámetros de pruebas: estilo, tópico y tema.

```

fcontenidodigital.FContenidoDigital_Service service = new fcontenidodigital.FContenidoDigital_Service();
fcontenidodigital.FContenidoDigital_port = service.getFContenidoDigitalPort();

out.println("<center><br><b>Consultar: Contenido Digital</b></center>");

int estilo=4534;
topico="Fundamentos de Programación 1";
tema="Algoritmos y Resolución de Problemas";

List<String> listaContenidosDigitales = port.consultar(estilo,topico, tema);
    
```

Figura 15: Invocación del Servicio Web FContenidoDigital

El resultado final de toda la invocación se muestra en la Figura 16, donde se detalla la información como: Titulo, URL, Descripción, Herramientas, etc; asociado a uno de los contenidos digitales encontrados con el método consultar.

Contenido Digital (Limitada)!



Figura 16: Resultado de la Prueba al Servicio FContenidoDigital

C. Prueba de Integración con la Interfaz del Especialista

Esta prueba es de suma importancia para el sistema, porque verifica la comunicación desde la interfaz del especialista al servicio de contenidos digitales, y también se verifica la capacidad de búsqueda del sistema de la nube de fuentes de contenidos digitales, tanto para consultas internas como consultas externas. Ambas consultas necesitan tres parámetros para realizar la búsqueda, estos parámetros son: estilo, tópico y tema. La interfaz también posee validaciones (ver Figura 17) que permiten minimizar los errores a la hora de realizar dichas consultas, donde todos los parámetros son obligatorios y el estilo debe ser un número de cuatro dígitos sin espacio.

Los valores de pruebas para el caso de la consulta interna son los mismos que se han estado usando en anteriores pruebas, como se puede observar en la Figura 18. La consulta externa se usa como tópico "Inteligencia Artificial", y tema "Planificación". El resultado que se debe obtener para cada caso es un listado de contenidos digitales con su información asociada.

Figura 17: Validaciones de los Campos para las Consultas

Figura 18: Resultados de las Consultas

En la Figura 18 se puede apreciar el resultado de cada consulta, la cual es mostrado en la parte inferior de las misma. El resultado de la consulta interna se estudiará en la prueba de integración con el Proyecto Madre. En el caso de la consulta externa, al revisar la información de cada contenido, se pudo apreciar que los contenidos que están al principio guardan más relación con lo buscado, y a medida que se va llegando al final se va alejando del resultado óptimo. Hay que resaltar que la calidad de esta búsqueda depende del buscador externo que contiene esta fuente de contenidos, y por dicho motivo, son considerados como contenidos digitales no certificados.

D. Prueba de Integración con la Plataforma Madre

El sistema de contenidos digitales, a través de la nube de fuentes de conocimiento, se integra al proyecto madre como un nuevo servicio que aporta recursos a la nube de auto-formación. En la Figura 19 se observa la interfaz de la nube de auto-formación, donde un estudiante consulta los módulos inscritos y tiene la opción “Ver Cursos en Línea”, para solicitar contenidos digitales relacionados tanto con su estilo de aprendizaje como con el módulo inscrito.

Figura 19: Interfaz de la Nube de Auto-Formación

El objetivo de esta prueba es comprobar que la nube de auto-formación consume la información contenida en el servicio web de los contenidos digitales a través del servicio FContenidoDigital, que brinda la seguridad de limitar los métodos a

solo los necesarios para este nivel de cliente. Para realizar esta prueba se usa un estudiante con estilo de aprendizaje 4534, que corresponde al Paradigma Activo, la Figura 14 detalla las características de este paradigma. Otro dato importante a considerar en la prueba es el tópico y tema a consultar desde la nube de auto-formación, el tópico a usar es “Fundamentos de Programación 1”, y el tema es “Algoritmos y Resolución de Problemas”. La respuesta que se debe obtener es un listado de contenidos digitales adecuados a estas necesidades.

En la Figura 20 se observa el resultado de la consulta desde la nube de auto-formación a los contenidos digitales. El proceso que sigue la consulta comienza en la invocación del servicio web desde la nube de auto-formación del proyecto madre, luego el servicio web FContenidoDigital recibe la solicitud y hace un llamado al servicio web que busca los contenidos, allí se hace la búsqueda según su tema y tópico, una vez encontrado los contenidos se invoca a la nube de aprendizaje con el estilo de aprendizaje, con esa información se procede a organizar y clasificar los contenidos, y finalmente, retorna toda la información al estudiante.

Figura 20: Interfaz de la Nube de Auto-Formación Consultando Contenidos Digitales

Para determinar si la respuesta de la prueba es correcta, los contenidos deben coincidir con algunas de las propiedades (actividades, herramientas e instrumentos) del paradigma Activo, y que estos contenidos deben estar ordenados de mayor a menor coincidencia. Se estudiaron los contenidos, donde se pudo constatar que el primer contenido coincide con siete de estas propiedades, y que va disminuyendo las coincidencias hasta llegar al último contenido con cuatro coincidencias. La otra característica a considerar es que los contenidos deben estar relacionados con el tópico y tema pedido, para esto se verificó el resumen de cada contenido, donde se apreció que el contenido si guarda relación con lo buscado.

VI. COMPARACIÓN CON OTROS TRABAJOS

La comparación de la calidad del sistema buscador de contenidos digitales, con respecto a otros sistemas que posean características similares de búsquedas de contenidos alojados en la Internet. A continuación se muestra la Tabla I considerando los siguientes criterios:

Tabla I: Tabla Comparativa de Sistemas Desarrollados

Sistemas / Criterios	Euro-peana [15]	BDU2 [16]	DigitalNZ Search [17]	Este trabajo
Filtros para la Búsqueda	Tipo de archivo, idioma, fecha, país, copyright y proveedor	Tipo de archivo, autores, títulos y materias	Tipo de archivo, proveedor, fecha y copyright	Estilo de aprendizaje
Gestión de Información	Ontología y búsquedas semánticas	Ontología y búsquedas semánticas	Ontología y búsquedas semánticas	Ontología y búsquedas semánticas
Aprendizaje	No Posee	No Posee	No Posee	No implementado actualmente
Interfaz	Página Web, API, Widget	Página Web	Página Web, API	Página Web, Servicio Web
Tipos de Archivos	Texto, imagen, vídeos, sonidos y 3d	Cualquier tipo de archivo alojado en los repositorios	Cualquier tipo de archivo alojado en los repositorios	Cualquier tipo de contenido alojado en la internet
Interoperabilidad	JavaScript Object Notation (JSON)	Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)	Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), XML Sitemap, API y RSS feed	JavaScript Object Notation (JSON)
Adaptabilidad	IND.	IND.	IND.	Permite agregar nuevos filtros de búsquedas y nuevas clasificaciones en los contenidos
Escalabilidad	Permite agregar nuevas fuentes de búsquedas	Permite agregar nuevos repositorios	Permite agregar nuevas fuentes de búsquedas	Permite agregar nuevas fuentes de búsquedas y nuevas características en los contenidos

- **Filtros para la Búsqueda:** Capacidad del sistema para ajustar los resultados de los contenidos digitales ofrecidos.
- **Gestión de Información:** Capacidad del sistema centrada en los procesos relacionados con el almacenamiento, el tratamiento y difusión del conocimiento explícito, que se encuentra representado en la metadata de los contenidos digitales.
- **Aprendizaje:** Capacidad del sistema para aprender.
- **Interfaz:** Capacidad del sistema para ofrecer distintas interfaces.
- **Tipos de Archivos:** Capacidad del sistema para clasificar los distintos contenidos digitales.
- **Interoperabilidad:** Capacidad del sistema para comunicarse con otros sistemas.
- **Adaptabilidad:** Capacidad del sistema para ajustarse a los distintos cambios del entorno.
- **Escalabilidad:** Capacidad del sistema para crecer sin perder calidad en los servicios ofrecidos.

Los tres sistemas usados para la comparación se basan en la gestión de los metadatos y no en el almacenaje de los contenidos digitales, ya que es la característica principal en el sistema buscador de contenidos desarrollado. El primero es Europea [15], es la biblioteca digital europea, de acceso libre, que reúne contribuciones ya digitalizadas de reconocidas instituciones culturales de los 28 estados miembros de la

Unión Europea. El segundo es BDU2 [16], es un proyecto iniciado por el SIU (Sistema de Información Universitario) de Argentina, que permite reunir recursos de información de valor académico de libre disponibilidad para el usuario final. El último es DigitalNZ Search [17], es una iniciativa con cerca de 200 socios, liderados por la Biblioteca Nacional de Nueva Zelanda, que buscan fomentar la reutilización de los contenidos digitales.

En cuanto al criterio del filtro de búsqueda, la diferencia con este trabajo, es porque usan filtros muy típicos como: tipo de archivo, título, etc. En cambio, el sistema buscador de contenido utiliza un filtrado basado en el estilo de aprendizaje, lo que permite entregar contenidos más adaptados a las capacidades de aprendizaje particulares de una persona.

En relación con el aprendizaje, ninguno de los sistemas poseen mecanismos que permita aprender, aunque la ontología de este trabajo ya posee conceptos que permite en una futura actualización incluirlo, por ejemplo, el concepto "Índice", que permitiría al cliente valorar los contenidos entregados en las búsquedas, permitiendo aumentar o disminuir su prioridad en próximas búsquedas de los clientes en el sistema.

En el resto de los criterios muy semejantes las características de los sistemas. En gestión de información, todos usan ontologías para almacenar los metadatos, y realizar búsquedas semánticas. En interfaz, todos usan la página web como principal entorno gráfico, y se apoyan en otras interfaces para permitir la comunicación con otros sistemas. En tipos de archivos, como los sistemas manejan metadatos pueden utilizar cualquier tipo de contenido que puedan referenciar. En inter-

operabilidad, todos implementan mecanismos para compartir los metadatos de los contenidos digitales con otros sistemas. En adaptabilidad, no se logró determinar las capacidades adaptativas de los sistemas a comparar. En escalabilidad, todos permiten incluir nuevas fuentes de búsquedas y/o repositorios.

VII. CONCLUSIÓN

La presente investigación logró desarrollar el sistema buscador de contenidos digitales usando el paradigma ODA, con este propósito se especificaron las capas CIM, PIM y PSM, apoyado en los diversos diagramas UML, culminando en cada capa con su ontología específica. El sistema se desarrolló como servicio web, permitiendo la interoperabilidad entre los distintos sistemas que conforman la plataforma del Proyecto Madre, y para el manejo de la ontología y procesos de razonamiento se usó OWL-API y JFact, respectivamente. Por último, el estándar Ontology for Media Resource permitió manejar la información sobre las distintas características de los contenidos digitales, aunado a esto se amplió el estándar con nuevas características particulares, para ofrecer los contenidos según los distintos paradigma de aprendizajes.

Este trabajo es innovador, ya que provee toda una plataforma para la búsqueda de contenidos más adecuados a cada estilo de aprendizaje que tenga un estudiante, entregando los contenidos priorizados según la mayor coincidencia con su estilo, logrando apoyarse en su proceso de auto-aprendizaje. También, el sistema permite su uso en otras plataforma educativas, gracias a que esta implementado en un servicio web, siempre y cuando se respete los parámetros de búsqueda del mismo.

Como trabajos futuros se propone agregar capacidades para la carga de información de contenidos publicados a través de distintos metadatos multimedia, como por ejemplo, los formatos OGG, LOM (Learning Object Metadata), entre otros. También se propone explotar conceptos aun no usados en la ontología, como Audiencia e Índice, que permitan refinar más las búsquedas. Asimismo, se propone experimentar con la capacidad de la ontología para almacenar información de las partes de un contenido digital, esto permitiría ofrecer solo las partes de un contenido que más se adapten a un paradigma específico.

REFERENCIAS

- [1] J. Delors, *La Educación o la Utopía Necesaria*, Informe a la UNESCO de la Comisión Internacional sobre la Educación. La Educación Encierra un Tesoro, pp. 13–36, 1996. http://www.unesco.org/education/pdf/DELORS_S.PDF.
- [2] J. Aguilar, D. Gutiérrez y D. Hernández, *Proyecto Madre*, Informe Técnico, Departamento de Ciencias Computacionales, Universidad de Los Andes, Mérida, 2012.
- [3] R. Shackelford, A. McGettrick, R. Sloan, H. Topi, G. Davies, R. Kamali, J. Cross, J. Impagliazzo, R. LeBlanc y B. Lunt, *Computing Curricula 2005: The Overview Report*, ACM SIGCSE Bulletin, vol. 38, no. 1, pp. 456–457, 2006.
- [4] J. Fuentes, *Implementar un Prototipo de un Sistema de Gestión de la Nube de Aprendizaje Usando el Paradigma ODA*, Universidad de Los Andes, Mérida, Venezuela, 2014.
- [5] O. Portilla, *Plataforma Basada en SOA Usando el Paradigma ODA (Ontology Driven Architecture)*, para la Gestión de una Nube de Fuentes de Conocimiento Compuesta por Objetos de Aprendizaje, Universidad de Los Andes, Mérida, Venezuela, 2014.
- [6] OMG, *MDA Guide, Version 1.0.1*, Object Management Group, pp. 62, 2003.
- [7] P. Tetlow, J. Pan, D. Oberle, E. Wallace, M. Uschold y E. Kendall, *Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering*, W3C Working Draft, Febrero 2005. <http://www.w3.org/2001/sw/BestPractices/SE/ODA>.
- [8] J. Aguilar y K. Moreno, *Metodología para el Desarrollo de Aplicaciones Basado en el Paradigma ODA*, CEMICID, Departamento de Computación, Universidad de Los Andes, Mérida, Venezuela, 2014.
- [9] O. Nájera, *Tipología de Contenido Digital*, Abril 2004. <http://portal.educ.ar/debates/educacionyitic/inclusion-digital/tipologia-de-contenido-digital.php>.
- [10] F. Gastaminza, *Documentos Digitales: Hipertexto y Edición Digital*, Universidad Complutense de Madrid, 2005. <http://pendientedemigracion.ucm.es/info/multidoc/prof/fvalle/tema20.htm>.
- [11] W. Lee, T. Bürger, F. Sasaki, V. Malaisé, F. Stegmaier y J. Söderberg, *Ontology for Media Resource 1.0*, W3C Working Draft, Febrero 2012. <http://www.w3.org/TR/mediaont-10>.
- [12] *We're Empowering Learning in the Classroom and Around the Globe*, Edx inc, 2015, <https://www.edx.org/about-us>.
- [13] F. Wright, *What do Librarians Need to Know About MOOCs?*, D-Lib Magazine, vol. 19, no. 3/4, Marzo/Abril 2013, <http://www.dlib.org/dlib/march13/wright/03wright.print.html>.
- [14] K. Moreno, *Prototipo de Plataforma Web para la Gestión de la Nube de Auto-Formación Basado en Ontologías*, Universidad de Los Andes, Mérida, Venezuela, 2014.
- [15] Europeana, 2015, <http://www.europeana.eu>.
- [16] BDU, 2015, <http://bdu.siu.edu.ar/cgi-bin/inicio.pl>.
- [17] DigitalNZ, Biblioteca Nacional de Nueva Zelanda, 2015. <http://www.digitalnz.org>.

Un Mecanismo de Respuestas a Consultas en Presencia de Nulos

Josué Ramírez¹, Leonid Tineo¹
ramirezjosue@usb.ve, leonid@usb.ve

¹Departamento de Computación y Tecnología de la Información, Universidad Simón Bolívar, Caracas, Venezuela

Resumen: En cualquier descripción del mundo real se pueden distinguir modelos de tratamiento para la imprecisión a nivel de los atributos: valores probabilísticos, valores difusos, distancias, solidez y completitud, y valores nulos. Varios trabajos previos centran sus esfuerzos en dar un tratamiento adecuado a los valores nulos. Algunos de ellos los ignoran bajo el argumento de no tener suficiente significancia estadística. Otros usan técnicas de reescritura tratando de evitar, en lo posible, condiciones que involucran valores nulos. Una estrategia muy conocida es la llamada "imputación" que consiste en cambiar los valores nulos en la base de datos por valores sustitutos elegidos con algún tipo de criterio establecido. Usando técnicas de inferencia es posible hacer estimación de valores en lugar de los nulos. Cualquiera de estos mecanismos tiene sus desventajas como pueden ser muy poca aplicabilidad práctica, inviabilidad de aplicación o introducción de ruido. Específicamente el presente trabajo plantea la aplicación de un método en bases de datos relacionales, en donde no se realiza la imputación, sino una estimación en forma dinámica en el momento de ejecutar una consulta que involucre un atributo que tenga valores nulos. De esta manera se evitan las desventajas antes descritas. Por otro lado, para la estimación del valor que será usado en la consulta se plantea el uso de reglas de asociación difusa, técnica ésta que permite obtener un conjunto de reglas para estimar los valores nulos en función de los demás valores presentes en los registros y cuya aplicación se extiende a atributos con valores cuantitativos y no sólo categóricos.

Palabras Clave: Valores Nulos; Reglas Asociación Difusas; Procedimientos Almacenados; Bases de Datos Incompletas; Consultas Relacionales.

Abstract: In any description of the real world can be distinguished treatment models imprecision level attributes: probabilistic values, fuzzy values, distances, soundness and completeness, and nulls. Several previous studies focus its efforts on providing adequate treatment for null values. Some of them ignore them on the grounds of not having enough statistical significance. Others use rewriting techniques trying to avoid, if possible, conditions involving null values. A well-known strategy is called "imputation" that is to change nulls in the database by substitutes chosen values with some sort of criteria established values. Using inference techniques it is possible to estimate values instead of zero. Any of these mechanisms has its disadvantages such as very little practical applicability, infeasibility of application or introduction of noise. Specifically, this paper presents the application of a method in relational databases, where the imputation is not made, but an estimate dynamically when running a query involving an attribute that has nulls. In this way the disadvantages described above are avoided. Moreover, for estimating the value to be used in the query it raises fuzzy association rules, this technique is for obtaining a set of rules for estimating zero depending on other values in the registers and whose application extends to attributes with quantitative values and not just categorical.

Keywords: Null Values; Fuzzy Association Rules; Store Procedures; Incomplete Data Bases; Relational Queries.

I. INTRODUCCIÓN

Las bases de datos incompletas constituyen un problema común en muchos contextos. Especialmente en las bases de datos relacionales se modela la incompletitud a través de la pseudodescripción denominada nulo [1]. Este valor tiene

asociada diversas interpretaciones y se han realizado diversos esfuerzos o trabajos para resolver el tratamiento del mismo.

Algunos trabajos proponen extensiones teóricas al modelo relacional con la intención de aceptar la existencia del nulo como estrategia para el manejo de la imprecisión [2][3][4].

Sin embargo, estos trabajos no han sido de mucha aplicación práctica.

También se ha propuesto ignorar los nulos, lo cual resulta de mayor sencillez, pero tampoco es de mucha utilidad práctica, pues su aplicabilidad está muy limitada. Para ignorar la presencia de valores nulos, se requiere que la proporción de los mismos no sea significativa estadísticamente, condición que no siempre se cumple [5].

Otros desarrollos consideran las estrategias para estimar los nulos y existen varias propuestas en la literatura. Algunas suponen el cumplimiento de ciertas restricciones que deben satisfacer los datos para garantizar buenos resultados [6][7][8]. Otras técnicas que han resultado efectivas resultan complejas de implementar o se usan en ámbitos muy específicos o proveen soluciones difíciles de implementar o auditar [9][10][11][12][13][14].

En cuanto a la sustitución del valor nulo por una estimación, lo cual se conoce como imputación, no es fácil seleccionar la técnica correcta porque esto depende de múltiples factores y además no existen reglas específicas que guíen la escogencia o bien una vez que ésta se realiza es importante también aplicar correctamente la técnica para evitar resultados erróneos [5][15][16].

En este artículo se propone un mecanismo automatizado que permite dar respuesta a consultas en base de datos relacionales en presencia de nulos, mediante estimación dinámica, subsanando las dificultades mencionadas de las estrategias y técnicas antes descritas.

En la sección II se describen las áreas de investigación relacionadas con nuestra propuesta. En la sección III se detalla la solución propuesta. En la sección IV se explican los experimentos y resultados más relevantes del sistema desarrollado. Por último, se presentan las conclusiones del artículo y las líneas de trabajo futuras.

II. BASES TEORÍCAS

A continuación se describen las áreas de investigación que sirvieron de base para la propuesta planteada en este artículo.

A. Reglas de Asociación

El modelo de reglas de asociación es muy conocido en su versión clásica para la obtención de patrones nuevos y significativos en minería de datos. Según [17], el modelo clásico de reglas de asociación concibe que el ítem es el objeto básico de interés. Un conjunto de ítems se conoce como *itemset*. Por otro lado una transacción es un hecho compuesto que agrupa un conjunto de ítems. Intuitivamente, las reglas son asociaciones que vinculan la presencia de ítems dentro de transacciones.

Más formalmente se puede definir una regla de asociación de la siguiente manera: Sea I un conjunto de ítems (*itemset*) y T un conjunto de transacciones con ítems en I , ambos conjuntos se asumen finitos. Una regla de asociación es $A \cap C = \emptyset$ una expresión de la forma $A \Rightarrow C$, donde $A, C \subseteq I$, $A, C \neq \emptyset$ y Esta regla significa que cada transacción de T que contiene A también contiene C .

Se consideran dos medidas de interés para evaluar las reglas de asociación: el soporte y la confianza, las cuales se basan en el concepto de soporte de un *itemset*, que se interpreta como la proporción de ocurrencia conjunta de uno o más ítems en una transacción. Las técnicas de minería procuran descubrir (minar) reglas fuertes, esto es reglas con soporte y confianza mayores que los umbrales definidos por el usuario. Sin embargo, a fin de disminuir el número de reglas minadas y que éstas sean más interesantes en [17] se propuso usar el factor de certeza. El factor de certeza es la medida de la variación de la probabilidad de que C esté en una transacción cuando se consideran sólo las transacciones donde se encuentra A .

Los valores del factor de certeza están entre $[-1,1]$. Su valor es positivo si la asociación entre A y C es positiva, es 0 cuando son independientes, y es negativo si están asociados negativamente. Específicamente, un factor de certeza positivo mide el grado de incremento en la probabilidad de que C se encuentre en una transacción, dado que A se encuentra. Una interpretación similar puede realizarse para valores negativos del factor de certeza [18].

El uso de las reglas de asociación se ha extendido al realizarse adaptaciones al algoritmo básico propuesto por Agrawal para poder buscar reglas en bases de datos incompletas [19]. Las reglas de asociación se han usado en las bases de datos relacionales, en donde incluso se han propuesto varias arquitecturas que incorporan la minería de reglas de asociación. Además, entre otras aplicaciones se han usado para la estimación de valores nulos [4][20].

B. Reglas de Asociación Difusa

En un esfuerzo para obtener patrones más significativos y subsanar deficiencias ante la incertidumbre que suele presentarse en los datos, el modelo de reglas de asociación se extendió mediante aplicación de teoría de conjuntos difusos, surgiendo las reglas de asociación difusa [18].

Una transacción difusa en el modelo de reglas de asociación difusa [18] es definido como conjunto difuso. Esto es, un conjunto en que la membresía de sus elementos es gradual en el intervalo real $[0,1]$. Los conjuntos difusos permiten dar una interpretación numérica de términos vagos del lenguaje natural denominados etiquetas lingüísticas [21].

Para todo $i \in I$, $\tilde{\tau}(i)$ denota el grado de membresía de i en una transacción difusa $\tilde{\tau}$. Análogamente, para $A \subseteq I$,

$$\tilde{\tau}(A) = \min_{i \in A} \tilde{\tau}(i) \quad (1)$$

Un conjunto de N transacciones difusas con que agrupan a M ítems puede representarse como una matriz de $N \times M$ donde cada entrada $\alpha_{ik} = \tilde{\tau}_i(i_k)$

En [18], las medidas de interés, como el soporte y la confianza, se extienden al caso difuso usando un enfoque semántico basado en la evaluación de sentencias cuantificadas.

Las reglas de asociación difusa, entre otras aplicaciones, se han utilizado para la estimación de nulos [22], mas no han sido integradas a un mecanismo de consultas a bases de datos relacionales.

C. Modelo para la Extracción de Reglas Difusas Mediante Niveles de Restricción

En [23] se desarrolla un modelo formal para representar y evaluar reglas de asociación, tanto precisas como difusas, generalizando de una manera natural las medidas de interés del caso preciso al difuso. Para la extracción de las reglas de asociación se evalúa la frecuencia de las cuatro posibles combinaciones de dos *itemsets*: antecedente y consecuente.

Otro concepto útil en este modelo es el α -corte de un conjunto difuso. Esto se refiere al conjunto preciso de los elementos con membresía mayor o igual a un nivel α . Una propiedad difusa en un universo U puede ser representada por un conjunto de realizaciones precisas y que en el caso particular de un conjunto difuso se referirían a los α -cortes, los cuales serán llamados RL (niveles de restricción) [24].

Definición 1. [24] Un RL-set Λ es un conjunto finito de niveles de restricción $\Lambda = \{\alpha_1, \dots, \alpha_m\}$ verificando $1 = \alpha_1 > \alpha_2 > \dots > \alpha_m > \alpha_{m+1} = 0$ para $m \geq 1$.

En términos generales, el RL-set de una propiedad atómica representada por un conjunto difuso A sería como en la definición siguiente:

Definición 2. [24] Sea A un conjunto difuso definido sobre el referencial X. Entonces el RL-set para A viene dado por:

$$\Lambda_A = \{A(x) \mid x \in X\} \cup \{1\} \quad (2)$$

Donde A(x) es el grado de pertenencia de x al conjunto difuso A.

Definición 3. [24] Sean P, Q dos propiedades difusas con RL-representaciones (Λ_P, ρ_P) , (Λ_Q, ρ_Q) . Entonces, $P \wedge Q$, $P \vee Q$, $\neg P$ son propiedades difusas representadas por $(\Lambda_{P \wedge Q}, \rho_{P \wedge Q})$, $(\Lambda_{P \vee Q}, \rho_{P \vee Q})$ y $(\Lambda_{\neg P}, \rho_{\neg P})$ respectivamente, donde

$$\Lambda_{P \wedge Q} = \Lambda_P \cup \Lambda_Q \quad (3)$$

$$\Lambda_{\neg P} = \Lambda_P \quad (4)$$

Y para todo $\alpha \in (0, 1]$,

$$\begin{aligned} \rho_{P \wedge Q}(\alpha) &= \rho_P(\alpha) \cap \rho_Q(\alpha) \\ &= x \in X \mid P(x) \geq \alpha \wedge Q(x) \geq \alpha \end{aligned} \quad (5)$$

$$\rho_{\neg P}(\alpha) = \overline{\rho_P(\alpha)} = x \in X \mid \neg(P(x) \geq \alpha) \quad (6)$$

Donde \overline{Y} es el complemento usual de un conjunto preciso Y.

Ahora en función de las definiciones anteriores se procede a ilustrar que forma tendría una tabla de contingencia, que en el modelo mencionado se denomina tabla-4ft. Sean los

conjuntos difusos $\tilde{\Gamma}_A$ y $\tilde{\Gamma}_B$ definidos en un conjunto de transacciones difusas \tilde{D} como $\tilde{\Gamma}_A(\tilde{\tau}) = \tilde{\tau}(A)$ y $\tilde{\Gamma}_B(\tilde{\tau}) = \tilde{\tau}(B)$ respectivamente. Se puede denotar sus RL-representaciones como $(\Lambda_{\tilde{A}}, \rho_{\tilde{A}})$ y $(\Lambda_{\tilde{B}}, \rho_{\tilde{B}})$.

Ahora se consideran los conjuntos de ítems A, B y sus respectivas combinaciones $A \wedge B$, $A \wedge \neg B$, $\neg A \wedge B$ y $\neg A \wedge \neg B$, las cuales forman una partición de la base de datos \tilde{D} . También, se pueden considerar los conjuntos difusos asociados definidos en \tilde{D} con sus respectivas RL-representaciones: $(\Lambda_{\tilde{A} \wedge \tilde{B}}, \rho_{\tilde{A} \wedge \tilde{B}})$, $(\Lambda_{\tilde{A} \wedge \neg \tilde{B}}, \rho_{\tilde{A} \wedge \neg \tilde{B}})$, $(\Lambda_{\neg \tilde{A} \wedge \tilde{B}}, \rho_{\neg \tilde{A} \wedge \tilde{B}})$ y $(\Lambda_{\neg \tilde{A} \wedge \neg \tilde{B}}, \rho_{\neg \tilde{A} \wedge \neg \tilde{B}})$.

Para cada $\alpha \in \Lambda_X$, $\rho_Y(\alpha)$ es un conjunto preciso y se puede calcular su cardinalidad de la manera usual: $|\rho_Y(\alpha)|$. De forma análoga, para cada $\alpha_i \in \Lambda_{\tilde{A}} \cup \Lambda_{\tilde{B}}$ se puede definir $a_i = |\rho_{\tilde{A} \wedge \tilde{B}}(\alpha_i)|$, $b_i = |\rho_{\tilde{A} \wedge \neg \tilde{B}}(\alpha_i)|$, $c_i = |\rho_{\neg \tilde{A} \wedge \tilde{B}}(\alpha_i)|$ y $d_i = |\rho_{\neg \tilde{A} \wedge \neg \tilde{B}}(\alpha_i)|$. Cada una de estas cardinalidades formarían parte de la tabla-4ft asociada a α_i (ver Tabla I), denotada como $M_{\alpha_i} = 4\text{-ft}(\tilde{\Gamma}_A, \tilde{\Gamma}_B, \tilde{D}, \alpha_i)$:

Tabla I: Tabla-4ft Asociada a α_i

M_{α_i}	$\tilde{\Gamma}_B$	$\tilde{\Gamma}_{\neg B}$
$\tilde{\Gamma}_A$	a_i	b_i
$\tilde{\Gamma}_{\neg A}$	c_i	d_i

La generalización de las medidas de interés es como sigue:

Sea $A \subseteq I$ un *itemset* y $(\Lambda_{\tilde{A}}, \rho_{\tilde{A}})$ la RL-representación asociada al conjunto difuso $\tilde{\Gamma}_A$ en \tilde{D} . Entonces, el soporte extendido de A en el conjunto de transacciones difusas \tilde{D} se define como

$$\text{sop}(A) = \sum_{\alpha_i \in \Lambda_{\tilde{A}}} (\alpha_i - \alpha_{i+1}) \left(\frac{|\rho_{\tilde{A}}(\alpha_i)|}{|\tilde{D}|} \right) \quad (7)$$

Al considerar la parte derecha de la fórmula (7) y usar la tabla-4ft asociada a los *itemsets* A y B para el cálculo del nivel de restricción α_i se obtiene lo siguiente:

$$\frac{|\rho_{\tilde{A}}(\alpha_i)|}{|\tilde{D}|} = \frac{a_i + b_i}{a_i + b_i + c_i + d_i} \quad (8)$$

La fórmula (8) da el soporte de un *itemset* cuando la base de datos es precisa y $a_i + b_i + c_i + d = n$ es constante para cualquier restricción y representa el número de transacciones difusas en \tilde{D} . A continuación se muestran el soporte y la confianza de una regla difusa. Sean $A, B \subseteq I$ dos *itemsets* disjuntos y $(\Lambda_{\tilde{A}}, \rho_{\tilde{A}})$, $(\Lambda_{\tilde{B}}, \rho_{\tilde{B}})$ las *RL*-representaciones asociadas a los conjuntos difusos $\tilde{\Gamma}_A$ y $\tilde{\Gamma}_B$ en \tilde{D} . Entonces, el soporte de la regla difusa $A \rightarrow B$ en \tilde{D} se define como:

$$Sop(A \rightarrow B) = sop(A \wedge B) \quad (9)$$

$$sop(A \wedge B) = \sum_{\alpha_i \in \Lambda_{\tilde{A}} \cup \Lambda_{\tilde{B}}} (\alpha_i - \alpha_{i+1}) \left(\frac{|\rho_{\tilde{A} \wedge \tilde{B}}(\alpha_i)|}{|\tilde{D}|} \right) \quad (10)$$

Y la confianza de la regla difusa $A \rightarrow B$ en \tilde{D} se define como:

$$Conf(A \rightarrow B) = \sum_{\alpha_i \in \Lambda_{\tilde{A}} \cup \Lambda_{\tilde{B}}} (\alpha_i - \alpha_{i+1}) (\Rightarrow_I(a_i, b_i)) \quad (11)$$

$$\Rightarrow_I(a_i, b_i) = \frac{|\rho_{\tilde{A} \wedge \tilde{B}}(\alpha_i)|}{|\rho_{\tilde{A}}(\alpha_i)|} \quad (12)$$

Finalmente, se indica el factor de certeza. Sean $A, B \subset I$ dos *itemsets* disjuntos en D y sea $4ft\ A, B, D = \langle a, b, c, d \rangle$ su tabla-4ft asociada. El cuantificador \equiv_{FC} asociado al factor de certeza viene dado por:

$$\equiv_{FC}(a, b, c, d) = \begin{cases} \frac{ad - bc}{(a+b)(b+d)} & \text{si } ad > bc \\ 0 & \text{si } ad = bc \\ \frac{ad - bc}{(a+b)(a+c)} & \text{si } ad < bc \end{cases} \quad (13)$$

Ahora en función de la definición anterior del cuantificador factor de certeza se procede a extenderlo al caso difuso. En este caso se considera a A y B como dos *itemsets* disjuntos difusos en \tilde{D} y su tabla-4ft para cada nivel de restricción $\Lambda_{\tilde{A}} \cup \Lambda_{\tilde{B}}$ que representa la unión de los *RL*-sets de $\tilde{\Gamma}_A$ y $\tilde{\Gamma}_B$. En función de lo anterior el cuantificador \equiv_{FC} extendido al caso difuso es:

$$\equiv_{FC}(A \rightarrow B) = \sum_{\alpha_i \in \Lambda_{\tilde{A}} \cup \Lambda_{\tilde{B}}} (\alpha_i - \alpha_{i+1}) (\equiv_{FC}(a_i, b_i, c_i, d_i)) \quad (14)$$

D. Estrategia de Poda de Reglas

El uso de reglas de asociación difusa implica la posibilidad de obtener una base de reglas de gran tamaño lo cual tiende a aumentar con el número de transacciones.

Es necesario depurar el proceso de minado para acotar la cantidad de reglas a obtener lo cual puede disminuir también el tiempo de ejecución y el espacio para el almacenamiento de reglas.

Una de las formas de depuración es la poda de reglas, la cual consiste en una serie de criterios o técnicas que permiten filtrar una gran cantidad de reglas.

La poda se basa en ciertas propiedades o estructuras subyacentes de las reglas que permiten la aplicación de diversos métodos como los de agrupamiento de reglas, ontologías, hipergrafos, cobertura en función de la estructura de reglas, los de cobertura informativa, entre otros [25][26][27].

Algunos mecanismos de poda, como es el caso de [27], destacan por su sencillez y facilidad de implementación. Este último se basa en la idea de calcular un conjunto de reglas más pequeño (conjunto de reglas informativo) pero que tiene la misma capacidad predictiva que la base de reglas obtenida por el minado convencional.

A continuación se resume la teoría relacionada con la obtención del conjunto de reglas informativo, mostrándose sólo aquellos que serán de utilidad para la poda de reglas del mecanismo propuesto [27]:

Definición 4. Dado un conjunto de reglas de asociación R y un *itemset* P , se dice que la predicción de P desde R es una secuencia de *ítems* Q la cual es generada al usar las reglas de R en orden descendente de confianza. Para cada regla r cuyo antecedente es un subconjunto de P , cada consecuente de r es agregado a Q . Después de agregar un consecuente a Q , todas las reglas con este consecuente se eliminan de R .

Para excluir aquellas reglas que nunca han sido usadas en la predicción se presenta la siguiente definición:

Definición 5. Sea R un conjunto de reglas de asociación y sea R^1 el conjunto de reglas de un solo consecuente en R .

Un conjunto de reglas de asociación R_I es informativo sobre R si (1) $R_I \subset R^1$; (2) $\forall r \in R_I, \nexists r' \in R_I$ tal que $r' \subset r$ y $conf(r') \geq conf(r)$; y (3) $\forall r'' \in R^1 - R_I, \exists r \in R_I$ tal que $r'' \supset r$ y $conf(r'') \leq conf(r)$.

Teorema 1. Sea R un conjunto de reglas de asociación. Luego el conjunto informativo de reglas R_I sobre R_A es el más pequeño subconjunto de R_A tal que, para cualquier *itemset* P , la secuencia de predicción de P desde R_I es igual a la secuencia de P desde R .

El teorema anterior expresa una propiedad muy importante del conjunto informativo de reglas, es decir, que su capacidad de predicción (de los consecuentes en la base de reglas) es igual al conjunto original de reglas pero es un subconjunto de este último y además es el más pequeño posible.

Lema 1. Si $sop(A) = sop(A \cup B)$, entonces para cualquier Z , la regla $XY \rightarrow Z$ y todas aquellas reglas más específicas no ocurren en el conjunto informativo de reglas.

El lema anterior permite descartar las reglas derivables las cuales no deben estar en el conjunto informativo de reglas. Se dice que una regla es derivable si su confianza y soporte pueden derivarse de otras reglas más generales. Específicamente, una regla $A \rightarrow B$ es derivable si hay un conjunto R de reglas, todas más generales que $A \rightarrow B$, tal que $A \rightarrow B$ y su soporte y confianza pueden ser obtenidos de R .

Lema 2. Si $sop(A \neg C) = sop(A \cup B \neg C)$, entonces la regla $A \cup B \rightarrow C$ y todas aquellas reglas más específicas no ocurren en el conjunto informativo de reglas. Donde $sop(A \neg C) = sop(A) - sop(A \cup C)$, es decir, el soporte de los *itemsets* que contienen a A pero no contienen a C.

E. Etiquetas Lingüísticas Basadas en el Contexto

En un trabajo anterior [28], se definió una extensión de consultas difusas sobre bases de datos relacionales en la que se interpretan términos vagos dependiendo del contexto. Aunque la definición de etiquetas difusas depende mucho de la preferencia del usuario, puede adecuar su interpretación a los datos reales, lo cual se conoce como el contexto de los datos.

Definición 6. [28] Un conjunto difuso es una generalización del concepto convencional de conjunto. Sea U un dominio específico, un conjunto difuso A es determinado por $\mu_A : X \rightarrow [0,1]$, llamada la función de membresía. Para cualquier $x \in U$, la medida $\mu_A(x)$ es conocida como el grado de membresía de x. Así, el conjunto difuso es definido como una colección de pares ordenados:

$$A = \{(x, \mu_A(x)) \mid x \in U\} \quad (15)$$

Definición 7. [28] Sea U un dominio numérico $x_1 \leq x_2 \leq x_3 \leq x_4 \in U$, se define la forma lineal de las funciones de membresía: *trapezoidal*(x_1, x_2, x_3, x_4) como la función de membresía $\mu : X \rightarrow [0,1]$ dada en la Figura 1; se define también *left shoulder*(x_2, x_3, x_4) como *trapezoidal*($-\infty, x_2, x_3, x_4$) y *right shoulder*(x_1, x_2, x_3) como *trapezoidal*($x_1, x_2, x_3, +\infty$) (ver Figura 1).

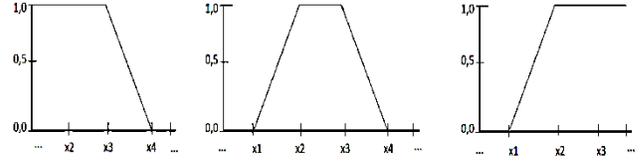


Figura 1: Forma Lineal de las Funciones de Membresía de Izquierda a Derecha: *left shoulder*(x_2, x_3, x_4), *trapezoidal*(x_1, x_2, x_3, x_4) Y *right shoulder*(x_1, x_2, x_3)

Definición 8. [28] Un *Marco de Cognición* $\langle U, F, \preceq \rangle$ es definido como una familia de conjuntos difusos F definido sobre el mismo universo del discurso U . El nivel de granularidad K de un *Marco de Cognición* es la cardinalidad de F . Resulta conveniente para cualquier aplicación establecer un orden total \preceq sobre el marco de cognición. Se denota A_i al i -ésimo conjunto difuso en el *frame* F para $i \leq K$.

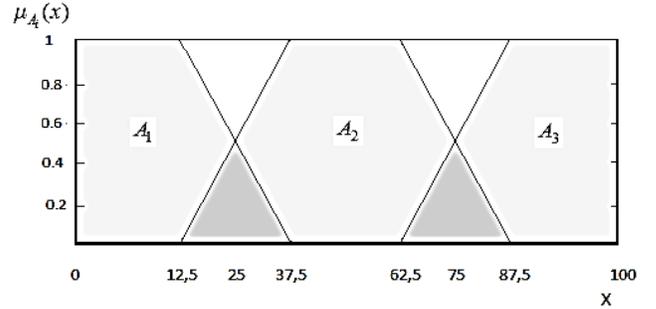


Figura 2: Ejemplo de Marco de Cognición con $k=3$

Definición 9. [28] Un Marco de Cognición es denominado una partición difusa si, y sólo si:

$$\forall A \in F \ A \neq \emptyset \quad (16)$$

$$\bigcup_{A \in F} A = U \quad (17)$$

A continuación en la Tabla II se presentan los modelos teóricos para la interpretación de términos vagos basados en el contexto hasta $k=3$. Donde k es la granularidad de la categorización, I es el orden de la etiqueta y la tercera columna es la semántica de la i -ésima etiqueta. Además, cada P_q es el q -ésimo percentil para los valores del atributo.

Estos modelos representan las reglas semánticas para encontrar el significado de una etiqueta dependiente del contexto de unos datos específicos y usando los valores percentiles extraídos de dichos datos, según [28]. Estos percentiles permitirán inferir los términos vagos asociados a una etiqueta en términos de conjuntos difusos, los cuales serán expresados en función de las funciones de membresía difusa asociadas a los mismos.

Tabla II: Modelos Teóricos para la Interpretación de Términos Vagos Basados en el Contexto

K	I	Función de Membresía
2	1	$left\ shoulder(P_{0}, P_{37.5}, P_{62.5})$
	2	$right\ shoulder(P_{37.5}, P_{62.5}, P_{100})$
3	1	$left\ shoulder(P_{0}, P_{12.5}, P_{37.5})$
	2	$trapezoidal(P_{12.5}, P_{37.5}, P_{62.5}, P_{87.5})$
	3	$right\ shoulder(P_{62.5}, P_{87.5}, P_{100})$

III. MECANISMO PROPUESTO

En esta sección, se detalla la solución propuesta. En la sección A se describe el modelo planteado para tratar consultas en bases de datos incompletas. En la sección B, se describen las medidas de interés aplicadas por el componente de minería de reglas difusas perteneciente al mecanismo propuesto. En la sección C se describen los algoritmos principales que implementan el mecanismo propuesto.

A. Modelo de Tratamiento de Consultas

Aquí se plantea la aplicación de un método en bases de datos relacionales, donde no se realiza la imputación sino una estimación en forma dinámica en el momento de ejecutar una consulta que involucre un atributo que tenga valores nulos. De esta manera se evitan las desventajas antes descritas asociadas a la imputación de valores, antes descritas.

Por otro lado, se plantea el uso de reglas de asociación difusa, técnica ésta que permite obtener un conjunto de reglas tanto para atributos con valores continuos como con valores categóricos [29]. Este conjunto de reglas serán obtenidas sólo en función de los valores presentes en la base de datos y serán usadas para estimar los valores nulos que pudieran presentarse para los atributos que conforman la consulta. Estos valores estimados no reemplazarían los de la base de datos como en el caso de la imputación ni tampoco se hacen extensiones al modelo relacional como se realiza en los enfoques antes descritos, evitándose de esta forma las desventajas asociadas a la implementación de los mismos. También el usuario podrá conocer cuáles nulos pudiera arrojar su consulta y en qué casos se sustituyeron por estimaciones sólo a nivel de los resultados obtenidos de la consulta y en función de las preferencias expresadas por dicho usuario.

Con este planteamiento se pueden realizar consultas a la base de datos incompleta sin modificar su estado original y obteniendo un conjunto de respuestas más amplio gracias a la estimación de los valores nulos.

En particular el valor que se propone como estimado de un valor nulo sería una etiqueta difusa del atributo a estimar, que tiene de por sí una incertidumbre asociada a su representación y permite la adaptación a las preferencias particulares del usuario en cuanto a la precisión deseada para la estimación. Estas etiquetas son definidas combinando las preferencias del usuario y el contexto de los datos.

En cuanto al motor de inferencia utilizado para estimar los nulos, se cuenta con las reglas de asociación difusa que han probado ser de amplia aplicación y altamente configurables por parte del usuario. Cabe destacar que las reglas difusas deben ser generadas haciendo uso del motor de inferencia integrado. Una vez que la base de reglas es generada y almacenada en la base de datos por el motor de inferencia, es usada para realizar estimaciones en las consultas.

El modelo subyacente para minar las reglas de asociación difusa es un modelo maduro y extrapolable a muchos contextos y que permite una extensión natural de las medidas de interés de las reglas de asociación al caso difuso [23].

Otro aspecto importante en el algoritmo de minería de reglas de asociación difusa es el uso de técnicas de poda a fin de mantener acotada la cantidad de reglas y descubrir aquellas que son representativas de los patrones que se pueden derivar de la base de datos. Esto permite mejorar el uso del espacio de almacenamiento y reducir el tiempo de procesamiento con respecto a otros algoritmos de minería de reglas tradicionales.

Por otro lado, a diferencia de soluciones existentes anteriormente mencionadas, el mecanismo estimación de nulos aquí propuesto se integra completamente al sistema gestor de bases de datos, como se evidencia en la Figura 3.

Se propone como estimación del valor de un atributo faltante una etiqueta difusa. Para la estimación de dicha etiqueta, se consulta una base de reglas de asociación difusa. Esta base de reglas se obtiene al minar la base de datos del usuario y de acuerdo a las etiquetas difusas que este defina.

Como puede apreciarse en la Figura 3, tanto las etiquetas, la base de datos difusa y la base de reglas se guardan en tablas en forma permanente en la base de datos. De esta manera están a disposición del usuario y completamente integradas al sistema gestor de bases de datos.

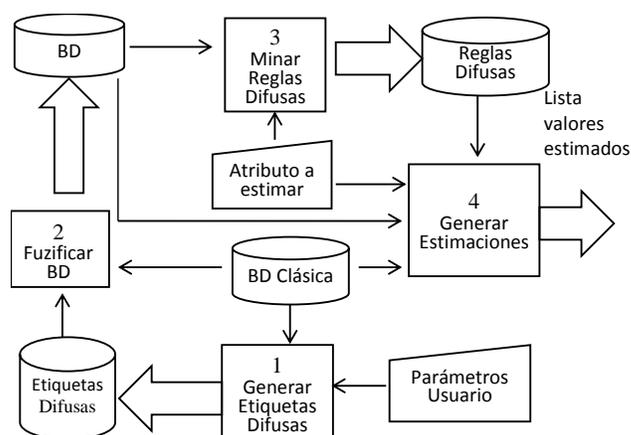


Figura 3: Arquitectura del Mecanismo de Tratamiento de Consultas en Bases de Datos Incompletas

B. Medidas de Interés del Modelo

Para el cálculo de la base de reglas difusas es necesario seleccionar un conjunto de medidas de interés adecuadas. Como se observó en la sección II, el modelo formal que se

propuso para la extracción de reglas difusas extiende de una manera natural las medidas de interés clásicas como lo son el soporte, la confianza y la certeza al caso difuso.

La ventaja de usar las medidas de interés extendidas al caso difuso reside en la capacidad expresiva que tiene la lógica difusa para representar el razonamiento humano. Por ejemplo, considere el valor x de un atributo a dentro de una transacción difusa. Suponga que dicho atributo a tiene un marco de cognición (partición difusa) asociado con dos etiquetas adyacentes o solapadas e_1 , e_2 (ver Figura 2). Además, suponga que el valor x del atributo se a ubica en el espacio donde se solapan e_1 y e_2 . En este caso, el grado de pertenencia del valor x a la transacción se puede representar por dos g_1 y g_2 , correspondientes a e_1 y e_2 respectivamente, estos grados son complementarios, es decir $g_1+g_2=1$. De este modo, cuando se calcule una medida de interés que involucre la etiqueta e_1 ésta se tomará en cuenta para el cálculo de la cardinalidad, si g_1 es mayor o igual al mínimo grado exigido por el usuario para la medida de interés (análogamente para e_2). Lo anterior tiene como ventajas una mayor flexibilidad para el cálculo de la cardinalidad, una mejor adaptación al razonamiento humano (el cual es gradual) y se minimiza la pérdida de información en el proceso de cálculo de dichas medidas.

C. Diseño de Algoritmos

En este apartado se describen los procesos y algoritmos principales que componen el mecanismo propuesto para el tratamiento de consultas en bases de datos incompletas.

El usuario define de acuerdo a sus preferencias, las etiquetas difusas que conforman el marco de cognición para los atributos que tendrán un tratamiento difuso. La interpretación de las etiquetas se hace basado en el contexto de la base de datos clásica D sobre la cual se quiere hacer la minería y las consultas. Siguiendo el modelo teórico [28] se generan los conjuntos difusos que definen cada etiqueta y se almacenan en una base de datos.

Con estas etiquetas lingüísticas se hace un proceso de fuzzificación. Esto es, se genera una base de datos difusa \tilde{D} asociada a D . \tilde{D} contiene los mismos registros que D pero representados en forma difusa. Para aquellos atributos que tiene un tratamiento difuso, en cada registro se sustituye el valor por la etiqueta correspondiente. En caso que el valor del atributo caiga en un intervalo de solapamiento, se tienen dos etiquetas e_1 y e_2 con sus respectivos grados g_1 y g_2 .

Las etiquetas lingüísticas y la base de datos difusa se utilizan para minar las reglas de asociación difusa. En el caso que el usuario no introduzca etiquetas lingüísticas el sistema se comporta como un sistema preciso generando una base de reglas clásica.

Para cada atributo a del esquema de la base de datos D se genera un conjunto C_a de *itemsets* de cardinalidad 1. C_a contiene todos los *itemsets* de la forma $\{a=v\}$, siendo v un valor posible para a en la base de datos difusa \tilde{D} .

Nótese que puede haber tanto valores precisos como difusos, dependiendo del tratamiento que el usuario quiera dar a cada atributo. En consecuencia, cuando se calcule el valor

estimado de un atributo, si para este se definió un marco de cognición, el estimado será una etiqueta, de lo contrario, será un valor clásico.

A continuación se presentarán los algoritmos de minería de reglas de asociación difusa por niveles. Las reglas de nivel k son aquellas que tienen k ítems en el antecedente. No todas las reglas son admisibles, sino aquellas que cumplen con los niveles de calidad establecidos.

Algoritmo para minar de reglas difusas de nivel 1

Entradas:

\tilde{D} base de datos difusa asociada a D

b atributo en el esquema de D al cual estimar su valor

C_A conjunto de *itemsets* de cardinalidad 1 para atributos distintos de b , $C_A = \bigcup_{a \neq b} C_a$

C_b conjunto de *itemsets* de cardinalidad 1 para b

min_sop mínimo para la medida de soporte

min_conf mínimo para la medida de confianza

min_cert mínimo para la medida de confianza

Salida:

\tilde{R}_1 conjunto de reglas de asociación difusa nivel 1 minadas a partir de \tilde{D}

C'_A conjunto de *itemsets* frecuentes de cardinalidad 1 para atributos distintos de b

C'_b conjunto de *itemsets* frecuentes de cardinalidad 1 para el atributo b

Inicio

1) Inicializar C'_A en C_A y C'_b con C_b

// todos los *itemsets* son potencialmente frecuentes

2) Para todo $i_A \in C'_A$ si $sop(i) < min_sop$ entonces

$C'_A := C'_A - \{i_A\}$ // el *itemset* no es frecuente)

3) Para todo $i_b \in C'_b$ si $sop(i) < min_sop$ entonces

$C'_b := C'_b - \{i_b\}$ // el *itemset* no es frecuente)

4) Inicializar \tilde{R}_1 en \emptyset

5) Para todo $i_A \in C'_A$, $i_b \in C'_b$

si (// la regla supera la calidad

$sop(i_A \rightarrow i_b) \geq min_sop \wedge$

$conf(i_A \rightarrow i_b) \geq min_conf \wedge$

$cert(i_A \rightarrow i_b) \geq min_cert$

) entonces $\tilde{R}_1 := \tilde{R}_1 \cup \{i_A \rightarrow i_b\}$

Fin

Las reglas de nivel k se obtienen a partir de las de nivel $k-1$, es aquí donde se aplican los criterios de poda explicados en las bases teóricas de esta propuesta.

Algoritmo para minar de reglas difusas de nivel k

Entradas:

\tilde{D} base de datos difusa asociada a D

b atributo en el esquema de D al cual estimar su valor

C'_A conjunto de *itemsets* frecuentes de cardinalidad 1 para atributos distintos de b

C'_b conjunto de *itemsets* frecuentes de cardinalidad 1 para el atributo b

\widetilde{R}_{k-1} conjunto de reglas de asociación difusa nivel $k-1$ minadas a partir de \tilde{D}

min_sop mínimo para la medida de soporte

min_conf mínimo para la medida de confianza

min_cert mínimo para la medida de confianza

Salida:

\widetilde{R}_k conjunto de reglas de asociación difusa nivel k minadas a partir de \tilde{D}

Inicio

1) Inicializar \widetilde{R}_k en \emptyset

2) Para todo $i \rightarrow i_b \in \widetilde{R}_{k-1}$, $i_A \in C'_A$

si (// la regla supera la poda

$sop(i) \neq sop(i \cup i_b) \wedge$

$sop(i) - sop(i \cup i_b) \neq sop(i \cup i_A) - sop(i \cup i_A \cup i_b)$

)

\wedge (// la regla supera la calidad

$sop(i \cup i_A \rightarrow i_b) \geq min_sop \wedge$

$conf(i \cup i_A \rightarrow i_b) \geq min_conf \wedge$

$cert(i \cup i_A \rightarrow i_b) \geq min_cert$

) entonces $\widetilde{R}_k := \widetilde{R}_k \cup \{ i \cup i_A \rightarrow i_b \}$

Fin

El conjunto de reglas de asociación difusa minada a partir de la base de datos D se obtiene por la unión de las reglas de todos los niveles. El proceso de minería es iterativo y hasta que se llegue a un punto fijo, es decir, cuando se llegue a un nivel en que no se generen nuevas reglas. El conjunto de reglas obtenido se almacena en una base de datos a fin de poder ser usado al momento de una consulta.

A continuación se presenta el algoritmo para la estimación del valor de un atributo, En principio, este algoritmo se invocaría si el usuario desea obtener una respuesta estimada en lugar de un nulo.

Algoritmo para estimación de atributo

Entradas:

d registro en la instancia de una base de datos D

b atributo en el esquema de D al cual estimar su valor

\tilde{D} base de datos difusa asociada a D

\widetilde{R} base de reglas de asociación difusa minada de D

Salida:

v estimado para b en el registro d

Inicio

- 1) Recuperar la versión difusa \tilde{d} de d en la base de datos \tilde{D} .
- 2) Recuperar las reglas en \widetilde{R} con b en el consecuente, cuyo antecedente coincide con \tilde{d}
- 3) Ordenar las reglas recuperadas descendientemente por soporte, confianza y certeza
- 4) Elegir la primera en el orden, en caso de empate entre dos reglas se toma cualquiera de ellas
- 5) Seleccionar el consecuente de la regla elegida como v .

Fin

IV. EXPERIMENTOS Y RESULTADOS

A. Diseño de Experimentos

Los datos elegidos para probar el sistema se encuentran explicados en [8], donde se estudia el uso de herramientas de aprendizaje automático para estimar dos parámetros de eficiencia energética de edificios residenciales. Todas las variables son numéricas dado que el mecanismo de estimación no es tan exigido en su rendimiento como cuando procesa atributos numéricos que posean etiquetas difusas. Esto se debe a la etapa previa de fuzzificar la base de datos, buscar las etiquetas difusas y evaluarlas durante una consulta.

El conjunto de datos se compone de los siguientes variables numéricas HL (*Heating Load*) y CL (*Cooling Load*), en función de otras ocho variables de entrada, también numéricas: RC (*Relative Compactness*), SA (*Surface Area*), WA (*Wall Area*), RA (*Roof Area*), OH (*Overall Height*), O (*Orientation*), GA (*Glazing Area*), GAD (*Glazing Area Distribution*). En el presente análisis se usará la variable de salida HL y se le identificará como HL1, mientras que las variables de entrada se denominarán como RC1, SA2, WA3, RA4, OH5, O6, GA7 y GAD8.

En cuanto a la escogencia de los atributos que usarían etiquetas se realizaron una serie de pruebas combinando los atributos de a pares a fin de determinar el efecto de la interacción de los mismos en el rendimiento y precisión del algoritmo. De las pruebas realizadas se determinó preliminarmente que la combinación de los atributos RC1 y SA2 disminuía el rendimiento, debido quizás a que los atributos no son independientes [8], y debido a esto se eligieron las combinaciones RC1, WA3, RA4, GA7, HL1 y

SA2, WA3, RA4, GA7,HL1. Los atributos OH5, O6 y GAD8, pese a ser numéricos son considerados categóricos dada su baja granularidad y por lo cual no se definieron etiquetas para ellos.

Una vez elegidos los atributos el primer tipo de pruebas fue variando la cantidad de nulos y de etiquetas presentes en los atributos. En función de lo anterior se construyeron distintos escenarios para los cuales se obtuvieron distintas bases de reglas, las cuales a su vez se utilizaron para estimación. En la generación de las reglas se hicieron pruebas para medir la efectividad de los criterios de poda incorporados en el sistema.

El objetivo de estas pruebas es medir el impacto en el algoritmo en la medida que aumenta la cantidad de nulos o dicho de otra forma, en la medida que disminuye la información disponible. En cuanto al uso de etiquetas éstas permiten por un lado disminuir la granularidad de los atributos que las usan y por el otro al incrementarse su uso también aumenta la incertidumbre en los datos dada la naturaleza imprecisa de las mismas.

Adicionalmente a las pruebas preliminares para comprobar el funcionamiento del algoritmo, se realizaron unas pruebas de validación cruzada para comparar el mecanismo propuesto con otros dos [8]. Los dos mecanismos a comparar con el mecanismo de estimación son el método de los mínimos cuadrados iterativamente re ponderados o en inglés IRLS (*Iteratively Reweighted Least Squares*) y RF (*Random Forests*). La estructura de las pruebas es la descrita en [8] y consistió en repetir cien veces la ejecución del algoritmo. Para cada repetición fueron seleccionadas dos sub muestras, una del 10% de los datos totales como conjunto de prueba y la otra con el resto 90% como conjunto de datos de entrenamiento. Antes de escoger ambas sub muestras en cada repetición los datos totales son permutados. Finalmente en cada repetición se registra el MRE (*Mean Relative Error*), que luego se promedia para las cien repeticiones, y se especifica a continuación según [8]:

$$MRE = 100 \cdot \frac{1}{S} \sum_{i \in Q} |y_i - \hat{y}_i| / y_i \quad (18)$$

Donde \hat{y}_i es el valor estimado para el valor actual y_i de la variable HL1, perteneciente al valor i -ésimo del conjunto de entrenamiento o el de prueba. S representa el número de registros del conjunto de entrenamiento y Q son los índices de ese conjunto.

Cabe destacar que para probar el mecanismo propuesto se usa un MRE donde $|y_i - \hat{y}_i| / y_i$, que representa el error relativo en cada estimación, es 0 si la etiqueta estimada corresponde a alguna etiqueta representada en la base de datos difusa y 1 en caso contrario. Por otro lado, dado que se tienen etiquetas lingüísticas definidas, el conjunto de registros con el valor de la etiqueta es un conjunto difuso. Debido a esto, S se calcula con la cardinalidad $\sum count$ de Zadeh [21], es decir se suman los grados de membresía de los elementos al conjunto. Así, si en una transacción hay una

sola etiqueta esta suma 1, pero si hay dos etiquetas e_1 y e_2 con grados g_1 y g_2 respectivamente, la transacción suma g_1 para e_1 y g_2 para e_2 . Por lo cual el valor de S es menor o igual al número de registros del conjunto de entrenamiento o el de prueba, dada la interpretación difusa que se le da a cada registro.

Por otra parte se realizaron 300 pruebas de validación cruzada: 100 para reglas con etiquetas sólo en el consecuente y 100 para cada una de las dos combinaciones de atributos con etiquetas en el antecedente y consecuente.

B. Resultados Obtenidos

Como puede observarse en la Tabla II y en la Figura 4 el porcentaje de estimación tiene una tendencia parecida independientemente de la presencia o no de valores nulos.

Esta tendencia se puede explicar ya que en cada momento el algoritmo construye la base de reglas considerando sólo la información disponible. Esta base de reglas posee una buena capacidad de generalización, ya que su poder predictivo no se ve afectado significativamente por la presencia de valores nulos en la muestra.

Por otro lado se puede apreciar una ligera disminución en el porcentaje de acierto promedio según la Tabla III cuando se utilizan etiquetas sólo en el consecuente de 94% versus un promedio de 89% cuando se usan etiquetas en antecedente y consecuente (ver último renglón), lo cual se puede confirmar en la Figura 4 y en el caso de usar valores nulos en distintas proporciones (ver Figura 5 y Figura 6).

Esta disminución pudiera deberse a que los atributos del antecedente no poseen una granularidad muy fina (2-12 valores diferentes), si se comparan con el atributo del consecuente (586 valores diferentes) [8], y al usar etiquetas difusas para ellos su granularidad se hace menos fina aún. Esto último genera una menor variedad de antecedentes y en consecuencia una menor variedad de reglas, disponiendo el algoritmo de menos oportunidades de generar una base de reglas de mayor calidad en cuanto a su poder predictivo.

Tabla III: Porcentaje de Aciertos %R Promedios por cada Tipo de Prueba para el Método Propuesto

Etiquetas Difusas	Solo Cons	Antecedente y Consecuente		
		RCl,WA3, RA4,GA7	SA2,WA, RA4,GA7	%R PROM
Nulos				
sin nulos	95	92	87	90
Solo consecuentes	15%	96	93	92
	30%	94	89	89
	45%	90	88	87
Antecedentes y Consecuentes	15%	93	91	90
	30%	97	92	93
	45%	95	92	91
%R Promedio		91	87	90
	94	89		

Otra razón que pudiera justificar una disminución en la precisión, es que al usar etiquetas tanto en el antecedente como en el consecuente se produce un aumento de la incertidumbre en la información necesaria para minado de la base de reglas, haciendo que ésta sea de menor calidad.

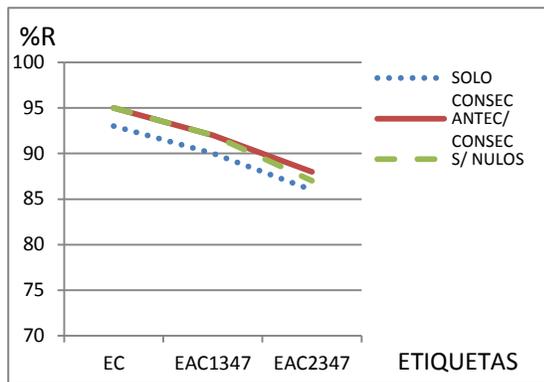


Figura 4: Porcentajes de Estimación para el Uso de Nulos para Etiquetas en el Consecuente y en el Antecedente para los Atributos 1347 y 2347

Por otro lado, el algoritmo pudo en todos los casos por encima de un 50% de reglas del total de reglas calculadas (ver Figura 5 y Figura 6). Este porcentaje de poda resulta ser significativo, ya que se debe mencionar que al podar se previene de generar aquellas derivadas de las reglas podadas con lo cual la cantidad minada será lo más pequeña posible. De esta manera, el algoritmo propuesto reduce considerablemente tiempo de procesamiento y espacio de almacenamiento.

Adicionalmente, se observa una tendencia a disminuir el porcentaje de reglas podadas al aumentar el porcentaje de nulos (ver Figura 5 y Figura 6).

En primer lugar, en el caso de valores nulos sólo en el consecuente al aumentar los mismos esto trae como consecuencia una disminución de los valores que deben ser estimados (valores en el consecuente) y por ende la cantidad de reglas minadas tiende a disminuir lo que a su vez reduce el porcentaje de poda requerido como puede verse en la Figura 5. En esta misma figura se puede apreciar que el uso de etiquetas en el antecedente y en el consecuente redujo el porcentaje de poda requerido en contraste con sólo etiquetas en el consecuente, lo cual pareciera indicar un aumento en la calidad informativa de las reglas minadas por el uso de etiquetas.

En segundo lugar, en el caso de valores nulos en el antecedente y consecuente al aumentar los nulos también aumenta la incertidumbre, la cual en el caso de etiquetas en el antecedente y en el consecuente se incrementaría aún más. Pese a lo anterior se observa en la Figura 6 que los porcentajes de poda se mantuvieron bastante similares a los de la Figura 5. Esto pudiera indicar una gran estabilidad del

componente de minado del sistema frente a niveles altos de incertidumbre en los datos.

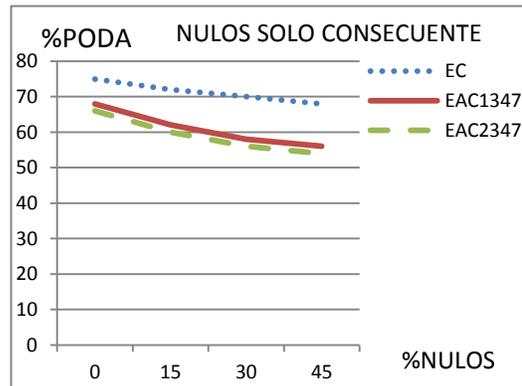


Figura 5: Porcentajes de Poda de Reglas para el Uso de Etiquetas en el Consecuente y en el Antecedente con Nulos sólo en el Consecuente

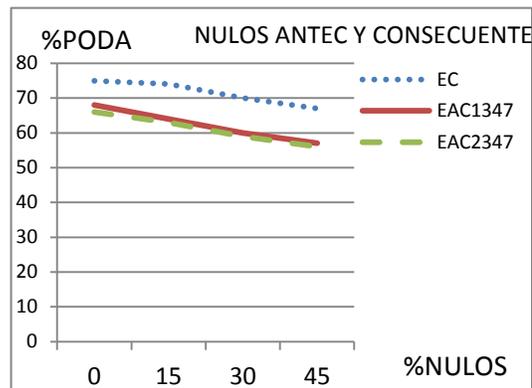


Figura 6: Porcentajes de Poda de Reglas para el Uso de Etiquetas en el Consecuente y en el Antecedente con Nulos en el Antecedente y Consecuente

Con respecto a las pruebas de validación cruzada (ver Tabla IV), el error promedio para los dos últimos casos que usan etiquetas en antecedente y consecuente es muy similar a diferencia de lo obtenido en las pruebas preliminares, lo cual puede indicar que no hay una diferencia apreciable entre usar el atributo 1 ó usar el 2 en cuanto a la precisión del algoritmo se refiere.

Por otro lado, se afianzó la tendencia obtenida en las pruebas preliminares en cuanto a que el error es menor usando sólo etiquetas en el consecuente a cuando se usan etiquetas en el antecedente y consecuente, obteniéndose en este último caso un error aproximadamente de 1/4/5.

Con respecto al método propuesto se puede observar de acuerdo a los resultados reflejados en la Tabla V que el error del mismo supera al del método RF pero es inferior a IRLS. Lo anterior se podría deber a que los datos usados para evaluar los métodos no son normales según [8], lo cual afecta a IRLS a diferencia del método propuesto que no requiere que los datos cumplan algún supuesto y por lo tanto la

precisión no se ve afectada. Además, pese a que la precisión es menor que RF, a diferencia de los otros dos métodos, tiene la flexibilidad de permitir modelar incertidumbre en todos los atributos usados como base para estimar y de acuerdo a las preferencias que el usuario de la base de datos ha preestablecido.

Tabla IV: Comparación de MRE entre las Pruebas Preliminares versus Validación Cruzada por cada Categoría de Prueba en cuanto al Uso de Etiquetas

	Uso de Etiquetas		
	<i>Consecuente</i>	<i>Anteced 1347-Consec</i>	<i>Anteced2347-Consec</i>
Prueba preliminar	5,45	7,79	12,6
Validación cruzada (error ± stdev)	7,39 ± 3,64	13,42 ± 6,59	13,14 ± 6,34

Por otro lado, la interpretabilidad de los estimados es más fácil en el método propuesto, debido a que se da como resultado una etiqueta lingüística la cual es más cercana al lenguaje natural. Esto permitiría, por ejemplo, que un experto en el contexto del problema pueda auditar con más facilidad la semántica de los patrones minados a fin de evaluar su fiabilidad.

El algoritmo, a diferencia de los otros dos métodos comparados, es bastante configurable en varios aspectos, como por ejemplo, en cuanto a los parámetros necesarios para su ejecución como lo son las medidas de interés. Éstas se les pueden asignar distintas combinaciones de valores a fin de experimentar distintos niveles de exigencia en cuanto a la precisión deseada. También se podría adaptar el algoritmo para no excederse de una cantidad máxima de reglas a ser minadas. Además, las etiquetas lingüísticas que usa el algoritmo se pueden definir siguiendo diversas estrategias de acuerdo a las preferencias del usuario y que en el caso particular de este trabajo se usó la definición de etiquetas basadas en el contexto [28].

El mecanismo propuesto tiene la versatilidad en cuanto a las características de los datos tratados, ya que trabaja con valores cuantitativos como categóricos en comparación con ILRS, además de soportar la presencia de nulos en los datos usados para realizar el proceso de minería en comparación con los otros dos métodos que no admiten nulos.

Otra ventaja resaltante con respecto a los otros dos métodos es que el método propuesto se encuentra encapsulado dentro del manejador de base de datos a través de una arquitectura de acoplamiento medio. Esta última característica le confiere una gran ventaja, con respecto a los otros métodos que se encuentran programados externamente a la base de datos, ya que se puede referenciar directamente desde la sintaxis SQL en las operaciones sobre los datos. Esto hace que la minería sea una extensión de las actividades habituales que un usuario especializado realiza dentro del manejador y aprovechándose además las ventajas de procesamiento, almacenamiento y rendimiento del manejador para la ejecución del algoritmo.

Por último, en cuanto al almacenamiento en la base de datos de las reglas minadas, esto es muy ventajoso debido a que no son requeridas nuevas ejecuciones del motor de inferencia. Lo anterior permite que el mecanismo propuesto pueda usarse para realizar tantas estimaciones como se requiera. En particular los otros dos métodos comparados, no disponen de la ventaja de almacenamiento permanente de su base de conocimientos en un manejador de base de datos.

Tabla V: Comparación de Método Propuesto con IRLS y RF

IRLS	RF	Método Propuesto
10,09 ± 1,01	2,18 ± 0,64	7,39 ± 3,64

V. CONCLUSIONES Y TRABAJOS FUTUROS

El objetivo fundamental del presente trabajo fue proveer un mecanismo automatizado que permitiera dar respuesta a consultas en base de datos relacionales en presencia de nulos, mediante la aplicación de reglas de asociación difusas que permitan estimar el valor de tales atributos. Este objetivo fue logrado a través de un modelo sencillo de tratamiento de consultas en bases de datos incompletas. En base a este modelo se construyó un mecanismo automatizado totalmente implementado en un manejador de base de datos a través del uso de funciones, procedimientos y estructuras de datos propias del manejador. Por otro lado, se definieron e implementaron como parte de este mecanismo las siguientes componentes:

Implementación de la estrategia automática de definición de etiquetas lingüísticas basadas en el contexto. Con esta estrategia el usuario de una manera sencilla puede dotar a su base de datos de una interpretación difusa más adaptada a la estructura de los datos sin necesidad de ser un experto en la semántica de los mismos. Implementación de un algoritmo de minería de reglas de asociación difusas basado en la extensión a un modelo formal para la representación y evaluación de reglas de asociación para la extracción de reglas difusas mediante niveles de restricción. Este modelo, además de ser sólido y probado a través de los años en diversos contextos, permitió darle de una manera bastante natural e intuitiva, una interpretación difusa a las medidas de interés clásicas usadas en la minería de reglas de asociación.

Un aspecto a resaltar del mecanismo es su aprovechamiento potencial a través de cualquier aplicación que haga consultas a bases de datos relacionales. Adicionalmente como valor agregado, se pueden explotar las bondades de escalabilidad, robustez y capacidades de administración propias de un SGBD y en éste está integrado el proceso de minería y no como un proceso externo, pudiéndose además guardar en la base de datos la base de reglas generadas para usos futuros. Además, una vez realizadas las diversas pruebas de rendimiento del algoritmo implementado se obtuvieron buenos resultados de estimación usando etiquetas difusas en diversos atributos, tanto en el antecedente como en el consecuente o atributo a estimar. Por otro lado, se comprobaron las bondades del algoritmo ante la presencia de nulos en la base de datos, los cuales no degradaron el

rendimiento en forma significativa del algoritmo. Adicionalmente, se comprobó experimentalmente la capacidad de éste para podar una cantidad significativa de reglas en base a los criterios de poda utilizados, los cuales reducen significativamente el tiempo de cómputo del algoritmo y el espacio de almacenamiento requerido. Finalmente se puede destacar que el mecanismo desarrollado no modifica la base de datos al realizar sus estimaciones evitándose así las desventajas asociadas a la imputación.

Entre los aspectos a desarrollar a futuro se pueden destacar:

Explorar la sensibilidad del algoritmo en cuanto al rendimiento se refiere y en cuanto a la calidad de la base de reglas generadas al realizar diferentes combinaciones para el soporte, confianza y certeza. Experimentar con otras medidas de interés investigadas en la literatura consultada a fin de que sean implementadas e integradas en el mecanismo desarrollado. De esta forma se podría probar el motor de inferencia implementado para la obtención de otro tipo de bases de reglas que sean útiles en otros contextos. Otro de los parámetros que se puede variar en cuanto al motor de minería, es explorar el uso de otras estrategias de definición de etiquetas distintas a las basadas en el contexto a fin de comparar la calidad de la base de reglas obtenida. También, se sugiere realizar diversas pruebas para estudiar el grado en el cual el motor de inferencia del mecanismo propuesto se ve afectado por la presencia de una menor o mayor cantidad de etiquetas difusas versus una menor o mayor proporción de nulos.

AGRADECIMIENTOS

Damos gracias a Aquél que aún en lo más difuso o incierto es capaz de proporcionarnos un orden: “pues Dios no es Dios de confusión, sino de paz.” (1 Corintios 14:33)

REFERENCIAS

- [1] A. Motro, *Management of Uncertainty in Database Systems*, Book Modern database systems ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, 1995.
- [2] E. F. Codd, *Extending the Database Relational Model to Capture More Meaning*, ACM Transactions on Database Systems, vol. 4, no. 4, pp. 397-434, 1979.
- [3] Z. Ma, W.J. Zhang, W.Y. Ma, *Extending the Relational Model to Deal with Probabilistic Data*, Journal of Computer Science and Technology, vol. 15, no. 3, pp. 230-240, May 2000.
- [4] L.B. Othman, S.B. Yahia, *Yet Another Approach for Completing Missing Values*, CLA, vol. 4923 of Lecture Notes in Computer Science, pp. 155-169, Springer, 2006.
- [5] L. Useche, D. Mesa, *Una Introducción a la Imputación de Valores Perdidos*, Terra Nueva Etapa, año/vol. XXII, no. 031, Universidad Central de Venezuela, Caracas, Venezuela, pp. 127-151, 2006.
- [6] W.C. Hou, Z. Zhang, N. Zhou, *Statistical Inference of Unknown Attribute Values in Databases*, in Proceedings of the second International Conference on Information and Knowledge Management, ACM, Washington, DC, USA, pp. 21-30, November 1993.
- [7] J. Díaz, *Comparación entre Árboles de Regresión CART y Regresión Lineal*, Tesis de Maestría, Universidad Nacional de Colombia, Medellín, Colombia, 2012.
- [8] A. Tsanas, A. Xifara, *Accurate Quantitative Estimation of Energy Performance of Residential Buildings Using Statistical Machine Learning Tools*, in Energy and Buildings, vol. 49, pp. 560-567, 2012.
- [9] S.M. Chen, H.R. Hsiao, *A New Method to Estimate Null Values in Relational Database Systems Based on Automatic Clustering Techniques*, Information Sciences: an International Journal, vol. 169, no. 1, pp. 47-69, January 2005.
- [10] J.W. Wang, C.H. Cheng, *An Efficient Method for Estimating Null Values in Relational Databases*, Knowledge and Information Systems: an International Journal, vol. 12, no. 3, pp. 379-394, Agosto 2007.
- [11] K. Pandole, N. Bhargava, *Comparison and Evaluation for Grouping of Null Data in Database Based on K-Means and Genetic Algorithm*, International Journal of Computer Technology and Electronics Engineering (IJCTEE), vol. 2, no. 3, pp. 204-209, June 2012.
- [12] S. Infante, J. Ortega, y F. Cedeño, *Estimación de Datos Faltantes en Estaciones Meteorológicas de Venezuela Via un Modelo de Redes Neuronales*, Revista de Climatología 1578-8768, 2008.
- [13] W. C. Beltran, H. Jaudoin, and O. Pivert, *Estimating Null Values in Relational Databases Using Analogical Proportions*, Information Processing and Management of Uncertainty in Knowledge-Based Systems, Springer International Publishing, Montpellier, France, pp.110-119, July 2014.
- [14] K.N. ElSayed, *Estimating Null Values in Database Using CBR and Supervised Learning Classification*, International Journal of Advanced Computer Science and Applications (IJACSA), vol. 5, no. 6, June 2014.
- [15] D. Otero García, *Imputación de Datos Faltantes en un Sistema de Información sobre Conductas de Riesgo*, Tesis en Master en Técnicas Estadísticas, Universidad de Santiago de Compostela, Galicia, España, 2011.
- [16] A.P. Goicoechea, *Imputación Basada en Árboles de Clasificación*, Eustat Available in: <http://www.eustat.es/documentos/datos/ct>, vol. 4, 2002.
- [17] F. Berzal, I. Blanco, D. Sánchez, and M. I. A. A. Vila, *Measuring the Accuracy and Interest of Association Rules: A New Framework*, Intelligent Data Analysis, vol. 6, no. 3, pp. 221-235, 2002.
- [18] M. Delgado, N. Marín, D. Sánchez, and M.A. Vila, *Fuzzy Association Rules: General Model and Applications*, IEEE Transactions on fuzzy systems, vol. 11, no. 2, pp. 214-225, 2003.
- [19] M. Kryszkiewicz, H. Rybinski, *Incomplete Database Issues for Representative Association Rules*, Lecture Notes in Computer Science, vol. 1609/1999, pp. 583-591, 1999.
- [20] A.A. Chavan, V.K. Verma, *Treatment of Missing Values for Association Rules: A Recent Survey*, International Journal of Computer Applications, vol.70, no. 26, pp.1-4, May 2013.
- [21] L. A. Zadeh, *Fuzzy Sets. Information and Control*, vol. 8, no. 3, pp. 338-353, 1965.
- [22] C. Shyi-Ming, H. Chung-Ming, *A New Approach to Generate Weighted Fuzzy Rules Using Genetic Algorithms for Estimating Null Values*, Expert Systems with Applications, vol. 35, no. 3, pp. 905-917, 2008.
- [23] M.D.R. Jiménez, *Modelado Formal para Representación y Evaluación de Reglas de Asociación*, Tesis Doctoral, Universidad de Granada, España, 2010.
- [24] D. Sánchez, M. Delgado, and M. A. Vila, *A Restriction Level Approach to the Representation of Imprecise Properties*, in Proceeding of IPMU 2008, Málaga, Spain, pp. 153-159, June 2008.
- [25] S. Kannan, R. Bhaskaran, *Association Rule Pruning Based on Interestingness Measures with Clustering*, International Journal of Computer Science Issues (IJCSI), vol. 6, no. 1, November 2009.
- [26] S. Chawla, J. G. Davis, and G. Pandey, *On Local Pruning of Association Rules Using Directed Hypergraphs*, in Proceedings of the International Conference on Data Engineering (ICDE), Boston, MA, USA, vol. 4, pp. 832-841, Marzo-Abril 2004.
- [27] J. Li, H. Shen, and R. Topor, *Mining Informative Rule Set for Prediction*, Journal of Intelligent Information Systems, vol. 22, no. 2, pp. 155-174, March 2004.
- [28] C. Jiménez, H. Álvarez, and L. Tineo, *Context-Dependent Fuzzy Queries in SQLf*, On the Move to Meaningful Internet Systems: OTM 2012, Springer Berlin Heidelberg, pp. 763-779, 2012.
- [29] A.Y. Rodríguez, J.F. González, J.F. Martínez-Trinidad, J.A. Carrasco-Choa, J. R. Shulcloper, *Minería de Reglas de Asociación sobre Datos Mezclados*, Reporte Técnico no. CCC-09-001, Coordinación de Ciencias Computacionales, INAOE, México, 2009.

Un Estudio Empírico sobre el Impacto de la Mutación en la Variedad Genética y la Rata de Convergencia de un Algoritmo Genético Simple

Sandra Lima¹, Minoru Akiyama²
slima@ucla.edu.ve, minorucho@gmail.com

¹ Unidad de Inteligencia Artificial, Universidad Centroccidental Lisandro Alvarado, Barquisimeto, Venezuela

² Departamento de Matemática, Universidad Centroccidental Lisandro Alvarado, Barquisimeto, Venezuela

Resumen: Los Algoritmos Genéticos Simples (AGS), son métodos estocásticos de búsqueda que emulan los principios básicos de la evolución biológica. La dinámica del AGS se centra en los cambios realizados, mediante selección, cruce y mutación, a los genotipos de una población de individuos, los cuales representan posibles soluciones al problema planteado. En consecuencia, estos operadores deben ser cuidadosamente diseñados para mantener un equilibrio entre la variedad genética y la rata de convergencia poblacional. Esta investigación plantea un estudio empírico sobre algunos operadores de mutación no adaptativos y analiza sus efectos en la población, en cuanto a la variedad genética y la rata de convergencia medida en épocas. Se presenta una novedosa pseudo-métrica de Distancia Poblacional entre Generaciones, cuyo método es flexible y dinámico, para evaluar la evolución temporal de la variedad entre poblaciones genéticas. Se excluye el operador de cruce en el proceso evolutivo para estudiar solo el impacto de la mutación en la convergencia y en la variedad poblacional. Luego se experimenta con el algoritmo genético completo para establecer comparaciones entre los resultados obtenidos.

Palabras Clave: Operador Genético; Mutación; Algoritmo Genético Simple; Variedad Genética; Convergencia Poblacional; Distancia Genética.

Abstract: Simple Genetic Algorithms (SGA), are stochastic search methods that emulate the basic principles of biological evolution. Dynamics of SGA focuses on changes made by selection, crossover and mutation, to the genotypes of a population of individuals which represent possible solutions to the problem. Consequently, these operators must be carefully designed to maintain a balance between genetic variation and population convergence. This research presents an empirical study of some non-adaptive mutation operators and analyzes their effects on the population, in terms of genetic variation and rate of convergence measured by epochs. A novel pseudo-metric population distance between generations is presented, whose method is flexible and dynamic, for the evaluation of the evolution of genetic variation among populations. The crossover operator is excluded of the evolutionary process to only study the impact of the mutation on the population convergence and population variety. Then we experiment with the complete genetic algorithm to establish comparisons between the obtained results.

Keywords: Genetic Operator; Mutation; Simple Genetic Algorithm; Genetic Diversity; Population Convergence; Genetic Distance.

I. INTRODUCCIÓN

Los Algoritmos Genéticos Simples (AGS), propuestos en la década de los 60 por John Holland y sus estudiantes [1] mientras buscaban simular sistemas adaptativos biológicos en medios artificiales, representan hoy en día uno de los métodos de computación evolutiva más usados y explorados en la literatura especializada. La dinámica del AGS se centra en los cambios realizados a los genotipos de una población de individuos que representan cada uno una posible solución al problema planteado. Estos cambios se aplican de manera repetitiva a través de los llamados operadores evolutivos o de

cambio como el cruce, la mutación y la selección, creando ciclos conocidos como generaciones o épocas evolutivas.

Debido a que el AGS opera sobre los genotipos de una población, es importante mantener un equilibrio entre la variedad genética y la rata de convergencia poblacional. Esto implica que para que el AGS sea eficiente (y eficaz), los operadores de cambio deben ser cuidadosamente diseñados y/o configurados para evitar los fenómenos conocidos como desviación genética (genetic drift) y convergencia prematura; el primero ocasiona que todo el banco genético de la población avance hacia un mismo patrón (individuo) y el segundo, que la convergencia se 'estaque' en pocas generaciones o épocas

evolutivas, posiblemente en algún óptimo local. Esta investigación plantea un estudio empírico sobre algunos operadores de mutación no adaptativos y analiza sus efectos en la población considerando la variedad genética y la tasa de convergencia medida en épocas.

II. EL OPERADOR DE MUTACIÓN COMO RESPONSABLE DE LA VARIEDAD GENÉTICA

El operador de mutación es la principal herramienta evolutiva en un AGS. Si bien el cruce con recombinación introduce capacidades de adaptación ante un ambiente específico, la mutación es responsable de introducir nuevo material genético en los alelos que serán transmitidos a las nuevas generaciones. Esto puede resultar en un impacto positivo si el fenotipo presenta una mayor adaptación al contexto evaluado, negativo si el resultado produce un menor nivel de adaptación, o neutro si los cambios en el genotipo no alteran el fenotipo y por ende el nivel de adaptación permanece igual. En cualquier caso, es la herramienta necesaria para evitar caer en óptimos locales cuando la diversidad genética del grupo carece del potencial para hacerlo.

Sin embargo, la mutación frecuentemente es considerada como un operador secundario de poca importancia, debido a que tiende a distraer al algoritmo durante la búsqueda de convergencia [2]. Si la tasa de mutación es grande, la diversidad genética resultante ubica los fenotipos de los individuos en regiones dispersas en el espacio solución y por tanto obstruye la convergencia, mientras que una tasa muy pequeña resultará en una exploración muy minuciosa de una región específica y por ende demora la convergencia con la debilidad de caer en la atracción de un óptimo local. Por ello es interesante plantearse cuál sería la tasa de mutación ideal para un tipo de problema dado y la variabilidad que dicha tasa de mutación le introduce a una población genética.

Debido a la naturaleza estocástica de los operadores del AGS y sus relaciones, no es posible construir un modelo determinista y en consecuencia, predictivo del comportamiento del algoritmo frente a una determinada configuración de parámetros y un problema de una naturaleza dada. Sin embargo, los ensayos y pruebas empíricas proporcionan una firme base estadística para comprender mejor estos operadores y la mecánica del AGS, ayudando así al diseño eficiente, formulación y escogencia de parámetros para un problema específico. Se plantea entonces este trabajo de investigación de campo, donde se aísla el efecto de la mutación en la variedad genética y convergencia poblacional de un AGC, para una vez aplicadas las diferentes métricas y los exámenes estadísticos de rigor, formular planteamientos de discusión del desempeño y papel de la mutación en el camino evolutivo.

III. DISEÑO DEL AGS

En las investigaciones de Fogarty y De Jong [3][4] se usó un operador de mutación basado en el original propuesto por Holland en su Algoritmo Genético Simple [1], donde los genes mutantes son escogidos de manera aleatoria mediante un experimento de Bernoulli evaluando todo el cromosoma.

En este caso el AGS está constituido por una población de “n” individuos. Cada individuo de la población se encuentra

representado por un cromosoma binario en código Gray de longitud “m”. La ventaja de utilizar código gray es que mantiene una distancia Hamming entre los cromosomas de manera que pequeños cambios en el genotipo introduzca pequeños cambios en el fenotipo y viceversa [5]. La longitud del cromosoma se divide en partes iguales de acuerdo al número de variables de la función que se desea optimizar; cada segmento del cromosoma representa a una variable en el fenotipo.

El algoritmo funciona como sigue: se decodifica, de Gray a real normalizado, el segmento correspondiente a cada variable. Luego se calcula la función objetivo para asignar el fitness o valor de adaptación al individuo. Los valores fitness de los individuos son utilizados por el operador de selección para elegir dos padres que generan dos hijos en la nueva generación. Está configurado para trabajar estilo “ruleta rusa” [1]. Luego se aplica el operador de cruce en un punto para cada uno de los segmentos [1]. Este paso se omite para los experimentos que aíslan el operador de cruce. Finalmente, se aplica mutación en los bits o alelos del cromosoma seleccionado, de acuerdo a configuraciones que serán detalladas en la siguiente sección. Este proceso se repite por un máximo de generaciones “maxgen” dado.

IV. OPERADORES DE MUTACIÓN NO ADAPTATIVOS

Dos aspectos importantes en la implementación de un operador de mutación para un AGS, son el tipo, modo o forma como se aplicará el cambio al genotipo, y la tasa o frecuencia estadística con la que ocurrirán estos cambios [6]. En este sentido, algunas investigaciones empíricas han sido desarrolladas para profundizar en los efectos de este operador; Fogarty estudió 5 esquemas de tasa de mutación variable [7] en un modelo de quemador industrial.

De Jong [4] mostró que con un tamaño de población pequeño y una alta tasa de mutación, el desempeño del algoritmo mejora durante las primeras generaciones utilizando una métrica que considera la convergencia de la población. Bäck y Schütz [2] [8] en un análisis más reciente argumentaron que para un tipo de problema específico, la tasa de mutación óptima es $1/N$ donde N es el número de bits del cromosoma. Ochoa, Harvey y Buxton [9] obtuvieron el umbral del error a partir del cual la tasa de mutación produce peores soluciones y su incidencia en la presión selectiva.

A continuación se diseñan e implementan varios operadores de mutación no adaptativos, los cuales se diferencian por la cantidad de ruido o cambio que introducen en los cromosomas. Los individuos elegidos para mutar son determinados aleatoriamente por una probabilidad fija de mutación. El cambio introducido en un cromosoma se aplica a segmentos de longitud variable y se cambian los alelos dentro de ese segmento mediante una variable de Bernoulli, la cual retorna un valor booleano dependiendo de la proporción de ruido dada como parámetro.

A. Mutación Clásica

El operador de mutación se aplica a todos los individuos seleccionados para la nueva generación, en alelos escogidos aleatoriamente por la proporción de ruido. A nivel fenotípico, podría afectar a todas las variables de la función objetivo. La inclusión de ruido en un cromosoma está directamente relacionada con la variable proporción de ruido.

B. Muta Uno

En cada cromosoma, muta un único alelo elegido al azar. Se considera el operador con menor inclusión de ruido.

C. Mutan Segmentos Aleatorios

Dentro de cada cromosoma, se aplica ruido a un segmento de longitud aleatoria. Este segmento puede incluir, a nivel fenotípico, una o varias variables de la función objetivo.

D. Muta Variable

Se elige al azar una variable del fenotipo por cada individuo de la población. En el cromosoma, se aplica ruido al segmento que representa dicha variable.

V. MÉTRICAS DE EVALUACIÓN PARA LA CONVERGENCIA Y VARIEDAD POBLACIONAL

A. Rata de Convergencia

La convergencia poblacional se medirá en términos del fitness promedio por generación a lo largo de todos los experimentos. Como el promedio se ve afectado por valores extremos encontrados por el proceso de mutación, también se calculará la mediana poblacional como el valor central que reúne el 50% de los individuos y fija la tendencia de la población hacia el óptimo. Además, se considerará el valor mínimo de mediana por experimento que con un error máximo dado, reúna al menos al 25% del total de generaciones.

B. Variedad Poblacional

1) *Distancia Hamming entre Vecinos*: Se calculará la métrica de Distancia Haming Promedio entre vecinos de la población genética para estudiar la variedad poblacional. La relación entre vecinos es circular, es decir, se asume que el vecino del último individuo es el primero de la población.

2) *Cantidad Promedio de Individuos Diferentes*: La cantidad de individuos diferentes por generación puede describir la tendencia de la población hacia la convergencia. Un individuo se diferencia de otro en al menos un alelo.

3) *Distancia Poblacional entre Generaciones*: Se plantea una medida poblacional basada en distancia entre conjuntos en un espacio métrico [10], donde los elementos son los individuos de la población actual, representados en su forma binaria como la matriz A, la cual es comparada con la población de la generación anterior denotada como la matriz B. Cuando la distancia es menor más parecidas son las poblaciones y viceversa.

Por métrica se tiene una función $d: \zeta_m \times \zeta_m \rightarrow R$ que cumple las siguientes propiedades:

- $\forall a, b \in \zeta_m, d(a, b) \geq 0$
- $\forall a, b \in \zeta_m, d(a, b) = 0 \leftrightarrow a = b$
- $\forall a, b \in \zeta_m, d(a, b) = d(b, a)$
- $\forall a, b, c \in \zeta_m, d(a, c) \leq d(a, b) + d(b, c)$

Donde ζ_m representa el espacio de cromosomas. Algunas métricas que se pueden usar en ζ_m son las siguientes:

- *Métrica de Hamming Normalizada*

$$d_H(a, b) = \frac{1}{m} \sum_{j=0}^{m-1} \delta(a_j, b_j), \forall a, b \in \zeta_m$$

$$\text{Donde, } \delta(a_j, b_j) = \begin{cases} 0 & \text{si } a_j = b_j \\ 1 & \text{si } a_j \neq b_j \end{cases},$$

$$a = a_0, \dots, a_{m-1}; b = b_0, \dots, b_{m-1} \quad (1)$$

- *Métrica Trivial*

$$d_T(a, b) = \begin{cases} 0 & \text{si } a = b \\ 1 & \text{si } a \neq b \end{cases}, \forall a, b \in \zeta_m \quad (2)$$

- *Métrica de Cantor Normalizada*

$$d_C(a, b) = \frac{1}{2 \times (1 - \frac{1}{2^m})} \sum_{j=0}^{m-1} \frac{\delta(a_j, b_j)}{2^j}, \quad \forall a, b \in \zeta_m \quad (3)$$

Las métricas que aportan más al estudio de la variedad genética a lo largo de las generaciones son las métricas de Cantor y de Hamming. La métrica de Cantor introduce una variación en la distancia de Hamming en cuanto a que le da menor peso a las diferencias de genes a medida que estos ocupan una posición más a la derecha de los cromosomas.

En el caso de funciones objetivo multivariables cada cromosoma del AGS está compuesto por un número finito de subcromosomas $a = a_{(1)}, a_{(2)}, \dots, a_{(l)}$ concatenados en ese orden. En este caso se denomina $\zeta_m = \zeta_{m_1+m_2+\dots+m_l}$ a la colección de todos los cromosomas de este tipo para hacer énfasis en que un cromosoma esta subdividido en l subcromosomas, el primero de longitud m_1 , el segundo de longitud m_2 y así sucesivamente. En este trabajo, los subcromosomas tienen igual longitud.

Considerando esta concatenación se puede diseñar nuevas medidas de distancia para el cromosoma a partir de la combinación de diferentes métricas para cada segmento, según las características del espacio solución. Una forma sencilla de calcular la distancia global d_g entre dos cromosomas sería calculando un promedio de distancias entre sus segmentos:

$$d_g(a, b) = \frac{d_x(a_{(1)}, b_{(1)}) + \dots + d_x(a_{(l)}, b_{(l)})}{l} \quad (4)$$

Donde d_x es la métrica de distancia individual elegida para cada segmento. También se podría trabajar en función de los máximos o mínimos entre las distancias individuales. La función d_g es una métrica y el par (ζ_m, d_g) es un espacio métrico completo y compacto. Llamaremos al espacio (ζ_m, d_g) espacio de g-cromosomas.

a) Población de Cromosomas

Se tiene una población de n individuos y a cada individuo del AGS le corresponde un cromosoma de longitud m . Esta población se representa por una matriz $A_{n \times m}$ donde cada componente o alelo es cero (0) o uno (1).

Denotemos por $M_{n \times m}$, al conjunto de todas las matrices de tamaño $n \times m$ compuestas por cromosomas en ζ_m . El interés se centra en construir una métrica adecuada que describa cuando dos poblaciones $A, B \in M_{n \times m}$ son cercanas o distantes. Tal métrica debe tomar en cuenta las siguientes consideraciones:

- La distancia entre A y B debe ser definida comparando los individuos, es decir, las filas de A con las filas de B.

- Debe tomarse en cuenta que los individuos de una población pueden cambiar de lugar sin alterar a la población. Por ejemplo, $A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$ y $B = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ son dos matrices muy distintas que representan a la misma población. Por tanto, la métrica debe tomar en cuenta el orden de las filas de una matriz.
- El número de veces que cada cromosoma se repite en una población debe ser considerado. No es lo mismo la población dada por:

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \text{ que la dada por: } B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Un censo poblacional permite establecer estas diferencias entre poblaciones [10]. Si $A \in M_{n \times m}$ representa una población, se extiende dicha matriz de manera que a cada fila $A^{(r)}$ se le agrega un nuevo subcromosoma $A_{(l+1)}^{(r)}$ de longitud m_{l+1} , siendo l la cantidad de subcromosomas que representan el problema a resolver. Este subcromosoma puede representar en código binario un valor según los siguientes criterios:

- El número de veces que la fila $A^{(r)}$ se encuentra repetida en la matriz A.
- El número de veces que la fila $A^{(r)}$ se encuentra repetida en la matriz B.

El valor resultado se encuentra acotado en $[0, n]$. En consecuencia, la longitud m_{l+1} debe ser escogida de forma tal que permita su representación en binario. Utilizando el ejemplo anterior, con $m_{l+1} = 3$, al aplicar el censo según el criterio 'i' las matrices A y B se convierten en:

$$A_c = \begin{bmatrix} 1 & 1 & 0 & 0 & | & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & | & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & | & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & | & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & | & 1 & 0 & 0 \end{bmatrix} \text{ y } B_c = \begin{bmatrix} 1 & 1 & 0 & 0 & | & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & | & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & | & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & | & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & | & 1 & 0 & 0 \end{bmatrix}$$

Aplicando el censo, A_c y B_c contienen una mejor información sobre la población que las dadas por las matrices A y B. Igualmente en las poblaciones censadas, la distancia entre individuos puede calcularse siguiendo las métricas antes descritas.

b) Pseudo-Métrica de Distancia Poblacional entre Generaciones

Sean las matrices $A, B \in M_{n \times m}$ y sus respectivas matrices censadas A_c y B_c . Llamaremos distancia dirigida de A hacia B, al valor:

$$\vec{D}(A, B) = \max_{1 \leq r \leq n} (\min_{1 \leq s \leq n} d_g(A_c^{(r)}, B_c^{(s)})) \quad (5)$$

Se observa que para r fijo, $\min_{1 \leq s \leq n} d_g(A_c^{(r)}, B_c^{(s)})$ mide la distancia que hay entre el cromosoma del individuo r en la población A con el individuo de la población B que más se le parece. Así, $\vec{D}(A, B)$ determina que tan diferente es la población de A con respecto a B. Esta relación no es

conmutativa ya que no se cumple que los individuos más cercanos a la población A, sean los mismos cuando se calcula \vec{D} en la población B. Formalmente se define la pseudo-métrica de distancia poblacional $d: M_{n \times m} \times M_{n \times m} \rightarrow \mathbb{R}$ como la función dada por:

$$d(A, B) = d(B, A) = \max\{\vec{D}(A, B), \vec{D}(B, A)\} \quad (6)$$

$$\forall A, B \in M_{n \times m}$$

Es una pseudo-métrica ya que las poblaciones A y B pueden ser distintas y sin embargo $d(A, B) = 0$. Aun en este caso, aunque $A \neq B$ las poblaciones que ellas representan son exactamente las mismas, solo habría que permutar las filas de A para obtener las filas de B. Dos individuos serán cercanos si son parecidos en el genotipo y si se repiten de forma semejante en la población censada.

Se aplicará esta métrica de distancia a las poblaciones formadas por la generación anterior y la nueva a lo largo del proceso evolutivo.

VI. EXPERIMENTACIÓN

Los experimentos comienzan con una población inicializada aleatoriamente. Se realizan 05 experimentos de 400 generaciones cada uno para una población de 30 individuos. Los individuos están compuestos por un cromosoma de longitud 32 en un contexto de minimización. Adicionalmente, se incluye un sub-cromosoma Gray de longitud 07 para el cálculo de las matrices censadas según el criterio 'i'. Los parámetros restantes son decididos en tiempo de ejecución. Entre ellos están: el tipo de mutación, acotamiento de valor para las variables de la función objetivo a optimizar, probabilidad de cruce, probabilidad de mutación y proporción de ruido introducido en los cromosomas mutados. Una probabilidad de cruce con valor 0 es utilizado para excluir dicho operador del proceso evolutivo y estudiar únicamente el impacto de la mutación en la convergencia y en la variedad poblacional. También se utiliza el algoritmo genético completo para establecer comparaciones entre los resultados obtenidos.

Para el cálculo de la distancia poblacional entre generaciones, se aplica la métrica de Cantor normalizada (3) en cada uno de los segmentos del cromosoma y la distancia entre individuos viene representada por el promedio resultante. En el caso de la convergencia, además del fitness y mediana promedio por generación, en cada experimento se cuenta el valor mínimo de la mediana que con un error máximo de 0.001 reúna al menos al 25% del total de generaciones. En este caso se podría concluir que el algoritmo ha convergido a dicho valor. Se eligieron 03 funciones bidimensionales en un contexto de minimización:

- $f_1(x, y) = (x - 2)^2 + (y - 2)^2 + 1$. Esta función objetivo tiene forma parabólica y tiene un mínimo conocido para $x = 2, y = 2, f_1(x, y) = 1$.
- $f_2(x, y) = x^2 \text{Sen}(x) + y^2 \text{Sen}(y) + 4$. Es una función multimodal con varios puntos máximos. Alcanza su mínimo en $f_2(x, y) = -44,53$.
- $f_3(x, y) = (xy(1 - x)(1 - y)\sin(9\pi x)\sin(9\pi y))^2$. Esta función muestra una malla interesante donde hay

muchos valores de (x,y) que minimizan la función en cero (0).

Se consideraron 04 casos experimentales donde se aplican los 04 operadores de mutación no adaptativos descritos en la sección IV:

- a) Baja probabilidad de mutación con una conservadora inclusión de ruido en los cromosomas mutados.
- b) Baja probabilidad de mutación con una alta inclusión de ruido en los cromosomas mutados.
- c) Alta tasa de mutación con una conservadora inclusión de ruido en los cromosomas mutados.
- d) Alta tasa de mutación con alta inclusión de ruido en los cromosomas mutados.

Para referirse a un experimento específico se tomará por ejemplo la notación '1a' para indicar que corresponde a la función objetivo Nro. 1 en el caso a. A continuación las discusiones de cada caso.

A. Caso a

Se probó $f_1(x,y)$ con $(x,y) \in [-2,6]$, $p_{cruce} = 0.8$, $p_{mutac} = 0.05$, $ruidomutac = 0.3$.

Los resultados indican que no hay una diferencia marcada entre aplicar el algoritmo genético completo y aislar el operador de mutación ya que la tasa es muy pequeña y el ruido introducido también lo es. La distancia poblacional muestra cambios de distancia de una generación a otra con una amplitud aproximada menor o igual a 0.1. La variabilidad entre individuos es relativamente constante.

En cuanto a la distancia dirigida, la población final 'F' se parece más a la inicial 'I' que al contrario y esto se corrobora a lo largo de los todos casos tanto si se aísla la mutación como si se considera el AGS completo. Es un resultado interesante ya que a nivel biológico esto consecuencia natural del proceso evolutivo (Figura 1). Por los resultados de la mediana se considera que el algoritmo ha convergido y que el cruce beneficia la variabilidad por encima de la mutación.

B. Caso b

Se probó f_1 con $(x,y) \in [-2,6]$, f_2 con $(x,y) \in [-6,8]$, $p_{cruce}=0.8$, $p_{mutac}=0.05$, $ruidomutac=0.6$ y f_3 con $(x,y) \in [-6,6]$, $p_{cruce}=0.8$, $p_{mutac}=0.05$, $ruidomutac=0.7$. Para el caso 1b se observa que hay mayor variabilidad en el cálculo del fitness y mediana promedio con respecto al caso a (Figuras 2 y 3). Esto se debe a que los individuos elegidos para mutar contienen mayor cantidad de ruido. El operador 'muta variable' se encuentra alejado de la tendencia global hacia el óptimo y los que más se acercaron fueron 'muta uno' y 'segmentos aleatorios'. El cálculo de la mediana demuestra la convergencia del algoritmo con resultados similares al caso 1a (Tabla I).

En el caso 2b se observan resultados muy similares al caso 1b en relación a la tendencia de los operadores. En cuanto al cálculo de la mediana los operadores que muestran mayor tendencia al óptimo fueron 'segmentos aleatorios', 'muta uno' y 'mutación clásica', aunque no se alcanza la convergencia en todos los experimentos (Tabla II). Considerando solo mutación, el comportamiento de cada operador, a pesar de no ser el ideal, se mantiene cercano al mínimo global. El

operador 'muta variable' continúa alejado de la tendencia global hacia el óptimo (Figura 4).

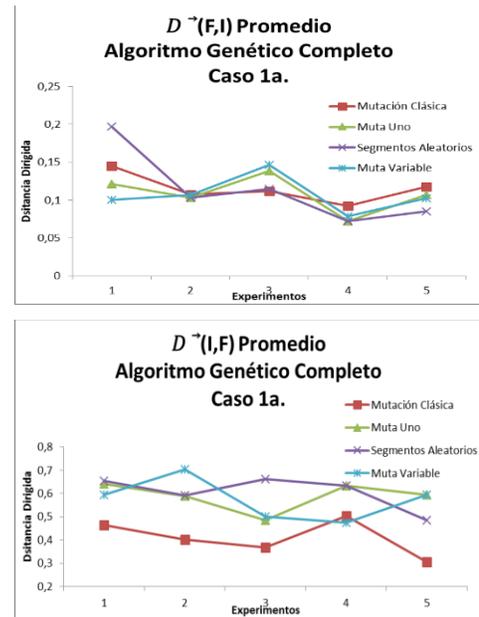


Figura 1: Distancia Dirigida Promedio entre las Poblaciones I y F Caso 1a

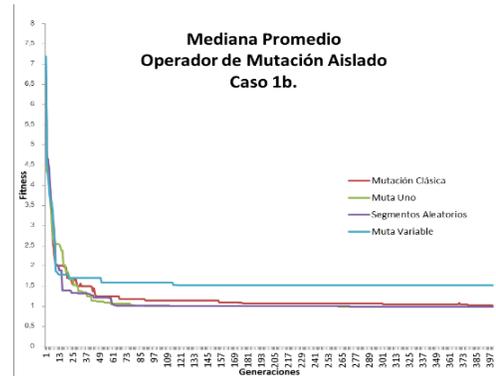


Figura 2: Mediana Promedio - Caso 1b

Tabla I: Mediana Mínima - Caso 1b

Exp	Mutación Clásica	Muta Uno	Segmentos Aleatorios	Muta Variable
1	1,00194	1,000067	1,000005	1,065322
2	1,000618	1,00026	1,000391	2,137649
3	1,001181	1,000166	1,001044	1,083951
4	1,008906	1,000593	1,0004	1,036116
5	1,004692	1,000286	1,000093	1,077673

En caso 3b la mediana refleja convergencia del algoritmo, donde la mutación aislada obtuvo mejores resultados (Tablas III y IV). Sin embargo, por ser una función de superficie oscilante, el fitness promedio refleja fluctuaciones importantes en los operadores 'mutación clásica' y 'segmentos aleatorios' (Figura 5).

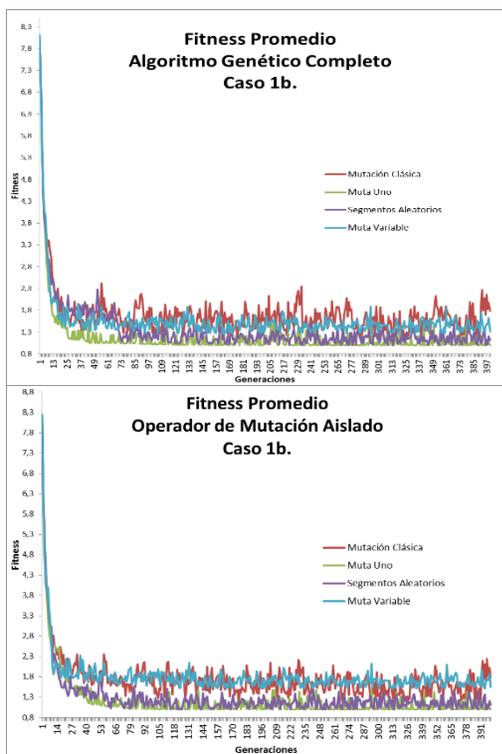


Figura 3: Fitness Promedio - Caso 1b

Tabla II: Mediana Mínima - Caso 2b

Exp	Mutación Clásica	Muta Uno	Segmentos Aleatorios	Muta Variable
1	-43,599379	-44,16573	-43,520587	-43,488156
2	-43,869805	-44,16526	-44,165029	-38,996075
3	-43,816227	-44,16496	-44,165046	-42,113712
4	-43,845938	-44,16576	-44,165408	-36,265633
5	-44,043416	-44,16523	-24,028261	-43,700752

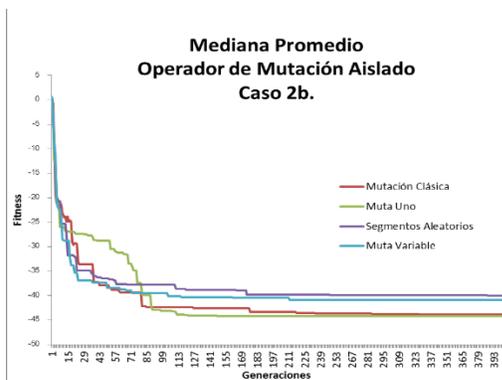


Figura 4: Mediana Promedio - Caso 2b

Tabla III: Mediana Mínima: AGS - Caso 3b

Exp	Mutación Clásica	Muta Uno	Segmentos Aleatorios	Muta Variable
1	0,000588	0,000635	0,000077	0,000538
2	0,000761	0,00054	0,000177	0,000708
3	0,000245	0,000552	0,000374	0,000393
4	0,000852	0,000922	0,000757	0,00051
5	0,000395	0,000151	0,00031	0,000016

Tabla IV: Mediana Mínima: Operador de Mutación Aislado - Caso 3b

Exp	Mutación Clásica	Muta Uno	Segmentos Aleatorios	Muta Variable
1	0,000357	0,000358	0,000068	0,000267
2	0,000015	0,000011	0,000146	0,000015
3	0,000342	0,000342	0,000342	0,000342
4	0,000026	0,000207	0,000111	0,000821
5	0,000004	0,000807	0,000081	0,000594

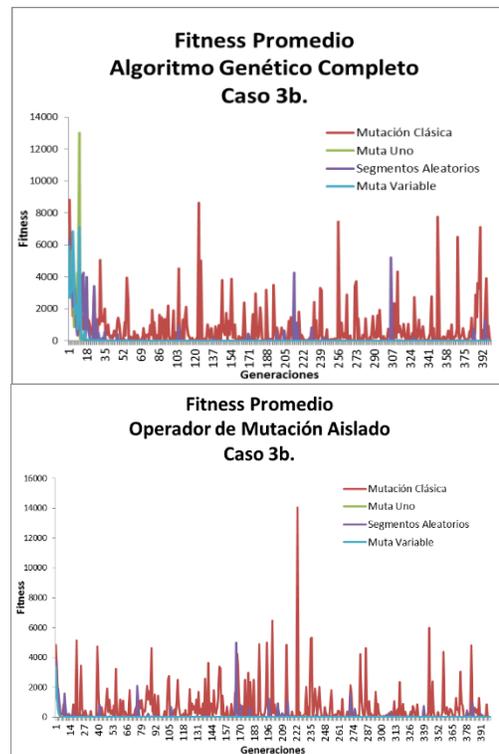


Figura 5: Fitness Promedio - Caso 3b

En la Figura 6 la distancia poblacional aumentó su amplitud en 0.03 aproximadamente con respecto al caso 1a. En caso de 2b la dispersión se mantuvo de forma similar a 1b para los distintos operadores. En el caso 3b la dispersión fue mayor por el aumento del ruido en los cromosomas mutados. Esto indica que a nivel genético la variedad se mantiene pese al cambio de función objetivo. Para el caso b, como lo muestra por ejemplo la Figura 7, la tendencia evolutiva de la distancia dirigida entre poblaciones 'I' y 'F' es similar a la observada en el caso a.

La distancia Hamming promedio entre vecinos de la Figura 8 muestra mayores fluctuaciones que el caso a por el ruido introducido en los cromosomas mutados. Del mismo modo, en el caso 2b y 3b la distancia Hamming tiende a disminuir a medida que transcurren las generaciones, siendo los operadores 'muta uno' y 'muta variable' los de menor distancia entre vecinos.

En la Figura 9 los operadores 'muta uno' y 'muta variable' se aprecian más compactos en sus valores promedios. 'segmentos aleatorios' y 'mutación clásica' exhiben mayor cantidad promedio de individuos diferentes. Estos resultados son parecidos en los casos 2b y 3b. Con el operador de mutación aislado los individuos tienden a ser más parecidos en su genotipo; en este caso, una tasa de cruce alta, introduce mayor variabilidad que la proporción de ruido.

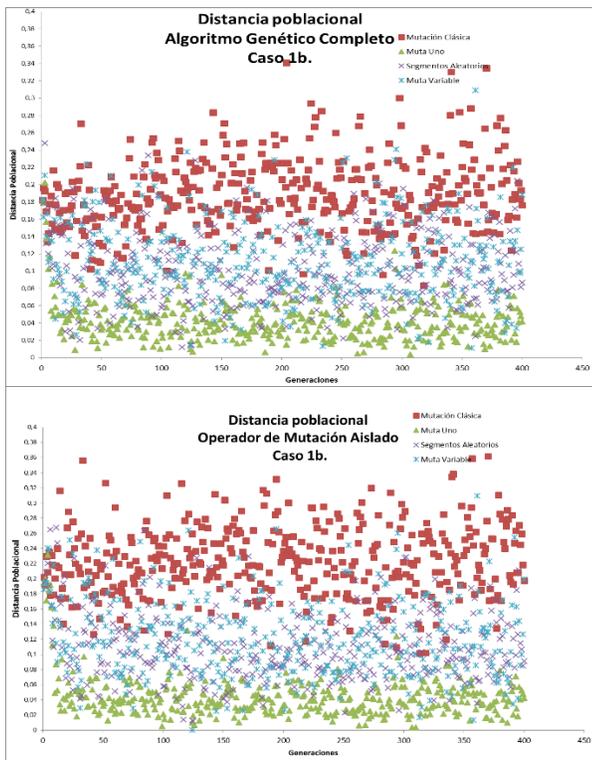


Figura 6: Distancia Poblacional Promedio - Caso 1b

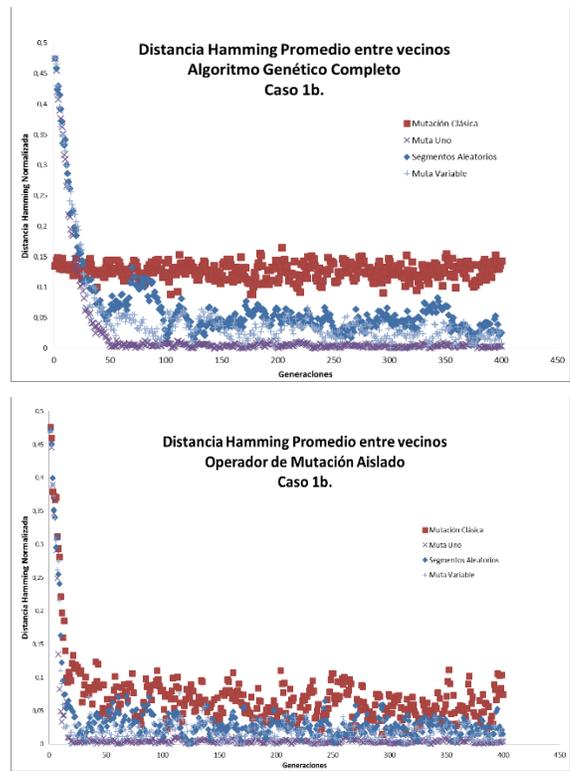


Figura 8: Distancia Hamming Promedio entre Vecinos - Caso 1b

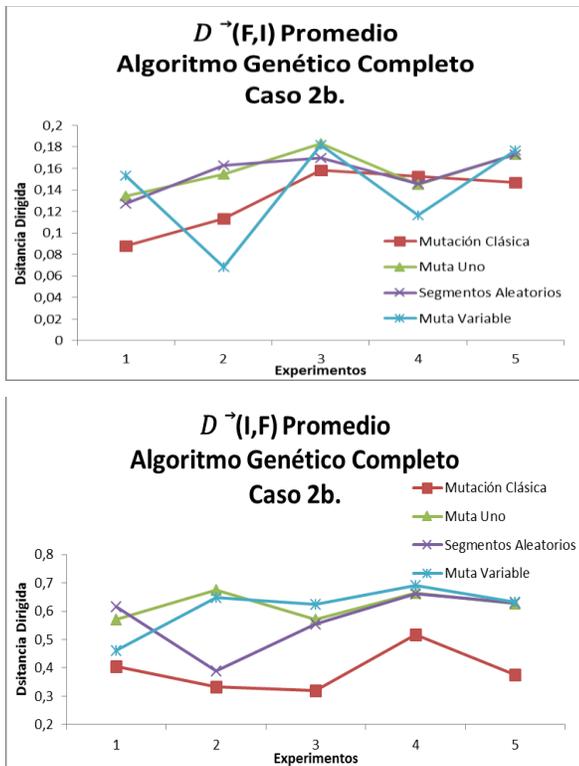


Figura 7: Distancia Dirigida Promedio entre las Poblaciones I y F. Algoritmo Genético Completo - Caso 2b

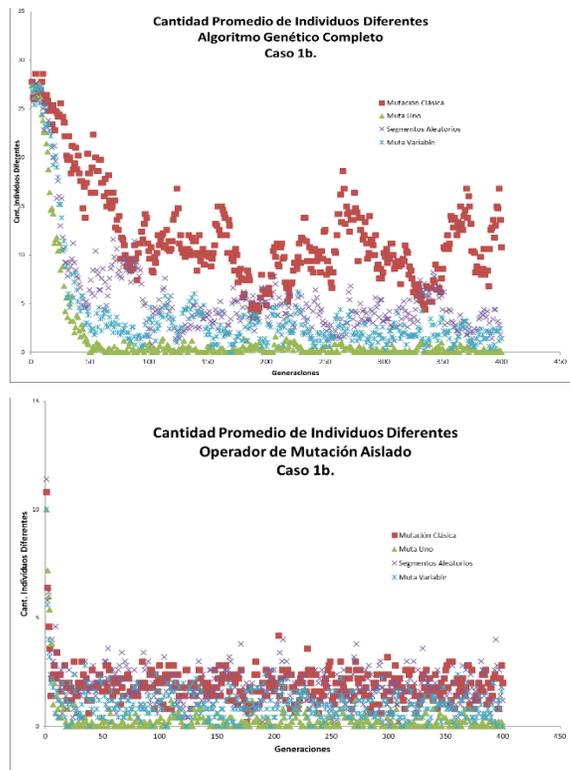


Figura 9: Cantidad Promedio de Individuos Diferentes - Caso 1b

C. Caso c

Se probó $f_1(x,y)$ con $(x,y) \in [-2,6]$, $f_2(x,y)$ con $(x,y) \in [-6,8]$, $p_{cruce} = 0.8$, $p_{mutac} = 0.3$, $ruidomutac = 0.2$. Los expertos recomiendan utilizar una tasa de mutación muy pequeña menor a 0.05, por tanto se considera 0.3 una alta tasa de mutación.

En la Figura 10 se observa que los operadores 'segmentos aleatorios' y 'muta uno' obtuvieron una tendencia hacia el óptimo pero con fluctuaciones. El operador de peor comportamiento fue 'mutación clásica' ya que hay aproximadamente un 30% de individuos elegidos que un 20% de probabilidad de que cada alelo mute, lo cual produce mayor cantidad de ruido poblacional que en consecuencia, lo aleja del promedio hacia el óptimo. En este caso tampoco ayuda a la convergencia, la variabilidad adicional introducida por el operador de cruce; por ello no se reportan los resultados del AGS completo. En términos generales tienen tendencia hacia el óptimo pero no se alcanza la convergencia deseada.

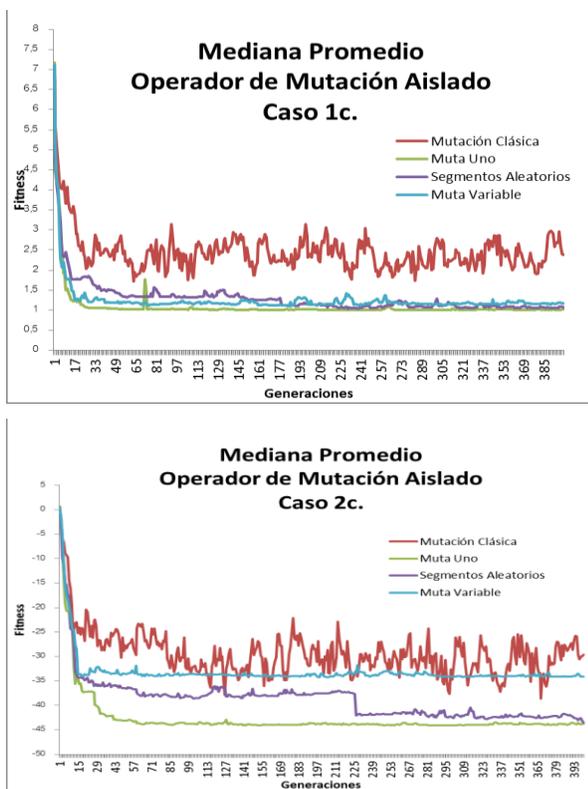


Figura 10: Mediana Promedio - Caso c

En la Figura 11 la distancia poblacional tiene un comportamiento más compacto que el mostrado en el caso b, y el operador 'mutación clásica' imprime mayor distancia entre poblaciones. La distancia dirigida entre las dos poblaciones se mantuvo relativamente similar en comparación con el caso b (Figura 12).

La distancia Hamming promedio entre vecinos de la Figura 13 muestra menos fluctuaciones que en el caso 1b y con mayor parecido al caso 1a por el ruido introducido en los cromosomas mutados. Pese a que tiene alta tasa de mutación, las fluctuaciones son más suaves tendientes a la disminución. Esto indica que la variedad genética se puede acotar, sin

importar la función objetivo, manteniendo constantes las demás condiciones.

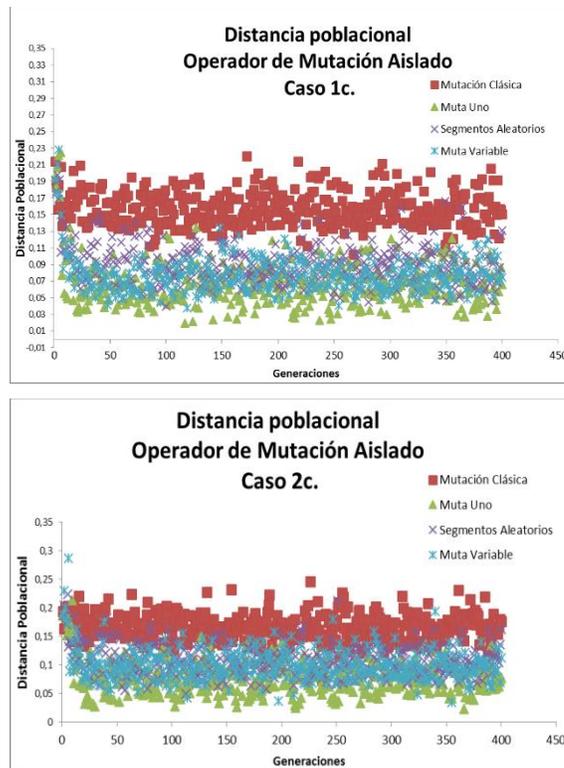


Figura 11: Distancia Poblacional Promedio - Caso c

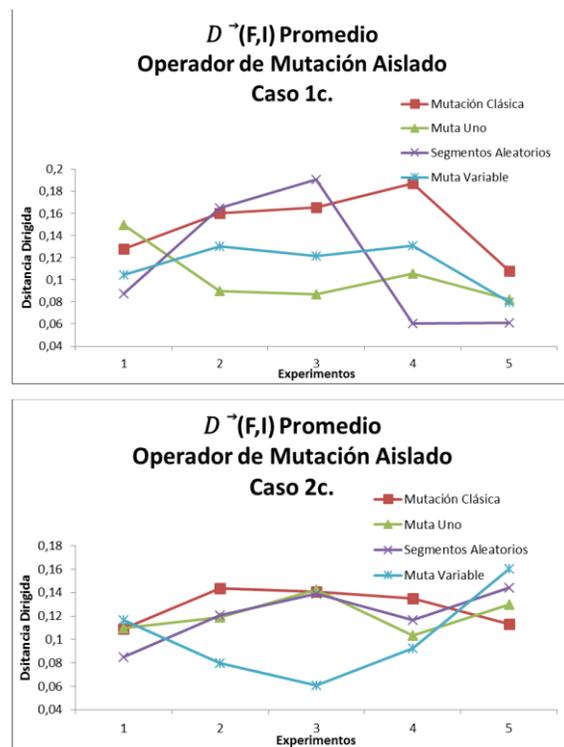


Figura 12: Distancia Dirigida Promedio Población F a I - Caso c

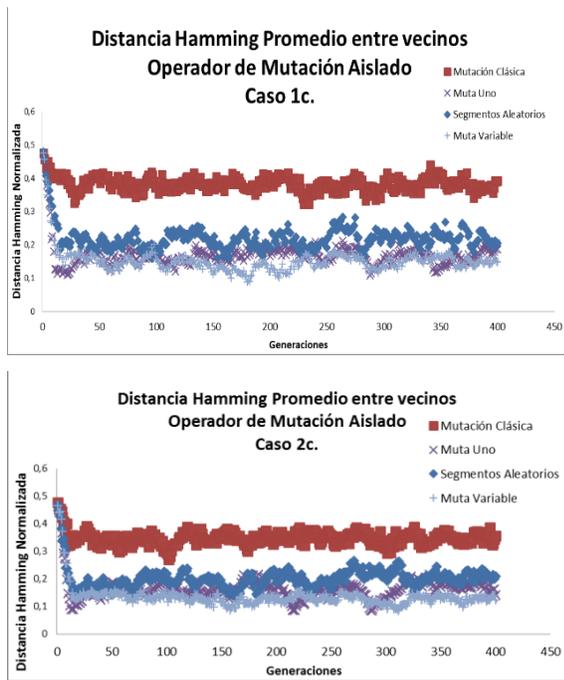


Figura 13: Distancia Hamming Promedio entre vecinos - Caso c

Comparando con las gráficas del caso b, los operadores de la Figura 14 contienen individuos más separados oscilando entre 5 y 15, observándose incluso más dispersión en el AGS completo. En promedio la ‘mutación clásica’ sigue exhibiendo mayor cantidad de individuos diferentes.

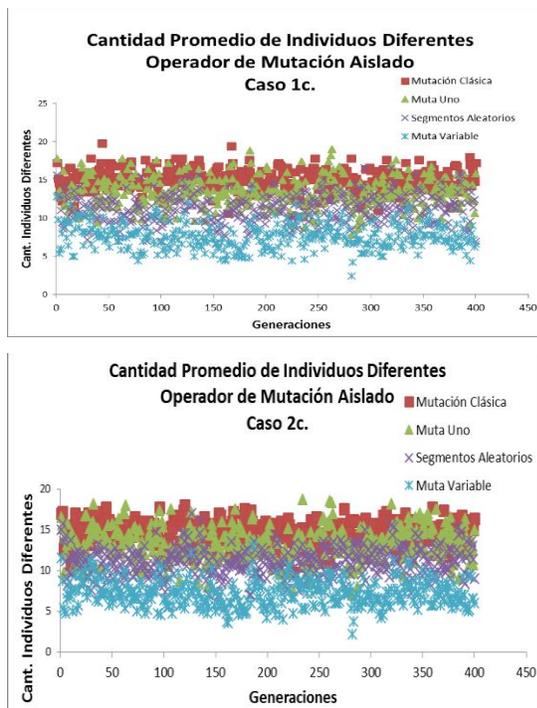


Figura 14: Cantidad Promedio de Individuos Diferentes - Caso c

D. Caso d

Se probó f_1 con $(x,y) \in [-2,6]$, $p_{cruce}=0.8$, $p_{mutac}=0.5$, $ruidomutac=0.7$ y f_3 con $(x,y) \in [-6,6]$, $p_{mutac}=0.3$, $ruidomutac=0.7$.

La distancia Hamming entre individuos es más alta en promedio que en los casos anteriores (entre 0.2 y 0.5), y lo mismo sucede en el cálculo de la cantidad de individuos diferentes: entre 5 y 30 individuos diferentes para las funciones estudiadas. La distancia poblacional se mantuvo en un rango constante por el hecho de que los alelos continuamente cambian de valor y hace que los individuos se mantengan a una misma distancia entre sí (Figura 15).

Para el cálculo de la convergencia, los operadores que lograron acercarse al mínimo óptimo fueron los que introducían menos variación: ‘muta uno’ y ‘muta variable’, pero en términos generales el algoritmo no convergió demostrando que altas tasas de mutación combinadas con una alta proporción de ruido convierten al algoritmo en una búsqueda prácticamente aleatoria.

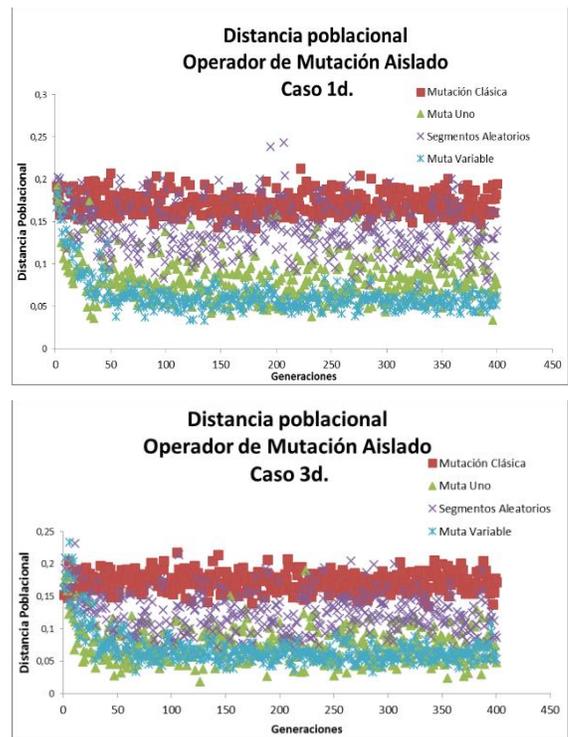


Figura 15: Distancia Poblacional Promedio - Caso d

VI. CONCLUSIONES

Se compararon los resultados de diferentes operadores de mutación mediante la aplicación de medidas que estudian la variedad genética y la convergencia de poblaciones compuestas por alelos binarios. Se diseñó una novedosa pseudo-métrica de distancia poblacional entre Generaciones, cuyo método es flexible y dinámico, para evaluar empíricamente la evolución temporal de la variedad entre poblaciones, lo cual podría extenderse a cromosomas no binarios.

En el estudio de variabilidad genética, los resultados indican que no hay una diferencia significativa entre tener o no el operador de cruce para tasas de mutación y ruido bajas. En el caso a, una alta proporción de cruce beneficia la variabilidad por encima de la mutación ya que ésta última se presenta en muy pocos individuos de la población y muta pocos alelos del cromosoma. En todas las funciones objetivo se logró la convergencia.

Con altas tasas de mutación y ruido (caso d) el algoritmo se convierte en una búsqueda prácticamente aleatoria y difícilmente se alcanza el criterio de convergencia. Queda al margen lo que puedan hacer el operador de selección y cruce.

Por otro lado, los resultados presentados por el operador de mutación aislado del proceso evolutivo con una tasa medianamente alta, el comportamiento de cada operador, a pesar de no ser el ideal, se mantiene alrededor de un mínimo local cercano al óptimo. Esto sugiere, coincidiendo con De Jong [6], un algoritmo genético que al principio sólo contemple mutación y al obtener un espacio solución candidato, incorpore el cruce y disminuya la mutación, para lograr una convergencia más rápida y dar soluciones a diferentes problemas de optimización considerados intratables. Por otro lado, manteniendo el operador de mutación aislado, se observó menor cantidad de individuos diferentes concentrados en un rango menor.

En cuanto a la pseudo-métrica de distancia poblacional dirigida, los resultados muestran que la población final 'F' se parece más a la inicial 'I' que al contrario. Se puede interpretar como una consecuencia del proceso evolutivo al que es sometida la población. En el cálculo de la distancia dirigida entre las poblaciones 'I' y 'F', se observó que a mayor proporción de ruido menores fluctuaciones se presentan en cada uno de los operadores.

Para el caso de altas tasas de mutación y ruido la distancia poblacional global es más compacta, a diferencia de bajas tasas de mutación donde la distancia poblacional es mucho más dispersa. Las sucesivas mutaciones hacen que la población vuelva a una configuración anterior que mantiene una distancia global y entre individuos relativamente estable. Con estos resultados, es posible establecer una cota de variabilidad genética, para cualquier función objetivo, manteniendo constantes ciertas condiciones.

A continuación se plantean conclusiones para cada uno de los operadores.

Mutación Clásica: A pesar de introducir cambios importantes en el cromosoma (reflejado en el fitness promedio) la mediana refleja cierta tendencia de la población hacia un mínimo, lo que hace que, por un lado converja, y por otro que explore nuevas posibilidades en busca del óptimo. Con una tasa de mutación medianamente alta y ruido conservador el operador mutación clásica por sí solo, exhibe buenos resultados en el proceso evolutivo. Posteriormente, hace falta el operador cruce para acercar la solución al óptimo.

En general se ve afectado por la cantidad de ruido en los cromosomas mutados y por ello no logra en varios casos alcanzar la convergencia utilizando la mediana mínima por experimento. Los resultados indican que los individuos se mantienen con una variabilidad relativamente constante y menor en comparación a los demás operadores.

Muta Uno: Tiene un buen desempeño en términos generales y en casi todos los casos alcanza la convergencia. Presenta el peligro de caer en un mínimo local por ser el operador de

menor ruido en los cromosomas mutados, como lo reflejan los resultados de Mediana Mínima por caso. En cuanto a la convergencia, tiene valores de mediana y media bastante cercanos al óptimo y en cuanto a variedad genética, en términos generales, es el operador de menor distancia poblacional global, distancia Hamming entre vecinos, cantidad promedio de individuos diferentes. Cuando existe una tasa alta de mutación, es similar al operador 'muta variable' en las medidas de variedad genética.

Muta Variable: Presenta un pobre desempeño por afectar solo una dimensión del espacio solución; esto hace cambiar la búsqueda en muchas direcciones sin lograr la convergencia.

Segmentos Aleatorios: Es una mutación intermedia entre muta uno y mutación clásica, lo cual produce buenos resultados en la convergencia y no tiene alto impacto en los individuos de la población cuando se aplica cierta cantidad de ruido. Mantiene un espaciamiento relativamente constante en los genotipos de los individuos. Tiene comportamiento similar a 'mutación clásica' en el cálculo de la media y la mediana. En cuanto a la variedad genética y distancia poblacional, con tasas de mutación conservadoras, exhibe un comportamiento similar a 'muta variable'. Para alta cantidad de ruido, el comportamiento en cuanto a variedad genética es similar a 'mutación clásica'. Los resultados de Mediana Mínima por experimento reflejan que para una tasa baja, el operador es robusto al alcanzar la convergencia.

En trabajos futuros se pretende utilizar análisis espectral que arroje indicadores que permitan medir la variedad poblacional y comparar estos resultados con operadores de mutación adaptativos. Además extender las métricas a poblaciones no binarias.

REFERENCIAS

- [1] J. H. Holland, *Outline for a Logical theory of Adaptive Systems*. Journal of the ACM (JACM), vol. 9, no. 3, pp. 297–314, 1962.
- [2] T. Bäck and M. Schütz, *Intelligent Mutation Rate Control in Canonical Genetic Algorithms* in Foundations of Intelligent Systems, Springer, pp. 158–167, 1996.
- [3] T. C. Fogarty, *Varying the Probability of Mutation in the Genetic Algorithm* in Proceedings of the third International Conference on Genetic Algorithms, pp. 104–109, 1989.
- [4] K. A. De Jong, *Analysis of the Behavior of a Class of Genetic Adaptive Systems*. 1975.
- [5] F. Gray, *Pulse Code Communication*. Mar, 1953.
- [6] D. E. Goldberg, *Genetic algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, vol. 1989.
- [7] T. C. Fogarty, *Varying the Probability of Mutation in the Genetic Algorithm* in Proceedings of the Third International Conference on Genetic Algorithms, pp. 104–109, 1989.
- [8] T. BVack, *Optimal Mutation Rates in Genetic Search* in Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 2–8, 1993.
- [9] G. Ochoa, I. Harvey, and H. Buxton, *Optimal Mutation Rates and Selection Pressure in Genetic Algorithms* in GECCO, pp. 315–322, 2000.
- [10] A. Papadopoulos, *Metric Spaces, Convexity and Nonpositive Curvature*. European Mathematical Society, vol. 6, 2005.

Shader Tool: Un Ambiente Web de Desarrollo Integrado para GLSL

Luiyit Hernández¹, Esmitt Ramírez¹
luiyit.hernandez@gmail.com, esmitt.ramirez@ciens.ucv.ve

¹ Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: Un shader se define como una unidad de programa escrita en un lenguaje de sombreado para la GPU, con la finalidad de producir imágenes (*render*). Actualmente, existen herramientas y tecnologías Web que realizan el despliegue de este contenido en navegadores basados en HTML5. Por ello, se ha impulsado el desarrollo de soluciones que apoyan la implementación de programas de shaders para el lenguaje GLSL, parte del estándar OpenGL. Sin embargo, de acuerdo a nuestra investigación, no existen aplicaciones Web disponibles con un criterio unificado y con todos los elementos necesarios involucrados en el proceso de desarrollo. En este trabajo se presenta una solución llamada Shader Tool que contiene un conjunto de herramientas que incorporan y enlazan todos los elementos requeridos para el desarrollo de shaders. La solución emplea plantillas, gráficos 3D, interactividad y tecnología de sincronización para la implementación de código GLSL. De acuerdo a los resultados obtenidos en las pruebas realizadas, se determinó que la solución optimiza adecuadamente los procesos y cuenta con un nivel de rendimiento significativamente alto. Así, se muestra el potencial de la aplicación como un ambiente integrado completo para el desarrollo de shaders en la Web.

Palabras Clave: Shader; Programación en la GPU; GLSL; MVC; WebGL; GPU.

Abstract: A shader is defined as a unit of program written in a shading language to the GPU to produce images (*render*). Nowadays, there are tools and Web technologies which used the display of this content in HTML5-based browsers. Thus, it has guided the development of solution to support the implementation of shader programs to the GLSL language, as part of OpenGL standard. However, being our research, there are not available Web applications with unified standard and which contains all involved elements in the development process. In this paper, a solution called Shader Tool is presented, it contains a set of tools which incorporated and linked all features to the shader development. This is a tool which uses templates, 3D graphics, interactivity and synchronization technologies to the implementation of GLSL code. According to the results of the tests, it was determined that the solution adequately optimizes processes and has a significantly higher level of performance. In this way, we shown the potential of the application as a complete integrated environment to the development of shaders in the Web.

Keywords: Shader; GPU Programming; GLSL; MVC; WebGL; GPU.

I. INTRODUCCIÓN

Un software de procesamiento gráfico 3D permite la creación y manipulación de modelos 3D por computadora. Estas aplicaciones son empleadas tanto para la construcción de imágenes como para la animación por computadora. En la actualidad existe un número considerable de aplicaciones con propósitos muy diversos y de ámbito *standalone* y *online*.

Estas aplicaciones han permitido el crecimiento en la manera de percibir y crear contenido gráfico, generando cambios sustanciales en diversas áreas. Desde el año 2013, los navegadores han aumentado su rendimiento y son capaces de ofrecer una plataforma para albergar aplicaciones de alto procesamiento gráficos. La mayoría de los navegadores han adoptado la tecnología WebGL [1], que permite crear aplicaciones con un

rendimiento óptimo de gráficos 2D y 3D en el navegador, empleando la unidad de procesamiento gráfico (GPU).

Actualmente, existen herramientas en la Web basadas en WebGL y bibliotecas especializadas en el área de los gráficos 2D/3D, que recrean un ambiente de desarrollo de aplicaciones gráficas. Del mismo modo, muchas de éstas incluyen programas especiales dirigidos a ser ejecutados en los procesadores gráficos, llamados shaders. Dichos programas son ejecutados de manera independiente por la GPU de un computador para un cierto programa de despliegue gráfico.

La forma tradicional de construcción de shaders es empleando un computador con alguna aplicación *standalone*. Dado que dichos programas no están enfocados para la Web, es ideal contar con las mismas capacidades de desarrollo en el entorno



(a) Google Body Browser

(b) Editor 3D Castle Defender

(c) The Pulse of the Planet - Twitter

Figura 1: Ejemplos del Uso de WebGL en Diversas Aplicaciones en la Web

Web. Sin embargo, en soluciones actuales disponibles no existe un criterio unificado de implementación. De hecho, en algunas aplicaciones no se percibe la incorporación de todos los elementos requeridos para realizar la programación y ejecución de shaders en GLSL.

En este trabajo se presenta el desarrollo de un sistema integrado para el desarrollo de shaders para el lenguaje de OpenGL, haciendo uso de tecnología Web de vanguardia, y de arquitecturas pensadas para ofrecer un óptimo rendimiento y opciones de escalabilidad. De esta manera se busca proveer un conjunto de herramientas Web que incorporen y enlacen todos los elementos involucrados en el desarrollo de programas GLSL, además de proporcionar accesos a la información relacionada con éstos. Shader Tool, nombre de la solución propuesta, permite un libre acceso a una biblioteca de recursos dentro de una plataforma de usuarios que está disponible desde cualquier dispositivo con soporte a un browser con HTML5. De forma puntual, nuestras principales contribuciones son:

- Una solución computacional para la creación de shaders en GLSL bajo un entorno de desarrollo Web.
- Un conjunto de librerías de recursos para la programación de shaders.
- Un sistema Web que almacena los trabajos realizados por cada usuario dentro de Shader Tool.

Este trabajo se organiza como sigue: en la Sección II es presentada una breve introducción a las tecnologías para el despliegue en la Web. La Sección III muestra una recopilación de los trabajos previos relevantes para nuestra investigación, así como un cuadro comparativo de las necesidades en un sistema integrado de desarrollo para la construcción de shaders. Luego, la Sección IV explica en detalle todos los aspectos asociados a Shader Tool como parte de nuestra investigación. La Sección V presenta los detalles propios de la interfaz de la aplicación Web. Una serie de experimentos para corroborar nuestro trabajo se muestra en la Sección VI. Por último, la Sección VII muestra una serie de conclusiones y recomendaciones a ser aplicadas.

II. TECNOLOGÍAS PARA EL DESPLIEGUE EN LA WEB

La Web siempre ha sido un medio visual, sin embargo hasta hace poco, fue de uso exclusivo de un grupo de aplicaciones

específicas. La mayoría portales Web basados en CSS y JavaScript, gracias a la incorporación de tecnologías como el elemento canvas de HTML5 [2], la especificación WebGL [1] y las imágenes SVG [3], el desarrollo Web ahora es más dinámico y adaptable. El uso combinado de toda esta tecnología puede dar como resultados aplicaciones, juegos o herramientas de alto impacto. La Web ofrece posibilidades de uso diversos y los editores de shaders es uno de ellos.

El despliegue de gráficos 3D ha tomado un gran impulso en los últimos años, especialmente en el ámbito de los videojuegos. Por ello se han desarrollado APIs especializadas para facilitar los procesos en todas las etapas de generación de gráficos por computador. Éstas han demostrado ser primordiales para los fabricantes de componentes, proporcionando un medio de acceso al hardware de una manera abstracta.

Desde hace dos décadas esta tendencia ha ido aumentando y se unió a la tecnología Web existente. De esta forma, surgen nuevas formas de producir gráficos 2D/3D y demandas en el ámbito de la Informática Gráfica. Una gran tendencia de desarrollo para su incorporación fue el uso de lenguajes de script para la Web como JavaScript (JS). Basados en este lenguaje, varios grupos de desarrollos coinciden que dicho lenguaje cuenta con la plataforma más sólida para ser considerado al momento de crear estándares y APIs que permitan el desarrollo de aplicaciones gráficas en los navegadores modernos [4].

WebGL es una API escrita totalmente en el lenguaje JavaScript para el despliegue interactivo de gráficos 2D/3D sobre cualquier navegador compatible, y sin el uso de plugins. WebGL permite el uso de aceleración a través de la GPU para cálculos físicos, procesamiento de imagen y efectos gráficos.

Entonces, WebGL es la solución para el despliegue de gráficos por computadora más aceptada en la actualidad por estar fundamentada en OpenGL, y por contar con la capacidad de estar presente en los navegadores de forma nativa [5]. Basados en WebGL, han surgido diversas bibliotecas de desarrolladores independientes que ofrecen una capa de abstracción para el despliegue de gráficos 2D/3D. La Figura 1 muestra ejemplos de despliegues: (a) una enciclopedia visual del cuerpo humano, (b) un editor de bloques 3D para personajes, y (c) un indicador visual de tendencias en Twitter para un hashtag.

En WebGL se emplean programas de shaders, los cuales

son pequeñas unidades de programas que son ejecutadas por el procesador gráfico para lograr acelerar el despliegue de gráficos 2D/3D. En WebGL, existen dos tipos de shaders: vertex shader y fragment shader, los cuales invocan a partes programables del pipeline de OpenGL. El vertex shader proporciona las posiciones para cada vértice que se dibujará, y el fragment shader proporciona el color de cada píxel a ser dibujado.

Estos shaders están escritos en GLSL (OpenGL Shading Language) [6]. Un shader puede ser incluido en una página Web con WebGL como texto codificado en un archivo fuente de JavaScript, como archivos independientes incluidos con el uso de la etiqueta `<script>`, ó se recupera desde el servidor como texto sin formato. El código JavaScript se ejecuta en la página y luego se envía para su compilación utilizando la API de WebGL y ejecutados en la GPU del dispositivo.

De esta forma básica, se ejecuta WebGL en los browser modernos, a continuación se presenta una serie de trabajos previos relacionados a nuestra investigación.

III. TRABAJOS PREVIOS

Hasta hace poco un editor de texto era un programa que permitía exclusivamente crear y modificar archivos digitales compuestos por texto sin formato, conocidos comúnmente como archivos de texto plano. La forma de evaluar los editores de texto ha cambiado, y actualmente es posible, en pocos segundos, contar con un editor en la Web para su uso inmediato sin instalaciones o configuraciones. Según Parisi [7], partiendo de esta base, existen diversos editores que junto con WebGL permiten diseñar páginas Web con alto nivel de interactividad, construir modelos 3D, escribir código fuente en algún lenguaje de programación, e inclusive desarrollar shaders propios para WebGL. Siendo este último aspecto el aspecto principal de nuestra investigación.

En la actualidad, los editores de shaders existentes se esfuerzan por ofrecer la combinación más adecuada entre una aplicación Web y un procesador de gráficos 3D. El nivel de innovación en cada implementación varía, sin embargo, características sociales y la creación de comunidades son las más usuales.

Un ejemplo de estos editores es Shdr [8], el cual es un visor/editor y validador de shaders en línea desarrollado por Despoulain. Está soportado por WebGL, Three.js [9], Ace.js [10], RawDeflate.js [11] y jQuery [12]. Shdr combina parte de la tecnología en línea comúnmente aplicada solo a páginas o sistemas Web y la vertiente actual en herramientas gráficas en la nube. La Figura 2 muestra un ejemplo de la interfaz presentada en Shdr para el modelo de Dragón de la Universidad de Stanford [13].

Es interesante destacar que Shdr cuenta con indicadores visuales (*code highlight*), sombreado de líneas seleccionadas, *snippets* (parte reusable de código fuente), cambio de contexto entre shaders y opciones comunes de manejo de archivo como descargar, abrir y guardar el trabajo actual. Por otra parte, el despliegue es aplicado en modelos 3D descargados al equipo local desde un servidor.

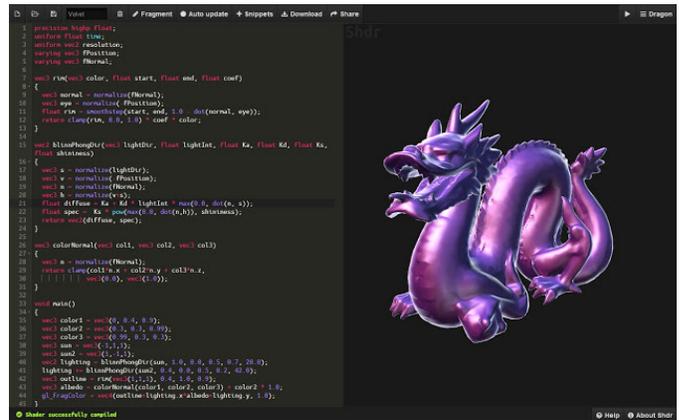


Figura 2: Interfaz de Shdr Mostrando un Modelo de Dragón [13]

Por su parte, GLSL Sandbox [14] es una aplicación creada por una comunidad de programadores para compartir y distribuir shaders. Cuenta con herramientas de edición y despliegue de efectos que lo categorizan como un editor, y posee una amplia galería de efectos y un visor a pantalla completa. Shader Lab (SL) [15] es otro editor que nace de las limitaciones propias del desarrollo nativo y la característica multiplataforma deseable entre equipos Unix y Windows. SL está significativamente limitado en cuanto a características de usabilidad. Las opciones de edición de código son prácticamente ninguna, ofreciendo 3 áreas de texto simple, sin indicadores visuales o alguna característica que potencie la experiencia de los usuarios.

Otro ejemplo notable es Kick.js [16], la cual es una biblioteca para navegadores Web modernos. Kick.js provee una API que permite crear una abstracción de WebGL y construir aplicaciones o juegos en los navegadores de manera sencilla. Basados en esta biblioteca, se desarrolló un editor de efectos 3D muy completo basado en las características fundamentales que puedan ser necesarias. Entre estas destaca una interfaz de usuarios simples sin mayor innovación en cuanto a su aspecto visual. No obstante, en cuanto a funcionalidades es el editor más completo que en la actualidad se encuentra disponible en la Web de acuerdo a nuestra investigación. Sus módulos novedosos son el uso de texturas múltiples, ajuste de variables (tipo uniform), panel de reporte de estado de efectos y diversas configuraciones del contexto 3D. Adicionalmente permite cargar y almacenar efectos, crearlos desde su inicio o usar alguno de los ejemplos incluidos en la aplicación.

Shader Toy (ST) [17] en su plataforma incluye compilación en tiempo real del código del shader, una amplia comunidad de desarrolladores, discusiones abiertas por medio de comentarios, insumos para los shader como videos, webcam, audio, texturas, hora del día y la posición del cursor, visualización en pantalla completa, editor con resaltado de sintaxis, guardar, compartir y publicar los shaders. El alcance de la herramienta es amplio, permite desarrollar efectos simples como animaciones de *sprites*, o composiciones 3D complejas tales como terrenos procedurales. El conocimiento técnico para utilizar ST es alto.

Son varias las etapas y requerimientos de software y hardware

que se necesitan para desarrollar un programa GLSL. Aunado a ello, para este proceso con tecnología Web se requiere considerar aspectos como seguridad en la red, velocidad de transferencia, estilo gráfico, manejo de sesión, redes sociales, etc. Los trabajos estudiados cuentan con un sub grupo de estas características.

Por ello, en la Tabla I se presenta una breve comparativa tomando en cuenta los aspectos más relevantes al tratarse de un editor de shader en línea. El objetivo es evaluar de manera más explícita sus características y las posibles fallas o carencias que presentan.

Tabla I: Comparación de Funcionalidades entre los Editores Estudiados. La letra S Representa "Sí", la letra N Representa "No"

Característica	SHDR	SandBox	SL	Kick	ST
Ayuda	S	N	N	S	S
Auto completado	N	N	N	N	N
Nro. de línea	S	S	N	S	S
Descargar código	S	N	N	N	N
Indicador gráfico	S	S	N	S	S
Colapso de secciones	S	N	N	S	S
Comodidad edición	5/6	3/6	1/6	5/6	3/6
Editor parámetros	N	N	N	S	N
Uso de texturas	N	N	N	S	S
Shaders (V. ó F.)	V-F	F	V-F	V-F	F
Biblioteca recursos	N	N	N	S	S
Galería de efectos	N	S	N	N	S
Manejo de sesiones	N	N	N	S	S
Compartir	N	N	N	S	S
Layout flexible	S	S	N	N	S

En la Tabla I se observan un conjunto de características comunes presentes y ausentes en los editores Shdr, GLSL Sandbox, Shader Lab, Kick.js y Shader Toy. De esta forma, en este trabajo se propone un editor, llamado Shader Tool, que contenga la mayor cantidad de características deseables, basados en el estudio comparativo mostrado.

IV. SHADER TOOL

Shader Tool es una solución que contempla el desarrollo de un ambiente integrado, con una serie de características deseables en un ambiente Web para la construcción de shaders GLSL. Desde el punto de vista técnico, la solución emplea el uso de la arquitectura de dos niveles cliente-servidor, REST y un sistema centralizado de datos.

Cada nivel de la solución se encuentra dividido en módulos: el lado del servidor atiende aspectos como el acceso a los datos, la representación de las entidades y la presentación y respuesta de los recursos solicitados; el lado del cliente incluye funciones para la optimización de la transferencia de datos, diseño de componentes Web, enrutador, entre otros.

La descripción de las arquitecturas de desarrollo de software, sus módulos funcionales y el modelo de comunicación implementado para el intercambio de información entre ambos niveles (cliente-servidor) se muestran a continuación.

A. Arquitectura Cliente-Servidor

Como lo describe Purewal [18], las aplicaciones Web usan la infraestructura de la Web (protocolos, lenguajes, entre

otras) para su funcionamiento. Sus componentes principales son los navegadores y servidores Web, permitiéndole tener acceso a la interfaz de usuario de manera global. Entre las ventajas que ofrece esta plataforma se encuentran su alto nivel de accesibilidad, soporte multiplataforma y facilidad en el mantenimiento.

Debido a la naturaleza de la solución planteada y el interés de aplicar métodos y herramientas actuales, la taxonomía Web utilizada en el desarrollo fue la orientada a servicio en un modelo cliente-servidor. En este tipo aplicación Web ofrece servicios especializados, requiriendo una lógica de negocio acorde a cada uno de ellos.

Desde el punto de vista funcional, se puede definir el modelo cliente-servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma. En este modelo, el cliente envía un mensaje solicitando un determinado servicio a un servidor (realiza una petición), y este envía uno o varios mensajes con la respuesta (ver Figura 3).

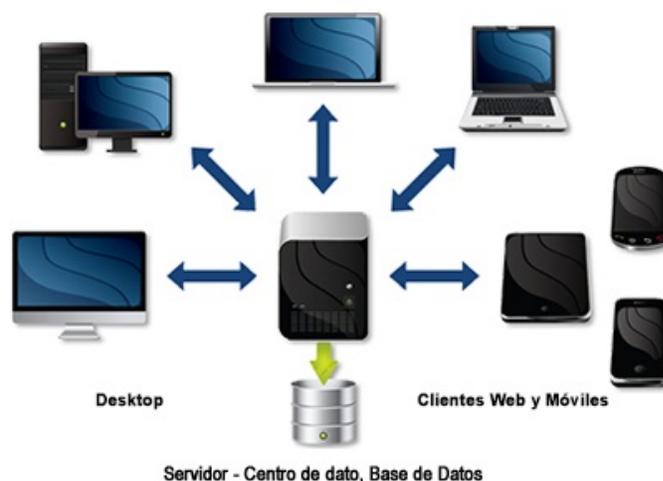


Figura 3: Modelo Básico Cliente-Servidor Empleada en Shader Tool

La lógica de negocio se realiza en el servidor y el cliente se ocupa de la interacción con el usuario. Esta arquitectura permite distribuir físicamente los procesos y los datos en forma más eficiente, lo que en computación distribuida afecta directamente el tráfico de la red, reduciéndolo significativamente.

B. Lado Cliente

El cliente es el proceso que permite al usuario formular los requerimientos. Este maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario (GUI), además de acceder a los servicios distribuidos en cualquier parte de una red. Las funciones que lleva a cabo el cliente se resumen en los siguientes puntos:

- Administrar la interfaz de usuario.
- Interactuar con el usuario.
- Procesar la lógica de la solución y hacer validaciones locales.
- Generar requerimientos de bases de datos.

- Recibir resultados del servidor.
- Dar formato a resultados.

El proceso cliente fue implementado siguiendo estándares de desarrollo, y patrones de arquitecturas adaptados a una aplicación Web. El patrón de diseño MVC (Modelo-Vista-Controlador) es la base de la arquitectura en este nivel de la solución, el cual contribuyó a la separación de los componentes relacionados con los datos y componentes de la interfaz de usuario. La separación de las capas permitió tener a nivel de desarrollo un código legible, flexible y reusable.

La estructura modelada establece una comunicación intrínseca entre las tres capas dentro del patrón MVC. Así, los archivos HTML de la solución representan la vista y debe ser separada del controlador y el modelo. Igualmente, el controlador prepara los datos a utilizar, captura los eventos de la vista y realiza las acciones pertinentes sobre el modelo para modificar los datos. El controlador es importado en el archivo de vista correspondiente para que este tenga efecto. En la Figura 4 se muestra el patrón MVC empleado en Shader Tool, así como los diferentes componentes que interactúan y componen cada capa.

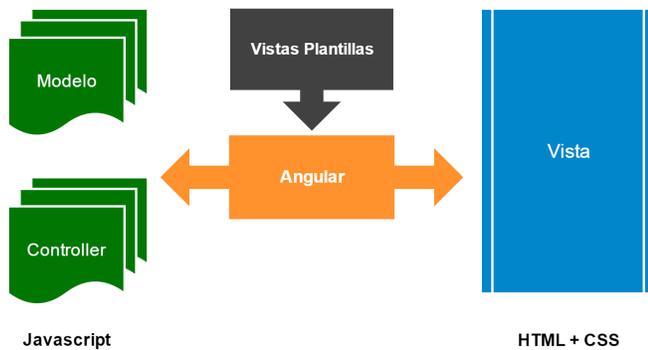


Figura 4: Patrón MVC Planteado en Shader Tool Empleando AngularJS

En nuestra propuesta, se empleó el framework AngularJS [19] para controlar las funcionalidades aplicadas en el lado cliente. Cada capa del diseño constituye una serie de componentes que proveen ciertos servicios o funcionalidades que permiten controlar aspectos como la solicitud de recursos al servidor, estado de la solución, componentes Web, entre otros.

En términos más precisos, Shader Tool fue diseñada con el esquema *single-page* en AngularJS. Este define una arquitectura formada por 8 módulos principales: application root, module, config, route, view, controller, directive y factories/services. La interacción entre ellos se muestra en la Figura 5.

El módulo Application representa el elemento raíz de la solución del lado cliente. Angular dispone del atributo ngApp para indicar dentro del documento HTML cual elemento se utiliza para iniciar la aplicación. El módulo Module permite incluir en el sistema componentes externos; el módulo Config establece la configuración de diversos parámetros de inicio y ejecución de la Shader Tool; y el módulo Route enlaza las URLs a los controladores y vistas.

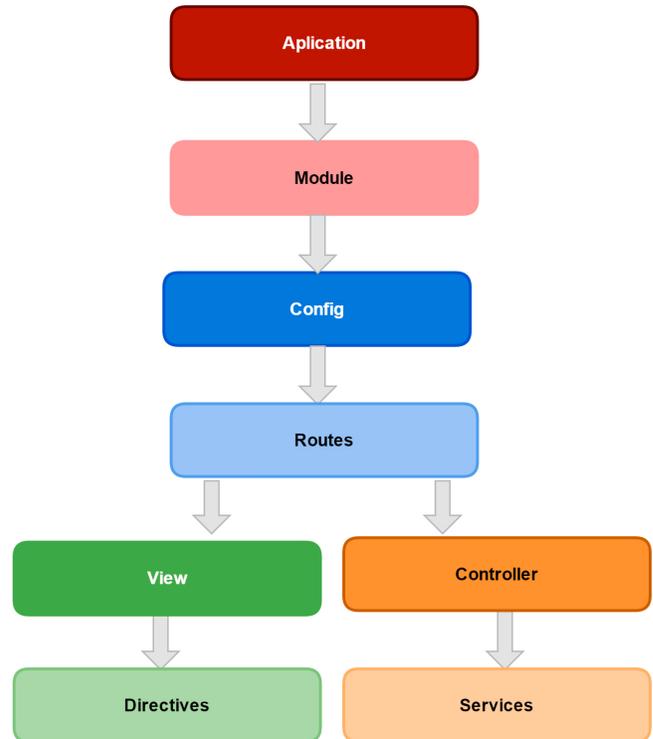


Figura 5: Módulos de la Arquitectura Provista por AngularJS

El módulo Controller emplea funciones de Javascript para aumentar el alcance del framework. El módulo View está formado por todos los archivos HTML que componen la solución. Por último, el módulo Directives permite declarar componentes Web, y el módulo Service describe una representación de un proceso u objeto.

Nuestra propuesta está compuesta por 10 vistas, y durante el desarrollo se implementaron varios componentes Web, factorías y servicios relevantes para el funcionamiento de la solución. Diversas bibliotecas y herramientas fueron empleadas con el fin de complementar nuestra solución, tales como jQuery [12], Bootstrap [20], UI-Layout [21], ngCropper [22] y uiAce [10].

Por último, la solución cliente está formada por 3 paquetes principales que hacen uso de los diversos módulos mencionados para ordenar y estructurar las funcionalidades. El paquete *Graphics* ordena los diferentes objetos involucrados en el proceso de carga, procesamiento y despliegue de un programa de shader. El paquete *Directives* contiene todos los componentes Web desarrollados. Y el paquete *Filters* define pequeños procesos que permiten transformar objetos o cadenas de caracteres para ser mostradas en la vista con un patrón en particular.

C. Lado Servidor

El lado servidor se encarga de atender a múltiples clientes que hacen peticiones de despliegue y manejo de los shaders y usuarios. Por una parte, al proceso cliente se le denomina como front-end; y al proceso servidor se le conoce con el término

back-end. El servidor maneja todas las reglas del negocio y los recursos de datos.

Adicionalmente, en la estructura del servidor, se complementa la arquitectura con la inclusión de un servidor de aplicación y base de datos. En Shader Tool se incorporó un servidor de aplicación que permite generar contenido de manera dinámica, y permite la toma de decisiones de acuerdo a parámetros de entrada incluidos en las peticiones por parte de cada cliente.

El servidor de base de datos está presente para apoyar de forma directa la persistencia de datos. Esta es requerida para almacenar y recuperar el estado de las entidades e información que pueda generarse a través de la solución. Todos los componentes descritos se resumen en la Figura 6, donde en la parte superior (color azul) se observan los componentes del servidor, y en la parte inferior un cliente conectado desde un portátil (lado izquierdo - color verde) y desde un dispositivo móvil (lado derecho - color rojo).

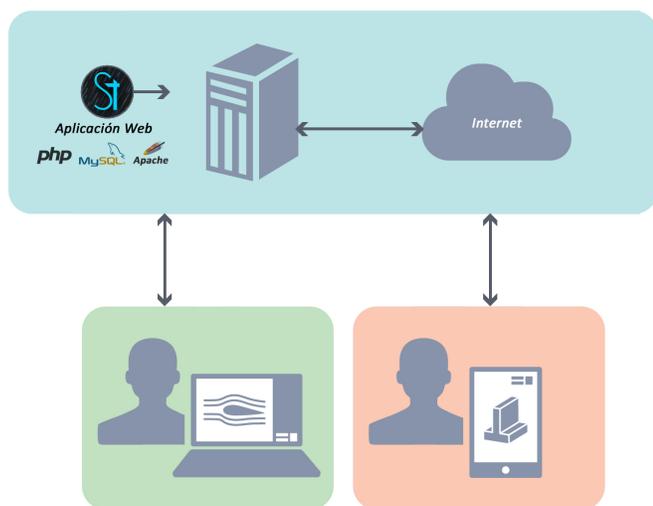


Figura 6: Arquitectura de Comunicación y Software del Lado Servidor

Para concebir la solución de forma abstracta y adaptable, el servidor de aplicación es un sistema distribuido dividido en capas (i.e. patrón de arquitectura multicapa) para separar los componentes de acuerdo a su función. Luego de un proceso de análisis se establecieron las siguientes capas funcionales: persistencia de datos, acceso a datos, entidades, y lógica de negocio.

A nivel técnico el trabajo presenta otros aspectos relevantes que forman parte del núcleo principal de la aplicación, sin embargo, estos son detalles a nivel de implementación del lado del cliente y servidor que son de mayor interés en el área de la programación. Aun así, es importante destacar las características asociadas al resultado global obtenido enmarcado en un ambiente Web de desarrollo integrado para GLSL.

V. INTERFAZ DE SHADER TOOL

La solución interactiva creada, permite al usuario manipular el sistema de forma gráfica haciendo uso de la entrada estándar de teclado y ratón. Esta puede ser definida como un sistema

integrado de desarrollo que permite implementar programas de shaders para el lenguaje GLSL, organizada en secciones y componentes.

La interfaz gráfica está dividida en cinco grandes secciones. Cada una de estas secciones está agrupada en sub-elementos según sus características, funcionalidades y similitudes. Por ejemplo, todos los comandos para la programación de los programas de shaders se encuentran agrupados en la parte izquierda de la barra de menú. Como puede verse en la Figura 7, las cinco secciones principales son las siguientes:

- 1) Barra de menús: Corresponde a la fuente de comandos más común. En la barra de menús se encuentran todos los métodos y funcionalidades de la solución. Está dividida en 3 partes bien marcadas: opciones de editor, opciones de despliegue y perfil de usuario.
- 2) Panel de Editor: Define el área en la cual se escribe el código de los programas de shaders. Cada programa (de vértice y fragmento) disponen de este espacio, sin embargo, solo uno podrá estar activo en un momento dado.
- 3) Panel de vista: Permite percibir el contenido de la escena, en la cual se aplica el resultado tras la compilación de ambos programas de shaders. Junto al panel de editor, puede ser ajustado su tamaño a gusto del usuario.
- 4) Área de Resultado: Muestra el resultado obtenido tras la compilación del shader. Está ubicada en la parte inferior del panel de vista, y agrupa en dos pestañas los mensajes para cada programa.
- 5) Área de Notificación: Despliega las notificaciones generadas por procesos iniciados por el usuario. Está ubicada en la parte superior derecha del panel de vista.

A. Panel de Editor

El panel del editor permite a los usuarios escribir el código que componen su shader. El editor dispone de tres elementos de interfaz mostrados en la Figura 8, que ayudan a la interacción: Número de línea, colapso de renglones de código y menú contextual de edición.

Es posible colapsar/abrir una región de código presionando en un indicador en forma de flecha (ver Figura 8). Posee un menú contextual que permite opciones como deshacer y rehacer el estado del editor, seleccionar todo el código y los comandos estándar copiar, pegar y cortar.

B. Panel de Vista

Es la ventana donde se despliega la escena vista desde cierta perspectiva. Esta ventana posee numerosas opciones que le permiten al usuario diferentes configuraciones al momento de visualizar la escena. La primera opción de configuración es proyectar la escena con una vista perspectiva u ortogonal.

La Figura 9 muestra un ejemplo de proyección perspectiva dentro del panel vista de un modelo de auto, junto con el menú contextual. Otro elemento de configuración del panel son los atributos de despliegue del objeto en la escena. Entre los parámetros que pueden ser adaptados están: despliegue

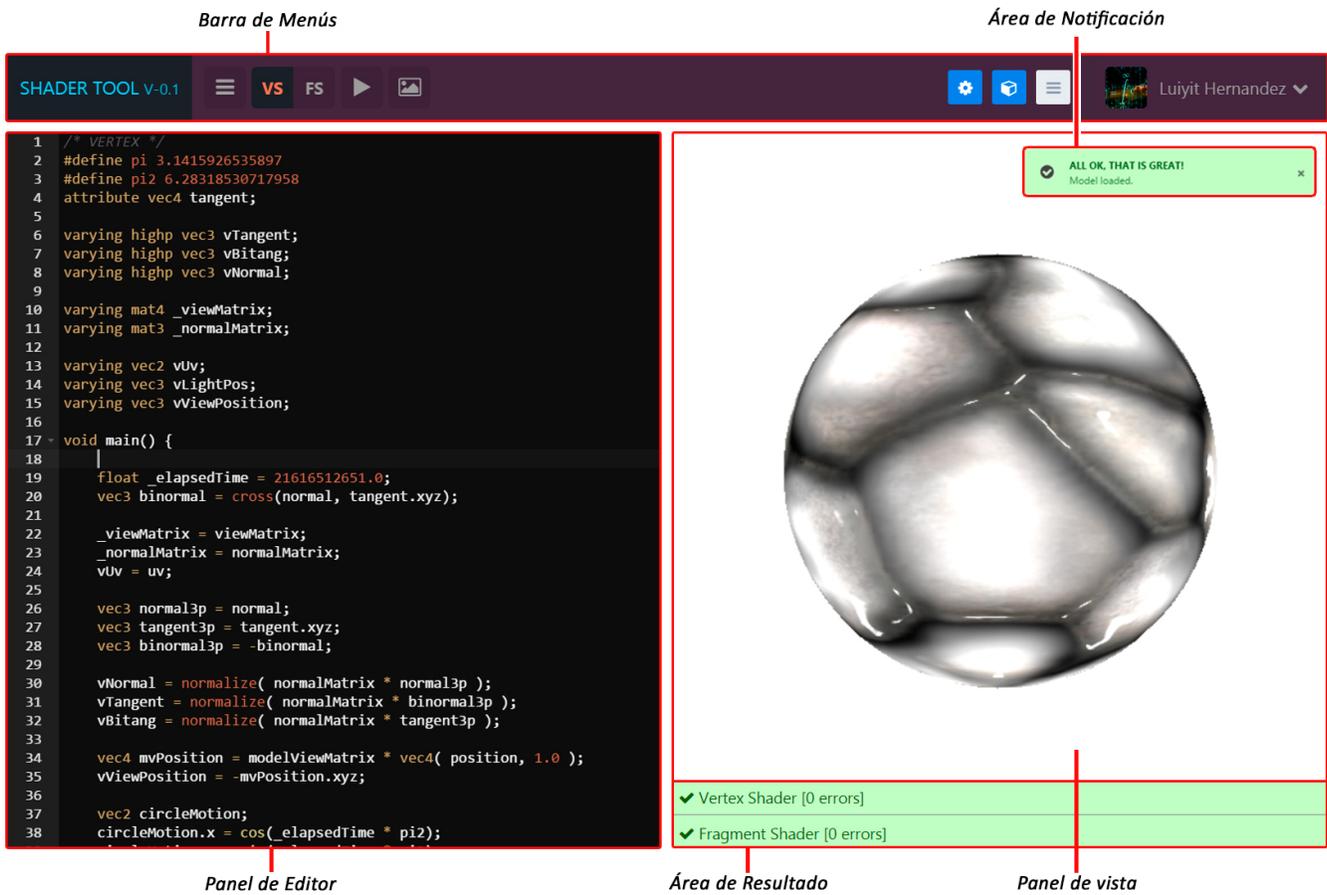


Figura 7: Diferentes Secciones de la Página Editor de Shader Tool (Lado Cliente)

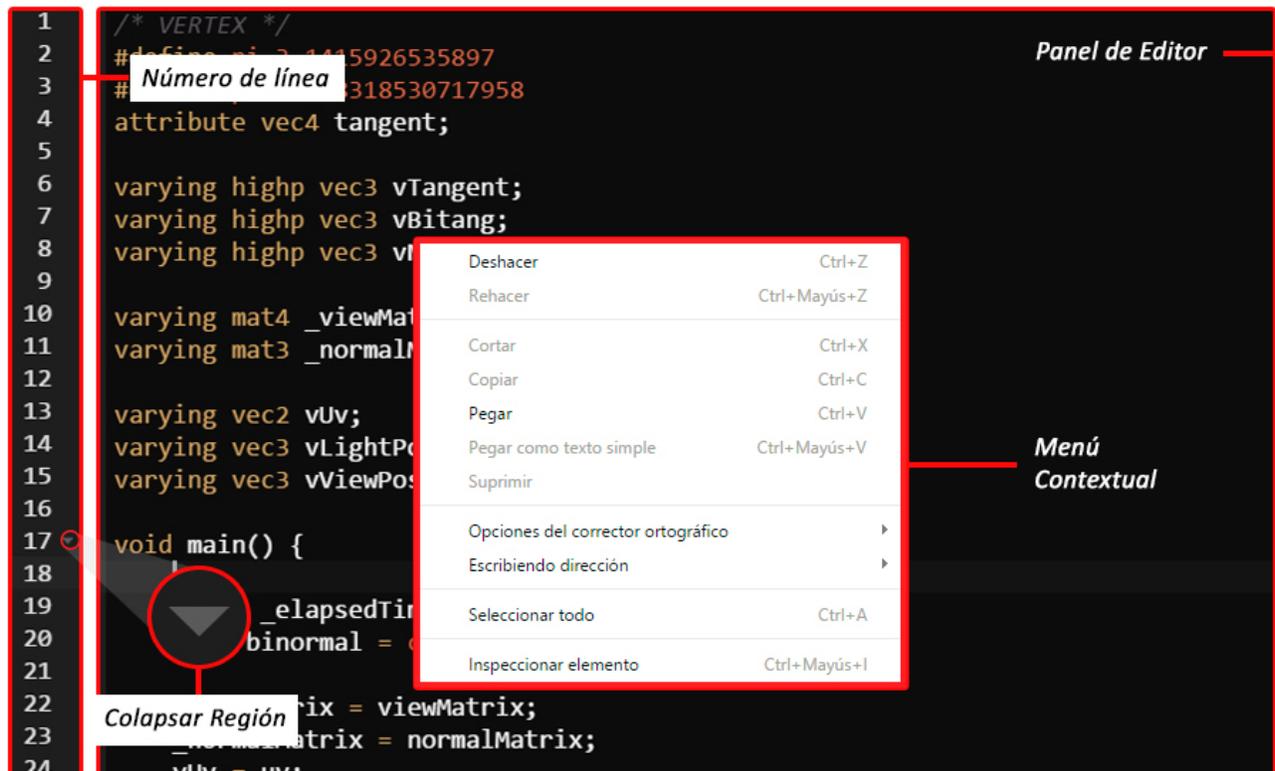


Figura 8: Elementos que Conforman el Panel de Editor

de las caras delanteras/traseras, activar/desactivar el zBuffer, desplegar el objeto como malla/puntos/relleno.

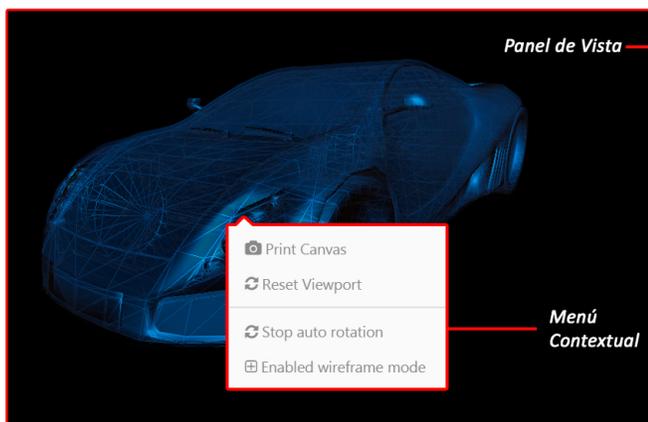


Figura 9: Elementos que Conforman el Panel de Vista

Igualmente, la cámara también puede ser adaptada y ajustar valores del frustum. Es importante destacar que estas opciones de configuración son accesibles desde la barra de menú principal, y a través del menú contextual que se muestra con el click derecho en cualquier área del panel.

C. Modales Funcionales

La solución Shader Tool cuenta con un gran número de opciones y herramientas de configuración asociados al proceso de programación de los shaders. Teniendo en cuenta que el espacio más relevante de la ventana del browser fue apropiadamente dispuesto para los dos paneles principales, se empleó un sistema de modales para agrupar funcionalidades particulares. De esta manera las diferentes funciones de la solución son accedidas a demanda.

Cada modal se superpone a los paneles principales. Esto permite ordenar la información y controles en un espacio independiente sin alterar la disposición de la interfaz del usuario. Las funciones más relevantes que usan el sistema de modal son: crear Shader, cargar Shader, biblioteca de texturas, biblioteca de modelos y perfil de usuario.

Por ejemplo para cargar un shader a Shader Tool, se dispone de un modal que permite seleccionar cuál shader se cargará. Para ello, se presenta una lista de shaders la cual es dividida en páginas formando grupos de cinco, y el shader actual es mostrado con un icono en forma de v en la columna de opciones de la tabla. Para seleccionar un shader, se debe hacer clic sobre el botón de acción en color azul al final de cada fila. En la Figura 10 se muestran todos los elementos involucrados en dicho modal.

Dada la dinámica de la solución, existen diversos modales que resultan apropiados en sistemas con resoluciones de menor tamaño o donde la selección de opciones requiera una mejor precisión (i.e. tablets y teléfonos inteligentes).

Un ejemplo de ellos es el modal para añadir un nuevo modelo a la biblioteca de modelos de un usuario. Para ello, se cuenta con

un área donde se puede seleccionar el modelo arrastrándolo y soltándolo en una región dispuesta para ello, o seleccionándolo desde el selector nativo del browser. En ambos casos, el archivo se validará y procesará.

Una vez seleccionado el modelo se creará de forma automática una escena que desplegará el modelo. Para garantizar el correcto despliegue, el modelo es normalizado y centrado antes de ser mostrado. Mientras el área de pre-visualización está activa, se activan dos opciones adicionales: descartar modelo, y detalles del modelo. La escena y tabla de detalles del modelo es mostrado en la Figura 11 donde se observa los diálogos modales para la carga de un modelo, y su descripción.

VI. PRUEBAS Y RESULTADOS

En esta sección se presentan una serie de pruebas para validar nuestra propuesta. Así, se muestran pruebas cuantitativas y cualitativas obtenidas por el uso de la solución. Dado que es una solución Web, resulta interesante medir la velocidad de descarga de los diversos recursos, la eficiencia del proceso de compilación y ejecución de los shaders y la experiencia de los usuarios en términos de utilidad, facilidad de uso y cumplimiento de los objetivos de la investigación.

A. Detalles de Implementación

El ambiente Web desarrollado requirió de numerosas de bibliotecas y clases (RESTful, Scene, Chameleon, entre otras) en los lenguajes PHP v 5.4 y Javascript, las cuales fueron divididas en dos niveles (cliente-servidor); los cuales contienen los métodos y propiedades necesarios para el funcionamiento de Shader Tool.

La biblioteca creada del lado servidor se encarga de procesar las peticiones recibidas, y prepara el entorno para atenderla y dar respuesta, entre otras funciones. Del lado del cliente las clases desarrolladas se dividen en 3 grupos principales:

- 1) Gestión de las peticiones y respuestas obtenidas.
- 2) Gestión de las tareas involucradas en el proceso de despliegue de la escena 3D.
- 3) Gestión de componentes Web para encapsular funcionalidades y gestión de manejo de eventos en la interfaz de usuario.

B. Experimentación Cuantitativa

Estos experimentos tienen como objetivo medir el tiempo y recursos necesarios para la carga de los diversos recursos

Number	Name	Last Update	Options
1	Light	Apr 15, 2015 5:18:04 AM	<input checked="" type="checkbox"/>
2	Example	Apr 15, 2015 5:17:12 AM	<input type="checkbox"/>
3	Textures	Apr 15, 2015 5:17:24 AM	<input type="checkbox"/>
4	Testing Editor	Apr 15, 2015 5:17:30 AM	<input type="checkbox"/>
5	Basic	Apr 15, 2015 5:17:44 AM	<input type="checkbox"/>

Lista de modelos

Modelo actual

Seleccionar Shader

Paginación

Figura 10: Lista de Shaders Asociados a un Usuario

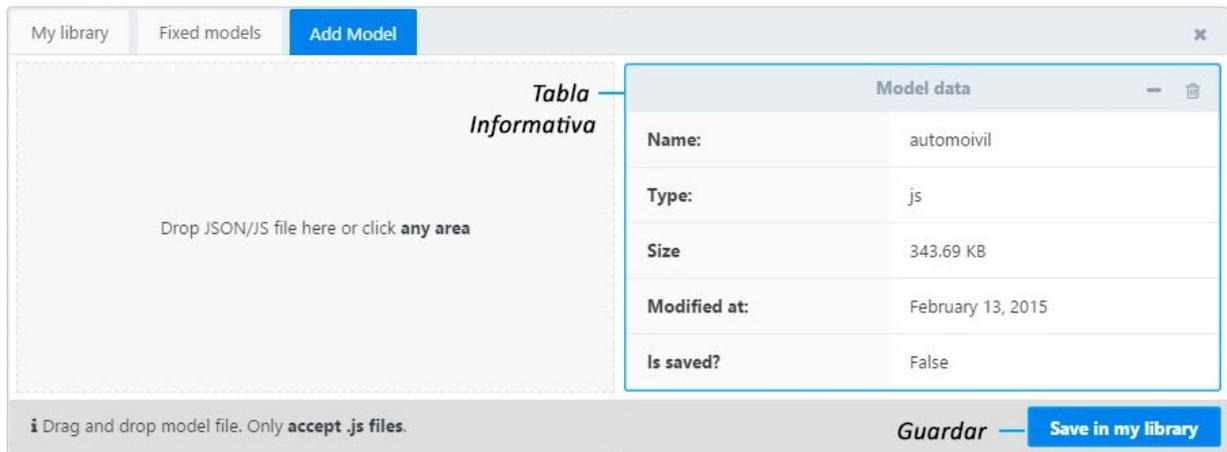


Figura 11: Diálogo Modal para la Carga y Descripción de los Modelos

gestionados en la solución desarrollada en este trabajo. Entre las pruebas se incluyen: velocidad de carga de los modelos en varias configuraciones de red, velocidad de procesamiento de los modelos y tiempo de carga y compilación de los shaders usando 4 técnicas de sombreado.

1) *Procesamiento de Modelos:* Para realizar las pruebas de velocidad de carga y procesamiento se desarrolló un pequeño paquete *benchmark* que se integra a la solución y reproduce los procesos involucrados en las pruebas. Las clases implementadas registran el tiempo de respuesta del servidor y el tiempo requerido para procesar dicho recurso luego de tenerlo disponible para su uso. El proceso completo involucrado en las pruebas incluye las siguientes etapas:

- Solicitar lista de recursos disponibles (identificadores) del lado servidor para realizar las pruebas. Esta etapa es previa al registro del tiempo.
- Procesar cada identificador de manera individual. Se solicita el recurso completo al servidor.
- Al recibir el modelo, se valida la respuesta y se inicia el procesamiento de los datos.
- Se transforma la cadena de caracteres que contiene información de geometría a una estructura en formato manipulable y estructurado.
- Se procesa la estructura y se crea un objeto propio de la solución manipulable por el framework de despliegue.
- Con la información de geometría se crea un objeto modelo, el cual posee propiedades adicionales necesarias para el despliegue e interacción con el resto de los elementos de la solución.

El proceso fue aplicado a una muestra de 10 modelos con distinciones claras en cuanto al número de vértices y triángulos involucrados. Las redes simuladas en las pruebas fueron 2G (450 kbps), 3G (1 Mbps) y 4D (4 Mbps). En la Tabla II se muestran los resultados obtenidos del proceso completo (*TTFB* significa tiempo de espera del primer byte).

Los resultados obtenidos indican un aumento en el tiempo requerido para descargar y procesar los modelos a medida que la velocidad de la red utilizada es menor. Es claro que la velocidad de descarga recomendada se encuentra entre 1 y 4

Tabla II: Tiempos de Carga de los Modelos y su Valor Promedio por cada Red (en Color Azul)

Modelo	Vértices	Triángulos	2G	3G	4G	TTFB
<i>Tetrahedron</i>	4	4	0.45 s	0.35 s	0.32 s	0.34 s
<i>Cube</i>	8	12	0.65 s	0.36 s	0.36 s	0.31 s
<i>Cylinder</i>	147	228	3.15 s	1.33 s	0.89 s	0.68 s
<i>Plane</i>	256	450	3.80 s	1.81 s	0.88 s	0.36 s
<i>Ring</i>	297	512	4.38 s	2.54 s	0.94 s	0.42 s
<i>Torus</i>	336	600	7.23 s	2.97 s	1.46 s	0.40 s
<i>TorusKnot</i>	768	1536	16.76 s	8.02 s	3.48 s	0.47 s
<i>Sphere</i>	1296	2380	27.31 s	10.21 s	4.85 s	0.63 s
<i>Text</i>	2292	4572	28.09 s	11.88 s	4.72 s	0.53 s
			9.18 s	3.92 s	1.79 s	0.46 s

MB. De igual forma se puede evidenciar que el tiempo de procesamiento de cada modelo no es un factor crítico en el tiempo total, teniendo en cuenta que el promedio obtenido en una red con una velocidad de transferencia alta es de apenas 0.53 segundos.

Es importante destacar que las pruebas se realizaron bajo un entorno red estándar, y se promediaron los resultados obtenidos en 20 simulaciones para cada tipo de red estudiado. Adicionalmente hay que tener en cuenta que en los tiempos indicados no se están considerando los milisegundos correspondientes a la fase búsqueda de DNS (*DNS Lookup*) y tiempo de bloqueo (*Proxy negotiation* o *Establishing Connection*).

Para complementar el análisis de los resultados, se estudió el tiempo de respuesta que ofrece la solución una vez que el recurso ha sido descargado del servidor. En la Tabla III se muestra el tiempo de procesamiento de los datos recibidos para cada modelo analizado, así como la cantidad de memoria RAM que requiere cada objeto. Las fases que conforman el procesamiento del modelo son:

- Validar la información recibida (integridad del objeto).
- Crear la instancia del objeto de geometría a partir de las cadenas de caracteres recibidas.
- Crear objeto 3D a partir de la geometría resultante.

Al crear la instancia del objeto, queda disponible para su uso en cualquier otro proceso. Teniendo en cuenta que son varias las rutas que pudiera tener el modelo (despliegue

Tabla IV: Resultados Obtenidos en 3 Configuraciones de Hardware para 5 Shaders Distintos

Shader	# líneas	¿Textura?	GeForce GTX 750	AMD Radeon HD 8550G	ARM Mali-400
<i>Basic</i>	9	N	0.0043 s	0.0055 s	0.0177 s
<i>Sample Texture</i>	15	S	0.0047 s	0.0183 s	0.0195 s
<i>Displacement</i>	35	S	0.0085 s	0.0094 s	0.0257 s
<i>Per Fragment Lighting</i>	45	S	0.0077 s	0.0122 s	0.0317 s
<i>Normal Mapping</i>	153	S	0.0153 s	0.0278 s	0.0613 s
<i>Frames por segundo mínimo / máximo</i>			60 fps / 60 fps	57 fps / 60 fps	25 fps / 60 fps

Tabla III: Cantidad de Memoria y Tiempo de Procesamiento de los Modelos

Modelo	Memoria	Tiempo de Procesamiento
<i>Tetrahedron</i>	27 kB	0.42 s
<i>Cube</i>	78 kB	0.45 s
<i>Dodecahedron</i>	231 kB	0.49 s
<i>Cylinder</i>	1.4 MB	0.45 s
<i>Plane</i>	2.8 MB	0.46 s
<i>Ring</i>	3.3 MB	0.48 s
<i>Torus</i>	3.8 MB	0.74 s
<i>TorusKnot</i>	9.8 MB	0.55 s
<i>Sphere</i>	15.3 MB	0.61 s
<i>Text</i>	29.1 MB	0.65 s

principal, pre-visualización, asociación, listar datos básicos para su selección, entre otros), la prueba de procesamiento se evalúa hasta este punto, considerando única y exclusivamente el tiempo requerido para transformar los datos recibidos desde el servidor, en un objeto Javascript válido.

Las pruebas realizadas evidencian que la velocidad de la red utilizada para acceder a la solución tendrá un alto impacto en la velocidad de respuesta. EL mayor porcentaje de tiempo en obtener un recurso corresponde al proceso de descarga, teniendo en cuenta que en promedio el valor TTFB es de 0.5 segundos. Durante este tiempo de espera por parte del cliente, se llevan a cabo entre otros procesos, la recepción de la petición, su procesamiento y respuesta. Manteniendo un tráfico bajo, la solución del lado servidor es capaz de atender las peticiones en fracciones de segundo.

Por otra parte, el procesamiento de los modelos luego de recibir el 100% de los datos oscila entre 0.42 segundos y 0.74 segundos, lo cual indica que en conjunto (cliente y servidor) la solución requiere en promedio de 1 segundo para procesar cualquier modelo que estén ajustado a las muestras utilizadas.

2) *Procesamiento de Shaders:* El proceso encargado de gestionar y procesar el código de los shaders es uno de los principales dentro de Shader Tool. Las pruebas asociadas a los programas GLSL se dividen en dos partes. La primera parte registra el tiempo de procesamiento del shader, desde el momento en que se recibe toda la información desde el servidor, hasta crear las estructuras necesarias y compilar el código. Los resultados varían de acuerdo a las capacidades de procesamiento, tanto del procesador como de la tarjeta gráfica del equipo que ejecuta la solución del lado del cliente.

Para validar el comportamiento de la solución en varios escenarios, las pruebas se llevaron a cabo en tres configuraciones de hardware distintas: GeForce GTX 750, AMD Radeon HD 8550G y ARM Mali-400. En la Tabla IV se muestran los

valores obtenidos, donde los shaders analizados son:

- *Basic:* shader con operaciones básicas. Se procesa de forma simple la posición de cada vértice y se asigna un color estático a cada fragmento.
- *Sample Texture:* incluye el manejo de un mapa de bits en el programa de fragmento. Para definir el color de cada fragmento se hace uso de las coordenadas de texturas y se extrae la muestra directamente de la textura.
- *Displacement:* modifica dinámicamente los vértices y colores. Se configuran variables uniformes y atributos a cada vértice para animar la geometría, implicando enviar en cada cuadro los datos actualizados a la tarjeta gráfica.
- *Per Fragment Lighting:* técnica de iluminación en la cual la iluminación se basa en cada fragmento de la imagen final. Se incluyen componentes de luz difuso y especular, así como también el uso de los vectores normales para cálculos de incidencia de luz sobre cada fragmento.
- *Normal Mapping:* ilumina y simula relieve en superficies planas empleando dos mapas de textura.

Los resultados obtenidos al procesar cada shader muestran que tanto en computadores personales como en dispositivos móviles el tiempo requerido por la solución está por debajo de $\frac{1}{10}$ s. Teniendo en cuenta la velocidad de carga y procesamiento de los modelos, esta cifra se encuentra en un rango aceptable, ya que en el peor de los casos el shader será procesado en el mismo tiempo que los modelos. Las pruebas realizadas muestran una relación entre el número de líneas y tiempo requerido para ser procesadas por número de operaciones aritméticas involucradas en el shader, lo cual indica que las optimizaciones realizadas al código de cada programa es un factor importante a considerar.

Otro valor importante asociado a los shaders es la velocidad con la que el despliegue se realiza. Las imágenes por segundos (*frame per seconds* - fps) es la medida de la frecuencia a la cual un reproductor de imágenes reproduce distintos fotogramas (*frames*). Estos fotogramas están constituidos por un número determinado de píxeles que se distribuyen a lo largo de una red de texturas. La frecuencia de los fotogramas es proporcional al número de píxeles que deben generarse, incidiendo en el rendimiento del ordenador que los reproduce. En la parte inferior de la Tabla IV se muestran los valores mínimos y máximos de despliegue para cada shaders analizado.

3) *Velocidad de Carga:* Inicialmente la ubicación del servidor de pruebas no fue un aspecto a considerar. Sin embargo, al momento de realizar las pruebas resultó beneficioso una distancia física considerable entre el servidor y el equipo cliente, con el fin de obtener resultados en la velocidad

Tabla V: Tiempos Obtenidos en Diversos Sistemas Operativos y Browsers para el Despliegue

Dispositivo - Sistema Operativo	Browser	1er despliegue	2do despliegue	TTFB
Desktop - Microsoft Windows 8	Chrome 40	32.451 s	2.505 s	1.079 s
Desktop - Microsoft Windows 8	Firefox	35.880 s	1.707 s	0.681 s
Desktop - Microsoft Windows 8	IE11	28.965 s	0.471 s	1.077 s
Motorola Moto G - Android 4.3	Chrome	34.385 s	4.703 s	1.452 s
Nexus 5 - Android 4.1	Chrome	33.962 s	4.197 s	1.477 s
Nexus 7 Landscape - Android 5.0	Chrome	33.865 s	3.849 s	1.201 s
iPhone 4.0 - iOS 5.1	Safari	23.392 s	4.835 s	0.001 s
	<i>Tiempo promedio</i>	27.207 s	2.823 s	0.736 s

más fiables. Así, se realizaron las conexiones de pruebas desde Caracas, Venezuela hacia un servidor en Virginia, USA. Actualmente, Shader Tool se encuentra alojado en la dirección http://ccg.ciens.ucv.ve/~esmitt/projects/shader_tool.

El ambiente integrado está formado por un conjunto de archivos en formato Javascript, CSS, HTML, entre otros. Por su naturaleza, todo el contenido y archivos necesarios para el funcionamiento de la solución deben ser descargados de forma íntegra. De igual manera, los diversos módulos deben ser iniciados y enlazados.

Para estimar el rendimiento de la solución en cuanto a tiempo de inicio, se realizaron simulaciones de descarga combinando diversas plataformas de sistemas operativos actuales y browser comerciales. En la Tabla V se muestra el tiempo requerido en el cual la aplicación se inicia y se encuentra disponible para su uso.

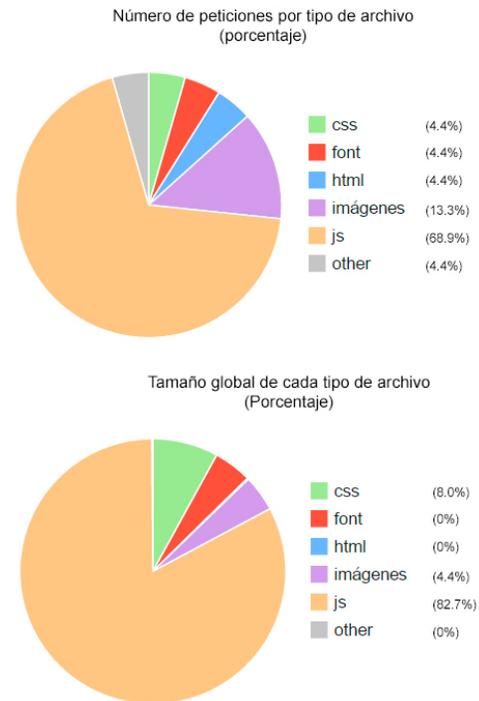
En la ejecución de las pruebas se empleó una conexión de rendimiento promedio (Móvil 3G - 780 kbps/330 kbps) y las peticiones se realizaron, como se mencionó anteriormente, desde un equipo ubicado en Dallas, VA, USA. El objetivo de realizar las pruebas desde un equipo que esté significativamente distante del servidor que aloja la solución, es generar los resultados bajo condiciones que representen una exigencia mayor en cuanto al nivel de optimización de la solución.

Por otra parte, al evaluar los resultados de inicio se observa una mejora en el despliegue de la aplicación en accesos posteriores a la carga inicial. En todos los casos de estudio se obtiene una mejora superior al 80 %. También se muestra que el tiempo necesario se mantiene estable en los diversos dispositivos y sistemas operativos.

Para ilustrar la composición final de la solución y la división de los recursos descargados en las pruebas se muestra en la Figura 12, donde se muestra el resumen porcentual de cada tipo de archivo descargado y su tamaño medido en kilobytes.

Se observa que la solución está conformada principalmente por archivos Javascript (js en la Figura 12). Aunque la solución contempla el uso de aproximadamente 15 documentos HTML, éstos son gestionados usando un servicio de optimización llamado *TemplateCache*. El funcionamiento del servicio se basa en almacenar en memoria una función Javascript que da acceso al contenido de cada vista. Cada archivo HTML ahora puede ser accedido mediante una palabra clave.

Basado en ese concepto, cada archivo HTML es representado

**Figura 12:** Tipos de Archivo en Número de Peticiones y Tamaño

por un archivo JS el cual es incluido como parte de la aplicación. Este procedimiento incrementa significativamente el tiempo de respuesta de la aplicación una vez que el usuario accede a los diferentes módulos, ya que todas las vistas fueron descargadas en la inicialización de la aplicación.

C. Experimentación Cualitativa

Las pruebas cualitativas se basaron en recopilar las distintas experiencias de los usuarios al utilizar la solución en términos de facilidad de uso, utilidad y cumplimiento de los objetivos de la investigación. El instrumento diagnóstico utilizado fue una encuesta en torno a 8 preguntas contenidas en 4 tópicos principales:

- 1) Utilidad de la interfaz gráfica de usuario.
- 2) Facilidad en la creación de shaders.
- 3) Evaluación cualitativa de los recursos disponibles.
- 4) Utilidad de la solución en términos de tiempo, esfuerzo y recursos utilizados.

La síntesis de los resultados obtenidos en la encuesta agrupados por tópicos se puede visualizar en la Figura 13. El rango de valoración fue establecido basado en una escala de Likert de 5 niveles; donde 5 representa un desempeño óptimo dentro de las características del tópico y 1 representa deficiencias en uno o varios aspectos asociadas.

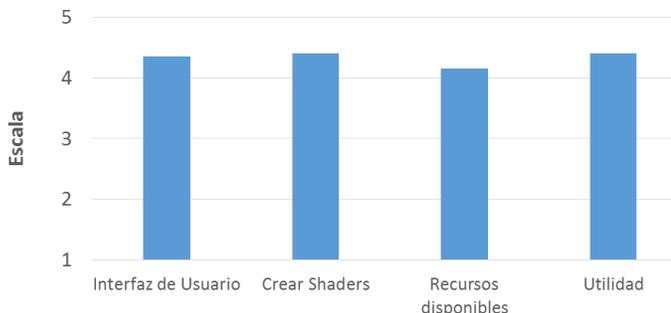


Figura 13: Valoración Promedio de cada Tópico Evaluado de la Encuesta

La encuesta fue realizada a 20 estudiantes de la Escuela de Computación de la Universidad Central de Venezuela. Los encuestados forman parte de las asignaturas impartidas en el área de Computación Gráfica debido a que requieren conocimientos técnicos involucrados en el desarrollo de shaders en el lenguaje GLSL. La mejor y peor valoración (en promedio de los 4 rubros) fue retirada de la encuesta.

Los resultados indican que todas las herramientas interactivas proporcionadas por la interfaz gráfica de usuario facilitan en gran medida el uso de la solución, ya que las mismas permiten ajustar el área del editor y viewport a voluntad, acceso a las diferentes opciones a través de múltiples interfaces y la posibilidad de ejecutar comandos directamente desde atajos de teclado. Igualmente, se pudo validar que el proceso de desarrollo de shaders es considerado práctico y rápido por los usuarios. En todo momento se conoce claramente sobre cual programa GLSL se está trabajando, la solución permite rápidamente cambiar entre los programas y los resultados del Shaders son mostrados rápidamente en el panel de despliegue.

Por otro lado, Shader Tool permite a los usuarios usar una biblioteca de modelos, texturas y shaders estándar da un valor agregado a la herramienta. De esta forma, los pueden usuarios crear shaders y concentrarse exclusivamente en la programación y no en los recursos necesarios. Por último, la velocidad de carga de la solución es el aspecto más cuestionado. Los usuarios no coinciden en una calificación uniforme en este rubro. Una parte de los encuestados indican que el desempeño es adecuado y otro grupo que es bajo o ineficiente. Sin embargo, hay que considerar que la calificación obtenida está asociada a la velocidad de conexión que posee el usuario.

VII. CONCLUSIONES Y RECOMENDACIONES

En este trabajo se presenta un entorno Web integrado que permite desarrollar programas de sombreado para OpenGL, haciendo uso de frameworks avanzados de despliegue y herramientas de marcado y estilo. Durante nuestra investigación

se analizaron todas las variables y condiciones que intervienen durante el desarrollo de programas GLSL en un ambiente de Web. Con dicha información se estudiaron las limitaciones de los editores actuales disponibles en la actualidad y se encontró que se debía desarrollar un sistema integrado que con tecnología Web de vanguardia, y con componentes adecuados de una interfaz gráfica de usuario multiplataforma.

El análisis de la interacción del usuario con la solución desarrollada y los diversos principios de diseños como la jerarquía visual, legibilidad e integridad, fueron aplicados para reducir la curva de aprendizaje de la solución y de todas sus funcionalidades. De acuerdo a los resultados obtenidos, se encontró que el conjunto de componentes e interfaces gráficas que se diseñaron e implementaron permiten al usuario interactuar con el sistema de manera sencilla, facilita el desarrollo de los shaders y su asociación con elementos como texturas, modelos y variables de control.

La solución permite cargar y procesar los recursos principales empleados en la fase de despliegue del shader. Los indicadores de carga, controles de páginas y carga fragmentada de recurso en los modales que despliegan la lista de modelos 3D y texturas, fueron introducidos como mejoras para ayudar a la velocidad de carga e interacción con los usuarios. Shader Tool ofrece a los usuarios una biblioteca pública que puede ser utilizada en sus shaders. Igualmente, cada programador puede crear y gestionar sus modelos de manera independiente.

Los resultados de la encuesta indican que existen factores que presentan la solución como una posible herramienta de entrenamiento y formación interactiva para la comunidad de estudiantes y programadores del área de computación gráfica.

A futuro, el sistema puede ser adaptado para permitir la incorporación de un mayor número de parámetros para el despliegue; como incluir la posibilidad de manipular parámetros de transparencia o blending en los materiales asociados a los modelos, uso de una fuente de luz configurable y posibilidad de adaptar las funciones aplicadas a la prueba de profundidad.

Nuestra solución permite incluir variables de tipo uniforme a sus shaders, sin embargo, no existe ninguna herramienta que permita registrar variables personalizadas de tipo atributos o uniformes. Una solución sería incorporar un nuevo modal a la interfaz de usuario que permita gestionar (crear, modificar y eliminar) las variables asociadas a un shader en particular. Es importante mencionar que los requerimientos necesarios para la implementación de esta funcionalidad del lado servidor fueron previstos, y actualmente el esquema de base de datos y la API REST están preparados para soportarla.

La investigación realizada podría ser la base para el desarrollo de un sistema más complejo que permita el implementar shaders en un contexto 2D o post procesamiento. Dicho sistema podría permitir la elección por parte del usuario del tipo de shader que se desea codificar y configurar el ambiente de desarrollo en base a la elección. Esto implicaría el desarrollo de módulos adicionales como el manejo de sprites y la manipulación de buffers de despliegues propios de OpenGL.

Teniendo en cuenta los resultados obtenidos en las pruebas de rendimiento aplicadas a los shaders, resulta viable realizar tareas de procesamiento gráfico más exigentes. Por lo cual, resultaría interesante realizar adaptaciones en la solución para despliegues de volúmenes 3D a través de técnicas como volume rendering.

REFERENCIAS

- [1] Khronos Group, *OpenGL ES 2.0 for the Web*, <https://www.khronos.org/webgl>.
- [2] World Wide Web Consortium (W3C), *HTML Canvas 2D Context*, <http://www.w3.org/TR/2dcontext> [W3C Candidate Recommendation del 2 de Julio 2015].
- [3] World Wide Web Consortium (W3C), *Scalable Vector Graphics (SVG)*, <http://www.w3.org/TR/SVG> [W3C Recommendation del 16 de Agosto 2011].
- [4] M. Myers, *A Smarter Way to Learn JavaScript: The New Approach That Uses Technology to Cut Your Effort in Half*, 1ra edición, CreateSpace Independent Publishing Platform, 2014.
- [5] K. Matsuda y R. Lea, *WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL*, 1ra edición, Addison-Wesley Professional, 2013.
- [6] R. Rost, B. Licea-Kane, D. Ginsburg, J. Kessenich, B. Lichtenbelt, H. Malan y M. and Weiblen, *OpenGL Shading Language*, 3ra edición, Addison-Wesley Professional, 2007.
- [7] T. Parisi, *Programming 3D Applications with HTML5 and WebGL: 3D Animation and Visualization for Web Pages*, 1ra edición, O'Reilly Media, 2014.
- [8] T. Despoulain, *Shdr*, <http://shdr.bkcore.com>.
- [9] Community Project, *JavaScript 3D library - Three.js*, <http://threejs.org> [Código Fuente - <https://github.com/mrdoob/three.js>].
- [10] Community Project, *Ace - The High Performance Code Editor for the Web*, <http://ace.c9.io> [Código Fuente - <https://github.com/ajaxorg/ace>].
- [11] I. Yuta, *Zlib.js*, <https://github.com/imaya/zlib.js>.
- [12] The jQuery Foundation, *jQuery*, <https://jquery.com>.
- [13] Stanford University Computer Graphics Laboratory, *The Stanford 3D Scanning Repository*, <http://graphics.stanford.edu/data/3Dscanrep> [Modelo - Dragon].
- [14] R. Cabello, *GLSL Sandbox*, http://mrdoob.com/projects/gsl_sandbox [Código Fuente - <https://github.com/mrdoob/gsl-sandbox>].
- [15] B. Bass, *WebGL Shader Lab*, <http://goo.gl/iFsvb>.
- [16] Community Project, *Kick.js*, <http://goo.gl/01sxRs> [Código Fuente - <https://github.com/mortennobel/KickJS>].
- [17] I. Quilez y P. Jeremias, *Shadertoy*, <https://www.shadertoy.com>.
- [18] S. Purewal, *Learning Web App Development*, 1ra edición, O'Reilly Media.
- [19] Google Inc., *AngularJS*, <https://github.com/kooroo/ngCropper>.
- [20] Community Project, *UI Layout*, <http://layout.jquery-dev.com> [Código Fuente - <https://github.com/twbs/bootstrap>].
- [21] F. Balliano y K. Dalman, *Bootstrap*, <http://getbootstrap.com> [Código Fuente - <https://github.com/twbs/bootstrap>].
- [22] Community Project, *AngularJS Wrapper of jQuery Image Cropping Plugin*, <https://github.com/kooroo/ngCropper>.

DanceNet: Entorno Colaborativo para la Enseñanza de Coreografías empleando un Kinect

Andrés Mayo¹, Esmitt Ramírez²
aemc_64@hotmail.com, esmitt.ramirez@ciens.ucv.ve

¹ Escuela de Informática, Universidad Católica Andrés Bello, Caracas, Venezuela

² Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: El baile es una actividad artística donde el cuerpo se expresa a través de movimientos rítmicos. Es posible escribir un baile o danza mediante una composición, en un proceso llamado coreografía. Durante los últimos años la forma predominante para aprender una coreografía ha sido asistiendo a clases en alguna academia de baile, centro cultural o gimnasio con instructores profesionales. No obstante, estas formas están asociadas a factores sociales como disponibilidad del aprendiz y horario de las clases, vergüenza o temor del aprendiz ante otras personas, inversión económica, entre otras. Así, se propone una solución basada en la creación de un entorno colaborativo virtual llamado DanceNet, donde las personas puedan aprender y enseñar coreografías mediante el uso del Kinect como dispositivo de captura de movimientos. La solución se basa en la enseñanza mediante repetición de diversos pasos, y consta de una aplicación para grabar y comparar posiciones de una coreografía. Además, contiene un sitio web para compartir coreografías y conectar a usuarios de distintas partes del mundo interesados en bailar. Nuestras pruebas demostraron la efectividad de DanceNet para comparar movimientos en una coreografía entre diferentes usuarios, así como integrar a personas dentro de un entorno colaborativo con interés en el baile.

Palabras Clave: Kinect; Coreografía; Entorno Colaborativo; Procesamiento Digital de Imágenes; Baile.

Abstract: Dance is an artistic activity where the body express itself using rhythmic movements. It is possible to write a dance using a composition in a process called choreography. During the last years, the predominant way to learn a choreography has been into attended in a class, in some dancing school, cultural centre or with professional instructors in a gym. However, these ways are related to social aspects such as apprentice availability, classes schedule, shame or fear in front other students, economic investment, and so on. Thus, in this paper we proposed a solution based in the creation of a virtual collaborative environment called DanceNet, where people can to learn and teach choreographies using the Kinect as acquisition device. The solution employs the learning by repetition of several dancing steps, and it contains a recorder application which compares the positions into a choreography. Moreover, it includes a web site to share and connect users from different world locations sharing the interest on dance. Our tests shown the effectiveness of DanceNet to compare movements in a choreography between different users, also to integrate them in a collaborative environment with interest in dancing.

Keywords: Kinect; Choreography; Collaborative Environment; Digital Image Processing; Dance.

I. INTRODUCCIÓN

El baile se considera como una actividad totalmente creativa y libre, donde es posible ejecutar movimientos diferentes para expresar los ritmos y tonalidades de una música o canción. Para aprender a bailar, se cuentan con diferentes fuentes de aprendizaje tales como clases de baile, coreógrafos, entre otros. Por otra parte, la mayoría de las clases solo cubren un estilo de baile en particular, el material musical es seleccionado por el instructor, los alumnos se limitan a los pasos dados por el instructor, entre otros factores asociados a dicha experiencia.

Específicamente, para aprender una coreografía se debe dominar las posturas y pasos esenciales de dicha coreografía. Inicialmente, esto se obtiene a través de la imitación, es decir, reproduciendo cada uno de estos movimientos basados en una demostración hasta completar toda la coreografía que se desea aprender. Esto puede resultar un proceso complejo en su ejecución, tal como se muestra en la Figura 1 donde se observa un dibujo planimétrico de una coreografía hecha por la reconocida coreógrafa Trisha Brown.

En la mayoría de los casos, el proceso de aprender una coreografía requiere de una inversión de tiempo, dinero, traslado al

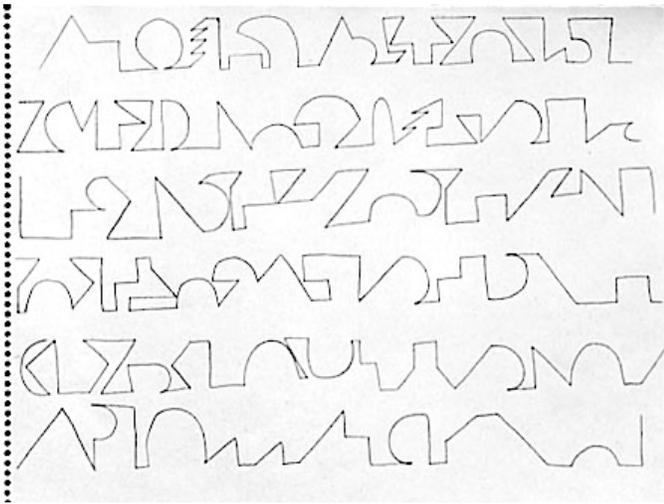


Figura 1: Ejemplo de Dibujo Planimétrico de una Coreografía Realizada de Forma Manual. Tomado de www.trishabrowncompany.org

lugar de aprendizaje, enfrentar la timidez ante otras personas, entre otras. Por ello, han surgido alternativas tales como bailoterapias, zumba, programas de tv, videos en Internet, y videojuegos. Estas alternativas siguen las mismas dinámicas de las clases aunque algunas no están enfocadas al aprendizaje sin posibilidad de recibir algún tipo de retroalimentación de su ejecución.

Actualmente muchas de estas fuentes no hacen uso de las tecnologías actuales y de su expansión a nivel mundial para apoyar a las personas en el aprendizaje del baile o coreografías. Por ello, proponemos una alternativa novedosa que cubre muchas de las desventajas antes mencionadas e involucra tecnologías actuales de alcance masivo.

En este trabajo se propone DanceNet: un entorno colaborativo virtual donde las personas puedan aprender y enseñar coreografías mediante el uso de un dispositivo de captura de movimientos de bajo costo. Este entorno está integrado por una aplicación donde se utiliza un Microsoft Kinect® como un dispositivo de captura de movimientos y un sitio web que funciona como red social de interacción entre los usuarios.

Este artículo se organiza de la siguiente forma: la Sección II muestra una breve introducción del baile en la era moderna así como lo trabajos asociados en esta dirección que emplean dispositivos de bajo costo para el aprendizaje de un baile y coreografía. La descripción completa de nuestra propuesta llamada DanceNet, se detalla en la Sección III. La Sección VI muestra la experimentación realizada para corroborar la eficacia de nuestra propuesta. Finalmente, en la Sección VII se presentan las conclusiones y trabajos futuros sugeridos de nuestro trabajo.

II. EL BAILE EN LA ERA MODERNA

Cohen y Matheson [1] definen la danza como un arte, y lo explican desde la época del ballet de la corte hasta lo que denominan los límites de la ruptura en los años 90. Del mismo modo, aclaran que no se conoce a ciencia cierta desde cuando

el hombre comenzó a expresarse con su cuerpo utilizando movimientos, pero asegura que su primer registro fue en los festivales de primavera hechos por Dionisio.

De este modo, se puede definir a la danza o el baile como una forma de arte que emplea los movimientos del cuerpo (usualmente con música) como una forma de expresión, de interacción social, con fines de entretenimiento, artísticos o religiosos. Dicho movimiento se realiza en un espacio (e.g. escenario) con una parte o todo el cuerpo del ejecutante, bajo cierto compás o ritmo. El baile es considerado una forma de comunicación por su uso de un lenguaje no verbal entre los seres humanos, donde el bailarín o bailarina expresa sentimientos y emociones a través de sus movimientos y gestos.

Existen diversos estilos de baile, desde el ballet, hasta el breakdance y pasando por el krumping. Estos estilos mayormente dependen de factores sociales, culturales, estéticos, artísticos y morales, así como también del rango de movimiento funcional hasta las técnicas virtuosas. Independientemente del estilo, en cada baile se requiere tener flexibilidad y movimiento del cuerpo, así como un cierto grado de condición física.

Dentro del baile existe el concepto de coreografía, que es el arte de diseñar una secuencia de movimientos y formas con el cuerpo o con parte de éste [2]. La persona que crea coreografía, se le conoce como coreógrafo. Una coreografía se puede realizar con un número variado de bailarines, que va desde un ejecutante, en pareja o grupos.

La danza se compone de diversos elementos básicos que se interrelacionan, logrando transmitir emociones al público y para el mismo bailarín. Estos elementos son: movimiento, ritmo, expresión corporal, espacio y el estilo [3]. El uso predominante de uno u otro elemento no es siempre pareja. En ciertos bailes predomina el ritmo, en otros el uso del espacio, en otros el estilo, etc. De acuerdo al tipo y género de baile, se acentuará el uso de uno u otro elemento. Dentro del baile se van creando nuevos géneros y variantes al crear nuevas coreografías y de esta forma se van diversificando.

Los elementos (movimientos) y características específicas de una danza, se utilizan para elaborar una coreografía y a partir de ellos inventar nuevos movimientos para crear nuevas coreografías. La coreografía también es empleada en situaciones especiales, como en la cinematografía, el teatro, musicales, conciertos, performances, eventos y presentaciones artísticas.

A. El Uso de la Tecnología en la Danza

Según Ryden [4], el Kinect como dispositivo de adquisición de imágenes, ha ampliado gran parte de las investigaciones a nivel mundial en diversos centros de investigación debido a la creación de nuevos algoritmos de interacción háptica. Dichos algoritmos son empleados en diversas áreas donde se emplea la tecnología como parte de las actividades diarias, áreas como Medicina, Psicología, Ingeniería, Artes, entre otras.

Un algoritmo de reconocimiento de gestos para danzas clásicas de la India empleando un Kinect es presentado por Saha et al.

[5], donde pueden extraer las emociones expresadas basados en la postura dentro del baile hecho por los bailarines. De hecho, para bailes de la India existen diversas investigaciones que buscan estudiar los movimientos de sus danzas generalmente para e-learning [6][7][8]. Otra caso relevante es el reconocimiento de gestos en la danza tradicional de la etnia Kazajo ubicada en la zona norte de Asia Central [9], donde se presentan movimientos muy particulares que representan estados de ánimo.

Recientemente, Ozcimder [10] presenta un trabajo donde expone la comunicación empleando el movimiento del cuerpo en términos de control cooperativo. Dicho estudio, se basa en extender el trabajo previo de Baillieul y Ozcimder [11] basado en la enseñanza de la salsa a través de un instructor/guía a un grupo de personas. Así, se busca extender un análisis a las interacciones de un líder en el baile de salsa siguiendo una serie de reglas para las transiciones de sus movimientos. Dichas reglas son planteadas como una topología de la teoría de nudos (*knot theory*) [12].

En el caso de la danza clásica o ballet donde los movimientos requieren de un control preciso del cuerpo, existen investigaciones donde se estudia las posturas del cuerpo humano [13][14], se crean estructuras para ser replicadas en robots humanoides [15], y se emplea visión estéreo para capturar adecuadamente posturas propias del ballet [16].

Del mismo modo, además de realizar la captura y reconocimiento de ciertas posturas o gestos, se añaden otras tecnologías como la realidad virtual. En el 2010, Chang et al. [17] proponen un sistema de entrenamiento basado en la captura de movimientos y realidad virtual en el esquema profesor-alumnos. En dicha investigación, los alumnos deben seguir a un profesor virtual que al final ofrece una retroalimentación a cada estudiante por los movimientos realizados (capturados por cámaras web). Basado en esta idea, en el 2012 Yang et al. [18] presentaron un generador automático de lecciones de danza.

Como se mencionó anteriormente, la captura de los gestos y posturas de baile puede ser capturado por un dispositivo de adquisición de imágenes, por ejemplo un Kinect. En el 2011, el robot Asimo creado por Honda [19] pudo reproducir en tiempo real los movimientos de una persona capturada con un Kinect [20]. Esta tarea fue empleada para demostrar su uso en la danza y en ejercicios más precisos como teleoperaciones a través de cinemática inversa. De hecho, el estudio de la danza en robots en un amplio campo de estudio en la robótica [21].

Alexiadis et al. [22] presentan un sistema automático de evaluación de la calidad de un bailarín empleando un Kinect. Posteriormente en el 2012, Essid et al. [23] presentan una mejora a dicho trabajo presentando una versión online sobre el mismo ambiente virtual transmitido a diferentes alumnos conectados vía web a un profesor de baile. Igualmente, existen otras investigaciones recientes que buscan evaluar dichos movimientos que fueron capturados por un Kinect [24][25][26].

Desde el punto de vista del entretenimiento la empresa Mi-

crosoft, empleando su consola de juego Xbox, tiene disponible una gran cantidad de videojuegos donde su punto central es el baile capturado con el Kinect [27]. Entre los más destacados se encuentran Dance Central (con múltiples versiones), Just Dance (con múltiples versiones), Michael Jackson: The Experience, The Black Eyed Peas Experience, Dance Paradise, Dance Master, entre muchos otros. En la Figura 2 se puede observar un instante del juego Dance Central [28] donde se observan los pasos que debe seguir un jugador para obtener puntos dentro del juego.



Figura 2: Un Instante del Juego Dance Central Desarrollado por Harmonix Music System

En el baile, la utilización de dispositivos de captura como el Kinect, es un amplio campo de estudio debido a su importancia en el aprendizaje (virtual o presencial) que facilitan la retroalimentación entre un instructor o profesor, y alumnos o bailarines. A continuación, explicamos en detalle nuestra propuesta DanceNet como un cúmulo de funciones para el aprendizaje de coreografías (gestos y posturas) empleando un Kinect.

III. DANCENET

Empleando el Kinect como dispositivo de captura, un dispositivo de despliegue/salida (e.g. una tv, un monitor) y un computador convencional, es posible ejecutar la propuesta aquí planteada. DanceNet es el nombre de nuestra solución que consiste en un entorno virtual para el aprendizaje y enseñanza de coreografías con dispositivos de bajo costo. La Figura 3 muestra una fotografía de nuestra propuesta en acción.

Nuestra solución contiene un método de aprendizaje y enseñanza de coreografías donde se especificarán las partes del cuerpo humano que serán capturadas con el uso del Kinect. Así, se tendrá una interacción entre el usuario aprendiz, software y usuario coreógrafo. También se detalla una evaluación de las ejecuciones, es decir, el sistema de puntuaciones y escalas así como los márgenes de errores que maneja el software.

Desde un punto de vista general, DanceNet está formado por el sistema de captura y procesamiento de los movimientos, y de



Figura 3: Un Instante de DanceNet, Jugado por Natasha Villamizar de 22 Años de Edad

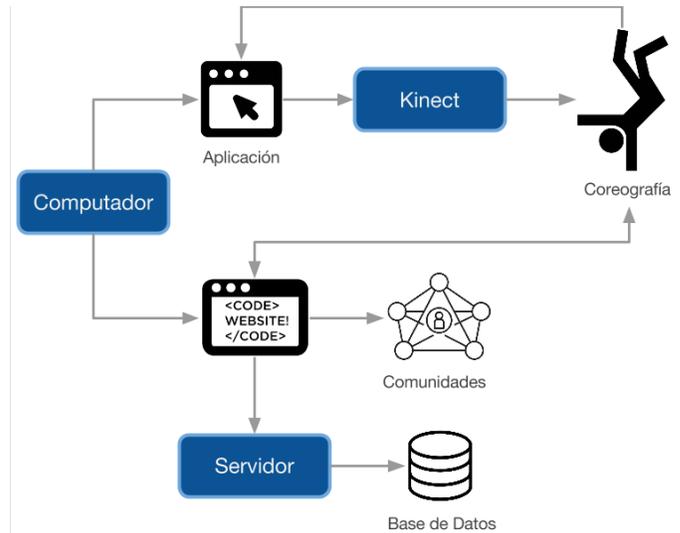


Figura 4: Esquema General de los Componentes Principales de Nuestra Propuesta

un portal web para la difusión/socialización de las coreografías creadas.

El sistema de captura y procesamiento de movimientos se encarga de la creación, reproducción y comparación de coreografías. Es aquí donde se requiere el Kinect como dispositivo de captura de movimientos. En este sistema los usuarios pueden tanto grabar una coreografía y definir las posturas o pasos para aprenderla, como ejecutarla y comparar dicha ejecución con respecto a la original, además de practicar las posturas o pasos definidos por un coreógrafo.

El portal se encarga de la publicación y distribución de las coreografías, así como ser el punto de reunión para los distintos usuarios que quieran aprender, enseñar coreografías o simplemente bailar. En este los usuarios pueden subir, descargar y calificar coreografías, participar en comunidades y realizar comentarios.

De forma general, es posible esquematizar nuestra solución como se muestra en la Figura 4. Se puede observar que existe una aplicación que controla al dispositivo de captura de movimiento (i.e. Kinect) para obtener los pasos y movimientos de una coreografía. Por otro lado, está un sitio web compuesto por un servidor web y almacenamiento permanente (i.e. base de datos) para el manejo y control de comunidades de usuarios.

A continuación se explica de forma detallada las funcionalidades de DanceNet. Para ello, se separa de acuerdo a su área de acción en subsistema de movimientos y subsistema de distribución.

IV. SUBSISTEMA DE MOVIMIENTOS

En el subsistema de movimientos es posible definir dos tareas principales:

- 1) Compartir o enseñar una coreografía.
- 2) Aprender o bailar una coreografía.

Desde el punto de vista técnico denominamos a la persona que utilizará DanceNet como usuario, independientemente si es coreógrafo o no. Entonces, en el caso 1) el usuario utiliza la aplicación para poder grabarse bailando alguna canción o pieza musical, y después puede colocar las posturas o pasos necesarios para aprender la coreografía. Por último, debe guardar la coreografía como un archivo en disco. Dicho archivo, puede alojarse en el portal web donde también puede compartirla a través de las comunidades.

En el caso 2), el usuario debe primero buscar en el portal web una coreografía que desee bailar o aprender y descargar el archivo correspondiente. Este archivo debe ser leído por la aplicación lo que permite que el usuario pueda practicar las posturas o pasos definidos antes de ejecutar la coreografía y conocer su calificación. Finalmente podrá compartir su experiencia en las comunidades.

Para ello, se tiene una aplicación que consta de un módulo de detección y registro de movimientos, un módulo de comparación y análisis de movimientos, y un módulo de aprendizaje y enseñanza de coreografías.

El módulo de detección y registro de movimientos permite detectar las partes del cuerpo humano, las cuales forman distintas posiciones a lo largo del tiempo mientras se reproduzca un audio seleccionado por el usuario. Estas posiciones detectadas son reproducidas en un modelo humano que es observado por el usuario mientras ejecuta la coreografía. El registro de las posiciones produce la coreografía digitalmente.

El módulo de comparación y análisis de movimientos se encarga de comparar en tiempo real la ejecución de una coreografía realizada por un usuario en base a la realizada por el usuario autor de la coreografía. El módulo permite indicar a través del modelo humano cuando una parte del cuerpo y/o posición en el espacio coincide o no coincide según se defina.

Una vez grabada la coreografía, el módulo de aprendizaje y enseñanza permite al usuario autor de la coreografía in-

dicar aquellos elementos como posturas, pasos y comentarios necesarios para su aprendizaje. Para indicar una postura, se selecciona en una coreografía el instante de tiempo que represente la postura. Para indicar un paso se debe seleccionar el intervalo de tiempo en la coreografía que represente el paso. Una coreografía podrá ser exportada por el usuario autor e importada por otros usuarios en donde podrán seleccionar y aprender estos elementos previamente definidos individualmente.

Adicionalmente, es posible ejecutar una coreografía para posteriormente ser evaluada, es decir, realizar un análisis general indicando a través de una escala qué tanta semejanza tuvo la ejecución con respecto a la original, así como cuáles posturas y pasos fueron no fueron dominados.

A. Módulo de Detección y Registro

La primera tarea a realizar es separar la imagen del usuario con respecto al fondo donde se encuentra. DanceNet obtiene solamente la imagen que representa la silueta de la persona que se encuentre frente a Kinect, eliminando cualquier otro detalle de fondo. La silueta se dibuja en color verde (solo para colocar un color uniforme que haga contraste). El fondo se dibuja en color negro, es decir, el valor de RGB (0,0,0).

La Figura 5 muestra un ejemplo de una imagen capturada (parte (a)) y su procesamiento para extraer solo la silueta del usuario, marcado en color verde con fondo en color blanco (parte (b) y el fondo es blanco solo para fines demostrativos en este trabajo).

Para que la representación sea lo más fiel posible se registran las posiciones de todos los puntos soportados por Kinect (i.e. 25 puntos). Sin embargo, no es obligatorio que los 25 puntos sean detectados todo el tiempo durante la grabación, es decir, para una posición específica se registran solo los puntos que son detectados en el instante de tiempo en que ocurre dicha posición. En caso de no ser detectados, son ignorados, tomando un % mínimo de captura.

Una vez capturada las posiciones, se requiere comparar las posiciones obtenidas con una previamente almacenada para su análisis. A continuación se explica dicho módulo.

B. Módulo de Comparación y Análisis

En la comparación entre ejecuciones de una coreografía en necesario conocer:

- 1) Si una parte del cuerpo coincide.
- 2) El grado de precisión de la posición realizada.
- 3) El grado de precisión general de la ejecución realizada.

Para saber si una parte del cuerpo coincide, se compara la posición de la parte capturada en los ejes cartesianos X , Y y Z con respecto a la posición original. Así, se define un rango en metros que establece los límites de los valores donde se puede encontrar la parte capturada para cada eje. En la Tabla I se presenta un ejemplo para la posición de la mano derecha.



(a) Ejemplo de Imagen Capturada



(b) Extracción Realizada por DanceNet

Figura 5: Proceso Inicial de Adquisición de una Imagen con el Kinect y su Procesamiento en DanceNet

Tabla I: Valores Empleados en la Comparación de una Parte del Cuerpo Capturado

Parte del cuerpo	Mano derecha
Posición en X	0.5 m
Rango	1 m
Valores admitidos en X	[-0.5, 1.5] m

Según estos valores, en cualquier otra ejecución si el usuario coloca su mano derecha entre -0.5 m y 1.5 m en el eje X , la parte coincide con la posición original en el eje X . De esta forma, una parte detectada se considera que coincide si su posición para cada eje se encuentra dentro del rango establecido, en caso contrario, se considera que no coincide.

Desde el punto de vista gráfico, la Figura 6 muestra un ejemplo para un punto detectado por Kinect en la coreografía grabada. El rango definido se muestra a través del cuadro verde, por lo que esa misma parte en cualquier otra ejecución deberá colocarse dentro de ese cuadro para considerarse un acierto.

El grado de precisión de la posición realizada se obtiene aplicando la comparación anterior a todas las partes que fueron detectadas por la ejecución original y almacenando el número de partes que coincidieron. Luego se calcula un porcentaje de precisión cuyo resultado puede ir desde 0% si ninguna



Figura 6: Rango para Comparar Partes en ejes X y Y

parte coincidió o 100% si todas las partes coincidieron. Este porcentaje se obtiene a través de la ecuación 1.

$$Pr(p) = \frac{Co}{Gr} \times 100 \quad (1)$$

donde Pr representa al valor de la precisión en la posición p , Co indica el número de coincidencias, y Gr el número de posiciones grabadas. Dependiendo del resultado obtenido, se califica la posición de la persona que esté ejecutando la coreografía en un rango de 0 a 2 donde: 0 marcado en color rojo indica el estado *Malo*; 1 marcado de color naranja indica el estado *Regular* y; 2 marcado en color verde indica el estado *Bueno*.

Para cada valor, la silueta de la persona cambiará según el color representado. En la Figura 7 se puede observar el rango definido aplicado a una posición. Nótese que a medida que el movimiento se aleje de la esperada el resultado será marcado en color rojo.



Figura 7: Muestra Visual del Rango Aplicado a una Posición

Estos valores son registrados por el sistema para cada una de las posiciones. Con dichos valores se calcula el porcentaje de precisión de la ejecución e con respecto a la coreografía grabada, como se muestra en la ecuación 2.

$$Pr(e) = \frac{\sum_{i=1}^n Ca}{2n} \times 100 \quad (2)$$

donde n representa el número total de posiciones, y Ca la calificación obtenida en el rango $[0, 2]$. Así, mientras sea mayor el porcentaje obtenido, mayor es el dominio de la coreografía que tenga el usuario.

Finalmente, de acuerdo a nuestras pruebas, definimos la asociación precisión-calificación de *Malo* a un rango de precisión de $[0, 49]\%$, de *Regular* a un rango de precisión de $[50, 74]\%$, y de *Bueno* a un rango de precisión de $[75, 100]\%$.

C. Módulo de Aprendizaje y Enseñanza

La idea principal del módulo de aprendizaje y enseñanza es permitir a DanceNet añadir posturas, pasos y comentarios para el aprendizaje de la coreografía. Esta funcionalidad ocurre una vez que el usuario ha finalizado la grabación de la coreografía y se ha generado tanto el video como las posiciones capturadas. Cabe destacar que las posiciones capturadas consisten en una serie de valores (x, y) para una distancia $depth$ del usuario frente al Kinect.

Para ello, se construyó un formulario que permite añadir, modificar o eliminar posturas y movimientos de un listado. Una postura se define como una posición del cuerpo en un instante de tiempo dado dentro de una coreografía. Por su parte, un movimiento consiste en una secuencia de posturas en un período de tiempo menor a la duración total de la coreografía.

En el formulario el usuario debe escribir el nombre, tiempo y comentarios relacionados a la postura o pasos. Dicho formulario emplea un reproductor de video para que el usuario pueda observar la ejecución que realizó. Así, se puede ubicar la posición del video en el tiempo definido del elemento seleccionado de la lista, además de añadir o editar un elemento, y obtener los tiempos a través de la posición del video (ver Figura 8). Para ello, fue necesaria la construcción de estructuras de datos eficientes para el almacenamiento y obtención de las posturas y pasos.

Gracias al uso de las estructuras de datos (basados en colas de prioridad), es posible comprimir el video capturado, los comentarios, las posturas, y los pasos junto con todas las posiciones en un formato propio (archivo de extensión .cho). Dicho formato va a permitir exportar/importar toda una sesión coreográfica para su posterior comparación.

En la comparación, se consideran nuevamente los métodos explicados con la ecuación 2 y el indicador visual de color para cada rango. La Figura 9 muestra un ejemplo de ejecución donde del lado derecho (color verde) es la coreografía almacenada en formato .cho en DanceNet. Del lado izquierdo (color naranja), se muestra al usuario repitiendo los pasos. Para dicho ejemplo, el usuario obtuvo un 53.51% de precisión para una canción de 1:46 m de duración.

Cabe destacar que DanceNet es una herramienta para aprendizaje de ciertas coreografías hechas previamente por un usuario. Al mismo tiempo, al solo permitir el uso de siluetas se

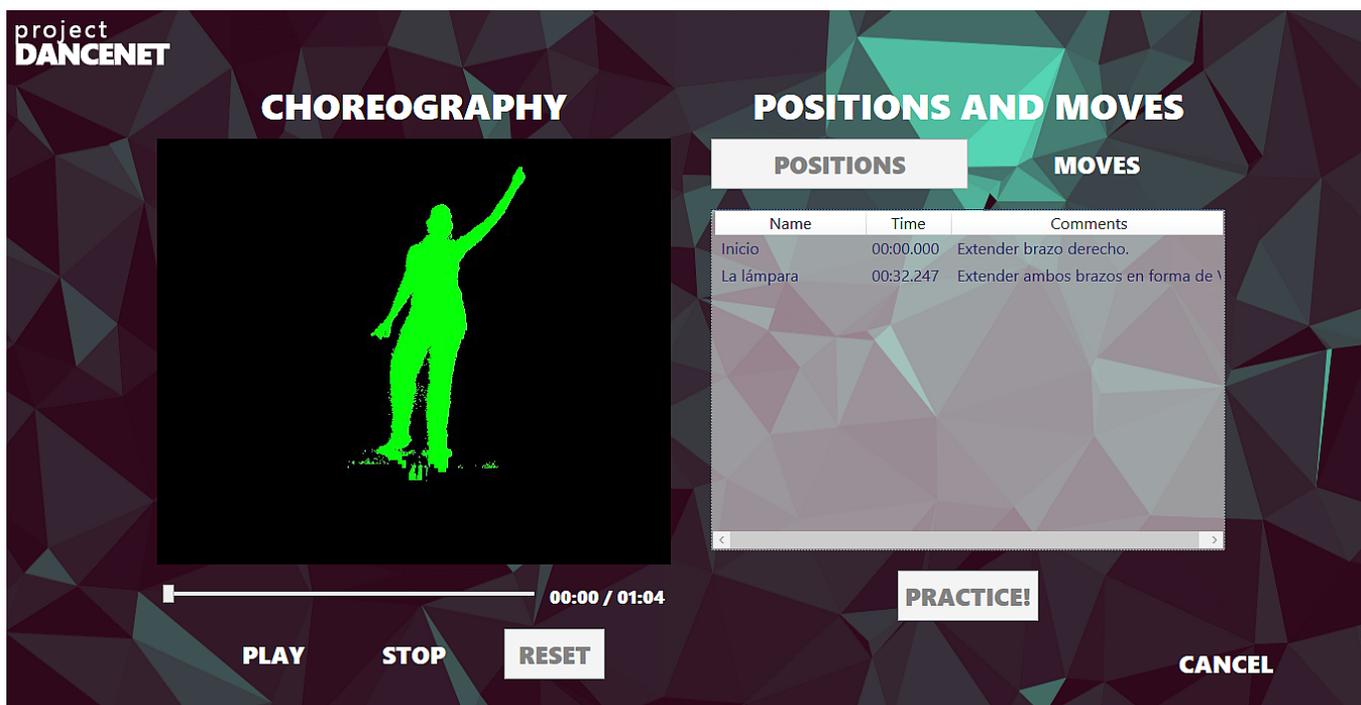


Figura 8: Imagen Capturada donde se Muestra un Ejemplo de la Lista de Posturas (positions) y Pasos (moves)

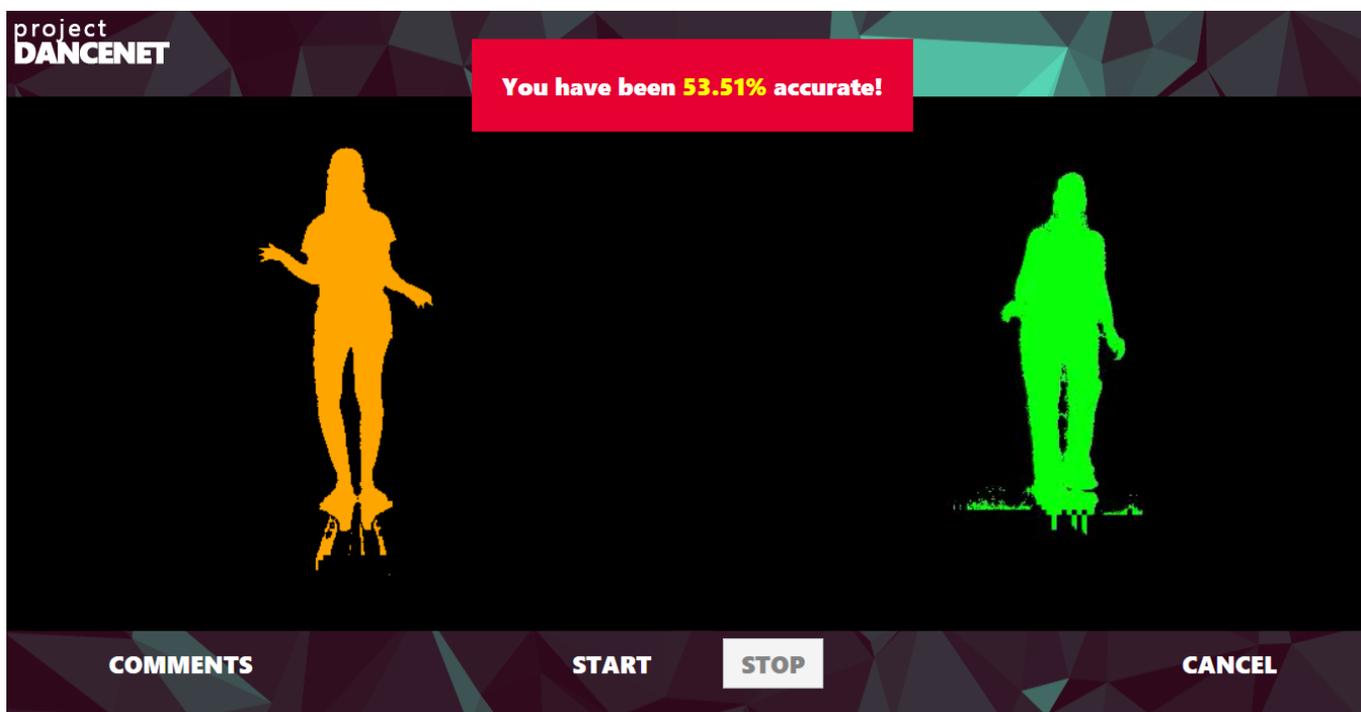


Figura 9: Un Ejemplo de la Ejecución Realizada por un Usuario (Lado Derecho) con una Coreografía Previa (Lado Izquierdo)

preserva la identidad tanto del coreógrafo (quien grabó) y del bailarín (quién la ejecuta posteriormente). Ahora, debe existir algún mecanismo de intercambio, distribución, y generación de coreografías para poder tener una amplia variedad para la enseñanza y aprendizaje. Por ello, nuestra propuesta incluye un sistema de distribución constituido por un sitio web.

V. SUBSISTEMA DE DISTRIBUCIÓN

El subsistema de distribución pretende simular un entorno colaborativo para las coreografías en una comunidad.

A. Entorno Colaborativo

El término *entorno colaborativo*, generalmente hace referencia a una red virtual que facilita la colaboración e interacción

entre los miembros de una comunidad online. En este sentido, se está potenciando al máximo la característica comunicativa de la informática; su capacidad para desarrollar de forma espontánea todo un sistema de información y proyectos, núcleos socioculturales, servicios informativos, etc. Esto resulta posible a través de la web 2.0, un término que en líneas generales se refiere a un conjunto de plataformas que facilitan y favorecen esa interacción online.

Estos entornos colaborativos abarcan desde los editores de documentos compartidos como Google Docs, sitios web abiertos y editables como los wikis, foros y blogs como MySpace o Blogger, o incluso las notorias redes sociales como Twitter o Facebook, que incluyen perfiles y herramientas de conexión para aportar un enfoque colaborativo dentro de la comunidad.

Un entorno colaborativo permite a un grupo de personas acceder a un determinado espacio virtual donde pueden compartir cualquier tipo de documentos, pudiendo cada uno de los miembros del grupo de trabajo revisar los documentos, versionarlos, calificarlos o modificarlos, quedando registrada la fecha de la última modificación y pudiendo acceder también al historial de las diferentes versiones realizadas. Además, estos entornos disponen de herramientas de comunicación en tiempo real (chat o conferencia) como en diferido (correo electrónico) [29].

B. Portal Web

En DanceNet se cuenta con un portal web conformado por un módulo de carga y descarga de coreografías, un módulo de perfiles, un módulo de comunidades y un módulo de estadísticas.

El módulo de carga y descarga permite que una persona pueda ver, buscar, subir coreografías exportadas y descargar coreografías publicadas para ser usadas en la aplicación. El módulo de perfiles permite a los usuarios tener su propio perfil con información básica sobre el usuario. Un perfil puede ser calificado por otros usuarios de forma positiva o negativa, también puede ser indicado como favorito. En el perfil se puede observar aquellas coreografías que el usuario se encuentra aprendiendo, coreografías dominadas, coreografías favoritas, usuarios favoritos (bailarines/coreógrafos) y coreografías creadas.

El módulo de comunidades permite crear espacios donde distintos usuarios podrán discutir acerca de una temática definida. Un usuario podrá crear una comunidad, unirse y escribir comentarios en ella. Igualmente, en el perfil de cada usuario, se podrá observar aquellas comunidades a las que pertenece. Las coreografías contarán con un híbrido entre un perfil y una comunidad, es decir, su propia página donde se podrá observar la información básica de la coreografía, calificarla, así como realizar comentarios. En esta página los usuarios podrán indicar si se encuentran aprendiendo la coreografía o si ya la dominan.

Por último, el módulo de estadística que consiste en distintas clasificaciones (*ranking*) que podrán ser observados por los

usuarios como comunidades más activas, coreografías más descargadas, coreografías con mayor cantidad de calificaciones positivas y coreógrafos más activos.

C. Módulo de Carga y Descarga

Un usuario puede exportar (descargar) una coreografía creada desde la PC conectada al Kinect en formato .cho. Posteriormente, es posible importar (cargar) dicho archivo al sitio web. En el sitio web, una vez importada la imagen el servidor extrae el video del archivo y toma el primer cuadro de éste para usarlo como imagen referencial (*thumbnail*). El proceso completo para importar una coreografía se muestra en la Figura 10.

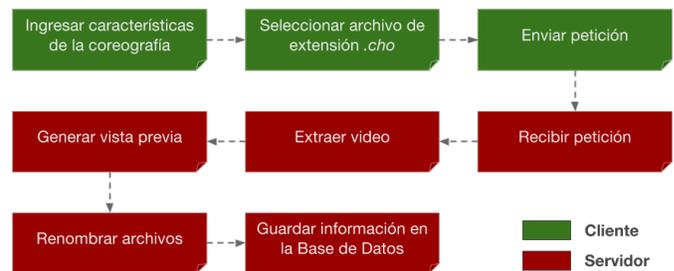


Figura 10: Esquema del Proceso para Importar una Coreografía

Se puede observar en la Figura 10, donde desde el lado cliente (color verde) se ingresan las características de la coreografía y se selecciona el archivo de coreografía (formato .cho). Luego, se envía la petición que la recibe el lado servidor (color rojo). En el servidor, el archivo .cho extrae/descomprime el video para generar una vista previa o imagen referencial. Para manejar una estructura interna, se renombran la coreografía recibida y se almacena en una base de datos.

D. Módulo de Perfiles

La información básica de los usuarios se encuentra en el módulo de perfiles. Cada usuario en el sitio web es tratado tanto bailarín como coreógrafo, de esta forma es posible conocer las coreografías que se encuentra aprendiendo, aquellas que ya ha aprendido, sus coreografías favoritas así como las coreografías que ha realizado. Esta información también es accesible desde la página de una coreografía, la cual funciona como el perfil de una coreografía. Se puede conocer los usuarios que se encuentran aprendiendo la coreografía como los usuarios que la han aprendido. El perfil de la coreografía también muestra la información básica de la coreografía, permite ver el video, descargar el archivo .cho, realizar comentarios y calificar la coreografía.

La Figura 11 muestra un ejemplo de perfil de una coreografía subida al portal web. En esta se puede observar el video, la información básica, calificar la coreografía y descargar el archivo de extensión .cho.

E. Módulo de Comunidades

Las comunidades son espacios creativos para la discusión de una temática definida. Así como los usuarios son autores de

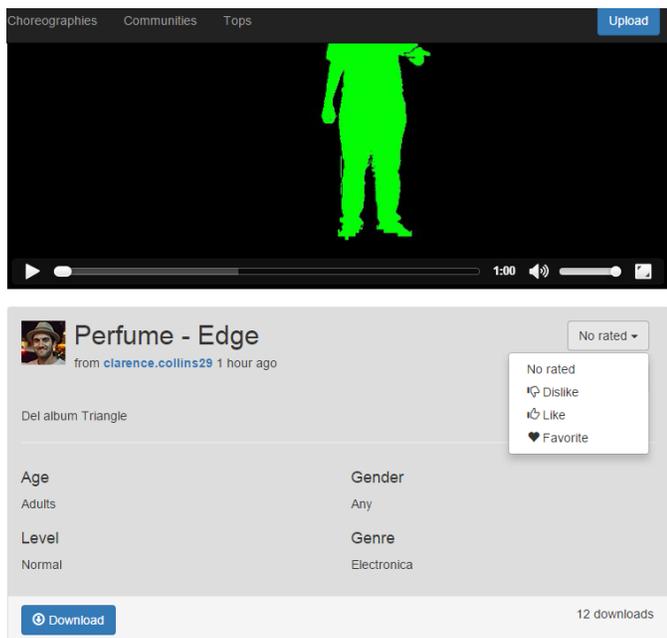


Figura 11: Ejemplo de un Perfil de Usuario que Permite ser Calificado por Otros Usuarios del Portal

las coreografías, también lo serán de las comunidades. Un usuario puede crear una comunidad para discutir sobre un tema a través de un formulario en el que deberá ingresar el nombre de la comunidad, una descripción y una imagen de referencia. Otros usuarios podrán unirse a la comunidad para poder realizar comentarios y conocer a otros usuarios que tengan intereses en común.

La Figura 12 muestra un ejemplo de una comunidad. En este caso, el nombre es Michael Jackson, donde se puede observar quién lo creo, qué miembros lo forman y los espacios de discusión (comentarios) dentro del portal.

F. Módulo de Estadísticas

Las estadísticas consisten en diferentes listados en los que los usuarios consumidores podrán conocer lo mejor del portal web como aquellos usuarios creadores podrán esforzarse por obtener mejores puestos dentro de estos listados. Coreografías más descargadas, coreografías mejor calificadas, coreógrafos más activos y comunidades más activas son los listados que fomentarán la participación dentro del usuario web.

Una vez estudiados todos los aspectos de nuestra propuesta, se realizaron unas pruebas para demostrar la efectividad de DanceNet como un entorno colaborativo interactivo y virtual para el aprendizaje y enseñanza de coreografías mediante el uso de un Kinect.

VI. EXPERIMENTACIÓN

El subsistema de movimientos de DanceNet fue desarrollado bajo el sistema operativo Windows 8.1 empleando el IDE Visual Studio 2013. El lenguaje base fue C# con interfaces gráficas WPF (*Windows Presentation Foundation*), empleando el

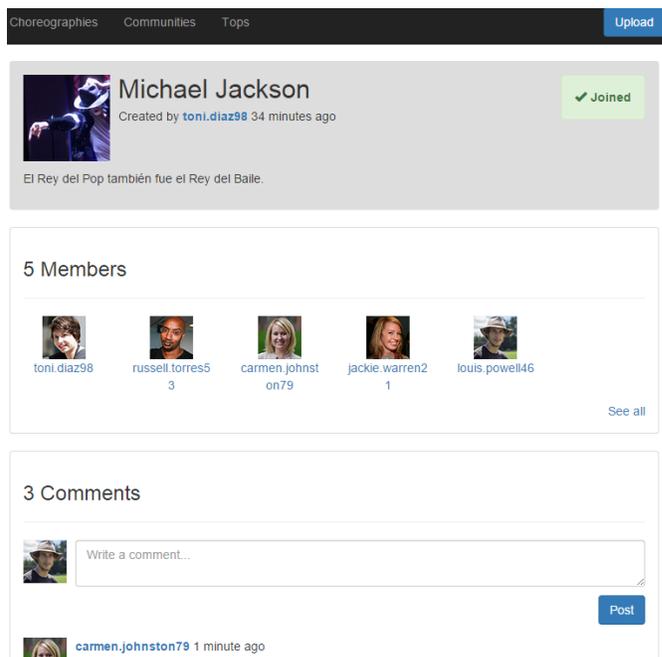


Figura 12: Ejemplo de una Comunidad, Llamada Michael Jackson, Dentro del Portal de DanceNet

SDK de Kinect en su versión 2.0. Por otro lado, la generación de los videos fue con la biblioteca AForge.NET (versión 2.2.5) y su procesamiento con FFmpeg (versión 2.6.3).

Por su lado, el subsistema de distribución requirió como lenguajes base a PHP, HTML, CCS y JavaScript. Del mismo modo, se emplearon las bibliotecas/frameworks Laravel, Bootstrap, y JQuery. Todo sobre un servidor Apache (versión 2.4.12) y un manejador de base de datos MySQL en su versión 5.6.24.

Como parte de los experimentos realizados, se definió ciertas propiedades específicas del video las cuales se muestran en la Tabla II.

Tabla II: Valores por Defecto Definidos para el Video de una Coreografía

Característica	Valor
Ancho de la imagen	512 píxeles
Alto de la imagen	424 píxeles
Cuadros por segundo	30
Códec	MPEG4 (.mp4)
Tasa de bits	1.000.000 bits

Una vez definido los parámetros iniciales, se realizaron 3 pruebas distintas: de precisión, de aprendizaje y de enseñanza. Por último, se realizó una encuesta para conocer la opinión de usuarios comunes ante DanceNet. A continuación cada una de éstas.

A. Pruebas de Precisión

Estas pruebas consisten en evaluar la precisión con la que se comparan los movimientos en distintos niveles. Para ello se necesitaron 2 bailarines. El primer bailarín se encargó de

grabar una coreografía de 2 minutos, mientras que el segundo se encarga de bailarla 5 veces indicadas por los niveles de precisión de la aplicación. La coreografía grabada era conocida por ambos bailarines, sin embargo, no se revelaron los resultados hasta terminar la última ejecución.

Se ajustó la comparación de posiciones de acuerdo a los niveles de precisión presentados en la Tabla III.

Tabla III: Ajustes de Precisión de DanceNet

Nivel	Rango	Bueno	Medio	Alto
Muy baja	0.30 m	0% - 29%	30% - 49%	50% - 100%
Baja	0.25 m	0% - 39%	40% - 59%	60% - 100%
Regular	0.20 m	0% - 49%	50% - 69%	70% - 100%
Alta	0.15 m	0% - 59%	60% - 79%	80% - 100%
Muy alta	0.10 m	0% - 69%	70% - 89%	90% - 100%

Luego de precisar los parámetros para las pruebas con los bailarines, se capturaron los datos resultados de las ejecuciones. La Tabla IV muestra dichos valores.

Se pudo observar que a medida que aumenta el nivel el bailarín (desde la ejecución #1 hasta la ejecución #5) debía ser mucho más preciso (segunda columna) para alcanzar un porcentaje de precisión alto (última columna). Con estos resultados se comprobó que la comparación de movimientos y las calificaciones obtenidas son válidas.

Tabla IV: Ejecuciones de Pruebas de Precisión

Nro. de Ejecución	Nivel de precisión	% de precisión
1	Muy baja	94.13%
2	Baja	76.55%
3	Regular	59.93%
4	Alta	43.48%
5	Muy alta	20.75%

B. Pruebas de Aprendizaje

Estas pruebas permitieron medir el aprendizaje de los bailarines a través de diferentes ejecuciones de una coreografía y su revisión. Se utilizó un grupo de 6 personas: 1 coreógrafo que grabará una coreografía de 2 minutos y 5 personas deberán repetirla 4 veces intentando lograr el mejor porcentaje de precisión. Al final de cada ejecución los bailarines pudieron ver una repetición en video de ejecución para que pudieran revisarla.

Los resultados de las ejecuciones realizadas se presentan en la Tabla V.

Tabla V: Ejecuciones de Pruebas de Aprendizaje

Usuario	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4
A	22.62%	37.14%	66.56%	82.98%
B	44.33%	55.02%	61.94%	67.28%
C	39.00%	47.86%	53.27%	62.20%
D	23.61%	44.02%	56.62%	66.74%
E	42.57%	63.06%	69.71%	78.28%

Se puede observar en los resultados que a medida que los usuarios (bailarines) realizaban las coreografías, eran capaces de mejorar sus porcentajes de precisión en cada nuevo intento,

donde este aumento en promedio resultó ser de 12.36% entre ejecuciones.

C. Pruebas de Enseñanza

Permitieron comprobar la utilidad de las herramientas de enseñanza incluidas en la aplicación. Para estas pruebas se utilizó un coreógrafo y dos bailarines. El coreógrafo grabó una coreografía de 2 minutos que se descomponía principalmente en 4 pasos de 10 segundos. Se indicaron tanto estos 4 pasos como las posturas necesarias para el aprendizaje de la coreografía. Los bailarines se encargarían de realizar la coreografía 2 veces, solo que a uno de ellos se le permitió practicar los 4 pasos definidos previamente.

Los resultados de las ejecuciones de prueba se muestran en la Tabla VI.

Tabla VI: Ejecuciones de Pruebas de Enseñanza

Usuario	Ejecución 1	Ejecución 2
Con práctica de pasos y posturas	48.71%	74.33%
Sin práctica de pasos y posturas	25.49%	42.20%

Con estos resultados se pudo observar que la práctica permite obtener un porcentaje de precisión mucho más elevado incluso desde la primera vez que se realiza la coreografía. De esta forma el bailarín podría obtener un porcentaje cercano al 100% realizando menos ejecuciones.

D. Encuesta

Con el fin de conocer la reacción de usuarios promedio (en edades entre 20 y 35 años) ante el uso de DanceNet, se realizó una encuesta a 15 personas con las siguientes preguntas:

- 1) ¿Considera que DanceNet ofrece herramientas para aprender y enseñar coreografías?
- 2) ¿Los movimientos grabados son fáciles de reconocer?
- 3) Durante la comparación, ¿los movimientos se califican (bueno, regular, malo) de forma adecuada?
- 4) ¿Considera que practicar las posturas y pasos de una coreografía facilita su aprendizaje?
- 5) Según su opinión, ¿la aplicación es fácil de manejar?

Esta encuesta de evaluación se basa en la escala originalmente propuesta por Likert empleando 5 niveles: totalmente de acuerdo, de acuerdo, regular, desacuerdo y totalmente en desacuerdo. La Figura 13 muestra el resultado del promedio de la encuesta realizada, donde item-i representa la pregunta i-ésima.

Según los usuarios DanceNet es una aplicación con la que pueden tanto aprender y enseñar coreografías de forma fácil, donde la interfaz gráfica propuesta entre teclado y ratón y además con controles por movimientos no representa obstáculos para el aprendizaje. Aunque solo se captura la silueta de la persona, los movimientos son fáciles de reconocer y se pueden ejecutar sin dificultad. En la comparación de movimientos, los usuarios aceptan las calificaciones obtenidas con las que pueden observar los momentos en los cuales se equivocaron y

RESULTADOS: ENCUESTA SOBRE LA APLICACIÓN

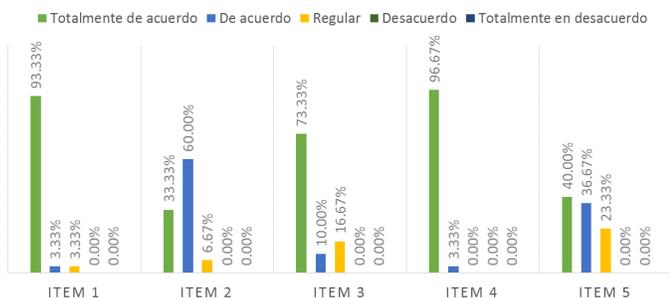


Figura 13: Resultados de una Encuesta Formada por 5 Preguntas ante 15 Usuarios sobre DanceNet

mejorar la precisión de la ejecución, precisión que puede ser mejorada rápidamente con la práctica de posturas y pasos.

VII. CONCLUSIONES

En este trabajo se presentó DanceNet, un entorno colaborativo basado en herramientas modernas que facilitan el aprendizaje y la enseñanza de coreografías, empleando el Kinect para la captura de movimiento de bailarines. De este modo, se presentan dos entes principales: una aplicación conectada al Kinect para la captura y manejo de coreografías y un portal web que permite una distribución, difusión, aprendizaje y enseñanza de las coreografías realizadas con la aplicación.

El uso de colores en las siluetas para la comparación de las diversas posiciones entre coreografías, logra una forma rápida y original de que los usuarios reconozcan cuando sus posiciones coinciden o no. El uso de porcentajes permite establecer correctamente la precisión de las ejecuciones y ayuda a ajustar los movimientos de los usuarios practicantes. Igualmente, la definición de posturas y pasos en la coreografía representa una herramienta sencilla para que los usuarios puedan facilitar el aprendizaje de sus coreografías en pocos pasos. Así, se ahorra tiempo al momento de realizar prácticas repetitivas.

Con los perfiles de usuario, se da una importante opción de difusión, distribución y socialización dentro de una comunidad. Las calificaciones de los perfiles ayudan a que los usuarios puedan conocer si deben mejorar sus coreografías. Estas comunidades representan espacios libres donde los usuarios pueden discutir sobre alguna temática en particular como estilos de baile, géneros de música, eventos, artistas, e incluso utilizar su creatividad para aprovecharlas y transformarlas aulas virtuales.

En un futuro, se propone como aspecto principal incluir coreografías en parejas o grupos. Actualmente, DanceNet está diseñado para la ejecución coreográfica de una sola persona. Igualmente, se propone añadir la opción de definir las partes a ser detectadas por la aplicación para personas con discapacidades visuales media (i.e. configuración de colores). Desde el punto de vista del portal web, una opción importante a ser añadida en un futuro es la creación de tutoriales o ayudas creadas por la misma comunidad.

Del mismo modo, a pesar de tener unas mecánicas de juego (i.e. porcentaje de acierto, votaciones en el portal), sería

ideal contar con otras que ofrezcan niveles de bailarines (fácil, medio, avanzado) y algunas recompensas a medida del progreso del juego para agregar incrementar las estrategias de gamificación en DanceNet y motivar a una mayor cantidad de usuarios.

REFERENCIAS

- [1] S. Cohen y K. Matheson, *Dance As a Theatre Art: Source Readings in Dance History from 1581 to the Present*, Princeton Book Company, 2da edición, 1992.
- [2] J. Winearls, *Choreography: The Art of the Body*, Dance Books Ltd, 2da edición, 2011.
- [3] F. Yarel, *Elementos de la Danza*, <http://danza102.blogspot.com/2013/05/elementos-de-la-danza.html>.
- [4] F. Ryden, *Tech to the Future: Making a "Kinect" with Haptic Interaction*, IEEE Potentials, vol. 31, no. 3, pp. 34–36, Mayo 2012.
- [5] S. Saha, S. Ghosh, A. Konar, y A.K. Nagar, *Gesture Recognition from Indian Classical Dance Using Kinect Sensor*, in proceedings of the Fifth International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN), pp. 3–8, Madrid, España, Junio 2013.
- [6] S. Saha, S. Ghosh, A. Konar, y R. Janarthanan, *Identification of Odissi Dance Video Using Kinect Sensor*, in proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1837–1842, Mysore, India, Agosto 2013.
- [7] S. Saha, S. Ghosh, A. Konar, y R. Janarthanan, *A Study on Leg Posture Recognition from Indian Classical Dance Using Kinect Sensor*, in proceedings of the International Conference on Human Computer Interactions (ICHCI), pp. 1–6, Chennai, India, Agosto 2013.
- [8] S. Gupta y S. Goel, *PoGest: A Vision Based Tool for Facilitating Kathak Learning*, in proceedings of the Seventh International Conference on Contemporary Computing (IC3), pp. 24–29, Noida, India, Agosto 2014.
- [9] A. Nussipbekov, E. Amirgaliyev y M. Hahn, *Kazakh Traditional Dance Gesture Recognition*, Journal of Physics: Conference Series, vol. 495, no. 012036, Mayo 2012.
- [10] K. Ozcimder, *Communication Through Motion in Dance with Topological Constraints*, in proceedings of the American Control Conference (ACC), pp. 178–183, Portland, USA, Junio 2014.
- [11] J. Baillieul y K. Ozcimder, *The Control Theory of Motion-Based Communication: Problems in Teaching Robots to Dance*, in proceedings of the American Control Conference (ACC), pp. 178–183, Montreal, Canadá, Junio 2012.
- [12] C. Livingston, *Knot Theory*, ser. Mathematical Association of America Textbooks, 1ra edición, The Mathematical Association of America, 1993.
- [13] S. Saha, A. Banerjee, S. Basu, A. Konar, y A.K. Nagar, *Fuzzy Image Matching for Posture Recognition in Ballet Dance*, in proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ), pp. 1–8, Hyderabad, India, Julio 2013.
- [14] A. LaViers y M. Egerstedt, *The Ballet Automaton: A Formal Model for Human Motion*, in proceedings of the American Control Conference (ACC), pp. 1–6, San Francisco, USA, Junio 2011.
- [15] A. LaViers, Y. Chen, C. Belta y M. Egerstedt, *Automatic Sequencing of Ballet Poses*, IEEE Robotics Automation Magazine, vol. 18, no. 3, pp. 87–95, Septiembre 2011.
- [16] F. Guo y G. Qian, *Dance Posture Recognition Using Wide-Baseline Orthogonal Stereo Cameras*, in proceedings of the 7th International Conference on Automatic Face and Gesture Recognition, pp. 481–486, Southampton, Inglaterra, Abril 2006.
- [17] J.C.P. Chan, H. Leung, J.K.T. Tang y T. Komura, *A Virtual Reality Dance Training System Using Motion Capture Technology*, IEEE Transactions on Learning Technologies, vol. 4, no. 2, pp. 187–195, Abril 2014.
- [18] Y. Yang, H. Leung, Y. Lihua y L. Deng, *Automatic Dance Lesson Generation*, IEEE Transactions on Learning Technologies, vol. 5, no. 3, pp. 191–198, Julio 2012.
- [19] American Honda Motor Co. Inc., *Asimo*, <http://asimo.honda.com>.
- [20] E. Guizzo, *Asimo Can Copy Your Dance Moves*, <http://spectrum.ieee.org/automaton/robotics/humanoids/asimo-can-copy-your-dance-moves>.
- [21] H. Peng, C. Zhou, H. Hu, F. Chao y J. Li, *Robotic Dance in Social Robotics – A Taxonomy*, IEEE Transactions on Human-Machine Systems, vol. 45, no. 3, pp. 281–293, Junio 2015.
- [22] D. Alexiadis, P. Kelly, P. Daras, N. O'Connor, T. Boubekour y M. B. Moussa, *Evaluating a Dancer's Performance Using Kinect-based Skeleton Tracking*, in proceedings of the 19th ACM International Conference on Multimedia, pp. 659–662, Scottsdale, USA, Noviembre 2011.

- [23] S. Essid, D. Alexiadis, R. Tournemene, M. Gowing, P. Kelly, D. Monaghan, P. Daras, A. Dreameau y N.E. O'Connor, *An Advanced Virtual Dance Performance Evaluator*, in proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2269–2272, Kyoto, Japón, Marzo 2012.
- [24] A. Liutkus, A. Dreameau, D. Alexiadis, S. Essid y P. Daras, *Analysis of Dance Movements Using Gaussian Processes: Extended Abstract*, in proceedings of the 20th ACM International Conference on Multimedia (MM'12), pp. 1375–1376, Nara, Japón, Octubre 2012.
- [25] A. Dreameau y S. Essid, *Probabilistic Dance Performance Alignment by Fusion of Multimodal Features*, in proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3642–3646, Vancouver, Canadá, Mayo 2012.
- [26] M. Kyan, G. Sun, H. Li, L. Zhong, P. Muneesawang, N. Dong, B. Elder, y L. Guan, *An Approach to Ballet Dance Training Through MS Kinect and Visualization in a CAVE Virtual Reality Environment*, ACM Transactions on Intelligent Systems and Technology, vol. 6, no. 2, pp. 23:1–23:37, Marzo 2015.
- [27] Microsoft Corporation, *Xbox 360 + Kinect*, <http://www.xbox.com/en-SG/Kinect/Games>.
- [28] Harmonix Music Systems, Inc., *Dance Central*, <http://www.xbox.com/en-US/games/dance-central>.
- [29] E. Barkley, C. Howell y P. Cross, *Collaborative Learning Techniques*, ser. A Handbook for College Faculty, 2da edición, Jossey-Bass, 2014.

Aplicación para Dispositivos Móviles Android con Capacidad de Procesamiento de Consultas Realizadas sobre Documentos RDF Almacenados en la Nube

Jesús Soto, Amadís Martínez
jesusotox.35@outlook.com, aamartin@uc.edu.ve

Departamento de Computación, FaCyT, Universidad de Carabobo, Valencia, Venezuela

Resumen: Los dispositivos móviles se han convertido en parte de nuestra cotidianidad, permitiendo la realización de tareas habituales de manera sencilla y práctica. La plataforma Android ha crecido conjuntamente con los dispositivos móviles al estar presente en gran parte de los mismos. Por otro lado, la *Web Semántica* brinda la posibilidad de crear *software* capaz de procesar su contenido y realizar deducciones lógicas. Para ello, el W3C propone *Resource Description Framework* (RDF), un lenguaje para expresar metadatos, y SPARQL, un lenguaje de consultas sobre RDF. Los dispositivos móviles cuentan con capacidades limitadas de procesamiento, memoria y energía, dificultando el almacenamiento de un repositorio RDF local. Este trabajo propone el desarrollo de un mecanismo para Android, que permita consultar repositorios RDF en la nube, para superar la limitación del almacenamiento local del repositorio. Para esto, se estudiaron los componentes del *framework* de Android para la implementación del mecanismo que permitiera un uso compartido del mismo. Igualmente, se analizó el impacto del mecanismo sobre la tarea de responder consultas. Los componentes diseñados se implementaron en un prototipo completamente funcional para consultas SPARQL a repositorios en la nube desde un dispositivo con Android, al igual que una aplicación para la demostración y evaluación del rendimiento del mecanismo. Finalmente, se realizaron pruebas experimentales para evaluar la calidad del modelo planteado, utilizando datos sintetizados y reales. Los resultados obtenidos muestran que, en promedio, este enfoque es viable para la creación de aplicaciones Android basadas en la *Web Semántica* a través de consultas a repositorios RDF.

Palabras Clave: Android; *Web Semántica*; RDF; SPARQL; Aplicación; Servicio.

Abstract: Mobile devices have become part of our daily life, making fast and easy to perform common tasks. The Android platform has grown along with the mobile devices, being present in wide variety of them. Furthermore, the Semantic Web provides the ability to create software capable of processing its own content and make logical deductions. For this, the W3C proposed the Resource Description Framework (RDF), a language for expressing meta-data, and SPARQL, a query language for RDF. Mobile devices possess limited processing, memory and energy capabilities, making difficult to store a RDF repository locally. This paper proposes the development of a mechanism for Android, which allows to access RDF repositories in the cloud, to overcome the limitation of the local storage of the repository. For this, the Android framework components were studied in order to implement a mechanism that allows a shared usage of it. Similarly, the impact of the mechanism was analyzed on the task of query answering. The designed components were implemented in a fully functional prototype to make SPARQL queries to cloud-stored repositories from a Android device, as well as an application for demonstration and performance evaluation of the mechanism. Finally, several tests were performed to assess the quality of proposed model using synthesized and real data. The results show that, on average, this approach is feasible for building Semantic Web based Android applications through queries to RDF repositories.

Keywords: Android; Semantic Web; RDF; SPARQL; Application; Service.

I. INTRODUCCIÓN

Es muy notoria la relevancia que han adquirido los dispositivos móviles; en estos últimos años se han convertido aceleradamente en parte de la vida de cada uno de nosotros al ofrecer la posibilidad de ejecutar tareas, que solían ser algo complejas, ahora desde la palma de la mano. En particular, la plataforma

Android [1] es una de las más populares y reconocidas debido a que ha logrado abarcar una gran cantidad de dispositivos en el mercado. Además, por ser una plataforma de código abierto, ha podido ser adaptada y optimizada para su uso en dispositivos de distintos fabricantes.

Android es un sistema operativo basado en el *kernel* de Linux,

actualmente es desarrollado y mantenido por la compañía Google Inc. Android consta de una interfaz de usuario basada en la manipulación directa, por lo que está diseñado primordialmente para pantallas táctiles de teléfonos inteligentes y *tablets*.

Por otra parte, la *World Wide Web* (WWW) es un sistema que permite el acceso a una enorme colección de documentos enlazados, los cuales se encuentran dispersos en millones de computadores a lo largo del mundo conectados a través de *Internet*. El funcionamiento de la *Web* ha sido posible por medio de un conjunto bien establecido de estándares, los cuales garantizan varios niveles de interoperabilidad: la existencia de un protocolo de comunicaciones para el intercambio de datos (*Transmission Control Protocol / Internet Protocol* – TCP/IP) garantiza el nivel de interoperabilidad técnica; la definición de un protocolo para la recuperación de documentos (*HyperText Transfer Protocol* – HTTP) y la existencia de una estructura para la presentación de contenidos (*HyperText Markup Language* – HTML) garantizan el nivel de interoperabilidad sintáctica [2].

La presentación de contenidos en la *Web* fue concebida originalmente para su procesamiento directo por usuarios humanos. Esto significa que, aunque la información que reside en los documentos de la *Web* es legible por las máquinas, no es entendible por ellas. Los usuarios humanos pueden utilizar las máquinas para realizar búsqueda de información en los documentos de la *Web*. Sin embargo, el resultado de esta búsqueda debe ser interpretado por el usuario humano: las máquinas pueden presentar la información al usuario, pero no pueden inferir cuál es la información relevante o útil con respecto a la búsqueda realizada [2].

La *Web Semántica* (*Semantic Web*) es una propuesta basada en la idea de construir una infraestructura de semántica legible por las máquinas para la información en la *Web*, con la intención de superar las limitaciones explicadas anteriormente. Para que esto sea posible, la *Web Semántica* requiere que se establezca el nivel de interoperabilidad semántica, el cual, a su vez, requiere la definición de un conjunto de nuevos estándares, no sólo para la forma sintáctica de los documentos, sino también para la descripción de su contenido semántico. En este sentido, el *WWW Consortium* (W3C) ha liderizado la elaboración de propuestas que sirven como base para esta infraestructura: *Resource Description Framework* (RDF) junto con *RDF Schema* (RDFS) proporcionan los mecanismos fundamentales para la descripción de interrelaciones semánticas entre recursos en la *Web* [3].

Para la plataforma Android, existen dos sistemas de administración de datos RDF que permiten crear aplicaciones basadas en la *Web Semántica*: *AndroJena* [4] y *OpenRDF Sesame* [5]. Estos son sistemas de propósito general para la creación, almacenamiento y manipulación de documentos RDF. Ambos soportan consultas a través del lenguaje SPARQL, el cual permite hacer búsquedas sobre los recursos de la *Web Semántica* utilizando distintas fuentes de datos.

Sin embargo, hablar de dispositivos móviles es sinónimo de

recursos limitados, poder de procesamiento restringido, escasa memoria RAM, poca capacidad de almacenamiento secundario y limitada cantidad de energía para operar. Por lo tanto, lo que *AndroJena* y *Sesame* tienen en común, es que operan siempre y cuando el repositorio de datos esté almacenado en el mismo dispositivo, es decir, localmente. Dadas las pocas capacidades de los dispositivos móviles y los grandes tamaños que pueden llegar a tener los repositorios, es difícil pensar que se pueda tener alguno almacenado dentro de un dispositivo móvil, consumiendo toda o gran parte de la capacidad de almacenamiento, esto reduciría drásticamente la eficiencia del dispositivo.

En virtud de lo antes expuesto, el objetivo principal de esta investigación consiste en el desarrollo y evaluación de una aplicación para dispositivos móviles con Android que utilice un servicio Android compartido capaz de realizar consultas, utilizando el lenguaje SPARQL, a un repositorio RDF que no esté almacenado localmente en el dispositivo. En otras palabras, el repositorio estará almacenado en la nube, de modo que se obtengan sólo los resultados de interés sin necesidad de desperdiciar espacio en un repositorio completo, preservando mejor el rendimiento y la capacidad de almacenamiento del dispositivo móvil.

Los aportes de este trabajo se resumen a continuación:

- Un servicio Android compartido que permite a cualquier aplicación del dispositivo hacer uso de él para consultar, a través del lenguaje SPARQL, a repositorios RDF almacenados en la nube y utilizando HTTP como protocolo de comunicación, solventando así, la limitación del almacenamiento local del repositorio en el dispositivo móvil.
- Una aplicación Android que hace uso del servicio Android desarrollado para consultar un *endpoint* SPARQL, obtener los resultados mediante su descarga y posteriormente procesarlos.
- Una extensa fase de evaluación compuesta por un estudio experimental sobre la tarea de evaluar el rendimiento del mecanismo desarrollado al realizar consultas en SPARQL desde un dispositivo móvil a un *endpoint* SPARQL.

Este documento fue estructurado en cinco secciones, incluyendo la introducción. En la Sección II se presenta el marco teórico, los antecedentes y los trabajos relacionados con esta investigación. La Sección III describe el diseño del mecanismo planteado para obtener solución al problema de consultar repositorios RDF desde dispositivos móviles con Android. La Sección IV contiene los resultados experimentales obtenidos por el esquema desarrollado, presentando el estudio experimental de su rendimiento; además, se compara el rendimiento entre los diferentes formatos de respuesta a las consultas. Finalmente, en la Sección V se presentan las conclusiones de la investigación, las recomendaciones finales y los trabajos futuros.

II. MARCO TEÓRICO

A. Trabajos Relacionados

Resource Description Framework (RDF) es una propuesta del W3C para expresar metadatos acerca de información

de recursos que se encuentran en la *Web* direccionados por URIs. El objetivo de RDF consiste en describir recursos de información de una manera comprensible por máquinas, y que estas descripciones puedan ser procesadas por aplicaciones. En el modelo de datos de RDF, la estructura básica es la terna RDF (RDF *triple*), la cual es de la forma (s, p, o) , donde s (sujeto) es el recurso que se está describiendo, p (predicado) es una propiedad del recurso s y o (objeto) es el valor de la propiedad p . Un conjunto de ternas RDF se denomina grafo RDF (RDF *graph*) y un conjunto de uno o más grafos RDF se denomina fuente de datos RDF (RDF *dataset*).

Un sistema de administración de datos RDF es un *software* de propósito general que permite la creación, almacenamiento persistente y manipulación de fuentes de datos RDF. Además, debe ofrecer soporte para dos tareas fundamentales: (1) responder consultas realizadas por usuarios y agentes de *software* sobre fuentes de datos RDF y (2) razonamiento semántico sobre fuentes de datos RDF, para descubrir interrelaciones entre los recursos descritos [2].

A medida que las fuentes de datos RDF han crecido en magnitud, así como la repercusión que han tenido los dispositivos móviles en la sociedad, se han incrementado los esfuerzos para buscar soluciones viables que permitan mejorar el desempeño y escalabilidad con respecto al almacenamiento y consulta de estas fuentes de datos RDF desde estos dispositivos. En el contexto de Android y la *Web Semántica* se han desarrollado dos sistemas de administración de datos RDF, con el objetivo de ofrecer acceso eficiente a fuentes de datos RDF:

1) *AndroJena*: AndroJena [4] es un sistema de administración de RDF de código abierto (*open source*), implementado en JAVA y es el *porting* de Jena para la plataforma Android de Google. Al ser un *porting* de Jena, AndroJena ofrece las mismas características adaptadas a la plataforma Android, como una interfaz de programación (API) para la manipulación de grafos RDF como conjuntos de ternas RDF. AndroJena fue extendido con ARQoid [6] y TDBoid para así, soportar de la misma manera consultas SPARQL [7] y almacenamiento persistente en dispositivos móviles con Android.

2) *OpenRDF Sesame*: Sesame [8] es un sistema de administración de RDF de código abierto (*open source*), implementado en JAVA y disponible para descarga [5]. Sesame cuenta con la cualidad de que puede ser ejecutado en dispositivos móviles con Android. Para la manipulación y consulta de grafos RDF, Sesame ofrece una interfaz de programación (API) llamada SAIL. A través de esta API, permite almacenar los grafos RDF en memoria principal, en memoria secundaria o en un sistema manejador de bases de datos relacionales, utilizando una técnica basada en el modelo relacional tradicional denominada tabla de ternas (*triple table*).

Una vez almacenadas las ternas de un grafo RDF, Sesame ofrece tres operaciones básicas sobre la base de datos asociada con el grafo RDF original: (1) agregar (*add*), que inserta una terna RDF en la base de datos, (2) borrar (*remove*), que elimina una terna RDF del grafo y (3) obtener (*get*), que retorna todas las ternas RDF que coinciden con un patrón de acceso de la forma (s, p, o) , donde s , p y o pueden ser

constantes o variables. La versión más reciente ofrece soporte para consultas SPARQL.

Pese a que ambos sistemas de administración RDF (AndroJena [4] y *Sesame* [5]) están adaptados para su ejecución en la plataforma Android, permitiendo así almacenar un repositorio RDF en el dispositivo móvil y realizar consultas al mismo, tienen la desventaja que ambos asumen y requieren que el almacenamiento del repositorio sea local, y debido al gran tamaño que estos pueden ocupar, podrían consumir la totalidad del almacenamiento del dispositivo o incluso no ser posible su almacenamiento si el dispositivo es muy restringido.

B. Bases Teóricas

1) *SPARQL*: Es un lenguaje de consultas sobre RDF estandarizado por el W3C que se utiliza para expresar consultas que permiten interrogar diversas fuentes de datos. SPARQL permite a sus usuarios escribir consultas sobre datos que siguen las especificaciones RDF. SPARQL provee un conjunto completo de operaciones de consultas analíticas como lo son JOIN, SORT, AGGREGATE para datos en los que su esquema forma parte de los mismos datos en lugar de requerir una especificación externa del esquema. La información del esquema (ontología) por lo general es provista de manera externa, para permitir que distintas fuentes de datos puedan ser combinadas sin ambigüedades.

2) *Endpoint SPARQL*: Un *endpoint* SPARQL es un servicio *Web* que permite a los usuarios consultar fuentes de datos RDF haciendo uso del lenguaje SPARQL. Los resultados son normalmente retornados en uno o varios formatos procesables. De tal manera que un *endpoint* SPARQL es concebido como una interfaz hacia una fuente de datos RDF.

3) *Jena*: Jena [9][10] es un sistema de administración de RDF de código abierto (*open source*), implementado en JAVA y disponible para descarga [10]. Jena ofrece una interfaz de programación (API) para la manipulación de grafos RDF como conjuntos de ternas RDF. A través de esta API, Jena permite almacenar los grafos RDF en memoria principal o en memoria secundaria, utilizando dos técnicas basadas en el modelo relacional tradicional: (1) Tabla de ternas, y (2) Tabla de propiedades.

Jena fue extendido con dos componentes: (1) ARQ [11], una máquina de ejecución para consultas SPARQL que incluye optimización basada en costos y (2) *Tuple DataBase* (TDB) [12], un sub-sistema para almacenamiento persistente de grafos RDF que ofrece integración con ARQ para procesar consultas SPARQL.

4) *Spring Boot* y *Spring Web MVC*: El *framework* Spring [13][14] es una plataforma JAVA que ofrece soporte de infraestructura para el desarrollo de aplicaciones JAVA. Spring tiene como objetivo hacer que el desarrollo de aplicaciones JAVA más sencillo y promover las buenas prácticas de programación, permitiendo una programación basada en objetos planos de JAVA (*Plain Old JAVA Objects* – POJO).

Spring está compuesto por diversos módulos que ayudan a tareas específicas, uno de ellos es *Spring Web MVC*, que contiene componentes para construcción de aplicaciones *Web*

Spring autónomas con configuración automática y servicios *Web REST*.

Por otra parte, Spring Boot es la solución del *framework* Spring para crear aplicaciones con la mínima configuración inicial posible, debido a que posee librerías (tanto propias como de terceros), para poder iniciar la ejecución de la aplicación con el mínimo esfuerzo.

C. Servicios Android

Un servicio (*service*) [15] en Android es un componente de aplicación capaz de realizar operaciones de larga duración en segundo plano (*background*) y no provee una interfaz gráfica de usuario. Algún otro componente de la aplicación podría iniciar un servicio y éste continuará ejecutándose en segundo plano, incluso si el usuario cambia a otra aplicación.

Adicionalmente, un componente puede enlazarse (*bind*) a un servicio para interactuar con él e incluso desempeñar comunicación entre procesos. Existen esencialmente dos tipos de servicios en Android:

- **INICIADO (*started*):** Un servicio es “iniciado” cuando algún componente (como por ejemplo un *Activity*) lo inicia a través de la llamada al método `startService()`. Una vez iniciado, un servicio puede ejecutarse en segundo plano de manera indefinida, incluso si el componente que lo inició es destruido por el sistema. Por lo general, un servicio iniciado ejecuta operaciones concretas y no retorna resultado alguno al componente que lo inició.
- **ENLAZADO (*bound*):** Un servicio es “enlazado” cuando un componente se enlaza a él a través de la llamada al método `bindService()`. Un servicio enlazado ofrece una interfaz cliente-servidor (*client-server*) que permite a otros componentes interactuar con él, enviar peticiones (*requests*), obtener resultados e incluso desempeñar comunicación entre procesos. Un servicio enlazado se ejecuta sólo mientras exista algún otro componente enlazado a él. Múltiples componentes pueden enlazarse a un mismo servicio en paralelo, pero una vez que todos ellos se desenlacen, el servicio es destruido por el sistema.

Aunque existen estos dos tipos de servicios con características bien definidas, cabe la posibilidad de crear un servicio que funcione combinando características de los dos tipos, es decir, un servicio puede ser iniciado (*started*) para ejecutarse indefinidamente y al mismo tiempo permitir que otros componentes se enlacen (*bind*) a él.

Sin importar si un servicio es iniciado, enlazado, o ambos, por defecto los servicios en Android son privados de la aplicación que los crea, es decir, sólo los componentes de esa misma aplicación pueden hacer uso del servicio, y además son creados y ejecutados en el mismo proceso de la aplicación a la que pertenecen. Sin embargo, un servicio puede programarse de manera que pueda ser utilizado por componentes de cualquier aplicación del dispositivo móvil.

III. DISEÑO DE LA SOLUCIÓN

La Figura 1 presenta un diagrama en el que se destacan los módulos principales que componen la solución del problema

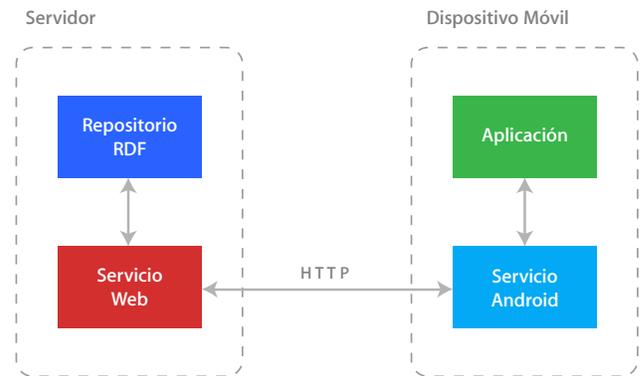


Figura 1: Componentes que Integran la Solución

planteado.

En el diagrama podemos notar que la arquitectura propuesta está compuesta por cuatro componentes principales, que cumplen un papel muy específico de la solución. En primer lugar, el *Repositorio RDF* es el encargado de almacenar y gestionar los documentos RDF. El *Servicio Web* ofrece la posibilidad de poder obtener los datos del repositorio RDF para exponerlos a través de un contenedor *Web*, que es donde reside el servicio. El *Servicio Android* es un componente en el dispositivo móvil que permite establecer la conexión con el servidor *Web* para enviarle la información necesaria, de modo que éste es capaz de responder las consultas, y posteriormente devolver los resultados al dispositivo móvil. Por último la *Aplicación Android* es quien hace uso del *Servicio Android* para poder consultar y obtener resultados del *Repositorio RDF* que toma lugar en la nube.

En las subsecciones siguientes se explican con detalle cada uno de los componentes que integran este mecanismo para el procesamiento y consulta de fuentes de datos RDF desde dispositivos móviles con Android.

A. Repositorio de Datos RDF

En esta subsección se presenta el *framework* RDF utilizado para el almacenamiento y procesamiento eficiente de documentos RDF.

Jena [10][9] es un sistema de administración de RDF de código abierto (*open source*), implementado en JAVA. En el ámbito de la solución planteada, lo primero se debe destacar es el hecho de que Jena esté implementado en JAVA, permitiendo una mejor interacción y una integración más directa entre el *framework* y el servicio *Web*. A pesar de que el lenguaje nativo de consultas de Jena es RDQL, se incluyó soporte para SPARQL a través de ARQ haciendo uso de heurísticas de optimización para lograr un procesamiento eficiente de consultas, y dado que las consultas a enviar desde el dispositivo móvil estarán en SPARQL, es primordial que el *framework* utilizado tenga capacidad de procesar las consultas en ese lenguaje.

Adicionalmente, Jena ofrece un eficiente mecanismo para el almacenamiento en memoria secundaria del repositorio RDF haciendo uso de TDB, este método de almacenamiento es el

ideal cuando se trata con fuentes de datos de gran tamaño. Además, al estar integrado con ARQ, soporta consultas en SPARQL directamente sobre el repositorio en memoria secundaria. Finalmente, Jena goza de una gran popularidad y documentación que permiten su fácil implantación, además de ser uno de los *frameworks* más eficientes y rápidos en responder consultas SPARQL.

B. Servicio Web

En esta subsección se presenta el *framework* utilizado para desplegar el servicio *Web* que forma parte del esquema de la solución, permitiendo la creación del *endpoint* SPARQL.

Spring Boot [16][17] es un *framework* de aplicación de código abierto (*open source*) y un contenedor de inversión de control (*inversion of control*) para el desarrollo de aplicaciones en la plataforma JAVA. Spring Boot es la solución del *framework* Spring [13][14] para crear aplicaciones con la mínima configuración inicial posible debido a que posee librerías, tanto propias como de terceros, para poder iniciar la ejecución de la aplicación con el mínimo esfuerzo usando la técnica de convención sobre configuración (*convention over configuration*).

Spring Boot provee la capacidad de desplegar (*deploy*) un servicio *Web* haciendo uso del *framework* Spring Web MVC [18], este ofrece una arquitectura Modelo-Vista-Controlador (*Model-View-Controller* – MVC) y componentes preparados para el desarrollo de aplicaciones *Web* flexibles. Las siguientes son algunas de las características más destacadas de Spring *Web* MVC que tienen lugar como parte del esquema de la solución planteada:

- Creación de aplicaciones Spring autónomas con configuración automática y servicios *Web* con API REST.
- Permite a los programadores desarrollar todo tipo de componentes de aplicación usando planos objetos de JAVA (*Plain Old Java Objects* – POJO).
- Provee herramientas diseñadas para la fase de producción como métricas, verificación de estado (*health check*) y configuración externalizada.

Este conjunto de características ayudan a identificar los motivos del uso de Spring Boot como *framework* para el servicio *Web* que forma parte del esquema de la solución. Spring Boot al ser un *framework* para aplicaciones de JAVA, permite una fácil, rápida y eficiente integración e interacción con Jena, por lo que Jena es utilizado como un conjunto de librerías añadidas a Spring Boot para permitir la gestión, almacenamiento y procesamiento de consultas con datos RDF.

De esta manera, gracias a que ambos *frameworks* comparten el lenguaje de programación JAVA, es posible realizar la integración de forma tal que el servicio *Web* pueda exponer el repositorio RDF a través de una API REST bien definida para el procesamiento de consultas en SPARQL. En el servicio *Web* que se ha implementado, el procesamiento de consultas SPARQL se logra una vez que el *DispatcherServlet* recibe la solicitud HTTP, extrae los datos, y los proporciona a Jena para la ejecución de la consulta en el repositorio RDF.

Una vez que el repositorio RDF (gestionado por Jena) ejecuta la consulta en el repositorio y obtiene los resultados, éstos se almacenan como un archivo dentro del servidor y se retorna al controlador la dirección del archivo recién creado, para ser devuelto en el cuerpo del mensaje de respuesta HTTP y así el archivo pueda ser luego descargado.

El formato del resultado de la consulta dependerá de lo especificado en la solicitud HTTP, actualmente se tiene soporte para RDF/XML [19], RDF/JSON [20] y texto plano. Por supuesto, el tamaño del resultado puede variar según el formato en que se exprese, pero eso no afecta el tiempo del procesamiento de la consulta.

C. Servicio Android

En esta subsección se presenta el mecanismo utilizado dentro del dispositivo móvil que permite la comunicación con el servidor *Web*, previamente descrito, para solicitar resultados de consultas escritas en lenguaje SPARQL desde el dispositivo y su posterior entrega a la aplicación que lo solicita.

Recordando un poco el marco teórico, un servicio (*service*) en Android es un componente de aplicación capaz de realizar operaciones de larga duración en segundo plano (*background*) y no provee una interfaz gráfica de usuario. Algún otro componente de la aplicación podría iniciar un servicio y éste continuará ejecutándose en segundo plano incluso si el usuario cambia a otra aplicación. Adicionalmente, un componente puede enlazarse (*bind*) a un servicio para interactuar con él e incluso desempeñar comunicación entre procesos (*Inter-Process Communication*) [15].

En líneas generales, un servicio en sí es bastante simple, ofreciendo dos funcionalidades principales:

- Facilidad para las aplicaciones de “informar” al sistema acerca de algún trabajo que se quiera ejecutar en segundo plano (incluso cuando el usuario no está directamente interactuando con la aplicación). Esto corresponde con los servicios “iniciados” (*started*) los cuales permiten programar alguna operación para ser ejecutada hasta que el mismo servicio o algún otro componente detenga su ejecución explícitamente.
- Facilidad para las aplicaciones de “exponer” algunas de sus funcionalidades a otras aplicaciones. Esto corresponde a los servicios “enlazados” (*bound*) los cuales permiten mantener una conexión consistente con algún componente a fin de interactuar con él.

Ahora bien, en el ámbito del esquema de la solución planteada, la principal tarea a cumplir por este componente es proporcionar una interfaz común en el sistema Android, que pueda ser utilizada por cualquier otro componente, para proveer la facilidad de realizar consultas y obtener sus resultados, usando el lenguaje SPARQL, sobre algún repositorio RDF que viva en la nube y no localmente.

Para lograr esto, el servicio desarrollado, que lleva por nombre *SparkleService*, combina características de los diferentes tipos de servicios. Es “enlazado” para poder exponer su funcionalidad a otros componentes de la aplicación y además proveer

una interfaz cliente-servidor que permita una comunicación consistente y directa entre el componente y SparkleService, de modo que la aplicación externa pueda enviar peticiones y obtener resultados de SparkleService. Y es “iniciado” para poder ejecutar las descargas de los resultados en segundo plano y de manera indefinida hasta que finalice y se detenga su ejecución.

Además, SparkleService es un servicio exportado, esto proporciona la capacidad de exponer su funcionalidad a todas las aplicaciones y componentes en el dispositivo móvil, sin importar si esos componentes pertenecen a la misma aplicación de la que forma parte el servicio.

SparkleService es el único componente de la aplicación a la que pertenece, de modo que al instalar dicha aplicación, sólo se estará instalando el servicio y nada más. La razón para esto es permitir que el servicio pueda ser instalado y gestionado de manera independiente a otras aplicaciones del dispositivo. Adicionalmente, al ser SparkleService el único componente, permite que éste se ejecute en un proceso diferente al de las aplicaciones externas que puedan hacer uso de él, esto tiene la ventaja de que no se vean afectadas mutuamente en caso de que alguno de los dos, por una u otra razón, sean destruidos por el sistema.

Debido a que SparkleService y la aplicación que desee usarlo se encuentran en procesos separados, es necesario utilizar mecanismos de comunicación entre procesos para el paso de mensajes entre las partes. En Android existen dos mecanismos de comunicación entre procesos: (1) A través del lenguaje AIDL (*Android Interface Definition Language*); y (2) a través de la clase `Messenger`.

Para SparkleService se utilizó la clase `Messenger` para la comunicación entre procesos, debido a su facilidad de implementación y su uso intuitivo. Este mecanismo permite definir, mediante el uso de constantes, los posibles mensajes o solicitudes que puede recibir SparkleService, al igual que los mensajes de respuesta a las aplicaciones que hagan uso de él.

Independientemente del mecanismo escogido para la comunicación entre procesos, el sistema de Android permite el paso de mensajes entre procesos con un tamaño máximo de 1 MB, razón por la cual SparkleService ofrece la posibilidad de realizar la descarga de una consulta como un archivo y almacenarlo en la memoria del dispositivo (a la que ambas partes pueden acceder) para luego notificar a la aplicación solicitante la ruta completa en donde se almacenó el archivo para que dicha aplicación pueda leerlo y procesarlo.

Para la comunicación con el servicio *Web*, SparkleService utiliza mecanismos convencionales para manejar conexiones HTTP, y de acuerdo a la solicitud que reciba el servicio, se construye la solicitud HTTP adecuada para dar respuesta a la solicitud recibida. Los componentes que deseen hacer uso de SparkleService pueden especificar el URI del *endpoint* que gestionará las consultas, sin embargo tiene una API definida para la comunicación con el *endpoint* que debe ser respetada por el mismo para la correcta comunicación entre las partes.

En este caso, el servicio *Web* implementado (especificado en la sección anterior) implementa correctamente la API para una comunicación exitosa.

SparkleService tiene la capacidad para comunicar al cliente cualquier evento que suceda durante su ejecución, no únicamente comunicarle el resultado de una consulta, sino además si ocurre algún error en la comunicación con el servidor o en la consulta, le será notificado al cliente respectivo de la misma manera.

D. Aplicación Android

En esta subsección se describe la aplicación Android desarrollada que hace uso de SparkleService para realizar y recibir resultados de consultas en SPARQL al repositorio RDF almacenado en la nube y expuesto mediante el servicio *Web* desplegado, utilizando una API REST.

La aplicación Android desarrollada lleva por nombre *Sparkle*, y es una aplicación de ejemplo para demostrar cómo una aplicación externa puede hacer uso de SparkleService. Es bueno reiterar que así como la aplicación *Sparkle*, cualquier otra aplicación puede hacer uso de SparkleService, y *Sparkle* es la demostración de cómo pueden hacerlo.

El objetivo de *Sparkle* es demostrar cómo realizar consultas en SPARQL a repositorios RDF en la nube mediante SparkleService, de modo que la interfaz gráfica de la aplicación es bastante sencilla y enfocada a lo que se quiere mostrar. La interfaz gráfica consiste de un campo de texto en donde se introduce el URI del *endpoint* en donde se encuentra el repositorio RDF, un selector para seleccionar el formato deseado del resultado de la consulta, una área de texto que será en donde se ingresará el texto que corresponde a la consulta en lenguaje SPARQL, un *checkbox* para especificar si se quiere descargar la consulta en un archivo o no, y por último un botón para enviar la solicitud de consulta a SparkleService.

Del lado de *Sparkle* se implementa de igual manera un `Messenger` (llamémoslo *A*) que le será enviado a SparkleService para que éste sea capaz de saber el cliente al que debe responder, asimismo, el `Messenger A` es quien recibirá y manejará los mensajes enviados desde SparkleService.

Cuando *Sparkle* se enlaza (*bind*) con SparkleService mediante la llamada al método `bindService()`, el servicio responde enviando un *Binder* a *Sparkle*, que contrasta con el `Messenger` que existe dentro de SparkleService, de manera que el *Binder* es necesario para crear un segundo `Messenger` (llamémoslo *B*) en *Sparkle*, encargado de enviar las solicitudes de consulta a SparkleService.

Al pulsar el botón para ejecutar una consulta, *Sparkle* verifica que los campos estén correctos y de ser así se toma y se empaqueta toda la información en un mensaje que luego le es enviado a SparkleService pidiendo resolver la consulta y obtener el resultado mediante la descarga de un archivo con el resultado.

En cualquiera de los dos casos, cuando *Sparkle* es notificado sobre el resultado de la consulta, se abre otra vista que muestra

los resultados en un gran campo de texto, u opcionalmente en una lista de elementos utilizando un *parser* para tomar de forma individual los elementos del resultado (dependiendo del formato en que esté expresado el resultado) y listarlos.

E. Protocolo de Transmisión de Datos

Como se ha expresado múltiples veces, el esquema de comunicación para que un dispositivo móvil pueda tener acceso a la nube, será a través del uso de HTTP. Intuitivamente lo que se puede pensar de HTTP es básicamente un diálogo o lenguaje del cual un dispositivo móvil puede hacer uso para entablar una comunicación con la nube, es un protocolo de aplicación para comunicación entre dispositivos en la red.

HTTP es un protocolo a nivel de aplicación para sistemas de información hipermedia. HTTP ha sido utilizado por la *World Wide Web* desde 1990. Es un protocolo genérico que puede ser utilizado para una gran variedad de tareas más de allá de su uso para hipertexto.

HTTP es un protocolo cliente-servidor, esto quiere decir que las interacciones que son ejecutadas a través de HTTP, son iniciadas siempre por el cliente que quiere acceder o solicitar algún recurso del servidor. De modo que el servidor posee una serie de recursos que puede proveerle al cliente, y el cliente envía solicitudes al servidor solicitando alguno de esos recursos, dicho servidor responde al cliente ya sea con el recurso que solicitó o algún mensaje indicando si pudo o no procesar la solicitud [21].

Pero, ¿por qué utilizar un protocolo de navegador *Web* para comunicarse con la nube? HTTP provee ciertas propiedades, una de ellas es que obtenemos una interfaz uniforme que podemos proveer vía HTTP a servicios o recursos que se encuentran en un servidor. Por ejemplo, si el cliente es un navegador *Web* y envía una solicitud HTTP que el servidor pueda procesar, dicha solicitud es exactamente la misma que si la hubiese enviado un dispositivo móvil.

Además, existe una inversión muy significativa, tanto en los *frameworks* como en librerías, balanceadores de carga (*load balancers*) y cualquier otra infraestructura desarrollada para HTTP, que hacen natural en uso de HTTP como mecanismo de comunicación con la nube.

En el ámbito de SparkleService y el servicio *Web* implementado, se utilizan mensajes de solicitud HTTP para enviar toda la información necesaria al servicio *Web* de modo que éste sea capaz de procesar la consulta en SPARQL solicitada. En esa solicitud se utiliza el método POST de HTTP pues éste permite enviar una cantidad de datos indefinida en el cuerpo (*body*) de la solicitud HTTP, es de hecho en el cuerpo donde se envía el *string* que representa la consulta en lenguaje SPARQL.

De manera análoga, también se utilizan mensajes de respuesta HTTP para recibir la información necesaria para que el cliente sea capaz de descargar el archivo con el resultado de la consulta en el formato especificado en la solicitud. La información recibida en este mensaje de respuesta HTTP es un *string* en el cuerpo de la respuesta en formato JSON

representando un objeto con la información necesaria para el análisis y la descarga del archivo que guarda el resultado de la consulta; este objeto JSON incluye hora de respuesta (*timestamp*), nombre generado del archivo (*filename*), URL del archivo y tamaño del archivo (*size*).

IV. RESULTADOS EXPERIMENTALES

A. Configuración de los Experimentos

1) *Plataforma Computacional*: Para realizar los experimentos se requirieron de tres equipos: (1) un computador que juega el papel del servidor (la nube) que contiene el repositorio RDF y lo expone a través del servicio *Web*, (2) un dispositivo móvil con Android que juega el papel del cliente donde reside el servicio y la aplicación Android que se comunica con el servidor que contiene el repositorio RDF para realizar consultas y, (3) un enrutador (*router*) al cual están conectados de manera inalámbrica el servidor y el dispositivo móvil para así permitir la comunicación entre ambos mediante una red inalámbrica de área local (WLAN) utilizando el protocolo HTTP. Además de los equipos, fue necesario tener instalado JAVA para el desarrollo y ejecución de los experimentos. A continuación se describe las características de cada componente:

- JAVA(TM) SE Runtime Environment (build 1.7.0_71-b14).
- El servidor es un computador con procesador *Intel Core i7-3820* de 4 núcleos a 3.6 GHz y 16 GB de memoria RAM a 2133 MHz bajo el sistema operativo Windows 8.1 de 64 bits.
- El dispositivo móvil es un teléfono LG Nexus 5 con procesador *Qualcomm Snapdragon 800 Quad-core* a 2.3 GHz y 2 GB de memoria RAM bajo el sistema operativo Android 5.1 Lollipop.
- El *router* es un TP-LINK TL-WR941ND de 300 Mbps a 2.4 GHz.

2) *Casos de Prueba*: Los experimentos se realizaron sobre distintas fuentes de datos, tanto sintetizadas como reales. Los datos sintetizados utilizados en este estudio experimental fueron generados por el Lehigh University *Benchmark* (LUBM) [22]. Este *benchmark* está basado en la ontología Univ-Bench, la cual describe universidades, departamentos y las actividades académicas que ocurren en ellos. Cada fuente de datos está representada por la expresión $Univ(N, S)$, la cual denota la fuente de datos que contiene N universidades generadas a partir de la semilla S , comenzando por el índice 0. Para efectos de este estudio experimental, se generaron cinco fuentes de datos de prueba: $Univ(1, 0)$, $Univ(5, 0)$, $Univ(10, 0)$, $Univ(20, 0)$ y $Univ(50, 0)$, las cuales contienen datos de 1, 5, 10, 20 y 50 universidades, respectivamente. Las características de estas fuentes de datos (número de ternas RDF que contienen y tamaño en disco) se muestran en la Tabla I.

Para los datos reales se utilizaron dos fuentes de datos de DBpedia [23] el cual provee datos extraídos del contenido de Wikipedia distribuidos en distintas fuentes de datos. La primera fuente de datos contiene información de geolocalización en latitud y longitud de todos los lugares alrededor

Tabla I: Características de las Fuentes de Datos Sintetizadas

Fuente de datos	Número de ternas RDF	Tamaño (MB)
<i>Univ</i> (1, 0)	103 074	17,27
<i>Univ</i> (5, 0)	645 649	108,28
<i>Univ</i> (10, 0)	1 316 322	220,79
<i>Univ</i> (20, 0)	2 781 322	468,68
<i>Univ</i> (50, 0)	6 888 642	1 163,49

Tabla II: Características de las Fuentes de Datos Reales

Fuente de datos	Número de ternas RDF	Tamaño (MB)
<i>Dataset #1</i>	2 105 024	297,475
<i>Dataset #2</i>	15 849 011	2 115,141

del planeta que están indizados en Wikipedia, esta fuente será referida como *Dataset #1*. La segunda fuente de datos es una combinación entre los datos de geolocalización de la primera fuente, e información muy específica y detallada de propiedades para cada uno de los lugares indizados en Wikipedia, esta fuente será referida como *Dataset #2*. Las características de estas fuentes de datos (número de ternas RDF que contienen y tamaño en disco) se muestran en la Tabla II.

En esta parte del estudio experimental se consideraron dos conjuntos de consultas a evaluar:

- *Benchmark 1*: El primer banco de pruebas pertenece a LUBM. LUBM ofrece 13 consultas de prueba, las cuales toman en consideración los siguientes factores: tamaño de la entrada, selectividad, complejidad, utilización de información jerárquica y capacidad de inferencia lógica. Las consultas de prueba fueron diseñadas en términos de los criterios antes mencionados con énfasis en tamaño grande de la entrada y alta selectividad. En este estudio experimental se modificaron las consultas que involucran la tarea de razonamiento semántico de modo que no sea necesario aplicarlo para darles respuesta.
- *Benchmark 2*: El segundo banco de pruebas pertenece a las fuentes de datos reales de DBpedia, y consiste en dos conjuntos de cinco consultas conjuntas, el primer conjunto para el *Dataset #1* y el segundo conjunto para el *Dataset #2*; las consultas fueron diseñadas respetando los criterios de diseño descritos en el *benchmark 1*, así que tampoco involucran la tarea de razonamiento semántico para ser respondida.

3) Métricas de Evaluación:

- *Tiempo de respuesta de las consultas*: tiempo de respuesta (medido en segundos) de las consultas de prueba, cada una de las cuales fue ejecutada cinco veces sobre las distintas fuentes de datos y, posteriormente, promediados los resultados de las ejecuciones. La medición de este valor se realizó a nivel de código, tomando el instante de tiempo previo a la ejecución de la consulta, y el instante de tiempo al finalizar la consulta para luego calcular su diferencia.
- *Tiempo de almacenamiento de los resultados*: tiempo promedio que toma procesar la consulta para darle formato y almacenarla en un archivo local en el servidor para su posterior transmisión al cliente (medido en segun-

dos), se tomarán en cuenta los formatos RDF/XML [19], RDF/JSON [20] y texto plano. Para la medición de este valor se tomaron los tiempos sin el almacenamiento y los tiempos con el almacenamiento para calcular la diferencia entre ellos promediada a cinco veces.

- *Tamaño en disco del archivo con los resultados*: tamaño de archivo con los resultados de cada consulta (medido en MB). Para obtener este valor se consultó directamente el tamaño del archivo resultante en el explorador de archivos del sistema operativo.
- *Tiempo de transmisión de los resultados al dispositivo móvil*: tiempo promedio que toma la transferencia del archivo con los resultados de la consulta del servidor al dispositivo móvil (medido en segundos). Para la medición de este valor se toma el tiempo en que el dispositivo móvil pide al servidor la descarga del archivo y el tiempo en que finaliza la descarga en el dispositivo móvil para luego calcular su diferencia, esto se repite cinco veces para así obtener el promedio.
- *Tiempo total de la operación de consulta desde el dispositivo móvil*: tiempo promedio (medido en segundos) que toma la totalidad de la operación de consulta desde el dispositivo móvil, esto es, desde que el dispositivo solicita al servidor responder una consulta, hasta que los resultados terminan de descargarse. Por consiguiente, este tiempo incluye todas las métricas de tiempo citadas en esta sección. Para la medición de este valor se toma el instante de tiempo en que el dispositivo realiza la solicitud, y el instante de tiempo en que finaliza la descarga de los resultados para luego calcular su diferencia, este proceso se realizó cinco veces por cada consulta para calcular el promedio.

B. Análisis de los Resultados

1) Hipótesis Iniciales:

- Dado un formato solicitado por el dispositivo móvil para la representación del resultado de la consulta, este resultado tomará un tiempo diferente de almacenamiento en el archivo del lado del servidor y, posteriormente, el tiempo de transferencia al dispositivo móvil. Esto se debe a que cada formato posee un algoritmo de procesamiento distinto y cada representación tiene una estructura diferente, lo que resulta en que un mismo archivo resultante pueda tener diferentes tamaños en disco dependiendo del formato.
- La velocidad de transferencia (o velocidad de descarga) del archivo resultante de una consulta al dispositivo móvil tiende a depender del ancho de banda disponible en la red local. Además, esta velocidad no necesariamente se mantiene en el mismo valor durante el tiempo de descarga. Sabiendo esto, es posible que un mismo archivo tome distintos tiempos de descarga.
- Debe existir una diferencia en el tiempo de respuesta de las consultas evaluadas en *cold cache* y en *warm cache*: el tiempo de evaluación de consultas en *warm cache* debe ser menor que en *cold cache* en el sistema de administración de datos RDF considerado (Jena), el cual debe estar en

Tabla III: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre *Univ(1, 0)* de LUBM

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	1,009	0,702	0,990	0,625	0,978	0,590
Q02	1,095	0,651	1,078	0,622	1,008	0,578
Q03	1,066	0,613	1,007	0,582	0,974	0,567
Q04	1,134	0,631	1,071	0,616	1,053	0,570
Q05	1,226	0,965	1,205	0,780	1,148	0,663
Q06	2,088	1,452	2,057	1,417	1,926	1,201
Q07	1,144	0,708	1,098	0,643	1,037	0,624
Q08	3,656	2,480	3,190	2,100	2,613	1,544
Q09	1,351	0,781	1,281	0,687	1,166	0,625
Q10	1,077	0,751	1,072	0,689	1,055	0,593
Q11	1,144	0,737	1,033	0,682	1,127	0,650
Q12	1,034	0,649	1,006	0,631	0,968	0,623
Q13	1,619	0,887	1,502	0,875	1,441	0,787
Promedio	1,434	0,924	1,353	0,842	1,269	0,740

capacidad de mejorar su rendimiento si existen datos pre-cargados en la memoria caché.

- Debe existir una relación entre la cardinalidad del resultado de una consulta y el tamaño del archivo que almacena ese resultado, entendiendo que, a mayor cardinalidad, mayor será el tamaño del archivo dado que se deben almacenar más datos.

2) *Resultados Obtenidos:* Solamente se mostrarán los resultados correspondientes a la quinta métrica de evaluación (tiempo total de la operación de consulta desde el dispositivo móvil) la cual, a su vez, incluye los resultados parciales de las otras métricas de tiempo ya descritas.

En este apartado se estudia el comportamiento y la eficiencia del esquema implementado sobre la tarea de enviar una solicitud y obtener los resultados mediante su descarga. Para esto, se analizan los resultados experimentales obtenidos con respecto al tiempo promedio (medido en segundos) del total de la operación de consulta, desde que el dispositivo móvil realiza la solicitud al servidor, hasta que se termina de descargar el archivo con los resultados. Cada consulta fue ejecutada cinco veces sobre las distintas fuentes de datos consideradas y almacenadas previamente en el repositorio RDF del lado del *endpoint*.

BENCHMARK 1: En las Tablas III, IV, V, VI y VII se presentan los resultados obtenidos con respecto al tiempo total de la operación de consulta desde el dispositivo por cada una de las 13 consultas para las cinco fuentes de datos sintetizadas de LUBM en *cold cache* y en *warm cache*. En cada tabla se hace una comparación entre los formatos RDF/XML [19], RDF/JSON [20] y texto plano.

BENCHMARK 2: En las Tablas VIII y IX se presentan los resultados obtenidos con respecto al tiempo total de la operación de consulta desde el dispositivo por cada uno de los bancos de cinco consultas para las fuentes de datos reales provistas por DBpedia en *cold cache* y en *warm cache*. En cada tabla se hace una comparación entre los formatos RDF/XML [19], RDF/JSON [20] y texto plano.

Tabla IV: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre *Univ(5, 0)* de LUBM

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	1,084	0,717	1,059	0,667	1,029	0,562
Q02	1,831	0,807	1,768	0,728	1,659	0,708
Q03	1,139	0,660	1,078	0,644	1,039	0,612
Q04	1,082	0,618	1,068	0,747	1,064	0,718
Q05	1,389	0,860	1,261	0,810	1,217	0,664
Q06	5,873	3,881	5,136	3,617	4,402	2,963
Q07	1,240	0,806	1,208	0,722	1,187	0,685
Q08	3,678	2,644	3,472	2,349	2,835	1,757
Q09	2,493	0,886	2,318	0,846	2,239	0,777
Q10	1,115	0,711	1,050	0,663	1,002	0,624
Q11	1,154	0,773	1,124	0,667	1,047	0,647
Q12	1,168	0,711	1,063	0,690	1,031	0,662
Q13	3,669	2,491	3,510	2,575	2,670	1,538
Promedio	2,070	1,274	1,932	1,210	1,725	0,994

Tabla V: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre *Univ(10, 0)* de LUBM

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	1,096	0,691	1,072	0,669	1,059	0,637
Q02	2,664	0,795	2,546	0,694	2,482	0,631
Q03	1,168	0,654	1,092	0,641	1,058	0,589
Q04	1,204	0,681	1,139	0,671	1,055	0,710
Q05	1,378	0,808	1,195	0,788	1,171	0,784
Q06	11,575	8,699	10,346	7,588	7,326	4,535
Q07	1,223	0,771	1,120	0,745	1,105	0,685
Q08	3,986	2,943	3,539	2,527	3,133	2,014
Q09	3,136	0,927	3,109	0,910	3,128	0,879
Q10	1,100	0,730	1,065	0,662	0,972	0,652
Q11	1,222	0,754	1,109	0,732	1,082	0,682
Q12	1,087	0,665	1,066	0,623	1,049	0,607
Q13	4,577	3,190	1,066	2,759	3,283	2,230
Promedio	2,724	1,716	2,266	1,539	2,146	1,203

Tabla VI: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre *Univ(20, 0)* de LUBM

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	1,125	0,689	1,083	0,671	1,067	0,661
Q02	6,137	0,902	5,634	0,873	5,294	0,843
Q03	1,175	0,700	1,124	0,616	1,069	0,611
Q04	1,177	0,756	1,166	0,671	1,104	0,633
Q05	1,389	0,893	1,210	0,729	1,189	0,686
Q06	24,473	21,217	19,837	16,355	11,845	8,866
Q07	1,190	0,773	1,143	0,766	1,122	0,747
Q08	3,882	2,849	3,582	2,527	2,836	1,718
Q09	5,131	1,313	5,085	1,132	4,808	1,109
Q10	1,127	0,705	1,114	0,669	1,051	0,629
Q11	1,159	0,759	1,141	0,707	1,116	0,643
Q12	1,237	0,745	1,174	0,738	1,138	0,715
Q13	8,213	6,901	7,569	5,976	5,281	3,888
Promedio	4,417	3,016	3,912	2,495	2,994	1,673

Los resultados del presente experimento corroboran los valores obtenidos para las otras métricas de evaluación para ambos *benchmarks*, el formato texto plano es ideal si se quisiera sólo visualizar los resultados, más no interpretarlos o manipularlos.

Tabla VII: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre *Univ(50, 0)* de LUBM

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	1,090	0,706	1,072	0,680	1,043	0,653
Q02	5,578	1,710	5,249	1,462	5,124	1,179
Q03	1,158	0,687	1,085	0,664	0,997	0,586
Q04	1,104	0,748	1,068	0,700	1,037	0,668
Q05	1,467	0,898	1,282	0,806	1,221	0,688
Q06	62,091	57,493	56,994	52,418	41,861	37,096
Q07	1,220	0,797	1,201	0,749	1,079	0,756
Q08	4,557	2,994	3,519	2,455	3,156	1,981
Q09	2,743	1,703	2,578	1,533	2,416	1,399
Q10	1,147	0,721	1,094	0,683	1,074	0,652
Q11	1,184	0,735	1,109	0,698	1,048	0,673
Q12	1,179	0,789	1,171	0,750	1,152	0,735
Q13	23,257	20,254	17,315	14,559	13,998	10,806
Promedio	8,290	6,941	7,287	6,012	5,785	4,452

Tabla VIII: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre el *Dataset #1* de DBpedia

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	8,675	3,842	8,433	3,814	8,024	3,745
Q02	12,689	7,556	12,124	7,029	12,009	6,628
Q03	11,157	6,638	10,535	6,500	10,316	6,322
Q04	13,753	7,978	13,463	7,623	12,981	7,355
Q05	35,766	33,442	33,412	30,494	27,901	25,527
Promedio	16,408	11,891	15,593	11,092	14,246	9,915

Tabla IX: Tiempo Total (Segundos) de la Operación de Consulta desde el Dispositivo Móvil sobre el *Dataset #2* de DBpedia

Consultas	RDF/XML		RDF/JSON		Texto Plano	
	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>	<i>cold</i>	<i>warm</i>
Q01	1,135	0,749	1,049	0,684	1,006	0,642
Q02	10,839	8,621	9,104	6,194	7,773	5,214
Q03	1,423	0,854	1,273	0,833	1,243	0,805
Q04	3,026	1,643	3,062	1,726	2,956	1,604
Q05	1,506	0,878	1,478	0,816	1,350	0,841
Promedio	3,586	2,549	3,193	2,051	2,866	1,821

En efecto, en todas las tablas del experimento, el promedio del tiempo total de la operación de consulta es menor en texto plano, tanto en las fuentes de datos reales como sintetizadas, en *cold cache* y *warm cache*.

En este mismo sentido, el formato RDF/JSON es el más adecuado cuando se pretende manejar y/o interpretar los resultados. Esto se confirma en estos resultados tanto para las fuentes de datos sintetizadas como las reales en *cold cache* y *warm cache* dado que el promedio del tiempo total de la operación de consulta en formato RDF/JSON es menor a la misma medida en formato RDF/XML. A pesar de que el formato RDF/JSON toma más tiempo que RDF/XML para dar formato y almacenar los resultados de la consulta, RDF/JSON lo compensa al generar un archivo más ligero que se traduce en menor tiempo de descarga.

Los experimentos para esta métrica fueron totalmente independientes a los experimentos de las métricas anteriores, por lo que los tiempos de descarga pueden diferir ligeramente

respecto al experimento de la métrica previa, dado que la velocidad de descarga, a pesar de que en promedio fue 1.1 Mbps, puede variar durante la descarga, afectando así, el tiempo de la misma, pero al estar promediados todos los valores a cinco veces, esta diferencia se torna menos notoria al generar un valor más preciso.

En general, los tiempos de las consultas en *warm cache* son mejores que los de las consultas en *cold cache* en hasta dos órdenes de magnitud, dando a entender que conforme el servidor reciba más solicitudes de consulta, éste las responderá con mayor rapidez, lo cual resulta muy conveniente desde un ámbito de plataforma móvil como lo es Android. La excepción a la regla es la Tabla VIII en donde la diferencia de tiempos entre *cold cache* y *warm cache* no es tan notoria atribuyendo este hecho a la alta selectividad del banco de consultas para esa fuente de datos, aún así, el tiempo de respuesta en *warm cache* tiende a mejorar en el tiempo.

En conclusión, se puede decir que dados estos resultados de los experimentos, se observa que el esquema de solución planteado genera unos resultados bastante aceptables para dar solución al problema planteado en las métricas consideradas, dado que los resultados muestran que es viable la consulta en lenguaje SPARQL a un repositorio RDF almacenado en la nube, tanto en tiempo como en espacio.

El problema principal, además del tiempo de respuesta, es el espacio en memoria del dispositivo, y en los resultados obtenidos, la consulta Q06 de LUBM sobre la fuente de datos *Univ(50, 0)* es la que produce un resultado con mayor cardinalidad, generando un archivo de 47,3 MB en formato RDF/JSON y 57,08 MB en formato RDF/XML. Sin embargo, esto es un tamaño inferior al de la fuente de datos sobre la que se obtuvieron los resultados, la cual cuenta con un tamaño de 1 163,49 MB. De esta manera, los resultados de la consulta Q06 de LUBM son más fáciles de manipular y almacenar en el dispositivo móvil.

V. CONCLUSIONES Y TRABAJO FUTURO

Con respecto a los resultados del estudio experimental se puede mencionar las siguientes conclusiones:

- En cuanto al tiempo de respuesta de las consultas en el *endpoint*, existe una diferencia importante entre los tiempos de respuesta de las consultas en *cold cache* y *warm cache*, diferencia que mantiene su notoriedad a través de todas las fuentes de datos independientemente del número de ternas de las mismas. Debido a esto, se puede inferir que una vez desplegado el servicio *Web* para procesar las consultas entrantes, éste será capaz de responderlas cada vez más rápido conforme al número de solicitudes que vaya procesando. Adicionalmente, es destacable el hecho de que hay un grupo de consultas en las que el tiempo de respuesta se mantiene en valores muy cercanos tanto en *cold cache* como en *warm cache* sobre las distintas fuentes de datos, en contraste, hay otras consultas de las que puede observarse cómo los tiempos varían en función del número de ternas de la fuente de datos sobre la cual se ejecuten. Esto último puede

atribuirse a que las consultas en las que el tiempo varía en función del número de ternas de la fuente de datos, tienen un poder de selectividad mayor y por consiguiente, requieren un proceso más exhaustivo al momento de seleccionar las ternas resultantes.

- Con respecto al tiempo que toma el *endpoint* en almacenar los resultados de las consultas en un archivo según el formato solicitado, en promedio es razonable, aún cuando la cardinalidad de la solución es alta. El formato RDF/JSON es, en promedio, el que consume más tiempo para dar formato a los resultados y almacenarlo en el archivo. El formato texto plano, en contraste, es el que consume menos tiempo en aplicar formato y almacenarse, esto podría llevar a pensar que este formato es el más conveniente para expresar resultados. Sin embargo, el formato texto plano está diseñado principalmente para visualizar su contenido, los formatos RDF/XML y RDF/JSON son los adecuados si se quisieran procesar e interpretar los resultados.
- En cuanto al tiempo de descarga de los resultados y el tamaño de los mismos en disco, el formato texto plano fue, en promedio, el más ligero en cuanto tamaño y el que consumió menos tiempo durante la descarga en el dispositivo móvil, pues como es lógico pensar, al ser el este formato el que ocupa menos espacio, también es el más rápido en descargar por tener menos datos que transferir. Si se requiere que los resultados se una consulta puedan ser procesados e interpretados, la mejor opción es el formato RDF/JSON, pues a pesar de que este formato en el experimento anterior resultó ser el más lento, en esta ocasión compensa ese tiempo de formato y almacenamiento en buena escala, siendo finalmente más ligero y por consiguiente más rápido al momento de la descarga que RDF/XML. El formato JSON en general, cuenta con *parsers* que presentan un rendimiento superior en la plataforma Android en comparación con el formato XML, haciendo así, más rápido y eficiente el manejo de datos en formato JSON en el dispositivo móvil.
- En referencia al tiempo total de la operación de consulta desde el dispositivo móvil, los resultados corroboran los valores obtenidos en los experimentos anteriores. Ya se había llegado a la conclusión de que el formato texto plano es ideal si se quisiera sólo visualizar los resultados, más no manipularlos, y en efecto, en este experimento el promedio del tiempo total de la operación de consulta es menor en texto plano, tanto en las fuentes de datos reales como sintetizadas en *cold cache* y *warm cache*. También se había mencionado que el formato RDF/JSON es el más adecuado cuando se pretende manipular e interpretar los resultados, esto queda confirmado en este experimento dado que, en promedio, el tiempo total de la operación de consulta en formato RDF/JSON es menor a la misma medida en formato RDF/XML tanto para fuentes de datos sintetizadas como reales en *cold cache* y *warm cache*. Esto a pesar de que el formato RDF/JSON consume más tiempo que RDF/XML para dar formato y almacenar los resultados de una consulta, pero RDF/JSON lo compensa al generar un archivo más ligero que se traduce en menor

tiempo de descarga. De manera general, los tiempos de las consultas en *warm cache* son mejores que los de las consultas en *cold cache* en hasta dos órdenes de magnitud, dando a entender que, conforme el servidor reciba más solicitudes de consulta, éste las responderá con mayor rapidez, lo cual resulta muy conveniente desde un ámbito de plataforma móvil como lo es Android

Finalmente, los resultados obtenidos muestran que, en el caso promedio, el esquema planteado presenta un rendimiento razonable con respecto a todas las métricas de evaluación que se definieron para el estudio experimental sobre las fuentes de datos sintetizadas y reales consideradas. En general se obtuvo que el formato RDF/JSON es el más adecuado para consultar desde un dispositivo móvil por ser el más ligero y el más rápido de interpretar en el dispositivo móvil, al ser más ligero en tamaño, se mostró que el tiempo de descarga es menor, aunque hay que notar que RDF/XML también funciona para la tarea de interpretar los resultados. Texto plano por su lado, es el más conveniente cuando se necesite únicamente visualizar los resultados sin requerir que sean procesados dado que este formato es el que resultó ser más ligero y más rápido de almacenar y transferir.

A pesar que los experimentos realizados proporcionan evidencia de las bondades de las soluciones obtenidas por el mecanismo desarrollado en esta investigación, es necesario realizar un estudio más profundo que permita analizar el rendimiento y comportamiento del servicio SparkleService y la aplicación Sparkle al ejecutar otras tareas. Como trabajo futuro se propone realizar las siguientes extensiones al prototipo desarrollado:

- Añadir soporte para los tipos de consultas no considerados en el estudio experimental, actualmente sólo tiene soporte para consultas de tipo SELECT, por lo que se recomienda ampliarlo a las consultas de tipo CONSTRUCT, ASK y DESCRIBE.
- Para una gestión más completa de documentos RDF desde el dispositivo móvil, se recomienda añadir soporte a la operación UPDATE de SPARQL, de esta manera, podrían insertarse nuevas ternas RDF en un grafo RDF en la nube o, de manera análoga, eliminar ternas RDF.

REFERENCES

- [1] *Android*, <http://www.android.com>
- [2] A. Martínez, *BHyper: Procesamiento Eficiente de Documentos RDF con Hipergrafos Dirigidos*, PhD Thesis, Universidad Simón Bolívar, Caracas, Venezuela, April 2012.
- [3] *Guía Breve de Web Semántica*, <http://www.w3c.es/Divulgacion/GuiasBreves/WebSemantica>.
- [4] *AndroJena - Jena Android Porting*, <https://code.google.com/p/androjena>.
- [5] *OpenRDF.org, home of Sesame*, <http://www.openrdf.org>.
- [6] *ARQoid - Porting of Jena's ARQ to the Android Platform*, <https://code.google.com/p/androjena/wiki/ARQoid>.
- [7] S. Harris, A. Seaborne, and E. Prud'hommeaux, *SPARQL 1.1 Query Language*, W3C Recommendations, Tech. Rep., 2013.
- [8] J. Broekstra, A. Kampman, and F. V. Harmelen, *Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema*, in proceedings of the 1st International Semantic Web Conference (ISWC 2002), series Lecture Notes in Computer Science, vol. 2342, ISWC. Sardinia, Italy, June 2002, pp. 54–68.

- [9] J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, *Jena: Implementing the Semantic Web Recommendations*, in proceedings of the 13th International World Wide Web Conference, series Alternate Track Papers & Posters. New York, USA, May 2004, pp. 74–83.
- [10] *Jena – A Free and Open Source Java Framework for Building Semantic Web and Linked Data Applications*, <https://jena.apache.org>.
- [11] *ARQ – A SPARQL Processor for Jena*, <http://jena.sourceforge.net/ARQ>.
- [12] *TDB – A SPARQL Database for Jena*, <https://jena.apache.org/documentation/tdb>.
- [13] P. Webb, D. Syer, J. Long, S. Nicoll, R. Winch, A. Wilkinson, M. Overdijk, C. Dupuis and S. Deleuze, *Spring Framework Reference Documentation*, 4th edition, Pivotal Software, 2014.
- [14] *Spring Framework - Core Support for Dependency Injection, Transaction Management, Web Applications, Data Access, Messaging, Testing and More*, <http://projects.spring.io/springframework>.
- [15] *Android Services*, <http://developer.android.com/guide/components/services.html>.
- [16] P. Webb, D. Syer, J. Long, S. Nicoll, R. Winch, A. Wilkinson, M. Overdijk, C. Dupuis and S. Deleuze, *Spring Boot Reference Guide*, 1st edition, Pivotal Software, 2015.
- [17] *Spring Boot - An Opinionated View of Building Production-ready Spring Applications*, <http://projects.spring.io/spring-boot>.
- [18] P. Webb, D. Syer, J. Long, S. Nicoll, R. Winch, A. Wilkinson, M. Overdijk, C. Dupuis and S. Deleuze, *Web MVC Framework*, 4th edition, Pivotal Software.
- [19] D. Beckett, *RDF 1.1 XML Syntax Specification*, W3C Recommendations, Tech. Rep., February 2014.
- [20] A. Seaborne, K. G. Clark, L. Feigenbaum, and E. Torres, *SPARQL 1.1 Query Results JSON Format*, W3C Recommendations, Tech. Rep., March 2013.
- [21] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, *Hypertext Transfer Protocol - HTTP/1.1, RFC 2324*, June 1999.
- [22] *Lehigh University Benchmark (LUBM)*, <http://swat.cse.lehigh.edu/projects/lubm>.
- [23] *DBpedia*, <http://wiki.dbpedia.org>.

Sistema Inteligente de Reconocimiento de Estados Cerebrales

Junior Altamiranda¹, Jose Aguilar^{1,2}, Luis Hernández³
altamira@ula.ve, aguilar@ula.ve, hernanderator@gmail.com

¹ Departamento de Computación, Universidad de Los Andes, Mérida, Venezuela

² Universidad Técnica Particular de Loja, Loja, Ecuador

³ Laboratorio de Fisiología, Universidad de Los Andes, Mérida, Venezuela

Resumen: En este trabajo se propone una arquitectura para el reconocimiento de los estados cerebrales en un roedor. Nuestro Sistema utiliza la minería de datos para dicha tarea, y se basa en dos módulos que permiten estudiar los fluidos extraídos del cerebro. El primer módulo está basado en la Teoría de Resonancia Adaptativa de las Redes Neuronales, que permite clasificar los datos obtenidos mediante Electroforesis Capilar para identificar los neurotransmisores que se activan en el cerebro bajo diferentes condiciones fisiológicas. El segundo módulo reconoce los patrones de neurotransmisores en el cerebro, a partir de los cuales puede determinar una serie de trastornos producto de alteraciones, almacenamiento o liberación de éstos; este módulo está basado en un Sistema Clasificador Difuso. Como caso de estudio se analiza el Glutamato en roedores, el resultado de las pruebas indica que nuestro sistema es suficientemente útil y eficiente para ayudar a analizar las muestras extraídas del cerebro.

Palabras Clave: Reconocimiento de Patrones; Minería de Datos; Redes Neuronales Artificiales; Sistema Clasificador Difuso.

Abstract: This paper proposes an architecture for the recognition of a rodent brain states. Our system uses data mining for that task, and is based on two modules that allow to study the fluid from the brain. The first module is based on the theory of resonance Adaptive of the networks neural, which allows to classify the data obtained by capillary electrophoresis to identify the neurotransmitters that are activated in the brain under different physiological conditions. The second module recognizes patterns of neurotransmitters in the brain, from which you can determine a series of disorders product alteration, storage or release of these; This module is based on a fuzzy classifier system. As case study analyzes the glutamate in rodents, the result of tests indicates that our system is sufficiently useful and efficient to help analyze samples extracted from the brain.

Keywords: Pattern Recognition; Data Mining; Artificial Neural Networks; Fuzzy Classifier System.

I. INTRODUCCIÓN

La importancia de disponer de técnicas y herramientas de análisis de información almacenada en grandes depósitos de datos representa una alternativa efectiva para lograr abordar problemas en el área de Medicina, específicamente, para analizar los desórdenes del comportamiento que caracterizan enfermedades neurodegenerativas determinadas principalmente por alteraciones en los neurotransmisores. Ahora bien, ese análisis de información también posibilita entender las bases biológicas de la conciencia y los procesos mentales por medio de los cuales percibimos, aprendemos y memorizamos [1][2]. Por ejemplo, el poder interpretar los gráficos generados por la electroforesis capilar, reconociendo patrones, identificando las sustancias químicas que intervienen en las interacciones químicas, entre otras cosas, a través de un método automatizado para la visualización y análisis de los electroferogramas, resulta ser altamente deseable [3][4][5][6].

Algunos trabajos en procesamiento de electroferogramas son los siguientes. En [3][4][5] se usa la transformada wavelet discreta como una herramienta de procesamiento de señales, para hacer análisis multiresolución de los electroferogramas (reducir la dimensión de los datos sin perder la información relevante, para mejorar la calidad de ellos), de tal manera de detectar eventos temporales y suprimir el ruido. En [6] se propone un método de corrección de la línea base (LB) de señales electroforéticas, que explota la representación wavelet a baja resolución de la señal original, para mejorar la calidad de los electroferogramas.

Dada la gran cantidad de datos presentes en el Laboratorio de Fisiología de la Universidad de Los Andes, producto de los experimentos que se realizan para analizar la información proveniente de los fluidos químicos del cerebro de un roedor, surge la necesidad de diseñar e implantar un Sistema Inteligente de Reconocimiento de Estados Cerebrales, que permita analizar e interpretar los datos, y de esta manera

conocer una serie de trastornos producto de alteraciones, almacenamiento y liberación de neurotransmisores en el cerebro.

Nosotros proponemos una arquitectura para reconocer los estados cerebrales en un roedor, compuesto por dos módulos. El primer módulo analiza las sustancias químicas presentes en el cerebro del roedor, para identificar los neurotransmisores presentes en él. Este módulo está basado en las Redes Neuronales Artificiales; particularmente, en la Teoría de Resonancia Adaptativa (ART) propuesta en [2], que permite clasificar los datos obtenidos mediante la electroforesis capilar e identificar los neurotransmisores que se activan en el cerebro bajo diferentes condiciones psicológicas. El segundo módulo reconoce los patrones de los neurotransmisores que actúan en un momento dado en el cerebro, está basado en [1], el cual propone un Sistema Clasificador Difuso. De esta manera, con la información obtenida con la arquitectura se define el estado actual del cerebro (la actividad de un roedor).

II. MARCO TEÓRICO

A. Neurotransmisores

Los neurotransmisores son las sustancias químicas que se encargan de la transmisión de señales entre las neuronas por medio de un Potencial de Acción (PA), desde una neurona (pre-sináptica) hasta la siguiente (post-sináptica), a través de las sinapsis, produciendo una determinada respuesta fisiológica, la cual puede ser inhibitoria o excitadora. Por ser un neurotransmisor debe producir siempre el mismo efecto en el receptor. Los neurotransmisores se encuentran en el terminal de las neuronas (axón), donde estimulan las fibras musculares para contraerlas en el proceso de transmisión de señales. La neurona tiene como función principal la propagación del potencial de acción (impulso o señal nerviosa) a través del axón, para inducir una respuesta en las otras neuronas o células efectoras. Las células efectoras incluyen los músculos esquelético y cardíaco, y las glándulas exocrinas y endocrinas reguladas por el sistema nervioso. La conducción de un impulso a través del axón es un fenómeno eléctrico causado por el intercambio de iones Na^+ y K^+ a lo largo de la membrana [7].

La sinapsis se establece entre neuronas, o en la periferia entre una neurona y un efector (por ejemplo, un músculo). La conexión funcional entre dos neuronas puede establecerse entre el axón y el cuerpo celular, entre el axón y la dendrita (la zona receptiva de la neurona), entre un cuerpo celular y otro, o entre una dendrita y otra. La neurotransmisión puede aumentar o disminuir para generar una función, o para responder a los cambios fisiológicos. Muchos trastornos neurológicos y psiquiátricos son debido a un aumento o disminución de la actividad de determinados neurotransmisores (NT), y muchas drogas pueden modificarla; algunas producen efectos adversos (por ejemplo, los alucinógenos), y otras pueden corregir algunas disfunciones patológicas (por ejemplo, antipsicóticos) [7]. Los neurotransmisores han sido clasificados en:

1. **Inhibidores:** Actúan sobre receptores asociados a canales iónicos, abren canales de cloro, producen una hiperpolarización de la membrana post-sináptica, y disminuyen la actividad neuronal. Entre los inhibidores tenemos: gama amino butírico (gaba), la taurina, la glicina y la alanina.

2. **Excitadores:** Actúan sobre receptores asociados a canales iónicos, abren los canales de sodio, producen despolarización de la membrana post-sináptica y aumentan la actividad neuronal. Entre los excitadores tenemos: el homocisteico, el aspartato y el glutamato.

Para que una sustancia química sea considerada como un neurotransmisor debe cumplir los siguientes criterios [7]:

- Debe encontrarse en el área pre-sináptica del terminal de un axón.
- Las enzimas necesarias para su síntesis también se encuentran presentes en el área pre-sináptica.
- En condiciones fisiológicas, la estimulación de la neurona ocasiona su liberación en cantidades suficientes como para ejercer un efecto fisiológico.
- Existen mecanismos en la sinapsis para terminar rápidamente con su acción por destrucción o recaptación.
- Su aplicación directa en la terminar post-sináptica ocasiona una respuesta idéntica a la producida por estimulación de la neurona.

B. Electroforesis Capilar

La palabra electroforesis es de origen griego. Está compuesta de la palabra $\eta\lambda\epsilon\kappa\tau\rho\nu$ (se pronuncia electrón) que significa ámbar y el verbo $\phi\omega\rho\epsilon\omicron$ (se pronuncia foréo) que significa llevar de un lado a otro [8]. La electroforesis capilar es una técnica de separación de pequeñas y grandes moléculas que utiliza un campo eléctrico para transportar sustancias químicas de un extremo al otro en el interior de un tubo de diámetro fino – entre $2\ \mu\text{m}$ y $150\ \mu\text{m}$ – llamado capilar [2].

En la electroforesis capilar el conductor es una solución electrolítica, rodeada de un aislante que es la pared del capilar. Al aplicar un voltaje entre los dos extremos de un capilar lleno con solución electrolítica, se produce un flujo llamado “flujo electrosmótico” que arrastra todas las partículas que se encuentran en la solución., pero al mismo tiempo dichas partículas están sometidas a la acción de un campo eléctrico que las hace separarse según su carga (ver Figura 1).

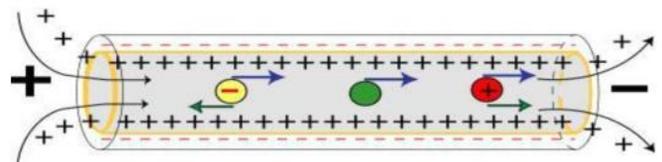


Figura 1: Movimiento de las Moléculas Dentro del Capilar [8]

La espectrofotometría es una de las técnicas experimentales más utilizadas para la detección específica de moléculas de distinta naturaleza (biomoléculas, contaminantes, etc.), y estados de agregación (sólido, líquido, gas). El fundamento físico-químico de la espectrofotometría está relacionado con la capacidad de las moléculas de absorber energía luminosa y almacenarla en forma de energía interna. Como consecuencia, el espectro de absorción, es decir, la luz absorbida en función de la longitud de onda, constituye una verdadera señal de identidad de la molécula (dos moléculas distintas presentarán espectros de absorción distintos) [4].

C. Estados Cerebrales

En los humanos surgen una serie de trastornos clínicos debido a alteraciones, almacenamiento, liberación o degradación de los neurotransmisores. En la mayoría de los casos es causada por trastorno de algún neurotransmisor en alguna parte del cerebro. Algunos de esos trastornos son [7]:

- **Ansiedad:** Son un grupo de síndromes caracterizados por síntomas de preocupación excesiva, miedo intenso, hipervigilancia y síntomas somáticos en ausencia de una situación peligrosa. Puede ir acompañado de síntomas físicos, como fatiga, temblor, tensión muscular, dolor de cabeza y náuseas. Es tipo de trastorno refleja una disminución del neurotransmisor gaba.
- **Depresión:** Es un trastorno de estado de ánimo que provoca perturbaciones emocionales; si la persona está deprimida o sufre una enfermedad depresiva, experimenta un cambio constante del estado de ánimo que le deja una sensación de tristeza, desvalorización y desamparo. Este tipo de trastorno está caracterizado por un desequilibrio de la serotonina y la noradrenalina.
- **Alzheimer:** Es un trastorno que provoca pérdida de funciones y muerte de neuronas en las regiones del cerebro que son responsables del aprendizaje y la formación de la memoria, lo que conlleva a la pérdida de la memoria y la demencia. Los síntomas son causados por una deficiencia del neurotransmisor acetilcolina.
- **Parkinson:** Es un trastorno degenerativo progresivo del sistema nervioso central, caracterizado por el disfuncional control motor causado por la pérdida de las neuronas que producen dopamina y noradrenalina.
- **Epilepsia:** Es una enfermedad neurológica crónica que se caracteriza por las convulsiones recurrentes espontáneas.
- **Esquizofrenia:** Es un trastorno psiquiátrico grave que hace perder contacto con la realidad. Se caracteriza por una profunda discontinuidad del pensamiento y la percepción que afecta atributos básicos del ser humano, incluidos el lenguaje y el sentido del yo. Este trastorno aumenta la dopamina
- **Lesión cerebral:** Es un trastorno asociado con convulsiones prolongadas. La lesión estimula la liberación excesiva de glutamato que origina aumento de calcio y sodio, y la muerte neuronal.
- **Migraña:** Es un complejo número de síntomas que comprende ataques de dolores de cabeza, náuseas, vómitos, e hipersensibilidad a la luz y el sonido. Se caracteriza por un desequilibrio en la concentración de serotonina.

Por otro lado, otros conceptos importantes para establecer el estado cerebral son los siguientes:

- **Función neuronal:** refleja la acción que cumplen las células nerviosas ante el efecto de un neurotransmisor. En un circuito nervioso pueden existir neuronas que producen y liberan determinado neurotransmisor, dejar de liberarlo, o simplemente no son afectadas.
- **Función circuito neuronal:** representa el efecto de un neurotransmisor al funcionamiento de determinado circuito nervioso en diferentes roedores, indicando si el

circuito nervioso funciona igual en todos o difiere en algunos.

Con esos conceptos, es posible determinar el estado cerebral del cerebro basado en el patrón de neurotransmisores presente en él en un momento dado.

D. Red Neuronal ART

Esta red utiliza el dilema de la estabilidad y la plasticidad del aprendizaje. Estos dilemas plantean los siguientes interrogantes [9][10]:

- ¿Cómo una red puede aprender nuevos patrones? (plasticidad del aprendizaje).
- ¿Cómo una red puede retener los patrones previamente aprendidos? (estabilidad del aprendizaje).

En respuesta a estos dilemas, Grossberg y Carpenter desarrollaron la denominada Teoría de Resonancia Adaptativa (Adaptive Resonance Theory: ART) [10][11]. Esta teoría se aplica a sistemas competitivos (redes con aprendizaje competitivo) en los cuales cuando se presenta cierta información de entrada sólo una de las neuronas de salida de la red (o una por cierto grupo de neuronas) se activa, alcanzando su valor de respuesta máximo después de competir con las otras. Esta neurona recibe el nombre de vencedora (winner-take-all unit).

Lo que se pretende es categorizar (agrupar) los datos que se introducen en la red. Las informaciones similares son clasificadas formando parte de la misma categoría, y por lo tanto, deben activar la misma neurona de salida (la neurona vencedora). Las clases o categorías deben ser creadas por la propia red (aprendizaje no supervisado), a través de las correlaciones entre los datos de entrada.

La teoría de la resonancia adaptativa se basa en la idea de hacer resonar la información de entrada con los representantes o prototipos de las categorías que reconoce la red. Si entra en resonancia con alguno es suficientemente similar, la red considera que pertenece a dicha categoría, y únicamente realiza una pequeña adaptación del prototipo almacenado representante de la categoría para que incorpore algunas características del dato presentado. Cuando no resuena con ninguno (no se parece a ninguno de los existentes, recordado por la red hasta ese momento), la red se encarga de crear una nueva categoría con el dato de entrada como prototipo de la misma. Estas redes suelen denominarse ART y ART2 [11]. Ambas difieren en la naturaleza de la información que se presenta de entrada. La red ART trabaja con vectores de entrada binarios, mientras que ART2 es capaz de procesar informaciones continuas o analógicas. En nuestro caso utilizaremos la red ART2.

E. Sistemas Clasificadores Difusos

Los Sistemas Clasificadores (SC) fueron propuestos por Holland como un modelo de aprendizaje por refuerzo, basado en Algoritmos Genéticos (algoritmos de búsqueda que emplean técnicas tomadas de la evolución natural), que permiten una representación flexible del conocimiento. Básicamente son sistemas basados en reglas (classifiers) diseñadas para interactuar con su entorno, y aprender de él mediante la asignación de pesos a cada regla y la creación de nuevas reglas a partir de las anteriores; es decir, aprenden

reglas sintácticamente simples que, en su conjunto, resuelven un problema determinado.

Los SC poseen como característica primordial, la adaptabilidad necesaria para poder aprender un comportamiento, lo cual es lograda a través de los algoritmos genéticos, modificando la población inicial de reglas, garantizando que las nuevas reglas sean mejores que las anteriores. Las características principales de un SC se resumen en los siguientes aspectos [3]:

- Las reglas tienen la forma de una sentencia lógica (condición/acción), identificando una condición con una acción a realizar en caso de que la condición se cumpla.
- Un SC permite la activación paralela de reglas durante un ciclo dado.

Un SC posee una arquitectura básica de 3 subsistemas, tal como se muestra en la Figura 2.

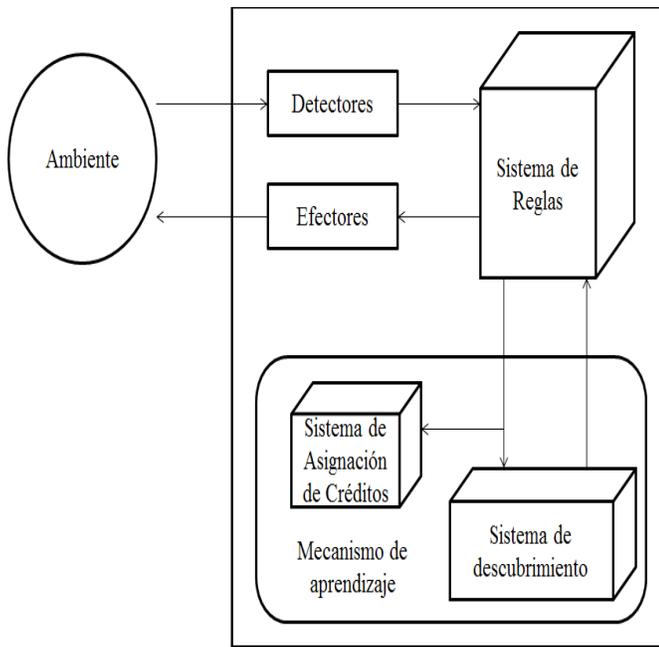


Figura 2: Arquitectura de un SC [3]

Un SCD es un sistema cuyas reglas están basados en la teoría de LD, el cual integra los mismos elementos de un SC, pero trabajando en un marco difuso.

Así, en un SCD las reglas o clasificadores son reglas difusas, es decir, los elementos <condición> y <acción> tienen características difusas. De esta manera, la activación de una regla se logra cuando se verifica el cumplimiento de las instancias en su <condición> para los valores de las variables difusas que provienen del ambiente.

Debido a la connotación difusa de las reglas del SCD, es necesario establecer que elemento del proceso de inferencia difusa define el valor de una regla. El peso de cada regla sería el elemento tomado en cuenta cuando se establezca el valor del crédito en cada ciclo de reglas y mensajes, ya que dicho peso es el que indica el grado de activación de la condición de la regla.

Los Algoritmos Genéticos son utilizados para adaptar todos los elementos involucrados en el proceso de inferencia difusa, como se muestra en la Figura 3.

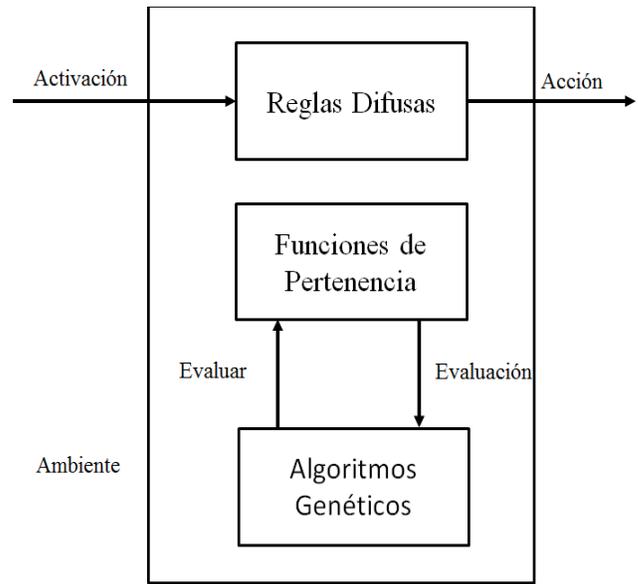


Figura 3: Elementos del SCD [3]

III. PROBLEMA

En el Departamento de Fisiología, en la Facultad de Medicina de la Universidad de los Andes, se realizan experimentos para determinar cambios bioquímicos en el cerebro de roedores y seres humanos, con el fin de entender las interacciones que suceden en el cerebro cuando se realiza una actividad dada. Cuando esto ocurre hace que se active una parte del cerebro y que actúen en él sustancias específicas para realizarlas. Esto permite entender el funcionamiento del cerebro humano, ya que muchas de las sustancias e interacciones ocurren igual para los roedores y los humanos

Al realizar dichos experimentos se extrae las sustancias químicas del cerebro del roedor a través de un capilar, luego este líquido es procesado para obtener los electroferogramas (ver Figura 4), con los datos obtenidos se genera una gráfica que representa el conjunto de interacciones que ocurren según las sustancias que actúan. Todos los datos son almacenados en una base de datos para ser analizada de forma manual y poder descubrir las sustancias e interacciones que genera el proceso, y así generar conocimiento. Por lo tanto, todo esto hace que la mayoría de la información no pueda ser analizada dada la gran cantidad de datos, y parte del conocimiento se pierde.

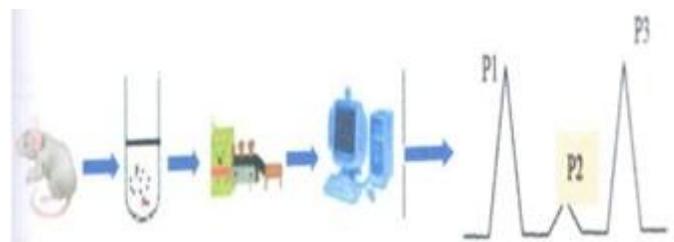


Figura 4: Proceso para Obtener los Electroferogramas (Elaboración Propia)

Así en este trabajo se propone un sistema basado en Redes Neuronales y Lógica Difusa que permita analizar e interpretar datos, y de esta manera conocer una serie de trastornos producto de alteraciones, almacenamiento, liberación o degradación de las sustancias químicas.

Al analizar los electroferogramas se obtiene la interacción de los neurotransmisores. En particular, los electroferogramas están caracterizados por un conjunto de picos que deben ser estudiados, ya que cada uno de ellos representa una sustancia química que actúa en un momento determinado en el cerebro del roedor mientras realiza una actividad (ver Figura 5).

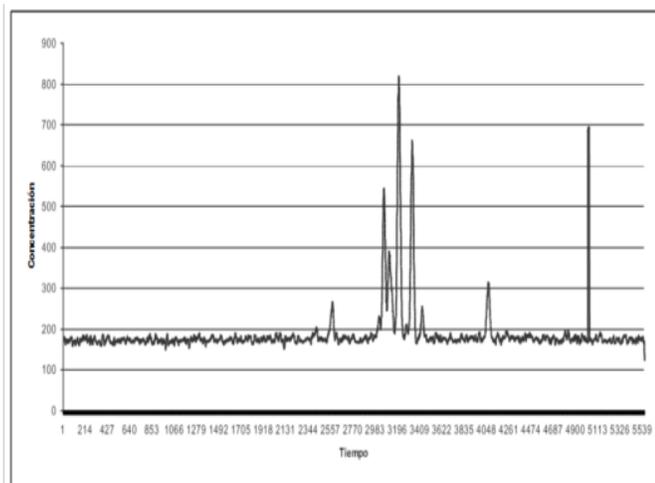


Figura 5: Electroferograma (Gráfica Correspondiente a una Muestra Extraída de un Roedor)

A partir del electroferograma, se puede determinar lo siguiente [3][4]:

- Cada pico representa un neurotransmisor con efecto excitador/inhibidor.
- Un neurotransmisor está presente solo una vez.
- Cada pico del electroferograma está caracterizado por 3 variables: altura, área, ancho. Siendo la altura la más utilizada y representativa.

Así, en general el proceso de análisis de electroferogramas es caracterizado por los siguientes aspectos:

- Los electroferogramas recogen un conjunto de datos del cerebro del roedor. Estos contienen información importante que se encuentra oculta, por lo que es necesario identificarla, extraerla e interpretarla
- Se han de definir medidas cualitativas (área, altura, punto inicio, punto fin, ancho de los picos) para cada uno de los neurotransmisores identificados.
- Para cada uno de los neurotransmisores objeto de estudio se deben hacer gráficas de las variables (área, altura) y cálculos estadísticos (media, varianza, desviación estándar), para observar el comportamiento de éste en diferentes muestras.
- Los resultados deberán ser presentados de una manera entendible para el ser humano.

A partir de las consideraciones anteriores, se logra entender y analizar los neurotransmisores presentes en un momento dado en el cerebro usando electroferogramas. Ahora, todos esos aspectos de análisis de cada electroferograma lo realizan un experto neurofisiólogo, lo que hace que este proceso de procesamiento de electroferogramas sea muy lento, subjetivo, y además, con una gran pérdida de información oculta.

En ese sentido, se requiere de un sistema que permita identificar de forma automática los neurotransmisores presentes en los electroferogramas, determinar los cambios y tendencias de éstos en diferentes muestras, y especificar si un neurotransmisor es liberado o no por las células nerviosas, para de esta manera establecer el estado actual del cerebro.

El sistema propuesta está dividido en 3 fases, como se muestra en la Figura 6:

- Fase 1: Identificación de los neurotransmisores usando una Red Neuronal ART
- Fase 2: Análisis estadístico de los neurotransmisores presentes
- Fase 3: Reconocimiento de los estados cerebrales, por medio del diagnóstico de los trastornos en los neurotransmisores, usando un clasificador difuso.

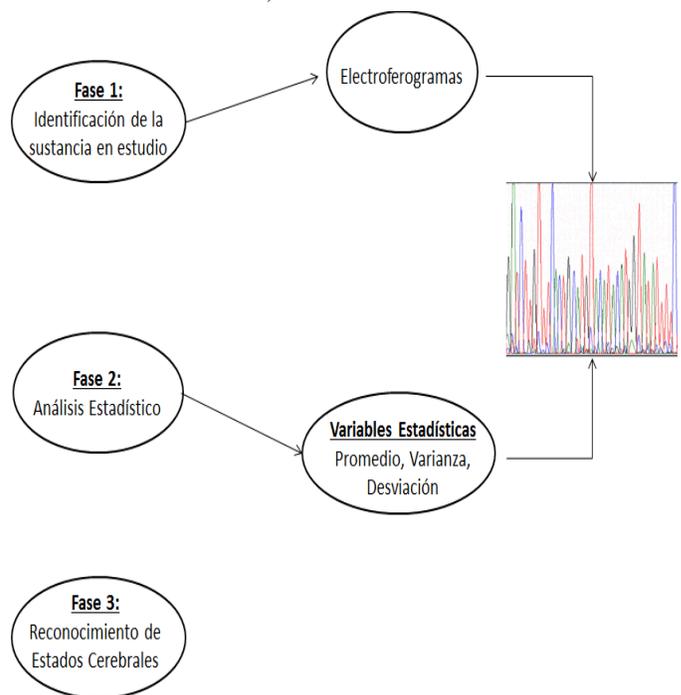


Figura 6: Fases del Proceso de Reconocimiento de Estados Cerebrales (Elaboración Propia)

El sistema propuesto permite generar datos estadísticos sobre los neurotransmisores presentes en el cerebro, y analiza dichos datos con el fin de generar información útil sobre las actividades celulares que ocurren en el cerebro, así como las sustancias que actúan. A continuación pasamos a explicar el comportamiento del sistema propuesto.

IV. IDENTIFICACIÓN DE LOS NEUROTRANSMISORES

El sistema requiere inicialmente de una fase de preprocesamiento de los electroferogramas. Así, luego de tener los electroferogramas, es necesario filtrar los datos y

separarlos en sectores, para así obtener los picos los cuales representan las sustancias químicas presentes en la muestra. Este proceso se encuentra detallado en [3]. Los métodos que se utilizan son:

- Filtro Savitzky – Golay: es un filtro utilizado en sistemas de datos que contiene ruido en la señal, para sustituir los datos originales por datos que permitan suavizar la señal original, sin producir distorsión de fase; preservando la anchura de los picos de las señales a las que se le aplican.
- Extracción de los picos de la muestra: permite dividir la gráfica en sectores y seleccionar cada sector que representa un pico en ésta.

A. Clasificador Utilizando la Teoría de Resonancia Adaptativa

La representación de picos obtenida en el paso anterior, es analizada con el objetivo de clasificar los picos según las diferentes sustancias químicas candidatas (neurotransmisores). En este trabajo, ese proceso se automatiza usando la red neuronal artificial ART2. Así, el proceso de identificación de los picos de la muestra es realizado por una red ART2.

La red neuronal ART2 utiliza como entrada los valores de punto inicio y punto fin con valores enteros. La clasificación es no supervisada, por lo tanto, esta red es capaz de crear clases estables ante la presentación de secuencias de entradas arbitrarias, y fija las fronteras entre estas sin una fase de entrenamiento previo.

Su funcionamiento consiste en que cada vez que la red neuronal ART2 recibe una nueva entrada (punto inicio, punto fin) reacciona activando una y solo una de las neuronas de salida (neurotransmisor) según un parámetro de vigilancia (ρ). Cada una de estas neuronas representa a uno de los diferentes neurotransmisores (clases) que se ha creado en entradas anteriores. En caso de que la entrada no se parezca lo suficiente a alguna de las clases ya aprendidas, se crea una nueva clase que representa a la entrada y se le asigna una neurona de salida nueva. Por lo tanto, la red neuronal ART2 en el sistema puede identificar la entrada de la red (punto inicio, punto fin) como miembro de un neurotransmisor (clases) ya definido, o como uno nuevo.

Al finalizar la clasificación, cada uno de los picos de la muestra va a estar representado por su nombre, su entrada a la red, y la clase a la que pertenece (neurotransmisor) (ver Figura 7).

P1	PTO INICIO	PTO FIN	CLASE
P2	PTO INICIO	PTO FIN	CLASE
.....			
Pn	PTO INICIO	PTO FIN	CLASE

Figura 7: Representación de los Elementos Resultantes Luego de la Clasificación de los Picos (Elaboración Propia)

V. ANALISIS ESTADISTICO - DESCRIPTIVO

En esta fase se extraen las características de cada uno de los picos que representan los neurotransmisores; consiste en obtener algunas medidas estadísticas a partir de los datos, que permitan analizar los cambios y variaciones de los neurotransmisores en diferentes muestras.

Particularmente, las propiedades extraídas de los picos del electroferograma son: el área, la altura, el punto de inicio, el punto de fin y el ancho (ver Figura 8).

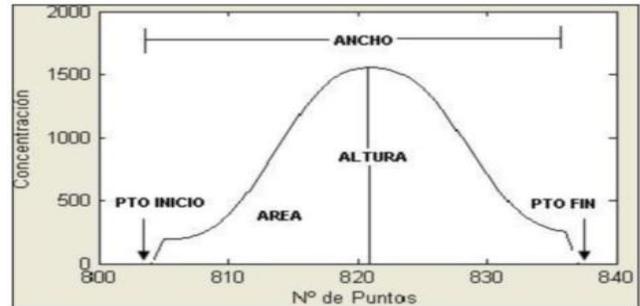
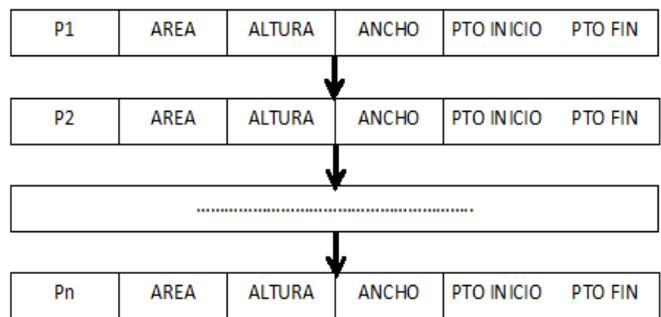


Figura 8: Características de cada Pico Presente en un Electroferograma (Elaboración Propia)

Para obtener las características de los picos se siguieron los siguientes pasos:

- Punto Inicio: Es el lugar donde comienza un pico.
- Punto Fin: Es el lugar donde termina un pico.
- Ancho: es la diferencia entre el punto final y el punto de inicio del pico, este valor nos permite conocer la amplitud de la curva que representa la sustancia química.
- Altura: es el valor máximo en el conjunto de datos que representan la gráfica del pico; este valor es importante, ya que representa la máxima concentración de la sustancia química en ese punto.
- Área: es el valor de la región limitada por la curva de la gráfica y el eje x de coordenadas, permite conocer el volumen del pico.

Estas medidas son almacenadas para cada pico en un repositorio de datos (ver Figura 9). Así, el sistema tiene una base de datos con toda esta información, de todas las muestras que ha analizado.



P1 ... Pn representan los picos que se encuentran en la muestra

Figura 9: Representación del Repositorio de Datos (Elaboración Propia)

La extracción de las sustancias químicas del cerebro se realizan en varios roedores, o se toman varias muestras de un mismo roedor, por lo que, se obtienen numerosos electroferogramas donde se almacenan las variaciones en los valores de altura y área de los picos que se encuentran en éstos (ver Figura 10), estos procesos de recolección o generación de información producen volúmenes tales que superan las capacidades humanas para analizarlas. Esta limitación se debe a varios factores, entre los cuales tenemos la disponibilidad en tiempo, la incapacidad de relacionar grandes volúmenes de datos, entre otros.

	P1		P2		P3			Pn	
	Área	Altura								
Muestra 1										
Muestra 2										
Muestra 3										
...										
Muestra n										

Figura 10: Estructura para Almacenar los Datos Obtenidos de los Picos de la Gráfica (Elaboración Propia)

Las variables estadísticas más importantes en este trabajo son:

- **Altura de los picos:** Una vez identificado el pico que representa el neurotransmisor, se estudia los cambios de altura de éste en distintos electroferogramas, para detectar el tipo de efecto que se produce: excitador, inhibidor o nada. Para cada pico con los datos almacenados en el repositorio de datos de diferentes muestras se pueden construir dos tipos de gráficas:
 - Sin variación en los valores de la altura: el valor de altura de un neurotransmisor permanece igual para diferentes muestras (ver Figura 11); esto representa que no ocurre nada en el cerebro de roedor.

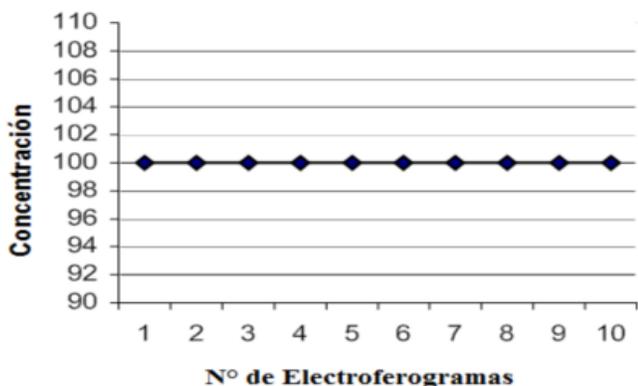


Figura 11: Gráfica donde No Existe Variación en el Valor de la Altura de un Neurotransmisor para Diferentes Muestras (Elaboración Propia)

- Variación en los valores de la altura: permite conocer los cambios que ocurren en el cerebro, ya que los valores de altura de un neurotransmisor cambian para diferentes muestras (ver Figura 12), lo que representa un efecto excitador o inhibitorio de este.

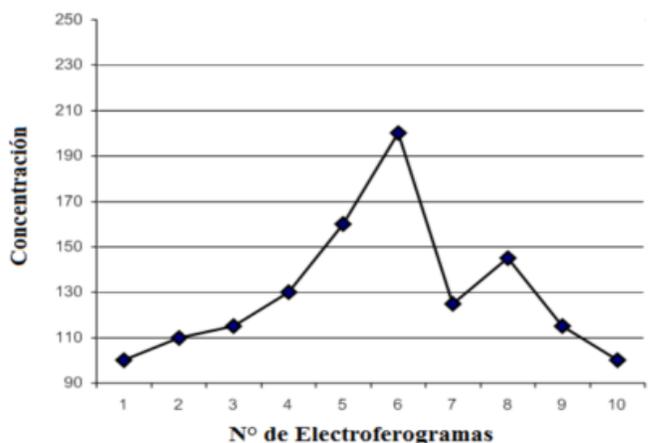


Figura 12: Variación de los Valores de la Altura de un Neurotransmisor para Diferentes Muestras (Elaboración Propia)

- **Promedio de Altura:** El promedio es una medida de tendencia central, y se calcula según (1), para observar la variación en el tiempo de todos los valores de altura que puede alcanzar un neurotransmisor. Permite comparar los cambios de éste en diferentes experimentos (muestras en estado normal, o muestras manipuladas cuando el roedor ha sido tratado con alguna sustancia), y de esta manera determinar si ha sido liberado o no.

$$\overline{Altura} = \frac{\sum_{i=1}^n altura_i}{n} \quad (1)$$

Dónde: altura representa el valor de la altura del neurotransmisor en cada muestra; y n es el número de muestras.

- **Desviación estándar:** Mide la variabilidad existente en el promedio de la altura del neurotransmisor en estudio, se calcula según (2). La desviación estándar indica cuánto tienden a alejarse los valores de altura que puede alcanzar una sustancia, respecto al promedio. Esto indica los cambios a nivel del circuito nervioso. Una desviación estándar alta indica valores de altura lejos del promedio, por lo tanto, el efecto del neurotransmisor produce cambios acentuados en el circuito neuronal. Una desviación pequeña indica que los valores de altura están agrupados cerca del promedio, y el circuito nervioso funciona igual en todos los roedores objetos de estudio.

$$Des_{altura} = \sqrt{\frac{\sum_{i=1}^n (altura_i - \overline{Altura})^2}{n}} \quad (2)$$

VI. RECONOCIMIENTO DE ESTADOS CEREBRALES

Para el reconocimiento de los estados cerebrales se utiliza un SCD. La estructura del SCD está basada en la arquitectura propuesta en [3], y permite analizar los datos y generar el estado cerebral en los roedores a partir de los neurotransmisores. En específico, los componentes principales de nuestro SCD se describen a continuación.

A. Estructuras de las Reglas Genéricas

- Si $\langle altura_{TipoSustancia} \rangle$ entonces $\langle efecto \rangle$
- Si $\langle altura_{TipoSustancia} \rangle$ y $\langle promedioAltura_{TipoSustancia} \rangle$ entonces $\langle funciónNeuronal \rangle$
- Si $\langle promedioAltura_{TipoSustancia} \rangle$ y $\langle desviaciónEstandar_{TipoSustancia} \rangle$ entonces $\langle funciónCircuitoNeuronal \rangle$
- Si $\langle altura_{TipoSustancia} \rangle$ y $\langle promedioAltura_{TipoSustancia} \rangle$ y $\langle desviaciónEstandar_{TipoSustancia} \rangle$ entonces $\langle efecto \rangle$ y $\langle funciónNeuronal \rangle$ y $\langle funciónCircuitoNeuronal \rangle$ y $\langle enfermedadAsociada \rangle$

B. Definición de las Variables Difusas y los Conjuntos Difusos

Representa las variables difusas usadas en las reglas genéricas para el análisis de los neurotransmisores, los cuales son: Gaba; Serotonina, Acetilcolina, Histamina, Dopamina, Glutamato, Aspartato y Glicina. En la Tabla I se muestran las variables difusas con sus respectivos conjuntos difusos [3].

Tabla I: Variables Difusas y Conjuntos Difusos

Tipo de Variables	Variables Difusas	Conjuntos Difusos
Variables de Entrada	alturaGaba	baja, igual, alta
	alturaSerotonina	baja, igual, alta
	alturaAcetilcolina	baja, igual, alta
	alturaDopamina	baja, igual, alta
	alturaHistamina	baja, igual, alta
	alturaGlutamato	baja, igual, alta
	alturaAspartato	baja, igual, alta
	alturaGlicina	baja, igual, alta
	promedioAlturaGaba	bajo, intermedio, alto
	promedioAlturaSerotonina	bajo, intermedio, alto
	promedioAlturaAcetilcolina	bajo, intermedio, alto
	promedioAlturaHistamina	bajo, intermedio, alto
	promedioAlturaHistamina	bajo, intermedio, alto
	promedioAlturaGlutamato	bajo, intermedio, alto
	promedioAlturaAspartato	bajo, intermedio, alto
	promedioalturaGlicina	bajo, intermedio, alto
	Variables de Salida	desviaciónEstandarGaba
desviaciónEstandarSerotonina		baja, alta
desviaciónEstandarAcetilcolina		baja, alta
desviaciónEstandarDopamina		baja, alta
desviaciónEstandarHistamina		baja, alta
desviaciónEstandarGlutamato		baja, alta
desviaciónEstandarAspartato		baja, alta
desviaciónEstandarGlicina		baja, alta
Efecto		Inhibidor, sin efecto, excitador
funciónNeuronal		Liberar, no liberar, no afecta
funciónCircuitoNeuronal		Produce cambios, no produce cambios
Ansiedad		baja, igual, alta
Depresión		baja, igual, alta
Alzheimer		baja, igual, alta
Parquinson	baja, igual, alta	
Epilepsia	baja, igual, alta	
Esquizofrenia	baja, igual, alta	
LesiónCerebral	baja, igual, alta	
Migraña	baja, igual, alta	

En la Tabla I, vemos que la variable difusa *enfermedadAsociada* es descrita por los tipos de trastornos cerebrales (Depresión, Migraña, entre otros).

C. Definición de las Funciones de Pertenencia

A continuación se definen las funciones de pertenencia de cada conjunto difuso asociado a cada variable difusa

- Las variables difusas alturaGaba, alturaSerotonina, alturaAcetilcolina, alturaHistamina, alturaDopamina, alturaGlutamato, alturaAspartato, alturaGlicina, están caracterizadas por la función de pertenencia mostrada en la Figura 13. El dominio es [0 mv, 4000 mv], indicando la mínima y máxima altura que puede alcanzar cada pico que representa una sustancia en particular.

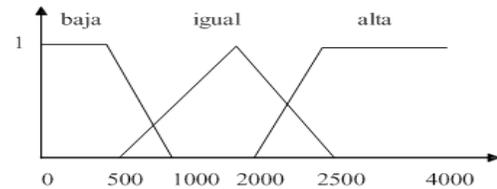


Figura 13: Función de Pertenencia de las Variables Difusas Altura para los Neurotransmisores (Elaboración Propia)

- Las variables difusas promedioAlturaGaba, promedioAlturaSerotonina, promedioAlturaAcetilcolina, promedioAlturaHistamina, promedioAlturaDopamina, promedioAlturaGlutamato, promedioAlturaAspartato, promedioAlturaGlicina, están caracterizadas por la función de pertenencia mostrada en la Figura 14. El dominio es [0 mv, 4000 mv].

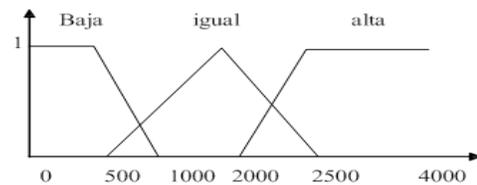


Figura 14: Función de Pertenencia de las Variables Difusas Promedio de Altura para los Neurotransmisores (Elaboración Propia)

- Las variables difusas desviaciónEstandarGaba, desviaciónEstandarSerotonina, desviaciónEstandarAcetilcolina, desviaciónEstandarHistamina, desviaciónEstandarDopamina, desviaciónEstandarGlutamato, desviaciónEstandarAspartato, desviaciónEstandarGlicina, están caracterizadas por la función de pertenencia mostrada en la Figura 15. El dominio es [0 mv, 1000 mv].

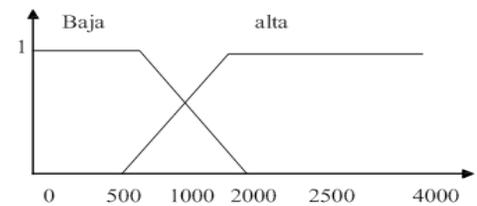


Figura 15: Función de Pertenencia a la Variación Estandar para los Neurotransmisores (Elaboración Propia)

- La variable efecto está caracterizada por la función de pertenencia mostrada en la Figura 16. El dominio es [0%, 100%].

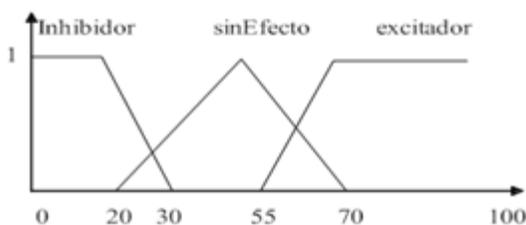


Figura 16: Función de Pertenencia de las Variables Difusa Efecto (Elaboración Propia)

- La variable difusa funciónNeuronal está caracterizada por la función de pertenencia mostrada en la Figura 17. El dominio es [0%, 100%].

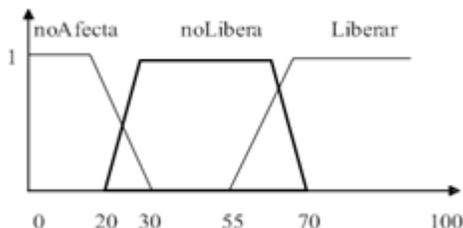


Figura 17: Función de Pertenencia de las Variables Difusa Función Neuronal (Elaboración Propia)

- La variable difusa funciónCircuitoNeuronal está caracterizada por la función de pertenencia mostrada en la Figura 18. El dominio es [0%, 100%].

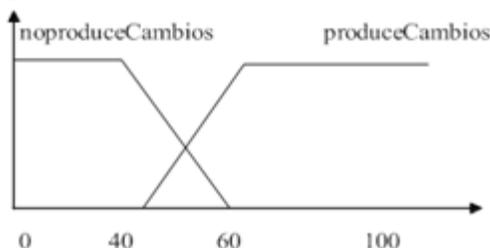


Figura 18: Función de Pertenencia de las Variables Difusa Circuito Neuronal (Elaboración Propia)

- Las variables difusas Ansiedad, Depresión, Alzheimer, Parkinson, Epilepsia, Esquizofrenia, LesiónCerebral, Migraña, están caracterizadas por la función de pertenencia mostrada en la Figura 19. El dominio es [0%, 100%].

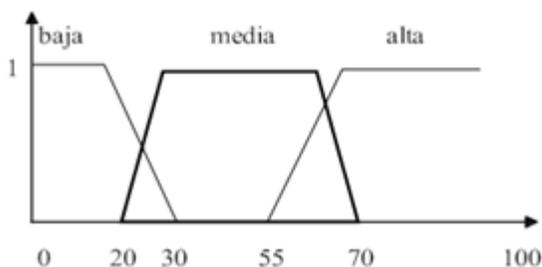


Figura 19: Función de Pertenencia de las Variables Difusas: Ansiedad, Depresión, Alzheimer, Parkinson, Epilepsia, Esquizofrenia, Lesión Cerebral, Migraña (Elaboración Propia)

VII. CASO DE ESTUDIO

A. Definición del Caso de Estudio

Para ejemplificar el funcionamiento de nuestro sistema, se evalúa las variaciones de los niveles extracelulares del

neurotransmisor glutamato en el núcleo del cerebro, y sus implicaciones en las emociones, principalmente en el miedo y en los trastornos de ansiedad.

B. Experimento

El objetivo del experimento es determinar la variación del nivel extracelular del neurotransmisor glutamato en el cerebro de un roedor, bajo los efectos de las siguientes sustancias: jugo, Cloruro de Litio (LiCl), Cloruro de Sodio (NaCl) o ninguna sustancia.

Las pruebas fueron hechas con 3 grupos diferentes, tomando 15 electroferogramas por roedor perteneciente a cada grupo experimental, el tiempo entre cada electroferograma fue de 2 segundos. Cada electroferograma representa una muestra bajo ciertas condiciones (con efecto o sin efecto de sustancias), estos datos son almacenados, y se obtuvieron los electroferogramas respectivos (ver Figura 20). En cada electroferograma se identificó el pico que representa el neurotransmisor en estudio, en este caso el glutamato.

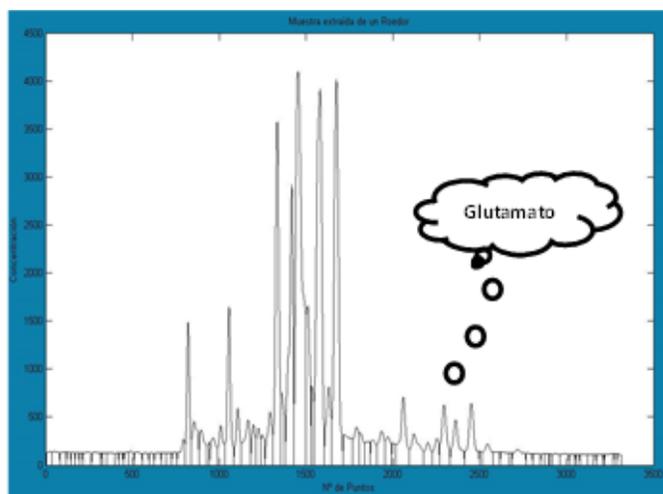


Figura 20: Muestra 1 del Grupo A_01 Extraída del Roedor (Elaboración Propia)

- Grupo 1 (sin los efectos de las sustancias): este grupo está conformado por 6 roedores, que representan las muestras básicas, éstas son muestras donde los roedores no fueron tratados con ninguna sustancia.
- Grupo 2 (bajo los efectos de jugo y Cloruro de Litio (LiCl)): este grupo está conformado por 6 roedores, que fueron tratados bajo los efectos de la sustancia LiCl. Se tomaron las 3 primeras muestras sin darles nada, luego se les dio Jugo a cada roedor para las muestras entre 4 y 6, y por último de la muestra 7 en adelante se les dio LiCl.
- Grupo 3 (bajo los efectos de Cloruro de Sodio (NaCl)): este grupo está conformado por 6 roedores, que fueron tratados bajo los efectos de las sustancias jugo y NaCl. Se tomaron las 3 primeras muestras sin darles nada, luego se les dio Jugo a cada roedor entre las muestras 4 y 6, y por último, de la muestra 7 en adelante se les dio NaCl.

Usando el primero módulo del sistema, a partir de los electroferogramas obtenidos, se clasificaron los picos del 1 al n, donde cada número representa una sustancia química diferente. Luego se obtuvieron las características del pico que

representa el glutamato en diferentes muestras. Estos son los datos que se utilizaron para analizar el cerebro de un roedor (ver Tabla II).

Tabla II: Características del Pico que Representa el Glutamato para Diferentes Muestras

Número de Pico	Altura	Área	Punto Inicio	Punto Fin	Ancho	Clase
34	459,77	11904,41	2268	2329	61	34
34	733,37	18454,32	2263	2328	65	34
30	400,03	10045,55	2264	2325	61	34
36	825,08	19003,8	2271	2327	56	34
33	408,05	10577,72	226	2325	60	34

A continuación, el sistema utiliza los valores almacenados en la Tabla II para obtener los valores promedio de altura y desviación estándar (ver Tabla III). Utilizando el segundo módulo.

Tabla III: Grupo I

Electroferograma	Altura Promedio Glutamato	Desviación Estándar Glutamato
1	218,2	64,32
2	222,2	65,15
3	350	188,42
4	269,4	167,62
5	204	142,75
6	171,25	41,44
7	170,67	23,71
8	195	61,49
9	337	83,44
10	295,67	174,59
11	181,67	27,93
12	209,67	24,17
13	333,67	139,47
14	240,5	186,78
15	188,25	65,05

En la Tabla III se observa la variación de los valores en las muestras. Estos valores se estudian para llegar a conclusiones con respecto a cómo funciona el cerebro de un roedor, en este caso, según como varía el glutamato. Los resultados de la Tabla III representan la entrada al tercer módulo, para reconocer el estado cerebral que tiene un roedor. Para el reconocimiento de los estados cerebrales, cada pico se define por un conjunto de variables difusas, por ejemplo en la Tabla IV se definen las variables de entrada/salida que utiliza el sistema para el análisis del neurotransmisor glutamato.

Tabla IV: Variables de Entrada/Salida

Tipo de Variable	Variable Difusa
Entrada	alturaGlutamato
	promedioAlturaGlutamato
	desviaciónEstandarGlutamato
Salida	Efecto
	FunciónNeuronal
	FunciónCircuitoNeuronal
	Miedo
	Ansiedad

Los conjuntos difusos para las variables de la Tabla IV fueron definidos en la Sección IV en este trabajo. De igual forma, el sistema tiene definidas las funciones de pertenencia para cada conjunto difuso asociada a cada variable difusa. Los rangos del universo están determinados de igual forma en la Sección IV.

De acuerdo a la estructura de reglas genéricas que describe el sistema para el análisis de glutamato, se establecieron las siguientes reglas específicas:

- Si alturaGlutamato es alta entonces efecto es excitador
- Si alturaGlutamato es baja y promedioGlutamato es alto entonces funciónNeuronal es liberar
- Si promedioAlturaGlutamato es alto y desviaciónEstandarGlutamato es alta entonces funciónCircuitoNeuronal es produceCambios
- Si alturaGlutamato es alta y promedioGlutamato es alto y desviaciónEstandarGlutamato es baja entonces efecto es excitador y funciónNeuronal es liberar y funciónCircuitoNeuronal es No Produce Cambios y enfermedadAsociada es Ansiedad.

Para el glutamato, nuestro sistema tiene inicialmente instanciada 14 reglas. La definición de las reglas específicas en nuestro sistema para cada neurotransmisor es similar para el resto de ellos. Nuestro prototipo fue construido haciendo uso de la herramienta para el desarrollo de SCD presentada en [12]. Esta herramienta tiene un mecanismo de adaptación (AM), que permite la modificación de las reglas durante el tiempo de ejecución con el fin de adaptarlas. Cuando el sistema llama a AM para cada regla, introduce una nueva relación entre los conjuntos difusos (entre la condición y la acción), para generar nuevas instancias de reglas. La AM establece nuevos rangos para los conjuntos difusos definidos, y genera funciones de pertenencia diferentes a las que fueron presentadas originalmente. La evolución de las funciones de pertenencia permite corregir errores de diseño y/o actualizar la información del sistema a los cambios que están sucediendo en el ambiente.

Por ejemplo, la evolución de las funciones de pertenencia de la variable difusa alturaGlutamato, donde las funciones de pertenencia inicialmente eran de tipo trapezoidal en el rango (500, 650, 850, 1000), fueron sustituidas por (432, 650, 850, 1000). Igualmente se observa la evolución de la función de pertenencia para promedioAlturaGlutamato, inicialmente era de tipo trapezoidal, y al evolucionar la función de pertenencia cambio al tipo triangular en el rango (0, 110, 220).

C. Resultados

El sistema muestra las reglas activadas en el proceso de inferencia, y genera las respectivas soluciones (que es la salida de nuestro sistema). Los resultados para el Grupo I, en el caso de glutamato: alturaGlutamato = 600, promedioAlturaGlutamato = 504,75 y desviaciónEstandarGlutamato = 273,58. El sistema genera una solución asociada a la variable “efecto” (76,77%), lo que indica un efecto excitante dado por un aumento de la altura en el pico que representa el glutamato por la acción que produce el LiCl.

La variable “funciónNeuronal” (59,92%), en este caso indica que la función cerebral es afectada por la LiCl y causa que algunas neuronas liberen un poco de glutamato. Por otro lado,

la “funcionCircuitoNeuronal” (24,69%) en este caso indica que para los 6 roedores que componen el experimento, ocurren pocos cambios en la zona del cerebro en estudio, y la operación de esta zona cerebral funciona igual para todos los roedores del Grupo I.

De esta manera, el sistema puede determinar que el roedor está bebiendo, pero, además, las consecuencias de la bebida (por ejemplo, de LiCl). De acuerdo con las variaciones en los niveles de la glutamato debido al LiCl, el cerebro puede tener perturbaciones, y el sistema difuso es capaz de asociar posibles trastornos en él, como el miedo, la ansiedad, la depresión.

Particularmente, de acuerdo con los grupos experimentales en el estudio, podemos observar que LiCl aumenta el glutamato y afecta el cerebro, como se muestra en la Figura 21, donde el promedio de altura del glutamato desde la muestra 6 aumenta (comienza en el grupo 2). Además, entre los grupos 1 y 3 se observa una diferencia en la altura del glutamato, que identifica que un roedor está bebiendo (en este caso, NaCl, grupo 3) con respecto a cuándo el roedor no está bebiendo (grupo 1). Por lo tanto, comparando las escalas del promedio de las alturas del neurotransmisor para los 3 grupos, nuestro sistema ha detectado varias cosas:

- Cuando un roedor está bebiendo algo (grupo 2 y 3) o no (grupo 1). Así, se puede determinar la actividad del roedor.
- Los trastornos, como el miedo, la ansiedad, la depresión, se puede asociar a los valores de los neurotransmisores, y de esta manera, se determina el estado cerebral de la persona.

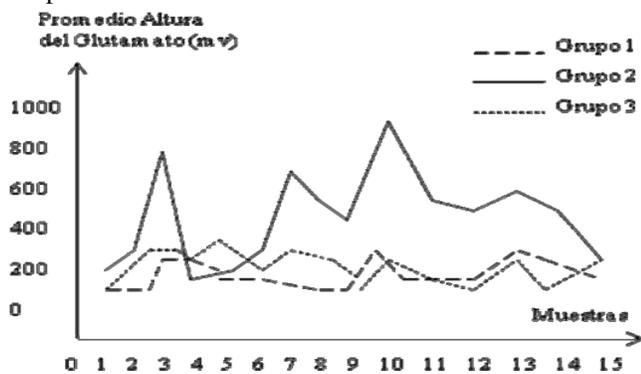


Figura 21: Promedio de Altura del Glutamato para los Diferentes Grupos Experimentales (Elaboración Propia)

VIII. CONCLUSIONES

A partir de los resultados obtenidos, el sistema puede determinar el estado actual del cerebro de un roedor bajo la siguiente premisa: “el estado actual del cerebro puede ser determinado por el comportamiento de los neurotransmisores activos en un momento dado en el cerebro”.

Basado en lo anterior, se ha desarrollado un sistema para extraer el conocimiento que se encuentra oculto en los neurotransmisores del cerebro. Básicamente, el sistema transforma los neurotransmisores presentes en un momento dado en el cerebro, en patrones que describen el estado cerebral.

Así, el sistema permite analizar las bases biológicas (según los neurotransmisores presentes) que determinan en un momento dado el estado cerebral. Próximos estudios extenderán estos resultados, para analizar su uso para el estudio de la conciencia, de los procesos mentales por medio de los cuales percibe, aprende y memoriza el cerebro humano, debido a que estos procesos ocurren de manera semejante en los roedores y los humanos. También, se extenderá para incluir la utilizando de ontologías Biomédicas en el sistema propuesto, de tal manera de explotar el conocimiento almacenado en ellas durante el proceso de análisis del estado cerebral.

En particular, el caso de estudio consistió en analizar el neurotransmisor glutamato. Se analizó el glutamato específicamente, por la facilidad para localizarlo y/o detectarlo en un electroferograma. Se implementó un prototipo para determinar la variación del glutamato en la amígdala del cerebro, inducida por la acción de varias sustancias (jugo, LiCl, NaCl). Los resultados pueden ser considerados muy prometedores; el sistema descubre patrones de las sustancias químicas y obtiene resultados consistentes.

Los resultados obtenidos indican que el sistema propuesto es útil para ayudar a analizar las muestras extraídas del cerebro, porque provee un conjunto de datos estadísticos y graficas que muestran el comportamiento de los neurotransmisores en el cerebro en diferentes muestras. Por otro lado, el sistema contiene reglas médicas, que les permiten a los biólogos descubrir alteraciones cerebrales producto de cambios en los neurotransmisores. A partir de todo lo anterior, un experto puede hacer un análisis más exacto de lo que acontece en el cerebro.

AGRADECIMIENTO

Al Proyecto CDCHTA I-1407-14-02-B de la Universidad de Los Andes por su apoyo financiero. Dr. Aguilar ha sido parcialmente financiado por el Proyecto Prometeo del Ministerio de Educación Superior, Ciencia, Tecnología e Innovación de la República de Ecuador.

REFERENCIAS

- [1] J. Aguilar y L. Hernández, *Design and Implementation of a Patterns Recognition System for Analysis of Biological Liquids*, IEEE Latin America Transactions, vol. 7, no. 1, pp. 12 - 26, 2009.
- [2] J. Altamiranda, J. Aguilar y L. Hernández, *Sistema de Reconocimiento de Patrones de Sustancias Químicas Cerebrales Basado en Minería de Datos*, Computación y Sistemas, vol. 19, no. 1, pp. 89-107, 2015.
- [3] G. Ceballos, J. L. Paredes y L. Hernández, *Data Processing and Pattern Recognition in High-Throughput Capillary Electrophoresis*, 17th European Signal Processing Conference (EUSIPCO), Glasgow, Scotland, August 2009.
- [4] G. Ceballos, J. L. Paredes y L. Hernández, *A Novel Approach for Pattern Recognition in Capillary Electrophoresis Data*, en las memorias del IV Congreso Latinoamericano de Ingeniería Biomedica (CLAIB), Isla de Margarita, Venezuela, Septiembre 2007.
- [5] J. Cuadros, J. L. Paredes y G. Ceballos, *Herramientas de Software para el Análisis de Datos*, Mecánica Computacional, vol. XXVII, pp. 3299-3315, 2008.
- [6] J. L. Paredes y E. Sosa, *Corrección de Línea Base en Datos Electroforéticos Usando Optimización Local del Algoritmo Legnt en el Dominio Wavelet*, Interciencia, vol. 34, no. 8, 2009.
- [7] K. Barrett, S. Barman, S. Boitano y H. Brooks, *Ganong Fisiología Médica*, McGraw Hill, 20013.

- [8] V. León, *Manual de Usuario del Onice*, Universidad de Los Andes, Facultad de Medicina, Departamento de Fisiología, Mérida, Venezuela, 2004.
- [9] G. Carpenter y S. Grossberg, *The ART of Adaptive Resonance Theory by a Self-Organizing Neural Network*, IEEE Computer, vol. 21, no. 3, pp. 77-88, 1988.
- [10] G. Carpenter y S. Grossberg, *Pattern Recognition by Self-Organizing Neural Network*, MIT Press, 1991.
- [11] G. Carpenter y S. Grossberg, *ART2 - Self - Organization of Stable Category Recognition Codes for Analog Input Patterns*, Applied Optics, vol. 26, pp. 4919-4930, 1987.
- [12] G. González, J. Aguilar, *Reconocimiento de Patrones Usando Sistemas Clasificadores Difusos*, en las memorias del Congreso de Computación Aplicada, pp. 57-59, San Cristobal, Venezuela, Octubre 2006.

Índice de Autores

A

Aguilar, José	1, 36, 107
Akiyama, Minoru	60
Altamiranda, Junior	107

C

Cardinale, Yudith	1
-------------------	---

D

Delgado, Mayela	14
Dos Santos, Ricardo	36

G

Grillo, Marcos	1
Guevara, María	26

H

Hernández, Luis	107
Hernández, Luiyit	70

L

Lima, Sandra	60
Losavio, Francisca	14

M

Martínez, Amadís	95
Matteo, Alfredo	14
Mayo, Andrés	83

O

Ortega, Dinarle	26
-----------------	----

R

Ramírez, Esmitt	70, 83
Ramírez, Josué	48

S

Soto, Jesús	95
-------------	----

T

Tineo, Leonid	48
---------------	----

V

Valera, Pablo	26
---------------	----

REVECOM

Sociedad Venezolana de Computación

La Sociedad Venezolana de Computación está comprometida con el impulso de una nueva generación académica y profesional en nuestra área de saber para el desarrollo del país.

Los conceptos y puntos de vista expresados en los trabajos publicados en este libro representan las opiniones personales de los autores y no reflejan el juicio de los editores o de la Sociedad Venezolana de Computación.

ISSN: 2244-7040



9 772244 704006

www.svc.net.ve/revecom

