Sociedad Venezolana de Computación

Revista Venezolana de Computación

REVECOM

Revista Venezolana de Computación

Sociedad Venezolana de Computación

ISSN: 2244-7040

Vol. 1, No. 2
Diciembre 2014

REVECOM

Revista Venezolana de Computación

Sociedad Venezolana de Computación

Editores:
Eric Gamess, Wilmer Pereira, Yudith Cardinale

ISSN: 2244-7040

Vol. 1, No. 2
Diciembre 2014

Editorial

¿Qué es Computación?

La computación es una disciplina académica completa que trata sobre el computar. Es completa en cuanto a que cubre los cuatro sectores del quehacer académico: la filosofía, la ciencia, la tecnología y la praxis, todo ello en una dimensión humana. La filosofía de la computación es el espacio del pensamiento crítico que trata los fundamentos que sustentan el estudio científico de la computación. La ciencia de la computación es el tratamiento lógico-formal de la matemática en movimiento, vista en una perspectiva funcional. La tecnología de la computación estudia las formas de producir los artefactos lógicos y físicos que en su operación modelen los procesos de cómputo de las funciones. Finalmente, la praxis de la computación cubre los aspectos relativos al manejo eficiente y efectivo de los productos de la tecnología de computación y su aplicación en la solución de los problemas de la humanidad, dentro del marco de las leyes, la ética y la moral.

Intrínseco al concepto de computar está el concepto de función, entendida ésta como una relación unívoca; es decir, una relación en la cual ningún elemento de su dominio está vinculado con más de un elemento. Un ejemplo en el dominio de los seres humanos es la relación genética de ser madre, sólo se tiene una.

Se considera que fue Leibniz el primero en utilizar el término “función” para referirse a las fórmulas que al representarse gráficamente, utilizando la técnica de coordenadas ortogonales introducida anteriormente por Descartes, describía el comportamiento de cuerpos en movimiento. A partir de allí, el análisis funcional se desarrolló inicialmente centrado en el concepto de continuidad y de valuación simple de una sola variable independiente, aunque posteriormente se abrió a la incorporación de múltiples variables independientes.

En su propósito de modelar la realidad observable, estas restricciones se flexibilizaron dando cabida a la valuación múltiple y a los múltiples puntos de discontinuidad. Con Dirichlet, culminó este proceso al manejar funciones discontinuas en todos los puntos de su dominio, con lo cual la fórmula perdía su valor descriptor y se sustituía con el modelo de multiplicidad de políadas, cada una de las cuales representaba las coordenadas de la función. Esta idea es central en la concepción de función que utilizó Frege en su formalización de la lógica simbólica.

La idea de Boole, de hacer un álgebra de la lógica, requería tratar el concepto colectivo de clase como un individuo. Esta idea, a través de Schröder, inspiró a Cantor a desarrollar su teoría de conjuntos transfinitos, como lo es el conjunto de los números reales, objetos elementales del análisis funcional. Los trabajos de Frege y Cantor se conjugaron en Russell que motivó el proyecto formalista de Hilbert para fundamentar la matemática, proyecto que magistralmente acotó Gödel con sus teoremas de incompletitud, usando para ello una variante de la axiomatización de von Neumann, traducida simbólicamente por Bernays, de la teoría de conjuntos previamente axiomatizada por Zermelo con posteriores modificaciones de Fraenkel y Skölem.

Schönfinkel, asistente de Hilbert al igual que Bernays, mediante el uso de operadores morfológicos que denominó combinadores, redujo el estudio de funciones poliádicas al de funcionales monádicos que permitían funciones como argumentos y valores de otras funciones. Curry, en su trabajo de doctorado

con Hilbert, conoció la obra de Schönfinkel y fortaleció sus propias ideas en el tema para desarrollar su lógica de combinadores.

La segunda guerra mundial propició la concentración en la Universidad de Princeton de luminarias de la ciencia como Einstein, Oppenheimer, Weyl, Gödel y von Neumann. Church dirigía un grupo de estudiantes de doctorado entre los que se encontraban Rosser, Kleene y Turing. Es este el grupo forjador de lo que luego se llamaría "Ciencia de la Computación" con importantes aportes posteriores de investigadores del MIT, y de las universidades de Stanford, Carnegie-Mellon y Cambridge, entre otras.

Turing formuló la pregunta de si era posible construir máquinas que mostraran un comportamiento equivalente al que hasta entonces solo podían mostrar los seres pensantes. Minsky en el MIT elaborando modelos computacionales del funcionamiento de las redes neuronales, McCarthy en Stanford desarrollando el lenguaje LISP para el procesamiento de simbolismos basado en los trabajos de Church y Simon en Carnegie-Mellon automatizando los procesos de toma de decisiones, impulsaron el proyecto de Inteligencia Artificial, cuya meta es modelar digitalmente todas las funciones observadas en el comportamiento inteligente de los seres humanos.

La tecnología de la computación nace con Babbage en el siglo XIX, pero en su época no existía la tecnología que hiciese factible sus diseños, y se tuvo que esperar un siglo para su implementación, cuando el desarrollo de la electrónica digital le dio factibilidad. Pero nueva tecnología conlleva nuevas arquitecturas y son Turing y von Neumann los que aportan los nuevos modelos, que han evolucionado para mejorar la eficiencia y capacidad de las máquinas, más no su poder de cómputo. La frontera de las posibilidades de esta tecnología promete una amplísima expansión con el dominio de la nanotecnología y la computación cuántica.

En sus primeros pasos de aprendizaje el niño aprende a contar, que no es otra cosa que computar la función sucesor. De igual forma, en la infancia de la civilización, se encuentran registro de las operaciones de acumulación y substracción, que no es otra cosa que computar las funciones de suma y resta. Esencial para el asentamiento primitivo de la humanidad fue el cómputo del tiempo. Pero no hubiese sido posible la exploración del espacio exterior, la investigación del origen del cosmos, de la composición primitiva de la materia, de la estructura del código genético, del funcionamiento de los organismos, o la conformación de la sociedad global, de no haber tenido la humanidad la poderosa herramienta de los mecanismos de cómputo automático. Lamentablemente, esta capacidad también puede ser usada para el mal y causar en forma acelerada la destrucción de la civilización y hasta del mismo planeta. Por ello, la importancia de acompañar el desarrollo de esta tecnología con el correspondiente fortalecimiento de los aspectos éticos y morales.

Jorge Baralt-Torrijos
Profesor de la Universidad Simón Bolívar

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 2014, se celebraron conjuntamente la Segunda Conferencia Nacional de Informática, Computación y Sistemas (CoNCISa 2014) y la Segunda Escuela Venezolana de Informática (EVI 2014), en la Universidad Católica Andrés Bello, Caracas, Venezuela.

La edición de este segundo número de ReVeCom está dedicada a los mejores trabajos presentados en CoNCISa 2014. 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

Miembros del Comité Editorial

Tabla de Contenido

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 1-11, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Propuesta de un Modelo Educativo Utilizando el Paradigma de la Nube

Jose Aguilar[1], Karla Moreno[1], Domingo Hernandez[1], Junior Altamiranda[1], Maria Viloria[1]
aguilar@ula.ve, zumkar8@gmail.com, dhh@ula.ve, altamira@ula.ve, magaviq@gmail.com

[1] Departamento de Computación, Universidad de Los Andes, Mérida, Venezuela

Resumen: Se presenta un modelo educativo para utilizarlo en carreras universitarias del área de ciencias de la computación, basado en los conceptos de aprendizaje activo, objetos de aprendizajes y aprender haciendo. Se orienta a la formación de profesionales que respondan a las necesidades del país en TICs y los perfiles definidos en IEEE/ACM. Se busca conjugar actividades científicas, de reflexión y de desarrollo de tecnologías, haciendo que sobre los grupos de investigación gravite todo el que-hacer de la carrera; bajo la gestión del conocimiento y desarrollo de productos tecnológicos basado en "aprender-haciendo". Por otro lado, actualmente el aprendizaje se ha difundido por medios electrónicos, se propone incorporar estrategias de educación a distancia, asistida por computador, entre otras. Además, se toma en cuenta las necesidades del estudiante, proponiendo grafos curriculares flexibles, adaptables a necesidades emergentes, basados en un proceso de auto-formación. También, se propone una estructura organizacional inspirada en las ¨nubes¨; específicamente, el efecto, la densidad y la dinámica interna de las nubes, moviéndose de forma flexible con el viento (el acontecer mundial, la industria, los retos que viene enfrentando la educación superior). El modelo está dividido en tres sub modelos: Modelo Curricular, Modelo Didáctico y Modelo de Evaluación.

Palabras Clave: Modelos Educativos; Paradigmas de Aprendizaje; Mallas Curriculares; Educación a Distancia; Aprendizaje Combinado.

Abstract: An educational model is presented for use in university courses in the area of computer science, based on the concepts of active learning, learning objects and learning by doing. Is oriented to train professionals that respond to the country's needs in TICs and the profiles defined in IEEE/ACM. It seeks to conjugate scientific activities, reflection and development of technologies, making on research groups gravitate all that-make career; under the management of knowledge and development of technological products based on "learning by doing". On the other hand, currently learning has diffused through electronic means; it is proposed to incorporate strategies for distance education computer-assisted, among others. Moreover, taking into account the student's needs, offering flexible, adaptable curriculum graphs to emerging needs, based on a process of self-formation. An organizational structure inspired by the clouds also proposed; specifically, the effect, the density and the internal dynamics of the clouds, moving flexibly with the wind (world affairs, industry, the challenges facing higher education). The model is divided into three sub models: Model Curriculum, Teaching Model and Evaluation Model.

Keywords: Educational Models; Learning Paradigms; Curricula; Distance Education; Learning Combined.

I. Introducción

El ambiente académico en el área de las Tecnologías de Información de la Universidad de Los Andes se considera desfasado de la realidad mundial que se vive actualmente. Esta realidad, enmarcada en lo que se conoce como la "Edad del Conocimiento" [1], está delineada por el avance continuo y dinámico de la tecnología y de la información, en la que existe un abastecimiento inmenso y disponible de conocimientos.

Por otro lado, en general, los profesores universitarios, por cuanto sólo pueden enseñar lo que saben en el salón de clase tradicional, construyen muros virtuales al conocimiento. Estos muros virtuales definen el dominio de conocimiento de las asignaturas que imparten. Hoy en día, gracias a Internet, la información teórica y las explicaciones necesarias para que los estudiantes adquieran sus conocimientos están completamente a sus alcances, en las más variadas formas y perspectivas. Se habla de un acceso total y global a la información. Particularmente, Internet es una herramienta que permite demoler esas paredes que limitan el conocimiento creado en el salón de clase, dando acceso a un océano de conocimiento [1]. En la edad del conocimiento se requieren nuevas aptitudes, tales como [2]: Aprender a Aprender y Desaprender, Comunicación y Colaboración, Pensamiento Creativo, Cultura Tecnológica, Cultura Global del ejercicio profesional,

Desarrollo de liderazgo, Autogestión de la carrera profesional, etc. La educación superior tiene ante sí el reto de dejar de ser un modelo de educación centrado en el profesor para convertirse en un modelo centrado en el aprendiz, cuyo énfasis se encuentre en proveer a los estudiantes de las herramientas y recursos que les permitan responsabilizarse de su propio aprendizaje. Se hace fundamental pasar del modelo tradicional a un modelo de aprender-haciendo, en el cual el estudiante adquiera el conocimiento y lo aplique en la realidad local, nacional y mundial.

Por otra parte, dentro del ámbito nacional existe un creciente requerimiento de profesionales integrales formados en las áreas de computación, telecomunicaciones, tecnologías de información y afines (Estadísticas, etc.), con pertinencia social para los requerimientos que actualmente demanda nuestro país. Organismos internacionales como la IEEE (Institute of Electrical and Electronics Engineers) y la ACM *(*Association for Computing Machinery*)* definen cinco perfiles que cubren tales áreas [3][4]. Tal como se describe en [5], diferentes Universidades de Venezuela ofrecen carreras que al abarcar algunos aspectos de formación en áreas de tecnología, se convierten en híbridos de los perfiles definidos por los organismos mencionados. Sin embargo, no se cuenta con los mecanismos de acreditación y estandarización para estas carreras.

A partir de lo expresado anteriormente, surge la necesidad proponer la creación de un Programa Nacional en Informática alrededor de las Ciencias Computacionales, inspirado en los perfiles definidos por la IEEE y la ACM, enfocado en las necesidades en Tecnologías de Información y Comunicación (TIC) del país; un programa insertado en las dinámicas de desarrollo nacional en tecnologías de información y telecomunicaciones, apalancado en un modelo educativo basado en el paradigma de aprendizaje "aprender-haciendo", que además cuente con mecanismos flexibles para la actualización constante del corpus de conocimiento, asegurando así una efectiva respuesta al cambio continuo en estas áreas y a los requerimientos de formación que vayan surgiendo en el país.

Se piensa en un modelo educativo para la Formación en Informática que nos permita ir construyendo un espacio sobre el *que-hacer universitario* desde la reflexión sobre el *deber ser* universitario. Esa condición permite una reflexión-acción permanente sobre el modo de hacer universitario, tal que el programa se vaya ajustando a las realidades concretas de su entorno. Además, por las características de la propuesta se requerirán condiciones particulares (a nivel organizacional, de infraestructura, de talento humano, etc.) que soporten las actividades que se lleven a cabo en la misma, las cuales en las estructuras clásicas actuales de nuestra universidad no es posible conjugar. Algunas de ellas son: la dinámica del programa es centrada en los grupos de investigación, las mallas curriculares son flexibles, el modelo pedagógico es basado en un aprender haciendo cuyas necesidades de soporte a dicho modelo es emergente, es basado en un proceso de auto-formación del estudiante, etc.

A. *¿Por qué un Nuevo Modelo Educativo?*

- El proceso de enseñanza propuesto por la UNESCO [6] comprende aspectos como: aprender a conocer, aprender a hacer, aprender a convivir y aprender a ser. Es una tendencia mundial.
- El ambiente académico de nuestras universidades nacionales se ha ido desfasando de las realidades mundiales, enmarcadas en lo que algunos autores la han llamado la "Edad del Conocimiento".
- La educación superior tiene ante sí el reto de dejar de ser un modelo de educación centrado en el profesor para convertirse en un modelo centrado en el aprendiz.
- Gracias a Internet y a todas sus aplicaciones, la información teórica y las explicaciones necesarias para que los estudiantes adquieran sus conocimientos están completamente a sus alcances, en variadas formas y perspectivas [7].

Lo anterior nos conduce a proponer un modelo que utilices las nuevas herramientas que se ofrecen para interactuar sobre Internet. El enfoque del modelo educativo propuesto se centra en el paradigma de aprender-haciendo [8][9][10]. Este modelo se compone de tres sub modelos: el modelo Curricular, el modelo Didáctico y el modelo de Evaluación; estos modelos se van a presentar bajo una metáfora de nubes interconectadas, El modelo de evaluación no se profundiza en éste artículo, ya que está muy interrelacionado con la estrategia que lleve a cabo el tutor o profesor que esté como responsable del módulo que el aprendiz esté llevando a cabo.

II. MODELO FILOSÓFICO. PARADIGMA DE NUBES

La propuesta que se presenta en este documento se basa en la filosofía de conformación de nubes, las cuales responden a ciertas cualidades que llamaremos el "efecto nube", que hace que sus dinámicas internas se muevan con el viento (con el acontecer mundial y nacional, con la industria, etc.). Dichas cualidades dinamizan las actividades, el que-hacer, etc., que en el seno de las mismas van emergiendo, pero además, permiten que se vayan entrecruzando entre ellas, generando una poderosa sinergia educativa.

A. *Concepto de Nube*

El concepto de nube es usado como metáfora para indicar que es un espacio denso en el que libremente se puede navegar, se puede caminar, cuyas fronteras son flexibles y cuyos elementos constitutivos están interrelacionados y pueden aparecer o desaparecer en función del conocimiento requerido en un momento determinado. Específicamente, serán una *acumulación de cosas* en un ámbito dado (mallas curriculares, fuentes de conocimiento, etc.) representan l*a abundancia de ese algo* necesario para la realización de los objetivos perseguidos con esa nube. Así, estamos hablando más de "densidad" que de "estructura", permitiendo que esta última surja de las dinámicas internas que se den en cada nube, con cualidades como:

- No hay un camino único de recorrido en ellas.
- Nos podemos detener dentro de una de ellas sin perder de vista a la nube (grupo) ni al resto de nubes como un todo.
- Al navegar en una de ellas se deja cierta oportunidad al caos, al igual que haría una partícula en una nube. Este caos controlado se traduce en creatividad, innovación, conectividad entre conceptos e ideas, que en una malla tradicional no se podrían obtener.

De esta manera, el aprendizaje se basa en conectar a alumnos, docentes, mercado laboral, tecnólogos, fuentes de conocimiento, estrategias de aprendizaje, a través de formas creativas y productivas que hacen la generación de conocimiento más agradable, apasionante. El modelo educativo propuesto se basa en la interrelación de tres nubes como son: La nube de Formación que responde al Modelo curricular, Las nubes de Aprendizaje y Conocimientos que responden al Modelo Didáctico. En este orden de ideas, se profundiza en la descripción de cada nube en las secciones siguientes. En la Figura 1 se muestra la estructura basada en nubes, la cual debe ser soportada por una plataforma tecnológica.

Figura 1: Estructura Organizacional basada en Nubes

B. Nube de Paradigmas de Aprendizaje

En esta nube aparecerán los paradigmas, las estrategias y las herramientas de aprendizaje. Su objetivo es aportar los mecanismos de aprendizaje necesarios para el proceso de auto-formación del estudiante. Ella guiará las dinámicas de auto-formación, establecerá formas de acreditar cursos, posibilitará espacios de intercambio, de trabajo colaborativo, de construcción colectiva del conocimiento, entre otras cosas. Permitirá navegar entre el mundo presencial y virtual en el proceso de aprendizaje, garantizando elementos humanísticos en el ambiente de aprendizaje. En específico, desde esta nube se generan actividades que permitan la inclusión de ejes transversales vinculados al ámbito humanístico, a las artes, a la creatividad, a la innovación, etc., de tal manera de formar ingenieros capaces de reconocer su entorno social.

En específico, esta nube debe velar por la pertinencia social nacional de las obras a desarrollar por los estudiantes durante su formación, por el proceso reflexivo sobre el hecho científico tecnológico, entre otras cosas. Las dinámicas que se generen deben posibilitar procesos de soberanía tecnológica, procesos de apropiación social del conocimiento, procesos de inserción del hacer en las dinámicas nacionales, entre otras cosas.

Las herramientas y servicios que preste/contenga esta nube debe posibilitar alcanzar las metas vinculadas a la adquisición de forma autónoma de conocimiento, competencias, etc. Además, esta nube debe proveer todo el soporte en Tecnologías de Información para la gestión de cursos, conocimiento, proyectos, etc., así como salones Ágiles, que estimulen el trabajo y la organización en equipo, alrededor de obras. Algunas cualidades de este módulo son:

- Se inspira en paradigmas del "aprender haciendo", buscando una participación activa de quien se forma en la constitución de una obra, la cual puede ser artística, tecnológica, científica, etc.
- En ese sentido, todos las formas de aprendizaje que promuevan el aprender haciendo (aprendizaje activo, RAIS, aprendizaje ágil, aprendizaje combinado, etc.) son formas posibles para dinamizar esta nube.
- El trabajo colaborativo, el compartir conocimiento, el trabajo multidisciplinario, serán aspectos que deben enriquecer el proceso de enseñanza. Esas competencias deben ser promovidas por las diferentes estrategias, herramientas, etc. que constituyan esta nube.
- Se requiere de multitud de herramientas y aplicaciones de Internet para gestionar espacios compartidos, toma de decisiones comunes, asignación de tareas y responsabilidades, votaciones, gestión de grupos, seguimiento de obras/proyectos, entre otros.

El proceso de aprendizaje al que coadyuva esta nube se enriquece permanentemente, conectando a alumnos, docentes, tecnólogos, etc. en formas creativas y productivas que hacen que el caldo de cultivo del conocimiento se vuelva más agradable, más intenso y apasionante. Los rasgos específicos de ese proceso de aprendizaje son:

- Interconectado: es fundamentalmente el aprendizaje en red.
- Conversacional: La red propicia y fomenta las conversación entre personas.
- Distribuido: la transferencia de conocimiento no es jerárquica ni unidireccional. Además, no existen roles definidos de aprendiz y maestro.
- Colaborativo: co-creación de conocimiento a partir de las múltiples aportaciones y conversaciones entre los diversos nodos que colaboran unidos por un interés común. El conocimiento surge de la comunidad, es decir, es un conocimiento social fruto de la inteligencia colectiva. Dicho conocimiento es plasmado en las múltiples obras en realización.
- Continuo: La generación de conocimiento no es un proceso con principio y final, es continua, supone el abandono de la búsqueda de metas estáticas y definitivas. Así, es un estado "beta permanente".
- Abierto: el conocimiento que se genera debe ser abierto. El valor no reside en proteger y acumular sino en compartir, ya que es la manera de asegurar que éste se mantenga vivo y siga evolucionando.
- Emergente: suele producirse de forma espontánea. Es un aprendizaje auto-liderado, que se debe más a razones de curiosidad, motivación, interés personal / colectivo.
- Ubicuo: el aprendizaje pueda tener lugar prácticamente en cualquier momento y lugar, lo que facilita una mayor integración entre información y experiencia práctica.
- Personalizable: El aprendizaje se adapta el perfil del individuo en función del reconocimiento de sus cualidades/habilidades particulares (a partir de las cuales se establecen sus propios modos de aprendizaje, limites en los procesos de enseñanza, etc.), sin dejar de fomentar las dinámicas colaborativas que permiten la complementariedad para lograr objetivos comunitarios. La gestión del conocimiento individual/colectivo pasa a ser una responsabilidad del individuo y del entorno social.

- Colectivo, se produce conocimiento colectivo que es libre, que es patrimonio de la humanidad, que es responsabilidad de todos y del cual todos debemos velar.
- Multidisciplinario: Las áreas "puras" de conocimiento se rebelan insuficientes para abordar determinados temas, y requieren la integración de múltiples disciplinas (por ejemplo: Bioinformática mezcla Biología, Informática y Estadística). Así, el valor del conocimiento puro cae ante el valor de la diversidad. En este aspecto son fundamentales los ejes transversales, así como la posibilidad de tomar créditos de otras carreras.

Esta nube debe garantizar que estas cualidades del proceso de aprendizaje se den a través de los paradigmas, estrategias y herramientas que ella provea [11][12][13]. Debe permitir un aprendizaje activo (volar un aeroplano, ya sea en un simulador o realmente), no pasivo (escuchar a los instructores de vuelo o leer los libros de instrucciones para volar un avión), a través de estrategias instruccionales que conllevan a los estudiantes a hacer y pensar sobre lo que hacen. Algunas ideas al respecto que en esta nube se deben promover son:

- Actividades que conlleven a un hacer, más que a escuchar pasivamente en un curso, una conferencia.
- Estrategias que incluyan ejercicios en los que los estudiantes apliquen el conocimiento a situaciones de la vida real, problemas concretos, etc.
- Mecanismos que conlleven a un *Aprendizaje Significativo*, aprendizaje en el que el estudiante relaciona el conocimiento con otros conocimientos, con otras experiencias, o con actividades o hechos de su cotidianidad.
- Dinámicas que coadyuven a un *Aprendizaje Relevante*, aprendizaje que provoca que el estudiante reestructure sus anteriores esquemas mentales. Así, el nuevo contenido asimilado permite adquirir nuevas habilidades más complejas.

C. Nube de Fuentes de Conocimiento

Está constituido por todo el conocimiento esparcido a través del mundo, en todas sus formas, desde todas las fuentes posibles. Su objetivo es posibilitar el mayor acceso al conocimiento disponible a nivel mundial, pero desde una mirada crítica al mismo. Las metodologías, herramientas y técnicas que conforman esta nube deben posibilitar el acceso crítico a ese conocimiento, según las dinámicas/actividades establecidas en las otras nubes [14][15][16]. Así, no estamos hablando de un acceso pasivo, neutro, al conocimiento, sino crítico, visto, además, desde ese proceso de auto-formación según la dinámica curricular establecida en la *nube de formación*, y desde el proceso de aprendizaje dictado por la *nube de aprendizaje*.

En esta nube, los aspectos humanísticos y sociales de formación juegan un rol fundamental, ya que son los que permitirán una aproximación al conocimiento con el ojo crítico del papel de la ciencia y tecnología en la sociedad. En ese sentido, el proceso de adquisición de conocimiento será desde el paradigma de *apropiación del conocimiento,* el cual está conformado por los siguientes aspectos, fundamentales a develar por las estrategias establecidas en esta nube:

- La obtención de todo el conocimiento de base detrás de un área cognitiva, un producto tecnológico, una teoría; además del conocimiento necesario para usarlo, para explotarlo, etc. en una obra.
- El análisis del contexto, la realidad social, etc. en el cual se realizó el desarrollo de ese conocimiento, esa obra tecnológica, así como su impacto, su influencia, etc. a través del tiempo en la sociedad.

Así, no es solamente acceder al conocimiento o adquirirlo pasivamente, sino que como ciudadano de este mundo se debe ser capaz de analizar todos los aspectos que abarcan dicho conocimiento. Esta nube requerirá de herramientas para organizar, analizar, explotar, criticar, dicho conocimiento (es decir, de gestión del conocimiento). Esas herramientas deben posibilitar lo establecido en la nube de aprendizaje (paradigmas, estrategias, etc.) y en la nube de formación (auto-formación, etc.). Esto permitirá hacer énfasis en la adquisición de conocimientos, en las destrezas para usarlo, y reflexionar sobre él para reconocerlo en el contexto social donde este inmerso.

Algunos aspectos resaltantes a nivel de las posibles fuentes de conocimiento son:

- Deben ser en lo posible auto-contenidas.
- Deben estar vinculadas a los módulos establecidos en la nube de formación.
- Deben ser posibles validar, verificar, dichas fuentes.

Algunos ejemplos de fuentes de conocimientos son:

- Formas virtuales: cursos en línea en la web, objetos de aprendizaje, programas de formación a distancia, talleres, etc.
- Cursos presenciales: cortos, escuelas de formación, cursos clásicos, etc.
- Laboratorios y centros de investigación: en estos casos se requerirán de mecanismos de validación de las pasantías realizadas en dichos espacios.

D. Nube de Formación

Esta nube es navegada por el estudiante. Es un proceso en el que se explota todo lo que ofrece las nubes de aprendizaje y de conocimiento, para que de forma autónoma construya su formación. El estudiante va construyendo su propia malla curricular, y va navegando a través de ella (ver Figura 2). La nube debe permitir esta navegación de una forma directa y natural. Esa formación se da a través de un construir, de un hacer, en el cual se va plasmando todo ese conocimiento que va adquiriendo el aprendiz, desde un modelo pedagógico del tipo "aprender - haciendo".

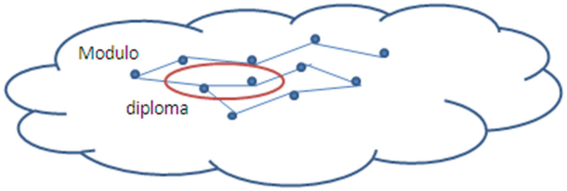


Figura 2: Forma de Navegación en la Nube de Formación

En general, algunas de las cualidades que caracterizan esta nube son:

- Está conformado por un conjunto de módulos de formación, que pueden ser agrupados en una materia, pero pueden ser tomados/estudiados individualmente. De esta manera, ellos son auto-contenidos. Cada módulo va aportando *créditos de conocimiento* en vías a la obtención de algún tipo de diploma, y tiene un conjunto de pre-requisitos claramente definidos.
- Los posibles diplomas a obtener en un recorrido de formación son caracterizados por un número mínimo de créditos que se van obteniendo en áreas específicas. Los créditos son de dos tipos: créditos de conocimiento y créditos tecnológicos/científicos, más adelante ahondamos en detalles sobre los mismos. Cada diploma (técnico medio, técnico superior, ingenieril, especialización, maestría, etc.) tendrá claramente definido los créditos a obtener en áreas específicas para optar a ese diploma. Las áreas específicas estarán inspiradas en las definidas por IEEE/ACM curricula [3], con otras áreas transversales humanísticas, sociales, apropiación tecnológica, etc., vinculadas al proceso de soberanía tecnológica y al principio de no neutralidad del conocimiento.
- Se proponen caminos a seguir (perfiles de formación) para lograr competencias específicas (por ejemplo, en Ingeniería de Software, en Ciencias de la Computación, maestría en Bioinformática, etc.), pero el estudiante es autónomo y va guiando su propio proceso de autoformación, pudiendo llevar a cabo una formación hibrida. Los diplomas serán únicos (ingeniero, licenciado, técnico medio, máster, etc.), con un certificado anexo indicando el perfil de formación (por ejemplo, si se siguió el camino propuesto en Ingeniería de Software será en eso el certificado específico). Como se expresó anteriormente, los perfiles de formación estarán inspirados en los definidos por IEEE/ACM curricula (Ingeniería de Software, Ingeniería del Computador, etc.), permitiendo que emerjan otros según las necesidades nacionales (por ejemplo, Computación Petrolera, etc.), y el acontecer mundial en esta área (por ejemplo, Maestría en Bioinformática, en Ambientes Inteligentes, etc).
- Cada estudiante explotará los recursos provistos por las otras nubes en ese navegar por esta nube, y se le irá certificando lo que va estudiando. Además, debe ir construyendo en su camino obras (se entenderá por obra al desarrollo de un proyecto, producto, objeto, para lo cual utiliza todo el conocimiento adquirido en los créditos académicos), en las cuales irá plasmando el conocimiento adquirido, mostrando las virtudes obtenidas en su proceso de auto-formación. Desde ese hacer se le acreditan los respectivos *créditos científicos/tecnológicos*.
- El paradigma de base es la inclusión de todos, entendiéndose por eso que cualquiera puede entrar a la nube de formación. Quien entre a la nube de formación debe estar consciente de los pre-requisitos exigidos por los módulos de las materias. Esos 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. En este último caso, es responsabilidad del que aprende de adquirirlos. Según el nivel de avance que tenga un individuo en la obtención de créditos, etc. se alcanza uno u otro título.
- Las formas de acreditaciones del conocimiento son diversas, pudiéndose usar las tradicionales de exámenes y trabajos, como aquellas que emerjan desde la nube de aprendizaje. En cuanto a los créditos tecnológicos, reflejados en la obra que este ejecutando quien se esté formando, debe reconocerse en esa obra ese conocimiento que ha adquirido, desde los principios de "aprender haciendo". Para ello, esa obra deberá defenderse y mostrarse ante un jurado público.
- Debe ser lo más flexible posible, posibilitándose la incorporación de nuevos módulos (actualización curricular), caminos posibles a recorrer (perfiles de formación), etc. Así, estamos hablando de mallas curriculares flexibles (varios perfiles), donde los estudiantes pueden cumplir créditos a lo largo de uno a más de ellos, o pueden cumplir una cantidad de créditos mínimos y graduarse, sin necesariamente estar asociados a un perfil en particular. Además, los estudiantes no deben esperar para graduarse en 3, 4 o 5 años, ellos acumulan créditos y pueden salir a buscar trabajo en cualquier momento, con los créditos que tienen hasta los momentos.
- Debe ser posible integrar áreas distintas que coadyuven a su formación profesional integral (por ejemplo, es permisible que un estudiante tome créditos de otras carreras, como el caso de neurofisiología, fundamental para computación neuronal), todo con la visión de fomentar una educación interdisciplinaria en la que los ingenieros puedan manejar conocimientos externos a los asociados con el perfil principal de la carrera

En cuanto a los créditos, hemos hablado de dos tipos (ver Figura 3), los cuales son:

- *Créditos académicos o de conocimiento:* corresponden a los créditos que se obtienen por adquisición de conocimiento a través de cursos, talleres, experiencias, pasantías, etc. Esos créditos se corresponden a módulos específicos de conocimiento, que pueden formar una materia o ser vistos individualmente. En general, todo diploma tendrá definido tres tipos de créditos académicos: créditos de conocimiento de base, créditos de conocimientos intermedios y créditos de conocimientos avanzados (especializados). La forma de adquisición de conocimiento puede ser siguiendo estrategias de aprendizaje presenciales, a distancia, etc. y para la obtención de los mismos podrán regir lógicas distintas, usarse fuentes de conocimiento diversas, según lo establecido en la nube de conocimiento.
- *Créditos tecnológicos/científicos* son los derivados de la evaluación de las obras que venga realizando el estudiante, sobre la cual debe plasmar ese conocimiento adquirido. Estos créditos son la base del paradigma "aprender haciendo", necesarios para garantizar el proceso de retroalimentación del modelo pedagógico a implantar. En ese sentido, la obtención de esos créditos debe reflejar el proceso de acompañamiento que se haga localmente en el obrar del estudiante, los procesos de enriquecimiento del conocimiento local por el conocimiento usado en la obra, las competencias adquiridas por el estudiante, la capacidad de integración de diversos conocimiento

(multidisciplinariedad), el reconocimiento de lo virtuoso de la obra, entre otras cosas. Para la obtención de esos créditos jugarán un papel fundamental los grupos de investigación de la Carrera y el entorno. Serán en el seno de ellos, y en el marco de proyectos con pertinencia nacional, en el cual su impacto social y humanístico debe estar reflejado. En general, los créditos podrán ser obtenidos a través de seminarios, artículos, y por supuesto, de manera obligatoria en la misma obra desarrollada. Es bueno aclarar que cada uno de los aspectos señalados antes (artículos, producto tecnológico, etc.) deben reflejar en su contenido ese conocimiento adquirido a través de los créditos académicos. Estos créditos también serán de varios tipos: créditos iniciales de definición de la obra, créditos del proceso de concepción de la obra, y créditos de consumación de la obra. Los mismos serán obtenidos durante el proceso de formación del estudiante.

- *Créditos docentes:* son los créditos que se les acreditan a los estudiantes que hayan tomados cursos similares a los de un perfil dado en el pasado. Para poderlos validar/acreditar, el estudiante podrá dictar un seminario, introducirlos en sus obras, etc.

En cuanto a los programas de estudio de los perfiles de formación, deben poseer las siguientes cualidades:

- Deben reflejar la integridad del conocimiento de la disciplina del área de Ciencias Computacionales caracterizada por ellos.
- Deben responder rápidamente a los cambios tecnológicos, a las necesidades nacionales, etc.
- Deben caracterizar claramente las capacidades/ competencias específicas que los estudiantes obtendrán.
- Deben promover la innovación y la creatividad.
- Deben proporcionar a los estudiantes que culminaron, una experiencia de aplicación de sus habilidades y conocimientos para resolver problemas reales (a través de la obra) de manera colaborativa.

El docente en este caso asume el *rol de acompañante, e*s orientador, es investigador, es capaz de producir y de guiar el desarrollo de una obra. Los estudiantes *asumen algunas responsabilidades* acerca de su propio aprendizaje, planteando iniciativas, propuestas de tareas, etc. (establece un contrato de aprendizaje). Además, es una *educación en el cual el docente* involucra su hacer en el desarrollo de obras nacionales.

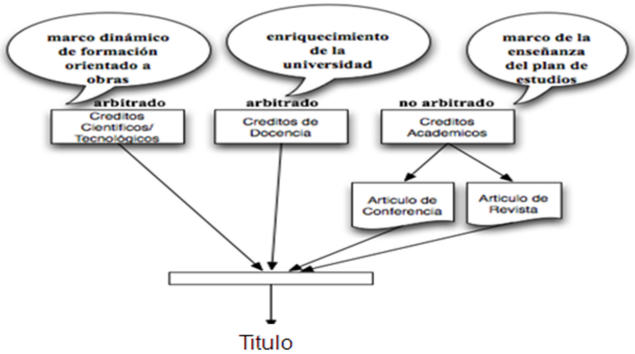


Figura 3: Tipos de Crédito en la Formación del Conocimiento de Estudiante

III. Propuesta para la Estructura Organizativa

El paradigma de nubes nos obliga a pensar en un conjunto de unidades que coadyuven, que posibiliten, las dinámicas que están detrás de cada nube. De esta manera, se requerirá de una estructura organizacional que posibilite el hacer de ellas. Algunos posibles componentes de esa estructura organizacional se muestran en la Figura 4 (el nombre de unidad es para definir a cualquier dependencia que se establezca como elemento atómico organizacional).

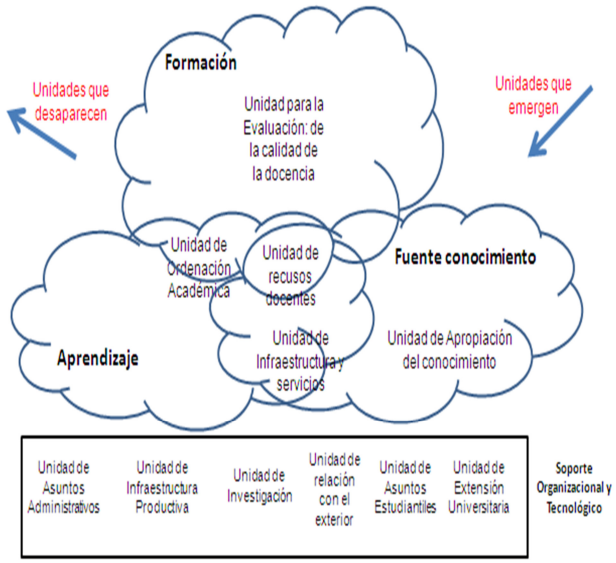


Figura 4: Estructura Organizacional de la Carrera

- Unidad de Ordenación Académica: responsable de la reflexiones y definiciones de los diplomas posibles a impartir (carreras técnicas, docencia de pregrado, estudios de Postgrado, y en general todos los diplomas que se impartan). Además, es responsable de las tareas de Acreditación y Evaluación de Carreras, definición de nuevas titulaciones, etc.
- Unidad de recursos docentes: estudia los diferentes paradigmas: Pedagógicos, de Accesibilidad del conocimiento, etc. A partir de ellos, define políticas a ser implementadas en la nube de aprendizaje.
- Unidad de Investigación: coordina las actividades de los grupos de investigación, garantiza la sinergia entre ellos, visualiza los proyectos donde las obras de los estudiantes se deben dar, etc.
- Unidad para la Evaluación: de la calidad de la docencia, del profesorado, del quehacer universitario, etc.: responsable de todas las actividades que permitan estudiar, valorar, el hacer de la facultad. Así, la definición de métricas, la implementación de mecanismos de medición, la discusión de resultados de procesos de evaluación, la definición de políticas de acción de mejoramiento académico, entre otras cosas, están entre sus actividades.
- Unidad de Asuntos Administrativos: responsables de las áreas de economía, finanzas, y en general, de todos los aspectos de Gestión Institucional.
- Unidad de Extensión Universitaria: responsable de la relación con el entorno social. Debe establecer

mecanismos de difusión del quehacer de la carrera, así como de relación con la Comunidad.

- Unidad de Asuntos Estudiantiles: responsable de velar por el hacer de los estudiantes. En ese sentido, debe coadyuvar a la armonía de la vida estudiantil con los principios de aprendizaje activo.
- Unidad de Infraestructura y servicios: responsable de la gestión de los Centros de Cálculo, de las herramientas y recursos informáticos docentes, de las redes, etc. Ella recoge los requerimientos de las otras unidades (por ejemplo, de la unidad de recursos docentes), y desde un hacer de investigación-acción provee las herramientas demandadas.
- Unidad de relación con el exterior: su responsabilidad es establecer convenios de trabajo, programas de cooperación, de relacionarse con los graduados, de las Relaciones Internacionales, etc. Además, del establecimiento, debe garantizar el hacer que se derive de esos convenios (que no sea letra muerta).
- Unidad de Infraestructura Productiva: es la responsable de definir, gestionar, articular, proyectos especiales, proyectos industriales o sociales, etc. con los diferentes actores de la Carrera. Además, debe promover, articular, la participación de la facultad a partir de su hacer con los proyectos de su entorno social.
- Unidad de Apropiación del Conocimiento: es la responsable de los cursos transversales, que a su vez lleven implícito la capacidad de criticar, reflexionar, contextualizar, ese conocimiento. En ese sentido, debe dinamizar todos los procesos de articulación de las reflexiones sobre la ciencia, tecnología y sociedad (cursos, conversatorios, herramientas de gestión de conocimiento a usar, etc.), Su principal función es cultivar el área de Informática Social.

IV. Modelo Didáctico

El modelo didáctico permite formar individuos con conocimientos, valores, destrezas, actitudes, y motivaciones que deben poner en práctica en cualquier ámbito de su vida cotidiana personal, social laboral. Para ello, se requiere de una formación integral que comprenda:

- **Formación intelectual**, que fomenta el pensamiento lógico, crítico y creativo; propicia una actitud de aprendizaje permanente que permite la autoformación.
- **Formación humana**, que aborda al sujeto en sus dimensiones emocional, espiritual y corporal.
- **Formación social**, que fortalece valores y actitudes para relacionarse y convivir con otros; propicia la sensibilización, el reconocimiento y la correcta ubicación de las diversas problemáticas sociales.
- **Formación profesional**, orientada al saber hacer de la profesión; incluye tanto una ética de la disciplina como nuevos saberes, para la inserción de los egresados en el actual mundo del trabajo.

El proceso educativo centrado en el aprendizaje se sustenta en cinco pilares:

- **Aprender a conocer** (Cognoscitivo), es decir, adquirir los instrumentos de la comprensión;
- **Aprender a hacer** (Actuacional), para poder influir sobre el propio entorno;
- **Aprender a convivir** (Valores), para participar y cooperar con los otros en todas las actividades humanas;
- **Aprender a ser** (Valores), un proceso fundamental que recoge elementos de los tres anteriores
- **Aprender a emprender** (Valores), para ser capaz de realizar retos científicos/tecnológicos desde su formación

Integrar los cinco tipos de aprendizajes en el proceso educativo conlleva a la consecución de un aprendizaje significativo en los estudiantes

Por otro lado, como el modelo educativo está centrado en el aprendizaje del formado, esto implica:

- Una adaptación permanente del paradigma pedagógico, según las necesidades del individuo en cada momento.
- Permitir al ser humano realizar su propia construcción de saberes significativos, así como el descubrimiento y desarrollo de sus potencialidades.
- Procesos de atención, retención, reproducción y reforzamiento, acordes con los estudiantes del mundo de hoy.

V. Modelo Curricular

Los profesionales egresados en esta carrera obtendrán el Título de Ingeniero en Computación e Informática.

A. Perfiles

- Ingeniería del Software
- Ingeniería del Computador
- Ciencias de la Computación
- Sistemas de Información

Así mismo, en esta carrera se prevé el otorgamiento del título de Técnico Superior por cada mención, denominado: Técnico Universitario en Computación e Informática, a los estudiantes que cursen y aprueben todas las unidades curriculares desde el primero hasta el noveno trimestre.

Adicionalmente, se plantea el otorgamiento por parte de la Universidad de Los Andes de una certificación programador, al aprobar las materias vinculadas al área de programación.

Estas salidas profesionales intermedias pretenden dar mayores oportunidades al estudiante para su incorporación temprana al mercado laboral y garantizar la continuidad de sus estudios hasta la culminación de la carrera.

Por último al egresado como Ingeniero en Computación e Informática tiene la oportunidad de cursar estudios de postgrado a nivel de:

- Magíster
- Doctor

B. Plan de Estudios

El programa se constituye de un plan de estudio donde el estudiante debe cubrir tres tipos de créditos:

- Créditos de conocimiento / académicos.
- Créditos docentes.

- Créditos científicos/tecnológicos

Para ello, el estudiante se va formando en tres ejes: en su formación profesional (Grafo Curricular), su formación humanística y social, y otro propio al paradigma aprender haciendo (ver Figura 5).

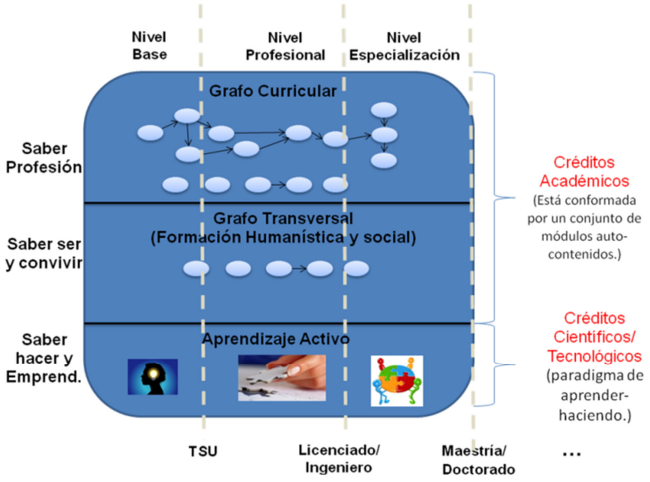


Figura 5: Modelo Curricular

Ese último eje conlleva actividades donde el estudiante debe aplicar su conocimiento adquirido, con créditos definidos para las mismas en el modelo curricular, como:

- La participación en proyectos de investigación (obras), ya sea como co-investigadores o investigadores noveles A partir de las dinámicas del tercer eje se determinan los mecanismos para vincular a los estudiantes a los procesos de investigación que los grupos de investigadores que soportan la carrera estén realizando.
- La participación en proyectos de desarrollo tecnológico (obras) que los grupos de investigadores que soportan la carrera estén desarrollando. Al igual que antes, las dinámicas del tercer eje deben tener mecanismos para vincular a los estudiantes a dicho procesos de desarrollo.
- Creación, desarrollo y mantenimiento de semilleros de investigación y desarrollo tecnológico. Esto con la finalidad de posibilitar el emprendimiento de los estudiantes, soportados por el plantel de investigación y desarrollo tecnológico que soporta la carrera, de nuevas ideas, áreas de investigación o nuevas tecnologías. Esto facilita que los estudiantes formulen y ejecuten sus propios proyectos de investigación y desarrollo tecnológico. En el caso de los cursos a distancias, los mismos serán reconocidos por el sistema de créditos de la carrera si están dentro de los cursos acreditados por la misma, y son plasmados en las obras en desarrollo y socializados a través de seminarios y talleres impartidos.
- A través de unidades de aprendizaje como el seminario de investigación, proyectos de grado, talleres, pasantías de investigación y desarrollo, cursos a distancias, donde se logra una complementación mutua entre la docencia y la investigación. Todas esas dinámicas se reflejaran en las obras que vaya desarrollando el estudiante, y tendrán créditos específicos en el modelo curricular.
- La implicación de los estudiantes en proyectos de extensión, tal como el cumplimento del Servicio Comunitario, o la participación en proyectos de investigación o desarrollo tecnológico con vinculaciones con el entorno social, que le permitan la apreciación de los contextos sociales, culturales, ambientales, humanísticos y éticos relacionados con la carrera.

El eje del saber hacer y emprender fungirá como un elemento integrador de conocimientos a través de las distintas actividades señaladas anteriormente, desde el "aprender haciendo", en las que se conectan, contextualizan e integran los conocimientos de las distintas disciplinas. Esto permitirá impulsar una visión más flexible e integrada del enfoque educativo que apunte a la interconexión entre disciplinas, promoviendo la inter y transdisciplinariedad.

C. Grafo Curricular

En este eje se forma el profesional en sus áreas de saberes, un ejemplo del contenido de los módulos auto-contenidos es presentado en las Figuras 6 y 7.

Módulos	Código	Temas	H	Pre-requisitos	Perfiles	Nivel
Fundamentos de Programación 1	PF1 32	Lógica de programación	4		Todos	NB
		Desarrollo de programas	2			
		Construcciones fundamentales.	8			
		Algoritmos y Resolución de Problemas.	6			
		Estructuras de Datos Básicas.	6			
		Subprogramas	6			
Fundamentos de Programación 2	PF2 42	Estructuras de datos avanzadas: archivos, listas, colas, arboles. etc.	20	PF1	Todos	NB
		Recursividad.	6			
		Programación Orientada a Objetos.	16			
Fundamentos de Programación 3	PF3 16	Programación Orientada a Eventos.	8	PF2	Todos	
		Programación Funcional.	8			
Matemáticas Discretas 1	MD1 6	Funciones, Relaciones y Conjuntos.	6		Todos	NB
Matemáticas Discretas 2	MD2 10	Lógica Básica.	10	MD1	Todos	NB
Matemáticas Discretas 3	MD3 6	Grafos y Árboles.	6	MD1	Todos	NB
Matemáticas Discretas 4	MD4 16	Probabilidad Discreta.	10	MD1	Todos	NB
		Muestreo y Estadística Descriptiva	6			
Algoritmos y Complejidad 1	AL1 28	Análisis Básico de Algoritmos.	4	PF2	Todos	NB
		Ordenamiento	8			
		Algoritmos Fundamentales de grafos.	12			
		Algoritmos matemáticos	4			

Figura 6: Modelo Curricular

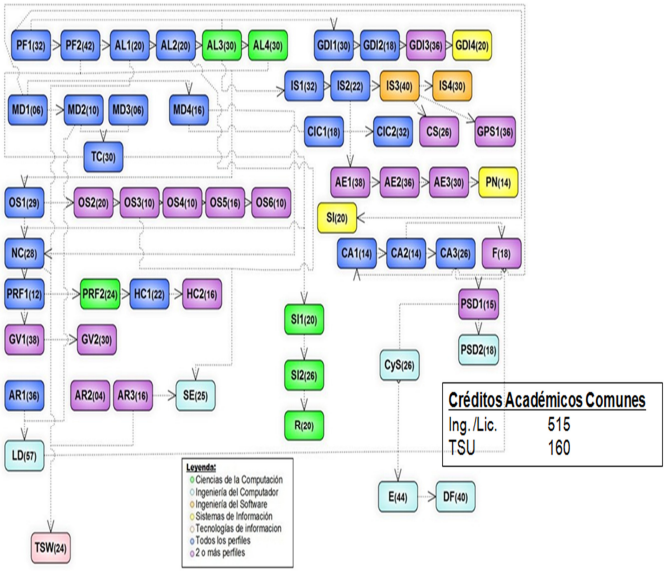


Figura 7: Grafo Flexible de todos los Perfiles

La Figura 6 muestra los módulos, cuales perfiles deben tomarlos, los temas que cubre, y las prelaciones que tiene. Un aspecto no reflejado alli, es que al terminar los módulos que

corresponden a un dominio de conocimiento (por ejemplo, los módulos PFi o MDi en la Figura 6), el estudiante debe tomar un taller en el que básicamente lo que hará será la obra donde aplicará el conocimiento obtenido en esos módulos. Es en ese taller donde van adquiriendo los estudiantes los créditos cuentificos/tecnológicos (es así como se materializa el eje aprendizaje activo, aprender haciendo). En ese taller trabajará con otros, se unira a grupos de investigación de la carrera, continuará proyectos transversales a varios talleres, etc.

D. Formación Transversal (Humanistica y Social)

Dependiendo del nivel las obligatorias son diferentes. Los TSU deben tener 42 créditos y los demás 100 créditos. En el caso de los TSU las materias definidas son obligatorias, en los otros hay algunas obligatorias, y para cubrir el resto de los créditos puede tomar otro grupo que son electivas (ver Figura 8 y Figura 9).

Módulos	Temas	H
Temas Profesionales y Sociales 1	Propiedad intelectual:	4
	Política Pública en tecnologías de información	4
	Temas Económicos en Computación	4
Temas Profesionales y Sociales 2	Responsabilidades Éticas y Profesionales	4
	Privacidad y libertades Civiles	2
	Crimen Computacional	4
Innovación y Nuevas Tecnologías	Tecnologías que han dado forma al mundo electrónico	4
	Procesos de innovación	4
	Importancia estratégica de la Web como plataforma	2
	Herramientas Web 2.0 y Web 3.0	4
	Entornos colaborativos virtuales.	2
	Gestión de conocimiento	2
Apropiación tecnológica	Introducción a la idea de desarrollo tecnológico.	2
	Procesos de apropiación tecnológica.	4
	La Universidad y la apropiación tecnológica.	3
	Diseño de procesos de transferencia en Tecnologías de Información.	3
Sociedad y Tecnologías de Información	Filosofía tecnológica.	4
	Tecnologías de información y proyectos sociales.	2
	Planificación y metodología de proyectos sociales	4
	Sociedad, conocimiento y tecnología.	2
	Informática y Desarrollo Comunitario	4
	Soluciones Informáticas para la Gestión Social	4
	Gobierno electrónico	2

Figura 8: Formación Transversal

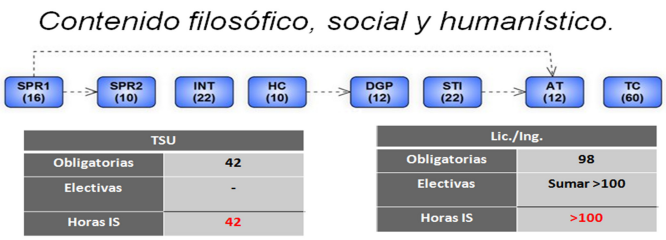


Figura 9: Grafo Transversal

E. Objetivo General del Programa de Formación

Formar a profesionales en las diferentes áreas vinculadas a la computación, con diferentes niveles de formación, en función de las dinámicas de cambio en los perfiles requeridos para profesionales de dichas áreas.

F. Objetivos Específicos del Programa de Formación

- Establecer un modelo curricular que permita la formación flexible actualizada en algunos de los perfiles definidos a nivel mundial en el ámbito computacional.
- Permitir la formación a diferentes niveles profesionales, con entradas y salidas intermedias, en función del deseo del alumno.
- Establecer mecanismo de formación inspirados en la idea de aprender haciendo, para garantizar la apropiación tecnológica del conocimiento.
- Fortalecer las competencias sociales del aprendiz, a partir de una formación a través de la construcción de una obra o producto.

G. Competencias Generales del Egresado

Toda la formación que se debe obtener con este programa debe garantizar:

- Entender y aplicar aspectos éticos, legales y sociales, relacionados con la práctica profesional.
- Comunicarse efectivamente con diversas audiencias.
- Reconocer la necesidad de un desarrollo profesional continuo.
- Comprender la necesidad de un desarrollo nacional científico-tecnológico soberano.
- Entender la no neutralidad del conocimiento, su papel emancipador.

H. Lineas de Investigación

Como uno de los elementos fundamentales para el aprendizaje activo es tener espacios donde se puedan desarrollar las obras, es fundamental tener presente los centros de gestión de conocimiento que estén trabajando en los diferentes dominios de la carrera. Los mismos pueden ser laboratorios, centros de investigación, etc., y si no existiesen en todas las áreas, sería necesario establecer las formas en que los estudiantes cubran todos los talleres (por pasantías, etc.).

Las actividades de dichos espacios de investigación es de vital importancia para nuestra propuesta, ya que es alrededor de ellos que girará la dinámica de "aprender haciendo". Particularmente, todos los créditos científicos/tecnológicos por parte de los estudiantes serán adquiridos a través de los trabajos realizados en esos espacios. Es decir, los talleres deberán ser realizados en los espacios de los grupos de investigación. Además, la carga docente de los investigadores, en la medida de lo posible, estará vinculada a sus actividades de investigación.

Por ejemplo, en el caso concreto de de la Universidad de Los Andes, actualmente el Departamento de Computación basa sus líneas de investigación en los trabajos que hacen los diferentes grupos, laboratorios y centros de investigación que hacen vida en él. El Departamento de Computación cuenta con los siguientes grupos, laboratorios y centros de investigación, cuyas líneas de investigación son:

- *LASDAI (Laboratorio de Sistemas Discretos, Automatización e Integración):* Robótica (Robots Industriales, Robots Móviles, Microbots, Arquitecturas de Robots, Simulación, Cooperación de Múltiples Robots, Inteligencia colectiva, Interacción humano-robot, Robot-humano, Planificación y navegación, Incertidumbre), Automatización Industrial (*Almacenes Automatizados, Sistemas a Eventos Discretos, Informática Industrial, Automatización Integral, Instrumentación), Visión por Computador (Control de calidad en sistemas de manufactura, Reconocimiento de objetos, Modelado del*

Entorno y Navegación de Robots Móviles, Reconocimiento de Rostros, emociones y expresiones).

- *GIDYC* **(**Grupo de Ingeniería de Datos y Conocimiento**):** Ingeniería de Software, Ingeniería de Métodos, Arquitecturas Empresariales, Sistemas de Información, Bases de Datos, Modelado de Datos, Computación Gráfica, Ambientes Virtuales, Sistemas de Flujo de Trabajo, Lenguajes Formales y Compiladores, Interacción Humano/Computadora, Sistemas Multiagentes, Ontologías, Web Semántica.
- *CEMISID (*Centro Estudios Microelectrónica Sistemas Distribuidos): Inteligencia Artificial (Técnicas Inteligentes (Redes Neuronales Artificiales, Computación evolutiva, Lógica Difusa, Enjambres de Partículas, Colonias Artificiales de Insectos, etc.), Sistemas Multiagentes, Web Semántica, Ontologías, Bio-informática), Automatización Industrial (Gerencia Integrada de Producción , Diagnosticabilidad y Supervisión Inteligente, Sistemas Multiagentes Tiempo Real), Computación autonómica y Aprendizaje (Sistemas de toma de decisión autonómica, Paradigmas basada en lógica temporal y redes bayesianas, Enfoques de aprendizaje para supervisar aplicaciones autónomas, Metodología para desarrollar aplicaciones autónomas basadas en el paradigma ODA, etc.), Ambientes inteligentes (Middleware reflexivos, Avatares, etc.), Paralelismo (Técnicas Especulativas para Multi-Hebras, Optimización de Asignación de tareas en plataformas multi-core, etc.).
- RESIDE (Grupo de Redes y Sistemas Distribuidos): Redes de computadores, Análisis por simulación y en escenarios reales del performance de protocolos de redes TCP/IP. Movilidad en Redes WiFi 802.11. Protocolos de Transporte en redes inalámbricas (802.11, 802.16) y alambradas. Control de Congestión en Redes TCP/IP. Transporte de datos y enrutamiento en redes ópticas. Sistemas de tiempo real. Computación de Alto Rendimiento. Computación Grid. Computación Paralela. Sistemas embebidos basados en Linux. Seguridad informática. Certificación electrónica e infraestructura de clave pública.

Además, otros centros de investigación que potencialmente estarán involucrados a esta propuesta son:

- LABSIULA (Laboratorio de Investigación de Sistemas Inteligentes).
- *CESIMO* (Centro de Investigación y Proyectos en Simulación y Modelos): Técnicas avanzadas en simulación y modelado, Lenguajes y plataformas de simulación y modelado**,** Modelado y simulación de sistemas ambientales, Modelado y simulación de sistemas socio económicos.
- SUMA (Sistema Unificado de Micro-computación Aplicada).

Vemos así, que en este caso el Dpto. de Computación estaría en capacidad de emprender el reto de creación de una carrera con las características planteadas.

VI. CONCLUSIONES

Este trabajo ha presentado una propuesta para la creación de una Carrera Experimental en el área de ciencias de la computación e informática, basada en una estructura curricular y organizacional que se inspira en el paradigma "aprender haciendo". Esta Carrera ofrece a los estudiantes la oportunidad de descubrir y construir su propio conocimiento a través de la ejecución de productos (obras), a través de las cuales les permite desarrollar actitudes positivas de creatividad y emprendimiento, y generar sentimientos de realización y satisfacción por lo ejecutado y logrado.

A través de las nubes de auto-formación y de fuentes de conocimiento se estimula a los estudiantes a sumar esfuerzos, capacidades y competencias, para elaborar producto a través de los cuales adquirirán elementos cognitivos adicionales. A su vez, esta estrategia permite darle mayor solidez a la relación estudiante-profesor. El profesor deja de ser un ente trasmisor de información, y se convierte en un compañero de viaje en ese largo camino de adquisición de conocimiento, siendo responsable, al igual que sus estudiantes, de la ejecución de los productos. Este factor diferencial entre un profesor trasmisor de información y un profesor responsable de la elaboración de un producto (obra), resulta en una transformación de la actividad de enseñanza-aprendizaje clásica.

Igualmente se puede señalar que de las dinámicas generadas por las nubes, se produce un incremento de interés en la búsqueda de conocimientos, reforzando la disciplina, el trabajo en equipo, la concreción de metas, el análisis y síntesis de situaciones reales, entre otras cosas. Este modelo de Carrera genera conductas profesionales que se resumen en un "hacer" constante, bajo responsabilidades claras y precisas, de un producto para satisfacer una necesidad individual o social. Así, la estructura de la Carrera genera una dinámica de aprendizaje que aumenta la motivación, brinda oportunidades de colaboración para construir conocimientos, aumenta las habilidades para la solución de problemas, consolida la relación estudiante-profesor, estimula el emprendimiento y la creatividad, empuja hacia el trabajo efectivo y de calidad, propicia hábitos de responsabilidad y disciplina, preparando a los estudiantes para su vida profesional dentro del contexto de la edad del conocimiento.

AGRADECIMIENTOS

Al proyecto No. 2013001030 del FONACIT, titulado "*Modelo Pedagógico y Plataforma Computacional para carreras vinculadas a las TICs basado en el Paradigma de las Nubes*", por el financiamiento otorgado para este proyecto.

REFERENCIAS

[1] G. Páez, B. Sandia. *Building a New Education Environment*. NE ASEE, Bridgeport, Connecticut, USA, 2009.

[2] J. Meister. *Universidades Empresariales*. McGraw Hill, Colombia, 2000.

[3] M. Dembo. *Motivation and Learning Strategies for College Success. A Self-Management Approach*. Lawrence Erlbaum Associates, NJ, 2000.

[4] ACM/IEEE, *Computer Science Curricula 2013*, http://ai.stanford.edu/users/sahami/CS2013/strawman-draft/cs2013-strawman.pdf.

[5] *Carreras en Computación, Informatica y Sistemas*, http://sercomputista.blogspot.com/2009/12/comparacion-entre-los-planes-de.html.

[6] UNESCO, 1998, http://www.unesco.org/es/efa.

[7] A. Kaplan and M. Haenlein, *User of the Word, Unite!. The Challenger and Opportunities of Social Media*, Business Horizons. vol. 53, no. 1, pp. 59-68, 2010.

[8] W. Penuel, B. Means and M. Simkins, *The Multimedia Challenge*, Teaching the Information Generation, vol. 50, no. 2, pp. 34-38, October 2000.

[9] C. Brewster and J. Fager, *Increasing Student Engagement and Motivation: From Time-on-Task to Homework*, Northwest Regional Educational Laboratory, Portland, Oregon, USA, 2000. http://www.nwrel.org/request/oct00/index.html

[10] J. Keller, *Strategies for Stimulating the Motivation to Learn*, Performance and Instruction Journal, vol. 26, no. 8, pp. 1-7.1987.

[11] K. Fisch and S. McLeod, *Did you Know*, http://www.youtube.com/user/xplanevisualthinking.

[12] R. Ferreiro, *Más Allá de la Teoría: El Aprendizaje Cooperativo. El Constructivismo Social. El Modelo Educativo para la Generación*, Nova Southeastern University, Florida, USA, 2001. http://www.redtalento.com/Articulos/WEBSITE%20Revista%20Magister%20Articulo%206.pdf

[13] R. Costaguta, *Una Revisión de Desarrollos Inteligentes para Aprendizaje Colaborativo Soportado por Computadora*, Revista Ingeniería Informática, Edición 13, 2006, http://www.maramora.com.ar/metodologia/garcia_e/texto_03.pdf.

[14] F. Díaz-Barriga y G. Hernández, *Estrategias Docentes para un Aprendizaje Significativo. Una Interpretación Constructivista*, McGraw-Hill Interamericana, México, Octubre 2004.

[15] S. Hernández, *El Constructivismo Social como Apoyo en el Aprendizaje en Línea*, Revista Apertura, vol. 7, no. 7. 2005.

[16] A. Muñoz, B. Sandia y G. Páez, *Un Modelo Ontológico para el Aprendizaje Colaborativo en la Educación Interactiva a Distancia*, en las memorias del I Congreso Iberoamericano de Enseñanza de la Ingeniería, Margarita, Venezuela. Noviembre 2009.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 12-22, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Modelos de Variabilidad con Requisitos no Funcionales en un Contexto de Producción Industrial de Software

Victor Esteller[1], Francisca Losavio[2], Alfredo Matteo[2], Oscar Ordaz[2]
vesteller@uc.edu.ve, francislosavio@gmail.com, alfredo.matteo@ciens.ucv.ve, oscarordaz55@gmail.com

[1] Departamento de Computación, Facultad de Ingeniería, Universidad de Carabobo, Valencia, Venezuela
[2] Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: La producción industrial de software es enfocada por la Ingeniería de Software como Líneas de Productos de Software (LPS) y se centra en representar características comunes y variables de productos, para lo cual se realiza un "modelo de variabilidad". Para la realización de dicho modelo se han propuesto diferentes enfoques, modelos, métodos y técnicas, las cuales tratan esencialmente la variabilidad respecto a los requisitos funcionales (RF); mientras que el problema de la variabilidad de los requisitos no funcionales (RNF) no ha sido aún resuelto. Por lo tanto, el objetivo de este trabajo es identificar aspectos claves que deben estar presentes en un proceso general de diseño del modelo de variabilidad que considere también los RNF, por lo que se hará una revisión sistemática de la literatura referente a los enfoques, modelos, métodos y técnicas vigentes que consideran modelos de variabilidad que explícitamente integran RF y RNF. Finalmente, a partir del estudio realizado, se hará una primera propuesta para un proceso de diseño de un modelo de variabilidad con RF y RNF, donde se considerará el estándar ISO/IEC 25010 para la especificación de los RNF para unificar la terminología del dominio respecto a las propiedades de calidad.

Palabras Clave: Línea de Productos de Software; Modelo de Variabilidad; Requisitos no Funcionales; ISO/IEC 25010; Revisión Sistemática.

Abstract: Industrial software production is focused by Software Engineering as Software Product Lines (SPL) and focuses on representing common and variable characteristics of software products, for which a "variability model" must be constructed. For the realization of this model different approaches, models, methods and techniques have been proposed, treating essentially the variability with respect to functional requirements (FR); while the problem of variability of non-functional requirements (NFR) has not treated as much and it is still an open a problem. Therefore, the aim of this work is to identify key aspects that must be present in a general design process of the variability model considering also NFR. In consequence, a systematic review of the literature on approaches, models, methods and techniques that consider existing variability models explicitly integrating NFR and FR is performed. Finally, from the results of the present study, an initial proposal for a design process of a variability model with FR and NFR is presented, considering the standard ISO/IEC 25010 to specify NFR and unify the terminology of the domain with respect to to quality properties.

Keywords: Software Product Line; Variability Model; Non-Functional Requirements; ISO/IEC 25010; Systematic Review.

I. Introducción

El desarrollo de sistemas de software a gran escala ha sido un desafío para investigadores y desarrolladores, y ha sentado las bases de la disciplina de la Ingeniería de Software, principalmente por lo complejo de la elaboración de un sistema como un todo o de cada uno de sus componentes individuales, y en particular, en lo referente a sus exigencias de calidad [1].

En este contexto, las Líneas de Productos de Software (LPS) [2], han sido propuestas para construir sistemas complejos, donde cada uno de sus componentes puedan ser reutilizados para el desarrollo de otros sistemas similares con requisitos distintos, facilitando a los desarrolladores la opción de administrar la configuración de una arquitectura genérica o arquitectura de referencia, es decir de seleccionar apropiadamente las propiedades comunes y variables de cada producto de software a reutilizar, para así generar nuevos productos.

Una LPS es definida como una colección o familia de sistemas de software que comparten un conjunto común y gestionado de características para un dominio, las cuales satisfacen necesidades específicas de un segmento particular del mercado y que son desarrolladas de una forma preestablecida, a partir de un conjunto común de activos de software [2]. Por otra parte, en [3][4] se utiliza la definición de un dominio, como el conjunto mínimo de propiedades que describen con precisión

una familia de problemas, para la cual se requiere una aplicación o solución computacional.

El término activo de software denota a un sistema, componente o producto de software diseñado expresamente para ser utilizado múltiples veces en el desarrollo de diferentes sistemas o aplicaciones; no solamente se consideran como activos a sistemas ejecutables sino también, por ejemplo, una especificación de requisitos, un modelo de negocio, una especificación de diseño o un patrón arquitectónico [5].

El proceso de desarrollo clásico de las LPS, denominado también proactivo o descendente, considera dos procesos que se llevan a cabo en paralelo: la Ingeniería del Dominio (ID) y la Ingeniería de Aplicación (IA) [6][7][8]. La ID captura información y representa el conocimiento sobre un dominio determinado, con la finalidad de crear activos de software reutilizables en el desarrollo de cualquier nuevo producto de una LPS [5].

Un aspecto central de la ID es construir la Arquitectura de Referencia (AR), o arquitectura del dominio, que consiste en una arquitectura genérica, cuyo núcleo básico de componentes está constituido por el modelo de características (en inglés "feature model"), que representa requisitos del software, funcionales o no, y el modelo de variabilidad (que representa las distintas maneras de generar nuevos productos reutilizando las características comunes del núcleo básico).

Un modelo de variabilidad [8] puede estar incluido o no en el modelo de características. Sus elementos conceptuales principales son los puntos de variación (en inglés "variation points") o componentes genéricos, que indican donde hay variabilidad en la AR y sus instancias o variantes (en inglés "variants") que son las componentes con soluciones concretas para configurar los nuevos productos. A partir de la AR debe ser posible generar todos los productos de la familia.

Para el desarrollo de los activos principales en una LPS, se requiere de un proceso de análisis del dominio, en el cual se identifica la parte común y variable de una familia de productos, formulándose así el modelo de variabilidad. Para esto, el estudio de la variabilidad en el análisis del dominio se ha convertido en un paradigma importante donde se han propuesto diferentes enfoques, modelos, métodos, aproximaciones y técnicas de análisis. En [5] se hace una revisión de métodos para el modelado de la variabilidad en general, y se considera que todas estas prácticas de análisis de dominio ayudan a construir, definir o utilizar modelos de variabilidad. Así mismo, en [9] se realiza una revisión de métodos de manejo de la variabilidad, sin tratarlos explícitamente como modelos.

Sin embargo, en ninguno de los trabajos [5][9] se tratan explícitamente los requisitos no funcionales (RNF), que en la literatura son también denominados indistintamente como atributos o requisitos de calidad.

Los enfoques comúnmente empleados para modelar la variabilidad en LPS enfatizan a la funcionalidad capturada a través de los requisitos funcionales (RF) y modelada generalmente por casos de uso. Por lo tanto, dentro de la óptica de las tendencias actuales, es importante que el proceso de especificación de la variabilidad tome en cuenta también los RNF y los correspondientes objetivos de calidad, tanto de RF como de RNF [12], porque permiten asegurar que el producto de software alcanzado cumpla con las características de calidad deseadas, para estar acorde con un proceso de producción industrial.

Con respecto al tópico: "tratamiento específico de la variabilidad de los RNF", que es el objetivo de este estudio, se han realizado numerosas investigaciones en la literatura. Para esto se hizo una revisión sistemática de acuerdo a la metodología de B. Kitchenham [10], y consistió en una búsqueda dirigida por estrategias, la selección y el filtrado de la literatura relativa al tópico de investigación planteado; con esta técnica se seleccionaron diez (10) trabajos que están directamente relacionados con el tópico y que se analizarán en el transcurso del presente trabajo.

A pesar de la existencia de múltiples aproximaciones sobre el modelado de las características esenciales comunes de los productos de un dominio y de su variabilidad, se observan muchas deficiencias, sobre todo en el abordaje de los RNF que no son directamente percibidos por el usuario, son difíciles de cuantificar, son especificados muy vagamente y generalmente se consideran en etapas posteriores al análisis. Estos aspectos incrementan enormemente la complejidad y disminuyen la eficiencia en el desarrollo, dificultando el mantenimiento, contradiciendo los principios del paradigma de la orientación a objetos.

Esto implica la necesidad de unificar criterios y el abordaje de los RNF en etapas tempranas, ya que de ellos se derivan los atributos de calidad asociados a los RNF, que deben ser medidos [11] y tomados en cuenta, ya que condicionan la calidad global del sistema. Además, se debe recalcar que las funcionalidades principales del dominio en general, derivadas de los RF, también exigen objetivos precisos de calidad. Es de resaltar que en un contexto de producción industrial, la calidad del producto representa un objetivo primordial a conseguir, cualquiera que sea la disciplina o enfoque empleado, es de máxima importancia; debe recordarse que la calidad del proceso determina la calidad del producto, [1]. Se define la calidad del producto como el conjunto de características o propiedades deseadas que deben estar presentes en el producto, considerando el punto de vista del usuario y del producto mismo [11].

Por otra parte, el hecho de considerar estándares de calidad, representa un aspecto central en la propuesta de este trabajo, ayudando a la construcción de un vocabulario común respecto a la calidad del producto de software. mediante el modelo de calidad, constituyendo así un activo de software importante para la construcción de la AR; debe recordarse que esta arquitectura genérica contiene el modelo de variabilidad para la generación de los diferentes productos. Establecer el modelo de calidad del dominio o vista de calidad del dominio [13], ayuda a definir los RNF globales del sistema, así como los requisitos de calidad que se derivan de las funcionalidades del usuario o funcionalidades implícitas, que son comunes a una familia de productos del dominio y representan la vista de calidad como un activo, siendo parte del conocimiento del dominio [4].

El tratamiento explícito de la variabilidad de RNF implica considerar requisitos de calidad no cuantificables directamente, como por ejemplo alta seguridad, disponibilidad e interoperabilidad; tomar en cuenta estos requisitos de calidad

implica, seleccionar variantes adicionales en la configuración de los productos específicos. Estos RNF de alto nivel no son medibles directamente y deben ser refinados hasta llegar a los elementos medibles. La idea es llegar a una granularidad que permita determinar la variabilidad no funcional para obtener productos concretos.

Por lo antes expuesto, determinar las características comunes y variables de una familia de productos es ampliamente estudiado y modelado durante el análisis del dominio, desde el punto de vista funcional; sin embargo no es así en lo que respecta a los RNF [8][12], hacia donde se dirige actualmente la investigación en el contexto de este trabajo. En este sentido Sepúlveda y Cachero [35] en respuesta a esta problemática, hacen una revisión de métodos de gestión de RNF en las LPS, sin embargo es un trabajo corto y no concretizan su propuesta de cómo facilitar la comunicación entre los grupos de trabajo.

Como consecuencia de la problemática planteada, nuestro trabajo tiene como objetivo principal revisar y comparar los diferentes modelos vigentes para la especificación de la variabilidad en la etapa de análisis del dominio, en particular aquellos modelos que contemplan también el tratamiento de la variabilidad respecto a los RNF.

Los resultados obtenidos en la revisión serán utilizados como base para plantear los aspectos esenciales que debe tener un modelo de variabilidad que integra RF y RNF, y un proceso de diseño del modelo de variabilidad de RF y RNF para una AR, inspirado en [32]. Además, se plantea cómo asociar a las características no funcionales del modelo de variabilidad, una especificación estándar utilizando la norma ISO/IEC 25010 [11].

El presente trabajo, además de esta introducción, se estructura de la siguiente forma: la segunda sección presenta la revisión sistemática que se siguió para seleccionar los enfoques. La tercera sección describe las técnicas, enfoques y modelos utilizados para expresar la variabilidad de RNF especificados como requisitos de calidad. La cuarta sección presenta la comparación y el análisis de resultados. En la quinta sección se formula una primera propuesta del proceso general de diseño del modelo de variabilidad con RF y RNF. Finalmente en la sexta sección se dan las conclusiones y perspectivas.

II. Revisión Sistemática

Para el estudio de la literatura vigente en el tema, se siguió la metodología de Revisión Sistemática o "Systematic Review" de B. Kitchenham [10], por sus ventajas en el área de la Ingeniería de Software para buscar, seleccionar y filtrar el enorme volumen de material literario disponible en un tema de investigación, y seleccionar aquellos trabajos directamente relacionados y de mayor interés; para tal fin se siguieron las fases o pasos recomendados en [10]:

A. Estrategia de Búsqueda y Fuentes de Información

La cadena de búsqueda utilizada en esta investigación se construyó de acuerdo a las siguientes pautas:

- Derivar los términos principales basado en los tópicos a ser investigados
- Determinar e incluir sinónimos y términos relacionados
- Combinar el orden de los términos de búsqueda

De acuerdo a estas estrategias se construyó la cadena de búsqueda, quedando de la siguiente manera:

<<software product line AND variability AND (quality attributes OR quality requirements OR non-functional requirements) >>

Nótese que los términos "quality attributes", "quality requirements" y "non-functional requirements" con frecuencia son usados indistintamente en la literatura.

Entre las Fuentes consultadas se pueden citar: IEEExplore, ACM Digital Library, Citeseer Library, Google Scholar. Además de los artículos conocidos y referenciados en las revisiones de [14][15]. También se consultaron como fuentes las actas "in extenso" de los congresos internacionales RCIS (2010) (Proceedings of the International Conference on Research Challenges in Information Science) y VaMoS (2010) (International Workshop on Variability Modelling of Software-intensive Systems).

B. Selección de Estudios

Se encontraron 401 artículos entre todas las fuentes mencionadas. Luego se removieron los duplicados. En una primera etapa de selección, revisando los resúmenes en caso de no tener el texto completo, sólo se incluyeron los que trataban el manejo de la variabilidad en LPS, considerando atributos o requisitos de calidad o requisitos no funcionales, como se especificó en la pregunta de búsqueda, resultando 37 trabajos.

C. Extracción de Información y Síntesis

Estos 37 trabajos fueron analizados detalladamente y sólo se seleccionaron 10 trabajos, ya que planteaban explícitamente un método, enfoque o técnica donde se aborda la variabilidad no funcional en LPS [12][15][16][17][18][19][20][21][22][23], constituyendo estos 10 trabajos un marco común de análisis.

III. Variabilidad de Requisitos no Funcionales

En los 10 trabajos seleccionados se encontraron diversos enfoques para análisis del dominio que consideran la variabilidad, donde se mencionan o toman en cuenta explícitamente los requisitos de calidad. Estos enfoques que contemplan técnicas, modelos y métodos, fueron seleccionados por su completitud, en cuanto a definición y organización de sus procesos, y buena documentación de los artefactos que se generan, además de ser bien conocidos por la comunidad científica del área. Gran parte de ellos coinciden en utilizar el Grafo de Interdependencias de "Softgoals", del inglés SIG ("Softgoals Interdependency Graph") [24] como herramienta para integrar RF y RNF, en el modelo de características y de variabilidad.

En lo que sigue, se hará una revisión detallada de los enfoques: Definiciones jerárquicas [16]; Modelo de redes bayesianas (BBN, "Bayesian Belief Network") [17][33][34]; Modelo basado en metas [18][25]; F-SIG ("Feature-Softgoal Interdependency Graph") [19]; COVAMOF ("ConIPF Variability Modeling Framework") [20]; Svamp ("Software Variability Modeling Practices") [21]; IQ-SPLE ("Integrated Quality Software Product Line Engineering") [22]; Método de análisis de RNF orientado a características para LPS [23]; Modelo de características extendidas EFM ("Extended Feature Model") [15], y finalmente el Método de Gurses [12]. Es de hacer notar que sólo se mostrarán los gráficos que están

estrechamente relacionados con la propuesta que se realiza en esta investigación.

A. Definiciones Jerárquicas

Esta propuesta [16] consiste en la definición de un modelo jerárquico para la ingeniería de requisitos, donde los requisitos para diferentes productos son representados en la misma jerarquía. A diferencia de otros métodos para especificar requisitos, este método analiza los requisitos de acuerdo a las categorías objetivos de diseño y decisiones de diseño, en lugar de enfatizar solamente los RF basados en las necesidades del cliente.

La jerarquía está estructurada como un árbol lógico "AND", en la cual los nodos superiores representan objetivos de diseño o RNF, tales como: controladores arquitecturales y atributos de calidad, que se supone que el sistema debe satisfacer. Los otros nodos representan decisiones de diseño o RF. Cada nodo en la definición jerárquica tiene una prioridad que refleja la importancia de él, con respecto a su padre.

Las prioridades representan productos específicos y están dados en forma de tuplas. Ellos son utilizados como mecanismo para describir productos en la definición jerárquica. Cuando hay un conflicto entre un objetivo y una decisión de diseño, se dice que el requisito es parcialmente satisfecho por la decisión de diseño.

Este método de decisiones jerárquicas ayuda a los diseñadores en el manejo de los requisitos y como interrelacionarlos, específicamente para resolver conflictos e inconsistencias entre decisiones de diseño y objetivos. También facilita la tarea de encontrar requisitos perdidos por medio de un recorrido en reverso en el árbol jerárquico. Además de esto, si un requisito es considerado ambiguo, es fácilmente re-definible agregando nuevos nodos de objetivos de diseño. Sin embargo, a medida que en este árbol se tenga mayor cantidad de productos, se incrementa la complejidad del mismo, pudiendo ser ineficiente.

B. Redes Bayesianas de Conocimiento (BBN)

Este enfoque [17][33][34] propone un método basado en aproximaciones para predecir la calidad en una línea de productos de software. El BBN ("Bayesian Belief Network") se basa en un modelo para determinar explícitamente el impacto de las variantes (decisiones de diseño especiales) sobre los atributos de calidad del sistema.

Este método contempla dos artefactos principales: modelo de características tipo FODA [26], el cual es utilizado para capturar los RF y el modelo BBN para capturar el impacto de las variantes funcionales sobre los atributos de calidad. Los atributos de calidad son representados como nodos en el modelo BBN, y son especificados con escalas. Por ejemplo: "alta" o "baja", y esas escalas son especificas para cada dominio, así de este modo en un dominio dado, la instancia "alta" referente por ejemplo al tiempo de respuesta de la característica eficiencia, es más o menos igual a 0,5 ms.

Se utilizan arcos dirigidos para relacionar una decisión de diseño o variante con un atributo de calidad. Probabilidades condicionales son utilizadas para cuantificar las relaciones conceptuales; un valor de probabilidad condicional sobre los arcos, refleja el conocimiento de los expertos del dominio referente a qué tanto una decisión de diseño influye en un atributo de calidad.

C. Modelo Basado en Metas (GORE)

En este enfoque [18], se propone la utilización de técnicas de la Ingeniería de Requisitos Orientada a Metas, del inglés "Goal-Oriented Requirements Engineering (GORE)" [27], aplicada al contexto LPS. Específicamente, se plantea un modelo basado en metas para representar la variabilidad, así como también se provee de una técnica visual de análisis de variabilidad. Este modelo es una versión restringida del modelo de metas del conocido "Framework NFR", del inglés "Non-Functional Requirements Framework" [24], el cual ofrece una aproximación sistemática para definir RNF, su trazabilidad y sus interdependencias. Además ayuda a los diseñadores a considerar las acciones necesarias para asegurar la calidad. El "Framework NFR" está a su vez basado en el SIG ("Softgoals Interdependency Graph").

Un SIG o grafo de interdependencias de "softgoals", es un diagrama propuesto para modelar RNF por medio de metas no funcionales o "softgoals", las cuales se cumplen en un cierto contexto de acción, por ejemplo Seguridad [Acceso a Cuentas] indica el nombre del "softgoal", "Seguridad", el cual debe ser satisfecho en el contexto de acción o componente funcional "Acceso a Cuentas" [18][27][28].

De acuerdo a lo observado, este método genera dos artefactos o sub-modelos: un modelo de metas funcionales ("hardgoals") y un modelo de metas no funcionales ("softgoals"). El primero representa objetivos funcionales y el segundo representa las condiciones o criterios que el sistema debe cumplir desde el punto de vista no funcional; las metas no funcionales se identifican específicamente con los RNF respectivos y requisitos de calidad, y se representan por el diagrama SIG. En [28], sólo se utiliza un SIG que contiene la información integrada de RF y RNF, lo cual facilita la trazabilidad de los requisitos de calidad.

La operacionalización de un "softgoal" es decodificada en el sub-modelo de metas funcionales como tareas o mecanismos que deben ser implementados para que el "softgoal" sea satisfecho. Ambos sub-modelos son árboles "AND" y "OR".

Las prioridades están dadas por cada "softgoal" en una escala percentil y las correlaciones son utilizadas para representar relaciones entre las funcionalidades "hardgoals" y los "softgoals". Las correlaciones tienen diferentes etiquetas de influencia (--, -, ?, +, ++).

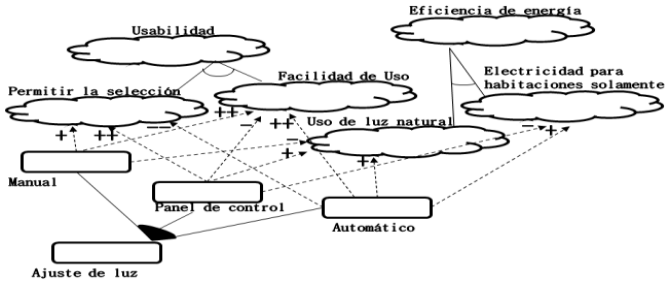


Figura 1: Combinación de Diagramas SIG con Metas Funcionales "Hardgoals" (Rectángulos Blancos) y no Funcionales "Softgoals" (Nubes Grises)

Estas etiquetas cualitativas son convertidas en valores cuantitativos, de la siguiente manera: El valor 1 para satisfactorio y otro valor para negación (ver Figura 1).

D. F-SIG (Grafo de Interdependencias entre "Features" y "Softgoals")

El objetivo principal de este método [19] es proveer un "framework" que permita representar decisiones de diseño en forma de interdependencias entre variantes funcionales y requisitos de calidad. Para ello se propone un nuevo modelo, que consiste en la unión de un modelo de características y un modelo de "softgoals", resultando en una extensión del FODA clásico [26], del inglés ("Feature-Oriented Domain Analysis"), al cual denominan F-SIG.

Es utilizado como método de análisis y modelado del dominio, que usa un modelo de características al estilo FODA para representar variabilidad. A su vez, en el F-SIG se consideran contribuciones explicitas e implícitas (donde una contribución es explicita cuando la selección de una variante influye directamente en un requisito de calidad, mientras es implícita cuando la selección de un requisito de calidad no está dado directamente por la selección de una variante funcional).

Al igual que el Modelo basado en metas [18], se tienen dos modelos FODA, el primero referente a funcionalidades de comportamiento del sistema y el segundo referente a decisiones de diseño, ambos relacionados con un modelo de "softgoals" (ver Figura 2).

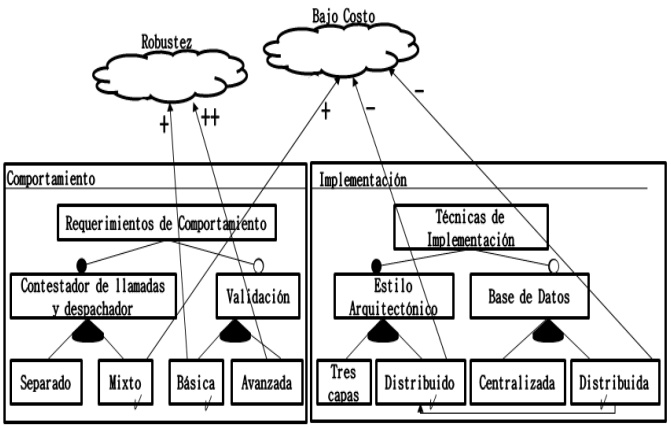


Figura 2: Ejemplo de un F-SIG que Muestra la Combinación de un SIG con Metas Funcional y no Funcional

E. COVAMOF

Es un "framework" para el modelado de la variabilidad en una familia de productos de software [20], contempla todas las capas de abstracción de la familia de productos, específicamente: - el modelo de características estilo FODA en el nivel más alto de abstracción que sería referente a las funcionalidades; - Diseño Arquitectónico en un nivel intermedio; e - Implementación de componentes en el nivel más bajo.

Este "framework" se basa en la utilización exhaustiva de puntos de variación y dependencias para capturar la variabilidad. Un punto de variación puede representar una funcionalidad, un artefacto o un valor de alguna variante de producto, mientras que una dependencia restringe la forma en que se seleccionan uno o más puntos de variación. Además proporciona un gráfico llamado CVV ("COVAMOF Variability View"), en el cual se refleja la variabilidad con las dos vistas de variación y de dependencia.

Los atributos de calidad pueden ser modelados con las dependencias, las cuales pueden especificar una propiedad que determina el valor de un atributo de calidad, tal como desempeño o utilización de la memoria. Aunque este "framework" no muestra claramente la manera de especificar y cuantificar el valor de dicho atributo.

F. Svamp ("Software Variability Modeling Practices")

Este enfoque [21] sintetiza y formaliza conceptos existentes en los enfoques de modelado de características tradicionales. A su vez, considera tanto el modelado funcional como la variabilidad de atributos de calidad en un nivel arquitectural. La especificación arquitectural se realiza por medio de diferentes puntos de vista. Específicamente, un modelo FODA y un modelo de punto de vista estructural especifican la estructura y las funcionalidades, así como también la variabilidad.

Adicionalmente, Svamp utiliza varios modelos para especificar una familia de productos de software: - un modelo de Kumbang [21], que consiste en modelos de puntos de vista estructurales y de características para representar las funcionalidades del sistema, expresados en UML 2.0, es decir su arquitectura; - un modelo de atributos de calidad que considera los requisitos de calidad asociados a los componentes en el punto de vista estructural del modelo de Kumbang; - un modelo de variabilidad de la calidad para expresar la variabilidad de esos atributos de calidad. Estos dos últimos modelos pertenecen a un perfil QA, del inglés "Quality Attribute profile". Cada uno de estos tres modelos es definido por una ontología, donde cada ontología provee un metamodelo para el modelado de los conceptos. Cada ontología del modelo de calidad define la dimensión técnica de un requisito de calidad. Por ejemplo, la ontología seguridad está definida por activos de seguridad, atributos, hilos, soluciones y métricas, mientras que la ontología fiabilidad define procesos, métodos, modelos y métricas. De este modo, cada ontología de calidad tiene asociados requisitos de calidad específicos.

Este enfoque provee una herramienta automatizada, desarrollada bajo una plataforma eclipse, que puede utilizarse para crear el modelo de Kumbang como instancia de una ontología. El usuario puede especificar las características funcionales de la familia de productos, los elementos arquitecturales y las restricciones entre ellos. La herramienta verifica que al menos una configuración válida pueda ser derivada del modelo. La herramienta tiene la desventaja de no ser interoperable, así como la carencia de poder transformar diferentes modelos de forma automática.

G. Método de Análisis de RNF Orientado a Características para LPS

Este método [23] se concentra en el análisis de variabilidad no funcional y en el modelo de decisiones orientado al "Framework NFR" [24]. Tiene como artefacto de entrada un modelo que combina un FODA para las características funcionales, con un SIG basado en el "Framework NFR" para los RNF. Como salida final se obtiene un modelo de decisión integrado de características funcionales estilo FODA y "softgoals".

El método consta de cuatro fases las cuales se describen a continuación:

1. Construcción del contexto

Se construye un modelo de características de contexto sobre el FODA inicial, el cual se lleva a cabo de acuerdo al dominio en estudio, del cual se derivan las incumbencias no-funcionales en forma de escenarios de ejecución, eventos, conceptos sociales, etc.

2. Identificación de la variabilidad no funcional

Se observan las variaciones no-funcionales del "framework NFR" con respecto al SIG inicial, soportándose en el modelo de contexto y en las plantillas o en inglés "templates NFR", esto es la codificación de un contexto genérico en el cual se deben satisfacer los RNF. Las variaciones no funcionales en el SIG pueden originarse de distintas maneras y para ello se realizan tres actividades de análisis de las cuales se puede obtener una perspectiva no funcional completa, incluyendo todos los "softgoals" necesarios en el dominio, su configuración de variabilidad, los niveles de "softgoals" candidatos y su operacionalización. Las actividades en cuestión son:

- Análisis de presencia de objetivos: se evalúa la presencia de "softgoals" de acuerdo a los "templates NFR" y a la información de contexto del dominio. Puede utilizarse para evaluar la existencia de "softgoals" en diferentes aplicaciones y también para proveer heurísticas para el análisis de variabilidad no funcional. Luego los "softgoals" no concernientes son removidos y los "softgoals" reservados son evaluados para convertirse en obligatorios u opcionales.
- Análisis de niveles NFR: se determina el nivel (bajo-medio-alto) para cada "softgoal". Un alto nivel siempre reemplaza uno bajo, pero si un "softgoal" de alto nivel está en conflicto con otro "softgoal", el de bajo nivel debe reservarse. El concepto de conflicto entre "softgoals", en inglés "trade-offs", en la siguiente etapa de operacionalización, es parte significante en el análisis de variabilidad.
- Análisis de operacionalización: se evalúan las posibles operacionalizaciones de acuerdo al nivel de los "softgoals" reservados considerando sus dependencias. El resultado puede ser: completamente satisfactorio, parcialmente satisfactorio o insatisfactorio. De acuerdo a esto, la configuración de la variabilidad de los niveles de los "softgoals" relacionados debe ser ajustada.

3. Integración de "softgoals"

Los "softgoals" identificados son incorporados al FODA inicial con variaciones desde una perspectiva no funcional. A su vez, se incorporan las operacionalizaciones de los "softgoals" relacionados. Las características funcionales se refinan de acuerdo a las incumbencias no funcionales que hayan sido adicionadas. De ser necesario, se añaden restricciones en los "softgoals" relacionados en el modelo de decisión.

4. Modelado de decisión orientado a "softgoals"

Finalmente, se obtiene un modelo de decisión que integra funcionalidades y "softgoals", donde se reflejan los conflictos entre "softgoals", dependencias, requisitos de entorno y restricciones.

Este método tiene como deficiencia que no considera la utilización de estándares de calidad, por lo cual puede presentarse deficiencias al momento de cuantificar y formalizar la especificación de RNF junto a los RF, imposibilitando a su vez, la refinación de RNF hasta llegar a elementos medibles.

H. Método IQSPLE

Este método [22][29], consta de ocho pasos, de los cuales los cuatro primeros que se refieren a procesos de ID, específicamente llevados a cabo en el espacio del problema y los cuatro últimos en la IA, en el espacio de la solución. Entre los modelos conocidos, este enfoque utiliza FODA [26], Estructuras jerárquicas, Modelo de casos de uso, Modelo de componentes y Modelo de calidad.

De los ocho pasos sólo se explicarán los cuatro primeros ya que corresponden a Procesos de ID, que es donde se ubica la presente investigación:

1. Análisis de requisitos

Se especifican las características comunes y variables. Al mismo tiempo se recopilan los requisitos de calidad. Se emplea el FODA tradicional

2. Modelado de características FODA y atributos de calidad

Además del FODA se genera de forma separada un árbol de calidad para modelar los atributos de calidad y sus valores típicos, para lo cual se emplea CVM, del inglés, "Compositional Variability Management". De esta manera, el árbol de calidad se puede utilizar para seleccionar los requisitos no funcionales. De igual forma, elementos particulares en el árbol se pueden relacionar con otros árboles (pudiendo inhabilitar la selección de alguna característica), y con modelos UML. En el caso, que se emplee una herramienta de soporte a UML; se utiliza un API para soportar la sincronización automática entre los modelos, haciendo que ciertas características de calidad desaparezcan si no tienen soporte en el modelo de solución

3. Modelado de artefactos de solución y especificación de la calidad

4. Por medio de un metamodelo basado en UML, los atributos de calidad y sus valores son adicionados a los modelos del espacio de la solución.

I. Modelo de Características Extendido EFM ("Extended Feature Model")

Este autor [15] al igual que F-SIG [19], COVAMOF [20], Svamp [21], BBN [17][33][34], IQSPLE [22] y Gurses [12], propone una extensión al modelo FODA clásico [26], relacionando las funcionalidades con respecto a los requisitos no funcionales que apliquen, aportando mejoras con respecto a la propuesta original, cuya notación es muy ambigua. En este enfoque los RNF son "equivalentes" a los atributos de calidad y son especificados como sub-características pudiendo ser medidos como un atributo previamente definido en el dominio.

De esta manera, los atributos de calidad pueden ser especificados por la relación entre uno o muchos con respecto a alguna funcionalidad. Los atributos de calidad pueden ser utilizados para reflejar información no funcional tal como: costo, velocidad, memoria RAM o tiempo de desarrollo que sean necesario para satisfacer las funcionalidades. Estos

atributos pueden contener valores de un rango específico pertenecientes a un dominio discreto o continuo (ej. Entero o real).

El EFM también puede ofrecer restricciones complejas referentes a RF y a RNF. Por otra parte, en [30] se propone una notación que se ha adoptado para ilustrar como una funcionalidad es decorada por atributos. De forma que las funcionalidades padre son decoradas con expresiones que son dependientes de los valores de los atributos de sus funcionalidades hijo (ver Figura 3).

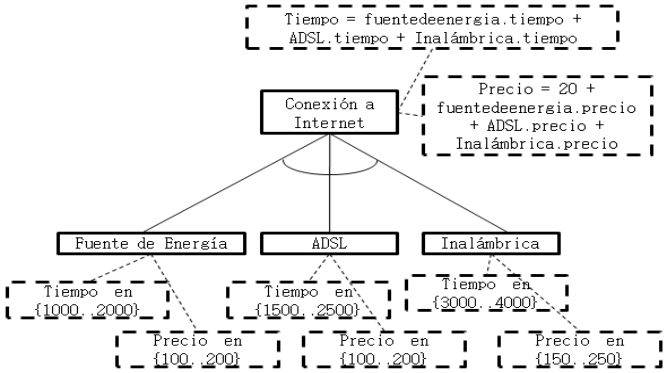


Figura 3: Ejemplo de un EFM

J. Método de Gurses

Este método [12] propone un proceso para construir un modelo de calidad complementario. Entre los modelos conocidos, también utiliza el modelo FODA clásico [26], el modelo EFM o FODA Extendido de [15], el "Framework NFR" de [24][23], el modelo F-SIG con la taxonomía de correlaciones de [19] para las interdependencias entre requisitos, y finalmente el estándar ISO/IEC 9126-1 (2001), incluyendo las métricas. Este método realiza el análisis del dominio y genera sus diferentes artefactos de la siguiente forma:

1. Se construye un diagrama de componentes del sistema de alto nivel (arquitectura del sistema).

2. Se genera un árbol FODA clásico por cada componente de la arquitectura, que contiene un grupo de características funcionales, que posiblemente son puntos de variación; las funcionalidades que no van a variar (comunes) para todos los productos no se colocan; se expresan las dependencias estructurales ("and", "or", "mandatory", etc.) entre las funcionalidades. Con lo que se obtiene básicamente un modelo de características funcional.

3. Cada funcionalidad de los árboles FODA es relacionada mediante una tabla utilizando las interdependencias implícitas "requires" o "excludes". Por ejemplo, la funcionalidad A "requires" B; si A es incluida, B también debe estar (A=>B); la funcionalidad A "excludes" B implica que A y B no pueden estar juntos, es decir A=> "NOT" B y B => "NOT" A. A<=>B se expresa como A=>B y B=>A.

4. Se hace un análisis cuantitativo con el modelo EFM [15], con etiquetas que contienen valores o rango de valores de restricciones impuestas a la funcionalidad.

5. Se realiza una unión disjunta de las etiquetas de las funcionalidades padres al estilo EFM. El modelo de calidad, que corresponde a un SIG, se obtiene relacionando las etiquetas compuestas de cada componente con las características de calidad de ISO/IEC 9126-1. Estas características de calidad son obligatorias para el producto (son etiquetadas como obligatorias en la terminología de FODA).

6. Se hace un análisis cualitativo con el "framework NFR", donde se construye un grafo SIG [24][23].

7. Se agregan al SIG los niveles "low", "medium", "high", como una escala de apreciación de la meta, al igual que [15] y luego se procede a hacer las operacionalizaciones.

8. Se construye el modelo de calidad complementario a partir del SIG con niveles cualitativos de apreciación ("low", "medium", "high").

9. Se hace una correspondencia entre las etiquetas compuestas del modelo cuantitativo y los objetivos del modelo cualitativo; se construye el árbol de ISO/IEC 9126-1 de forma descendente, del atributo de calidad a la funcionalidad; por ejemplo el atributo etiquetado "overall MTBF" corresponde a la sub-característica disponibilidad ("softgoal" en el SIG) y a la característica confiabilidad.

10. Se hace la correspondencia entre los rangos de valores especificados en las etiquetas del modelo cuantitativo y los niveles "low", "medium", "high" asociados a los "softgoals" del SIG.

11. Se relacionan las operacionalizaciones con las funcionalidades de donde provienen, en una tabla con "requires" y "exclude", construyendo un grafo similar al F-SIG [19].

12. Finalmente, se obtiene un modelo EFM con el modelo de calidad complementario.

IV. Comparación y Análisis entre los Enfoques, Técnicas y Métodos Estudiados

Para la comparación y evaluación de los diez métodos, técnicas y/o enfoques presentados en la sección anterior, se extenderá el "framework" propuesto en [14], el cual permite la comparación de métodos que modelan la variabilidad funcional y no funcional. Entre los aspectos generales tomados en cuenta para nuestra comparación, se tienen: los modelos para representar la variabilidad, el modelo de calidad para representar la variabilidad no funcional y el uso de estándares, la asociación entre características funcionales y no funcionales, y el empleo de herramientas automatizadas. A continuación se explica cada uno de los aspectos evaluados y su análisis de acuerdo a los métodos estudiados.

A. Modelos para Representar Variabilidad

De acuerdo a los resultados analizados, el modelo más utilizado para representar características funcionales es FODA [26], aunque en su definición original, las características pueden ser funcionales o no. Para representar requisitos de calidad los métodos más utilizados fueron el modelo SIG [27] y el modelo FODA Extendido (EFM) [15]. El enfoque IQ-SPLE [22], es muy expresivo al momento de representar la variabilidad, y a su vez está basado en "softgoals", por lo que es específico para caracterizar requisitos de calidad, ya que provee de estructuras especialmente pensadas para ese fin.

Sin embargo, es de resaltar la alta tendencia a realizar análisis de dependencias entre puntos de variación para especificar qué

requisitos no funcionales se ven implicados al seleccionar alguna variante de producto. Es decir que no está clara la trazabilidad entre los requisitos de calidad y los requisitos que los originan.

Una de nuestras proposiciones es que dentro del contexto de las LPS, un SIG pudiera servir de base para construir la arquitectura inicial del sistema, que después debe ser generalizada a una Arquitectura de Referencia; el modelo de calidad ISO/IEC 25010 puede ser utilizado para especificar el SIG.

Otro aspecto resaltante y en incremento es el énfasis que hacen los distintos métodos en la integración de modelos, principalmente con la finalidad de relacionar los RF con los RNF. Se ha observado que la trazabilidad entre los requisitos de calidad y su origen no está clara. A pesar de la importancia de considerar modelos de calidad en un proceso de producción industrial, solamente el enfoque de Gurses [12] y Svamp [21], utilizan y generan un modelo de calidad.

B. Modelo de Calidad para Representar la Variabilidad no Funcional y Uso de Estándares

La consideración de estándares ha tenido una baja aplicabilidad, en vista de que sólo Gurses [12] utiliza ISO/IEC 9126-1. A pesar que Svamp [21] genera entre sus artefactos un modelo de calidad, este no parece sustentado por ningún estándar.

A su vez, en el estudio de cada enfoque se determinó que la técnica de utilización de prioridades para requisitos de calidad ha tenido una aplicabilidad del 100%, lo cual puede ayudar (como se hace en [23]), a determinar los "softgoals" no concernientes que serán removidos y los "softgoals" reservados que serán evaluados para convertirse en obligatorio u opcional.

Por otro lado, se tiene que sólo IQ-SPLE [29] y Svamp [21] utilizan notación UML, lo cual resulta interesante de explotar en futuras propuestas, debido a su amplia aceptación a nivel de "stakeholders" y como lenguaje común en el área de la Ingeniería de Software.

Otro aspecto interesante que se puede extraer de este apartado de comparación es el análisis de niveles de variación de atributos de calidad, el cual tuvo una presencia de 50% entre los enfoques estudiados. Principalmente porque se encarga de determinar el nivel (bajo-medio-alto) para cada "softgoal". Donde un "softgoal" de alto nivel siempre reemplazará a uno de bajo nivel, pero si un "softgoal" de alto nivel está en conflicto con otro "softgoal" de bajo nivel, el de bajo nivel deberá conservarse.

C. Asociación entre Características Funcionales y no Funcionales

Se observa que la mayoría de los enfoques modelan características funcionales, mientras que una menor porción relaciona las características funcionales y no funcionales.

Por otra parte, el análisis cuantitativo tiene una consideración del 60% entre los enfoques, por la dificultad de asociar métricas a las características de calidad y funcionalidades a un alto nivel de abstracción [31]. Por lo cual sería conveniente considerar en propuestas futuras algún método o criterio que facilite dicha tarea a partir del conocimiento del dominio.

El análisis cuantitativo generalmente se representa con el modelo EFM [15]. Mientras que el análisis cualitativo (el cual tiene una consideración del 100% entre los enfoques) se hace apoyándose habitualmente en el modelo SIG.

Se observó la utilización de algunas técnicas, como la integración de modelos [22][23][29]. Adicionalmente, sólo IQ-SPLE [22][29], Gurses [12] y Svamp [21] proveen de una herramienta automatizada para generar modelos que de alguna forma representen la variabilidad.

Otro aspecto interesante, y que ha sido considerado en enfoques recientes es el análisis de la variabilidad composicional [22][23][29], en la cual además del FODA, se generan de forma separada un árbol de calidad para modelar los atributos de calidad y sus valores típicos, empleando CVM (Compositional Variability Management). De esta manera, el árbol de calidad se puede utilizar para seleccionar los requisitos no funcionales.

En nuestra propuesta, el árbol de calidad podría ser especificado según el estándar ISO/IEC 25010. La mayoría de los enfoques se concentran solamente en la ID, mientras que sólo COVAMOF [20], IQ-SPLE [29] y Gurses [12] consideran también la fase de IA y la relación entre ellas.

Pocos son los métodos que ofrecen un caso de estudio real completo en concordancia con sus etapas de realización. De igual manera, pocos son los métodos que poseen un modelo de proceso y conducen el proceso de forma sistemática y bien documentada; sólo un 30% de los métodos proveen de unos pasos claros para llevar a cabo el proceso de la variabilidad, como son [12][23][29].

Tabla I: Comparación entre Métodos de Modelado de Variabilidad que Consideran Requisitos de Calidad

Aspecto Evaluado / Técnica \ Método		GORE [18][27][28]	F-SIG [19]	COVAMOF [20]	Def. jerárquicas [16]	BBN [17]	EFM [15]	IQSPL [22]	Gurses [12]	Peng [23]	Svamp [21]
Modelos para representar la variabilidad	FODA [26]	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**1**
	EFM [15]	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**
	SIG [27]	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	Decisión [23]	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
	Ortogonal [8]	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**
	F-SIG [19]	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**1**
Modelo de Calidad para representar la variabilidad no funcional y uso de estándares	UML	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**
	Estándares de calidad	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	Modelo de calidad	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
Asociación entre características funcionales y no funcionales	Modela características funcionales	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	Relaciona características funcionales y no funcionales	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
Herramientas automatizadas		**1**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**

D. Herramientas Automatizadas

Se estudió la presencia de herramientas de soporte al método propuesto; se observó que sólo IQSPLE [29], Gurses [12] y

Svamp [21] poseen herramientas automatizadas de apoyo. Los aspectos antes discutidos pueden verse resumidos en la Tabla I.

En conclusión, los resultados del estudio fueron que sólo dos (2) trabajos utilizan el modelo de calidad y sólo uno (1) utiliza un estándar para el modelo de calidad, el ISO/IEC 9126-1 (2001), lo cual consolida nuestra propuesta de usar su actualización, el ISO/IEC 25010 [11] para unificar la terminología sobre calidad del producto software en diferentes dominios, como una de las buenas prácticas de la Ingeniería de Software.

V. Primera Propuesta

En vista del análisis efectuado, formulamos una primera propuesta para un proceso de diseño de un modelo de variabilidad para una AR que considera RF y RNF, en el cual se toma como base la combinación de algunos de los modelos estudiados: FODA Extendido (EFM) [15], el enfoque GORE [18][27][28] con el SIG en la notación de Supakkul y Chung [28] y tal como es utilizado por Gurses [12], además se utiliza una técnica análoga al F-SIG [19] para determinar los conflictos o "trade-offs" entre los requisitos de calidad; la especificación de los "softgoals" se realizará mediante el modelo de calidad estándar ISO/IEC 25010; la construcción de la AR se inspira en proceso descendente (en inglés "bottom-up") o reactivo, basado en la refactorización de las arquitecturas de productos existentes [32]:

- A partir del EFM [15] se construye el modelo SIG [28] representando los RNF o "softgoals", como características del modelo de calidad estándar y refinándolos hasta las sub-características y atributos. Los atributos, en este contexto, corresponden a la "realización" de operacionalizaciones (en inglés "operationalization"), que son componentes arquitectónicos. Este SIG puede "traducirse" a un diagrama de componentes en UML 2.0 y representa una Arquitectura Candidata (AC). La AC contiene un modelo de variabilidad con las variantes funcionales (por el EFM) y no funcionales por las operacionalizaciones del SIG que representan las propiedades de calidad (o RNF) asociadas a las funcionalidades.
- Finalmente, la AR se construye a partir de la AC, generalizando las variantes de la AC, que constituyen los puntos de variación del modelo de variabilidad.

De acuerdo a lo anterior, se propone un primer esbozo de proceso de diseño del modelo de variabilidad con RF y RNF para construir la AR de una LPS; el proceso propuesto establece como premisa: tener información sobre una familia de productos similares en cuanto a funcionalidades básicas y estilos arquitectónicos en un dominio dado. Y consta de siete fases que se detallan a continuación:

Fase I – Construcción del Contexto

Como punto de partida se construye un modelo de características extendido referente al contexto del dominio de estudio, obteniendo así un EFM inicial [15].

Nota: esta fase se inspira en el método seguido en el modelo EFM [15], considerando también un análisis de las similitudes entre componentes y conectores de los productos tomados en cuenta para el estudio.

Fase II – Análisis de Presencia de Objetivos

Se evalúa la presencia de "softgoals" de acuerdo a la información de contexto del EFM [15]. Los "softgoals" identificados se asocian a una característica del ISO-25010. Esto permite evaluar la existencia de "softgoals" en diferentes configuraciones.

Nota: esta fase se inspira del método de Gurses [12], quien sólo considera las características no comunes en el EFM. En nuestro caso se toman las comunes y no comunes.

Fase III – Identificación de la Variabilidad No Funcional

Se especifica por medio del SIG obtenido en el paso anterior del EFM, cada una de las características de calidad asociada a cada "softgoal", especificado como en [28], componente (contexto) y la lista de requisitos de calidad asociados.

Fase IV – Análisis de Prioridad

En esta fase las características de calidad del ISO-25010 no concernientes en cuanto a presentar conflictos o contradicciones son removidas y las características de calidad reservadas son evaluadas para convertirse en obligatoria u opcional, estableciendo prioridades (baja-media-alta) para cada característica. Una prioridad alta siempre reemplaza una baja, pero si una característica de alta prioridad esta en conflicto con otra de menor prioridad, esta ultima deberá reservarse.

Nota: Esta fase utiliza el enfoque F-SIG [19].

Fase V – Operacionalización de Características

Se evalúan las posibles operacionalizaciones de acuerdo a la prioridad de los "softgoals" reservados considerando sus dependencias. El resultado puede ser satisfactorio, regular, insatisfactorio. De acuerdo a esto la configuración de variabilidad de la prioridad de los "softgoals" deberá ser ajustada. Las operacionalizaciones o atributos de calidad resultantes serán los componentes arquitectónicos para la siguiente fase.

Nota: esta fase se inspira parcialmente en [32].

Fase VI – Integración de Componentes

El SIG obtenido se traduce en esta fase a un diagrama de componentes UML, representando así una AC. Esta arquitectura candidata se compone de un modelo de variabilidad con las variantes funcional EFM, y un modelo de variabilidad con las variantes no funcional representado por la Operacionalización del SIG, que representan las soluciones a las propiedades de calidad asociadas a las funcionalidades.

Nota: esta fase se inspira también en [32].

Fase VII – Modelo Arquitectónico de Referencia

Finalmente, se construye la AR a partir de la AC [32], incorporando como componentes la generalización de las variantes de la AC como puntos de variabilidad.

Nótese que la trazabilidad entre los modelos y artefactos obtenidos en las diferentes fases del proceso aquí planteado será considerada en la propuesta final del proceso completo de diseño, actualmente en desarrollo. Por otra parte debe observarse que las *Fases I* y *II* del proceso son relativas a capturar conocimiento sobre el dominio, el cual es necesario

para establecer el modelo de calidad y los estilos arquitectónicos más utilizados; en el proceso "bottom-up" este conocimiento puede ser extraído de los productos existentes estudiados utilizando técnicas de reingeniería, pero un análisis del dominio facilita esta tarea y por esto se han incluido como fases iniciales del proceso.

VI. CONCLUSIONES Y PERSPECTIVAS

De acuerdo a lo observado, la variabilidad de los requisitos de calidad está muy relacionada con la variabilidad de los requisitos funcionales y con el comportamiento global esperado del sistema; sin embargo esta trazabilidad es difícil de expresar y no se percibe claramente en muchos de los métodos estudiados. Al momento de especificar la variabilidad, es importante considerar un proceso sistemático que considere las mejores prácticas citadas anteriormente en los diferentes enfoques, tales como priorización y contribución entre funcionalidades, tratar explícitamente la variabilidad de los atributos de calidad y su trazabilidad en relación a las funcionalidades, profundizar estudios sobre análisis de "trade-offs" y la utilización de estándares para representar al modelo de variabilidad. Adicionalmente, no sólo se debe hacer análisis cualitativo, sino que se debe incluir en la especificación análisis cuantitativos, para lograr tener métricas tangibles y así asegurar la calidad de la producción industrial. Se pueden resaltar los métodos [12][23][29] por sus procesos bien documentados.

Finalmente, se ha presentado una primera propuesta, a partir del análisis realizado, que combina algunos de los enfoques estudiados, para obtener un modelo de variabilidad funcional y no funcional que considera la especificación de la calidad mediante el estándar ISO/IEC 25010 [11]. La formalización de esta primera propuesta está en vía de desarrollo; paralelamente se está definiendo un caso de estudio en el dominio de los sistemas de salud integrados con manejo de historias clínicas electrónicas para su aplicación y validación.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por los proyectos grupales DISofT No. 2011001343 del PEII-Fonacit; DesCCaP No. PG-03-8014-2011 y DARGRAF No. 03-8730-2013-1 del CDCH; además, por el Postgrado en Ciencias de la Computación, Facultad de Ciencias, Universidad Central de Venezuela y el Banco Central de Venezuela.

REFERENCIAS

[1] I. Sommerville, *Ingeniería de Software*. 9na Edición. Prentice Hall. 2011.

[2] K. Lee, K. Kang, and J. Lee, *Concepts and Guidelines of Feature Modeling for Product Line Software Engineering*, in proceedings of the 7th International Conference on Software Reuse: Methods, Techniques, and Tools, ISBN 3-540-43483-6, pp. 62–77, 2002.

[3] E. Berard, *Essays in Object-Oriented Software Engineering*. Prentice Hall, New York, NY, USA, 1992.

[4] F. Losavio, A. Matteo e I. Pacilli, *Proceso Dirigido por Objetivos para Análisis de Dominio bajo Estándares de Calidad*. Enl@ce Revista Venezolana de Información, Tecnología y Conocimiento, vol. 6, no. 3, pp. 13–30, 2009.

[5] G. Perrouin, *A Metamodel-Based Classification of Variability Modeling and Software Product Lines*, VARY Workshop, 2011.

[6] E. de Almeida, A. Alvaro, D. Lucredio, V. Garcia, and S. de Lamos Meira, *A Survey on Software Reuse Processes*, in proceedings of the International Conference on Information Reuse and Integration. IRI-IEEE, pp. 66–71, 2005.

[7] A. Helferich, G. Herzwurm, and S. Schockert, *Developing Portfolios of Enterprise Applications using Software Product Lines*, in proceedings of the Conference on Component Oriented Enterprise Applications (COEA), pp. 71–85, Erfurt, Germany, 2005.

[8] K. Pohl, G. Bockle, and F. van der Linden, *Software Product Line Engineering: Foundations, Principles, and Techniques*. Springer, 2005.

[9] L. Chen, M. Ali Babar, and N. Ali, *Variability Management in Software Product Lines: A Systematic Review*, Software Product Line Conference, pp. 81–90, Pittsburgh, PA, USA, 2009.

[10] B. Kitchenham and S. Charters, *Guidelines for Performing Systematic Literature Reviews in Software Engineering*. Keele University and University of Durham, Technical Report EBSE-2007–01, 2007.

[11] ISO/IEC FCD 25010, *Software Engineering – Software Product Quality Requirements and Evaluation (SQuaRE) – Quality Model*. International Organization for Standardization (ISO), 2011.

[12] O. Gurses, *Non-Functional Variability Management by Complementary Quality Modeling in a Software Product Line*, MSc Thesis, Graduate School of Natural and Applied Science of Middle East Technical University, 2010.

[13] F. Losavio and A. Matteo, *Reference Architecture Design Using Domain Quality View*, Journal of Software Engineering and Methodology, vol. 31, no. 1, pp. 47–61, 2013.

[14] L. Etxeberria and G. Sagardui, *Modeling Variation in Quality Attributes*, University of Mondragon, 2007.

[15] D. Benavides, S. Segura, and A. Ruiz, *Automated Analysis of Feature Models 20 Years Later: A Literature Review*. University of Seville, 2010.

[16] J. Kuusela and J. Savolainen, *Requirement Engineering for Product Families*, in proceedings of the 22nd International Conference on Software Engineering (ICSE), pp. 61–69, New York, NY, USA, 2000.

[17] H. Zhang, S. Jarzabek, and B. Yang, *Quality Prediction and Assessment for Product Lines*, in proceedings of the 15th International Conference on Advanced Information Systems Engineering. Springer-Verlag. vol. 2681, pp. 681–695, 2003.

[18] B. González, J. Leite, and J. Mylopoulus, *Visual Variability Analysis for Goal Models*, in proceedings of the 12th IEEE International Requirements Engineering Conference, pp. 198–207, 2004.

[19] S. Jarzabek, B. Yang, and S. Yoeun, *Addressing Quality Attributes in Domain Analysis for Product Lines*. IEEE Proceedings Software, vol. 153, no. 2, pp. 61–73, 2006.

[20] M. Sinnema, S. Deelstra, J. Nijhuis, and J. Bosch, *Modeling Dependencies in Product Families with COVAMOF*, in proceedings of the 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems, ISBN 0-7695-2546-6, pp. 298–307, 2006.

[21] M. Raatikainen, E. Niemela, V. Myllarniemi, and T. Mannisto, *Svamp, An Integrated Approach to Modeling Functional and Quality Variability*, in proceedings of the 2nd International Workshop on Variability Modeling of Software-Intensive Systems (VaMoS), pp. 89–96, 2008.

[22] R. Oberhauser, M. Medak, and J. Bartholdt, *Integrating Quality Modeling with Feature Modeling in Software Product Lines*. in proceedings of the Fourth International Conference on Software Engineering Advance, pp. 365–370, 2009.

[23] X. Peng, S. Lee, and W. Zhao, *Feature-Oriented Nonfunctional Requirements Analysis for Software Product Line*. Journal of Computer Science and Technology, vol. 24, no. 2, pp. 319–338, 2009.

[24] L. Chung, B. Nixon, E. Yu, J. Leite, and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*. Springer, Reading, Massachusetts, 2000.

[25] B. González, M. Laguna y J. Sampaio, *Análisis de Variabilidad con Modelos de Objetivos*. VII Workshop on Requirements Engineering (WER-2004), pp. 77–87, 2004.

[26] K. Kang, S. Cohen, J. Hess, W. Nowak, and S. Peterson, *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. Tech. Report, CMU/SEI-90-TR-21, SEI (Carnegie Mellon), Pittsburgh, PA 15213, 1990.

[27] P. Giorgini, J. Mylopoulus, and R. Sebastiani, *Goal Oriented Requirements Analysis and Reasoning in the Tropos Methodology*. Engineering Applications of Artificial Intelligence. vol. 18, no. 2, pp. 159–171, 2005.

[28] S. Supakkul and L. Chung, *Integrating FRs and NFRs: A Use Case and Goal Driven Approach*, 2nd ICSE, pp. 30–37, 2004.

[29] J. Bartholdt, R. Oberhauser, M. Medak, and A. Rytina, *Integrating Quality Modeling in Software Product Lines*. International Journal on Advances in Software, vol. 3, no. 1 & 2, pp. 161–174, 2010.

[30] D. Streitferdt, M. Riebisch, and I. Philippow, *Details of Formalized Relations in Feature Models Using ocl*, in proceedings of 10th IEEE International Conference on Engineering of Computer-Based Systems (ECBS 2003). USA IEEE Computer Society, pp. 45–54, 2003.

[31] F. Losavio, L. Chirinos, N. Lévy, and A. Ramdane-Cherif, *Quality Characteristics for Software Architecture*. Journal of Object Technology, vol. 2, no. 2, pp. 133–150, 2003.

[32] F. Losavio, O. Ordaz, N. Levy, and A. Baiotto, *Graph Modeling of a Refactoring Process for Product Line Architecture Design*, in proceedings of the XXXIX Latin American Computing Conference (CLEI 2013), vol. 1, Naiguatá, Venezuela, October 2013.

[33] N. Siegmund, *Scalable Prediction of Non-Functional Properties in Software Product Lines: Footprint and Memory Consumption*, Information and Software Technology, vol. 55, no. 3, pp. 491–507, 2012.

[34] A. von Rhein, *Strategies for Product-Line Verification*, in proceedings of the 35th International Conference on Software Engineering. San Francisco, CA, USA, 2013.

[35] S. Sepulveda y C. Cachero, *Requerimientos No Funcionales en las Líneas de Producto de Software y el Modelado de Características: Una Propuesta*. International Workshop on Software Engineering IWASE, vol. 3, Curicço, Chile, 2011.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 23-33, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Revisión Documental Sistemática en el Ámbito de la Ingeniería del Dominio para Líneas de Productos de Software Orientados a Servicios

Juan Herrera[1], Francisca Losavio[2], Alfredo Matteo[2], Oscar Ordaz[3]
jchr1982@gmail.com, francislosavio@gmail.com, alfredojose.matteo@gmail.com, oscarordaz55@gmail.com

[1] PFG Informática para la Gestión Social, Universidad Bolivariana de Venezuela, Caracas, Venezuela
[2] Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela
[3] Escuela de Matemática, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: Una Revisión Documental Sistemática (RDS) permite extraer conocimiento sobre un tema de investigación; esto se hace mediante preguntas adecuadas, a partir del gran volumen de información disponible en internet. RDS se enfoca en una pregunta de investigación que trata de identificar, evaluar, seleccionar y sintetizar evidencias de investigaciones de alta calidad relevantes para esa pregunta. El objetivo principal del trabajo es realizar una RDS en el tema de la Ingeniería del Dominio (ID) de Líneas de Productos de Software Orientados a Servicios (LPSOS), quien combina los enfoques de Líneas de Productos de Software (LPS) y de Arquitecturas Orientadas a Servicios o del inglés Service-Oriented Architecture (SOA). Esta investigación pretende determinar trabajos recientes existentes en la literatura en cuanto a actividades, artefactos y técnicas para un proceso de desarrollo del dominio, según Björner y de análisis del dominio según Pohl et al.. Como resultado de esta revisión se detectó, combinando los enfoques mencionados, que las fases principales de un proceso general de ID para LPSOS son: análisis, diseño e implementación, inspiradas en las fases definidas por Pohl et al. para la ID en un contexto de LPS. Dentro de estas fases se identificaron las actividades, artefactos y técnicas más usadas en la práctica. Estos resultados serán utilizados para definir un proceso sistemático de Análisis de Dominio para LPSOS, que además contemplará el tratamiento de requisitos de calidad, aspecto relevante para un contexto de producción industrial de software y no tratado sistemáticamente en los enfoques revisados.

Palabras Clave: Ingeniería del Dominio; Línea de Productos de Software; Arquitectura Orientada a Servicios; Línea de Productos de Software Orientadas a Servicios; Revisión Documental Sistemática; Calidad del Software.

Abstract: A Systematic Literature Review (SLR) allows to extract knowledge on a research topic; this is done through appropriate questions, from the large volume of information available on the internet. SLR is focused on a research question that tries to identify, appraise, select and synthesize high quality research evidence relevant to that question. The main goal of this work is to perform an SLR on the subject of Domain Engineering (DE) of Service-Oriented Software Products Line (SOSPL), who combines the approaches of Software Products Line (SPL) and Service-Oriented Architecture (SOA). This research aims to determine existent recent works in the literature regarding activities, artifacts, and techniques for development of the domain, according to Björner and domain analysis according to Pohl et al.. As a result of this review, main phases of a general DE process of for SOSPL were identified: analysis, design and implementation, inspired in the phases defined by Pohl et al. for DE in an SPL context. Within these phases activities, artifacts and techniques most used in practice were identified. These results will be used to define a systematic process of Domain Analysis for SOSPL, which also includes the treatment of quality requirements, relevant aspect for industrial software production the context, not systematically treated in the revised approaches.

Keywords: Domain Engineering; Software Products Line; Service-Oriented Architecture; Service-Oriented Software Products Line; Systematic Literature Review; Software Quality.

I. INTRODUCCIÓN

El desarrollo de sistemas de software de gran complejidad y de gran escala representa un desafío para los investigadores y desarrolladores en el área de la IS (Ingeniería de Software); surge entonces la tendencia al desarrollo basado en componentes reutilizables, para de ahí configurar nuevos sistemas con rapidez y de más bajo costo [1][2][3].

Desde finales de los ‘60, y aun más en la década de los ‘90, el paradigma de las LPS (Líneas de Productos de Software) ha

cobrado impulso en la industria del software. En lugar de desarrollar sistemas a partir de cero, estos deben ser construidos a partir de partes reutilizables. En vez de componer un sistema siempre de la misma manera, éste debería ser adaptado a las necesidades de los clientes (del inglés, "stakeholders"), donde éstos pueden elegir entre un conjunto de opciones de configuración. El enfoque de LPS permite la construcción de soluciones individuales basadas en un repositorio de componentes de software reutilizables. La necesidad de proporcionar soluciones individuales responde a diversas necesidades en el software con respecto a la funcionalidad, las plataformas destino y las propiedades no funcionales, como por ejemplo rendimiento y espacio de memoria [4].

Las LPS prometen distintos beneficios [4][5], de los cuales los más importantes son: a) adaptabilidad a las necesidades del cliente, b) reducción de costos, c) mejora de la calidad global y d) disminución del tiempo de comercialización. Este enfoque representa un gran desafío para los investigadores y desarrolladores en el área de la IS, principalmente por la tarea de desarrollar artefactos (activos o del inglés "assets") suficientemente genéricos para pensar en su reutilización y en la configuración de nuevos sistemas contextualizados acorde con las necesidades de los clientes [2][3].

Para el abordaje adecuado de este enfoque, la ILPS (Ingeniería de Líneas de Productos de Software) ofrece métodos y técnicas eficaces para la reutilización sistemática en el desarrollo de software con el fin de: a) apoyar arquitecturas de software configurables y b) permitir la personalización masiva de los sistemas de software [6][7]. Ha sido reconocida como un enfoque exitoso para la gestión de la variabilidad y la ingeniería de la reutilización. La ILPS consta de dos ciclos de vida principales: ID (Ingeniería del Dominio) e IA (Ingeniería de la Aplicación) [8]. La ID abarca el análisis y la identificación del ámbito de aplicación de la línea de productos, que incluye la captura de todo el conocimiento sobre el dominio de interés a través del modelado de partes comunes (del inglés "commonality") y de puntos de variación (del inglés "variability points") que están presentes en la arquitectura de la línea de productos.

Al considerar las fases del proceso de ID desde la perspectiva de las LPS [8], el *análisis del dominio* permite la captura y análisis del conocimiento sobre el dominio. El resultado principal del análisis del dominio es un artefacto denominado *modelo del dominio*. Este modelo representa las propiedades comunes y variables de la familia de sistemas en el dominio y las relaciones entre ellas. Conceptos del dominio y sus relaciones mutuas son analizados y modelados. Los conceptos del dominio representan el vocabulario del dominio y pueden ser representados por ontologías. Cada concepto luego es extendido por sus características comunes y variables y las dependencias entre ellos. Las características variables determinan el espacio de configuración para la familia de sistemas que se resuelve luego en la IA. El *diseño del dominio* produce la arquitectura abstracta genérica o de referencia (del inglés: Product Line Architecture, PLA) o RA (Reference Architecture) para la familia de sistemas, de acuerdo con los patrones arquitectónicos comúnmente aceptados (modelo-vista-controlador, en capas, etc.). Un ciclo completo de ID construye el conocimiento de las necesidades específicas de los diferentes grupos de interés para los que se lleva a cabo la adaptación y configuración de la LPS.

Por otra parte, la *Arquitectura Orientada a Servicios* (del inglés: Service Oriented Architecture, SOA) [9][10] es una arquitectura centrada en la comunicación en redes a través de servidores Web que actúan como intermediarios para satisfacer demandas de servicios por clientes distribuidos en ubicaciones geográficamente distantes. SOA ha cobrado enorme auge en la práctica reciente de desarrollo de software rápido, a bajo costo y es ampliamente tratada actualmente a nivel de investigación en la ingeniería de software; constituye un paradigma para el diseño, desarrollo e implementación de aplicaciones de computación distribuida e independiente de la tecnología basada en estándares para el descubrimiento, el enlazado y ensamblaje de servicios de software débilmente acoplados [11]. Un *servicio* se define como una unidad discreta de la funcionalidad de negocio que está disponible a través de un contrato de servicios [12]; puede ser visto como un componente de software reutilizable. SOA es una arquitectura para la construcción de soluciones empresariales basadas en servicios [13]. Particularmente se ocupa de la construcción independiente de servicios alineados al negocio que pueden ser combinados en procesos de negocio de alto nivel y soluciones computacionales en el contexto empresarial. El valor real de SOA no es solo la provisión adecuada de los servicios, sino más bien es cuando los servicios reutilizables se combinan para crear procesos de negocios ágiles y flexibles. SOA ha demostrado su rentabilidad en el desarrollo de sistemas de software interoperables, reutilizables, adaptables y de rápido desarrollo, por lo cual existen importantes ventajas mutuas en la convergencia de SOA y LPS [1][14][15][16][17][18][19], dando origen al enfoque de LPSOS (Líneas de Productos de Software Orientadas a Servicios).

En todos los puntos antes tratados, el aseguramiento de la calidad del sistema como producto de software, es una actividad crucial para el éxito, ya que se ubican en un contexto de producción industrial; por lo tanto, cuando se trata del desarrollo de LPS, la reutilización masiva de los activos de software hace que los atributos de calidad (propiedades medibles de un artefacto de software) impacten en la calidad de todos los productos de la LPS [20]. La evaluación de la calidad es una tarea difícil en los sistemas de software monolíticos, y lo es aún más cuando se trata de LPSOS, ya que requiere analizar las características de calidad de una LPS bajo SOA, considerando propiedades funcionales y no funcionales comunes y variantes [21][22]. Por otra parte, la evaluación de la calidad es una parte esencial en la optimización y configuración de la LPS, ya que debe proporcionar métricas cuantitativas globales para la calidad de la familia de sistemas, generalmente basadas en la especificación de la arquitectura, como por ejemplo: costo, desempeño, disponibilidad, confiabilidad, escalabilidad y seguridad [23], pero también específicas para un sistema miembro de la familia [24].

Ahora bien, en un contexto de LPSOS, particularmente durante el ciclo de ID, la determinación de la calidad del servicio (del inglés: Quality of Service, QoS) es fundamental, debido a que varía acorde a características individuales, por lo cual los ingenieros del dominio deben asegurar que los servicios que

van a formar parte del sistema satisfacen los rangos de valores establecidos por los QoS solicitados por las partes interesadas [23].

En los estudios anteriormente referenciados se han detectado las siguientes evidencias de problemas aún no completamente resueltos, para algunos de los cuales este trabajo pretende proporcionar una solución:

- No se describen en detalle las actividades a realizar para la adquisición del conocimiento del dominio.
- No se describen diferentes perspectivas en el dominio que puede ser analizadas, por ejemplo, las facetas del dominio, utilizadas en el enfoque de Bjorner [25].
- Los requisitos funcionales son los abordados en general, pero los requisitos no funcionales y/o requisitos de calidad son dejados de lado, desde las siguientes perspectivas [23]: características de calidad específicas de la línea de productos (mantenibilidad, complejidad, etc.) y características de calidad específica del dominio o calidad global (disponibilidad, seguridad, etc.).
- La determinación de las características de calidad no es realizada en las etapas tempranas del proceso de ID (análisis del dominio); se ha detectado que son de vital importancia para derivar de forma adecuada el conjunto de productos de la LPSOS.

Es de particular interés en el contexto de esta investigación, que se centra básicamente en la fase de Análisis del Dominio de la ID, por ser la más importante y previa a la derivación de los productos concretos de la LPS en la IA, identificar las actividades, artefactos y técnicas utilizadas en un proceso completo de análisis del dominio para el desarrollo de las LPSOS, diferenciándolas: a) por los artefactos de entrada y de salida, b) por el enfoque de diseño utilizado (proactivo o reactivo) para la arquitectura, con el fin de describir como se analiza y realiza el proceso de análisis del dominio en la ID y c) también es de especial interés identificar las técnicas (UML (Unified Modeling Language), FODA (Feature-Oriented Domain Analysis), etc.) utilizadas como notaciones para sus principales artefactos.

Una RDS (Revisión Documental Sistemática) [26] permite extraer el conocimiento sobre el estado del arte en un tema específico de investigación; este conocimiento es extraído, mediante preguntas adecuadas, a partir de un gran volumen de información disponible. En consecuencia una RDS de la literatura que aborda el tema de la ID de LPSOS es indispensable para determinar los alcances y limitaciones de las propuestas encontradas. El objetivo de este trabajo es utilizar la RDS para identificar las actividades, artefactos y técnicas comúnmente utilizadas en la ID encontrados en la literatura reciente, para luego integrarlos para definir un proceso único de análisis del dominio para LPSOS incorporando también aquellas actividades y/o artefactos faltantes.

Este trabajo está estructurado de la siguiente manera, además de esta introducción y las conclusiones: en la Sección II se presentan los enfoques a considerar; en la Sección III se presenta el análisis de la revisión documental sistemática realizada. En la Sección IV se discuten los resultados obtenidos. Finalmente, en la Sección V se da un resumen de los aportes de la investigación.

II. ENFOQUES A CONSIDERAR

A. ID (Ingeniería del Dominio)

Un *dominio* es un área de conocimiento que utiliza conceptos comunes para la descripción de los fenómenos, requisitos, problemas, capacidades y soluciones. Se asocia generalmente con una terminología bien definida o parcialmente definida. Esta terminología se refiere a los conceptos básicos del dominio, sus definiciones (es decir, sus significados semánticos), y sus relaciones. También puede referirse a las conductas que se desean, prohibidas, o percibidas dentro del dominio [27]. Un *dominio de aplicación* comprende cualquier aspecto en el cual la computación pueda aplicarse. Según Bjorner [25], básicamente existen tres clases de dominios de aplicación:

- La clase de aplicaciones que pueden caracterizarse por el apoyo a la enseñanza o el estudio de un campo: el software educativo o de formación, software experimental de demostración de teoremas, o similares.
- La clase de aplicaciones que pueden caracterizarse por el apoyo al desarrollo de sistemas informáticos propios: compiladores, sistemas operativos, sistemas de gestión de bases de datos, sistemas de comunicación de datos, etc.
- La clase de aplicaciones que pueden caracterizarse por el apoyo al desarrollo de software comercial o industrial, esto incluye los sistemas de información en general y las aplicaciones empresariales, entre otras.

Particularmente, la última clase de estos dominios de aplicación son los de interés en el trabajo de Bjorner [25]. Por ello, la selección del trabajo de este autor es argumentada por la razón que las aplicaciones implementadas mediante SOA fueron diseñadas con la intención de apoyar a través de esta arquitectura basada en un intermediario para la comunicación, los dominios de aplicación relacionados con los sistemas de información gerencial y/o empresarial que utilizan un estilo clásico de capas, como por ejemplo, capa de presentación, lógica y datos.

La ID es el conjunto de actividades cuyo objetivo es desarrollar, mantener y administrar la creación y evolución de los dominios [25][27]. Ha sido de especial interés para los sistemas de información y de las comunidades de ingeniería de software, por varias razones: a) la necesidad de mantener y reutilizar los conocimientos existentes, b) la necesidad de gestionar los crecientes requisitos de la variabilidad de la información y de los sistemas de software, y c) la necesidad de obtener, formalizar, y compartir conocimientos especializados en diferentes ámbitos, en evolución [27].

La ID como disciplina tiene una importancia práctica, ya que debe proporcionar los métodos y técnicas que pueden ayudar a reducir el tiempo de entrega al mercado, los costos de desarrollo y los riesgos en los proyectos, además que deben ayudar a mejorar la calidad del producto y el rendimiento sobre

una base consistente [27]. La ID es utilizada, investigada y estudiada en diversas áreas, principalmente:

- ILPS (Ingeniería de línea de productos de software)
- ILED (Ingeniería de lenguaje específico de dominio)
- Modelado conceptual.

Según Bjorner [25], antes de diseñar el software, se deben comprender sus requerimientos, y antes que se puedan desarrollar los requerimientos, se debe comprender el dominio de la aplicación en que se va a construir. Para comprender el domino de la aplicación, se debe realizar una etapa de análisis del dominio desde las perspectivas de las facetas del dominio bajo la óptica de los grupos de interés. Es decir, cada grupo de interés tendrá características particulares en cada una de las facetas que conforma el dominio de la aplicación. La ID reduce el esfuerzo de trabajo relacionado con algunos aspectos de la IS, especialmente los vinculados con la ingeniería de requisitos.

La Tabla I muestra la relación entre las fases de la ID expresada por Bjorner [25] y las fases expresadas por Pohl at al. [8]. Esto se hace con la intención de mostrar como los conceptos expresados por Bjorner en ID, complementan los de Pohl bajo el enfoque LPS. El enfoque propuesto por Bjorner fue seleccionado debido a que las fases del proceso para el desarrollo del dominio son claramente definidas y argumentadas, y debido al gran nivel de detalle en la descripción de las actividades realizadas, expresadas mediante las facetas (o vistas) del dominio.

El análisis del dominio basado en las facetas del domino desde las perspectiva de los stakeholders, permitirá conocer el dominio de la aplicación al considerar de forma más adecuada los aspectos funcionales y no funcionales (requisitos de calidad) que deben ser satisfechas desde diferentes vistas.

Tabla I: Relación entre las Fases de la ID según Dines Bjorner [25] y Pohl et al. [8]

Perspectiva de Bjorner [25]	Perspectiva de Pohl et al. [8]
Proceso de desarrollo del dominio: Identificación de los stakeholders; Adquisición del dominio; Análisis del dominio y formación de conceptos	*Análisis del dominio*: modelo de procesos de negocios; modelo de características
Modelado del dominio (facetas del dominio): Procesos del negocio; Intrínsecos; Tecnologías soportadas; Organización y gerencia; Reglas y regulaciones; Scripts (guiones); Comportamiento humano	*Diseño del dominio*: arquitectura genérica abstracta; arquitectura de referencia (top-down) y/o arquitectura de línea de productos (botton-up)

B. Proceso para el Desarrollo del Dominio, dentro de la ID Propuesto por Bjorner

Contempla las siguientes fases o subprocesos:

- identificación de los stakeholders
- adquisición del conocimiento del dominio
- análisis del conocimiento del dominio y formación de conceptos
- modelado del dominio (diseño del dominio)
- verificación y validación del dominio y
- teoría del dominio.

1) Los Stakeholders del Dominio y sus Perspectivas: Los stakeholders del dominio se refieren a una persona, o grupo de personas unidas de alguna manera, o por una institución, empresa o un grupo de este tipo, caracterizado por su interés común en el dominio o dependencia del dominio. Por la perspectiva de los stakeholders del dominio lo comprendemos a él, o una comprensión del dominio compartido por un grupo de stakeholders específicamente identificado; una vista que puede diferir de un grupo de interés a otro grupo de interés del mismo dominio. La identificación de las perspectivas de los stakeholders (por ejemplo, vistas) incorpora el desarrollo de principios, técnicas y herramientas. Sin la clara identificación y enlace con todas las partes relevantes interesadas del dominio no se puede aspirar a construir un modelo de dominio creíble.

2) Adquisición del Dominio: La adquisición del dominio comprende el proceso de obtención de datos sobre el dominio, esto es, obtener (capturar) datos (hechos) de las partes interesadas del dominio, de redactar las descripciones acerca de ellos, y de su estructura (es decir, su organización y/o clasificación aproximada de estas descripciones). Esto incluye la recolección de datos, de la literatura y de nuestras observaciones acerca del conocimiento del dominio. Este conocimiento incluye entidades fenomenológicas, funciones, eventos y comportamientos.

3) Análisis y del Dominio y Formación de Conceptos: El análisis del dominio comprende el estudio de la adquisición del conocimiento del dominio, con los siguientes objetivos: a) descubrir inconsistencias, conflictos y la falta de completitud dentro de ellas, b) formar conceptos de estas declaraciones de adquisición del conocimiento del dominio. La formación de conceptos del dominio comprende la abstracción de los fenómenos del dominio, en conceptos.

4) Modelado del Dominio (Modelo de Facetas del Dominio): Las facetas del dominio comprende un conjunto finito de formas genéricas de analizar un dominio, cada forma genérica representa una vista particular del dominio o faceta del dominio, de tal modo que las diferentes facetas abarcan conceptualmente diferentes vistas, y juntas conforman el dominio. A continuacion se enuncian las pricipales facetas del dominio según [25]:

- Procesos del negocio (conjuntos de acciones que se realizan en el dominio)
- Características intrínsecas (lo que es común a todas las facetas)
- Tecnologías de apoyo (implementación de las facetas)
- Organización y gestión

- Reglas y regulaciones (normativas)
- Comportamiento del recurso humano (stakeholders)

Un modelo de dominio se refiere al conjunto de uno o más modelos acordes con las facetas del dominio, estas pueden ser reescritas (y posiblemente formalizadas) en un modelo consolidado (un único modelo en donde confluyen todas las facetas).

Para modelar una faceta del dominio, primero hay que adquirirla; entonces se debe analizar lo que se ha adquirido, y formar conceptos de lo que se ha analizado, para luego describirla: a) aproximadamente, b) en términos de las entidades del dominio (terminología), c) de forma narrativa y d) posiblemente formalizando la faceta [25].

5) Validación del Dominio: La validación del dominio se refiere a la garantía, con los stakeholders, en particular los clientes, que las descripciones del dominio que se producen como resultado de la adquisición del dominio, análisis de dominio, la formación de conceptos y el modelado del dominio es acorde con la forma en que los grupos de interés miran el dominio.

C. Arquitectura Orientada a Servicios (SOA)

SOA se ha convertido en un concepto ampliamente estudiado y utilizado en la investigación y la práctica de la ingeniería de software. SOA es una arquitectura para el diseño, desarrollo e implementación de aplicaciones de computación distribuida e independiente de la tecnología basada en estándares para el descubrimiento, el enlazado y ensamblaje de servicios de software débilmente acoplados que son transmitidos mediante una red [11].

SOA proporciona el marco conceptual para la realización de sistemas orientados al servicio al permitir que los sistemas de software puedan ser compuestos y reconfigurados dinámicamente usando servicios detectables en la red. Desde el punto de vista de la ILPS, este modelo de despliegue promete beneficios significativos sobre el modelo tradicional de implementación del software como un producto: en primer lugar, la naturaleza dinámica de SOA significa que puede soportar las necesidades y expectativas del usuario en un entorno de cambios continuos; en segundo lugar, los servicios pueden ser combinados con diferentes configuraciones y contextos simplificando el despliegue de variantes de los productos adaptados a las necesidades de los distintos clientes, pero sobre todo basados en los mismos servicios básicos [28].

Particularmente SOA se ocupa de la construcción independiente de servicios alineados al negocio que pueden ser combinados en significativos procesos de negocio de alto nivel y soluciones en el contexto empresarial [29]. El valor real de SOA no es solo la construcción de los servicios, sino más bien cuando los servicios reutilizables se combinan para crear procesos de negocios ágiles y flexibles. La agilidad y flexibilidad se producen cuando los nuevos procesos pueden rápidamente y eficientemente ser creados a partir del conjunto de servicios existente [12].

SOA permite el ensamblado, orquestación y el mantenimiento de las soluciones empresariales para reaccionar rápidamente a los cambiantes requisitos empresariales [30].

D. Calidad del Software

La calidad ha sido típicamente definida como un nivel de excelencia, de conformidad con las especificaciones, de satisfacción de requisitos, de atributos distintivos, de estar libre de defectos, deficiencias y de las significativas variaciones para cumplir con las expectativas del cliente [31][32].

Establecer un modelo de calidad para el desarrollo del dominio de la aplicación que involucra los procesos de identificación de los stakeholders, adquisición del dominio, análisis del dominio y modelado del dominio, planteados en Bjorner [25], garantizaría la validez y confiabilidad del dominio desarrollado. El aseguramiento de la calidad del producto es una actividad crucial para el éxito de la industria del software, pero es, si cabe, más importante cuando se trata del desarrollo de líneas de producto software, dado que la reutilización masiva de activos de software hace que los atributos de calidad (propiedades medibles de un artefacto de software) de estos activos impacten en la calidad de todos los productos de una línea de producto [17].

La calidad se ha convertido en un atributo crítico de los productos de software ya que su ausencia produce pérdidas financieras, de salud, y a veces de vida. Al mismo tiempo la definición, o alcance, del dominio de la calidad del software ha evolucionado continuamente desde una perspectiva técnica a una perspectiva que abarca aspectos humanos tales como la facilidad de uso y la satisfacción [21][24].

III. Revisión Documental Sistemática

La RDS se llevó a cabo en el primer y segundo trimestre del 2014. Kitchenham indica que la gestión del proceso de revisión sistemática se fundamenta en tres fases principales: Planificación de la revisión, Realización de la revisión e Informe de la revisión [26]. En la Tabla II se describen las fases, las etapas que incluyen cada fase y los artefactos generados en el proceso de revisión sistemática.

Tabla II: Proceso de Revisión Documental Sistemática

Fases	Etapas	Artefactos
Planificación de la revisión	Justificación de la necesidad de una revisión.	Protocolo
	Especificación de la(s) interrogante(s) de investigación.	
	Desarrollo del protocolo de revisión.	
Realización de la revisión	Identificación de los estudios relacionados.	Estudios aceptados
	Selección de los estudios primarios.	
	Evaluación de la calidad de los estudios.	
	Extracción de los datos.	Modelo de datos
	Síntesis de los datos.	
Informe de la revisión	Especificación de los mecanismos de difusión.	Tablas resumen
	Presentación de los resultados	

A. Planificación de la Revisión

Esta fase define los objetivos de la investigación y el protocolo en que la revisión se ejecutará.

1) Identificación de la Necesidad para una Revisión Sistemática: Estrategias y actividades utilizadas durante el proceso de ID para el desarrollo de LPSOS, con el fin de determinar los alcances y limitaciones de las propuestas encontradas.

2) Objetivos: Revisar los procesos de desarrollo actuales en el área de la ID para el enfoque de LPS para el desarrollo de Familias de Productos Orientadas a Servicios; Identificar la estrategias y actividades utilizadas en las fases de análisis, diseño e implementación durante el proceso de ID en el desarrollo de LPSOS; Recolectar evidencia acerca de las investigaciones actuales que indiquen sus implicaciones en la práctica; Identificar problemas pendientes por resolver y posibles áreas para la mejora de estos procesos.

3) Desarrollo del Protocolo: El protocolo es un plan o conjunto de pasos a seguir en un estudio, constituido por las preguntas de investigación, las estrategias de búsqueda, criterios y procesos de selección, los criterios de evaluación de la calidad, el modelo y proceso de extracción de datos, y el plan de análisis de los datos.

Las preguntas de investigación tienen el objetivo de encontrar todos los estudios para comprender y resumir las evidencias acerca de los enfoques propuestos o utilizados durante el desarrollo del proceso de ID, las siguientes PIs (Preguntas de Investigación) fueron consideradas:

- PI1: ¿Cuáles son las actividades comúnmente realizadas en las etapas de análisis, diseño e implementación en el proceso de ID bajo el enfoque de LPSOS?
- PI2: Desde el punto de vista de las LPS implementadas con SOA. ¿Cuáles son los artefactos comúnmente utilizados y técnicas utilizadas para su construcción en la fase de ID para LPSOS?
- PI3: Desde el punto de vista de la fase de ID. ¿Cuál es el enfoque utilizado para el diseño de la arquitectura de referencia y/o arquitectura de líneas de productos (desde la perspectiva de la investigación realizada en [33]: enfoques proactivo y reactivo)?
- PI4: ¿Existen vistas de calidad que tomen en consideración algún estándar o modelo de calidad del producto (ISO9126-1, ISO25010 [32]) que tomen en consideración las características de calidad particulares para el diseño de LPSOS?
- PI5: ¿Existen herramientas de análisis o diseño que apoyen el proceso de ID para el desarrollo de LPSOS?

Con relación a la estrategia de búsqueda, primero se establecieron las fuentes de búsqueda para encontrar los estudios primarios, particularmente se seleccionaron los siguientes gestores de búsqueda: Google Academic, Scirus, ResearchGate y ScienceDirect.

Para establecer la estrategia de búsqueda, teniendo en cuenta las preguntas de investigación, inicialmente se identificaron las principales palabras de acuerdo a la terminología utilizada por Bjorner [25], que a continuación se enuncian: Software Product Line, Service-Oriented Architecture, Domain Engineering, Domain Stakeholders, Domain Acquisition, Domain Analysis, Domain Design, Domain Modeling, Domain Verification, Domain Validation, Analysis Tool.

Con relación a los criterios de inclusión y exclusión, aquí se establecen los criterios por los cuales los estudios serán evaluados para decidir si deben ser seleccionados o no en el contexto de la revisión sistemática.

Por lo tanto, los CI (Criterios de Inclusión) que se utilizan para incluir los estudios relevantes en esta RDS:

- CI1: Que el artículo aborde el área de la Ingeniería de Dominio para el desarrollo de Líneas de Productos de Software para Familias de Productos Orientados a Servicios utilizando SOA.

Los CE (Criterios de Exclusión) que se utilizan para excluir los estudios que no contribuyen a responder a las interrogantes de la investigación son:

- CE1: Que el artículo no aborde la integración de LPS y SOA como alternativa para el desarrollo de productos de software
- CE2: Que el articulo este escrito en un lenguaje diferente al inglés, portugués y español
- CE3: Que el artículo no esté disponible en el formato PDF
- CE4: Que el artículo tenga como contenido una presentación en un congreso o evento realizado por los autores.
- CE5: Que el articulo este repetido (el mismo artículo, pero con dos nombres diferentes), en este caso solo se acepta un solo ejemplar.

Con respecto a la extracción de datos y método de síntesis, se prevé la construcción de tablas para la obtención de datos relacionados con las preguntas de investigación, en caso de que sea aplicable.

En la selección inicial de las publicaciones de entre los resultados de búsqueda, se dio lectura a los títulos y resúmenes de las publicaciones encontradas. En la Figura 1 se muestra el modelo de datos inicial utilizado en la revisión sistemática.

Las publicaciones fueron almacenadas en carpetas independientes de acuerdo a la expresión de búsqueda utilizada, de esta forma facilitar la ubicación de los estudios para realizar el análisis con mayor fluidez y comodidad.

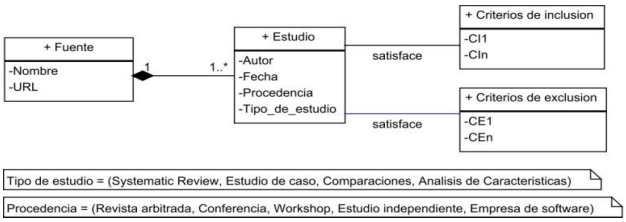


Figura 1: Modelo de Datos Inicial para la Revisión Sistemática

Del conjunto de publicaciones seleccionadas se procede a la extracción de la información relevante utilizando el modelo de datos que será utilizado en las fases de realización e informe de la revisión. En la recolección de datos, las citas textuales de los artículos analizados se conservarán en el idioma original (inglés) para evitar el posible sesgo al traducir y posteriormente parafrasear según la interpretación del investigador sobre la cita leída. Esto permite una vez realizados los aportes a la investigación, poder referirse a las citas por posibles malas interpretaciones de las mismas, para validar y para futuras referencias. No obstante, al final de la revisión sistemática los resultados finales reflejados en la fase de informe de la revisión se colocarán en español que es el idioma nativo del investigador.

Para estandarizar la forma en que la información estará representada, se definió un formato para recopilar datos de los estudios seleccionados. La Tabla III muestra el formato que será utilizado para registrar los resultados de la extracción de los datos.

Tabla III: Formato para la Recolección de Datos

Estudio				
Definición ID				
Etapas	***Actividades***	***Propósito***	***Artefactos de entrada***	***Artefactos de salida***
Análisis				
Diseño				
Implementación				
Enfoque				
Comentarios				

La Tabla IV describe las propiedades de extracción que serán utilizados a lo largo de la revisión sistemática.

Tabla IV: Propiedades de Extracción

Propiedades	**Descripción**
Estudio	Identificación del artículo objeto de estudio.
Definición	Concepto utilizado en el estudio para definir la ID.
Etapas	Indica las etapas utilizadas en los trabajos analizados para descubrir el proceso de desarrollo del dominio.
Actividades	Indica las actividades realizadas en cada una de las etapas anteriores.
Propósito	Objetivo o meta propuesta por la actividad a realizar.
Artefactos de entrada	Artefactos utilizados como insumos para poder realizar una actividad.
Artefactos de salida	Artefactos producidos como consecuencia de haber realizado una actividad.
Enfoque	Enfoque utilizado para el desarrollo del dominio; puede ser Proactivo o Reactivo.
Comentarios	Aspectos de interés reflejados en la investigación.

B. Realización de la Revisión

En esta fase, los estudios son identificados, seleccionados y evaluados de acuerdo con el protocolo previamente establecido. Como resultado de la búsqueda 231 estudios fueron localizados, de los cuales solo 67 estudios fueron aceptados para su análisis.

Para la selección final de los estudios aceptados se realizó un proceso filtrado que consistió en la aplicación de los criterios de inclusión y de exclusión, en la Tabla V se muestra un resumen de los resultados indicando la cantidad en números de los estudios aceptados y rechazados.

Tabla V: Relación de Estudios Localizados y Aceptados según la Fuente de Datos

	Artículos	
Fuentes de datos	***Aceptados***	***Localizados***
Google Academic	20	36
Scirus	30	167
ResearchGate	8	10
ScienceDirect	9	18
Total	**67**	231

En los estudios aceptados se descartaron los artículos repetidos donde los identificadores de los archivos (nombres) son iguales; también se descartaron aquellos artículos, que al examinar sus contenidos se encontró que eran idénticos, aun cuando sus identificadores eran diferentes.

A continuación en las Tablas VI y VII se muestra la distribución de los estudios aceptados agrupados por categorías o ámbitos de análisis, basados en la terminología utilizada en [25] en las etapas de desarrollo del dominio de aplicación.

Tabla VI: Etapas (Categorías) para el Desarrollo del Dominio [25]

Desarrollo del Dominio		
	Categorías de análisis	***Total***
1	Stakeholders Identification	2
2	Domain Acquisition	2
3	Domain Analysis and Concept Formation	32
4	Domain Modeling: (domain design)	18
5	Domain Validation and Verification	0
6	Domain Theory	0
7	Domain Documentation	1
Total		**55**

Tabla VII: Otras Categorías de Interés para la Investigación

Otras categorías de análisis	**Total**
Domain-Driven Design	2
Quality Model	1
Analysis Tool	0
Design Tool	0
Domain Architecture	9
Total	**12**

A continuación, un análisis más detallado se llevó a cabo en cada estudio incluido y el informe de la revisión.

C. Informe de la Revisión

En esta última fase, se presentan los resultados analíticos de la revisión sistemática. La extracción de los datos obtenidos y la síntesis del conocimiento considerando cada pregunta de investigación se discute a continuación.

Las respuestas a las preguntas de investigación versan sobre los aportes de la investigación realizada por Dines Bjorner [25] para el desarrollo del dominio de propósito general, del trabajo realizado por Pohl et al. [8] sobre el procesos de ID específicamente para las LPS, y de los trabajos realizados en [3][14][16][17][34][35][36][37][38][39] vinculados con las LPS para el desarrollo de FPOS (Familias de Producto Orientados a Servicios), que particularmente utilizan como arquitectura de implementación la SOA (Arquitectura Orientada a Servicios).

1) Con Relación a la Pregunta PI1y PI2: ¿Cuáles son las actividades comúnmente realizadas en las etapas de análisis, diseño e implementación en el proceso de ID bajo el enfoque de LPSOS?

Para responder a estas dos preguntas, se utilizaron las siguientes dos estrategias; primero, de los estudios aceptados inicialmente, se seleccionaron aquellos estudios que abordan o describen un proceso de desarrollo para LPSOS; segundo, para representar los resultados del análisis de estos estudios, y mostrarlos de una forma más amigable, se utilizó un modelo de datos para la sistematización de los datos. De los 67 estudios aceptados, solo 10 estudios fueron seleccionados, debido a que abordan y/o describen un proceso de desarrollo para la construcción de LPSOS, que es el objetivo principal de interés del estudio.

Las Tablas VIII, IX y X describen el conjunto de actividades, artefactos generados y técnicas utilizadas respectivamente en la construcción de LPSOS a partir de los estudios analizados: [3][14][16][17][34][35][36][37][38][39] vinculados con las LPS para el desarrollo de Familias de Producto Orientados a Servicios. Los 10 estudios analizados abordan el proceso de ID bajo el enfoque de LPSOS basándose en las siguientes 3 etapas: análisis, diseño e implementación.

Tabla VIII: Actividades Realizadas en las Fases de Análisis, Diseño e Implementación de la LPSOS

Etapas	Actividades
Análisis	Análisis de factibilidad de la LPS Análisis del dominio Análisis de características: similitudes y variabilidades de la LPS Análisis de los activos existentes Análisis de los requerimientos de la familia
Diseño	Identificación de los elementos arquitecturales Especificación de la arquitectura Definición de la técnica de implementación de la variabilidad
Implementación	Construcción de los componentes/servicios Plan de producción *(Roadmap)* de los productos de la LPS

Tabla IX: Artefactos Desarrollados en las Fases de Análisis, Diseño e Implementación de la LPSOS

Etapas	Artefactos
Análisis	Documento de factibilidad Modelo conceptual del dominio Modelo de características Listado de componentes/servicios disponibles Tabla de identificación de requerimientos
Diseño	Servicios/Componentes Arquitectura de referencia Modelo de configuración Modelo de componentes/servicios Modelo de estado Modelo de colaboración, interacción, comunicación. Mecanismos de implementación de la variabilidad.
Implementación	Componentes SPL desarrollados, reutilizados, o comprados a terceros. Composición de los componentes/servicios

Tabla X: Técnicas Utilizadas en las Fases de Análisis, Diseño e Implementación de la LPSOS

Etapas	Técnicas
Análisis	Diagrama de clases (UML) Notación de Modelado de Procesos de Negocios (BPMN) Diagrama de actividades (UML) Diagrama de Casos de Uso (UML) FODA, variantes de FODA Tablas Requisitos Funcionales – No Funcionales
Diseño	Service-Oriented Architecture (SOA) Service-Component Architecture (SCA) Vistas arquitecturales: Diagrama de clases (UML), Diagrama de componentes (UML), Diagrama de estado (UML), Diagrama de colaboración (UML). Lenguaje de Variabilidad Común (LVC) Mecanismos de configuración, generación
Implementación	SCA, SOA Orquestado/configuración/composición

2) Con Relación a la Pregunta "PI3": ¿Desde el punto de vista de la fase de ID, cual es el enfoque utilizado para el diseño de la arquitectura de referencia y/o arquitectura de líneas de productos?

Desde el punto de vista de la fase de ID, el enfoque utilizado para el diseño de la arquitectura de referencia y/o arquitectura de líneas de productos (desde la perspectiva de la investigación realizada en [33] por la mayoría de los estudios analizados fue el enfoque proactivo.

3) Con Relación a la Pregunta "PI4": ¿Existen vistas de calidad que tomen en consideración algún estándar o modelo de calidad del producto?

No existe evidencia de algún estudio que proponga o utilice alguna vista de calidad, modelo de calidad o implemente algún estándar de calidad del producto, utilizando ambos enfoques: LPS y SOA. Solo un trabajo [23] implementa estrategias que incorporan características de calidad en un enfoque LPSOS, es decir que consideran la calidad desde la perspectiva de QoS para la concepción de la LPS y toman en consideración la arquitectura de implementación SOA, sin embargo, aunque esta investigación hace un análisis superficial del modelo de calidad ISO9126-1 y del ISO25010 [32], argumentando que debe adecuarse a las particulares características de las LPSOS, no se propone una vista de calidad basada en alguno de los estándares antes mencionados.

Por otro lado, se evidenciaron dos trabajos independientes que abordan o proponen un modelo de calidad, uno de ellos describe un procedimiento para incorporar un modelo de calidad para una LPS [40], el segundo si aborda un modelo calidad basado en el ISO25010, pero solo para SOA bajo el paradigma de la Computación Orientada a Servicios (del inglés: Service-Oriented Computing, SOC) [41].

4) Con Relación a la Pregunta "PI5": ¿Existen herramientas de análisis o diseño que apoyen el proceso de ID para el desarrollo de LPSOS?

No se describen herramientas y/o prototipos que apoyen o aborden el proceso de desarrollo de la LPSOS, ni que utilicen alguna de vista de calidad del dominio aplicando un modelo de calidad o estándar (ISO9126, ISO25010, entre otros); sin embargo existen algunos estudios que abordan el proceso, por ejemplo desde la perspectiva de la LPS [5][20][42], particularmente se encuentra una herramienta en línea a través del enlace: www.splot-research.org.

Por otra parte, en el estudio [43] que aborda el análisis de las características de las herramientas del tipo open source que apoyan procesos de LPS, muestra que la mayoría de las propuestas se basan o parten del modelado de las características de la LPS [44][45][46].

IV. Discusión

Es conveniente y necesario describir aquellos aspectos que resaltan de cada uno de los 10 procesos analizados, aparte de las actividades realizadas en las etapas de análisis, diseño e implementación del dominio; entre ellas se destacan las siguientes evidencias:

Existen muy pocos estudios que abordan adecuadamente la identificación de los stakeholders (partes interesadas del dominio).

Particularmente no se identifican claramente los stakeholder (actores) involucrados para efectuar la adquisición del dominio, aspecto fundamental según Bjorner [25], para obtener una visión global del dominio de la aplicación.

Solo algunos estudios [3][36] identifican los roles de los stakeholders y en que etapas del proceso de desarrollo del dominio actúan o toman decisiones; pero solo en las fases de análisis, diseño e implementación del dominio.

También existen pocos estudios que aborden algún método, técnica o procedimiento relacionado con la adquisición del dominio.

En todos los estudios seleccionados para la investigación se observó que ninguno valida y/o verifica el dominio desarrollado por los ingenieros de software (arquitectura de referencia, componentes/servicios) contra el dominio especificado por los expertos del dominio del problema (conocedores de los procesos de negocios en la organización).

La mayoría de los estudios analizados, solo abordan las fases de análisis y modelado del dominio.

Solo algunos describen como implementar el dominio (arquitectura de referencia [47]).

La arquitectura de referencia descrita por los autores, no toman en cuenta alguna vista de calidad del domino, como el descrito en [48], o toman en consideración algún estándar de calidad, por ejemplo: ISO9126 o la versión actualizada ISO25010 [32].

Se evidencia en los estudios, que los conocedores del dominio por lo general no son del área de la informática o afines, si la organización requiere una LPS implementada a través de SOA, no se describe o enuncia como se realiza la transformación de las decisiones del negocio al BPM (Modelo de Procesos de Negocios), FM (Modelo de Características) en el caso de la LPS y los servicios que ha de soportar la arquitectura, particularmente para SOA; existe mucha ambigüedad y poca claridad en los estudios analizados.

Solo los stakeholders conocedores del dominio del problema con experiencia o estudios en el campo de la informática tendrían las habilidades o capacidades necesarias para realizar la tarea anterior con éxito o por lo menos con más precisión desde el punto de vista del desarrollo de LPSOS que se pretende realizar en la organización.

Tampoco se definen vistas de calidad y/o estándares de calidad para validar de manera formal el proceso de adquisición, análisis y diseño del dominio para la LPSOS [20][42].

Los 10 estudios abordan parcialmente la documentación total del dominio, según lo planteado por Bjorner [25], referente a las facetas del dominio.

Ninguno de los 10 estudios analizados aborda las 6 etapas (identificación de los stakeholders, adquisición del dominio, análisis del dominio, diseño del dominio, validación/ verificación del dominio y desarrollo de la teoría del dominio) para el desarrollo del dominio propuesta por Bjorner [25].

Se destaca que los 10 estudios abordan el proceso de desarrollo de la ID de forma diferente, es decir, aun cuando los procesos utilizan algunas actividades y artefactos comunes, cada uno de ellos realiza el proceso visto de forma global, con actividades, artefactos y técnicas diferentes.

V. Conclusiones

La principal contribución de este trabajo fue realizar una RDS para presentar una perspectiva detallada sobre las actividades, artefactos y técnicas que comúnmente se utilizan en un proceso de desarrollo para la ID de LPSOS. Existe una carencia en los estudios analizados en lo referente al uso de estándares de calidad en los procesos de desarrollo identificados y falta de trazabilidad entre los modelos utilizados, sobre todo en lo que respecta a los RNF (Requisitos No Funcionales) del modelo del negocio al modelo del sistema de software en un dominio dado. Esta revisión sistemática provee a la comunidad de desarrolladores el estado del arte en lo concerniente con el desarrollo completo de la disciplina de ID para LPSOS.

Actualmente se está definiendo un proceso de análisis de dominio para LPSOS, tomando en cuenta los análisis realizados a partir de esta RDS. Se ha seleccionado como caso de estudio para aplicar este proceso al dominio de los SIS (Sistemas Integrados de Salud). Este dominio es de particular importancia en esta investigación por tres aspectos claves: por una parte será utilizado para aplicar las principales actividades, artefactos y técnicas evidenciadas en la RDS para la ingeniería de requisitos de SIS, los cuales son sistemas de información, generalmente en 4-capas (presentación, proceso, datos y comunicación), orientados a servicios, ya que utilizan en su mayoría una arquitectura basada en SOA en su capa de comunicación. Por otra parte, en este dominio no se encuentra definida claramente una LPS y las arquitecturas tipo SOA que permiten la interoperabilidad de servicios, no abarcan funcionalidades esenciales de SIS, como por ejemplo la interoperabilidad de las historias clínicas del paciente, a menos que se tenga la adopción sistemática de estándares por parte de

la comunidad de salud; este requisito es uno de los más importantes del dominio. Por lo tanto, las tareas relacionadas con la salud del ciudadano, como son, por ejemplo, la de remitir un paciente a otros centros de salud especializados, se dificultan por la falta de interoperabilidad de las historias clínicas con otros centros de salud; se recalca que solo la adopción de estándares permitiría esta facilidad. A nivel internacional, para las historias clínicas electrónicas se encuentran ya definidos algunos estándares como el HL7, pero aún no comúnmente adoptado a nivel mundial. Finalmente, a nivel nacional tampoco hay estándares adoptados que permitan realizar la integración de estos sistemas. Es de hacer notar sin embargo, que ya fue aprobada como Ley Orgánica de la Nación la Ley de Telesalud, el 12 de Agosto del 2014[1]. El estudio de los requisitos planteados en esta Ley constituirá parte de las reglas del negocio a ser incluidas en el modelo del negocio del dominio de los SIS, del cual la LPSOS será derivada.

Con este caso de estudio se pretende demostrar las ventajas y/o fortalezas, y/o debilidades del proceso definido, el cual permitirá una primera validación del proceso.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por los proyectos grupales DISofT No. 2011001343 del PEII-Fonacit; DesCCaP No. PG-03-8014-2011 y DARGRAF No. 03-8730-2013-1 del CDCH; además, por el Postgrado en Ciencias de la Computación, Facultad de Ciencias, Universidad Central de Venezuela y el Banco Central de Venezuela.

REFERENCIAS

[1] P. Istoan, G. Nain, G. Perrouin, and J. Jezequel, *Dynamic Software Product Lines for Service-Based Systems*, in proceedings of the 2009 Ninth IEEE International Conference on Computer and Information Technology (CIT'09), Xiamen, China, October 2009.

[2] P. Istoan, *Software Product Lines for Creating Service Oriented Applications*, Masters internship at IRISA Rennes Research Institute, TRISKELL Research Team, Rennes, France, June 2009.

[3] F. M. Medeiros, E. S. de Almeida, and S. R. de Lemos Meira, *Designing a Set of Service-Oriented Systems as a Software Product Line*, in proceedings of the Fourth Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS), pp. 70-79, Salvador, Bahia, Brazil, September 2010.

[4] S. Apel, D. Batory, C. Kästner, and G. Saake, *Feature-Oriented Software Product Lines*, Springer, 2013.

[5] Q. Munir and M. Shahid, *Software Product Line: Survey of Tools*, Master's thesis, Linköping University, Department of Computer and Information Science, 2010.

[6] B. Mohabbati, M. Asadi, D. Gašević, M. Hatala, and H. Müller, *Combining Service-Orientation and Software Product Line Engineering: A Systematic Mapping Study*, Information and Software Technology, vol. 55, no. 11, pp. 1845-1859, 2013.

[7] R. Dos Santos and M. Fantinato, *The Use of Software Product Lines for Business Process Management: A Systematic Literature Review*, Information and Software Technology, vol. 55, pp. 1355-1373, 2013.

[8] K. Pohl, G. Bockle, and F. van der Linden, *Software Product Line Engineering: Foundations, Principles, and Techniques*, Springer, 2005.

[9] Wide Web Consortium, *Web Services Architecture Requirements*, W3C Working Group Note, 11 February, 2004.

[10] J. Nicolai, *SOA in Practice: The Art of Distributed System Design*, O'Reilly, Sebastopol, California, USA, 2007.

[11] M. Galster, P. Avgeriou, and D. Tofan, *Constraints for the Design of Variability-Intensive Service-Oriented Reference Architectures–An Industrial Case Study*, Information and Software Technology, vol. 55, no. 2, pp. 428-441, 2013.

[12] M. Rosen, B. Lublinsky, K. Smith, and M. Balcer, *Applied SOA: Service-Oriented Architecture and Design Strategies*,Wiley & Sons, 2008.

[13] J. Estefan, K. Laskey, F. McCabe, and P. Thornton, *Reference Architecture Foundation for Service Oriented Architecture Version 1.0*, OASIS Committee Draft, vol. 2, no. 14, 2009.

[14] M. Abu-Matar and H. Gomaa, *An Automated Framework for Variability Management of Service-Oriented Software Product Lines*, in proceedings of the IEEE 7th International Symposium on Service Oriented System Engineering (SOSE), pp. 260-267. San Francisco, California, USA, March 2013.

[15] A. Ezenwoke, S. Misra, and M. Adigun, *An Approach for e-Commerce On-Demand Service-Oriented Product Line Development*, Acta Polytechnica Hungarica, vol. 10, no. 2, 2013.

[16] P. G. G. Queiroz and R. T. V. Braga, *Uma Abordagem de Desenvolvimento de Linha de Produtos com uma Arquitetura Orientada a Serviços*, Master's thesis, ICMC-Universidade de Sao Paulo, Sao Carlos, Sao Paulo, Brazil, November 2009.

[17] S. Günther and T. Berger, *Service-Oriented Product Lines: Towrads a Development Process and Feature Management Model for Web Services*, in proceedings of the 12th International Software Product Line Conference (SPLC'08), pp. 131-136, Limerick, Ireland, September 2008.

[18] J. Lee and G. Kotonya, *Combining Service-Orientation with Product Line Engineering*, Software, IEEE, vol. 27, no. 3, pp. 35-41, 2010.

[19] C. Parra, X. Blanc, and L. Duchien, *Context Awareness for Dynamic Service-Oriented Product Lines*, in proceedings of the 13th International Software Product Line Conference (SPLC'09), pp. 131-140, San Francisco, California, USA, August 2009.

[20] H. J. González, *Integration of Quality Attributes in Software Product Line Development*, Master Tesis en Ingeniería del Software, Métodos Formales y Sistemas de Información (ISMFSI), 2012.

[21] W. Suryn, *Software Quality Engineering: A Practitioner's Approach*, IEEE, Published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2014.

[22] S. Wagner, *Software Product Quality Control*, Springer, 2013.

[23] B. Mohabbati, D. Gasevic, M. Hatala, and H. Muller, *A Quality Model and Evaluation Method for Service-Oriented Software Product Line and Configurable Business Processes*, ACM Transactions on Software Engineering Methodolgy, September 2013.

[24] L. O'Brien, L. Bass, and P. Merson, *Quality Attributes and Service-Oriented Architectures*, Software Engineering Institute, Carnegie Mellon University, 449, September 2005.

[25] D. Bjørner, *Software Engineering 3: Domains, Requirements, and Software Design*, Texts in Theoretical Computer Science, Springer-Verlag Berlin Heidelberg, 2006.

[26] B. Kitchenham, *Guidelines for Performing Systematic Literature Reviews in Software Engineering*, Version 2.3, EBSE Technical Report, Software Engineering Group, School of Computer Science and Mathematics, Keele University, UK and Department of Computer Science, University of Durham, UK, 2007.

[27] I. Reinhartz-Berger, A. Sturm, T. Clark, S. Cohen, and J. Bettin, *Domain Engineering. Product Lines, Languages and Conceptual Models*. Springer, 2013.

[28] G. Kotonya, J. Lee, and D. Robinson, *A Consumer-Centred Approach for Service-Oriented Product Line Development*, in proceedings of Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture (WICSA/ECSA 2009), pp. 211-220, Cambridge, UK, September 2009.

[29] Q. Gu and P. Lago, *On Service-Oriented Architectural Concerns and Viewpoints*, in proceedings of Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture (WICSA/ECSA 2009), pp. 289-292, Cambridge, UK, September 2009.

[1] http://conocimientolibre.cenditel.gob.ve/files/2014/05/LEY-TELESALUD.pdf

[30] M. Asadi, B. Mohabbati, N. Kaviani, D. Gaševic, M. Boškovic, and M. Hatala, *Model-Driven Development of Families of Service-Oriented Architectures*, in proceeding of the First International Workshop on Feature-Oriented Software Development (FOSD'09), pp. 95-102, Denver, Colorado, USA, October 2009.

[31] M. Allauddin, F. Azam, and M.J. Zia, *A Survey of Quality Assurance Frameworks for Service Oriented Systems*, International Journal of Advancements in Technology, vol. 2, no. 2, pp. 188-198, 2011.

[32] ISO/IEC 25010. Software Engineering - *Software Product Quality Requirements and Evaluation (SQuaRE) - Quality Model*, International Organization for Standarization (ISO), 2010.

[33] J. Herrera, F. Losavio, and A. Matteo, *Revisión Documental Sistemática de Enfoques y Técnicas para la Construcción de Arquitecturas en un Contexto de Líneas de Productos de Software*, Primera Conferencia Nacional de Computación, Informática y Sistemas (CoNCISa 2013). Naiguatá, Venezuela, Octubre 2013.

[34] P. Dolog, *Engineering Adaptive Web Applications*, Doctoral Dissertation, University of Hanover, Germany, 2006.

[35] A. Lapouchnian, *Exploiting Requirements Variability for Software Customization and Adaptation*, Doctoral Dissertation, University of Toronto, Canada, 2011.

[36] C. Parra, *Towards Dynamic Software Product Lines: Unifying Design and Runtime Adaptations*, Doctoral Dissertation, Université des Sciences et Technologie de Lille, Lille I, France, 2011.

[37] D. Smith and G. Lewis, *Service-Oriented Architecture and Software Product Lines: Pre-Implementation Decisions*, in proceedings of the 3rd International Workshop on Service Oriented Architectures and Product Lines (SOAPL'09), pp. 166-171, San Francisco, California, USA, September 2009.

[38] M. Bošković, D. Gašević, B. Mohabbati, M. Asadi, M. Hatala, N. Kaviani, and E. Bagheri, *Developing Families of Software Services: A Semantic Web Approach*, Journal of Research & Practice in Information Technology, vol. 43, no. 3, 2011.

[39] J. Lee, D. Muthig, and M. Naab, *A Feature-Oriented Approach for Developing Reusable Product Line Assets of Service-Based Systems*, Journal of Systems and Software, vol. 83, no. 7, pp. 1123-1136, 2010.

[40] A. Trendowicz and T. Punter, *Quality Modeling for Software Product Lines*, in proceedings of the 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'03), 2003.

[41] A. Goeb and K. Lochmann, *A Software Quality Model for SOA*, in proceedings of the 8th International Workshop on Software Quality (WoSQ'11), pp. 18-25, Szeged, Hungary, September 2011.

[42] S. Montagud, *Un Método para la Evaluación de la Calidad de Líneas de Productos Software basado en SQuaRE*, Master's Thesis, (In Spanish), Master en Ingenieria del Software Metodos Formales y Sistemas de Información. Universidad Politécnica de Valencia, Spain, 2009.

[43] S. Segura, D. Benavides, A. Ruiz-Cortés, and P. Trinidad, *Open Source Tools for Software Product Line Development*, Open Source and Product Lines, 2007.

[44] M. Acher, P. Collet, P. Lahire, and R. B. France, *Familiar: A Domain-Specific Language for Large Scale Management of Feature Models*, Science of Computer Programming, vol. 78, no. 6, pp. 657-681, 2013.

[45] L. Baresi, S. Guinea, and L. Pasquale, *Service-Oriented Dynamic Software Product Lines*, Computer, vol. 45, no. 10, pp. 42-48, 2012.

[46] J. Lee, D. Muthig, and M. Naab, *An Approach for Developing Service Oriented Product Lines*, in proceedings of the 12th International Software Product Line Conference (SPLC'08), Limerick, Ireland, September 2008.

[47] M. Njima, M. ter Beek, and S. Gnesi, *Product Line Architectures for SOA*, in proceedings of the 2011 International Conference on Software Engineering Research and Practice (SERP'11), Las Vegas, Nevada, USA, July 2011.

[48] F. Losavio, and A. Matteo, *Reference Architecture Design Using Domain Quality View*, Journal of Software Engineering & Methodology, vol. 3, no. 1, 2013.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 34-43, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Interfaz Móvil de una Aplicación Sensible al Contexto Utilizando Base de Datos Difusas

José Cadenas, Soraya Carrasquel, Rosseline Rodríguez, Daniela Ruiz, Ruben Serradas
jtcadenas@usb.ve, scarrasquel@usb.ve, crodrig@usb.ve, daru015@gmail.com, rubenserradas@gmail.com

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

Resumen: La forma como los seres humanos se comunican depende del contexto, determinado por el medio que lo rodea, la percepción de cada individuo y a factores personales intrínsecos. Este tipo de comunicación no se puede trasladar directamente a los sistemas tradicionales ya que se debe proporcionar explícitamente, es decir, el contexto es preestablecido por el diseñador del sistema. Esta es la razón por la que la mayor parte del contenido que éstos retornan no está personalizado. En la actualidad, se requiere desarrollar sistemas inteligentes sensibles al contexto añadiendo nuevas funcionalidades: la posibilidad de representar información imperfecta (imprecisa, incierta o vaga), así como consultar bases de datos utilizando etiquetas lingüísticas permitiendo aproximarse al lenguaje natural humano. Estas funcionalidades se pueden añadir utilizando bases de datos difusas. El desarrollo de sistemas personalizados permitirá filtrar la gran cantidad de información proporcionada debido a los avances en las Tecnologías de Información y Comunicación. Por otra parte, el uso de dispositivos móviles está aumentando considerablemente. Por estas razones, en este trabajo se presenta el desarrollo de un prototipo de interfaz móvil para una aplicación web sensible al contexto utilizando bases de datos difusas, donde cada usuario obtiene respuestas con valores personalizados de acuerdo a sus definiciones de etiquetas lingüísticas.

Palabras Clave: Bases de Datos Difusas; Aplicaciones Sensibles al Contexto; Interfaz Móvil.

Abstract: The way humans communicate is depending of the context, it is determined by the surrounding environment, the perception of each individual, and intrinsic personal factors. This type of communication cannot be transferred directly to computer systems, because we must provide the information to traditional systems in an explicitly way, that is the context is pre-established by the system designer. This is the reason because the most part of content that they deploy is not personalized. Nowadays, it is desirable to develop context-aware intelligent systems with new functionalities: the possibility to represent imperfect information (imprecise, uncertain or vague), and query databases using a language closer to human natural language. These aims can be achieved using fuzzy databases. The development of more personalized systems allows filter the large amount of information provided due to the advances of Information and Communication Technologies. Moreover, in actual world the use of mobile devices is increasing. In this proposal, we developed a mobile interface for a context-aware web application that uses a fuzzy database. In this interface, each user obtains personalized answers according to his/her definitions of linguistic labels.

Keywords: Fuzzy Database; Context-Sensitive Applications; Mobile Interface.

I. Introducción

Es muy frecuente que las respuestas, en el proceso de comunicación entre seres humanos, cambien dependiendo de diversos factores. Esto es especialmente cierto en la forma en que se comparten ideas: en distintos momentos pueden cambiar el significado de una misma palabra, entre otras cosas en función del entorno cambiante, la percepción de las personas y a factores intrínsecos tales como las emociones. Una de las causas es que los seres humanos comparten una información situacional implícita: el contexto, en el cual se basan para dar significado a las palabras. Lamentablemente, esta habilidad en el proceso de comunicación entre personas no se transfiere de una forma adecuada en la interacción entre las personas y los sistemas informáticos. Este proceso además es dinámico.

En la computación tradicional el usuario tiene mecanismos fijos, en su mayoría restrictivos, para proporcionar datos a la máquina y recibir información de ésta [1]. Dichos mecanismos no se adaptan a la actual sociedad de la información y el conocimiento, donde los dispositivos móviles toman cada vez más protagonismo. El usuario resulta sobrecargado de información, generalmente redundante o de poco interés para él y con pocas capacidades para su manipulación; además, está sujeto a consideraciones primarias de diseño, impuestas por el diseñador de las aplicaciones que usa habitualmente.

Muchas aplicaciones permiten crear perfiles de usuarios que pueden basarse tanto en la información proporcionada por el mismo como de la extraída por la interacción con otras aplicaciones, por ejemplo los sitios web por los que navega. Así, las personas se sentirán más identificadas con el sistema, el cual tendrá más datos de insumo para lograr personalizar las respuestas. A la teoría que sustenta este modelo de trabajo se le conoce como Sensibilización al Contexto [2], el cual es un tema de interés actual en el desarrollo de aplicaciones. Mediante este paradigma se puede lograr la personalización de las aplicaciones de manera que el comportamiento del sistema varíe según el contexto del usuario: ubicación geográfica, dispositivo que utiliza, el clima, percepciones de las personas, entre otros.

En el caso particular de aplicaciones cuyos usuarios trabajan en centros de salud, la atención que se le debe prestar a estos factores es crítica pues el ambiente obliga a tomar decisiones con rapidez. Además se agregan otros factores como los distintos roles producto de las diversas actividades que se realizan, así como variaciones bruscas en cuanto a circunstancias físicas, emocionales y sociales en que se encuentren [3]. Por último, las apreciaciones de un médico en este ambiente están muy ligadas a su experiencia personal por lo que debe la información que puede proveer a un sistema informático es imperfecta (imprecisa, incierta o vaga) [4].

Por ello, en el área de computación flexible (*Soft Computing*), se propone la Teoría Computacional de Percepciones. Esta teoría se basa en la capacidad de los seres humanos de calcular y analizar información por medio de percepciones que producen normalmente información imperfecta [5]. En la vida real se puede notar que las personas se comunican por medio de etiquetas en lugar de datos precisos pues es más natural hablar de una persona *alta* que indicar su estatura exacta: "mide 186 cm.". Sin embargo, estas etiquetas también dependen de la percepción de la persona que la esté utilizando. Es diferente la percepción de estatura *alta* que tiene un jugador de baloncesto a la que tiene un jinete de carreras de caballos.

En las bases de datos tradicionales se modela la información de un universo específico usando datos precisos. Sin embargo, en muchos casos representar la realidad donde hay muchos datos imperfectos se complica pues la gestión de los datos en estos sistemas tradicionales sufren el problema denominado "rigidez", debido a que utilizan el paradigma de consulta de coincidencia exacta [6]. Esto acarrea dos inconvenientes principales: la respuesta vacía (condiciones muy restrictivas) o demasiadas respuestas (condiciones poco estrictas) [7].

La Teoría de los Conjuntos Difusos [8] ha sido utilizada para modelar los datos imperfectos, incorporándolos, entre otras ramas, a las bases de datos tradicionales. Asimismo, la lógica difusa basada en esta teoría, se ha usado para expresar consultas flexibles (difusas) sobre datos que pueden ser precisos o imperfectos [9]. A este tipo de sistemas se les conocen como bases de datos difusas [10]. Utilizando este enfoque se pueden almacenar etiquetas con una semántica que exprese las preferencias de los usuarios. Por ejemplo, para describir el peso de un paciente, puede usarse la etiqueta "obeso" con su correspondiente función de pertenencia, en lugar de indicar el valor preciso de éste. Además, esto permite introducir la noción de gradualidad en los términos lingüísticos utilizados [11]. Adicionalmente, cada usuario puede definir etiquetas con valores personalizados para cada variable lingüística, permitiendo que los significados cambien al considerar el usuario como contexto.

El contexto del usuario ha sido ampliamente estudiado en áreas como Computación Ubicua, Interacción Persona Ordenador, Inteligencia Artificial, Sistemas de Recuperación y Recomendación. Sin embargo, muy poco se ha investigado en el área de Sistemas de Información utilizando Bases de Datos Difusas. En esta propuesta se desarrolla una interfaz móvil de una aplicación web sensible al contexto utilizando bases de datos difusas, tomando como trabajo previo el uso de dominios difusos con semántica adaptable propuesta en [12][13].

El presente trabajo está organizado en cuatro secciones después de la introducción. En la Sección II, se muestra el marco teórico que sustenta el desarrollo de la interfaz móvil de la aplicación sensible al contexto, en éste se explicarán diversos conceptos fundamentales para la investigación. En la Sección III se describen las tecnologías utilizadas en el desarrollo de la interfaz móvil y la motivación para su uso. En la Sección IV, se detallan los aspectos relevantes en el desarrollo de la propuesta, la metodología usada y las funcionalidades que brinda al usuario. Finalmente, en la Sección V, se exponen las conclusiones y se plantean los trabajos futuros.

II. Marco Teórico

La fundamentación teórica está constituida por dos áreas que tienen bases comunes unidas a la sensibilización al contexto para producir la interfaz móvil propuesta, es por ello que se presenta a continuación un breve marco teórico donde se describen las Bases de Datos Difusas, la Teoría Computacional de Percepciones y la arquitectura de una Aplicación Web sensible al contexto que utiliza Bases de Datos Difusas.

A. Bases de Datos Difusas

En el modelado de problemas del mundo real es frecuente encontrarse con categorías de objetos que no tienen un criterio bien definido de pertenencia tal como ocurre en un conjunto tradicional. Asimismo, estos objetos pueden tener información imperfecta difícil de definir en bases de datos tradicionales o que presentan imprecisión en el valor obtenido por determinada fuente. Una herramienta comúnmente utilizada para representar este tipo de información es la Teoría de Conjuntos Difusos [8].

En un conjunto difuso, sus elementos tienen asociado un grado de pertenencia al conjunto en el intervalo [0,1]. La función de pertenencia al conjunto difuso A de los elementos de un universo X se describe como $\mu_A: X \rightarrow [0,1]$, de forma que para cada x, $\mu_A(x)$ representa el grado de pertenencia del elemento x sobre el conjunto difuso A. Mientras más se acerque $\mu_A(x)$ a la unidad, mayor será la pertenencia al conjunto difuso. El 1 representa la pertenencia total al conjunto y el 0 la no pertenencia. Estos dos valores corresponden a la definición de pertenencia en conjuntos tradicionales.

Hay diversas formas de representar funciones de pertenencia, una de las más usuales es la que se simboliza gráficamente en forma de trapecio, el cual matemáticamente lo definen cuatro valores que constituyen los puntos de inflexión del trapecio (v_1,v_2,v_3,v_4), con $v_i \in X$, $i \in \{1,2,3,4\}$. Si los dos primeros valores son 1 representan un trapecio incompleto "abierto por la izquierda", mientras que si los dos últimos valores son

iguales al valor máximo del dominio entonces representan un trapecio incompleto "abierto por la derecha".

También se puede especificar la función de pertenencia a través de un conjunto difuso por extensión, indicando para cada elemento del dominio el grado de pertenencia de dicho elemento al conjunto difuso $\{x_1/grado_1, ... , x_n/grado_n\}$.

En las bases de datos difusas se utiliza esta teoría para representar datos imperfectos y para expresar consultas flexibles a bases de datos tradicionales, donde la condición de la consulta puede involucrar etiquetas lingüísticas que representan los valores difusos. Estas etiquetas son términos imperfectos (imprecisos, inciertos o vagos) del lenguaje natural utilizado por el ser humano.

Uno de los principales beneficios de un gestor de base de datos difuso es que proporciona una representación de los requerimientos de los usuarios más cercana a sus preferencias y al mundo real. Esto permite identificarse más fácilmente con la manera en que se muestran y capturan los datos, lo cual se utilizará para personalizar y sensibilizar las aplicaciones.

En una base de datos difusa se pueden encontrar atributos cuyos valores son afectados por algún tipo de imperfección. Existen distintos dominios difusos que pueden ser usados para representar estos valores. Estos dominios permiten modelar y representar datos imperfectos y/o flexibilizar las consultas. En la Figura 1 se muestra una jerarquía de clases de dominios difusos basados en la propuesta de [14].

En esta jerarquía, según la cardinalidad se pueden distinguir dos tipos de dominios difusos: los atómicos y los conjuntivos. En los atómicos, el atributo sólo puede tomar un valor mientras que en los conjuntivos, el atributo puede tener un valor múltiple representado por un conjunto difuso.

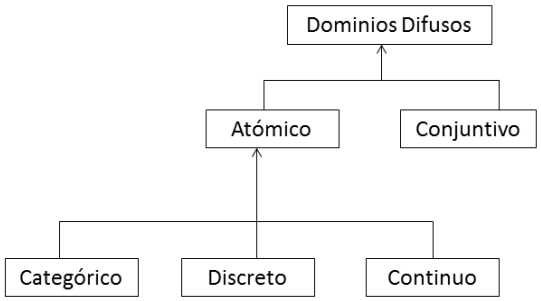


Figura 1: Jerarquía de Clases de Dominios Difusos

Los dominios atómicos se pueden clasificar en categóricos, discretos o continuos. En los dominios categóricos sus valores son etiquetas lingüísticas que no tienen una representación semántica asociada por lo que es necesario crear una matriz de semejanza para la comparación entre éstas. Como ejemplo, con este dominio se pueden representar los distintos tipos de marcha, deformada o no, que puede tener un paciente, usando las etiquetas lingüísticas: *Atáxica*, *Danzante*, *Parkinsoniana*, *Normal*. Luego, el médico especialista podría decidir, de acuerdo a su experiencia previa y opinión personal, que el grado de similitud entre una marcha *parkinsoniana* (aquella que aparece en forma típica en los síndromes parkinsonianos avanzados) y una marcha *normal* es de 0.3. Este valor se colocaría en la matriz de semejanza.

En los dominios discretos, los valores pueden ser etiquetas lingüísticas cuyos elementos están en un conjunto soporte discreto y el grado de pertenencia corresponde a la asociación de una etiqueta con un elemento del conjunto discreto. Por ejemplo, se puede expresar el tono flexor dorsal de un paciente utilizando las etiquetas lingüísticas *Atonía*, *Hipotonía*, *Hipertonía* y *Normotonía*; además existe una tabla estándar que utilizan los médicos para reflejar este tono muscular (usualmente los números naturales del 1 al 4). Para establecer correspondencia entre estas dos formas de expresar el tono flexor dorsal se usa una tabla de grados de similitud entre las etiquetas y los números.

Por último, en el dominio continuo cada etiqueta lingüística está definida por una función matemática cuyo conjunto soporte es continuo, usualmente se utiliza una función cuya representación gráfica es un trapecio (puede ser completo o abierto a uno de los lados). Como ejemplo, se pueden definir las etiquetas asociadas al atributo peso (*delgado*, *normal* y *obeso*) con funciones trapezoidales como se muestra en la Figura 2. Al atributo susceptible de ser representado por etiquetas lingüísticas se le denomina variable lingüística.

Como ejemplo de un dominio conjuntivo, se tiene que un paciente con problemas de marcha puede utilizar distintos tipos de dispositivos para mejorar su movilidad, lo cual puede representarse indicando para cada uno de tales dispositivos la frecuencia de uso por el paciente. Así se podría tener, como valor de la frecuencia para un paciente dado, un conjunto difuso de etiquetas similar a {andadera/1, bastón /0.75, muletas/0.50, silla de ruedas/0.25}.

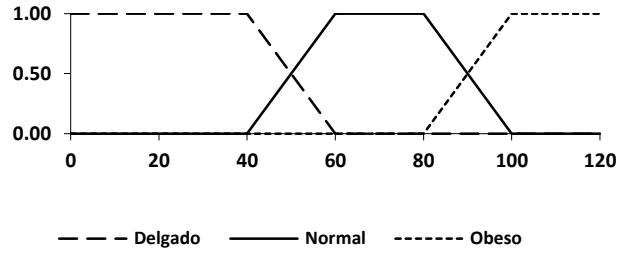


Figura 2: Etiquetas Lingüísticas Correspondientes a la Variable Lingüística "peso"

B. Teoría Computacional de Percepciones

Las decisiones que toma el ser humano basándose en percepciones son muy frecuentes, desde manejar una bicicleta hasta estacionar un carro, la información con la cual se cuenta en estas acciones es imperfecta, además se utilizan los estímulos recibidos por los sentidos (vista, tacto, oído, olfato y gusto) para realizar con éxito dichas tareas.

La Teoría Computacional de Percepciones (CTP) se inspira en esta capacidad particular de actuar del ser humano que razona y actúa basándose en información obtenida por las percepciones. En particular, cuando se habla de percepciones, la forma de medir es a través de etiquetas, produciendo una división gradual entre las distintas categorías.

Según Zadeh [5] las percepciones son *f-granular*, lo que significa que los límites de las clases percibidas son imprecisos o borrosos. De esta forma, los valores de los atributos son granulados, con un gránulo representado por un grupo de

valores (puntos, objetos), agrupados mediante etiquetas por su semejanza, proximidad o función. Por ejemplo, los gránulos de una variable lingüística como la edad de una persona, podrían ser las etiquetas: *muy joven*, *joven*, *promedio*, *viejo*, *muy viejo*. La importancia de CTP deriva del hecho que, en gran parte, la toma de decisiones y el razonamiento común está basado en percepciones humanas.

Además, las percepciones pueden ser afectadas por el contexto del usuario. Por ejemplo hablar de una persona *joven* no tiene el mismo significado cuando se requiere de un gimnasta que cuando se necesita a un investigador. En esta propuesta se permite al usuario crear en su perfil las definiciones de las etiquetas de acuerdo a sus necesidades, con la ayuda de la CTP.

C. Aplicación Web Sensible al Contexto usando Bases de Datos Difusas

Según [12], una aplicación web sensible al contexto puede gestionar datos contextuales (los almacena, modifica y utiliza), los cuales pueden ser implícitos y explícitos. Éstos están asociados a un perfil de un usuario determinado, permitiendo tomarlos como insumos para que el cambio de su valor pueda afectar la entrada de datos o la salida de una consulta realizada ya sea en su semántica, en su resultado o en su presentación.

En la Figura 3, se puede observar el esquema propuesto para una aplicación sensible al contexto utilizando una base de datos difusa basada en la propuesta de [12].

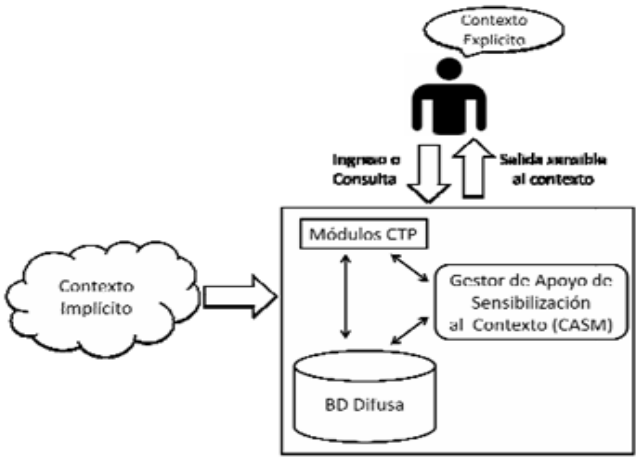


Figura 3: Esquema de una Aplicación Sensible al Contexto utilizando Bases de Datos Difusas

En este modelo los componentes son:

- Usuario, que interactúa con el sistema para poder establecer el contexto explícito: semántica, perfil, rol, estado (físico, emocional, social), actividad, situación y/o movilidad.
- Módulos CTP, corresponden a los módulos de almacenamiento y recuperación sensibles al contexto. Son traductores que se encargan de comunicarse con el gestor de apoyo de sensibilización al contexto (CASM) quien determina cómo éste afecta las entradas o salidas, antes de ser almacenadas o recuperadas de la base de datos. Su objetivo es hacer la transformación lingüística del lenguaje del usuario y los datos en bruto almacenados en la Base de Datos Difusa.
- Gestor de apoyo de sensibilización al contexto (CASM) es el encargado de gestionar el Catálogo Contextual, el histórico de las acciones del usuario y las reglas que permitan inferir el comportamiento de acuerdo al contexto. Este es el sistema experto que funciona como base de conocimiento para que los módulos CTP puedan hacer las transformaciones correspondientes.
- Base de Datos Difusa, que gestiona datos imperfectos siguiendo los postulados de la Teoría Computacional de Percepciones (CTP). Aquí se almacenan datos en bruto que luego serán procesados utilizando las reglas proporcionadas por el CASM para que los módulos CTP transformen el lenguaje con que interactúan los seres humanos.
- Contexto Explícito, es la información contextual proporcionada por el usuario o por alguna aplicación que se conecte al sistema. El usuario debe estar siempre identificado según los niveles de seguridad y privilegios. Es así que la primera información contextual explícita viene dada por la identificación del usuario. Otros aspectos considerados dentro del contexto explícito son: la semántica del usuario, su perfil y el rol que ejecuta el usuario.
- Contexto Implícito es la información contextual suministrada por sensores ambientales (como ubicación, tiempo, clima, aplicación, dispositivo, conexión) o agentes de software (servicios web) que infieren el contexto de acuerdo a la actividad o situación del usuario.

Este modelo fue utilizado como guía para el desarrollo de una aplicación web de escritorio y luego su interfaz móvil para el Examen Físico Articular de los pacientes del Laboratorio de Marcha del Hospital Ortopédico Infantil de Venezuela, lo cual se explica en más detalle en la Sección IV.

Debido a que este trabajo es una continuación de otras aportaciones desarrolladas, el diseño de la base de datos difusa y su implementación, así como el funcionamiento de los módulos CTP y CASM, son descritos con más detalle en [12][13][15].

III. Tecnologías Utilizadas

El diagrama de componentes de la aplicación puede observarse en la Figura 4. Las herramientas tecnológicas que se utilizaron para el desarrollo de la aplicación web sensible al contexto usando bases de datos difusas y su interfaz móvil se describen en esta sección.

A. Aplicación Web

La aplicación Web fue desarrollada en el ambiente XAMPP [16] de fácil instalación y configuración, el cual utiliza el servidor Apache 2.2, donde funciona el sistema implementado en PHP. Para la elaboración de las páginas Web se usó el lenguaje marcado HTML5 [17].

En cuanto al diseño de la interfaz móvil se aprovecharon distintos componentes de *Bootstrap* [18], un *framework* de diseño de aplicaciones web para dispositivos móviles, el cual contiene componentes prediseñados que se adaptan al tamaño del dispositivo. En particular se usaron botones sensibles al tamaño de la pantalla, alertas predeterminadas para mantener la

coherencia, iconos para presentar las opciones disponibles al usuario, conjuntos de formularios para simplificar la obtención de información, entre otros. *Bootstrap* está especificado en HTML, CSS, LESS y Javascript.

B. Funcionalidades

Las funcionalidades fueron desarrolladas usando dos lenguajes: PHP, antes mencionado, y JavaScript. PHP [16] fue utilizado para las conexiones con la base de datos desde el sistema móvil y para el manejo de datos y consultas. JavaScript [19] se utilizó para la realización de calendarios, efectos visuales sencillos y para la elaboración de gráficas de los datos.

C. Base de Datos

El gestor de bases de datos utilizado fue Oracle, por su capacidad objeto relacional, lo que permite la definición de dominios difusos mediante un enfoque orientado a objetos, tal como se propone en [12]. Esta capacidad no es provista en su totalidad por el resto de los manejadores disponibles, en particular aquellos de uso libre. Oracle permite representar objetos complejos, extender tipos de datos nativos, variedad de métodos y es idónea para operar con otros lenguajes de programación además de SQL.

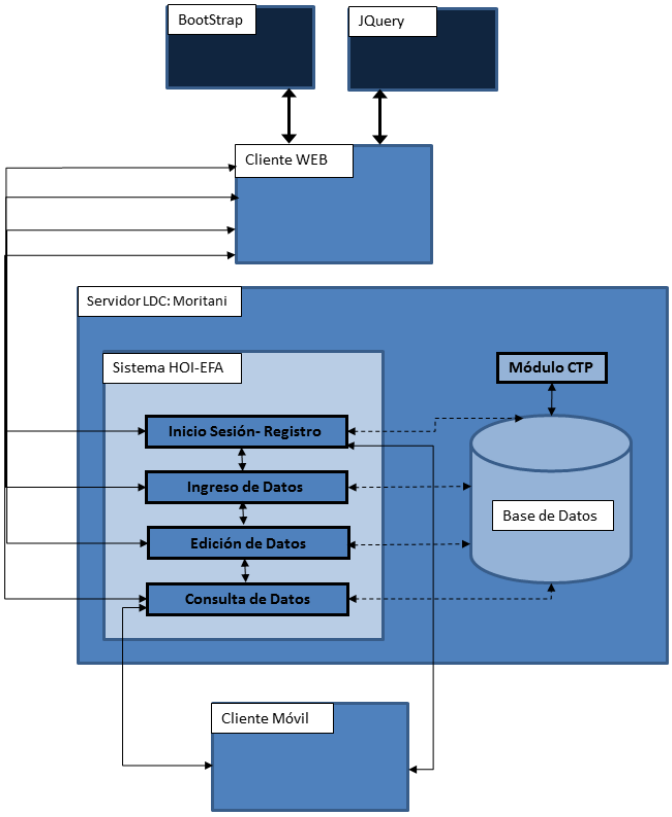


Figura 4: Diagrama de Componentes de la Aplicación

D. Documentación

El generador de documentación de PHP, PHPDoc [20], se utilizó para analizar el código fuente del sistema y producir una documentación navegable legible estilo un API (*Application Programming Interface*), útil para programadores que requieran continuar o mantener la aplicación.

IV. Aplicación Sensible al Contexto utilizando Bases de Datos Difusas

Para el desarrollo de la aplicación sensible al contexto según el modelo propuesto en [13], se tomó como caso de estudio el examen físico articular [15] que se efectúa en el Laboratorio de Marcha del Hospital Ortopédico Infantil [21]. En una primera fase se construyó una aplicación web estándar para computadores de escritorio. En la segunda fase, se desarrolló la interfaz móvil para dicha aplicación.

A. Antecedentes

El Hospital Ortopédico Infantil (HOI) está ubicado en la Av. Andrés Bello de Caracas, Venezuela. Fue fundado el 20 de abril de 1945 con el objetivo de combatir las secuelas de la Poliomielitis y la invalidez. Es el principal referente de centros de ortopedias en el país y actualmente también presta atención de adultos. El HOI posee un Laboratorio en Marcha, el cual es un sistema de medición de tecnología avanzada que se utiliza para el diagnóstico y tratamiento de enfermedades del sistema locomotor y neuromuscular, como la parálisis cerebral [21].

Dado que el área médica es un ejemplo donde se utilizan términos o etiquetas que dependen de las percepciones del usuario, se propone desarrollar una aplicación web sensible al contexto utilizando bases de datos difusas que facilite el registro y actualización del Examen Físico Articular (EFA), el cual es realizado por el Laboratorio de Marcha del HOI. Para ello se analizaron los diferentes atributos que conforman el EFA [15].

En el EFA se utilizan etiquetas para los rangos articulares de los miembros inferiores (cadera, pie, rodilla, tobillo) tales como “dentro de los límites normales (DLN)” de acuerdo a la observación de un evaluador médico. Así mismo, el especialista utiliza etiquetas para el peso de una persona (*delgado*, *normal* u *obeso*) y la talla (*bajo*, *normal*, *alto*). Como resultado de este análisis inicial se obtuvo el diseño de una base de datos difusa, la cual incluía atributos que permitiría valores de datos imperfectos. Luego, se definieron etiquetas lingüísticas para atributos como el peso, los tonos musculares relacionados a las articulaciones de los miembros inferiores, el tipo de marcha, los dispositivos utilizados por un paciente (bastón, silla de ruedas, andaderas, muletas), entre otros.

Algunos atributos, como el peso, admiten tanto valores precisos (la medida exacta del mismo en gramos) o valores difusos que corresponden a las etiquetas definidas por el usuario (*delgado*, *normal*, *obeso*). Así, al momento que el usuario realice una consulta que involucre tales atributos, el valor en el resultado se obtiene por la asociación de las etiquetas y la semántica que da el usuario autenticado que las definió; o en su defecto, por etiquetas definidas por un usuario experto con sus valores por defecto.

Como objetivo de la aplicación web se propuso incorporar la sensibilización al contexto utilizando las facilidades proporcionadas por las Bases de Datos Difusas para etiquetas lingüísticas de acuerdo a la percepción de cada usuario. De esta forma se garantiza que al ingresar un nuevo usuario el sistema adapta sus funcionalidades según el contexto en que se encuentre.

En el sistema cada usuario podrá definir sus etiquetas personalizadas. Al cambiar de usuario, lo cual es considerado

por el sistema como un contexto diferente, se podrán manipular los datos de acuerdo a su propia definición. Así, se obtendrán respuestas diferentes a las consultas según la semántica definida por el usuario, aunque la base de datos mantenga los datos "brutos" almacenados. Esto podrá visualizarse en las próximas secciones mediante ejemplos.

Es de hacer notar que aunque el contexto para esta aplicación esté representado sólo por el cambio de usuario, mediante la utilización del modelo propuesto en [12] se pueden incluir otras variables que representen el contexto tanto explícito (rol de la persona u objetivos) como implícito (localización, clima, fecha y hora). Estos pueden ser considerados en otros dominios de aplicación.

ellos. La clase *DT_objtyp* corresponde al dominio difuso continuo. Las clases que se heredan de ésta son todos los posibles atributos con valores imperfectos que pueden ser representados por trapezoides (de la clase *Trapezoid_objtyp*) que tiene el examen físico articular, como por ejemplo, el peso y la talla.

Además se tiene la clase *dominio_t* la cual se utiliza para generalizar el dominio difuso categórico (*etiqueta_t*) y el discreto (*dominio_fijo_t*). Por último se define la clase *UserDef_Objtyp* que representa los usuarios que el sistema toma como contexto.

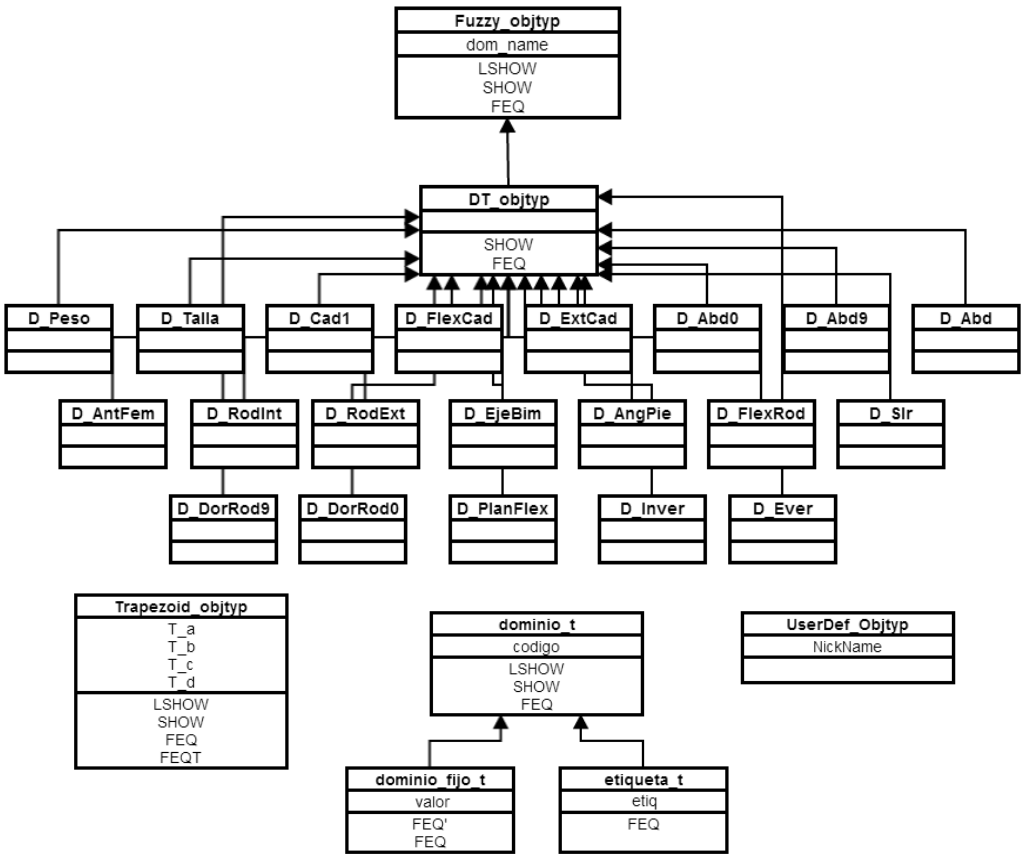


Figura 5: Diagrama de Clases de Dominios Difusos del EFA

B. Base de Datos Difusa

La base de datos del Examen Físico Articular (EFA) fue diseñada en un trabajo previo [15]. En este trabajo se extendieron los dominios difusos, luego de analizar los diferentes atributos del EFA que podrían tener valores imperfectos. En la Figura 5 se muestra el diagrama de clases que presenta los dominios difusos extendidos para incorporarlos como nuevos tipos de datos en la base de datos del EFA.

En un primer término aparece el tipo objeto difuso (*Fuzzy_objtyp*) que representa los dominios difusos. Esta clase posee los métodos LSHOW, SHOW y FEQ que permiten visualizar los valores difusos y realizar comparaciones entre

C. Interfaz Móvil

Para la adaptación del sistema web al ambiente móvil se tomaron en consideración los diversos obstáculos que presentan los dispositivos móviles, los cuales son mencionados en [22]:

- Tamaño de la pantalla: Estos dispositivos por lo general no ofrecen demasiado espacio para presentar información y opciones provistas por el sistema. Por ello, se seleccionó lo esencial de la aplicación web original, que debía presentarse al usuario en este entorno.
- Anchura y altura variables: Debido a la gran diversidad de dispositivos móviles con distintas dimensiones adaptados a todas las necesidades y gustos, fue necesario utilizar

herramientas que permitieran la creación de sitios webs y aplicaciones móviles adaptables al tamaño del dispositivo. Esta funcionalidad es provista por *Bootstrap*.

- Pantalla táctil: En un entorno móvil no se puede esperar la misma precisión que en un computador de escritorio donde se cuenta con un ratón o *touchpad*. Por ello, fue necesario diseñar botones que permitieran al usuario de un dispositivo pequeño poder observarlos y usarlos con facilidad.
- Dificultad de escritura: Por las razones mencionadas es deseable el usuario escriba lo menos posible. Para esto, se consideró en el diseño diversas facilidades que simplifican la realización de consultas e inserción de datos.
- Ambiente físico desafiante: La movilidad produce que el usuario pueda encontrarse en cualquier lugar, de manera que es necesario considerar aspectos como la iluminación, el ruido o el movimiento. Se eligieron colores que contrastan para facilitar la visibilidad en la mayor parte de las situaciones en la que se localice el usuario.

En la Figura 6 se pueden observar algunos elementos de la interfaz móvil. Los íconos en la parte superior corresponden al home, agregar paciente, agregar examen físico articular, consultar pacientes, eliminar registro de paciente y salir.

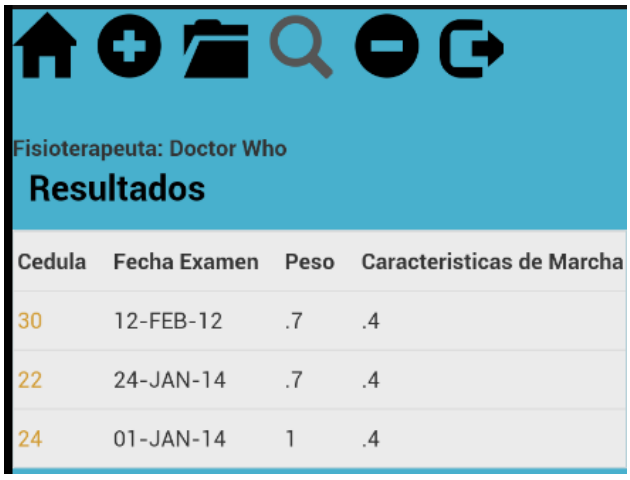


Figura 6: Resultado de la Consulta de Pacientes *Delgados* con Marcha *Normal*

D. Metodología

La metodología usada en el desarrollo fue la de Programación Extrema la cual es una disciplina del desarrollo de software basada en los valores de la comunicación, simplicidad y retroalimentación. Se fundamenta en una serie de reglas sencillas, con retroalimentación permanente, que permiten al equipo de trabajo medir los avances realizados y refinar sus prácticas para situaciones determinadas [23].

E. Funcionalidades Provistas

El diseño original contempló la posibilidad que el usuario pudiera interactuar de manera similar a la aplicación en ambiente de escritorio. El sistema en ambiente móvil incluye las funcionalidades que incorporan el contexto del usuario. Éstas se describen a continuación:

- Consulta de pacientes: inicialmente, la aplicación de escritorio sólo contempló la consulta por cédula de identidad. Dada la diversidad de lugares en los que el médico o fisioterapeuta pudiera encontrarse con poca información certera del paciente, se decidió aumentar esta funcionalidad agregando consultas por nombre o apellido. En éstas se utiliza la misma interfaz. La información desplegada del paciente se organizó en varias pantallas para evitar desplazamientos innecesarios al moverse en ella y permitir la posibilidad de visualizarla en su totalidad.
- Consulta general de pacientes: esta funcionalidad le permite al usuario realizar consultas difusas usando sus etiquetas lingüísticas, que han sido especificadas según sus preferencias. Estas preferencias fueron definidas previamente en la aplicación web. Por ejemplo, el médico puede solicitar los pacientes *delgados* con marcha *normal*, donde *delgado* y *normal* son etiquetas lingüísticas definidas por éste. Si el usuario no ha colocado sus preferencias se toman los valores por defecto, los cuales pueden ser modificados cuando el usuario desee. El resultado obtenido depende de las definiciones (ver Figura 6).
- Comparación entre pacientes: se da la posibilidad de consultas que retornen pacientes similares. Por ejemplo, obtener los pacientes que utilizan el mismo dispositivo de apoyo a su movilidad, que otro paciente dado. Debido a las limitaciones que presenta la pantalla de un dispositivo móvil se provee una respuesta con un máximo de ocho pacientes los cuales son ordenados por el grado de semejanza, de manera decreciente. Para esto se utiliza un "umbral" o valor en el intervalo [0,1), definido por el usuario, que filtra los resultados con un grado de semejanza menor al deseado. De esta manera, se evita la sobrecarga de información poco relevante. En la Figura 7 se puede ver el resultado de una consulta que compara a María Pérez con los demás pacientes cuyo grado de semejanza es mayor a 0.3.

F. Consultas Sensibles al Contexto

Cada usuario puede definir una etiqueta normal para el peso de una persona. Así, una vez efectuado el examen físico articular, a pesar de que se almacena en la Base de Datos un dato que puede ser preciso (por ejemplo 85 Kg) las consultas pueden ser difusas de acuerdo a las etiquetas definidas.

En la Figura 8 se muestra el resultado de la consulta "pacientes con peso *normal*" efectuada por el "usuario 1", mientras que en la Figura 9 se muestra el resultado de la misma consulta efectuada por el "usuario 2". En el primer caso la paciente "Andreina Loriente" aparece en el resultado de la consulta con un valor de pertenencia de 0.5 y el paciente "Andrés Loriente" con un valor de pertenencia de 0.33. Son los pacientes cuyo peso igual a *normal* tienen un valor de pertenencia distinto de cero. En el segundo caso, también aparece "Andreina Loriente", pero con un valor de pertenencia igual a 1, mientras que "Andrés Loriente" no aparece en el resultado de la consulta, esto indica que su valor de pertenencia es cero. En su lugar, aparece otro paciente "Verónica Hernández" con valor de pertenencia 0.33.

Figura 7: Resultado de una Comparación entre Pacientes

Nombres	Apellidos	Cedula	Id Historial	Fecha Examen	Peso
andreina	loriente	19380899	1	11-20-2012	.5
andres	loriente	19380900	3	11-20-2012	.33

Figura 8: Resultado de la Consulta "Pacientes con Peso *Normal*" para el "usuario1"

Nombres	Apellidos	Cedula	Id Historial	Fecha Examen	Peso
andreina	loriente	19380899	1	11-20-2012	1
veronica	hernandez	19380898	2	11-30-2012	.33

Figura 9: Resultado de la Consulta "Pacientes con Peso *Normal*" para el "usuario2"

En estas figuras, se puede notar claramente que los grados de pertenencia son distintos aunque se están accediendo a los mismos datos. Debido al hecho que cada doctor puede tener una interpretación diferente del peso normal de una persona, los resultados son distintos.

Otro ejemplo de la sensibilización al contexto es cuando el usuario puede realizar consultas por diferentes atributos difusos utilizando su propia semántica. En el resultado aparecerán las etiquetas definidas por los usuarios acompañados del grado de pertenencia del valor del atributo a dicha etiqueta (conjunto difuso). Esto puede observarse en las figuras 10 y 11.

Allí se muestran todos los EFA realizados a los pacientes en diferentes fechas, y los valores del peso y de la talla representados como conjuntos difusos. Cada etiqueta definida por el usuario viene acompañada de su grado de pertenencia.

Así por ejemplo, en la Figura 10, para el paciente "Jhosbert Abraham", según la definición del "usuario 1" tiene un peso *Sano* con grado 1 y talla *Normal* con grado 0.33 en el EFA realizado el "10-jun-14", mientras que el paciente "Andel" tiene peso *Flaco* con grado 0.44 y talla *Bajo* con grado 0.6. Si observamos la Figura 11, para el mismo paciente "Jhosbert

CI	Nombre	Fecha	Peso	Talla
19254844	Jhosbert Abraham	10-jun-14	0/Flaco; 0/Gordo 1/Sano;	0/Bajo; 0/Alto ,33/Normal;
19254844	Jhosbert Abraham	08-jun-14	0/Flaco; 0/Gordo 1/Sano;	0/Bajo; 0/Alto ,33/Normal;
20652525	Andel	08-jun-14	,44/Flaco; 0/Gordo 0/Sano;	,6/Bajo; 0/Alto ,33/Normal;
6368339	Jhon W	08-jun-14	0/Flaco; 0/Gordo ,53/Sano;	0/Bajo; 1/Alto 0/Normal;
81291961	Carlos	18-jun-14	0/Flaco; 0/Gordo 1/Sano;	0/Bajo; ,67/Alto 0/Normal;
3666968	Nancy	18-jun-14	0/Flaco; 0/Gordo 1/Sano;	,6/Bajo; 0/Alto 0/Normal;
9517539	Alejandro	20-jun-14	0/Flaco; ,2/Gordo 0/Sano;	0/Bajo; ,67/Alto 0/Normal;
19509671	Maria Isabel	12-jun-14	,2/Flaco; 0/Gordo 0/Sano;	0/Bajo; ,67/Alto 0/Normal;
5528577	Aleida	18-jun-14	0/Flaco; 0/Gordo ,17/Sano;	0/Bajo; 0/Alto 1/Normal;
6271739	Emiro	18-jun-14	0/Flaco; ,5/Gordo 0/Sano;	0/Bajo; ,67/Alto 1/Normal;
6271739	Emiro	08-feb-14	0/Flaco; 1/Gordo 0/Sano;	0/Bajo; ,67/Alto 1/Normal;
20652525	Andel	05-DEC-12	1/Flaco; 0/Gordo 0/Sano;	,6/Bajo; 0/Alto ,33/Normal;
3666968	Nancy	11-DEC-99	1/Flaco; 0/Gordo 0/Sano;	0/Bajo; 0/Alto ,8/Normal;
19509671	Maria Isabel	01-feb-00	1/Flaco; 0/Gordo 0/Sano;	0/Bajo; 0/Alto ,8/Normal;
81291961	Carlos	25-APR-13	0/Flaco; 0/Gordo 1/Sano;	0/Bajo; ,5/Alto 1/Normal;
9517539	Alejandro	16-nov-12	0/Flaco; 0/Gordo 1/Sano;	0/Bajo; ,4/Alto 1/Normal;

Figura 10: Consulta Difusa por Talla y Peso del Examen Físico Articular (EFA) de Pacientes del "usuario 1"

Abraham", en la misma fecha "10-jun-14", el resultado obtenido es diferente pues la definición del "usuario 2" considera otros valores y etiquetas. Este usuario define para el peso las etiquetas *Delgado*, *Normal* y *Obeso*, en lugar de *Flaco*, *Gordo* y *Sano*. Además, sus definiciones de los trapecios para estas etiquetas son diferentes pues los resultados no son los mismos. Para el "usuario 2", tiene un peso *Normal* con grado 1 y talla *Normal* con grado 0.8, mientras que el paciente "Andel" tiene peso *Delgado* con grado 0.7 y *Normal* con grado 0.8, y talla *Bajo* con grado 0.8 y *Normal* con grado 0.2. Es aquí donde se refleja que los usuarios usan lenguajes diferentes.

La sensibilización al contexto se nota en que aunque la base de datos de EFA de pacientes es la misma, los conjuntos difusos obtenidos para el peso y la talla son diferentes. Esto se debe a las diferentes definiciones de etiquetas para las mismas variables lingüísticas del "usuario 1" con respecto al "usuario 2". Por lo que al cambiar el usuario se cambia el contexto, y el sistema reconoce los diferentes lenguajes produciendo una salida personalizada de acuerdo a las definiciones de cada usuario.

V. Conclusiones y Trabajos Futuros

Como resultados principales de este trabajo se tiene la creación de la interfaz móvil para una aplicación web sensible al contexto utilizando bases de datos difusas. Su importancia radica en la posibilidad que varios usuarios, usando dispositivos móviles, puedan realizar la misma consulta simultáneamente en diversos lugares, a una base de datos difusa, donde la respuesta se adecue al contexto implícito y explícito de cada usuario.

En la aplicación web se definieron etiquetas lingüísticas para diferentes atributos del Examen Físico Articular (EFA) que eran factibles de ser tratados como difusos. Entre ellos están el peso, la talla, los tonos musculares, el tipo de marcha y los dispositivos utilizados por un paciente (bastón, silla de ruedas, andaderas, muletas), cuyas definiciones se adecuan a las preferencias de cada usuario. Esto permitió el manejo de la sensibilización al contexto en las consultas que accesan estos atributos, pues se producen resultados diferentes cuando las definiciones de los usuarios no coinciden. Adaptando así los resultados al contexto de cada usuario.

En la interfaz móvil se consideraron las diferentes características y/o limitaciones de estos dispositivos manteniendo la mayoría de las funcionalidades ofrecidas en la versión de escritorio. Sin embargo, hay funcionalidades como el registro de nuevos usuarios, que sólo pueden ser realizadas directamente en la aplicación web. Esto también muestra la adaptación al contexto.

La disposición de una interfaz móvil permite que cualquier médico, usuario de la aplicación, pueda realizar consultas al sistema dondequiera que se localice, obteniendo resultados distintos según sea su perfil (contexto explícito) y/o ambiente en que se ejecuta (contexto implícito). Esto incluye diferentes distribuciones, iconos, valores, cantidad de respuestas, entre otras. Adicionalmente, cuando un médico ingresa al sistema puede revisar su perfil e ingresar datos de nuevos pacientes y nuevos exámenes físico-articulares.

El uso de la lógica difusa como herramienta para especificar las consultas permite mayor expresividad y cercanía al lenguaje humano. Esto se logra con la definición de etiquetas lingüísticas adaptables al usuario, tales como *obeso*, *delgado*, *alto*, *bajo*, *atonía*, *normotonía*, cuyo uso es más intuitivo que expresarlos con valores precisos. Si bien cada etiqueta va de la mano de la percepción de la persona que la utiliza, existen estándares que se manejan en cuanto al cálculo común de expresiones referentes a tonos musculares, peso, forma de caminar, lo que hace que el aporte de la aplicación propuesta sea valorado en entornos médicos.

El desarrollo de esta aplicación implica un avance importante en relación al desarrollo de aplicaciones utilizando bases de datos difusas. En trabajos previos, se ha tenido que modificar el gestor de base de datos para lograr implementar consultas difusas (sin la representación de dominios difusos), lo cual dificulta el mantenimiento y produce incompatibilidad de versiones.

Por otro lado, el uso del modelo objeto relacional de Oracle, permitió la implementación del modelo de bases de datos difusas, con el apoyo de funciones y procedimientos almacenados, escritos en PL/SQL. Los cuales son fácilmente ejecutables, con pocas o casi ninguna modificación, en otras versiones del manejador.

En trabajos futuros se desea lograr la sensibilización al contexto basándose en datos capturados por sensores ambientales, tales como: ubicación del usuario, movilidad y clima.

Agradecimientos

Hacemos un reconocimiento a Jhosbert Contreras apreciado alumno, y actual colega, que colaboró en la culminación de este trabajo en su Miniproyecto de Desarrollo de Software de la Universidad Simón Bolívar. "*Toda buena dádiva y todo don perfecto desciende de lo alto, del Padre de las luces, en el cual no hay mudanza, ni sombra de variación*." (Santiago 1:17).

Referencias

[1] K. A. Dey, *Understanding and Using Context*. Personal and Ubiquitous Computing, vol. 5, no. 1, Febrary 2001.

[2] B. Schilit, N. Adams, R. Want, *Context-Aware Computing Applications, Mobile Computing Systems and Applications*, in proceedings of WMCSA, Santa Cruz, USA, 1994.

[3] J. Favela, A. I. Martinez-Garcia, *Context-Aware Mobile Communication in Hospitals*, Computer 36, no. 9, pp. 38–46, 2003.

[4] M.A. Vila, J.C. Cubero, J.M. Medina, O. Pons, *A Conceptual Approach for Dealing with Imprecision and Uncertainty in Object-based Data Models*, International Journal of Intelligent Systems, vol. 11, pp. 791–806, 1996.

[5] L. A. Zadeh, *A New Direction in A.I. Toward a Computational Theory of Perceptions*, AI Magazine, vol. 22, no. 1, 2001.

[6] G. Koutrika, *Personalized DBMS: an Elephant in Disguise or a Chameleon?*, Data Engineering, vol. 27, 2011.

[7] P. Roocks, M. Endres, S. Mandl, W. Kiebling, *Composition and Efficient Evaluation of Context-Aware Preference Queries*, in proceedings of 17th Int. Conference DASFAA, Busan, South Korea, 2012.

[8] L. A. Zadeh, *Fuzzy Sets*, Information and Control, vol. 8, 1965.

[9] J. T. Cadenas, *Una Contribución a la Interrogación Flexible de Bases de Datos: Optimización y Evaluación a Nivel Físico*. Tesis de Maestría. Universidad Simón Bolívar, Venezuela, Febrero 2008.

[10] L. Yan, Z. Ma, F. Zhang, *Fuzzy Sets and Fuzzy Database Models*, Fuzzy XML Data Management, pp. 31–81. Springer Berlin Heidelberg, 2014.

[11] F. Berzal, N. Marín, O. Pons, M.A. Vila, *Managing Fuzziness on Conventional Object-Oriented Platforms*. International Journal Intelligent Systems, vol. 22, pp. 781–803, 2007.

[12] J. T. Cadenas, N. Marín, M. A. Vila, *Fuzzy Domains with Adaptable Semantics in an Object-Relational DBMS*, Flexible Query Answering Systems; Lecture Notes in Computer Science, vol. 7022, pp. 497–508, 2011.

[13] J. T. Cadenas, N. Marín, M. A. Vila, *Una Primera Aproximación a la Semántica Adaptable al Contexto en Bases de Datos Difusas*, en las memorias de las II Jornadas Andaluzas de Informática, Septiembre 2011.

[14] L. Cuevas, *Modelo Difuso de Bases de Datos Objeto-Relacional: Propuesta de Implementación de Software Libre*. Tesis Doctoral. Universidad de Granada, España, Febrero 2009.

[15] A. Aguilera, R. Rodriguez, *Representación y Manipulación de Datos Médicos en Marcha Patológica*, Multiciencias, vol. 11, no. 1, pp. 76–84, 2011.

[16] APACHE FRIENDS, *XAMPP Apache + MySQL + PHP + Perl*, http://www.apachefriends.org/index.html.

[17] W3C, *HTML & CSS*. http://www.w3.org/standards/webdesign/htmlcss.

[18] @MDO and @FAT, *The Most Popular Front-End Framework for Developing Responsive, Mobile First Projects on the Web*, Bootstrap, http://getbootstrap. com.

[19] MDN, *JavaScript. Mozilla Developer Network*. https://developer. mozilla.org/es/docs/JavaScript.

[20] PHP, *Learn about phpDocumentor, phpDocumentor*. http://phpdoc.org/ docs/latest/index.html.

[21] *Fundación Hospital Ortopédico Infantil*. http://www.ortopedicoinfantil. org/hospital.

[22] J. Tidwell, *Designing Interfaces: Patterns for Effective Interaction Design*. O'Reilly, December 2010.

[23] L. Lindstrom, R. Jeffries, *Extreme Programming and Agile Software Development Methodologies*, CRC Press LLC, 2003.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 44-53, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Enfoque Formal de Verificación y Autómatas Temporizados Aplicados a Procesos Industriales

Luis Mendoza[1]
lmendoza@usb.ve

[1] Departamento de Procesos y Sistemas, Universidad Simón Bolívar, Caracas, Venezuela

Resumen: Una de las metas de la Ingeniería de Software y del Modelado de Procesos de Negocio es lograr que los diseñadores de negocios y los desarrolladores de sistemas construyan aplicaciones de software confiables, principalmente para aquellos *Procesos Industriales Críticos* (PIC); es decir, aquellos procesos que impactan directamente la misión de una industria. El uso de *Métodos Formales*, tales como la técnica de *Verificación Automática* y la teoría de *Autómatas Temporizados* (AT), ha demostrado que propicia la confiabilidad y la seguridad de PIC al aumentar su comprensión, revelando inconsistencias, ambigüedades e incompletitudes, que de otra manera pasan desapercibidas. Este trabajo presenta la integración de un *Enfoque Formal de Verificación* (EFV) con la teoría de AT para la especificación y la verificación automática del *Modelo de Tareas (MT) asociado a un PIC* (MTPI). Además, se introducen un conjunto de directrices para transformar modelos de PIC en AT. El resultado del uso del EFV con la verificación automática y los AT es una infraestructura metodológica que garantiza la exactitud del MTPI con respecto a la especificación de las propiedades iniciales derivadas de las buenas prácticas o reglas de negocio de una industria. Con el fin de mostrar la viabilidad de la propuesta, se discute el caso Celda de Producción, un típico ejemplo para evaluar metodologías para el diseño de sistemas industriales, el cual ilustra cómo integrar la verificación automática en las etapas tempranas del diseño de PIC.

Palabras Clave: Procesos Industriales Críticos; Verificación Automática; Autómatas Temporizados; Especificación; Métodos Formales.

Abstract: One of the goals of Software Engineering and Business Process Modeling is to enable that business designers and system developers build reliable software applications, mainly for those that support *Critical Industrial Processes* (CIP); i.e. those processes that directly impact the mission of an industry. The use of *Formal Methods*, such as the *Model Checking* technique and the *Timed Automata* (TA) theory, has shown that promotes the reliability and safety of CIP to increase their understanding, revealing inconsistencies, ambiguities and incompleteness, which otherwise so go unnoticed. This paper presents the integration of a *Formal Verification Approach* (FVA) with TA theory for the specification and model checking of the *Task Model (TM) associated with a CIP* (MTIP). In addition, a set of guidelines to transform CIP models in TA are introduced. The result of using the FVA with model checking and TA is a methodological infrastructure that guarantees the accuracy of MTIP with respect to the specification of the initial properties derived from good practices or business rules of an industry. In order to show the feasibility of the proposal, a Production Cell case, a typical example to evaluate methodologies for designing industrial systems, is discussed, which illustrates how to integrate model checking on the early stages of CIP design.

Keywords: Critical Industrial Processes; Model Checking; Timed Automata; Specification; Formal Methods.

I. Introducción

Con la automatización de procesos industriales y el soporte que las *Tecnologías de la Información y Comunicación* (TIC) le dan a las industrias, muchos procesos se han convertido en el elemento fundamental para el funcionamiento de las industrias, así como para la realización de gran cantidad de procesos y actividades de la vida diaria. Como resultado de esto, algunos de los procesos industriales que utilizan las organizaciones se han catalogado como críticos —aquellos procesos que impactan directamente la misión de una industria. Esto se debe a que su funcionamiento condiciona el trabajo diario de las industrias y de muchos aspectos de la vida humana, lo que incluye un buen tiempo de respuesta. Como consecuencia, estos *Procesos Industriales Críticos* (PIC) necesitan ser verificados, certificados u homologados, antes de su puesta en funcionamiento [1].

La verificación de los aspectos temporales de un PIC soportado por las TIC puede ser visto como una actividad de tres etapas

que tiene como objetivo validar la viabilidad de los requerimientos, el diseño y la implementación. Para examinar los requerimientos y el diseño, se hace necesario la especificación de un modelo del comportamiento temporal del PIC propuesto. De este modelo se comprueba su confiabilidad y seguridad. Con el fin de expresar los requerimientos temporales y las propiedades de diseño, el esquema de modelado debe contener dos primitivas clave: plazos de espera y tiempos de ejecución [2]; además, debe ser capaz de expresar la concurrencia inherente a los sistemas de tiempo real. La verificación de la implementación de un PIC consiste en predecir el peor comportamiento temporal del PIC cuando todos los retrasos se contabilizan, y la posterior afirmación de que en estas circunstancias todos los plazos se cumplen. Esto implica generalmente la planificación de los tiempos de respuesta y su análisis asociado.

Por ello, para minimizar posibles omisiones o inconsistencias con respecto a lo que se espera de los procesos críticos de una industria, tanto a nivel funcional como a nivel de rendimiento de acuerdo a restricciones temporales, es necesario invertir esfuerzo en la verificación de los PIC que van a ser implementados. Debido a la compleja naturaleza y dinámica de algunas industrias, los modelos basados en lenguajes y técnicas formales —técnicas, lenguajes y herramientas, definidos matemáticamente para especificar y verificar procesos y sistemas— son necesarios para el análisis del comportamiento, a través de la verificación automática y el diseño de PIC, así como para el mejoramiento del funcionamiento de los existentes.

En este trabajo se presenta la integración de un *Enfoque Formal de Verificación* (EFV) con la teoría de *Autómatas Temporizados* (AT) para la especificación y la verificación automática del *Modelo de Tareas asociado a un PIC* (MTPI). También se ofrece un conjunto de directrices de mapeo para transformar los modelos de PIC en AT que pueden ser verificados con UPPAAL [3]. El resultado del uso del EFV con la verificación automática y los AT es una infraestructura metodológica que garantiza la exactitud del MTPI con respecto a la especificación de las propiedades iniciales derivadas de las buenas prácticas o reglas de una industria. Con el fin de mostrar la viabilidad de nuestra propuesta, también se discute un caso de carácter industrial, la Celda de Producción, un típico ejemplo para evaluar metodologías para el diseño de sistemas industriales, el cual ilustra cómo integrar la verificación automática en las etapas tempranas del diseño de PIC. Como resultado, logramos como parte de nuestros objetivos soportar al ingeniero de modelado de PIC con métodos, modelos y herramientas que le permitan verificar los modelos de los PIC que construye, desde el mismo momento que los define y los diseña.

Se han desarrollado varios trabajos que permiten la verificación y el análisis de procesos industriales utilizando algunos lenguajes y formalismos. En [4] se presenta un estudio que examina los esfuerzos de la aplicación de la verificación formal en el modelado de procesos. El autor presenta las técnicas de verificación formal (autómatas, redes de petri y álgebras de proceso) que permiten simular y verificar modelos gráficos de PIC en la fase de diseño de los procesos; y reitera que estas técnicas pueden detectar y corregir errores de los modelos en etapas tempranas del diseño; antes de la implementación. Por otra parte, indica que hay estudios, por ejemplo en [5], para verificar diagramas de máquinas de estado de *Unified Modeling Language* (UML) con verificación automática, que deben ser tomados en cuenta. Por nuestra parte, hemos realizado algunos aportes orientados al diseño y la verificación de sistemas con criticidad, reflejados en [6]. En cuanto a la verificación composicional, está el trabajo expuesto en [7]. Todos estos aportes nos han permitido conformar un EFV. Este enfoque integra la verificación composicional y la técnica de verificación automática, aplicables a la verificación formal de PIC.

Este artículo está estructurado como sigue. En la Sección II se presentan los conceptos relacionados con la verificación de PIC, se expone la teoría de AT y se introduce la herramienta UPPAAL. En la Sección III se presenta el EFV y, a continuación, en la Sección IV, se indican las directrices para transformar un PIC en AT. Posteriormente, en la Sección V, se discute el caso de la Celda de Producción y el uso de la herramienta UPPAAL. Finalmente, en la Sección VI, se enumeran las conclusiones y se esboza el trabajo futuro.

II. MARCO CONCEPTUAL

A. Verificación de Procesos Industriales y Criticidad

Según [8], los procesos industriales tienen una vinculación con los flujos de trabajo (en inglés, workflows), al indicar que éstos son la automatización de un procesos industriales, total o parcialmente, durante la cual materia (prima o semiprocesada), información y/o tareas, son pasadas de un participante a otro por una acción conforme a un conjunto de reglas procedimentales. Para [9], los procesos son secuencias específicas de actividades de trabajo a través del tiempo y del espacio, con un inicio, un fin y unas entradas y salidas claramente definidas: *una estructura para la acción*.

Para este trabajo los procesos industriales con criticidad, denominados PIC, son los procesos industriales de una industria que son esenciales (es decir, críticos) para cumplir la declaración de su misión; es decir, aquellos que deben ser restaurados inmediatamente después de una interrupción para garantizar la capacidad de la industria afectada para proteger sus activos, hacer frente a sus necesidades fundamentales, y cumplir la normativa y los requisitos obligatorios [10]. Esta importancia típicamente se basa en una evaluación de las consecuencias que implicaría un fallo del equipo, actividad o proceso, en el servicio que presta la empresa. A menudo las necesidades de mejora de los PIC se deben a que éstos no llegan a brindar un rendimiento satisfactorio [11]; es decir, no propician el logro positivo de los factores críticos de la organización de la mejor manera. Como parte del rendimiento está el cumplimiento de plazos de tiempo que garanticen su correcta ejecución [11].

La criticidad está asociada con la fiabilidad o confiabilidad (en inglés, reliability) de procesos automatizados; es decir, con

la probabilidad de que un equipo o sistema opere sin fallo por un determinado período de tiempo, bajo unas condiciones de operación previamente establecidas [12]. Adicionalmente, a nivel industrial, se habla de la confiabilidad operacional, referida a la capacidad de una instalación o sistema integrado por procesos, tecnología (máquinas) y gente, para cumplir su función dentro de sus límites de tiempo y bajo un contexto específico de operaciones [13].

Por ello, los conceptos de verificación y validación son importantes en este trabajo. La verificación se enfoca en el tema de la consistencia interna de un modelo, mientras que la validación está relacionada con la correspondencia entre el modelo y la realidad [14]. La validación es el proceso de comparar la salida del modelo del PIC con el comportamiento organizacional. En otras palabras: comparar la ejecución del modelo del PIC con la realidad (física u otra cualquiera). La verificación es el proceso de comparar la implementación (una forma de ejecución) de un PIC (que puede ser un programa o una especificación formal detallada) con el modelo del PIC, para garantizar que la implementación es correcta con respecto al modelo del PIC. El término validación se aplica a aquellos procesos que buscan determinar si una implementación es correcta o no respecto al sistema real. De forma más sencilla, la validación trata sobre la cuestión ¿se está construyendo/modelando el PIC correcto?, mientras que la verificación responde a ¿se está construyendo/implementando correctamente el PIC?. La verificación comprueba que la implementación del modelo (una forma de ejecución detallada y formal) corresponde al modelo del PIC, mientras que la validación comprueba que el modelo del PIC corresponde con la realidad [14].

En este trabajo nos centramos en la verificación formal, ya que por las características de los PIC y el impacto de éstos en las industrias, además de lo complejo que son los sistemas y maquinarias que los soportan, es sumamente costoso y riesgoso esperar que el PIC esté implementado para validar que la implementación cumple con lo que se quiere en la realidad.

B. Teoría de Autómatas Temporizados

El recurso más utilizado actualmente en la verificación del modelo de un procesos industriales es el uso de autómatas que implementan la ejecución del proceso [15]. Los autómatas son modelos matemáticos de un sistema (o proceso) con entradas y salidas, pudiendo estar en cualquiera de un número finito de estados. Cada estado establece la relación entre entradas anteriores para alcanzar ese estado y las entradas posteriores (salidas) para alcanzar un estado siguiente [16]. La entrada es leída símbolo por símbolo (alfabeto del autómata), hasta que es *consumida* completamente —piense en ésta como una palabra que es leída por una cabeza lectora; la cabeza se mueve a lo largo de la palabra, leyendo un caracter (o símbolo) a la vez— una vez la entrada se ha agotado, el autómata se detiene.

El concepto de AT fue propuesto por primera vez por [17] como una extensión de la teoría de autómatas enfocada al modelado de sistemas en tiempo real [16]. De acuerdo con la teoría de AT, un autómata temporizado es un grafo dirigido finito anotado con condiciones y restablecimientos de los valores reales no negativos de relojes, donde un sistema se modela como una colección de máquinas de estados finitos y un conjunto finito de relojes [17]. En el esquema estándar, los relojes están sincronizados y pueden ser restaurados (*reset*) a través del paso de un estado a otro. Los relojes también se utilizan como guardas para las transiciones.

Definición 1: **Autómata temporizado.** Un *autómata temporizado* es una tupla $\mathcal{A} = \langle S, \Sigma, C, E, s_0 \rangle$ que consiste de los siguientes componentes:

- S es un conjunto finito. Los elementos de S son llamados los estados de $\mathcal{A}$.
- Σ es un conjunto finito llamado el alfabeto de acciones de $\mathcal{A}$.
- C es un conjunto finito llamado los relojes de $\mathcal{A}$.
- $E \subseteq S \times \Sigma \times B(C) \times P(C) \times S$ es el conjunto de transiciones de $\mathcal{A}$, donde
  - $B(C)$ es el conjunto de restricciones booleanas de los relojes de C, y
  - $P(C)$ es el conjunto potencia de C.
- s_0 es un elemento de S, llamado el estado inicial.

Una transición $\langle s, a, g, r, s' \rangle \in E$ es el cambio del estado s hasta s' con las acciones a, las guarda g y el reinicio del reloj r .

La teoría de AT estipula que el tiempo es continuo, pero la declaración de los relojes por lo general se limita a la utilización de valores enteros. El tiempo nunca es negativo así como que los relojes sólo pueden ser reiniciados a 0. El tiempo de vida está representado por el requerimiento de que algunos relojes específicos nunca pueden obtener un valor superior a un plazo determinado, o que no pueden alcanzarlo mientras no se supere un estado o una colección de estados. Las transiciones se definen para ser instantáneas. Por ejemplo, de acuerdo con la Figura 1, la transición de salida del estado S no puede ocurrir antes de $T = 3$.

Figura 1: Ejemplo de AT

En este ejemplo, T es un reloj; que se está reiniciando cuando se alcanza el estado S. En un modelo simple (es decir, el invariante $T < 5$ no existe), la única salida del estado S es cuando T es superior a 3; es decir, el ejemplo, ilustra la imposición de un retraso (*delay*). Un estado puede tener también un invariante temporal para forzar una transición de salida. La Figura 1 ilustra esto porque el estado S no puede salir antes de $T = 3$, sino que debe salir después de 3 y antes de $T = 5$. Si por alguna razón la transición no se puede tomar, el autómata contiene una condición de error o punto muerto (deadlock). Cuando hay dos o más posibles transiciones de un estado a otro, cada una es una transición válida.

Tal como se introdujo anteriormente, la semántica de un autómata temporizado se define como un sistema de transición en la que un estado o configuración consiste en una ubicación

(o estado) dada y los valores de los relojes en ese momento. Hay dos tipos de transiciones entre estados. El autómata puede aplazar por un tiempo (una transición de retardo) o bien seguir un camino permitido (una transición de acción). Una *acción temporizada* es un par (t, a), donde $a \in \Sigma$ es una acción tomada por el autómata temporizado $\mathcal{A}$ después de $t \in \mathbf{R}_+$ unidades de tiempo desde que $\mathcal{A}$ se ha iniciado. El tiempo absoluto t se llama *captura de tiempo* de la acción a. Una *traza temporizada* es una secuencia (posiblemente infinita) de acciones temporizadas $x_i = (t_1, a_1)(t_2, a_2)\ldots(t_i, a_i)\ldots$ donde $t_i \leq T_{i+1} \forall i \geq 1$.

Para modelar sistemas concurrentes (como los PIC), los AT pueden ser extendidos con la composición paralela. En las álgebras de procesos, se han propuesto varios operadores de composición paralela para modelar diferentes aspectos de concurrencia —véase, por ejemplo *Calculus of Communicating Systems* (CCS) [18] y *Communicating Sequential Processes* (CSP) [19]. Estos operadores algebraicos pueden adoptarse en la teoría de AT, la cual permite el intercalado de las acciones, así como la sincronización entre AT. Esencialmente, la composición paralela de un conjunto de AT es el producto de los autómatas, denominado *red de AT*.

Definición 2: **Red de AT.** Una *red de AT* es la composición paralela $RAT = \mathcal{A}_1 \parallel \cdots \parallel \mathcal{A}_n$ de un conjunto de autómatas temporizados $\mathcal{A}_1 \ldots \mathcal{A}_n$, llamados sub–procesos, combinados en un único sistema (o proceso) por un operador de composición paralela con la ocultación de todas las acciones internas. La comunicación síncrona entre los sub–procesos es establecida mediante acciones de entrada y salida. Se asume que el alfabeto de acciones constará de los símbolos para las acciones de entrada $c?$, las acciones de salida $c!$ y las acciones internas representadas por el símbolo τ.

Los AT están diseñados para que puedan ser comprobados utilizando la verificación automática. En la verificación automática *se comprueba si un modelo formal es correcto respecto a los requerimientos expresados en una lógica temporal* [20]. Intuitivamente, la verificación automática funciona mediante la exploración de todas las posibles transiciones de estado desde el estado inicial del sistema. Todos las trazas posibles desde el conjunto de estados existentes al principio del autómata se exploran para ver si un estado inseguro puede ser alcanzado o una condición de vivacidad no se cumple. La verificación automática es una técnica que requiere el apoyo de herramientas. En este trabajo se utiliza la herramienta UPPAAL [3], porque es compatible con la representación gráfica de los AT y permite que el usuario de la herramienta pueda interactuar con un programa de edición para crear y modificar modelos de AT; es decir, redes de AT.

C. UPPAAL

UPPAAL[1] [3] es una herramienta de software que permite modelar, validar y verificar sistemas en tiempo real. UPPAAL sirve como un lenguaje de diseño o de modelado que permite describir el comportamiento de un sistema o proceso como redes de AT ampliadas con variables de datos y de reloj.

La herramienta cuenta con un simulador interactivo que es una herramienta de validación que permite examinar las posibles ejecuciones dinámicas de un sistema durante las fases de diseño y proporciona una sencilla detección de los fallos antes de que este sea verificado por el comprobador de modelos. Por su parte, el comprobador de modelos (o verificador automático) puede chequear las propiedades —especificadas en una lógica temporal basada en *Timed Computational Tree Logic* (TCTL), que es una especialización de la semántica de la lógica temporal Computational Tree Logic (CTL), partiendo de la misma sintaxis— explorando el espacio de posibles ejecuciones del autómata. En UPPAAL, el *autómata producto* se calcula *on–the–fly* durante la verificación. También proporciona diagnósticos que se pueden abrir automáticamente a través del simulador y pueden usarse para visualizar e investigar las trazas de símbolos leídos del alfabeto [3].

Desde el punto de vista de los analistas/diseñadores de negocios, UPPAAL es fácil de usar; aunque para modelos complejos se requiere una considerable capacidad de cómputo por parte de la herramienta. Tal como se indicó anteriormente, el formalismo subyacente en UPPAAL es el de redes de AT [3], que es de fácil comprensión para la gente de negocios y reflejan fielmente el comportamiento de los participantes involucrados en la ejecución de un PIC. El verificador de la herramienta soporta una lógica temporal bastante expresiva y el simulador es muy apreciado por los responsables de analizar/modelar procesos, ya que es muy visual para todo el proceso de verificación.

III. ENFOQUE FORMAL DE VERIFICACIÓN

Con el fin de contribuir a la formalización del complejo campo del modelado de PIC, consideramos adecuado la incorporación de un modelo ejecutable de los PIC, conocido como Modelo de Tareas del PIC (MTPI), para lograr este propósito. Un MTPI puede ser considerado como el modelo más básico posible para obtener una descripción completa, exacta, coherente y bien definida del conjunto de actividades y tareas que deben realizar los participantes/máquinas en un PIC para cumplir los objetivos de la industria [21]. La formalización de un MTPI establece las bases para la verificación automática de los modelos de PIC.

Bajo este enfoque, la falta de verificación de propiedades funcionales y no funcionales de propiedades de los MTPI se debe principalmente a problemas semánticos y sintácticos causada por la incorrecta integración de: (1) la formalización de propiedades (metas y objetivos de la industria, representados por un modelo del PIC) en tiempo de diseño [22], y (2) el modelo ejecutable (MTPI) correspondiente en tiempo de ejecución [22] que es necesario derivar de cualquier PIC para llevar a cabo la verificación automática del modelo del PIC. En este trabajo se propone la construcción de un MTPI como una red de autómatas temporizados. Así, las restricciones temporales del modelo del PIC en tiempo de diseño pueden ser especificadas por los autómatas temporizados en tiempo

[1] http://www.uppaal.org

de ejecución. Al proceder de esta manera, podemos obtener la verificación completa de un MTPI a partir de un modelo inicial del PIC de una manera confiable y soportado por una herramienta de verificación automática como UPPAAL.

Sobre la base de los conceptos e ideas anteriores, se propone una adaptación del esquema conceptual, conocido como EFV, que integra la teoría de AT con la verificación automática, el cual se presenta en la Figura 2.

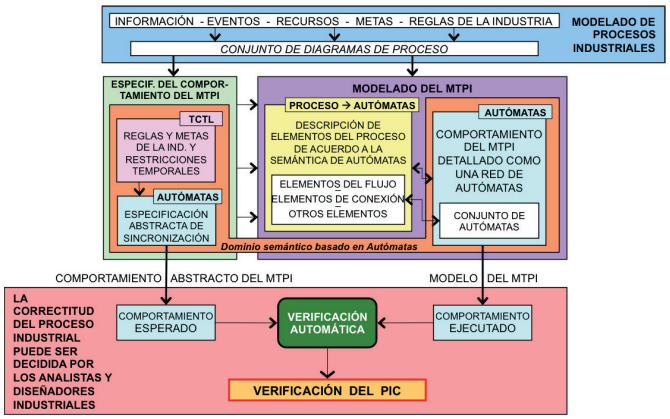


Figura 2: Enfoque Formal de Verificación Utilizando AT

El enfoque presentado en este artículo se basa en el hecho de que el desempeño de un PIC P está estructurado por el trabajo llevado a cabo por varios participantes o componentes (que pueden ser humanos o máquinas) verificados que trabajan en paralelo, $P = ||_{i:1..n} P_i$, donde el comportamiento de cada participante P_i satisface la propiedad ϕ_i, lo que representa la especificación del comportamiento que el participante debe exhibir. El principal objetivo es hacer posible la verificación del comportamiento de un PIC completo a partir de la verificación del comportamiento de sus participantes. En este sentido,

Definición 3: **Composicionalidad de propiedades.** Una propiedad ϕ es composicional si y sólo si para cualesquiera dos AT $\mathbf{A}_1$, $\mathbf{A}'_1$, y $\mathbf{A}_2$ con $\mathbf{L}(\mathbf{A}_2) \cap \mathbf{L}(\phi) = \emptyset$ se mantiene

$$(\mathbf{A}_1 \models \phi) \Rightarrow ((\mathbf{A}_1 \| \mathbf{A}_2 \models \phi) \vee \mathbf{A}_1 \| \mathbf{A}_2 \models \delta)) \quad \text{y} \qquad (1)$$

$$((\mathbf{A}_1 \sqsubseteq \mathbf{A}'_1) \wedge (\mathbf{A}'_1 \models \phi)) \Rightarrow (\mathbf{A}_1 \models \phi) \qquad (2)$$

Es decir, las propiedades locales son preservadas por la composición paralela de componentes cuando el etiquetado es disjunto:

Lema 1: Para dos ATs $\mathbf{A}_1$ y $\mathbf{A}_2$ y propiedades ϕ_1 y ϕ_2 con $\Sigma_1 \cap \Omega_2 = \emptyset$, $\Sigma_2 \cap \Omega_1 = \emptyset$, $\mathbf{L}(\mathbf{A}_1) \cap \mathbf{L}(\mathbf{A}_2) = \emptyset$ se mantiene:

$$((\mathbf{A}_1 \models \phi_1) \wedge (\mathbf{A}_2 \models \phi_2)) \Rightarrow (A_1 \| A_2 \models \phi_1 \wedge \phi_2). \qquad (3)$$

Prueba: Como $\mathbf{L}(\mathbf{A}_1) \cap \mathbf{L}(\mathbf{A}_2) = \emptyset$, podemos concluir que si ϕ_2 se mantiene, ésta no es influenciada por $\mathbf{A}_1$. Dada la asunción $\Sigma_1 \cap \Omega_2 = \emptyset$, $\Sigma_2 \cap \Omega_1 = \emptyset$, $\mathbf{L}(\mathbf{A}_1) \cap \mathbf{L}(\mathbf{A}_2) = \emptyset$, ambos AT son ejecutados independientemente en paralelo y así $\mathbf{A}_2 \models \phi_2$ implicará $A_1 \| A_2 \models \phi_2$, y $\mathbf{A}_1 \models \phi_1$ implicará $A_1 \| A_2 \models \phi_1$. Por consiguiente, $A_1 \| A_2 \models \phi_1 \wedge \phi_2$ ha sido probado.

Cada participante P_i debe satisfacer la invariante (ψ_i), la cual representa el comportamiento de los otros participantes que colaboran en el proceso con respecto a P_i. Se usa el símbolo especial $\neg\delta$ para denotar que el bloqueo del participante (es decir, que el participante caiga en un estado sin ninguna transición de salida) no puede producirse en ninguna ejecución posible en la que colabore. La propiedad ϕ y el invariante ψ que son satisfechos por el sistema P son obtenidas a partir de las propiedades locales ϕ_i ($\bigwedge_{i:1..n} \phi_i \Rightarrow \phi$) y de los invariantes locales ψ_i ($\bigwedge_{i:1..n} \psi_i \Rightarrow \psi$), respectivamente. Como resultado, podemos obtener la verificación completa del PIC utilizando el Teorema 1[2]:

Teorema 1: **Verificación de PIC.** Sea el PIC P estructurado en varios participantes trabajando en paralelo, $P = \|_{i:1..n} P_i$. Para un conjunto de $AT(P_i)$ que describen el comportamiento de los participantes P_i, las propiedades ϕ_i, los invariantes ψ_i, y el bloqueo δ, con $\bigcap_{i:1..n} \Sigma_i = \emptyset$, $\bigcap_{i:1..n} \Omega_i = \emptyset$, y $\bigcap_{i:1..n} \mathbf{L}(AT(P_i)) = \emptyset$, la siguiente condición se comple:

$$AT(P) \models (\phi \wedge \psi \wedge \neg\delta) \Leftrightarrow \|_{i:1..n} AT(P_i) \models \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta, \qquad (4)$$

donde $AT(P) = \|_{i:1..n} AT(P_i)$.

Prueba: Asumimos que el comportamiento del participante P_i puede ser descrito por el $AT(P_i)$, a fin de:

$$AT(P_i) \models (\phi_i \wedge \psi_i) \wedge \neg\delta \ .$$

Por el operador de composición paralela de los cálculos de procesos, la siguiente relación debe ser satisfecha:

$$AT(P) = \|_{i:1..n} AT(P_i) \ .$$

Si los $AT(P_i)$ satisfacen,

1) $\bigcap_{i:1..n} \Sigma_i = \emptyset$ y $\bigcap_{i:1..n} \Omega_i = \emptyset$, y
2) $\bigcap_{i:1..n} \mathbf{L}(AT(P_i)) = \emptyset$,

podemos afirmar que éstos pueden ser compuestos y, por el Lema 1, escribimos:

$$\|_{i:1..n} AT(P_i) \models \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta$$

Dado que la relación de satisfacción es cerrada con respecto al operador de conjunción, concluimos $\bigwedge_{i:1..n} \phi_i \Rightarrow \phi$, $\bigwedge_{i:1..n} \psi_i \Rightarrow \psi$, y por consiguiente:

$$AT(P) \models \phi \wedge \psi \wedge \neg\delta \Leftrightarrow \|_{i:1..n} AT(P_i) \models \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta \ . \qquad (5)$$

La condición suficiente $\phi \Rightarrow \bigwedge_{i:1..n} \phi_i$, $\psi \Rightarrow \bigwedge_{i:1..n} \psi_i$, puede ser demostrada considerando que aun en el caso donde todos los participantes P_i presenten un comportamiento diferente,

[2]Se denota $AT(P) \models \phi$ cuando el $AT(P)$ mantiene la propiedad ϕ para todos sus estados iniciales, es decir, $\forall s \in I \Rightarrow (AT(P), s) \models \phi$. Para los invariantes tenemos $AT(P) \models \psi$ cuando para todos los $s \in S$ se mantiene $(AT(P), s) \models \psi$. Para denotar que $AT(P)$ no contiene algún bloqueo, escribimos $AT(P) \models \neg\delta$.

ϕ y ψ podrán ser definidas como la conjunción de las propiedades locales de cada componente (ϕ_i, ψ_i).

La aplicación práctica del Teorema 1 incluye realizar un proceso de *chequeo de satisfacción* inductivo en el rango del número de participantes ($i : 1..n$) de nuestro PIC. La herramienta UPPAAL nos soporta en este chequeo, de acuerdo a los siguientes pasos de aplicación del EFV y del Teorema 1. El EFV consta de los siguientes procesos integrados de acuerdo a la técnica de verificación automática y la teoría de AT (ver Figura 2):

1) *Modelado del MTPI*. Se modela y especifica el comportamiento del MTPI con AT, obteniéndose la descripción completa de su comportamiento. Para lograr esto, se utilizan las directrices que se exponen en la Sección IV de este artículo. De esta forma, se obtiene la especificación del comportamiento de los participantes (personas y/o máquinas) que colaboran en el MTPI en tiempo de ejecución, de acuerdo a la secuencia discreta de eventos en los cuales los participantes del MTPI están involucrados.
2) *Especificación del comportamiento de MTPI*. En paralelo a lo anterior, los requerimientos y las restricciones temporales que el MTPI debe cumplir son especificados a través de fórmulas de la lógica temporal TCTL, que es interpretada por UPPAAL. Posteriormente, estas propiedades son expresadas como AT de manera automática por UPPAAL . Como consecuencia, las propiedades del MTPI se expresan en el mismo lenguaje de especificación que el modelo del MTPI (es decir, AT), pudiendo razonar y ejecutar la verificación automática en un mismo dominio semántico.
3) *Verificación del MTPI*. Finalmente, obtenidos los resultados de los procesos anteriores realizados en paralelo, procedemos a la verificación del comportamiento del MTPI vs. el comportamiento abstracto esperado de éste. Esto se realiza a través de la incorporación de la herramienta de software UPPAAL, la cual permite simular y analizar este comportamiento de una manera cómoda para los diseñadores de PIC.

IV. DIRECTRICES PARA TRANSFORMAR PIC A AT

Una de las contribuciones de este trabajo es proponer un conjunto sencillo de pasos para derivar el MTPI correspondiente al modelo de un PIC, como una red de AT. Esta transformación es la primera etapa necesaria para realizar la evaluación (es decir, el análisis cualitativo) de un PIC, tal como se mostró en la Sección III. Después, la red de AT generada por la transformación, lo que representa el MTPI, se analiza con el apoyo de UPPAAL, para proporcionar diversos indicadores de rendimiento empresarial (por ejemplo, tiempo de servicio, tiempo o tamaño de la cola) de un PIC.

Tal como ya se ha dicho, un PIC consiste en un conjunto de actividades coordinadas que se ejecutan a lo largo un entorno organizacional y técnico [23]. Un PIC se estructura en una serie de tareas que deben ser realizadas por participantes (personas o sistemas automatizados) limitados por un conjunto de condiciones de negocio y en un plazo de tiempo específico. Una tarea es una unidad atómica y lógica del trabajo que se lleva a cabo por completo por un recurso [24]. En estas tareas, los participantes colaboran siguiendo un flujo de trabajo (o proceso) definido por un servicio dado. Además, los PICs se ven limitados por un conjunto de reglas de negocio, las cuales deben ser acatadas y determinan la estructura de la información y las políticas de la industria [25].

El proceso (o sub–proceso) que ejecuta un participante pasa por varios estados. Estos estados están unidos por la secuencia de ejecución que representan las transiciones de estado especificados por el proceso. Las transiciones pueden entrar o salir de un estado y tienen asociadas las guardas. Los pases de mensajes corresponden a la sincronización de sub–procesos (o procesos separados).

Para transformar el modelo de un PIC en una red de AT, el modelo del PIC es separado en tres aspectos: el sub–proceso que modela el comportamiento de cada participante, representado por un autómata, las relaciones entre los sub–procesos que colaboran para lograr el comportamiento del proceso, representadas por la red de AT, y las restricciones temporales del PIC, derivadas de las reglas de negocio. En pocas palabras, los dos primeros son mapeados en los AT y en la sincronización entre AT (es decir, para conformar la red de AT correspondiente), respectivamente, y las restricciones son mapeadas como invariantes, guardas y asigaciones temporales en los AT que conforman la red.

A. Participantes

El comportamiento de un participante en un PIC se modela como un sistema de transición de estados o autómata, que se caracteriza por sus acciones por los canales de sincronización, las invariantes, guardas, y asignaciones. La siguiente definición formaliza el mapeo propuesto.

Definición 4: **Participante.** Un *participante* modelado en un PIC, corresponde a un autómata temporizado $\mathcal{A} = \langle S, \Sigma, C, E, s_0 \rangle$ que consiste en los siguientes componentes:

- S es un conjunto finito de los estados de $\mathcal{A}$.
- Σ es un conjunto finito llamado el alfabeto o acciones (entrada y salida) de $\mathcal{A}$, que es la comunicación sincrónica entre participantes.
- C es un conjunto finito de los relojes de $\mathcal{A}$.
- $E \subseteq S \times \Sigma \times B(C) \times P(C) \times S$ es el conjunto de transiciones de $\mathcal{A}$, donde
  - $B(C)$ es el conjunto de restricciones booleanas de los relojes de C, que representan las restricciones temporales, y
  - $P(C)$ es el conjunto potencia de C.
- s_0 es un elemento de S, llamado el estado inicial.

Una transición $\langle s, a, g, r, s' \rangle \in E$ es el cambio del estado s hasta s' con acciones a, guarda g y el reinicio del reloj r.

Las constantes referidas a los invariantes, las guardas y las asignaciones, son parámetros temporales; son valores que se deciden en función de otros aspectos; es decir, las relaciones

entre los participantes y las restricciones temporales establecidas por el modelo del PIC. Por ejemplo, para especificar el comportamiento de las tareas, los invariantes ($t \leq max$), las condiciones de las guarda ($t \geq max$) y las asignaciones o reinicio de los relojes ($t := 0$) se definen en los estados y transiciones correspondientes en el autómata temporizado. Por lo tanto, t es una variable de reloj que tiene el tiempo transcurrido de la tarea y el invariante y la guarda especifican que la transición de la tarea a cualquier otra tarea del flujo no puede ocurrir hasta que el tiempo de ejecución mínimo haya transcurrido (min), y debe ocurrir antes de que el tiempo máximo de ejecución haya transcurrido (max). Las constantes max y min son parámetros temporales.

B. Sincronización de Participantes

Un participante dentro de cualquier industria *debe* interactuar con otros participantes dentro de la misma industria o con otros participantes de diferentes empresas para llevar a cabo por completo un PIC. Por lo tanto, para obtener la representación formal del MTPI (ver la Figura 2), que describe cómo los participantes realizan las tareas y colaboran entre ellos, la red de AT debe ser construida. Como resultado, se obtiene una red de AT que especifica y establece los aspectos del comportamiento y de las restricciones temporales de los participantes involucrados en la ejecución del MTPI.

Definición 5: **Sincronización de participantes.** La *sincronización de participantes* se obtiene como la composición paralela $RAT = \mathcal{A}_1 \parallel \cdots \parallel \mathcal{A}_n$ de un conjunto de AT $\mathcal{A}_1 \ldots \mathcal{A}_n$, denominados participantes, combinados en un único sistema por un operador de composición paralela con la ocultación de todas las acciones internas. La comunicación síncrona entre los participantes se establece mediante acciones de entrada y salida a través de un canal c. Se asume que el alfabeto de acciones consta de los símbolos para las acciones de entrada $c?$, las acciones de salida $c!$ y las acciones internas representadas por el símbolo τ.

C. Guía para Realizar la Transformación

Según nuestra experiencia, la transformación de un modelo/diagrama de un PIC en una red de AT se puede lograr a través de los siguientes pasos:

1) *Incluir restricciones temporales en el modelo/diagrama del PIC*. Para la verificación de las propiedades temporales de un PIC, como el tiempo de respuesta de un participante, y las restricciones temporales que puedan derivarse de las reglas de negocio, tales como el tiempo de ejecución de las actividades, éstas deberán especificarse en el modelo/diagrama del PIC. Se recomienda utilizar algún tipo de anotación permitida por la técnica de modelado utilizada para diagramar el PIC, con la finalidad de facilitar el trabajo de los analistas/diseñadores de PICs.
2) *Obtener el autómata temporizado correspondientes a cada participante*. Con referencia a la Definición 4, se construye el autómata temporizado necesario para especificar el comportamiento de cada trabajador y que formaliza el flujo de trabajo que debe seguir el trabajador para colaborar en la ejecución del PIC.
3) *Asignar a los parámetros correspondientes del autómata temporizado los valores de las restricciones temporales indicados en el modelo/diagrama del PIC*. De acuerdo con las anotaciones indicadas en el modelo/diagrama del PIC, se definen los invariantes, las guardas y las asignaciones en el autómata temporizado.
4) *Obtener la red de TAs correspondiente a la colaboración entre participantes*. A partir de la Definición 5, se construye la red de AT que conforma el MTPI que especifica los aspectos de comportamiento y las restricciones temporales de los participantes que colaboran en el MTPI. La asignación de los canales en los AT por donde se transmiten las acciones de salida y de entrada se establece de acuerdo a la sincronización de los participantes. En función de las restricciones de concurrencia podría ser necesario cambiar la estructura de la red de AT.

Como resultado de aplicar los pasos anteriores, se construyen los AT (como los que se muestran en la Figura 5 y en la Figura 6 de la Sección V) y la red de AT es integrada (como se muestra en la Figura 7 de la próxima sección), con el fin de llevar a cabo el análisis cualitativo del PIC de una manera fácil, a un adecuado nivel de formalidad.

V. Celda de Producción

La Celda de Producción es un ejemplo típico para evaluar metodologías para el diseño de PIC automatizados [26]. Es un modelo de una instalación industrial para el procesamiento de metal en Karlruhe, Alemania. Desafortunadamente, la especificación original de la Celda de Producción no contiene ninguna restricción temporal y se ignora la sincronización temporal. Se han propuesto una serie de estudios de caso avanzados de la Celda de Producción, pero muchos de ellos son bastante diferentes (incluso de acuerdo a cuestiones temporales). En este trabajo nos quedamos con la especificación realizada en [27], la cual es una simplificación y le añade el tiempo en todas las operaciones y requerimientos. Esto tiene un impacto importante en cualquier diseño propuesto. Como se puede ver en la Figura 3, el modelo incluye varias máquinas que deben coordinarse con el fin de forjar piezas de metal [27].

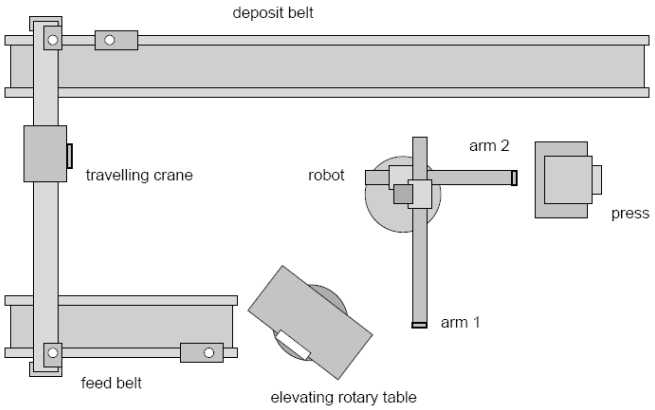


Figura 3: Vista Superior de la Celda de Producción [26]

Para modelar con realismo el comportamiento de la Celda de Producción, hay que considerar que todas las tareas toman un tiempo para su ejecución. Suponemos que la mayoría de las operaciones tienen un tiempo fijo; sin embargo, como un ejemplo de una especificación más real, para las tareas de la prensa se especifican los valores máximos y mínimos de ejecución. Por tanto, cualquier valor entre estos límites es aceptable.

Los requerimientos temporales sobre la Celda de Producción están vistos según el movimiento de una placa desde que entra hasta que sale de la celda. Por ello, se establece que el tiempo del peor caso para una placa (desde la llegada de la banda de alimentación hasta su remoción de la banda de depósito) es 200 unidades de tiempo [27]. Esto representa un tiempo límite para el modelo; es decir, una propiedad de vivacidad a verificar. En la definición original de la Celda de Producción no hay ninguna propiedad de seguridad; sin embargo, en el diseño que se hace en la próxima sección se introducen algunas obligaciones, que se traducen en invariantes de seguridad.

Tenga en cuenta que la definición textual o con alguna anotación gráfica sobre la Celda de Producción corresponde a la etapa 1 de la guía para la realización de la transformación presentada en la Sección IV. En la siguiente sección, estas anotaciones se utilizan para cumplir con el paso 3 de esta guía y para obtener el MTPI del PIC.

A. *Modelado y Verificación*

Ahora procedemos a modelar el MTPI; es decir, la ejecución y la sincronización de la red de AT de los participantes implicados en el MTPI de la Celda de Producción. Para obtener este modelo, se aplicaron las Definiciones 4 y 5, de acuerdo a los pasos 2, 3 y 4, de la guía para la realización de la transformación presentada en la Sección IV.

A continuación se muestra de forma resumida el modelado de uno de los elementos centrales del sistema con la herramienta UPPAAL: la prensa. La tarea de la prensa es forjar piezas de metal. La prensa consiste de dos planchas horizontales, donde la plancha inferior se mueve a lo largo del eje vertical. La prensa opera presionando la plancha inferior contra la plancha superior. Debido a que los brazos del robot están colocados en diferentes planos horizontales, la prensa tiene tres posiciones (ver Figura 4). En la posición inferior, la prensa es descargada por el brazo 2 del robot, mientras que en la posición del medio es cargada por el brazo 1.

La operación de la prensa es coordinada con los brazos del robot como sigue: primero, la prensa se abre en la posición inferior y espera hasta que el brazo 2 retire la pieza de metal y deje la prensa; a continuación, se mueve la plancha inferior a la posición del medio y espera hasta que el brazo 1 cargue la prensa y se retire; finalmente, la prensa se cierra forjando la pieza de metal. Esta secuencia de procesamiento es ejecutada cíclicamente. En la Figura 5 y en la Figura 6 se pueden observar los autómatas, realizados en la herramienta UPPAAL, que modelan el comportamiento de la prensa y el robot, respectivamente. Se puede apreciar que el robot es el componente más complejo de la Celda de Producción.

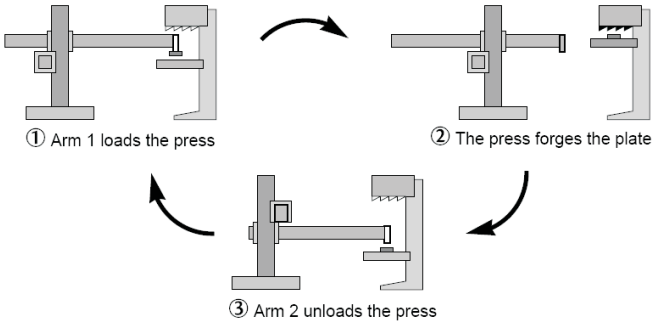


Figura 4: Vista Lateral de la Prensa y de los Brazos del Robot de la Celda de Producción [26]

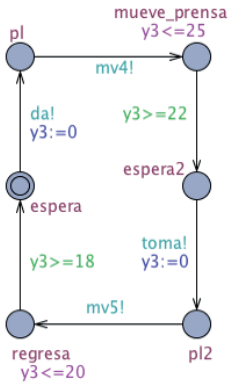


Figura 5: Autómata de la Prensa de la Celda de Producción

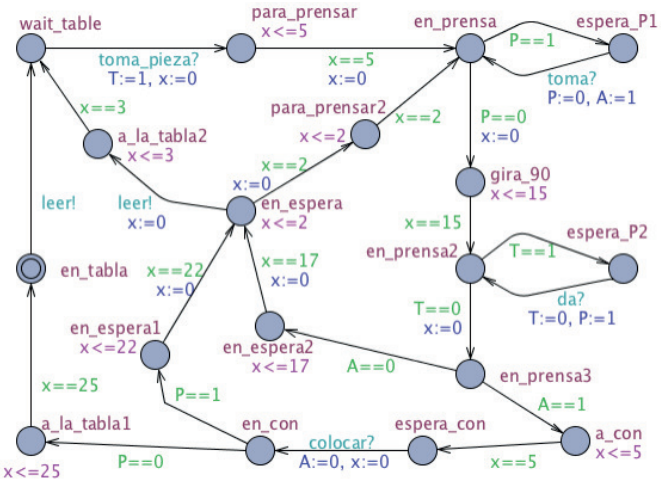


Figura 6: Autómata del Robot de la Celda de Producción

En la Figura 7 se muestra una pantalla del simulador de la herramienta UPPAAL, el cual contiene la red de AT que conforman el MTPI de la Celda de Producción. A través del simulador es posible apreciar la ejecución del MTPI completo, lo que permitió validar que el modelo funciona y que se puede proceder a la verificación de las restricciones temporales que debe satisfacer, en función de las reglas de la industria. En el caso de una contradicción o bloqueo al momento de simular el PIC a verificar (es decir, no se ha procedido a introducir las fórmulas en TCTL al verificador, tal como se muestra más adelante), el analista puede ajustar los parámetros de las restricciones temporales hasta conseguir los valores ideales

para el PIC modelado. Esto, de manera informal, corresponde a la verificacin de la ausencia de interbloqueos del MTPI.

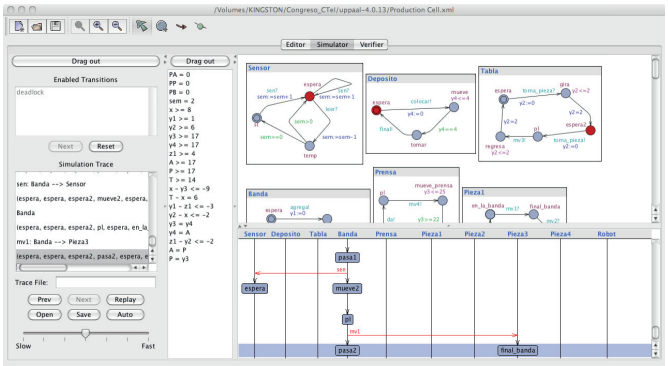


Figura 7: Captura de Pantalla del Simulador de UPPAAL para la Celda de Producción

En la Figura 8 se presenta la pantalla del verificador de la herramienta UPPAAL, donde se muestra la verificación de algunas propiedades del MPTI. Estas propiedades se especifican con una variación de la lógica temporal TCTL [3].

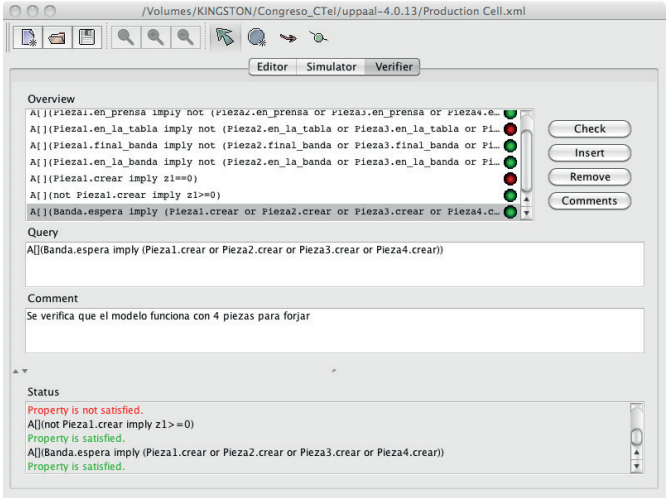


Figura 8: Captura de Pantalla del Verificador de UPPAAL para la Celda de Producción

Tal como se puede ver en la Figura 8, aquellas propiedades que aparecen con una señal verde en el lado derecho han sido verificadas con éxito; mientras que las que aparecen con la señal roja no han sido satisfechas por el modelo presentado en la Figura 7. Por ejemplo, UPPAAL indica que la siguiente propiedad se cumple (ver última propiedad en la lista con señal verde):

```
A[](Banda.espera imply (Pieza1.crear
or Pieza2.crear or Pieza3.crear or
Pieza4.crear)) ,
```

ya que se encontró que un mínimo de 4 piezas es necesario para mantener el modelo trabajando para analizar su comportamiento. Si se trabaja sólo con tres piezas, la Celda de Producción no se mantiene ocupada. Para asegurar que las piezas no se pueden adelantar entre sí, es necesario demostrar que sus ubicaciones están ocupadas bajo exclusión mutua. Es decir, si la Pieza1 está en el estado `en_tabla`, entonces otra pieza no puede estar en ese mismo estado. La excepción de esta regla es, por supuesto, estar en el estado `crear`. El resultado de verificación de estas propiedades son las que aparecen en la parte superior de la propiedad anterior.

La propiedad de vivacidad que se indicó al principio de esta sección, está representada por la siguiente fórmula de TCTL: `A[](not Pieza1.crear imply z1 <= 200) .`

Para este tipo de propiedad (acotada) de vivacidad, es posible reducir el límite superior hasta que la afirmación se convierta en falsa. El contra–ejemplo contrario puede ser observado con la condición:

`A[](not Pieza1.crear imply z1 < I97) ,`

que falla y que el simulador muestra la razón. En esta situación, tres piezas están delante la Pieza 1 y dos están por detrás, lo cual implica que hay 5 piezas para mantener trabajando el modelo bajo esta condición; pero anteriormente establecimos que trabajamos con 4 piezas, lo cual genera una contradicción.

VI. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo hemos presentado la integración de un EFV con la teoría de AT para la especificación y la verificación automática del MTPI. El uso del EFV con la verificación automática y los AT ha resultado en una infraestructura metodológica que garantiza la exactitud del MTPI con respecto a la especificación de las propiedades iniciales derivadas de las buenas prácticas o reglas de una industria. También se han propuesto un conjunto sencillo de lineamientos para la transformación de modelos de PIC a redes de AT. Este mapeo es la primera etapa necesaria para realizar la evaluación (es decir, el análisis cualitativo) de un PIC.

Se ha mostrado la viabilidad de la propuesta, a través de incorporación de la herramienta UPPAAL en la verificación de la Celda de Producción, un ejemplo de carácter académico para evaluar metodologías para el diseño de sistemas y procesos industriales. De esta manera, se ha ilustrado cómo integrar la verificación automática en las etapas tempranas del diseño de PIC. Finalmente, se la logrado demostrar que es posible soportar al ingeniero de modelado de procesos con métodos, modelos y herramientas que le permitan verificar los modelos de los PIC desde el mismo momento que los diseña.

Como trabajo futuro, se estima la conformación de las reglas de transformación que permitan concebir una herramienta de software para automatizar la obtención de los AT que reflejan fielmente los modelos de PIC realizados en alguno de los lenguajes gráficos no formales que normalmente se utilizan en la industria.

AGRADECIMIENTOS

Esta investigación fue parcialmente soportada por el Fondo Nacional de Ciencia, Tecnología e Innovación (FONACIT), mediante el proyecto G–2005000165.

REFERENCIAS

[1] P. Rodríguez, R. Alonso, and J. Snchez, *¿Cuál es la Madurez que Necesitarían los Procesos para el Desarrollo de Sistemas de Software Crítico?*, Revista Española de Innovación, Calidad e Ingeniería del Software, vol. 1, no. 2, pp. 31–41, 2005.

[2] A. Burns and A. Wellings, *Real–Time Systems and Programming Languages*, 3rd edition, Addison–Wesley Longman Publishing Co., Inc., 2001.

[3] G. Behrmann, A. David, and K. Larsen, *A Tutorial on* UPPAAL, Lecture Notes in Computer Science (LNCS), vol. 3185, pp. 200–236, 2004.

[4] S. Morimoto, *A Survey of Formal Verification for Business Process Modeling*, Lecture Notes in Computer Science (LNCS), vol. 5102, pp. 514–522, 2008.

[5] W. Yeung, K. Leung, J. Wang, and W. Dong, *Modelling and Model Checking Suspendible Business Processes Via Statechart Diagrams and CSP*, Science of Computer Programming, special Issue on: Increasing Adequacy and Reliability of EIS, vol. 65, no. 1, pp. 14–29, 2007.

[6] L. Mendoza, M. Capel, M. Pérez, and K. Benghazi, *Compositional Model–Checking Verification of Critical System*, Lecture Notes in Business Information Processing (LNBIP), vol. 19, pp. 213–225, 2009.

[7] M. Capel, L. Mendoza, and K. Benghazi, *Automatic Verification of Business Process Integrity*, International Journal of Simulation and Process Modelling, vol. 4, no. 3/4, pp. 167–182, 2008.

[8] L. Fischer, *2009 BPM and Workflow Handbook: Spotlight on Government*, BPM Handbook Series, Future Strategies, 2009.

[9] W. Kettinger, J. Teng, and S. Guha, *The Process Reengineering Life Cycle Methodology: A Case Study*, in Business Process Change: Reengineering, Concepts, Methods and Technologies, Idea Group Pub., pp. 211–244, 1998.

[10] M. Mijares and L. Mendoza, *Conceptual Model for the Specification of the Quality Properties of the Critical Business Process*, in proceedings of the 8th Iberian Conference on Information Systems and Technologies (CISTI 2013), vol. 1, pp. 315–322, Silvares, Portugal, June 2013.

[11] L. Meade and K. Rogers, *Selecting Critical Business Processes: A Case Study*, Engineering Management Journal, vol. 13, no. 4, pp. 41–46, 2001.

[12] ISO/IEC JTC 1/SC 7, *ISO/IEC 25000:2014 Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE*, ser. 35.080: Software, International Organization for Standardization, 2014.

[13] ISO/TC 176/SC 1, *ISO 9000:2005 Quality Management Systems – Fundamentals and Vocabulary*, ser. ISO 9000 Quality Management, International Organization for Standardization, 2005.

[14] R. Pressman, *Software Engineering: A Practitioners Approach*, 7th edition, McGraw–Hill, 2009.

[15] C. Baier and J.-P. Katoen, *Principles of Model Checking*, Cambridge, The MIT Press, 2008.

[16] J. Hopcroft, R. Motwani, and J. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 3rd edition, Prentice Hall, 2006.

[17] R. Alur and D. Dill, *A Theory of Timed Automata*, Theoretical Computer Science, vol. 126, no. 2, pp. 183–235, 1994.

[18] R. Milner, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science (LNCS), vol. 92, 1980.

[19] C. Hoare, *Communicating Sequential Processes*, International Series in Computer Science, Prentice–Hall International Ltd., 1985.

[20] E. Clarke, O. Grumberg, and D. Peled, *Model Checking*, The MIT Press, 2000.

[21] C. Duursma and U. Olle, *Task Model Definition and Task Analysis Process*, Technical Report, Brussels KADSII /M5/VUB/RR/004/2.0., Vrije University, 1994.

[22] W. Aalst, *Challenges in Business Process Analysis*, Lecture Notes in Business Information Processing (LNBIP), vol. 12, pp. 27–42, 2009.

[23] M. Weske, *Business Process Management: Concepts, Languages, Architectures*, Springer Berlin Heidelberg, 2007.

[24] C. Ouyang, E. Verbeek, W. Aalst, S. Breutel, M. Dumas, and A. Hofstede, *Formal Semantics and Analysis of Control Flow in WS–BPEL*, Science of Computer Programming, vol. 67, no. 2–3, pp. 162–198, 2007.

[25] M. Ortín, J. García, B. Moros, and J. Nicolás, *El Modelo de Negocio como Base del Modelo de Requisitos*, in Actas de las Jornadas de Ingeniería de Requisitos Aplicada, Sevilla, Spain, 2009.

[26] J. Lilius and I. Porres, *The Production Cell: An Exercise in the Formal Verification of a UML Model*, Technical Report, vol. 288, Turku Centre for Computer Science, 1999.

[27] A. Burns, *How to Verify a Safe Real–Time System: The Application of Model Checking and Timed Automata to the Production Cell Case Study*, Real–Time Systems, vol. 24, no. 2, pp. 135–151, 2003.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 54-62, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Sistema Informático para el Aprendizaje de la Guitarra Eléctrica bajo una Estrategia de Juego Interactivo

German Rodriguez [1], Kristian Cortés [1], José Suarez [1], Wilmer Pereira [1, 2]
garodriguezr.11@gmail.com, kristcort@gmail.com, jsuarez@ucab.edu.ve, wpereira@ucab.edu.ve

[1] Escuela de Ingeniería Informática, Universidad Católica Andrés Bello, Caracas, Venezuela
[2] Departamento de Computación, Universidad Simón Bolívar, Caracas, Venezuela

Resumen: Este trabajo ofrece una alternativa automatizada para el aprendizaje musical de la guitarra eléctrica, usando estrategias de juegos interactivos. Esta herramienta para la enseñanza musical, se implementó para dispositivos móviles con conexión a internet, permitiendo la comunicación con un servidor que brinda la información de texto, imágenes y sonidos; además permite la evaluación de los usuarios y medir el progreso del juego mediante preguntas. En el servidor se procesa la información que permite comparar notas, acordes y ritmos musicales emitidos por el guitarrista. Esto alimenta la dinámica del juego llevando las estadísticas de cada usuario. Así cada estudiante puede comparar sus resultados con el de los compañeros que también utilizan la aplicación. El sistema fue validado como herramienta para la enseñanza musical contra el método presencial tradicional, ya que se compararon los resultados de evaluaciones teóricas y prácticas para dos grupos de estudiantes de guitarra. El primer grupo estuvo conformado por personas que hicieron uso de la aplicación para aprender guitarra y el segundo por personas que asistían regularmente a clases presenciales de guitarra eléctrica. Los resultados muestran el potencial de la aplicación y las posibles mejoras para lograr un aprendizaje más eficiente.

Palabras Clave: Aprendizaje Musical; Juego Interactivo; Aplicación Cliente/Servidor; Reconocimiento de Notas; Acordes y Rítmica.

Abstract: This paper provides an automated alternative to the musical learning of the electric guitar, using interactive games strategies. This tool for music education was implemented for mobile devices with internet access, enabling communication with a server that provides the information in text format, images and sounds; also allows the user evaluation and measure progress of the game by questions. On the server the information is processed to compare notes, chords and rhythms issued by guitarist. This feeds the dynamics of the game, showing statistics for each user´s score. So every student can compare their results with those of colleagues who also use the application. The system was applied and validated as a tool for teaching music in compare with the traditional classroom method, the results of theoretical and practical evaluations for two groups of guitar students were compared. The first group consisted of people who took the application to learn guitar and the second by people who regularly attended face classes of electric guitar. The results show the potential of the application and possible improvements for a more efficient learning.

Keywords: Musical Learning; Interactive Game; Client/Server Application; Recognition of Notes; Chords and Rhythmic.

I. Introducción

El aprendizaje convencional de la teoría de la música y la ejecución de un instrumento, es un proceso lento que requiere mucha práctica, específicamente un trabajo constante y metódico de repetición. Los avances del aprendiz dependen de la disponibilidad de los profesores, una gran cantidad de tiempo y muchas veces recursos financieros del principiante. Algunas personas tienen la disponibilidad para cursos presenciales mientras que hay estudiantes, con otro tipo de obligaciones y responsabilidades, que les conviene un método más flexible. Más aún para este tipo de personas la disponibilidad debe ser inmediata para maximizar el aprovechamiento del tiempo. Es básicamente por esta razón que surge la necesidad de implantar la aplicación para plataformas móviles y así aprovechar cualquier ocasión para el autoaprendizaje y mantener al estudiante inmerso en su objetivo de mejorar su nivel como ejecutante de guitarra.

II. Trabajos Relacionados

Además de los manuales publicados por revistas especializadas de enseñanza de guitarra eléctrica y algunas aplicaciones sencillas vía Web, se está realizando mucha investigación en programas para el aprendizaje de diferentes tipos de instrumentos musicales. Específicamente se está proponiendo cada vez más el uso de la realidad aumentada para potenciar las posibilidades de ejecución del practicante. En [1], Liarokapis propone un sistema, en particular para guitarra, donde el aprendiz mediante realidad aumentada, puede ver claramente la posición de la mano y dedos sobre los

trastes y escuchar los sonidos. Esto claramente es de gran utilidad para el correcto posicionamiento ante el instrumento. No obstante no recibe ninguna retroalimentación de su ejecución sonora ni sobre su posicionamiento con respecto a la guitarra. Es decir, el sistema no le informa sobre sus avances y presupone que el estudiante tiene un conocimiento mínimo de teoría musical. En consecuencia no hay un seguimiento del aprendiz durante su trabajo con el sistema.

Por otro lado, en [2] al igual que el artículo de Liarokapis, este trabajo ofrece indicaciones con realidad aumentada de las buenas posiciones del aprendiz pero a diferencia del anterior artículo, utiliza una cámara para hacer reconocimiento de patrones para asegurarse de que el estudiante efectivamente si manipula la guitarra de la manera adecuada. Tampoco graba la ejecución del aprendiz ni la compara con una base de datos para verificar la exactitud de la ejecución. Tampoco graba sonidos …

El sistema propuesto en este artículo si graba el trabajo del practicante: notas, acordes y pautas rítmicas y compara su ejecución con patrones almacenados, en formato MP3, para asegurarse del desempeño del estudiante y poder ofrecer retroalimentación e indicaciones.

En lo que concierne a video juego y guitarra eléctrica sin ninguna duda la aplicación más conocida en el área es *guitar hero*. Este producto comercial ofrece al jugador melodías que debe seguir lo más fielmente posible mediante un dispositivo cuya interfaz es parecida a una guitarra eléctrica Gibson SG. El jugador aprende por repetición aunque no se le ofrece ningún conocimiento de música ni retroalimentación que le indique como mejorar su desempeño.

De hecho cuando es necesario dar retroalimentación sonora los problemas técnicos son numerosos pues las fuentes de ruido son dificiles de evitar y/o filtrar. Hay patentes que intentan resolver el problema como en [3] y [4] mediante sistemas computarizados para el reconocimiento de melodías. Específicamente en [3] se pretende enseñar a tocar cualquier instrumento agregando componente de video juegos competitivos para motivar al estudiante. En el caso del piano cambian la notación clásica de partituras por una nueva escala que se lee verticalmente. La segunda patente [4] es mucho más específica para el aprendizaje de la guitarra y da más indicaciones sobre cómo convertirse en un buen ejecutante, sin dejar de lado el aprendizaje de la teoría de la música.

III. Procesamiento de Sonido

En un primer momento se realizó un pequeño estudio comparativo para seleccionar el formato de audio más conveniente para el procesamiento de la información sonora. Para ello se grabó de antemano cada nota, acorde y pauta rítmica en dos formatos WAV y MP3 con un guitarrista experto. Cuando se desea comparar los resultados del estudiante, los sonidos generados, con su guitarra, se transfieren al servidor para cada formato y se comparan con lo almacenado en la base de datos. Con esto se verifican los tiempos de respuesta y el porcentaje de acierto de las muestras de sonidos. El objetivo es evaluar el desempeño para poder seleccionar el formato más adecuado para la aplicación.

Es claro que la calidad del formato juega un rol fundamental en la cantidad de espacio necesario para el almacenamiento. De hecho, el formato WAV por tener buena calidad al reproducir la melodía, ofrece una riqueza sonora superior a MP3. Sin embargo, a pesar de comprimir con más pérdida, el formato MP3 sólo deja de lado aspectos que en la mayoría de los casos no son captados por el oyente. Las estrategias de enmascaramiento por frecuencia y el enmascaramiento temporal [5] sólo obvian aquellos detalles que no son percibidos por el oído humano y en consecuencia el oyente no pierde información sonora significativa.

Con el formato WAV, utilizando diversos algoritmos sin delimitación de espectrograma, al comparar dos archivos de audio, la calidad de los resultados es excelente pero los algoritmos requerían entre 30 segundos y hasta 8 minutos para arrojar un resultado de la comparación. En cambio, con el formato MP3, aunque para la codificación y decodificación no existen soluciones implementadas nativamente para el SDK de *Android*, el tiempo de procesamiento es menor. Se utilizó la librería LAME [6], para C++ que puede ser adaptada a *Android* NDK [7] junto con el *plugin* "*MP3PLUGIN*" encargado de suministrar métodos para transformar un arreglo de bytes a un arreglo de tipo *short*, facilitando así la comparación de notas, acordes y pautas rítmicas.

Tabla I: Comparación WAV vs MP3

Formato del Archivo	Tamaño del Archivo	Porcentaje de Acierto	Tiempo de Procesamiento
WAV	353 KB	95%	Hasta dos minutos
MP3	51,5 KB	85%	Hasta 2 segundos

Una vez tomada la decisión por el formato MP3, se implementa en el dispositivo móvil, la recepción del audio entrante por el micrófono del teléfono y se codifica en una tabla gracias a la clase "*AudioRecorder*" [8] dentro de la librería "LAME" y se envía al servidor. La decodificación de sonido se hace en el servidor y no en el dispositivo móvil debido a que los módulos de comparación requieren tal capacidad de procesamiento que haría muy lento el procesamiento. Los resultados de esta evaluación están en la Tabla I.

IV. Servidor Web

Con la finalidad de tener un sistema con mayor poder de procesamiento para la gestión de usuarios, comparación de archivos de audio, almacenamiento de las estadísticas de usuarios y la base de conocimiento musical, se implementó la aplicación bajo una arquitectura cliente/servidor. Esto además hace el sistema más escalable y facilita el mantenimiento.

El servidor web, con la información recibida de la aplicación móvil, sigue el progreso del juego, envía archivos como imágenes y/o sonidos al aprendiz a guitarrista y recoge los resultados de esta interacción para mantener la dinámica del juego. Para ello ofrece los siguientes servicios:

- Ingresar a la aplicación con Facebook y vincular a sus amigos, siempre que estén registrados en la aplicación como seguidores.
- Ver los niveles del juego con sus imágenes y nombres, filtrando por el progreso del estudiante. Consultar detalles de seguidores, con su puntaje.

- Acceder a los seguidores de un estudiante diferente al que ingreso en la aplicación.
- Descargar el listado de pruebas por nivel.
- Acceder a la totalización del puntaje obtenido una vez que se termina el nivel.
- Consultar los reportes del estudiante.

V. Aplicación Móvil

Para conseguir una buena aceptación de parte de las personas interesadas en aprender teoría musical y ejercitarse como ejecutante de la guitarra eléctrica, es importante la motivación. Es por ello que la estrategia de videojuego interactivo, estructurado por niveles, es fundamental. El objetivo es que las pruebas sean lo suficientemente entretenidas para el usuario, mientras van aprendiendo conocimientos y nuevas destrezas, bajo un marco de competencia con sus amigos y usando como incentivo los puntajes de las pruebas.

A. Niveles

Para el sistema un nivel es el conjunto de 10 pruebas a ser completadas para progresar en el juego. Estos niveles están ubicados en la pantalla principal de la aplicación y pueden tener tres estados:

- **Disponible:** Son aquellos niveles, aún no completados por el estudiante, que deben concluirse para pasar al siguiente nivel.
- **Completado:** Son los niveles ya completados por el estudiante aunque están disponibles para su repetición. Estos no desbloquean nuevos niveles.
- **Bloqueados:** Son las actividades o niveles a los que el estudiante no puede acceder a menos que complete los niveles previos.

B. Pruebas

Para la enseñanza práctica y teórica de la guitarra eléctrica se deben considerar, tanto los fundamentos teóricos por los cuales se rige la música para su apreciación y escritura; como los principios básicos prácticos para la ejecución. Se deben tomar en cuenta desde las notas y acordes que emite la guitarra hasta las pautas rítmicas de una composición musical. Es por todo esto que se diseñaron varias pruebas, las cuales están en las siguientes categorías:

- **Teóricas:** ofrecen una breve descripción de un concepto musical, eventualmente con imágenes (Figura 1).

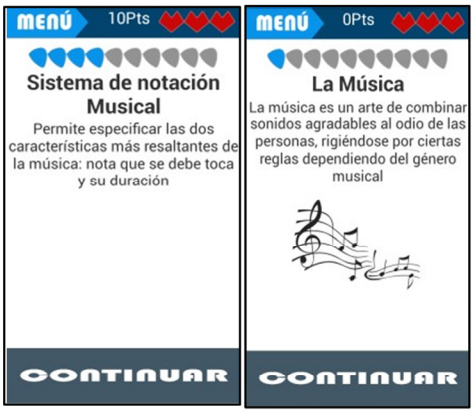


Figura 1: Información Musical Teórica

- **Pregunta V/F:** ofrecen la opción de avalar o rechazar algún concepto de la teoría musical, ya presentado en las pruebas teóricas (Figura 2).

Figura 2: Prueba de V/F o Respuesta por Selección de Imagen

- **Teórico-Prácticas para Notas:** Estas pruebas muestran al usuario una imagen con las indicaciones para poder emitir la nota o acorde con la guitarra. Junto con su descripción, el estudiante dispone de un audio del sonido asociado y la posibilidad de grabar su ejecución para la comparación con el patrón almacenado, mostrando gráficamente los sonidos en tiempo real en decibeles (Figura 3).

Figura 3: Respuesta sobre Notas con Teoría y Práctica

- **Teórico-Prácticas de Acordes Musicales:** Estas pruebas muestran al usuario una imagen que enseña el lugar donde se deben colocar los dedos para emitir el acorde. Junto a su descripción tiene un audio del cómo debería sonar el acorde musical y un grabador que registra los sonidos que ejecute el estudiante para compararlo con lo almacenado en el servidor. Se muestra gráficamente el sonido en tiempo real en decibeles (Figura 4).

Figura 4: Pregunta sobre Acordes con Teoría Práctica

- **Teórico-Prácticas de Pautas Rítmica:** Estas pruebas muestran una imagen sugiriendo la pauta rítmica, un metrónomo para que el usuario se guie al momento de la interpretación, una muestra de cómo debería sonar el ritmo y un grabador que registra los sonidos que toque el estudiante con su guitarra. Esta muestra se usa para compararla con lo almacenado en el servidor, mostrando los sonidos en tiempo real en decibeles (Figura 5).

Figura 5: Pregunta Teórico-Práctico de Pautas Rítmicas con Metrónomo

- **Prácticas de Notas y Acordes:** Estas pruebas indican como colocar los dedos en los trastes de la guitarra para poder ejecutar el sonido y un grabador que se encargue de registrar el resultado. Además de mostrarse gráficamente en tiempo real, se comparará con lo almacenado en el servidor (Figura 6).

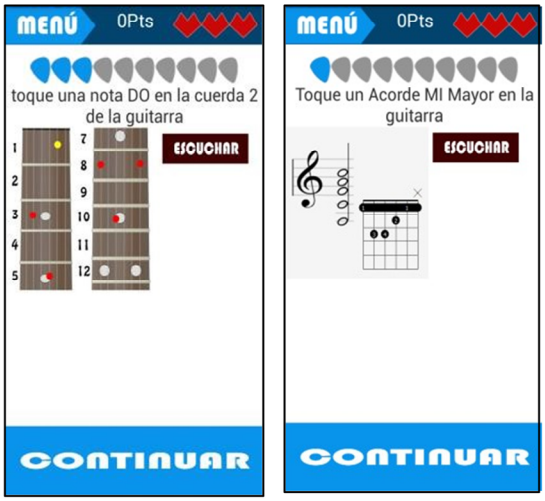


Figura 6: Pregunta Prácticas de Notas y Acordes

C. Notificaciones

Estas motivan al estudiante según la constancia con la que ingresa a los niveles de la aplicación. Se generan, después del ingreso del ejecutante, gracias a un calendario interno que se actualiza en función a la regularidad del aprendiz. También se le pueden enviar mensajes de alerta o felicitaciones a la bandeja de notificaciones de *Android* (Figura 7).

D. Integración con Facebook

Para lograr que la aplicación tenga un buen grado de aceptación, amigabilidad y simplicidad se decidió realizar el registro de jugadores mediante *Facebook*, ya que esta red social es muy utilizada a nivel mundial y brinda servicios que permiten tener un control de acceso, información del usuario y listado de amigos.

Para esta integración, *Facebook* provee un SDK para *Android* que proporcionar una documentación sencilla para su implementación y uso. El *framework Grails* posee un *plugin* para hacer uso de *Facebook* vía Web, en donde el sistema hace la vinculación entre los amigos que posee el usuario en *Facebook* y los convierte en seguidores del usuario-estudiante de la aplicación.

Figura 7: Notificación sobre la Interfaz de Android

E. Sincronización de Archivos

Dado que el tiempo de procesamiento puede ser considerable y para agilizar la interacción entre los estudiantes, resulta conveniente que los archivos de los servicios que se utilizan regularmente solo sean descargados la primera vez que se les necesita. Para ello esta información se almacena localmente en una base de datos, para el dispositivo móvil con *Android,* llamada *Sqlite*.

El servidor envía a la aplicación del estudiante un listado de los identificadores de los archivos necesarios para la interfaz gráfica. Esta lista es comparada contra los archivos de base de datos local y, en caso de no tenerlos, se descargan desde el servidor.

F. Amigos

Este módulo tiene la capacidad de listar los amigos asociados que tiene el guitarrista aprendiz, junto con sus puntajes hasta la fecha de la consulta. Además se pueden ver los amigos de los amigos con el objeto de que pueda invitarlos y así generar un ambiente de mayor competitividad (Figura 8).

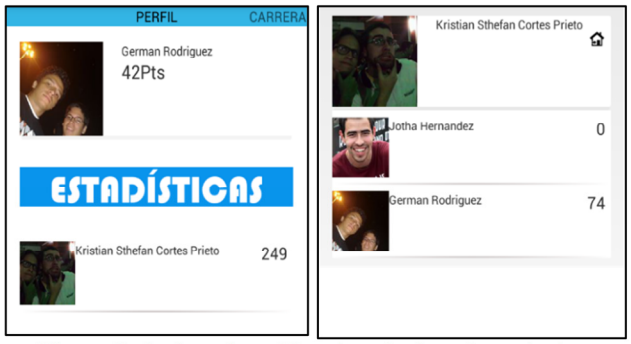


Figura 8: Amigos de un Usuario y Amigos de un Amigo

G. Reportes

Este módulo emite varios tipos de reportes, específicamente como serían las notas, acordes y pautas rítmicas tocadas por el aprendiz que consulta y por todos los demás estudiante. Algunas reportes son: notas o acordes más fáciles y difíciles de tocar, promedio de puntos ganados en teoría y en práctica, (Figura 9).

Figura 9: Pantallas de Reportes

VI. Análisis y Comparación de Notas, Acordes y Pautas Rítmica

Con la finalidad de implementar un método eficiente para el procesamiento de las notas, acordes y pautas rítmicas, se realizó una investigación de los algoritmos de comparación de archivos de audio. Se seleccionó el método de reconocimiento de vocales, donde se parte de la identificación de formantes de un sonido en el dominio de la frecuencia, para diferenciar una vocal de otra. Según la bibliografía es usado para reconocimiento de sonidos en general y efectivamente resultó ser adecuado aunque no está explícitamente diseñado para reconocimientos de notas, acordes y pautas rítmicas.

Por otro lado, para encontrar las características asociadas a las ondas de los audios, se utilizó la Transformación Rápida de Fourier, la cual permite cambiar el dominio de tiempo de un audio a un dominio de frecuencia, mostrando las frecuencias de los formantes más significativas de un sonido.

A fin de cuentas, las características más significativas para comparar e identificar sonidos son principalmente la frecuencia y el rango de tiempo donde ocurre dicha frecuencia. Para las notas es sencillo pues basta con identificar el formante más cercano a dicha frecuencia. En el caso de los acordes se utiliza el mismo método de comparación pero ubicando todas las frecuencias de las notas que debe poseer ese acorde. Para las pautas rítmicas se debe añadir, además de las frecuencias y rangos, los intervalos de tiempos entre las notas para que el algoritmo pueda verificar si las frecuencias se encuentran en los instantes de tiempo esperados.

- **Análisis y Comparación de Notas:** Para identificar una nota musical se parte de la frecuencia almacenada previamente en la base de datos. Con el rango en el cual se encuentran dichas notas, el espectrograma de tiempo-decibeles (Figura 10) y dada la decodificación del archivo de audio enviado por el estudiante, se comparan los archivos para determinar el sonido generado por el guitarrista aprendiz.

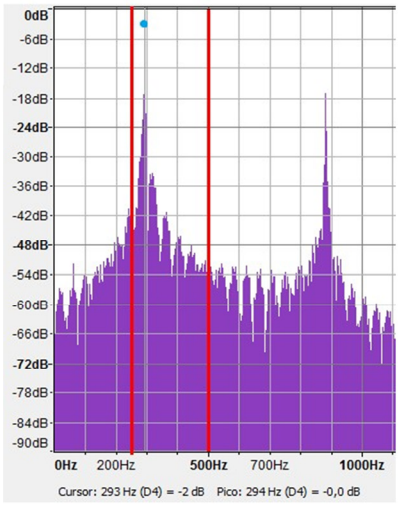


Figura 10: Verificación de una Nota con el Espectrograma de Frecuencia

Más detalladamente, el procedimiento consiste en tomar los valores en el rango donde debería estar la frecuencia de la nota buscada y se identifica cual es el primer formante o el formante con el valor más alto de decibelios. Con el programa *Audacity* se identifica el formante más alto, en este caso con el valor de 294 dB,

por lo que según los conocimientos teóricos, en este caso, el valor corresponde a la nota Re.

- **Análisis y Comparación de Acordes:** Sabiendo cada nota que compone el acorde y los rangos en que varían estas frecuencias, es posible lograr una comparación de las muestras obtenidas del estudiante y compararlas con lo almacenado en el servidor a través del método de comparación de acordes (Figura 11).

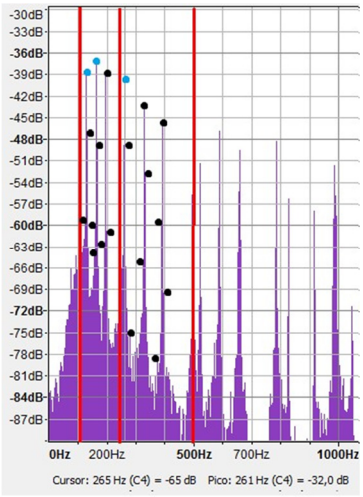


Figura 11: Verificación de un Acorde con el Espectrograma de Frecuencia

A partir del espectro de la muestra de audio se crean divisiones según los rangos de frecuencia en los que se encuentran las notas musicales que posee el acorde. En este caso, se trata del acorde Do Mayor, el cual posee las notas Do y La además del Do en otro rango de frecuencia. Una vez identificados los rangos se realiza el cálculo de los 10 formantes más altos de cada rango, los cuales son comparados contra las frecuencias buscadas para saber cuál es el formante más cercano a las frecuencias buscadas.

- **Análisis y Comparación de Pautas Rítmica:** Una vez identificadas las frecuencias de las notas musicales que posee la interpretación, los rangos de las frecuencias de estas notas musicales, y los tiempos entre cada una de ellas y, teniendo decodificada la muestra de audio del estudiante en un espectrograma Tiempo-Decibeles, se procede a realizar un análisis y comparación con los datos que se encuentran en la base de datos del servidor (Figura 12).

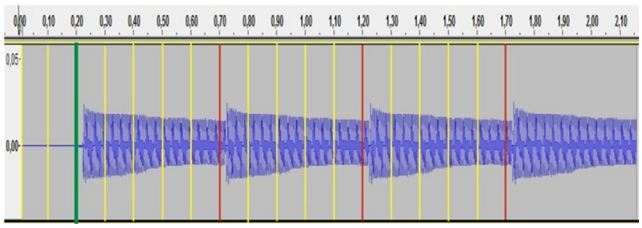


Figura 12: Verificación de una Rítmica

Con el algoritmo de reconocimiento de notas musicales, se van procesando los fragmentos de la muestra de audio, nota por nota dentro de la pauta rítmica. Una vez seleccionada las notas se acopla su aparición usando el espectrograma rítmico y con un rango de error fijado como aceptable, se identifica la periodicidad y por ende la pauta rítmica.

En general el cálculo de la similitud entre dos acordes o dos pautas rítmicas tiende a ser más inexacto que el de las notas. Aunque se pueden encontrar algunas imperfecciones en la muestra de audio almacenada, la principal fuente de error para la identificación bajo el ruido ambiental. Este introduce armónicas que alteran la codificación del sonido. Para mitigar el grado de error, se decidió analizar los 10 formantes más significativos con el objetivo de aproximar la frecuencia más cercana a la nota buscada.

VII. Módulo de Afinación

Para el módulo de afinaciones es preferible no acceder al servidor, ya que esto conllevaría a un amplio gasto de recursos y aumentaría el tiempo de espera del estudiante. Por esto se implementó dicho módulo en la aplicación móvil, donde mediante una interfaz de fácil uso (Figura 13), el usuario deberá selecciona la cuerda que se quiere afinar. La aplicación va dando una referencia de la acción que debe tomar el estudiante, sobre la guitarra, para afinarla.

Se reutilizaron algunos códigos de identificación de sonidos del servidor y se cargaron localmente en el dispositivo móvil. La intención era tener una interfaz rápida, sencilla e intuitiva, de fácil uso para el aprendiz, sin conocimiento previo o adiestramiento en el sistema.

Para conseguir un resultado similar al de los métodos implementados en el servidor se reutilizó el método de codificación de audio MP3 a un arreglo de *shorts* utilizando la librería "LAME", aplicando a este el análisis y comparación de frecuencias. Esto, por supuesto, sabiendo de antemano la nota de cada cuerda, sin aplicar los dedos en ningún traste.

El método de análisis y comparación de frecuencias para la afinación utiliza la estrategia *FFT* sobre las muestras en tiempo real desde el micrófono, que son decodificadas por la librería "LAME" para su inmediata comparación con lo almacenado en la base de datos local.

Dependiendo de los resultados de la comparación se pueden determinar cinco situaciones: cuerda afinada, cuerda floja, cuerda muy tensa, silencio de la guitarra o existencia de mucho ruido. Una vez detecta la situación que acontece, se le notifica al practicante, en tiempo real, que acciones debe tomar para afinar la cuerda.

Para lograr que la aplicación sea intuitiva para el usuario, se diseñaron seis cuerdas diferenciadas con colores. Además la condición de la cuerda, durante la afinación, se indica con colores. Mientras más roja una cuerda, su tono es más agudo de lo debido. Mientras más azul por el contrario el tono es más grave de lo necesario.

Las cuerdas están ubicadas en la misma posición que en la guitarra configurada para diestros. En la parte inferior de la ventana se encuentra un recuadro que señala gráficamente los cuatro estados del sonido percibido tal como lo percibe la aplicación y la acción que se espera el estudiante realice para afinar esa cuerda.

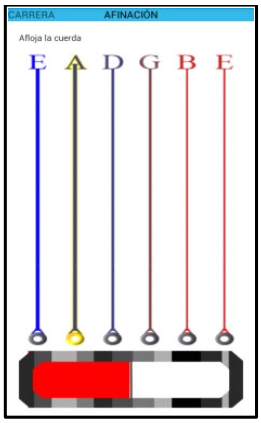


Figura 13: Módulo de Afinación

VIII. MEDICIÓN DEL APRENDIZAJE

Para conocer cuán útil puede resultar el sistema de enseñanza propuesto, se midió la diferencia de aprendizaje para dos universos de personas: un grupo que hiciera uso del sistema frecuentemente contra otro grupo que recibiera clases presenciales con un profesor que impartiera los mismos conceptos teóricos y prácticos.

Después de buscar entre los diferentes métodos de medición y recolección de datos, se seleccionó la metodología Pre-Test y Post-Test para analizar el aprendizaje entre las dos muestras de personas (Figura 14). Se constató, procesando probabilísticamente la información, cual de las dos metodologías logra mayor efectividad en la enseñanza teórico-práctica de la música usando guitarra eléctrica. Está metodología exige la realización de cuatro actividades, las cuales son:

- **Instrumentos de Recolección de Datos:** Desarrollar el instrumento Pre-Test y Post-Test, gracias a encuestas, para verificar el conocimiento de las personas en un ámbito en específico. En este caso el conocimiento teórico-práctico de la música y la guitarra eléctrica, antes y después de aplicarse los métodos de enseñanza.

  La metodología Pre-Test y Post-Test condiciona que las pruebas a ser desarrolladas deben tener un contenido equivalente o con el mismo comportamiento psicométrico. Por ello se montó en la aplicación propuesta el contenido didáctico que es presentado en los cursos presenciales, con pruebas teóricas y prácticas que manejan la misma terminología de las escuelas musicales.

  Cada una de las pruebas se dividió en dos partes, en la primera parte se hacían 15 preguntas de la teoría musical, mientras que en la segunda parte se realizaron 5 ejercicios prácticos con una guitarra para un total de 20 puntos. Tanto las preguntas teóricas como prácticas fueron evaluadas simplemente como correctas o incorrectas.

- **Validación de Instrumentos:** Las pruebas deben pasar por un proceso de verificación, donde se involucran especialistas tanto en la rama de la psicología como expertos en música y guitarra. Estas personas revisan el contenido y la estructura de las pruebas para validar que los resultados sean significativos y que la comparación y la medición sean la adecuada.

  Las personas que revisaron y argumentaron que la prueba seguían los lineamientos de una encuesta correcta se encuentran en la Tabla II.

Tabla II: Expertos que Validaron los Instrumentos

Psicólogo		Músico	
Nombre	**Cargo**	**Nombre**	**Cargo**
Kira Cook Aguilar (4 años de experiencia)	Psicóloga (CADH UCAB)	Raúl López (15 años de experiencia)	Profesor de Música (Academia Musical Luthier)
Ana Gabriela Pérez (23 años de experiencia)	Directora de la Escuela de Psicología (UCAB)	Javier Pérez (5 años de experiencia)	Profesor de Música (Academia Musical Luthier)
Juan Carlos Carreño (14 años de experiencia)	Profesor de Psicología (UCAB)	Víctor Solar (10 años de experiencia)	Profesor de Música (Complejo Cultural de Los Salias)

- **Definición de Criterios para la Selección de Poblaciones de Muestra:** Para la comparación estadística entre el aprendizaje mediante el método tradicional y la aplicación propuesta, se establecieron dos grupos de 15 personas cada uno, mayores de 15 años, que supiesen leer y escribir, sin diferenciar por género y que no tuvieran significativos conocimientos previos teórico-prácticos de música ni guitarra.

- **Aplicación del Instrumento:** Para la muestra que recibe clases con el método tradicional de enseñanza con un profesor (de ahora en adelante conocido como "Muestra Profesor") se realizó una búsqueda en región capital (Caracas) con el fin de ubicar las escuelas musicales en las que los estudiantes pudieran brindar apoyo para la realización de las encuestas. Se logró obtener la ayuda de los estudiantes de 3 escuelas: *Academia Musical Luthier*, *Escuela Lino Gallardo* y *Complejo Cultural y Deportivo de Los Salias*.

  Por otro lado, las personas que conformaron el grupo que haría uso del sistema para aprender a tocar guitarra eléctrica (de ahora en adelante conocido como "Muestra Aplicación") fueron voluntarios en la Universidad Católica Andrés Bello, personas cercana a estas últimas y algunos familiares que no tuvieran experiencia o relación alguna con la teoría o práctica de tocar un instrumento.

  La aplicación de estas evaluaciones evitaron sesgos relacionados con la intervención del profesor durante la realización de la prueba pre-test y la prueba post-test en cada una de las escuelas. Justamente al momento de la aplicación de las mismas, se pidió a los profesores que no ofrecieran ninguna ayuda a sus estudiantes.

- **Análisis de Resultados:** Con los datos recolectados por las encuestas, se procesó y analizó la información como se describe en [9] comenzando con el examen Pre-Test. Se procedió a aplicar estadística descriptiva y se analizaron con medidas de tendencia central (Moda y Media) y de variabilidad (Desviación estándar y Varianza), los resultados se presentan en la Tabla III.

Tabla III: Resultados de Tendencia Central y Variabilidad

Pre-Test	Muestra Profesor	Muestra Aplicación
Moda	1	1
Promedio	1.66	1.4
Varianza	2.23	0.97
Desviación Estándar	1.49	0.98

A partir de los resultados de tendencia central, se puede concluir que en general, ninguno de los grupos tenían conocimientos significativos con respecto a los temas mencionados en el Pre-Test ya que la moda es de 1 (Puntuación más repetida) y las medias tienen una calificación de 1.66 y 1.4, lo cual es un resultado muy bajo con respecto a la nota máxima (20).

Con respecto a las conclusiones arrojadas por los resultados de variabilidad obtenidos del cálculo de las varianzas (2.23 y 0.97) y las desviaciones estándar (1.49 y 0.98), se puede concluir que no existe para este caso una dispersión significativa de los resultados con respecto al promedio de cada una (1.66 y 1.4).

Finalmente se realizó una t-student para grupos independientes en la fase pre-test de la prueba donde se obtuvo una t de 0.576 con p establecida de 0.05 y una t obtenida de 0.570 (grados de libertad= 28), por lo que pueden indicarse que no existen diferencias significativas estadísticamente entre ambos grupos, es decir, se puede establecer que los grupos poseían notas similares antes de aplicar la intervención.

En una segunda parte del análisis se utilizaron las medias de los Pre-Test y los Post-Test para determinar el progreso de cada una de las muestras en el ámbito musical. El promedio de la muestra profesor obtuvo una mejoría de 9 puntos en el Post-Test, con respecto al promedio del Pre-Test mientras que el promedio de la muestra aplicación obtuvo una mejora de 15.53 puntos en el examen de Post-Test.

La segunda hipótesis parte de la verificación mediante una t-student para grupos correlacionados del progreso de las muestras del Pre-Test al Post-Test, En la muestra profesor se obtuvo una t= 14, con p = 0.00 (grados de libertad= 14, p criterio= 0.05) es decir, si existen diferencias significativas entre las notas de los alumnos en la prueba, antes de realizar la intervención con el profesor y luego de realizar la intervención.

A su vez, se aplicó en paralelo una t-student de grupos correlacionados para la muestra aplicación obteniéndose una t= 26.95 y una p = 0.00, (grados de libertad= 14, p criterio= 0.05), es decir, que si existen diferencias estadísticamente significativas entre las notas obtenidas antes y después de utilizar la aplicación o seguir la instrucción presencial.

Por último se realizó una t-student para grupos independientes entre las medidas de ambas muestras luego de aplicar la intervención, obteniéndose una tstudent= 6.902 y una p = 0.00, por lo que puede indicarse que si existen diferencias significativas entre ambos grupos en las notas obtenidas luego de utilizar ambos métodos, siendo mayores las notas obtenidas para la muestra aplicación (media= 16.93) en comparación a las obtenidas para la muestra profesor (media= 10.06). Esto permite comprobar una mayor efectividad de la aplicación por sobre la instrucción del profesor tomando como patrón las notas de los estudiantes en las pruebas.

Adicionalmente, se puede concluir que las oportunidades de estudio que permite una aplicación móvil sobre un profesor fueron determinantes para las diferencias entre los resultados de las dos muestras. Específicamente hay que resaltar: disponibilidad permanente que permite a la muestra aplicación practicar cuando lo deseen mientras que la muestra profesor estaba limitada a un horario que no incluía todos los días de la semana, Modulo de notificaciones para incentivar al usuario y ambiente competitivo gracias a la asociación de los amigos del estudiante por medio de Facebook.

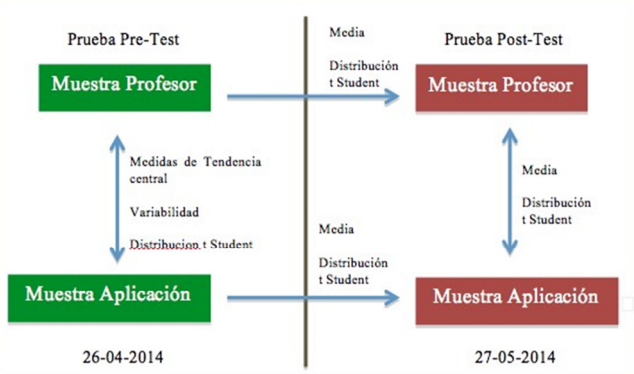


Figura 14: Descripción Gráfica de la Metodología Pre-Test Post-Test

IX. CONCLUSIONES

El sistema propuesto es sin duda una alternativa viable para el aprendizaje amigable de la guitarra eléctrica, tanto desde el punto de vista práctico como teórico.

En base a los resultados obtenidos se puede concluir lo siguiente:

- La precisión de las muestras de sonidos es claramente dependiente de la calidad del micrófono del dispositivo móvil. Por lo general estos dispositivos no tiene micrófonos de muy buena calidad.
- Dar el enfoque de videojuego, vinculando a los estudiantes con sus contactos en Facebook y contar con un módulo de reportes para las estadísticas de todos los jugadores, fue una decisión favorable para la aceptación de los aprendices con la aplicación. Esto genera una competitividad sana entre practicantes amigos y desconocidos que están registrados en la aplicación.
- Algunas de las pruebas de los algoritmos de reconocimiento de notas, acordes y pautas rítmicas se realizaron en base a sonidos artificiales (generados por computadora), por lo que existe la probabilidad de que cada uno de estos algoritmos tenga falsos positivos
- El ruido externo es un factor determinante en la precisión del sistema. El ambiente debía ser estrictamente controlado para lograr resultados satisfactorios.
- Gracias a las posibilidades graficas que brindan los dispositivos móviles actuales, los usuarios indicaron que la afinación de la guitarra eléctrica con la aplicación les

resultaba más intuitiva que con el afinador electrónico común.

- Aunque no se probó exhaustivamente, el reconocimiento de notas, acordes y pautas rítmicas también funciona para guitarras acústicas.
- Los análisis estadísticos efectuados arrojaron como resultado que las personas que utilizaron la aplicación móvil aprendieron más rápidamente los conocimientos teórico-prácticos que las personas que utilizaron el método tradicional. Esto demuestra que la inclusión de sistemas informáticos en el ámbito de aprendizaje musical es una solución favorable para las personas que se les dificulta tomar clases presenciales de música.

REFERENCIAS

[1] F. Liarokapis, *Augmented Reality Scenarios for Guitar Learning*, in proceedings of the Third International Conference on Eurographics - Theory and Practice of Computer Graphics, pp. 163–170, Canterbury, United Kingdon, June 2005.

[2] Y. Motokawa and H. Saito. *Support System for Guitar Playing using Augmented Reality Display*, IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2006), Santa Barbara, California, USA, October 2006.

[3] E. Lenz*, System and Method for Learning Music in a Computer Game*, US Patent US7893337, February 2011.

[4] J.C. Epstein*, Interactive Guitar Game Designed for Learning to Play the Guitar*, US Patent US20100137049 A1, June 2010.

[5] D. Barzola, R. Cabrera, et al, *Comparación entre Compresión de Audio en Diferentes Formatos de Imágenes Equivalentes y el Formato MP3*, Reporte Interno de la Escuela Superior Politécnica del Litoral, Guayaquil, Ecuador, 2008.

[6] M. Cheng, *Proyecto LAME*, http://lame.sourceforge.net, Octubre 2011.

[7] Google, *Android NDK,* https://developer.android.com/tools/sdk/ndk/index.html.

[8] Google, Android Developers, http://developer.android.com/reference/android/media/AudioRecord.html.

[9] F. Kerlinger, *Investigación del Comportamiento*, McGraw Hill, México, 2002.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 63-71, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Algoritmo para Contar Nodos en Redes Inalámbricas con Mensajes Retrasados

Manuel Contreras [1], Eric Gamess [2]
mcontre@ula.ve, eric.gamess@ciens.ucv.ve

[1] Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela
[2] Laboratorio de Comunicación y Redes, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: El conteo de objetos (personas, animales, dispositivos, vehículos, etc.) ha interesado la comunidad científica desde hace tiempos inmemoriales, ya que puede ser utilizado como una herramienta básica en muchas aplicaciones. Hasta la fecha, la mayoría de los trabajos hechos en el área están basados en cámaras digitales, cámaras de video, torniquetes, brazos mecánicos, tubos neumáticos, luz infrarroja y otros tipos de sensores. Sin embargo, pocos trabajos han sido desarrollados usando las comunicaciones inalámbricas para contar nodos. En este artículo, introducimos una versión mejorada de un algoritmo de conteo de nodos que habíamos propuesto para contar dispositivos equipados de una interfaz de red inalámbrica, donde se consideran además en el conteo total de nodos, aquéllos mensajes de conteo retrasados que pudieran ser enviados por los nodos en un tiempo posterior al inicialmente determinado. Para validar nuestra propuesta, simulamos el algoritmo con OMNeT++/INET y comparamos los resultados con la versión preliminar del algoritmo. Los resultados obtenidos en el conteo son alentadores, ya que la nueva versión del algoritmo mejora en forma sustancial el algoritmo preliminar. Además, las simulaciones muestran que el algoritmo no solamente funciona para nodos estacionarios, sino que también hace un conteo eficiente cuando los nodos son móviles con velocidades moderadas.

Palabras Clave: Redes Inalámbricas; Conteo; Algoritmos; Simulación; OMNeT++; INET.

Abstract: Counting objects (people, animals, devices, vehicles, etc.) has interested the research community since time immemorial, because it can be used as a basic tool in many applications. Up to now, most of the work done in the area is based on digital cameras, video cameras, turnstiles, mechanical arms, pneumatic tubes, infrared beams, and other types of sensors. However, just a few studies have been developed using wireless communications to count nodes. In this paper, we introduce an improved version of an algorithm for node counting that we previously proposed to count devices equipped with a wireless network interface, where we also consider in the total counting of nodes, the delayed counting messages that can be sent by nodes at a later time than the initially determined. To validate our proposal, we simulate the algorithm with OMNeT++/INET and we compare the results with the preliminary version of the algorithm. The results obtained in the counting are encouraging, since the new version of the algorithm substantially improves the preliminary algorithm. Furthermore, the simulations show that the algorithm not only works for stationary nodes, but also makes an efficient counting when nodes are moving with moderate speeds.

Keywords: Wireless Networks; Counting; Algorithms; Simulation; OMNeT++; INET.

I. Introducción

Contar objetos tales como personas, animales, dispositivos, vehículos, entre otros, ha sido de gran interés en diversos ámbitos o campos de estudio. Por ejemplo, las personas pueden ser contadas en eventos políticos, deportivos, musicales o sociales, por muchas razones que incluyen estadísticas de afluencia, capacidad, optimización de flujo, seguridad, entre otros. En el transporte público, la información del número de personas en las unidades o en los paraderos podría ser usada para mejorar el servicio entregado al aumentar o disminuir la frecuencia de las unidades según el volumen de pasajeros. En un estacionamiento, el número de vehículos estacionados es útil para determinar el número de puestos libres. Para la gestión eficaz de la fauna, la conservación y la ecología, es muy relevante obtener una estimación confiable del tamaño o densidad de las poblaciones de las especies de animales, en particular cuando están en peligro de extinción. Otro ejemplo es el conocimiento del número de personas que hay en lugares con abundante afluencia para así determinar los mejores patrones de seguridad en caso de alguna contingencia. En la actualidad, se intenta solventar todas estas tareas mediante operadores humanos o sistemas automáticos, principalmente basados en cámaras digitales, cámaras de video, torniquetes, brazos mecánicos, tubos neumáticos, luz infrarroja, sensores magnéticos, radares, o cables piezoeléctricos [1][2][3].

El método más tradicional para estimar el número de objetos es el conteo manual. En este sentido, observadores entrenados recolectan los datos necesarios que por medio de contadores automáticos [4], no se pueden obtener eficientemente. Según [5], un observador puede contar aproximadamente de 1000 a 2000 personas en una hora sin ayuda tecnológica. Ahora, con un contador (un dispositivo pequeño que cuenta pulsos), dicha persona puede duplicar la cantidad de personas contadas. Similarmente, otros objetos que pueden ser contados por un observador entrenado, son las bicicletas y los carros. Sin embargo, el método de conteo manual presenta varios inconvenientes como la necesidad de contratar personal para desempeñar este tipo de trabajo y la posibilidad que se cometan errores importantes en situaciones con alta densidad de objetos.

También, se puede estimar el número de objetos mediante los sistemas que usan contadores mecánicos. Los más representativos son los torniquetes que se emplean para contar personas en un sistema de control de acceso, una a la vez, presentando un alto grado de fiabilidad, un alto costo y un bajo grado de automatización del procesamiento de la información. Estos sistemas eliminan el costo de personal, pero reducen significativamente los espacios dedicados a la entrada/salida de personas, resultando en muchos casos en demoras e incomodidades en el tránsito de las mismas, debido a su inherente aparataje.

Otro caso es el uso de un contador térmico, el cual tiene un sensor que identifica el gradiente entre la temperatura ambiental y la temperatura corporal. Este tipo de contador es ineficiente cuando la temperatura ambiental es semejante a la corporal [6].

Otra técnica, basada en la tecnología infrarroja, usa un simple haz de luz que normalmente está conectado a una pantalla LCD (pantalla de cristal líquido) ubicada en el lado de la puerta de las unidades de transporte (por ejemplo, en trenes, autobuses, ferries). Cuando un pasajero entra en la unidad de transporte, se interrumpe o se corta el haz de luz, siendo detectado por un micro controlador que reacciona incrementando un contador. Sin embargo, esta técnica sufre de un problema denominado oclusión. La oclusión es generada por el amontonamiento de personas, es decir, al cruzar un grupo de personas al mismo tiempo, el contador sólo detecta una [6] ya que no hay discontinuidad en la intercepción del rayo.

Los sensores magnéticos o bucles inductivos [4] son usados en los sistemas de transporte donde son enterrados en el asfalto o puestos en la superficie de la carretera, de manera que cuando un vehículo pase sobre el mismo, el campo magnético del motor del vehículo modifique el campo registrado por el sensor. Esta tecnología permite además de detectar y contar vehículos, obtener su velocidad y clasificarlos según su tipo.

No obstante, durante los últimos años la comunicación de datos ha ganado un mayor interés e importancia, razón por la cual hoy en día, es común que los objetos (personas, animales, dispositivos, vehículos, etc) estén equipados de una interfaz inalámbrica para fines de comunicación, de modo que los algoritmos pueden aprovechar esta tecnología para contar objetos. Como principal objetivo de este trabajo, se propone el desarrollo de un algoritmo para contar nodos usando tecnologías inalámbricas, partiendo de las ideas planteadas en el algoritmo propuesto en [7], dónde se toman en cuenta en el conteo aquéllos mensajes retrasados que pudieran ser enviados por los nodos en un tiempo posterior al inicialmente establecido. Las simulaciones que corrimos muestran que esta nueva versión del algoritmo mejora sustancialmente el cálculo del número de nodos en comparación al algoritmo anterior [7], tanto en escenarios estacionarios, como para nodos móviles con velocidad moderada.

El resto de este artículo está organizado como sigue: En la Sección II, se hace una revisión de los trabajos relacionados. En la Sección III, se describe el algoritmo para contar nodos usando tecnologías inalámbricas, tomando en cuenta los mensajes retrasados. La Sección IV introduce brevemente las herramientas de simulación que se utilizaron para validar el algoritmo propuesto. En la Sección V, se presenta un análisis comparativo de los resultados de nuestras simulaciones. Finalmente, en la Sección VI se discuten las conclusiones del artículo y los trabajos futuros.

II. Trabajos Relacionados

La detección y el conteo de objetos representan una herramienta que tiene numerosas aplicaciones, y debido a su gran utilidad se ha venido efectuando en diversas formas o disciplinas con el uso de varias tecnologías que hacen que en la realidad sea aplicable a muchas situaciones. Actualmente, se pueden encontrar diferentes alternativas que permiten llevar a cabo la tarea de detectar y contar nodos con mayor o menor precisión, basadas primordialmente en métodos, técnicas o tecnologías convencionales "in-situ", tales como cámaras digitales, cámaras de video, torniquetes, brazos mecánicos, tubos neumáticos, luz infrarroja, sensores magnéticos, radares o cables piezoeléctricos, que son utilizados por ejemplo para: detectar y contar personas en los sistemas de vigilancia de los espacios públicos como centros comerciales, aeropuertos, eventos públicos; estimar el tamaño de una población de organismos (tal es el caso de las especies de animales); determinar el número de puestos libres en estacionamientos; contar pasajeros en el transporte público; contar vehículos en tiempo real en las carreteras, intersecciones, semáforos, entre otros. Por ejemplo, en los trabajos [8][9][10][11][12][13], los autores plantean diferentes métodos y técnicas para contar personas en los sistemas de vigilancia o contar pasajeros en el transporte público, utilizando registros de cámaras digitales o de video. En [14], los autores desarrollaron un algoritmo de conteo basado en el uso de una línea láser infrarrojo como referencia de un área de detección de personas. Los autores de [15][16] usaron el procesamiento de imágenes obtenidas de cámaras de video para contar vehículos en tiempo real. Ki [17] diseñó un sistema de conteo que hace un seguimiento de personas que transitan por una zona de detección con una red de sensores infrarrojos. Gros-Desormeaux et al. [18] implementaron dos algoritmos de conteo acústico para contar aves usando sensores equipados con micrófonos. Litzenberger et al. [19] presentan un sistema integrado que comprende un sensor óptico sensible al movimiento y un DSP de bajo costo para detectar y contar vehículos. En [20], se utiliza un dispositivo láser en carreteras con el objetivo de realizar un conteo y una clasificación de los vehículos en función de la información obtenida con el láser. Los autores de [21][22] utilizaron sensores láser para detectar y contar a las personas en una estación de autobuses. Los trabajos [23][24] se basan sobre imágenes aéreas para estimar poblaciones de animales. Kissell

y Nimmo [25] utilizan imágenes de cámaras infrarrojas para estimar la población de ciervos.

En el contexto de contar objetos utilizando las redes inalámbricas, son muy pocos los trabajos que se han desarrollado. Por ejemplo, Gamess y Mahgoub [26] presentan un enfoque basado en VANET, para determinar: (1) la posición del último vehículo y (2) el número de vehículos, en una línea de vehículos parados en un semáforo.

Tal como podemos apreciar, la mayor parte del trabajo realizado en este campo se basa en diseños con tecnologías "in-situ".

III. Algoritmo para Contar Nodos en Redes Inalámbricas con Mensajes Retrasados

En esta sección, se describe el algoritmo para contar nodos usando tecnologías inalámbricas, tomando en cuenta los mensajes retrasados que pudieran ser enviados por los nodos.

A. Consideraciones Básicas para Contar Nodos

El presente algoritmo representa una modificación de las ideas planteadas en la propuesta desarrollada en [7], considerándose además en el conteo total de nodos, aquéllos mensajes de conteo retrasados que pudieran ser enviados por los nodos en un tiempo posterior al especificado. El enfoque básico de dicho algoritmo es:

1. Propagar un mensaje "broadcast" (llamado COUNT_REQUEST) desde el nodo origen hacia los nodos que están lejos del mismo.
2. Propagar mensajes "unicast" (COUNT_REPLY) desde los nodos que se encuentran lejos del origen hacia el mismo, con el número total de nodos contados hasta el momento (llamado *Total* en el algoritmo).

La Figura 1 muestra un diagrama de tiempo relacionado con la propagación de los mensajes COUNT_REQUEST (enviados como broadcast) y los mensajes COUNT_REPLY (enviados como unicast), donde:

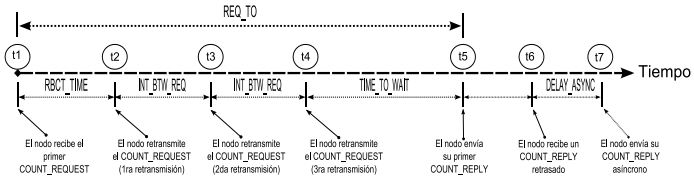


Figura 1: Diagrama de Tiempo

t1 = Es el tiempo en el cual el nodo recibe el primer COUNT_REQUEST.
t2 = Es el tiempo en el cual el nodo envía el primer COUNT_REQUEST.
t3 = Es el tiempo en el cual el nodo envía el segundo COUNT_REQUEST.
t4 = Es el tiempo en el cual el nodo envía el tercer COUNT_REQUEST.
t5 = Es el tiempo en el cual el nodo envía su COUNT_REPLY síncrono.
t6 = Es el tiempo en el cual el nodo recibe un posible mensaje COUNT_REPLY retrasado desde otro nodo.
t7 = Es el tiempo en el cual el nodo envía un posible COUNT_REPLY asíncrono.

La Figura 1 también muestra que los nodos retransmitirán tres mensajes COUNT_REQUEST en fila y un mensaje COUNT_REPLY síncrono después de un tiempo específico. Es necesario enviar varias veces (3 veces en nuestro algoritmo) el mensaje COUNT_REQUEST ya que es un mensaje de broadcast que puede entrar en colisión con otros mensajes sin que esta sea detectada. Además, es posible que el nodo reciba un mensaje COUNT_REPLY retrasado. En este caso, deberá volver a hacer su conteo antes de propagarlo hacia el origen con un COUNT_REPLY asíncrono. Los siguientes parámetros permiten determinar los tiempos de envío de los mensajes COUNT_REQUEST y COUNT_REPLY (véase Figura 1):

- RBCT_TIME (Rebroadcast Time): Es el tiempo entre la recepción del primer COUNT_REQUEST, y la primera retransmisión del COUNT_REQUEST por el nodo.
- INT_BTW_REQ (Interval Between Request): Es el intervalo de tiempo entre el envío de mensajes COUNT_REQUEST. Es decir, representa el tiempo entre dos mensajes COUNT_REQUEST consecutivos enviados por un nodo.
- REQ_TO (Request Timeout): Es el tiempo transcurrido entre la recepción del primer COUNT_REQUEST y el momento cuando el nodo tiene que enviar su mensaje COUNT_REPLY síncrono al nodo *nodeToGoBack*.
- TIME_TO_WAIT: Es el tiempo que un nodo espera después de la retransmisión del último mensaje COUNT_REQUEST y el envío del mensaje COUNT_REPLY al nodo *nodeToGoBack*. Este tiempo debe ser lo suficientemente grande como para permitir la propagación de los mensajes COUNT_REQUEST desde el nodo actual hacia los nodos que están lejos del origen, y la propagación de los mensajes COUNT_REPLY desde los nodos que se encuentran lejos del origen hacia el nodo actual. Este tiempo no es un valor constante sino que será computado por cada nodo de acuerdo a *HopLimit* y que tan lejos está del origen (llamado *HopAway* en el algoritmo).
- DELAY_ASYNC: Es el tiempo que un nodo espera después de la recepción de posibles mensajes COUNT_REPLY retrasados, para enviar un COUNT_REPLY asíncrono a su respectivo nodo *nodeToGoBack*.

B. Mensajes COUNT_REQUEST y COUNT_REPLY

Los mensajes COUNT_REQUEST y COUNT_REPLY comparten la misma Unidad de Protocolo de Datos (PDU) que se compone de seis campos (ver Figura 2).

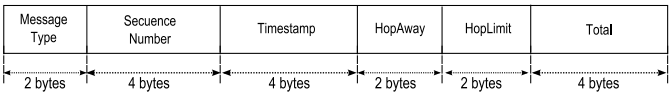


Figura 2: Mensajes COUNT_REQUEST y COUNT_REPLY

El campo *Message Type* puede ser 0 o 1, y es usado para identificar el tipo de mensaje. Un valor de 0 denota un

COUNT_REQUEST, mientras 1 indica un COUNT_REPLY. *Sequence Number* es usado para corresponder los mensajes COUNT_REQUEST con sus respectivos mensajes COUNT_REPLY, y para distinguir entre diferentes peticiones de un mismo nodo. *Timestamp* es una muestra de tiempo tomada en el origen cuando este inicia el proceso de conteo. Su finalidad es controlar los mensajes que están fuera de tiempo y los ataques de repetición. El origen y los nodos difunden mensajes COUNT_REQUEST junto con el argumento *HopAway* que representa el número de saltos que hay entre el origen del mensaje COUNT_REQUEST y su receptor. El origen que comienza el proceso debe especificar un valor de *HopAway* igual a 1. Cada nodo que retransmite el mensaje, seleccionará el más pequeño *HopAway* recibido hasta el momento y lo incrementará en 1. Cada vez que un nodo actualiza su campo *HopAway*, debido a la recepción de un mejor COUNT_REQUEST (más cercano al origen), tiene que actualizar su variable *nodeToGoBack*, que es una referencia al nodo con el más bajo *HopAway*, del cual el nodo actual ha recibido un mensaje COUNT_REQUEST. El campo *HopLimit* permite controlar hasta qué distancia (número de saltos) los mensajes COUNT_REQUEST son propagados desde el origen hacia los demás nodos, delimitando así el rango de conteo. El campo *Total* representa el número de nodos contados hasta el momento, siendo igual a 0 en mensajes COUNT_REQUEST. Es importante mencionar que antes de enviar un mensaje COUNT_REPLY (al nodo identificado por *nodeToGoBack*), un nodo debe actualizar este campo de acuerdo a los mensajes COUNT_REPLY recibidos hasta ahora, y agregar 1 a la suma para contarse a sí mismo.

C. Algoritmo

El origen primero propaga tres mensajes de tipo COUNT_REQUEST con un valor de *HopAway* igual a 1 (separado por INT_BTW_REQ). Cuando un nodo recibe el primer mensaje COUNT_REQUEST, hará lo siguiente:

- RBCT_TIME: El nodo propagará un COUNT_REQUEST con un *HopAway* igual al mínimo *HopAway* recibido en este tiempo +1. También almacenará en la variable *nodeToGoBack* el ID del nodo que envió el COUNT_REQUEST con el más pequeño valor de *HopAway*.
- RBCT_TIME + (1 * INT_BTW_REQ): El nodo retransmite el mensaje COUNT_REQUEST (debido a una posible colisión con el primer COUNT_REQUEST enviado). Si es necesario, los campos del mensaje COUNT_REQUEST son actualizados.
- RBCT_TIME + (2 * INT_BTW_REQ): El nodo retransmite el mensaje COUNT_REQUEST (debido a una posible colisión con el primer y segundo COUNT_REQUEST enviados). Si es necesario, los campos del mensaje COUNT_REQUEST son actualizados.
- REQ_TO: Si el nodo recibió mensajes COUNT_REPLY, entonces el nodo calcula el total de nodos basado en la variable *Total* recibida en los mensajes COUNT_REPLY (+1 para sumarse a sí mismo en el conteo total) y envía el resultado a su nodo *nodeToGoBack* como un mensaje unicast. Si después de REQ_TO el nodo no ha recibido ningún mensaje COUNT_REPLY, entonces genera un COUNT_REPLY con *Total* igual a 1 (este 1 representa el conteo de sí mismo) y lo envía al nodo *nodeToGoBack* como un mensaje unicast.
- DELAY_ASYNC: Si el nodo recibe mensajes COUNT_REPLY retrasados, entonces nuevamente vuelve a computar el número total de nodos basado en la variable *Total* recibida en los mensajes COUNT_REPLY (+1 para sumarse a sí mismo en el conteo total) y luego envía el resultado a su nodo *nodeToGoBack* como un mensaje COUNT_REPLY asíncrono.

D. Ejemplo de un Escenario de Propagación de Mensajes

En la Figura 3, se aprecia un escenario con 18 nodos, que incluyen el origen (identificado por O) que requiere un conteo. Los círculos alrededor de los nodos representan el rango de propagación de los mensajes enviados por ellos mismos.

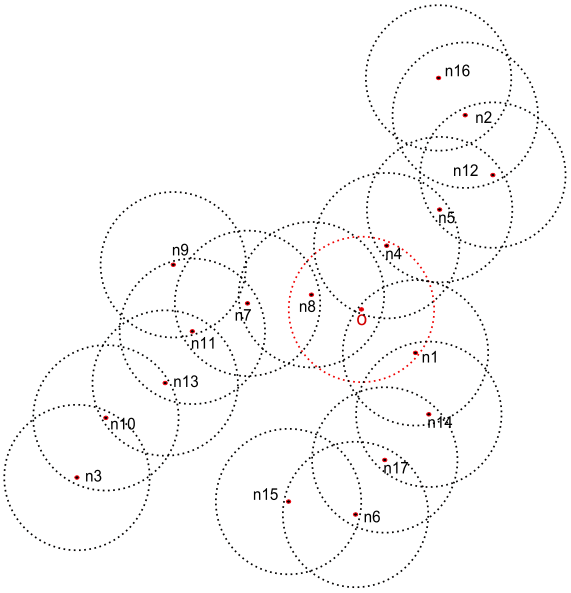


Figura 3: Ejemplo de un Escenario de Propagación de Mensajes

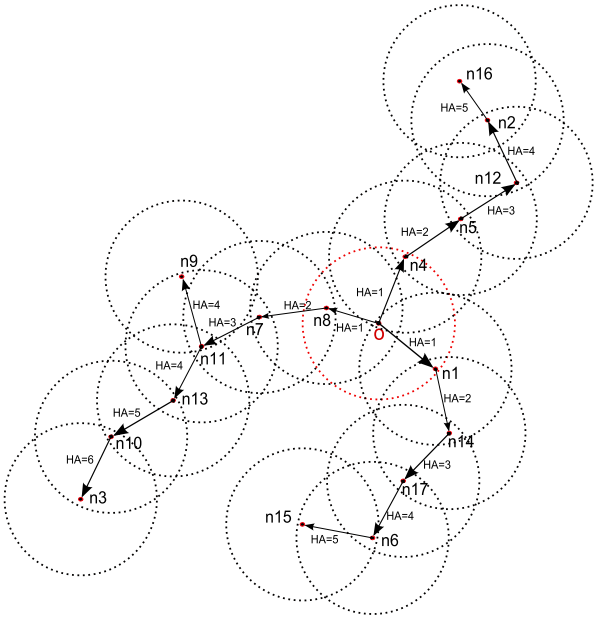


Figura 4: Propagación de Mensajes COUNT_REQUEST

En la Figura 4, podemos observar cómo los mensajes COUNT_REQUEST son propagados desde el origen hacia los nodos que están lejos de este, y cómo el valor de *HopAway* (representado por HA) es actualizado.

Asimismo, en la Figura 5 podemos ver cómo los mensajes COUNT_REPLY son propagados desde los nodos que están lejos del origen, hacia el nodo origen y cómo el valor de *Total* es actualizado.

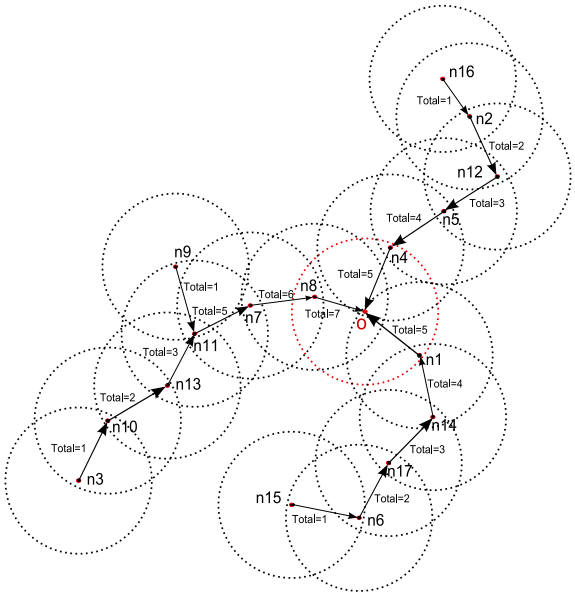


Figura 5: Propagación de Mensajes COUNT_REPLY

Por otra parte, los mensajes COUNT_REPLY asíncronos son enviados por un nodo con un ligero retardo de DELAY_ASYNC, siempre y cuando éste reciba mensajes COUNT_REPLY retrasados después de haber enviado su mensaje COUNT_REPLY síncrono (ver Figura 1), siendo la variable *Total* nuevamente actualizada. Por ejemplo, tal como se aprecia en la Figura 5, tanto n9 (*Total*=1) como n13 (*Total*=3) deberían enviar un mensaje COUNT_REPLY a n11. Supongamos que al vencer su tiempo REQ_TO, el nodo n11 solamente haya recibido el COUNT_REPLY de n13 (*Total*=3), eso es, n9 se ha retrasado y aún no ha enviado su mensaje COUNT_REPLY a n11. Entonces, n11 calculará su total de nodos basado en la variable *Total* (igual a 3) recibida de n13, agregándole 1 para contarse a sí mismo, y enviará su COUNT_REPLY síncrono con *Total*=4 a su *nodeToGoBack* (en este caso representado como el nodo n7). Ahora, supongamos que luego de haber enviado su COUNT_REPLY síncrono, el nodo n11 recibe el mensaje COUNT_REPLY pendiente de n9 (con *Total*=1). Entonces n11 nuevamente computará su número total de nodos de acuerdo a las variables *Total* recibidas de los nodos n9 (*Total*=1) y n13 (*Total*=3), al cual sumará 1 para representarse a sí mismo en el total, y esperará un tiempo de DELAY_ASYNC para enviar este resultado actualizado (igual a 5) a su *nodeToGoBack*, que es n7 (ver Figura 5) como un mensaje COUNT_REPLY asíncrono.

En base al diagrama de tiempo de la Figura 1, se determinó el tiempo que un nodo espera luego de la propagación del último mensaje COUNT_REQUEST y el envío del mensaje COUNT_REPLY síncrono a *nodeToGoBack* hacia el origen (llamado TIME_TO_WAIT), según el valor de su variable *HopAway* como:

Para el nodo origen (*HopAway*=0):

TIME_TO_WAIT = (*HopLimit* * RBCT_TIME * α) + (NUM_RBCT * (*HopLimit* – 1)) * (INT_BTW_REQ * β) + (*HopLimit* * DELAY * δ)

Para el caso de los últimos nodos (*HopAway* = *HopLimit*):

TIME_TO_WAIT = RBCT_TIME * α

Y para los demás nodos:

TIME_TO_WAIT = (*HopLimit* – *HopAway*) * (RBCT_TIME * α) + (NUM_RBCT * (*HopLimit* – 1 – *HopAway*) * (INT_BTW_REQ * β)) + ((*HopLimit* – *HopAway*) * DELAY * δ)

Donde los parámetros α, β y δ representan valores para acelerar la propagación de los mensajes COUNT_REQUEST y COUNT_REPLY y el parámetro DELAY es el tiempo que permite tomar en cuenta el tiempo de propagación de los mensajes COUNT_REPLY.

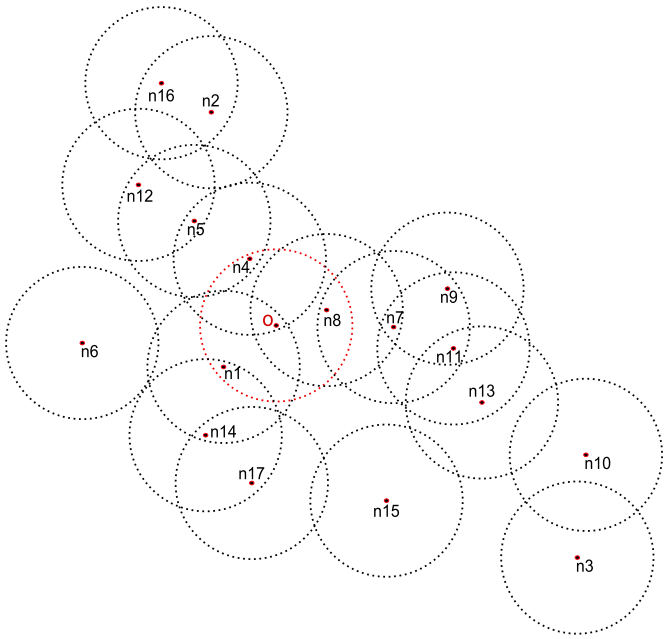


Figura 6: Explicación del Alcance del Conteo con HopLimit

Para aclarar todos los conceptos, consideremos el ejemplo del escenario de la Figura 6. Se puede observar que cuando el nodo origen envía un mensaje COUNT_REQUEST con *HopLimit* igual a 1, deberá contar un total de 3 nodos (n1, n4 y n8). Para un valor de *HopLimit* igual a 2, contará 6 nodos (n1, n4, n5, n7, n8 y n14). Para un valor de *HopLimit* igual a 3, el origen contará 10 nodos (n1, n4, n5, n7, n8, n9, n11, n12, n14 y n17). Ahora, para un valor de *HopLimit* igual a 4, el origen contará 11 nodos (n1, n4, n5, n7, n8, n9, n11, n12, n13, n14 y n17). En el caso de asignar un valor mayor que 4 para *HopLimit*, el conteo total de nodos no cambiará, es decir, el origen contará los mismos 11 nodos, ya que 6 nodos (n2, n3, n6, n10, n15 y n16) no pueden ser alcanzado desde el origen.

IV. Ambiente de Simulación

El presente algoritmo fue implementado en OMNeT++/INET. OMNeT++ es una herramienta de simulación de código abierto, basada en C++, multiplataforma (Windows, MacOS y Unix). Es un simulador de eventos discretos que permite modelar cualquier sistema compuesto por dispositivos que

interactúan entre sí, siendo una de sus principales fortalezas su interfaz gráfica de usuario (GUI), que permite crear archivos NED (un lenguaje de descripción para definir la estructura del modelo) e inspeccionar el estado de cada componente durante las simulaciones. Para simulaciones de redes de datos, OMNeT++ requiere de extensiones externas tales como INET. El "Framework" INET es un paquete de simulación de código abierto de comunicaciones de datos para el ambiente de simulación OMNeT++.

En la presente investigación, se decidió utilizar OMNeT++ por dos razones principales: (1) su potente interfaz gráfica de usuario que facilita la depuración de las simulaciones y (2) que es un proyecto muy activo con muchos modelos que se actualizan constantemente.

Se simularon diferentes escenarios donde los nodos fueron distribuidos aleatoriamente sobre un área cuadrada de 800m x 800m. El modelo de movilidad aleatorio RandomWPMobility fue seleccionado para reflejar el escenario más general de movilidad o desplazamiento de los nodos. Asimismo, para todas las simulaciones se seleccionó el estándar de comunicación inalámbrica WiFi (IEEE 802.11g) con una tasa de transferencia de 54 Mbps. El modelo de propagación en el espacio libre fue elegido para la atenuación de la señal con la distancia.

V. Resultados de las Simulaciones

En esta sección se presentan y se discuten algunos de los resultados de las simulaciones efectuadas.

A. Escenarios con Nodos y el Origen Estacionarios

Al igual que en el algoritmo planteado en [7], se ejecutó varias simulaciones para escenarios estacionarios, es decir, donde los nodos y el origen permanecen inmóviles.

Hicimos simulaciones con 50 nodos estacionarios, colocados aleatoriamente en un área cuadrada de 800m x 800m, con un valor del rango de propagación de 250 m, y un origen situado en el centro de la misma. La Tabla I muestra los resultados obtenidos cuando se varía los parámetros RBCT_TIME, INT_BTW_REQ y DELAY. Los resultados de las últimas 3 columnas son representados como valores *a/b*, donde *a* representa el resultado del algoritmo anterior [7] y *b* el resultado obtenido con el nuevo algoritmo. Tal como se aprecia en la Tabla I, la nueva versión del algoritmo contó un total de nodos más eficazmente que el algoritmo propuesto en [7]. Por ejemplo, para valores de los parámetros RBCT_TIME=0.01, INT_BTW_REQ=0.01 y DELAY=0.03, el origen debería contar 50 nodos. Ahora, cuando se ejecuta la simulación, el algoritmo propuesto en [7] contó 10 nodos con un tiempo de respuesta de 0.15s y un total de mensajes enviados igual a 155. Sin embargo, el nuevo algoritmo contó 47 nodos con un tiempo de respuesta de 1.08s y un número total de mensajes enviados igual a 178, resultando en una diferencia de conteo bastante notable. La Tabla I muestra la importancia de escoger bien los valores de los parámetros RBCT_TIME, INT_BTW_REQ y DELAY. En efecto, según los valores escogidos, se obtendrá un mejor tiempo de respuesta o una mayor precisión de conteo. Los desarrolladores de aplicaciones tendrán que seleccionar estos parámetros de acuerdo a sus necesidades [7].

Tabla I: Importancia de los Parámetros RBCT_TIME, INT_BTW_REQ y DELAY

RBCT_TIME, INT_BTW_REQ, DELAY	Rango de Propagación = 250m			
	Nodos Alcanzables	*Nodos Contados*	*Tiempo de Respuesta*	*Número Total de Msg Enviados*
0.01s, 0.01s, 0.03s	50	10/47	0.15 s/1.08 s	155/178
0.05s, 0.05s, 0.10s	50	15/49	0.63 s/1.39 s	158/181
0.05s, 0.05s, 0.20s	50	18/50	0.88 s/1.43 s	158/174
0.10s, 0.10s, 0.20s	50	23/50	1.26 s/1.90 s	167/180
0.10s, 0.10s, 0.40s	50	34/50	1.76 s/2.23 s	167/183
0.20s, 0.20s, 0.30s	50	43/50	2.28 s/2.68 s	173/177
0.25s, 0.25s, 0.30s	50	48/50	2.66 s/2.21 s	173/175
0.30s, 0.30s, 0.50s	50	50/50	3.24 s/2.93 s	173/173

En la Tabla II, variamos el número total de nodos estacionarios (10, 15, 20, 25, 30, 35, 40 y 50) ubicados aleatoriamente en un área cuadrada de 800m x 800m, y su rango de propagación (100m, 150m, 200m, 250m y 300m). Los resultados son representados como valores *a/b/c*, donde *a* denota el número de nodos que están dentro del alcance del origen usando enrutamiento multisalto (es decir, nodos que deben ser contados), *b* el número de nodos actualmente contados por el algoritmo propuesto en [7] (que de aquí en adelante denominaremos Algoritmo I), y *c* el total de nodos contados por el algoritmo propuesto en esta investigación (que llamaremos en lo sucesivo como Algoritmo II). Seleccionamos valores apropiados para los parámetros RBCT_TIME=0.2s, INT_BTW_REQ=0.2s y DELAY=0.4s, de acuerdo a los resultados encontrados en la Tabla I, iniciando el origen con un valor de *HopLimit*=3.

Tabla II: Conteo de Nodos en Diferentes Escenarios Estacionarios

Número Total de Nodos	Rango de Propagación en Metros				
	100m	*150m*	*200m*	*250m*	*300m*
10	1/1/1	1/1/1	3/3/3	10/10/10	10/10/10
15	2/2/2	4/4/4	7/7/7	14/14/14	15/15/15
20	2/2/2	4/4/4	11/11/11	19/19/19	20/20/20
25	4/4/4	8/8/8	14/14/14	23/22/23	25/24/25
30	5/5/5	14/14/14	24/23/24	29/28/29	30/30/30
35	7/7/7	16/16/16	29/27/29	34/33/34	35/35/35
40	8/8/8	22/22/22	34/33/34	39/39/39	40/37/40
50	11/11/11	28/26/26	48/43/47	50/50/50	50/50/50

Dado que existen nodos que no son alcanzables en estos escenarios, podemos observar que el algoritmo II mejora sustancialmente al algoritmo I. Por ejemplo, la Tabla II muestra que para un escenario de 50 nodos y un rango de propagación igual a 200 metros, el origen debería contar 48 nodos (en este caso 2 nodos no son accesibles desde el origen porque están fuera del rango de conteo). Ahora, cuando se ejecuta la simulación, el algoritmo I contó 43 nodos, en lugar de 48, lo cual es una buena aproximación. Sin embargo, el algoritmo II contó 47 nodos, siendo mucho más eficaz. Estas mismas diferencias se pueden apreciar para otros casos de simulación, tales como (25, 250), (25, 300), (30, 200), (30, 250), (35, 200), (35, 250), (40, 200) y (40, 300), donde el primer valor del par representa el número total de nodos mientras que el segundo valor representa el rango de propagación. Asimismo, la Tabla III proporciona información adicional (tiempo de respuesta y número total de mensajes enviados) para los mismos experimentos, para un rango de propagación de 250m. Por ejemplo, para un total de 35 nodos, 34 de los 34 nodos alcanzables por el origen fueron contados

efectivamente por el algoritmo II en 2.67s y con un número total de mensajes enviados igual a 101.

Tabla III: Tiempo de Respuesta y Número Total de Mensajes Enviados en un Escenario Estacionario Específico

Número Total de Nodos	Rango de Propagación = 250m			
	Nodos Alcanzables	*Nodos Contados*	*Tiempo de Respuesta*	*Número Total de Msg Enviados*
10	10	10	2.52 s	38
15	14	14	2.77 s	54
20	19	19	2.70 s	74
25	23	23	2.67 s	93
30	29	29	2.90 s	104
35	34	34	2.67 s	101
40	39	39	2.72 s	153
50	50	50	2.66 s	177

B. Escenarios con Nodos Móviles y el Origen Estacionario

En esta subsección, analizamos el rendimiento del algoritmo II en comparación con el algoritmo I, cuando los nodos tienen movilidad, pero el origen es estacionario y situado en el centro del área cuadrada de 800m x 800m. Para el desplazamiento de los nodos, se utilizó el modelo de movilidad aleatorio RandomWPMobility, variando la velocidad de los mismos de 1 mps (metros por segundo) a 45 mps, con un tiempo de espera de 0s (intervalo de tiempo entre alcanzar un destino y salir hacia un nuevo destino).

En la Tabla IV, variamos el rango de propagación de los nodos y el origen. Para todos estos escenarios, elegimos 30 nodos móviles, posicionados inicialmente de forma aleatoria. Los valores de los parámetros son: RBCT_TIME=0.2s, INT_BTW_REQ=0.2s y DELAY=0.4s. El origen inicia el proceso de conteo con *HopLimit*=3. Aquí también, los resultados son presentados como valores *a/b/c*, donde *a* denota el número de nodos que están dentro del alcance del origen usando enrutamiento multisalto, *b* el número de nodos actualmente contados por el algoritmo I, y *c* el total de nodos contados por el algoritmo II. Como era de suponer, podemos apreciar que el algoritmo II tiene una mejor precisión de conteo. También se puede observar que a medida que se incrementa la velocidad, en ambos algoritmos se pierde precisión en el conteo de nodos. Los mejores resultados son obtenidos cuando la velocidad varía de 1 a 20 mps.

Tabla IV: Conteo de Nodos cuando se Varía la Velocidad de los Nodos y los Valores del Rango de Propagación (Velocidad del Origen=0 mps)

Velocidad de Nodos (mps)	Rango de Propagación en Metros				
	100m	*150m*	*200m*	*250m*	*300m*
1	5/5/5	11/11/11	17/17/17	29/29/29	30/29/30
3	4/4/4	14/13/13	23/23/23	30/29/30	30/30/30
5	5/5/5	10/10/10	24/23/24	30/28/29	30/30/30
10	4/3/3	22/19/19	29/29/28	30/30/30	30/30/30
20	11/6/6	26/20/20	30/29/29	30/29/27	30/30/30
30	5/1/1	23/7/6	30/18/21	30/20/24	30/19/26
40	9/7/7	11/6/8	26/18/19	30/17/20	30/18/26
45	5/4/4	10/4/5	29/19/20	30/25/26	30/24/25

En la Tabla V, se optó por un rango de propagación fijo de 250m, variando tanto la velocidad de los nodos como el número de nodos. Según los resultados obtenidos, se deduce que en ambos algoritmos, igualmente como ocurre en los experimentos de la Tabla IV, se tiene un buen conteo de nodos para velocidades de hasta 20 mps; mientras que para velocidades altas, aunque el conteo se degrada, el algoritmo II es mucho más preciso. Por ejemplo, como se muestra en la Tabla V, para el caso de un número de 40 nodos en total y una velocidad de 45mps, el origen debería contar 40. Al ejecutar la simulación, el algoritmo I contó 30 nodos, mientras que el algoritmo II contó 33, siendo más efectivo al igual que para los demás casos.

Tabla V: Conteo de Nodos cuando se Varía la Velocidad de los Nodos y el Número Total de Nodos

Velocidad de Nodos (mps)	Número Total de Nodos						
	10	**15**	**20**	**25**	**30**	**40**	**50**
1	7/7/7	15/15/15	16/16/16	18/18/18	29/29/29	40/38/40	50/49/50
3	5/5/5	7/7/7	17/17/17	24/22/22	30/29/27	40/38/40	50/49/50
5	2/2/2	7/7/7	20/20/20	24/24/24	30/28/29	40/40/40	50/47/50
10	8/8/8	15/15/15	20/19/19	25/21/21	30/30/30	40/36/38	50/46/48
20	8/9/9	15/15/15	19/17/18	25/22/22	30/29/29	40/37/38	50/46/46
30	6/6/6	15/13/14	20/18/18	24/22/21	30/20/20	40/31/35	50/27/29
40	10/9/9	14/15/15	19/13/14	25/17/15	30/17/17	40/33/34	50/38/39
45	10/7/6	15/11/14	20/17/17	25/14/15	30/25/26	40/30/33	50/27/39

C. Escenarios con Nodos Estacionarios y el Origen Móvil

En esta subsección, se estudia comparativamente el comportamiento del algoritmo II con respecto al algoritmo I, para nodos estacionarios y un origen con movilidad. Al comienzo de las simulaciones, los nodos estacionarios son situados aleatoriamente en un área cuadrada de 800m x 800m, siendo colocado el origen en el centro de la misma, desplazándose de acuerdo al modelo de movilidad aleatorio RandomWPMobility, sin parar en las posiciones visitadas, es decir, con un tiempo de espera de 0s. Para estos experimentos, igualmente los valores de los parámetros son: RBCT_TIME=0.2s, INT_BTW_REQ=0.2s y DELAY=0.4s. El origen inicia el proceso de conteo con un *HopLimit*=3.

En la Tabla VI se presentan los resultados de la simulación de escenarios con 30 nodos, en los que se varía la velocidad del origen, así como el rango de propagación del mismo y de los nodos. Se puede observar que para ambos algoritmos, se tiene una gran precisión de conteo para velocidades de hasta 20 mps, que a medida que ésta se incrementa, la precisión del conteo disminuye, pero siendo significativamente más eficiente en el algoritmo II.

Tabla VI: Conteo de Nodos cuando se Varía la Velocidad del Origen y los Valores del Rango de Propagación

Velocidad del Origen (mps)	Rango de Propagación en Metros				
	100m	*150m*	*200m*	*250m*	*300m*
1	1/1/1	3/3/3	10/10/10	29/27/29	30/29/30
3	3/3/3	3/3/3	10/10/10	24/24/24	30/17/18
5	3/3/3	6/6/6	10/10/10	24/22/24	30/20/22
10	2/2/2	6/4/6	10/10/10	25/23/25	30/28/30
20	2/2/2	6/6/6	10/9/9	24/24/24	30/23/28
30	1/1/1	6/3/3	10/7/7	26/20/22	30/18/19
40	1/1/1	5/2/2	6/2/3	19/17/19	23/22/23
45	3/3/3	12/7/7	4/3/4	13/11/13	23/17/23

En la Tabla VII, se eligió un valor de rango de propagación fijo de 250m, pero se varió la velocidad del origen, así como el número total de nodos. De acuerdo a los resultados obtenidos en estos escenarios, el conteo de nodos fue muy similar a los experimentos llevados a cabo en la Tabla VI, es decir muy acertado para velocidades de hasta 20 mps, con una degradación del conteo para velocidades superiores, siendo el algoritmo II él que hace el mejor conteo.

Tabla VII: Conteo de Nodos cuando se Varía la Velocidad del Origen y el Número Total de Nodos

Velocidad del Origen (mps)	Número Total de Nodos						
	10	15	20	25	30	40	50
1	3/3/3	15/15/15	19/19/19	25/25/25	29/27/29	39/39/39	50/47/50
3	3/3/3	14/14/14	19/8/19	19/19/19	24/24/24	35/35/35	50/45/45
5	3/3/3	14/14/14	12/12/12	17/17/17	24/22/24	35/33/35	50/50/50
10	4/1/3	12/12/12	12/9/9	22/22/22	25/23/25	30/22/23	45/42/45
20	3/3/3	10/6/4	8/8/8	22/21/22	24/24/24	23/23/23	40/38/40
30	3/3/3	12/9/11	9/8/8	18/18/18	26/20/22	32/26/29	48/46/48
40	3/3/3	14/6/6	12/9/12	13/4/4	19/17/19	37/32/28	47/45/45
45	3/2/3	11/11/11	11/11/11	9/9/9	13/1113	37/21/20	48/40/40

D. Escenarios con Nodos y el Origen Móviles

En esta sección, se reporta y se analiza comparativamente los resultados de las simulaciones realizadas con los algoritmos I y II, para escenarios donde los nodos y el origen tienen movilidad. Al comienzo de las simulaciones, los nodos se sitúan aleatoriamente en un área cuadrada de 800m x 800m, mientras que el origen es ubicado en el centro de la misma. Tanto los nodos como el origen se desplazan según el modelo de movilidad aleatorio RandomWPMobility, con un tiempo de espera de 0s. Asimismo, se seleccionó los valores a continuación para los parámetros: RBCT_TIME=0.2s, INT_BTW_REQ=0.2s y DELAY=0.4s, iniciando el origen el proceso de conteo con un valor de *HopLimit* igual a 3.

La Tabla VIII representa los resultados de la simulación de escenarios con 30 nodos, donde variamos tanto su velocidad como la del origen, así como su rango de propagación.

Tabla VIII: Conteo de Nodos cuando se Varía la Velocidad de los Nodos y del Origen así como los Valores de Rango del Propagación

Velocidad (mps)	Rango de Propagación en Metros				
	100m	*150m*	*200m*	*250m*	*300m*
1	5/4/4	11/11/11	18/18/18	29/29/28	30/30/30
3	4/4/4	9/9/9	17/17/17	30/24/24	30/30/30
5	5/4/4	8/8/8	18/18/18	30/26/29	30/26/27
10	9/8/8	15/15/15	24/25/24	29/24/24	30/26/26
20	9/6/8	5/5/5	13/12/12	27/21/27	30/29/29
30	2/1/1	17/5/5	30/19/19	28/20/24	30/19/20
40	5/1/1	17/3/5	28/16/19	30/20/20	30/20/22
45	10/2/2	12/2/2	20/2/3	22/9/17	30/11/15

En la Tabla IX, se eligió un rango de propagación fijo de 250m, pero se varió la velocidad de los nodos y del origen, así como el número total de nodos. Al igual que en los experimentos anteriores (ver Tabla VIII), se obtuvo también un conteo eficaz para velocidades de hasta 20 mps. En tanto que para velocidades más altas, la precisión del conteo se degrada, pero siendo más acertada en el algoritmo II que en el algoritmo I.

Tabla IX: Conteo de Nodos cuando se Varían la Velocidad de los Nodos y el Origen así como también el Número Total de Nodos

Velocidad (mps)	Número Total de Nodos							
	10	15	20	25	30	35	40	50
1	10/10/10	15/15/15	16/16/16	25/25/25	29/29/29	35/35/35	40/40/40	50/50/50
3	8/8/8	15/14/15	17/17/17	25/25/25	30/24/24	35/35/35	40/40/40	50/45/45
5	6/6/6	15/13/15	19/17/17	24/23/23	30/26/29	35/35/35	40/40/40	50/48/48
10	6/6/6	14/11/9	20/20/20	25/24/25	29/24/24	35/30/34	40/40/40	49/48/48
20	5/5/5	11/11/11	10/9/9	25/20/23	24/21/21	31/27/27	34/26/28	49/46/49
30	3/3/3	10/3/4	20/17/17	15/15/15	28/20/20	23/21/21	37/29/33	50/36/47
40	10/9/10	14/10/10	19/12/12	20/3/17	30/20/20	34/22/25	39/20/20	50/21/28
45	10/7/9	15/12/12	19/14/16	23/6/12	22/9/13	35/7/30	40/35/36	50/15/19

VI. Conclusiones y Trabajos Futuros

En este artículo, se presenta una nueva versión de un algoritmo para contar nodos con tecnologías inalámbricas, basado en las ideas planteadas en el algoritmo propuesto en [7], donde se toman en cuenta aquéllos mensajes retrasados que pudieran ser enviados por los nodos y por tanto ser también contados. Se hace un estudio comparativo de ambos algoritmos.

Para validar y comparar los algoritmos, se realizaron dos tipos de simulaciones: (1) cuando los nodos y el origen son estacionarios, y (2) cuando la movilidad está involucrada en los experimentos, resultando en un cambio de topología entre los dispositivos. En función de las simulaciones ejecutadas, con valores apropiados de los parámetros, se deduce que para el conteo de nodos sobre escenarios estacionarios y móviles, el algoritmo II es mucho más eficiente que el algoritmo I (propuesto en [7]).

Como trabajo futuro, se tiene la intención de mejorar el nuevo algoritmo para obtener un conteo más eficaz cuando los nodos se mueven a altas velocidades, en base a la posición actual de estos, es decir, se agregaría un GPS a cada nodo. También, se pretende adaptar el algoritmo a redes inalámbricas multicanales, con la finalidad de reducir las colisiones que ocurren entre los mensajes enviados. Finalmente, estamos interesados en el estudio de la aplicación del algoritmo en el contexto vehicular (autopistas, vías urbanas, caminos rurales, parkings, entre otros) utilizando WAVE (Wireless Access in Vehicular Environment) [27], ya que podría ser utilizado en aplicaciones para mejorar la seguridad y la comodidad de los conductores y pasajeros.

Agradecimientos

Agradecemos al CDCH-UCV (Consejo de Desarrollo Científico y Humanístico) que en gran parte apoyó esta investigación bajo el número de subvención: PG 03-8066-2011/1.

Referencias

[1] E. Kell and E. Mills. *Traffic Detector Handbook*. U.S. Department of Transportation, Federal Highway Administration, 2nd Ed., pp. 1–39. USA, 1990.

[2] L. Klein. *Sensors Technologies and Data Requirements for ITS*. Artech House Publishers, Norwood, USA, 2001.

[3] L. Mimbela and L. Klein. *A Summary of Vehicle Detection and Surveillance Technologies used in Intelligent Transportation Systems*. Handbook, Federal Highway Administration, Intelligent Transportation Systems. USA, 2007.

[4] G. Leduc. *Road Traffic Data: Collection Methods and Applications*. European Commission, Joint Research Center, Institute for Prospective Technological Studies, Seville, Spain. 2008.

[5] T. Schweizer. *Methods for Counting Pedestrians*, in proceedings of the 6th International Conference on Walking in the 21st Century (Walk21-VI). "Everyday Walking Culture", Zurich, Switzerland, September 2005.

[6] J. Rojas. *Conteo de Personas mediante Videocámaras*. Tesis de Maestría en Ciencias (Matemáticas Aplicadas e Industriales). Universidad Autónoma Metropolitana, México City, México, Abril 2013.

[7] E. Gamess and M. Contreras. *A Proposal for an Algorithm to Count Nodes using Wireless Technologies*. International Journal of High Performance Computing and Networking. In press.

[8] C.-H. Chen, Y.-C. Chang, and D-.J. Wang. *People Counting System for Getting In/Out of a Bus Based on Video Processing*, in proceedings of the 8th International Conference on Intelligent Systems Design and Applications (ISDA'08), pp. 565–569, November 2008.

[9] D. Roqueiro and V. Petrushin. *Counting People using Video Cameras*. International Journal of Parallel, Emergent and Distributed Systems, vol. 22, article 3, pp. 193–209, USA, May 2007.

[10] H.-C. Chen, Y.-C. Chang, N.-J. Li, C.-F. Weng, and W.-J. Wang. *Real-Time People Counting Method with Surveillance Cameras Implemented on Embedded System*, in proceedings of the World Congress on

Engineering and Computer Science 2013. San Francisco, USA, October 2013.

[11] P. Lengvenis, R. Simutis, V. Vaitkus, and R. Maskeliunas. *Application of Computer Vision Systems for Passenger Counting in Public Transport*. Elektronika ir Elektrotechnika, vol. 19, no. 3, pp. 69–72, March 2013.

[12] T. Yahiaoui, L. Khoudour, and C. Meurie. *Real-Time Passenger Counting in Buses using Dense Stereovision*, Journal of Electronic Imaging, vol. 19, no. 3, July 2010.

[13] H. Celik, A. Hanjalic, and E. Hendriks. *Towards a Robust Solution to People Counting*, in proceedings of the 2006 IEEE International Conference on Image Processing. Atlanta, Georgia, USA, October 2006.

[14] G.-G. Lee, H-K. Kim, J.-Y. Yoon, J.-J. Kim, and J.-J. Kim, *Pedestrian Counting using an IR Line Laser*, in proceedings of the 2008 International Conference on Convergence and Hybrid Information Technology (ICHIT'08), Daejeon, Korea, August 2008.

[15] M. Lei, D. Lefloch, P. Gouton, and K. Madani. *A Video-Based Real-Time Vehicle Counting System using Adaptive Background Method*, in proceedings of the 4th IEEE International Conference on Signal Image Technology and Internet Based Systems, Bali, Indonesia, November 2008.

[16] M. Tursun and G. Amrulla. *A Video Based Real-Time Vehicle Counting System using Optimized Virtual Loop Method*, in proceedings of the 2013 International Workshop on Systems, Signal Processing and their Applications (WoSSPA). Algiers, Algeria, May 2013.

[17] C.-K. Ki. *Moving Object Counting with an Infrared Sensor Network*. Master Thesis in Computer Science and Engineering. Hong Kong University of Science and Technology. Hong Kong, August 2007.

[18] H. Gros-Desormeaux, P. Hunel, N. Vidot, and E. Stattner. *Acoustic Counting Algorithms for Wireless Sensor Networks*, in proceedings of the 6th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks, pp. 79–84. New York, USA, 2009.

[19] M. Litzenberger, B. Kohn, G. Gritsch, N. Donath, C. Posch, N. Belbachir, and H. Garn. *Vehicle Counting with an Embedded Traffic Data System using an Optical Transient Sensor*, in proceedings of the 10th International IEEE Conference on Intelligent Transportation Systems (ITSC'07), Seattle, WA, USA, September 2007.

[20] K. Fuerstenberg, J. Hipp, and A. Liebram. *A Laserscanner for Detailed Traffic Data Collection and Traffic Control*, in proceedings of the 7th World Congress on Intelligent Transport Systems (ITS'00), Turin, Italy, November 2000.

[21] F. Bu and C.-Y. Chan, *Pedestrian Detection in Transit Bus Application: Sensing Technologies and Safety Solutions*, in proceedings of the 2005 IEEE Intelligent Vehicles Symposium, pp. 100–105, Las Vegas, Nevada, USA, June 2005.

[22] K. Frerstenberg and U. Lages. *Pedestrian Detection and Classification by Laser Scanners*, in proceedings of the 9th European Engineers Cooperation International Congress, Paris, France, 2003.

[23] S. Descamps, A. Béchet, X. Descombes, A. Arnaud, and J. Zerubia. *An Automatic Counter for Aerial Images of Aggregations of Large Birds*, Bird Study, vol. 58, no. 3, pp. 302–308, 2011.

[24] B. Sirmacek, M. Wegmann, A.D.P. Cross, J.G.C. Hopcraft, P. Reinartz, and S. Dech, *Automatic Population Counts for Improved Wildlife Management using Aerial Photography*, in proceedings of the 2012 International Congress on Environmental Modeling and Software. Leipzig, Germany. July 2012.

[25] R. Kissell and S. Nimmo, *A Technique to Estimate White-Tailed Deer Odocoileus Virginianus Density using Vertical-Looking Infrared Imagery*, Wildlife Biology, vol. 17, no. 1, pp. 85–92, 2011.

[26] E. Gamess and I. Mahgoub. *A Novel VANET-Based Approach to Determine the Position of the Last Vehicle Waiting at a Traffic Light*, in proceedings of the 2011 International Conference on Wireless Networks (ICWN'11), Las Vegas, Nevada, USA, July 2011.

[27] *IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Multi-Channel Operation*. IEEE 1609.4. November 2006.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 72-80, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Descriptores Visuales de MPEG-7 Empleando la GPU

Víctor Felipe[1], Esmitt Ramírez[2]
victor.felipe@ciens.ucv.ve, esmitt.ramirez@ciens.ucv.ve

[1] Centro de Cálculo Científico y Tecnológico, Universidad Central de Venezuela, Caracas, Venezuela
[2] Centro de Computación Gráfica, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: El ser humano puede de forma intuitiva seleccionar un grupo de fotografías de un conjunto e identificarlas como similares. Para ello, se emplean criterios basados en procesos cognitivos de aprendizaje centrados en sus características visuales. En un computador, se trata de emular dicho proceso al calcular un vector de características para una imagen, el cual permite identificarlas por sus atributos como su luminosidad, color predominante, intensidad, tonalidad, entre otros. El estándar MPEG-7 es una representación de imágenes y video que define, entre diversos atributos, características llamadas descriptores visuales los cuales pueden ser empleados para la aplicación de funciones de similitud entre imágenes. Sin embargo, el costo computacional de obtener estos descriptores es elevado. En este trabajo proponemos realizar estos cálculos empleando arquitecturas paralelas ofrecidas por las tarjetas gráficas. De esta forma, se realizan modificaciones a la propuesta original de MPEG-7 para ser ajustadas a las GPUs y obtener resultados en menores tiempos manteniendo la eficacia. La experimentación realizada justifica nuestra propuesta al implementar descriptores visuales de color, textura y forma, aplicados a una gran cantidad de imágenes permitiendo determinar con precisión la similitud entre pares.

Palabras Clave: Descriptores Visuales; Procesamiento Digital de Imágenes; MPEG-7; GPUs; Fotomosaicos.

Abstract: The human being is able to intuitively select a group from a set of images, and identify them as similar. Thus, some criteria are used based on cognitive learning processes focused on their visual features. On a digital device, this process is emulated generating a vector of image features, allowing the identification of their attributes such as luminosity, predominant colors, intensity, tonality and others. MPEG-7 defines visual descriptors which can be used to apply similarity functions between images. However, the computational cost of this measurement is usually high. In this work, we propose an approach to accelerate these calculations using the parallel architectures offered by GPUs. Thus, modifications to the original MPEG-7 proposal were performed to adjust our implementation to these architectures in order to reduce the computational cost of these computations. The experimentation accomplished justified our work in the use of high amount of images to determine the precision of these visual descriptors to find similar images.

Keywords: Visual Descriptors; Digital Image Processing; MPEG-7; GPUs; Photomosaics.

I. Introducción

El MPEG (*Moving Picture Experts Group*) establecido en 1988 ha desarrollado una serie de estándares para el manejo, tratamiento, compresión y visualización de contenido digital audiovisual. Así nace el estándar MPEG-7 que se centra en proveer descripciones de imágenes, audio y video, lo cual contribuye al filtrado y la categorización de contenido. Para ello, MPEG-7 busca una forma simple de conectar los elementos del contenido audiovisual, así como encontrar y seleccionar de forma adecuada la información que un usuario requiere. El estándar permite el manejo de audio, modelos 3D, video e imágenes, siendo esta última el punto de interés para nuestro estudio.

Es conocido que el contenido presente en el mundo actual a través de los diversos medios es altamente audiovisual, con el objeto de llegar de forma directa a un usuario final. Diversos contenidos para publicidad, entretenimiento, educación, entre muchos otros, son primordiales y están presentes con mayor auge cada día. Así, la necesidad de mantener contenido inequívoco toma importancia. Un caso particular lo constituyen las imágenes, las cuales pueden estar duplicadas en diversos ámbitos dentro de un gran repositorio de datos (e.g. Internet), lo cual puede resultar en redundancia, necesidad de almacenamiento, mal uso, etc. Del mismo modo, resulta interesante encontrar muchas versiones ligeramente diferentes de una misma imagen. Por ejemplo, una aplicación para dispositivos móviles de reconocimiento de lugares turísticos que tome como entrada una fotografía de un lugar a ubicar (e.g. la Torre Eiffel en París, Francia).

Sin embargo, desde el punto de vista computacional el proceso de encontrar imágenes similares dentro de un gran banco de datos no es tarea trivial. Una buena técnica es llamada *Query by Example* que es empleada principalmente por sistemas de

consulta de imágenes mediante ejemplo (*Content-based Image Retrieval* - CBIR) [1], que permite buscar imágenes basadas en su contenido dentro del contexto de color, textura y forma.

Este trabajo se basa en la construcción de descriptores visuales del estándar MPEG-7 basados en color, textura y forma para conseguir un conjunto de imágenes similares dentro de un repositorio, teniendo como entrada una imagen base. Dado el alto cómputo requerido para obtener los descriptores del estándar, se realiza un diseño e implementación bajo una arquitectura paralela que ofrece un hardware de bajo costo como lo son las tarjetas gráficas. Para ello, se emplea la arquitectura CUDA (*Compute Unified Device Architecture*) como base de trabajo ofreciendo muy buenos resultados al aplicar las diversas funciones de similitud de los descriptores visuales. Así, nuestra propuesta presenta una implementación eficaz y eficiente del estándar aplicando modificaciones que permiten mejorar y adaptar los descriptores a un ambiente bajo la GPU (*Graphics Processing Unit*). En este sentido, se utilizan descriptores visuales para la generación de fotomosaicos como caso de estudio de una aplicación de tipo CBIR.

Este artículo se organiza como sigue: en la Sección II, muestra un resumen de los trabajos previos relacionados con nuestra investigación. La definición de los descriptores visuales del estándar MPEG-7 se presenta en la Sección III. La Sección IV presenta el enfoque utilizado para la implementación de cada uno de los descriptores, y en la Sección V se muestra la experimentación realizada y los resultados obtenidos de dicho enfoque. Finalmente, en la Sección VI se presentan las conclusiones de nuestra investigación y posibles trabajos futuros.

II. Trabajos Previos

Las aplicaciones basadas en CBIR, corresponden a una rama de estudios en diversos centros de investigación del mundo, existiendo actualmente gran cantidad de información sobre las tecnologías que las implementan, métricas, buenas prácticas, entre otras [2][3]. En especialidades como la medicina, se ha empleado para el diagnóstico de patologías conocidas basadas en imágenes radiográficas, muestras citológicas, MRI (*Magnetic Resonance Imaging*), entre otras.

De manera habitual, con el objetivo de mantener una consistencia adecuada entre diversas aplicaciones, éstas han optado por emplear el estándar MPEG-7 para poder persistir y ser de utilidad en diversos ámbitos [4].

Diversos trabajos asociados a los descriptores visuales del estándar MPEG-7 han sido desarrollados recientemente. En [5] se presenta el algoritmo *k*-medias que es utilizado para la obtención de características de color a partir de imágenes. Por otro lado, Sergyán [6] propone una medida de similitud que permite comparar imágenes en base a estas características. Recientemente, Felipe y Ramírez [7] presentaron una implementación eficiente del algoritmo *k*-medias empleando la GPU.

En cuanto al uso de descriptores de forma, Park et al. [8] proponen el cálculo de características de forma siguiendo un esquema en bloque a partir de sub-imágenes, considerando la información de los bordes presentes, tanto de forma local como global.

Por su parte Hosny presenta en su trabajo [9] una aproximación para el cálculo de los coeficientes de la transformada radial angular (ART), utilizada para determinar características de forma asociadas a imágenes, llevando a cabo un proceso de interpolación como lo muestra Xin [10] en su trabajo.

En este trabajo, planteamos un caso de estudio basado en el uso de fotomosaicos para demostrar la efectividad de los descriptores de MPEG-7 en la GPU. De este tópico, existen diversas investigaciones que presentan técnicas novedosas para su generación, variando su aspecto visual [11][12][13].

Enfocando la utilidad de los descriptores visuales a una aplicación, los buscadores convencionales permiten la búsqueda de elementos (e.g. imágenes) haciendo uso de sus metadatos tales como su nombre, lo cual dificulta en muchos casos encontrar los elementos deseados. La utilización de información que puede ser obtenida a partir de características tales como color, textura y forma, facilita la búsqueda de imágenes similares. Recientemente, la utilización de sistemas de esta naturaleza se ha incrementado debido a la alta disponibilidad y poder de procesamiento provisto por los dispositivos móviles (e.g. aplicaciones como *Google Goggles* [14] y *CamFind* [15]).

En la literatura presentada, las investigaciones existentes no hacen énfasis en el costo computacional que el cálculo de descriptores visuales representa, lo cual ha motivado la realización de este trabajo bajo una arquitectura paralela de grano fino como es el caso de las GPUs.

III. Descriptores Visuales de MPEG-7

Los descriptores visuales definidos en el estándar MPEG-7 describen contenido (i.e. imágenes y video) en base a características tales como color, textura, forma y movimiento:

- Color: Es una característica visual robusta a los cambios en el ángulo de visión, así como a traslaciones y rotaciones de las ROI (*Region of Interest*). El estándar MPEG-7 presenta diversos descriptores que representan distintos aspectos asociados al color.
- Textura: Contiene información estructural importante acerca de las superficies y su relación con el entorno haciendo referencia a los patrones visuales que tienen o sus propiedades homogéneas. Es una característica natural de cualquier superficie (e.g. césped, paredes de ladrillos, etc.).
- Forma: Provee información relevante para la búsqueda y comparación de imágenes, presentando formas elementales basadas en regiones y contornos, es decir, patrones estructurales de las superficies y su entorno.
- Movimiento: Los descriptores de color, textura y forma pueden ser empleados para la indexación tanto de imágenes como videos. Los descriptores de video proveen pistas poderosas relacionadas a la ubicación espacial de los objetos presentes en diversos cuadros.

Existen diversos descriptores presentes en el estándar MPEG-7 y para los fines de este trabajo se consideran tres de ellos, que permiten obtener información asociada a las características de color, textura y forma. A continuación, se muestra una breve descripción de cada uno.

A. Descriptor de Colores Dominantes

El descriptor de colores dominantes (*Dominant Color Descriptor* - DCD) provee una descripción compacta de los colores representativos en una imagen. Se define como un conjunto de k duplas como en (1), donde k representa el número de colores dominantes:

$$DCD = \{(c_i, w_i)\} \text{ donde } i \{0,1,...,k-2, k-1\} \quad (1)$$

El valor de c_i es representado por un vector de componentes de un espacio de color (e.g. en el caso del espacio RGB consta de 3 valores), mientras que w_i representa la proporción del número de píxeles de la imagen, asociados al color dominante c_i en el rango [0,1].

La tarea consiste en encontrar los k colores dominantes de una imagen. El algoritmo clásico para extraer dichos colores es el algoritmo k-medias.

B. Descriptor de Histograma de Bordes

El descriptor de histograma de bordes (*Edge Histogram Descriptor* - EHD) toma ventaja de la distribución espacial de los bordes presentes en una imagen. Para localizar la distribución de los bordes en regiones específicas de una imagen se divide el espacio en 4x4 sub-imágenes y éstas a su vez en bloques (ver Figura 1). De esta forma, se genera un histograma que representa esta distribución en una imagen.

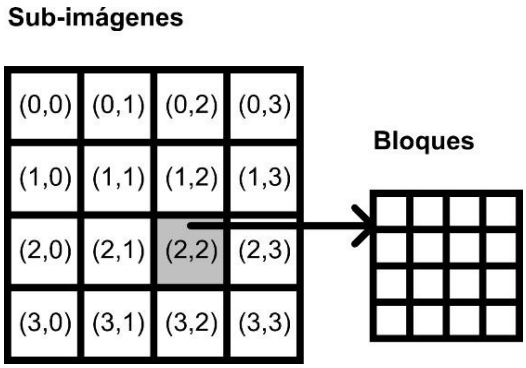


Figura 1: División del Espacio de una Imagen en Sub-Imágenes y Bloques

Cada bloque de una sub-imagen se puede clasificar como borde horizontal, vertical, diagonal 45º, diagonal 135º o no direccional. Luego de realizar el proceso de extracción del tipo de borde asociado a cada bloque, se determina la cantidad de bloques asociados a cada uno de los cinco tipos definidos. Dado que se tienen 16 sub-imágenes, se generan *5x16=80* contenedores para representar el histograma que recibe el nombre de histograma local de bordes (LEH).

Los valores de los contenedores son normalizados en base al número de bloques clasificados de un mismo tipo, obteniendo valores en el intervalo [0,1]. Por otro lado, se generan dos histogramas adicionales denominados histograma global de bordes (GEH) e histograma semi-global de bordes (SGEH), haciendo uso de la información del LEH, con el fin de incrementar la precisión del EHD.

C. Descriptor de Forma Basado en Regiones

El descriptor de forma basado en regiones (*Region-based Shape Descriptor* - RBSD) representa la distribución de píxeles dentro de un objeto o región en R^2. Este descriptor emplea la transformada radial angular (*Angular Radial Transform* - ART) [16] con valores complejos sobre un disco unitario en coordenadas polares.

Los coeficientes de la ART de orden p y q de una función de intensidades $I(x,y)$ están definidos por la proyección de una imagen de entrada en las funciones de base de la ART como se muestra en (2).

$$F_{pq} = \int_0^{2\pi}\int_0^1 V^*_{pq}(r,\theta) f(r,\theta) r dr d\theta \quad (2)$$

La función de base de la ART V^*_{pq} de orden p y q representa polinomios ortogonales continuos definidos en coordenadas polares sobre un disco unitario, mientras que el símbolo * representa su conjugada compleja, siendo la función $f(r,\theta)$ la correspondencia de la función de intensidades en coordenadas polares.

Con el objetivo de resolver las integrales de (2), se lleva a cabo una conversión (ver (3)), a una ecuación discreta aproximada para los coeficientes de la ART de orden p y q.

$$F_{pq} = \frac{1}{2\pi} \sum_{\forall i} \sum_{\forall j} f(r_i, \theta_{ij}) I_p(r_i) I_q(\theta_{ij}) \quad (3)$$

siendo las componentes I_p e I_q las representaciones de las partes radial y angular respectivamente de la función base de la ART. La función $f(r_i, \theta_{ij})$ se puede deducir a partir de la función original (i.e. imagen de entrada) empleando una interpolación basada en un spline cúbico.

IV. Enfoque Propuesto

Como se mencionó anteriormente, la generación de vectores característicos que representen de forma óptima una imagen no es una tarea trivial, dado que es posible obtener una gran cantidad de información para su representación. Los descriptores visuales de MPEG-7 permiten obtener información relevante a partir de imágenes en base a sus características tales como color, textura y forma.

Nuestra propuesta permite obtener estos descriptores visuales empleando la GPU, seleccionando el descriptor de colores dominantes, de histograma de bordes y de forma basado en regiones. Así, para dos imágenes de entrada I e I', se busca obtener los valores de sus descriptores y determinar su grado de similitud.

La obtención de los descriptores visuales de MPEG-7 involucra una gran cantidad de operaciones computacionales y por ello el empleo de la GPU. Desde este punto al referirse a hilos, se hace referencia a la secuencia de instrucciones más pequeña que es manejada independientemente por la tarjeta gráfica. A continuación, se describe el procedimiento para el cálculo de cada uno de los descriptores.

A. Colores Dominantes

El cálculo de los colores dominantes se realiza basado en el algoritmo k-medias. La idea básica es agrupar en k grupos una población de n individuos, basado en un criterio de similitud.

En un proceso de reducción de *C* colores iniciales de una imagen *I* (cuantización), se identifican los *k* grupos que representan los colores dominantes y los *n* individuos como los píxeles de *I*. En nuestra propuesta el cálculo computacional se realiza en la GPU y la CPU.

En la GPU, dada una imagen *I* y un conjunto de colores *C*, se determina para cada píxel el color que mejor lo represente empleando la medida de distancia euclidiana. Debido a la independencia entre los píxeles de *I*, se emplea un enfoque paralelo donde se agrupan hilos en conjuntos que denominamos bloques *B*.

En la CPU, se actualizan los valores centrales y se realiza el proceso de cuantización de la imagen en base a la información obtenida. Un bloque B_i procesa la fila *i* de *I*, mientras que un hilo *j* del mismo bloque se encarga de procesar los píxeles en las posiciones *(i, j + k x n)*, donde $k \geq 0$ y *n* representa el número de hilos de cada bloque. La Figura 2 ilustra esta distribución de bloques e hilos con *n = 4*.

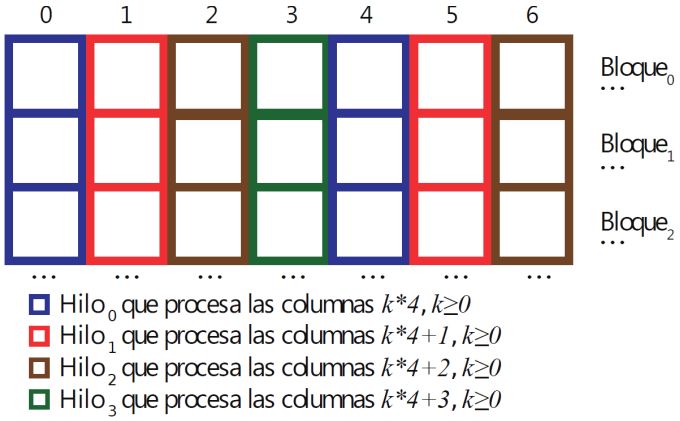


Figura 2: Esquematización de la Distribución de Píxeles en Hilos de Distintos Bloques con n = 4

El descriptor de colores dominantes basado en el algoritmo *k*-medias permite involucrar ponderaciones en el DCD que representan el número de ocurrencias de colores dominantes en la imagen *I* (i.e. imagen cuantizada).

Para calcular la distancia entre las imágenes *I* e *I'* se emplea la distancia entre las duplas generadas por sus respectivos descriptores I^A e I^B, como se muestra en (4).

$$D(I,I') = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left\|c_i^I - c_j^{I'}\right\| * \left|p_i^I - p_j^{I'}\right| \qquad (4)$$

Los valores de *M* y *N* representan el número de colores dominantes obtenidos de las imágenes *I* e *I'* respectivamente.

B. Histograma de Bordes

El descriptor de histograma de bordes (EHD) obtiene los tipos de borde por cada bloque creado en la imagen *I* y los clasifica de acuerdo a su tipo de borde de acuerdo a 5 posibles. Independientemente de la dimensión de *I*, se divide en un número predeterminado de bloques, existiendo una dependencia entre sus dimensiones y la resolución de la imagen.

Cada bloque se subdivide en cuatro sub-bloques y se calcula el valor medio de la luminancia de cada uno de ellos. Esta luminancia se emplea para obtener los bordes a través de una operación de convolución para calcular la magnitud de éstos (i.e. de los 5 bordes a clasificar). Si la mayor de las magnitudes obtenidas es superior a un umbral, el bloque se clasifica como la dirección asociada a dicha magnitud.

El estándar MPEG-7 define únicamente el uso del LEH, sin embargo el uso de información local puede no ser suficiente para una descripción correcta de la imagen. En este trabajo, consideramos dos histogramas adicionales para mejorar la precisión de este descriptor: el histograma global de bordes (GEH) y el histograma semi-global de bordes (SGEH). El GEH emplea información de todo el espacio de *I* y el SGEH considera distintas distribuciones de los bordes de *I*.

Es posible obtener los valores del GEH y SGEH directamente del LEH sin necesidad de llevar a cabo un procesamiento adicional.

La implementación de este descriptor en la GPU, utiliza un conjunto de hilos para procesar cada una de las 16 sub-imágenes (i.e. configuración de *4x4* sub-imágenes). A su vez, cada hilo realiza el cómputo requerido para determinar el tipo de borde asociado a cada uno de los bloques de las sub-imágenes, dando cobertura a todo el espacio de *I*.

Dado que los bloques seleccionados deben poseer la misma dimensión en sus valores de ancho y alto, existen regiones de las sub-imágenes que no son procesadas y que no repercuten en la clasificación del tipo de borde. En la Figura 3 se ilustra esta distribución sobre una imagen, donde las cuadrículas de color morado representan las sub-imágenes, las de color verde los bloques y las franjas de color naranja son las regiones no procesadas.

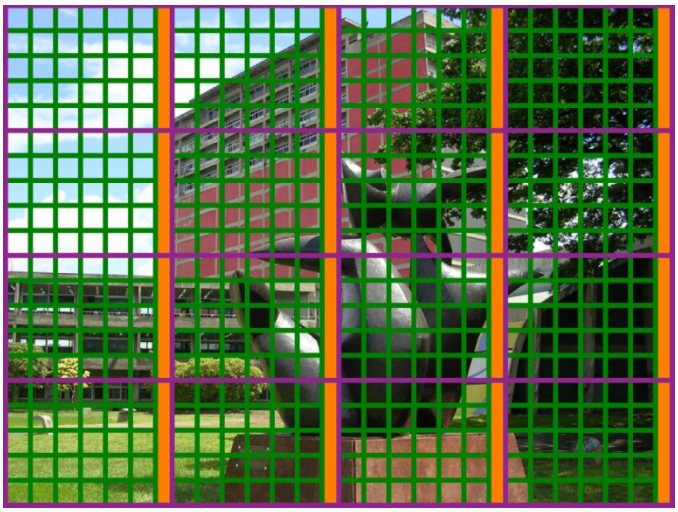


Figura 3: Un ejemplo de Distribución de las Sub-Imágenes, Bloques y Regiones no Procesadas

Con este descriptor, para calcular la distancia entre las imágenes *I* e *I'* se emplean los valores de los 3 histogramas calculados como una diferencia ponderada valor a valor de cada contenedor del LEH, GEH y SGEH.

C. Basado en Regiones

El descriptor de forma basado en regiones (RBSD) representa la distribución de píxeles dentro una imagen *I*. Dado que está basado en los bordes de los objetos y en sus píxeles internos, es posible describir objetos complejos dentro de la imagen que posean múltiples regiones discontinuas, así como objetos simples con o sin agujeros. Este descriptor pertenece a la amplia variedad de técnicas basadas en momentos, haciendo uso de la transformada radial angular (ART) con valores complejos, sobre un disco unitario en coordenadas polares.

Entonces, una imagen *I* expresada en coordenadas polares, se puede representar por un conjunto de valores radiales y angulares. Para cada valor radial se define un conjunto de valores angulares, como se muestra en la Figura 4. A medida que el valor radial aumenta, el número de componentes angulares generadas es mayor.

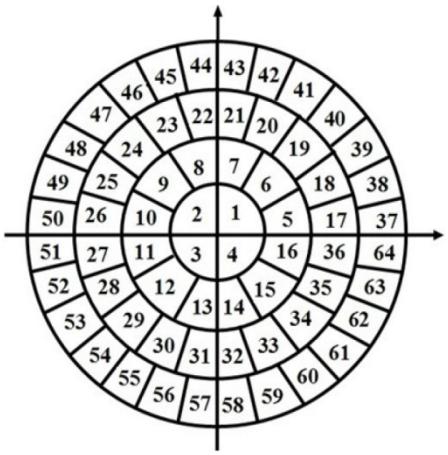


Figura 4: Representación de una Imagen en Coordenadas Polares para la Aplicación del RSBD

La idea consiste en dividir un disco unitario en sectores circulares no solapados y realizar una correspondencia directa a una imagen *I* de tamaño *M x N* (la cual se divide en rectángulos no solapados). Posteriormente, el cálculo de los coeficientes de la ART se realiza mediante (3).

En nuestra implementación paralela de (3), cada conjunto de hilos de la tarjeta gráfica ejecuta un valor angular y radial de las componentes de orden $0 \leq p < 3$ y $0 \leq q < 12$. El conjunto de valores generados se determina por el conjunto de hilos como se muestra en la Figura 5, acumulándolos una vez calculado.

Existen $3 \times 12 = 36$ posibles coeficientes de la ART así como un máximo de $4xN+2$ componentes angulares (donde N representa la dimensión de una imagen cuadrada). Cada conjunto de hilos calcula el aporte a (3) para cada uno de los posibles valores de estas componentes de la ART de orden p y q. Entonces, se requiere el doble de espacio de almacenamiento, $8xN+4$, por cada combinación de valores de p y q, dado que los coeficientes vienen representados por números complejos, almacenando sus componentes real e imaginaria.

Al emplear este descriptor para comparar dos imágenes *I* e *I'*, la similitud entre ambas puede determinarse al calcular la diferencia normalizada de las magnitudes de los coeficientes de la ART de *I* e *I'*.

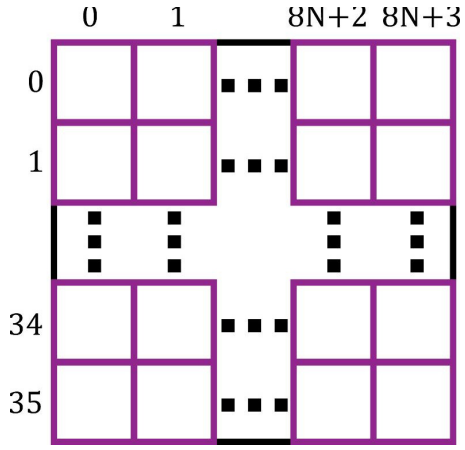


Figura 5: Distribución de los Conjuntos de Hilos para la Obtención del RSBD

Una vez calculados los tres descriptores visuales, se realizaron experimentos para determinar la eficacia y eficiencia de nuestra propuesta en un ambiente paralelo dentro de la tarjeta gráfica.

V. Experimentos y Resultados

Las pruebas realizadas fueron ejecutadas en una PC convencional con procesador *Core 2 Quad*, memoria RAM de 4 GB, y una tarjeta gráfica NVIDIA *GeForce* GT 520 con soporte para CUDA [17] en su versión 5.5. El software empleado fue bajo el sistema operativo *Windows 8* con el compilador de *Visual Studio 2012* y la biblioteca *OpenCV* [18] en su versión 3.0 para la manipulación de las imágenes. El lenguaje de programación CUDA C fue empleado para la paralelización de los descriptores.

Dada la versatilidad de nuestra propuesta, se dividió la realización de las pruebas en cuatro grandes módulos que consisten en la experimentación de cada uno de los descriptores propuestos y un caso de estudio para demostrar la precisión de éstos, empleando fotomosaicos.

A. Descriptor de Colores Dominantes

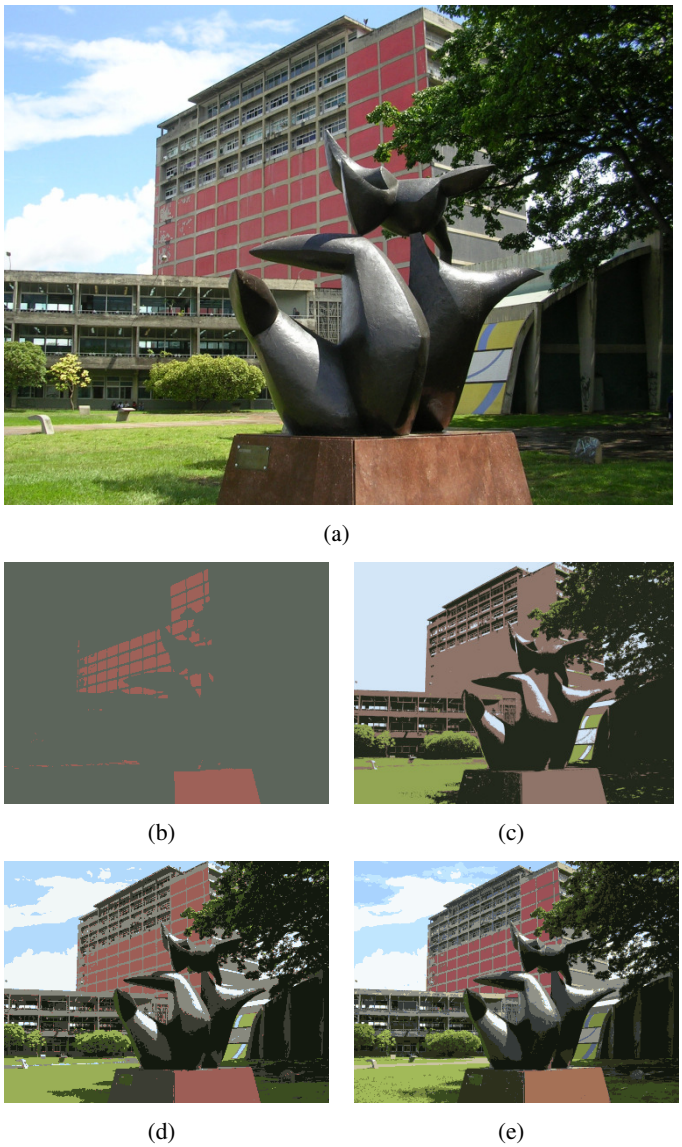
Se ha realizado una comparación entre 3 posibles valores que puede adoptar el parámetro k del algoritmo k-medias utilizado por este descriptor para la cuantización de una imagen de 2048×1536 píxeles. El tamaño de la imagen se debe a la consideración de un número de píxeles significativo para probar la eficiencia de nuestra propuesta (i.e. un total de $3.145.728$ píxeles).

La Tabla I muestra una comparación de los tiempos de ejecución del proceso de cuantización de una imagen para distintos valores de k (de un total de 15 ejecuciones), y el máximo error cuadrático medio porcentual (MSE) entre una imagen y su cuantización.

Tabla I: Tiempos y Error Cuadrático Medio de las Imágenes Cuantizadas

Valor de *k*	Tiempo medido en ms	Error Cuadrático Medio
2	274	79,86 %
4	504	38,09 %
8	1.021	9,16 %
16	1.580	6,67 %

En nuestras pruebas, se considera un valor de $k = 8$ como óptimo dado que dicho número de colores es suficiente para representar la información de color relevante con un error aceptable, como se presenta en la Figura 6. De esta manera, el DCD permite clasificar de forma adecuada una imagen dadas las características de color, con un algoritmo de agrupamiento como *k*-medias empleando 8 colores representativos de toda la imagen.



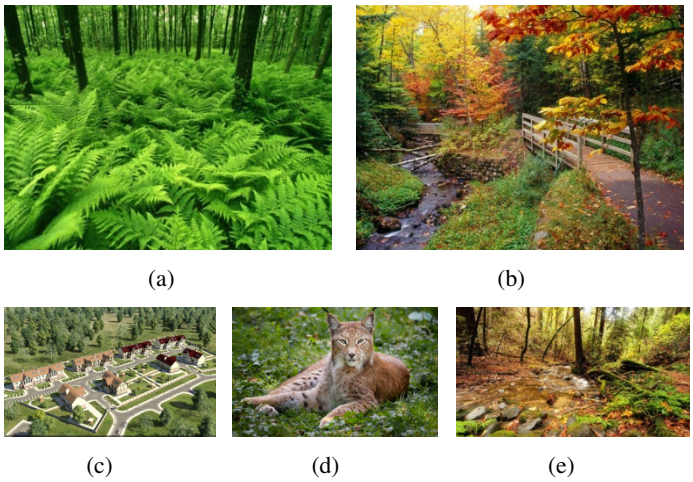
(a) (b) (c) (d) (e)

Figura 6: Diferentes Valores *k* de una: (a) Imagen Original, (b) k = 2, (c) k = 4, (d) k = 8 y (e) k = 16

B. Descriptor de Histograma de Bordes

Para este caso, nuestros experimentos consistieron en demostrar que el GEH y el SGEH incrementan la precisión del descriptor al considerar no solamente información local de bordes sino también patrones presentes en regiones más amplias. Para ello, se construyó una base de datos de 2.340 imágenes heterogéneas para que, dada una imagen original I y una imagen I' de la base de datos, se obtenga una lista de imágenes $I'_1, I'_2,\ldots, I'_k$ que represente los k mejores candidatos (i.e. según la medida de similitud para este descriptor) similares a I.

La Figura 7 muestra los k mejores candidatos para el EHD sin considerar el GEH y el SGEH, con $k = 4$.

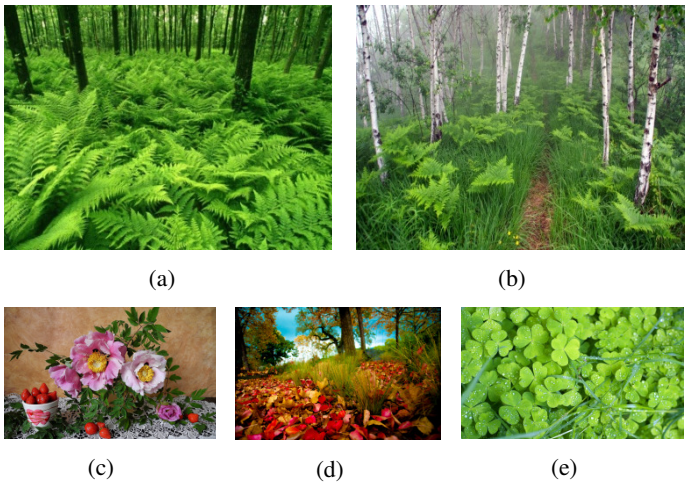


(a) (b) (c) (d) (e)

Figura 7: Aplicación del EHD sin Utilizar el GEH y SGEH sobre una Base de Datos de Imágenes y Como Imagen Base (a), y los Mejores Candidatos Obtenidos, de Mayor a Menor Similitud de (b) a (e)

Por otro lado, la incorporación de los histogramas GEH y SGEH permite obtener mejores candidatos al considerar información no solamente local a los bordes. Como se muestra en la Figura 8, el EHD presenta una mejoría en la similitud visual, considerando las imágenes I'_i de la base de datos.

Nótese que emplear el GEH y SGEH permite obtener detalles globales de la imagen original y determinar la presencia de helechos, troncos de árboles y ubicación de éstos. Las imágenes restantes presentan hojas y tallos asociados a la estructura de bordes de la imagen. Es importante destacar que el color no es considerado por este descriptor, basándose únicamente en los valores de sus intensidades.



(a) (b) (c) (d) (e)

Figura 8: Aplicación del EHD Empleando el GEH y SGEH sobre una Base de Datos de Imágenes y como Imagen Base (a) y los Mejores Candidatos Obtenidos, de Mayor a Menor Similitud de (b) a (e)

El tiempo promedio empleado para el cálculo del EHD asociado a una imagen es de aproximadamente 457 ms, siendo inferior al tiempo empleado por el DCD, dada la independencia

de las tareas que se llevan a cabo, aprovechando de mejor manera el paralelismo provisto por la arquitectura.

C. Descriptor de Forma Basado en Regiones

El EHD junto con el RBSD son los descriptores que permiten explotar en mayor medida el paralelismo provisto por la arquitectura CUDA. En el caso de este último, dado que el cálculo de los coeficientes de orden p y q de la ART es independiente para cada uno.

Para demostrar la efectividad de nuestra propuesta, se construyó una base de datos de 257 imágenes segmentadas a dos colores de marcas, logos y objetos con formas simples, donde se ha evaluado la precisión del RBSD. Tal como se explicó en la sección anterior, se calculan los coeficientes de la ART e igualmente que el EHD, se obtiene una lista de imágenes I'_1, I'_2,…, I'_k que representan los k mejores candidatos similares a una imagen base I.

Es importante destacar que este descriptor funciona de forma eficiente para imágenes con poco ruido ya que es muy sensible a cambios bruscos y continuos de frecuencia en la señal de una imagen. Por ello, se debe realizar un pre-procesamiento de la imagen llevando a cabo un proceso de umbralización (i.e. segmentación a dos tonos).

Otro aspecto a resaltar consiste en el tamaño de las imágenes, el cual no es superior a *100 x 100* píxeles debido a la gran cantidad de cálculos y el tiempo en obtener el resultado. Para tener una medida del tiempo, se realizó una versión secuencial empleando MATLAB para compararla con la versión desarrollada en CUDA C. La proporción en tiempo de ejecución es aproximadamente *38x* más rápida en la versión CUDA C con un tiempo promedio de 247 ms para 20 ejecuciones.

Una implementación secuencial en lenguaje C/C++ es aproximadamente *1.25x* más veloz que su contraparte en MATLAB - bajo las mismas condiciones - siendo igualmente más lenta que una versión desarrollada bajo algún ambiente paralelo como CUDA C.

Los descriptores de manera individual funcionan adecuadamente para un dominio de imágenes con ciertas restricciones de acuerdo a la similitud que se desea lograr (e.g. dimensiones, número de colores, variaciones abruptas y continuas de la frecuencia de la señal, etc.). Así, resulta interesante explorar estos tres descriptores de forma combinada para obtener un resultado adecuado en dominios generales dentro del procesamiento digital de imágenes.

De esta forma, se plantea un caso de estudio basado en fotomosaicos como una operación dentro de un sistema de búsqueda de imágenes de acuerdo a su color, textura y forma (i.e. sistema CBIR – *Content-Based Image Retrieval*).

D. Caso de Estudio: Fotomosaicos

Según Silvers [19], un mosaico se define como una obra compuesta por piezas de madera, piedra, cerámica, vidrio u otro material unidas mediante algún aglomerante, las cuales poseen diversas formas y colores, formando composiciones decorativas. De esta definición se desprende que un fotomosaico es un mosaico donde las piezas que lo componen son imágenes, las cuales se ordenan en una malla de parches sin que se solapen ni existan vacíos entre ellas.

La particularidad de los fotomosaicos radica en que, si existe una corta distancia entre éstos y el punto de vista, pueden apreciarse las imágenes que los componen. A medida que esta distancia aumenta, puede percibirse la silueta de una imagen formada por las imágenes observadas desde una corta distancia.

Para nuestros experimentos, se emplea la misma base de datos de 2.340 imágenes utilizada en las pruebas del EHD. Entonces, para cada imagen de la base de datos se calculan los descriptores de color, textura y forma explicados en nuestra propuesta. Posteriormente, se selecciona una imagen base I_b o imagen a utilizar para generar el fotomosaico, la cual se divide en parches. Para cada parche de I_b se calculan estos descriptores y se busca en la base de datos la imagen que presente mayor similitud con respecto al parche. El objetivo es conseguir y sustituir el candidato óptimo para cada parche de I_b.

Tomando una imagen I_b de tamaño *812 x 812* píxeles, ver Figura 9, con los descriptores normalizados en el rango [0,1] y fijando un número máximo de 100 veces que una imagen candidata puede ser seleccionada como óptima para evitar su continua aparición en la malla de parches, se tiene la Figura 10 que presenta algunos resultados con ciertas modificaciones en los parámetros de entrada.

Primeramente, la selección de I_b para la generación del fotomosaico no es la ideal debido a la gran presencia de colores en degradación sobre una misma región. Sin embargo, nuestro objetivo es verificar la eficacia de los descriptores de color, textura y forma. Los fotomosaicos generados que se observan en la Figura 10 presentan una gran cantidad de ruido debido a la elevada diferencia visual entre cada parche y sus adyacentes. Con una colección homogénea de imágenes se permitirá la atenuación de este ruido, ya que se dispone de mayor cantidad de imágenes similares que pueden ser seleccionadas por los descriptores visuales para ser sustituidas en la malla de parches definida sobre I_b.

Los fotomosaicos de la Figura 10a y Figura 10c han sido generados estableciendo la misma ponderación para cada uno de los descriptores visuales. Estos capturan la silueta básica de la imagen pero dada la heterogeneidad de las imágenes presentes en la base de datos (lo cual se ve reflejado en la diversidad de texturas y formas presentes), degradan el rendimiento del EHD y el RBSD.

Por otro lado, los fotomosaicos de la Figura 10b y Figura 10d han sido generados estableciendo una ponderación total para el DCD, ignorando la información dada por el EHD y el RBSD. Estos fotomosaicos presentan una mejora sustancial en su calidad visual, especialmente el fotomosaico de la Figura 10d con respecto a la Figura 10c, presentándose el ruido de forma más homogénea en toda la imagen. Esto muestra que la sensibilidad del DCD es menor que los otros descriptores visuales considerados (para el caso de bases de datos de imágenes heterogéneas).

Tabla II: Parámetros Empleados para la Generación de los Fotomosaicos

Figura	Número de parches	Tamaño de parches	Tamaño del fotomosaico	Ponderación *DCD/EHD/RBDS*
10ª	50 x 50 = 2.500	100 x 100 píxeles	5.000 x 5.000 píxeles ~71.5Mb	0.34 / 0.33 / 0.33
10b	50 x 50 = 2.500	100 x 100 píxeles	5.000 x 5.000 píxeles ~71.5Mb	1.00 / 0.00 / 0.00
10c	100 x 100 = 10.000	100 x 100 píxeles	10.000 x 10.000 píxeles ~286Mb	0.34 / 0.33 / 0.33
10d	100 x 100 = 10.000	100 x 100 píxeles	10.000 x 10.000 píxeles ~286Mb	1.00 / 0.00 / 0.00

Figura 9: Imagen Base I_b Empleada para la Generación de Fotomosaicos

Por otro lado, los fotomosaicos de la Figura 10b y Figura 10d han sido generados estableciendo una ponderación total para el DCD, ignorando la información dada por el EHD y el RBSD. Estos fotomosaicos presentan una mejora sustancial en su calidad visual, especialmente el fotomosaico de la Figura 10d con respecto a la Figura 10c, presentándose el ruido de forma más homogénea en toda la imagen. Esto muestra que la sensibilidad del DCD es menor que los otros descriptores visuales considerados (para el caso de bases de datos de imágenes heterogéneas).

En la Tabla II se presentan las características empleadas entre la imagen I_b y los fotomosaicos generados de la Figura 10.

VI. Conclusiones y Trabajos Futuros

En esta investigación se ha mostrado una versión de descriptores visuales del estándar MPEG-7 haciendo uso de la GPU para acelerar los procesos involucrados, manteniendo la eficacia de éstos con gran nivel de precisión en imágenes similares.

En nuestra experimentación se consideró cada uno de los descriptores de manera individual y su evaluación con una base de datos de imágenes bajo un esquema paralelo enfocado en la arquitectura CUDA. Así, cada descriptor permite identificar imágenes basado en un criterio de búsqueda para diversas aplicaciones.

Por otro lado, seleccionar como caso de estudio la generación de fotomosaicos basados en sistemas de tipo CBIR permite simular el uso de vectores característicos, tal como ocurre en los motores de búsqueda convencionales, para ubicar de forma rápida un conjunto de candidatos basados en una implementación paralela en la GPU de descriptores visuales del estándar MPEG-7. Es importante destacar, que el uso de las GPUs para acelerar el tiempo de cómputo se considera una rama de investigación actual debido a sus bajos costos, fácil uso y escalabilidad en las aplicaciones computacionales.

La calidad visual de los fotomosaicos depende enteramente de la combinación de los descriptores visuales. Esta calidad puede verse incrementada al realizar una selección de una base de datos de imágenes apropiada que sea lo suficientemente representativa en cuanto a las necesidades de una aplicación de esta naturaleza. Esta clase de aplicaciones es sumamente sensible a la base de datos de imágenes utilizada, por lo que es ideal que su contenido sea lo más homogéneo posible, tal que no se generen cambios visuales drásticos entre los parches de los fotomosaicos.

En el futuro se plantea la idea de implementar y probar otros descriptores visuales del estándar con el fin de agregar una mayor cantidad de parámetros en aplicaciones, en el contexto de búsquedas de tipo CBIR y realizar pruebas visuales basadas en las funciones de similitud definidas. Igualmente, se propone estudiar con más detalle los impactos al emplear un tipo de descriptor visual de acuerdo a ciertos criterios de búsqueda dentro de aplicaciones especializadas, con el fin de precisar el descriptor y sus parámetros de acuerdo a la naturaleza de dicha búsqueda (i.e. tomando el cuenta si se desea buscar por similitud de tono, forma, relieve, entre otros).

Referencias

[1] Y. Rui, T.S. Huang, and S.F. Chang, *Image Retrieval: Current Techniques, Promising Directions, and Open Issues*, Journal of Visual Communications and Image Representation, vol. 10, pp. 39–62, 1999.

[2] A.W. Smeulders, S. Member, M. Worring, S. Santini, A. Gupta, and R. Jain, *Content-Based Image Retrieval at the End of the Early Years*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pp. 1349–1380, 2000.

[3] R. Datta, D. Joshi, J. Li, and J.Z. Wang, *Image Retrieval: Ideas, Influences, and Trends of the New Age*, ACM Computing Surveys, vol. 40, no. 2, pp. 5:1–5:60, Abril 2008.

[4] M. Angelides and H. Aguis, *The Handbook of MPEG Applications: Standards in Practice*, 1st edition, John Wiley & Sons, 2011.

[5] J. Orallo, M. Ramírez y C. Ferri, *Introducción a la Minería de Datos*, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Pearson Prentice Hall, 2008.

[6] S. Sergyán, *Precision Improvement of Content-Based Image Retrieval Using Dominant Color Histogram Descriptor*, in proceedings of 1st WSEAS International Conference on Image Processing and Pattern Recognition at Budapest, pp. 197–203, Hungary, December 2013.

[7] V. Felipe y E. Ramírez, *K-Medias Empleando la GPU*, en las memorias del III Simposio Científico y Tecnológico en Computación (SCTC), Sesión de Posters, pp. 152, 2014.

[8] D. Park, Y. Jeon, and C. Won, *Efficient Use of Local Edge Histogram Descriptor*, in proceedings of the ACM Workshops on Multimedia, pp. 51–54, 2000.

[9] K. Hosny, *Accurate Computation of ART Coefficients for Binary Images*, in proceeding of the 1st Taibah University International Conference on Computing and Information Technology, vol. 1, 2012.

[10] Y. Xin, M. Pawlak, and S. Liao, *Accurate Computation of Zernike Moments in Polar Coordinates*, IEEE Transactions on Image Processing, vol. 16, pp. 581–587, 2007.

[11] M. Slomp, M. Mikamo, B. Raytchev, T. Tamaki, and K. Kaneda, *GPU-Based SoftAssign for Maximizing Image Utilization in Photomosaics*, International Journal of Networking and Computing, vol. 1, no. 2, pp. 211–229, 2011.

[12] G. Di Blasi, G. Gallo, and M. Petralia, *Smart Ideas for Photomosaic Rendering*, in proceedings of the Eurographics Italian Chapter, pp. 267–271, 2006.

[13] J. Kim and F. Pellacini, *Jigsaw Image Mosaics*, in proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques SIGGRAPH, pp. 657–664, 2002.

[14] Google Inc. *Google Goggles*, http://www.google.com/mobile/goggles.

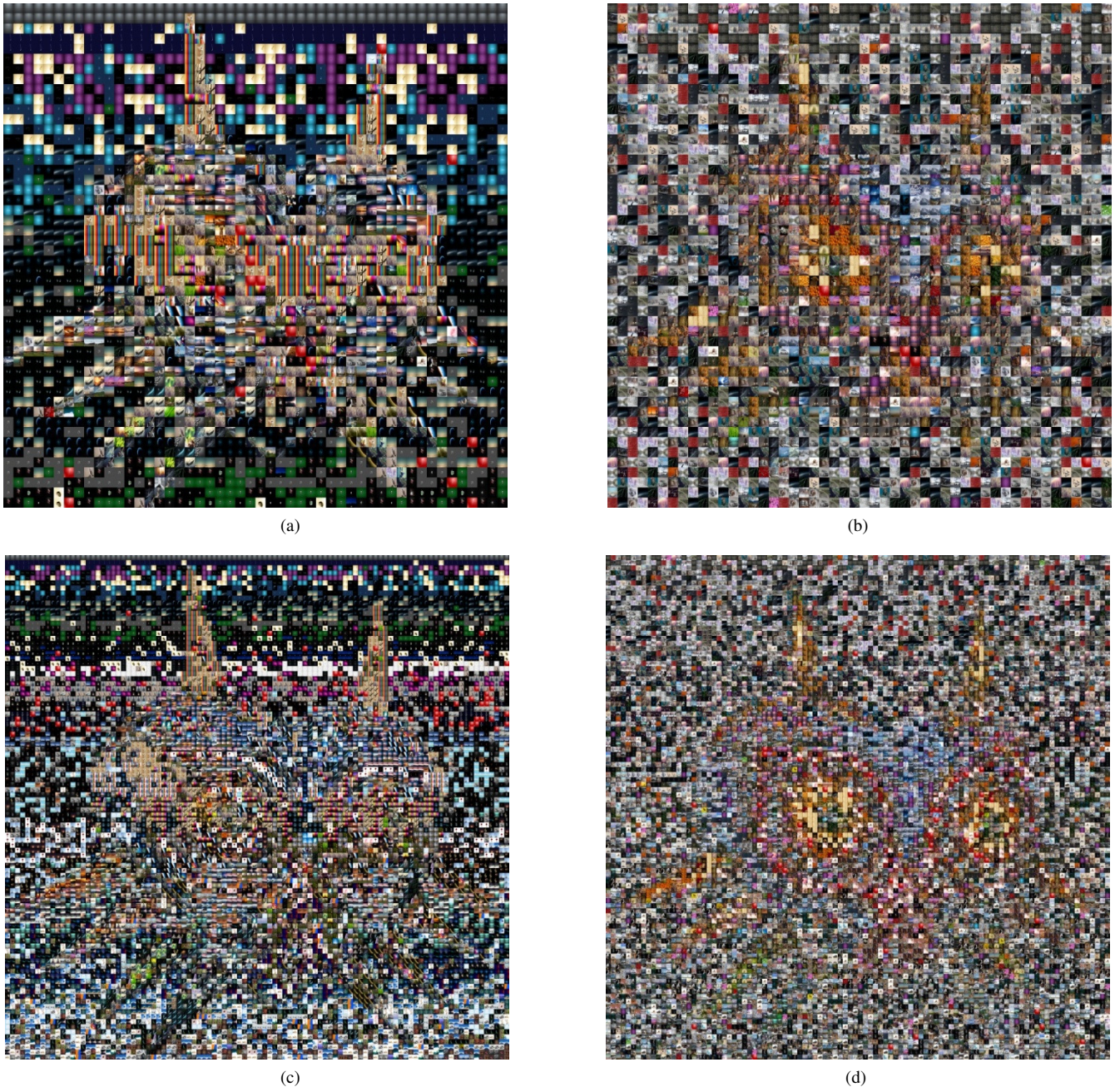
[15] Image Searcher Inc. *CamFind*, http://www.camfindapp.com.

[16] J. Ricard, D. Coeurjolly, and A. Baskurt, *Generalization of Angular Radial Transform*, in proceedings of the International Conference on Image Processing, vol. 4, pp. 2211–2214, October 2004.

[17] NVIDIA Corporation. *CUDA*, https://developer.nvidia.com/cuda-zone.

[18] Itseez. *OpenCV*, http://opencv.org.

[19] R. Silvers, *Photomosaics: Putting Pictures in Their Place*, M.S. thesis, Program in Media Arts & Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1996.



(a) (b)

(c) (d)

Figura 10: Fotomosaicos Generados con Diversos Parámetros en el Número de Parches y en la Ponderación de los Descriptores de Color, Textura y Forma. Con parches de 50 x 50 (a y b) y de 100 x 100 (c y d)

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 81-92, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Balance de Carga y Tolerancia a Fallas en OPTIMUS (OPTImized Mail distribUtion System)

Fabiola Rosato[1], Javier Argüello[1], Yudith Cardinale[1]
rosato.fcrm@gmail.com, javier@ldc.usb.ve, yudith@ldc.usb.ve

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

Resumen: En este trabajo presentamos la implementación de una Plataforma Distribuida de envío masivo de correos electrónicos, llamada OPTIMUS (OPTImized Mail distribUtion System), que permite distribuir eficientemente las cargas de trabajo entre múltiples servidores de correos, optimizar el uso de los recursos disponibles, garantizar tolerancia a fallas, obteniendo como resultado una robusta plataforma de envío. La arquitectura de OPTIMUS extiende la implementación de servidores Postfix para incorporar el manejo de base de datos, balance de carga y tolerancia a fallas, y está conformada por múltiples servidores Postfix extendidos, conectados en una plataforma Peer-to-Peer. Los resultados experimentales muestran dos aspectos importantes: (i) OPTIMUS supera notablemente el desempeño de Postfix en su versión original centralizada, para cantidades significativamente grandes de correos electrónicos, y (ii) los mecanismos de tolerancia a fallas de OPTIMUS, no generan una degradación en el desempeño global del sistema, durante su actividad normal (sin fallas) y realizan una recuperación totalmente efectiva si ocurren fallas.

Palabras Clave: Correo Electrónico; Postfix; Sistemas P2P; Distribución Masiva; Balance de Carga; Tolerancia a Fallas.

Abstract: In this work, we present the implementation of a Distributed Platform for Massively Emails distribution, called OPTIMUS (OPTImized Mail distribUtion System). It allows efficient workload distribution among multiple email servers, optimizes the use of available resources, and provides fault tolerance in a robust distribution platform. OPTIMUS architecture extends the Postfix server implementation to incorporate data base management, workload balance, and fault tolerance. It is composed by several extended Postfix servers, connected in a Peer-to-Peer platform. Experimental results show that with a few amount of emails, Postfix (with its original centralized implementation) and OPTIMUS have similar performance. However, with huge amount of emails, OPTIMUS highly overcomes the performance of the original centralized Postfix version.

Keywords: E-Mail; Postfix; P2P Systems; Masive Distribution; Load Balancing; Fault Tolerance.

I. Introducción

La necesidad de tener correo electrónico surgió en los años sesenta, a partir del momento en que las personas comenzaron a tener acceso compartido a una misma computadora. En aquel entonces, el correo electrónico se limitaba al intercambio de mensajes entre usuarios de una misma máquina. Los primeros correos electrónicos se enviaron alrededor de 1970, a través del predecesor del Internet que conocemos hoy en día, ARPANET. En aquel entonces, la entrega de los mensajes era un proceso relativamente sencillo y consistía en mover archivos entre un servidor y otro que atendían a muchos usuarios [1].

Con el paso de los años y con la rápida evolución del Internet, el propósito del "e-mail" cobró un significado distinto: comunicar a las personas sin importar la distancia entre ellas. Eventualmente el correo electrónico cobró gran importancia en la comunicación, de manera particular con el nacimiento e incorporación de los dispositivos móviles [2]. Por ejemplo, en el ámbito del mercadeo, ha conseguido un papel trascendental, lo cual ha generado un incremento sustancial en la utilización del correo electrónico con fines publicitarios [3].

Actualmente el volumen de correos diarios se calcula en más de 190 mil millones [4]; por esta razón este medio de comunicación es uno de los más influyentes en la sociedad. A raíz de esto, aparecen las compañías de envíos de correo masivo, que en su mayoría siguen un enfoque centralizado y requieren de servidores y otros equipos que incurren en gastos, tales como tiempo, consumo de CPU, consumo de luz, etc.

La tendencia de la evolución de computación en la última década deja de lado los sistemas centralizados y da paso a los sistemas distribuidos masivamente. Estos sistemas masivos

se caracterizan por interconectar una gran cantidad de nodos, incluso millones de nodos; creando la necesidad de plantear nuevas formas de comunicación que incluyen la comunicación uno-a-muchos y que supere al tradicional estilo uno-a-uno [5]. La distribución masiva de correos electrónicos no escapa a esta tendencia.

Hoy en día existe software de distribución masiva de correos disponible y de código abierto, tales como Sendmail [6] y Postfix [7], que permiten el envío masivo de correos mediante el uso de múltiples servidores. Sin embargo, existen diferentes factores que pueden influir en su rendimiento, tales como las características físicas de cada servidor, la velocidad de la red o el ancho de banda que poseen. La gran mayoría de los agentes de correo electrónico manejan sus colas en disco en archivos planos, lo cual puede provocar gran actividad de entrada/salida (I/O) que en un sistema masivo puede ser crítico, pues representa un gran consumo de recursos. Otro aspecto relevante es la asignación de correos a ser enviados entre los servidores disponibles, ya que si éstos no siguen una política adecuada de balance de carga, pueden desperdiciar el uso de los recursos. Así mismo el aspecto de tolerancia a fallas no es ampliamente considerado en estas aplicaciones clásicas de distribución masiva de correos electrónicos.

Se estima que una arquitectura distribuida puede minimizar estas limitaciones a través de un algoritmo eficiente de distribución de correos entre los servidores, que tome en cuenta información del contexto de ejecución (carga de los servidores, localización física de los servidores y de los clientes, etc.) para decidir en un determinado instante cuáles servidores conviene utilizar para una tarea específica de envío masivo a una cantidad determinada de clientes, esto se traduce en la implementación de un enfoque de balance de carga eficiente. Por otro lado, si se mejora el manejo de las colas de mensajes con el uso de una base de datos (en lugar de archivos planos), se considera que se podría reducir la actividad de entrada y salida, y por lo tanto minimizar el consumo de recursos. Además, la implementación de mecanismos de tolerancia a fallas brindarían confiabilidad y alta disponibilidad de los servicios de distribución masiva de correos electrónicos.

En este contexto, este trabajo plantea una arquitectura distribuida de envío masivo de correos electrónicos, basada en la extensión de servidores Postfix, para capacitar el envío de correos almacenados en una base de datos y balancear la carga entre los servidores, de manera de maximizar el envío de los correos y minimizar los costos, además de ofrecer mecanismos de tolerancia a fallas. Llamamos a esta nueva arquitectura OPTIMUS (OPTImized Mail distribUtion System). En un trabajo previo presentamos la arquitectura general de OPTIMUS, conformada por múltiples servidores Postfix extendidos, conectados en una plataforma Peer-to-Peer (P2P), considerando la incorporación de una base de datos y un algoritmo de balance de carga [8]. En este artículo detallamos la implementación de la primera versión de la plataforma e incorporamos los mecanismos de tolerancia a fallas con la finalidad de proveer una plataforma robusta que optimice el envío masivo de correos electrónicos. Los resultados experimentales muestran que para cantidades relativamente pequeñas de mensajes electrónicos, tanto Postfix (en su versión original centralizada) como OPTIMUS tienen aproximadamente el mismo desempeño. Sin embargo, para cantidades significativamente grandes, OPTIMUS supera notablemente el desempeño de la versión original de Postfix. Además, también mostramos experimentalmente que los mecanismos de tolerancia a fallas de OPTIMUS, no generan una degradación en el desempeño global del sistema durante su actividad normal (sin fallas) y que realizan una recuperación totalmente efectiva si ocurren fallas.

La presentación de nuestro trabajo está organizada como sigue. En la Sección II se describe el entorno de las plataformas de distribución de correos electrónicos, incluyendo la descripción detallada de la arquitectura de Postfix. La arquitectura de OPTIMUS se presenta en la Sección III. Los resultados experimentales se muestran en la Sección IV. La Sección V expone las principales diferencias entre OPTIMUS y Postfix original. Finalmente, la Sección VI presenta las conclusiones y el trabajo que planeamos continuar realizando.

II. Plataformas de Distribución de Correos Electrónicos

El correo electrónico surgió a principios de los años 70 y se utilizó por primera vez en ARPANET. Posteriormente, ARPANET fue sustituido por Internet, y con la rápida evolución de éste, el correo electrónico adoptó un rol predominante como medio de comunicación. Rápidamente, el correo electrónico se convirtió en un medio de comunicación dominante. Con la evolución de Internet, la redes se hicieron cada vez más complejas y empezaron a necesitar herramientas que facilitaran el intercambio y envío de archivos entre servidores y capaces de tratar la extensa gama nueva de servicios de correo [9].

Uno de los primeros programas utilizados para el intercambio de correo electrónico fue el paquete Sendmail [10], el cual tenía como objetivo lidiar con la diversidad de sistemas de correo. Inmediatamente, este servidor se convirtió en un programa que jugaba un rol predominante en Internet. Sendmail utiliza el protocolo SMTP y sigue siendo uno de los servidores más usados actualmente. Sin embargo, la estructura monolítica de su arquitectura se ha convertido en un factor que ha degradado su reputación y uso, ya que es propenso a numerosas fallas de seguridad y su configuración y mantenimiento pueden llegar a ser muy complicados.

Para sobreponer las limitaciones de Sendmail, surge Postfix como una alternativa segura y flexible [11][12]. Postfix fue desarrollado en los laboratorios de “IBM research”, por Wietse Venema, creador de software como SATAN y TCPwrappers, como reemplazo de Sendmail, que es considerado un sistema de envío de correo inseguro. Este agente de transporte de correo fue desarrollado a finales de los 90 y en un principio se llamó “IBM Secure Mailer”, hasta que se autorizó su distribución como una licencia de código abierto (*open source*) y se renombró como “Postfix”. La creación de Postfix fue impulsada por objetivos como confiabilidad, seguridad, desempeño,

flexibilidad, usabilidad y compatibilidad con Sendmail.

Las siguientes secciones describen la arquitectura, composición general del servicio de correo electrónico y los principales protocolos usados.

A. Componentes del Servicio de Correo

El correo electrónico está basado en una serie de estándares y protocolos que definen cómo son redactados y transferidos los mensajes. Dentro de este proceso intervienen distintos procesos que en conjunto llevan a cabo el envío del correo. Los principales componentes de un servicio de correo electrónico son:

- Agente de Usuario (MUA, por sus siglas en inglés de *Mail User Agent*), es el software que permite a los usuario leer y redactar los mensajes. No participan en el proceso de envío del correo.
- Agente de Entrega (MDA, por sus siglas en inglés de *Mail Delivery Agent*), tiene la tarea de almacenar el mensaje en un archivo o almacenarlo en alguna base de datos especializada para correos.
- Agente de Transporte (MTA, por sus siglas en inglés de *Mail Transport Agent*), es el encargado de determinar si el destino del mensaje es local o remoto. De ser local, el MTA entrega dicho mensaje al MDA, y en caso contrario, lo envía a otro Agente de Transporte ubicado en otro sistema. Si un mensaje no puede ser entregado, este agente es el encargado de devolver el mensaje al emisor original o notificar al administrador del sistema.

La interacción de estos componentes se muestra en la Figura 1. Para que los agentes de envío de correo electrónico puedan comunicarse entre sí, se estandarizó el intercambio de mensajes y de esta forma, diferentes programas pueden interoperar sin importar quién los desarrolle, siempre y cuando implementen uno de los protocolos: *Simple Mail Transfer Protocol (SMTP)* [13], *Extended Simple Mail Transfer Protocol (ESMTP)* [14] o *Local Mail Transfer Protocol (LMTP)* [15]. Tanto Sendmail, como Postfix pueden utilizar cualquiera de estos tres protocolos.

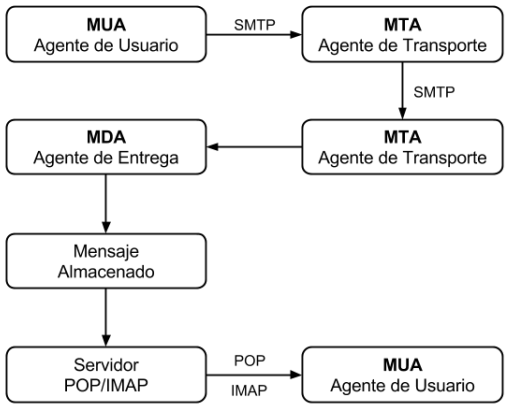


Figura 1: Flujo de un Mensaje [12]

B. Arquitectura de Postfix

1) ***Subdirectorios de las Colas de Postfix:*** Postfix tiene estructuras de tipo cola donde se almacenan los correos como archivos separados, utilizando identificadores únicos para cada nombre de archivo. Cada cola de mensajes es creada como un subdirectorio. Estos subdirectorios se presentan a continuación.

- **active**: Contiene los mensajes que están siendo procesados por el manejador de colas de Postfix.
- **bounce**: Contiene el registro de las causas por las cuales el mensaje fue rebotado.
- **corrupt**: Contiene los mensajes que no están en un formato adecuado.
- **defer**: Contiene los registros de las causas por las que ha sido demorado un envío, para que más adelante Postfix intente enviarlo de nuevo.
- **deferred**: Contiene los mensajes demorados que Postfix intentará enviar de nuevo.
- **incoming**: Contiene mensajes nuevos, recibidos de *hosts* SMTP remotos.
- **maildrop**: Contiene los mensajes entregados por el comando *sendmail* que están esperando ser procesados por Postfix.

2) ***Componentes:*** A diferencia de Sendmail, Postfix divide y asigna sus tareas a distintos programas individuales. La mayoría de los programas son demonios y son invocados por el *master daemon* quien es el primero en iniciar. Estos programas son los siguientes:

- **Bounce**: se encarga de devolver los correos rebotados a su emisor original. También es el encargado de diferir el correo dirigido a un *host* no disponible.
- **Cleanup**: procesa los encabezados del correo entrante y lo coloca en la cola *incoming*.
- **Qmgr**: procesa los mensajes en la cola *incoming*, determina dónde y cómo deben ser enviados e invoca al programa responsable de la entrega.
- **Error**: procesa el mensaje proveniente del *qmgr* y lo obliga a rebotar.
- **Local**: entrega el correo destinado a usuarios locales.
- **Pickup**: espera por mensajes en la cola *maildrop* y los envía al programa *cleanup* para que los procese.
- **Pipe**: reenvía el mensaje del *qmgr* a un programa externo.
- **Postdrop**: mueve los mensajes entrantes a la cola *maildrop*, cuando los usuarios normales no pueden escribir en ella.
- **Showq**: reporta los estatus de las colas de Postfix.
- **Smtp**: envía el mensaje a *hosts* externos utilizando el protocolo SMTP.
- **Smtpd**: servidor SMTP que recibe los correos provenientes de *hosts* remotos.
- **Trivial-rewrite**: recibe los mensajes del programa *cleanup* y se asegura que las direcciones de los encabezados cumplan el formato estándar.

3) ***Limitaciones:*** Uno de los problemas de Postfix al realizar envíos masivos, es que la cola de los correos entrantes, o "maildrop", está implementada como un subdirectorio con un conjunto de archivos que representan los correos, lo que

significa que al momento de escribir una gran cantidad de correos en la cola, se generará una gran actividad de entrada y salida. Por otro lado, se corre el riesgo de saturar la memoria provocando un desbordamiento de la cola. El otro problema radica en que no se podría tener, por ejemplo, un servidor a dedicación exclusiva para almacenar los correos y otro que se encargue de enviarlos; la arquitectura de Postfix no está diseñada para tales fines, de forma que todo tiene que estar centralizado en un único servidor, lo que disminuye la capacidad para aprovechar al máximo los recursos de cada máquina disponible.

En el caso en que Postfix, por alguna razón dejase de enviar correos, requeriría de un reinicio manual ya que no hay una reasignación automática que permita que los correos sean entregados de alguna otra manera alternativa. Las razones por las cuales Postfix puede detener su ejecución son fallas críticas del sistema, tales como corrupción del disco, problemas de memoria u otras.

A partir de esta problemática, una de las modificaciones que decidimos realizar a la implementación original de Postfix, fue separar su código en dos módulos: uno que se encargue de recibir y almacenar los correos entrantes por enviar y otro que se dedique exclusivamente al envío de dichos correos. Además se propone la utilización de una Base de Datos como representación de la cola de correos entrantes, para reducir el posible *overhead* de entrada y salida y con el fin de facilitar la distribución de la carga de trabajo entre los servidores.

- Minimizar la recuperación manual de fallas.
- Garantizar el 100% de los envíos.
- Garantizar el mantenimiento de la plataforma ante una falla, sin incurrir en un impacto negativo del rendimiento.

III. OPTIMUS: Plataforma de Distribución Masiva de Correos Electrónicos

En el marco del auge de los sistemas distribuidos, este trabajo propone una adaptación de los servidores de envío masivo de correo electrónico a una arquitectura distribuida que, gracias a los avances tecnológicos en esta materia, permita superar así las limitaciones actuales de desempeño, tolerancia a fallas y seguridad de las que aún adolecen los servidores de correo en la actualidad para su uso a gran escala.

Con la finalidad de demostrar la factibilidad de la implementación distribuida de una plataforma de envío masivo de correos electrónicos, se evaluaron los dos servidores más populares en la actualidad y de código abierto: Sendmail y Postfix.

Sendmail tiene la ventaja de presentar una gran versatilidad para su configuración, sin embargo su código es completamente monolítico, difícil de comprender y mucho más de extender con los cambios necesarios para ajustarlo a las nuevas necesidades. Por otro lado, Postfix, además de ser robusto, presenta un código modular, donde cada funcionalidad está impecablemente definida en un módulo separado y muy apegado a las definiciones teóricas, lo cual facilita su comprensión y su extensión. Por lo anterior, se seleccionó Postfix como el servidor de correo a utilizar para adaptarlo a la plataforma planteada. Hasta donde pudimos verificar y donde alcanza nuestro conocimiento, no existe un software de distribución masiva de correos electrónicos como el propuesto en este trabajo.

Las próximas subsecciones presentan la descripción detallada de la arquitectura de OPTIMUS, que comprende una base de datos NoSQL, una arquitectura basada en agentes P2P que facilita la implementación del balance de carga y estrategias de tolerancia a fallas.

A. Adaptaciones a Postfix: Incorporación de una Base de Datos NoSQL

Con la finalidad de facilitar la distribuición y asignación de los correos electrónicos, se reemplazó el manejo de dos de las principales colas que utiliza Postfix en su versión original. Se decidió utlizar una base de datos que gestione y centralice las colas de correos entrantes y diferidos para cualquier configuración posible de la plataforma. A continuación se especifican los cambios realizados a Postfix:

- **maildrop**: En la versión original de Postfix, está implementado como un directorio, en donde cada correo se representa como un archivo. En cambio, en la versión modificada de OPTIMUS, se migró el *maildrop* a una Base de Datos NoSQL para centralizar la cola, en caso de haber múltiples servidores Postfix, en una sola Base de Datos.
- **sendmail & postdrop**: En Postfix, estas dos funciones se encuentran en archivos distintos y entre los dos se encargan de escribir en el *maildrop*, con el formato pertinente, los correos a ser enviados. En la versión adaptada a OPTIMUS, se unificaron estos archivos en uno solo que inserte en la Base de Datos el correo especificado, conservando el formato original que utiliza Postfix.
- **pickup**: Originalmente, el *pickup* escanea constantemente el *maildrop*, revisa si hay correos por enviar y se dispone a mandar cada uno de los mensajes que allí se encuentran. Por otro lado, con las modificaciones realizadas para OPTIMUS, el *pickup* recibe a través de una comunicación vía *sockets* con un agente de correos, al que se le denominó *mail agent*, los identificadores de los correos que le corresponde mandar, que se encuentran en una Base de Datos local, luego selecciona cada correo y lo envía. Esto convierte al *pickup* modificado en un servidor que escucha constantemente dichas peticiones.
- **deferred**: Similar a los cambios realizados para simular la cola *maildrop*.

OPTIMUS utilza una base de datos conocida como *Redis* que funciona bajo el paradigma NoSQL [16]. Es un servidor que mantiene en memoria estructuras de datos del tipo clave-valor, lo que proporciona un gran desempeño y a su vez puede ser usada como una base de datos persistente. Normalmente, las base de datos NoSQL son recomendadas para aplicaciones que se benefician del esquema libre que proporciona este paradigma y que requieren un gran desempeño y escalabilidad. OPTIMUS tiene la particularidad de ser una plataforma con un gran número de operaciones de escritura por lo que *Redis*

es el candidato ideal para almacenar los correos electrónicos que serán procesados para su envío.

Como se observa en la Figura 2, la arquitectura original de Postfix no puede separarse en varios módulos que se ejecuten en distintos servidores, mientras que con los cambios realizados, es posible separar las funciones del *sendmail* y del *postdrop* originales, para que escriban en una Base de Datos que puede ser remota, mientras que a partir del *pickup* en adelante se encarguen, desde otro posible servidor, de enviar los correos. Otro aspecto importante a resaltar es la utilización de dos niveles de Bases de Datos NoSQL que mantienen en memoria principal la información esencial para el envío de los correos. El primer nivel es el remoto, que consta de una visión global y contiene la información acerca de todos los correos de la plataforma, facilitando su distribución. El segundo nivel es local, conformado por una Base de Datos local que dispondrá para cada servidor de correo, los mensajes delegados a éstos. Esta decisión de diseño se tomó en aras de minimizar las conexiones a la Base de Datos global que podría generar un *overhead* elevado o incluso llegar al límite de conexiones. De esta forma, se filtra la cantidad de accesos a la Base de Datos global por medio de los agentes *Scheduler* e *Info*, cuyas funcionalidades serán explicadas en la próxima sección.

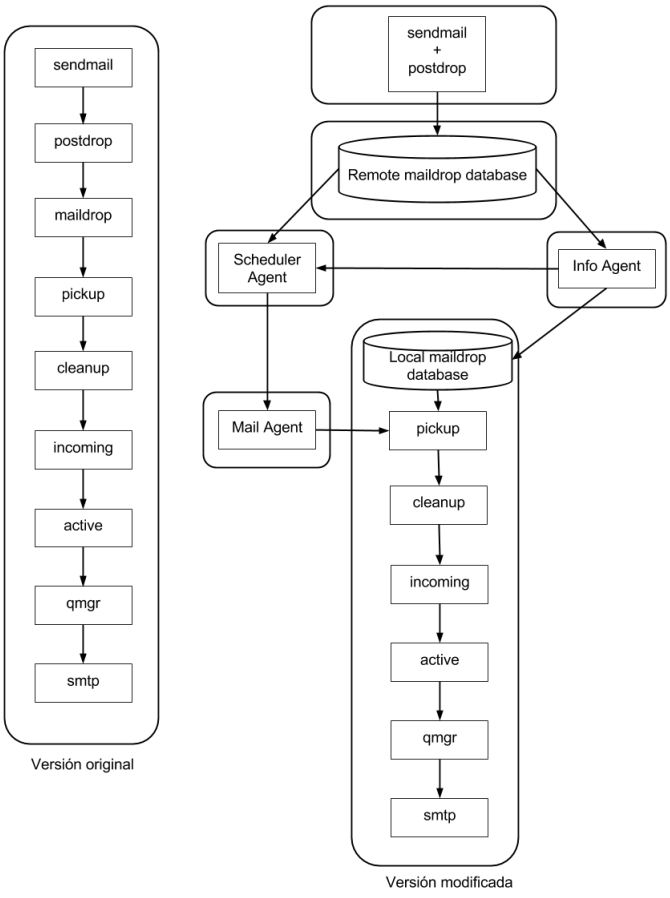


Figura 2: Flujo Original vs Flujo Modificado

B. Arquitectura P2P Multiagentes y Balance de Carga

La arquitectura de OPTIMUS se basa en capas (ver Figura 3), las cuales están representadas por tres agentes que se valen de las Bases de Datos NoSQL para llevar a cabo el envío de los correos:

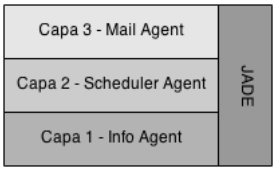


Figura 3: Arquitectura de OPTIMUS Basado en Capas

1) ***Capa 3 - Mail Agent****:* Agente encargado de contactar al demonio *pickup* de Postfix y enviarle los identificadores de los correos a ser procesados. Este agente recibe una notificación del *Scheduler Agent* la cual le indica qué lote de correo debe procesar. Luego, se dispone a enviar a Postfix todos los correos de ese lote. Todo *Mail Agent* está asociado a un único servidor Postfix. Esta capa no ofrece balance de cargas, ya que es quien está en contacto directo con el servidor Postfix y por lo tanto, el último eslabón de la cadena de envío; no obstante, los *Mail Agents* presentan rutinas de recuperación de fallas y se comunican con su base de datos local, en donde están representadas las colas de Postfix para almacenar los correos que le corresponden para su posterior envío.

2) ***Capa 2 - Scheduler Agent****:* Este agente es responsable de notificar y supervisar el envío de correo de uno o varios *Mail Agents* asociados. Recibe una notificación del *Info Agent* para indicarle la presencia de nuevos correos en la plataforma y posteriormente le notifica a los *Mail Agents* asociados si tienen correos por enviar, indicándoles el lote de mensajes respectivo. Luego monitorea los envíos y en caso de haber alguna demora en alguno de sus servidores de correo, redistribuye la carga. Los *Scheduler Agents* presentan tres características importantes: se encargan de balancear la carga al limitar el número de *Mail Agents* que pueden monitorear, de acuerdo a la cantidad de *Schedulers* y *Mail Agents* registrados en la plataforma. Por otro lado, se comunican con las bases de datos locales de sus *Mail Agents* para insertar los correos nuevos que deben enviar, de acuerdo a la información que obtengan de la base de datos global. Por último, presentan mecanismos de tolerancia a fallas, las cuales se detallarán más adelante.

3) ***Capa 1 - Info Agent****:* Este agente monitorea la base de datos global para determinar cuándo hay lotes de correos nuevos en la plataforma, enumerados por el *Sendmail*, según el orden de envío, y distribuye la carga de los correos por enviar entre los *Mail Agents* disponibles. Una vez distribuida la carga, notifica a los distintos *Scheduler Agents* que hay nuevos correos por procesar. Los *Info Agents* se enumeran de acuerdo al orden en el que inicien su ejecución, de manera que procese los lotes que sean múltiplos de su identificador, tal como se muestra en la Figura 4. De esta manera se garantiza que cada lote sea procesado por un *Info* distinto. Estos agentes se encargan de balancear la carga de los *Mail Agents*, al distribuir equitativamente la cantidad de correos que debe enviar cada

uno, y presenta mecanismos de tolerancia a fallas que se discutirán posteriormente.

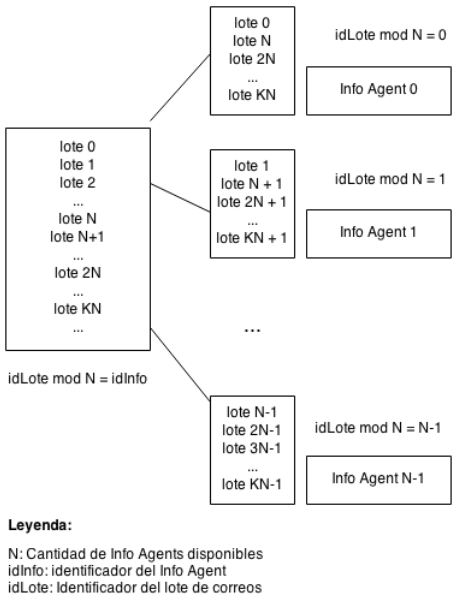


Figura 4: Distribución de Lotes de Correo entre Info Agents

Para establecer la plataforma multiagentes P2P se utilizó JADE (*Java Agent DEvelopment*), *framework* de software libre que simplifica el desarrollo de una arquitectura multiagente bajo el estandar FIPA. Se decidió utilizar esta herramienta gracias a que facilita la comunicación entre los agentes y su monitoreo. JADE provee mecanismos que permiten detectar eventos sobre los agentes que facilitan la gestión de los mismos dentro de OPTIMUS [17].

Cada uno de los agentes de las capas de la arquitectura de OPTIMUS, está implementado como un agente JADE y se utiliza este *framework* para gestionar la comunicación entre ellos, el ciclo de vida, la información de los servicios que ofrece cada uno y el estado de cada agente. Esto es posible gracias a los agentes inherentes al *framework* enumerados a continuación:

- *Agent Communication Channel (ACC)*: Provee el servicio de transporte de mensajes.
- *Agent Management System (AMS)*: Se encarga de la creación, eliminación, migración y monitoreo del ciclo de vida de los agentes. En la Figura 5 se ejemplifica el caso en el que un agente muere y el *AMS* detecta esta situación y lo notifica, enviando un evento, al resto de los agentes.
- *Directory Facility (DF)*: Se encarga de proveer información precisa y actualizada de los servicios que ofrecen los distintos agentes en la plataforma.

OPTIMUS se diseñó contemplando la eficiencia y distribución de tareas a través de todos sus agentes. En este sentido, los *Scheduler Agents* son responsables de notificar y supervisar a un conjunto de *Mail Agents*. La asignación *Scheduler Agent - Mail Agents* es dinámica y se realiza cuando un agente nace o termina cumpliendo con el Protocolo de Agentes OPTIMUS,

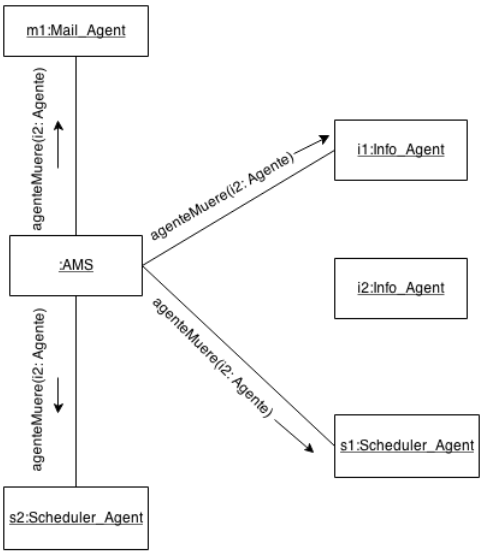


Figura 5: AMS: Notificar que un Agente Murió

con la finalidad de mantener el balance de carga entre los *Scheduler Agents*.

C. Protocolo de Agentes OPTIMUS: Balance de Carga

Cuando un *Mail Agent* nace, se llevan a cabo las siguientes operaciones:

- El *Mail Agent* envía un saludo a todos los *Scheduler Agents* disponibles en la plataforma anunciando su nacimiento.
- Cada *Scheduler Agent* realiza lo siguiente:
  - Consulta en la plataforma JADE el número de *Mail Agents* dispobibles.
  - Consulta en la plataforma JADE el número de *Scheduler Agents* disponibles.
  - Divide el número de *Mail Agents* entre el número de *Scheduler Agents* para obtener el máximo de *Mail Agents* que puede gestionar.
  - El *Scheduler* responde aceptando o rechazando el saludo. El saludo es aceptado si se cumple que: (i) El número de *Mail Agents* que tiene asociado es estrictamente menor que el máximo que puede gestionar, o (ii) Todos los *Schedulers* tienen asociados la misma cantidad de *Mail Agents*. El saludo es rechazado en caso contario.
- El *Mail Agent* recibe todas las respuestas y selecciona la primera que aceptó su saludo descartando el resto. Luego envía un *agent-hello* al *Scheduler Agent* seleccionado indicando su asociación.

Por ejemplo, en la Figura 6 hay dos *Scheduler Agents* y tres *Mail Agents*. Cuando nace *Mail Agent 4*, la cantidad máxima de cada *Scheduler Agent* sería 2. Por lo tanto, *Scheduler 1* rechaza el saludo y *Scheduler 2* lo acepta.

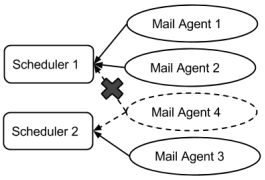


Figura 6: Nacimiento de un Mail Agent

Cuando un nuevo *Scheduler Agent* nace, se llevan a cabo las siguiente operaciones:

- El *Scheduler Agent* envía una notificación a todos los *Scheduler Agents* disponibles en la plataforma.
- Cada *Scheduler Agent* realiza lo siguiente:
  - Consulta en la plataforma JADE el número de *Mail Agents* disponibles.
  - Consulta en la plataforma JADE el número de *Scheduler Agents* disponibles.
  - Divide el numero de *Mail Agents* entre el número de *Scheduler Agents* antes de su nacimiento para obtener la cantidad máxima de *Mail Agents* que puede gestionar.
  - Si la cantidad de *Mail Agents* que tiene asociado es mayor que el máximo a gestionar, migra al nuevo *Scheduler Agent* la cantidad sobrante de *Mail Agents*.
  - Si la distribución de *Mail Agents* es equitativa entre los *Schedulers*, cada uno migra un *Mail Agent* al nuevo *Scheduler Agent*, sin que el nuevo llegue al máximo.

Por ejemplo, en la Figura 7 nace el *Scheduler 3* y notifica al resto de los *Schedulers* sobre su nacimiento (incluyéndose). Como antes habían 6 *Mail Agents* y dos *Schedulers*, el máximo era 3. Luego del nacimiento de *Scheduler 3*, el nuevo máximo sería 2, por lo que la carga de *Mail Agents* se redistribuye.

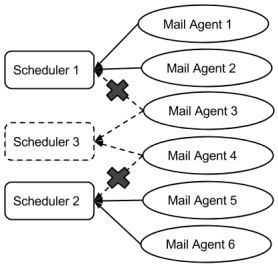


Figura 7: Nacimiento de un Scheduler Agent

D. Tolerancia a Fallas

Una de la principales motivaciones de este proyecto es la realización de una plataforma distribuida de envío masivo de correo electrónico que sea robusta y tolerante a fallas.

Como se mencionó anteriormente, la arquitectura de OPTIMUS está basada en capas, donde cada una de éstas desempeña un rol importante para la supervisión del resto de la plataforma. En líneas generales, cada capa ofrece tolerancia a sus capas subsiguientes.

El alcance de esta sección se enfoca en evaluar la tolerancia a fallas y recuperación de cada una de las capas de OPTIMUS y del resto de sus componentes.

*1) **Tolerancia de Ausencias**:* Al momento de distribuir los correos es posible encontrar escenarios de ausencia total de los agentes de las capas de OPTIMUS.

- Envío de correos en ausencia de agentes de capa 1 (*Info Agent*): Como se explicó anteriormente en el flujo normal de OPTIMUS, cuando se desee enviar un nuevo lote de correos, el *Sendmail* inserta el lote en la Base de Datos, como "lote entrante". En caso de que no exista ningún *Info Agent* que procese los lotes nuevos, estos persisten en la Base de Datos global hasta que se inicie algún *Info Agent*.
- Envío de correos en ausencia de agentes de capa 2 y/o de capa 3 (*Scheduler Agent* o *Mail Agent*): Cuando un *Info Agent* toma un lote de los "lotes entrantes", se asegura de que existan tanto algún *Scheduler Agent*, como algún *Mail Agent* disponibles para terminar de procesar y distribuir los correos. Este procedimiento se ilustra en el algoritmo mostrado en la Figura 8.

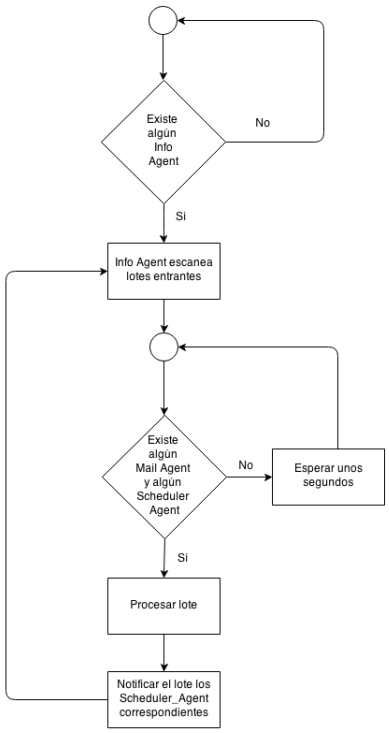


Figura 8: Algoritmo de Tolerancia a Fallas en Ausencia de Capa 2 y/o Capa 3

*2) **Tolerancia a Fallas de la Capa 1 - Info Agent**:* Cuando un *Info Agent* muere, el resto de los agentes de tipo *Info* recibe una notificación y reajustan sus identificadores, de manera de que se redistribuyan los lotes entre los *Info Agents* restantes, como se muestra en la Figura 9. Luego, cada uno consulta en la base de datos los lotes existentes, de forma tal que retome todos los lotes de correo que le pertenecían al agente que se detuvo. Esta rutina se ilustra en la Figura 10.

*3) **Tolerancia a Fallas de la Capa 2 - Scheduler Agent**:* Cuando un *Scheduler Agent* termina inesperadamente, se ejecutan los siguientes pasos:

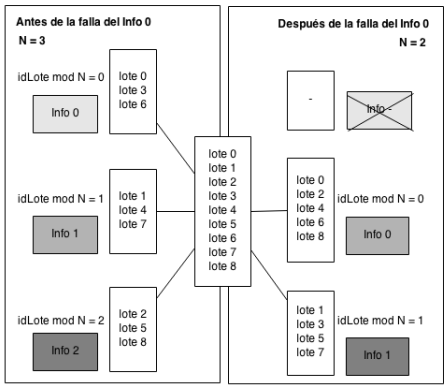


Figura 9: Recálculo de Identificadores de Info Agents

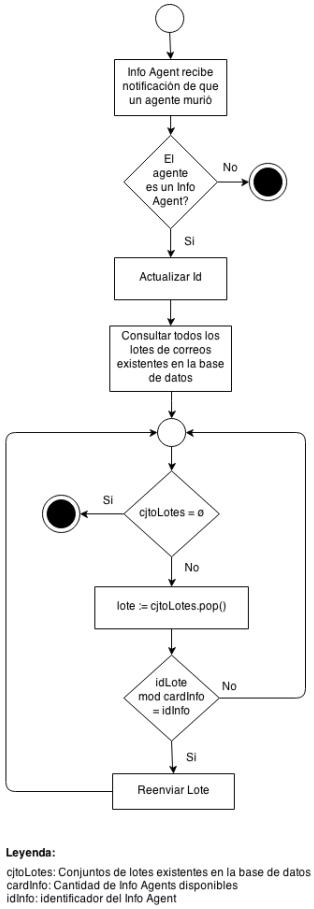


Figura 10: Algoritmo de Tolerancia a Fallas Capa 1

- Cada uno de sus *Mail Agents* detecta (a través de una excepción de JADE) que su *Scheduler Agent* terminó e inician el protocolo de nacimiento de *Mail Agents* para ser asignado a otro *Scheduler Agent*. Por ejemplo, en la Figura 11, el *Scheduler 2* falla y cada uno de sus *Mail Agents* se distribuye entre el resto de los *Schedulers*.
- Un *Info Agent* de capa 1, detecta la muerte del *Scheduler Agent*, el *Info* obtiene todos los *Mail Agents* que le pertenecían al *Scheduler* que murió, revisa si estos *Mail Agents* siguen en ejecución y de ser así, reasigna los lotes que les correspondía enviar. En caso que algún *Mail Agent* también haya fallado, se crea un lote nuevo con todos los correos del *Mail Agent* fallido y se reenvían. La Figura 12, muestra el algoritmo del *Info Agent* en el caso de detectar la falla de un *Scheduler Agent*.

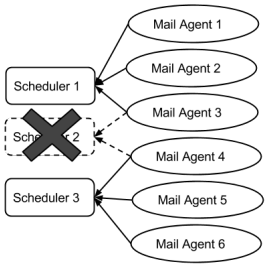


Figura 11: Redistribución de Cargas Cuando un Scheduler Agent Falla

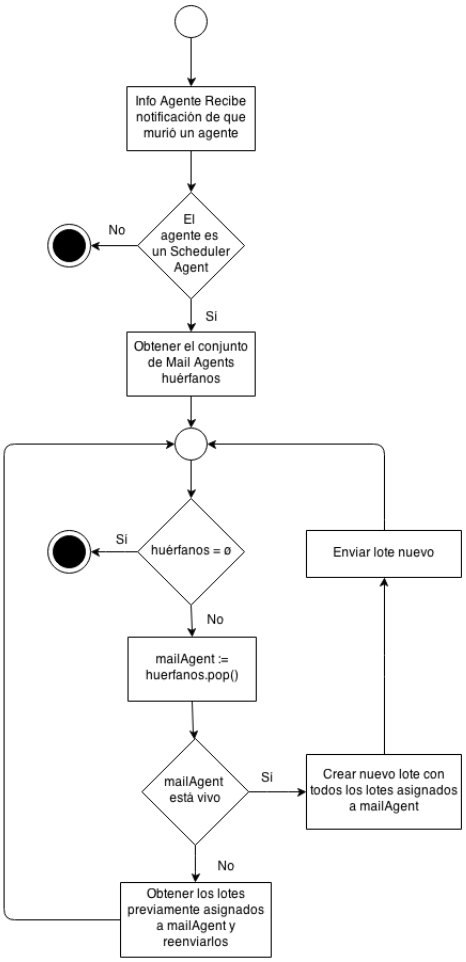


Figura 12: Algoritmo de Info Agent al Detectar Fallas Capa 2

4) ***Tolerancia a Fallas de la Capa 3 - Mail Agent****:* Cuando un *Mail Agent* termina inesperadamente, el *Scheduler Agent* responsable de ese *Mail Agent* detecta la falla (a través de una excepción de JADE) y realiza los siguientes pasos:

- Reasigna todos los lotes de correos que le correspondían al *Mail Agent* fallido, de acuerdo al algoritmo mostrado en la Figura 13.

- Inicia el protocolo de nacimiento de *Scheduler Agent* que le permite distribuir la carga de *Mail Agents* correctamente entre los *Scheduler Agents* disponibles.

Figura 13: Algoritmo de Tolerancia a Fallas Capa 3

Por otro lado, los *Scheduler Agents* revisan periódicamente si todos sus *Mail Agents* están enviando correos, de manera que detecte si alguno de ellos está retrasado en sus envíos, previendo que esto puede significar que *Postfix* presenta alguna falla. En este caso, se le solicita al *Mail Agent* retrasado, que compruebe su conexión con el servidor *Postfix* y en caso de no poder establecer esta conexión, el *Mail Agent* detiene su ejecución, forzando de esta forma que se reasigne su carga. Este comportamiento se ilustra en los algoritmos (para *Scheduler Agents* y *Mail Agents*) mostrados en la Figura 14.

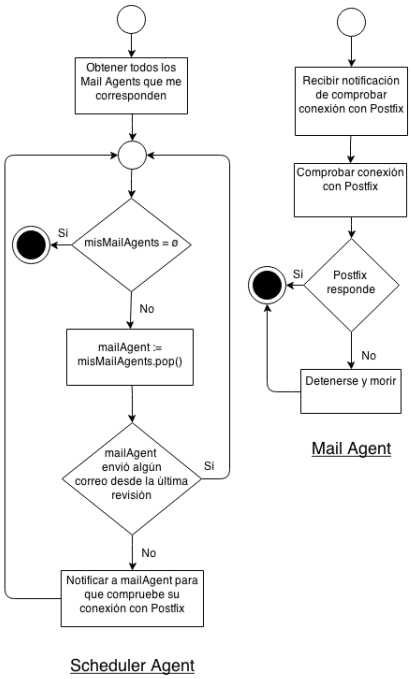


Figura 14: Algoritmos de Tolerancia a Fallas Capa 3: Mail Agent Demorado

5) ***Tolerancia a Fallas de JADE****:* El *framework* JADE permite crear plataformas distribuidas donde los agentes se ejecutan en contenedores asociados a la plataforma. Sin embargo, presenta una gran dependencia de un contenedor principal donde coexisten los agentes inherentes a JADE, el AMS y el DF. Esto último significa que existe un potencial punto de falla que puede ocasionar que toda la plataforma interrumpa su funcionamiento.

Combinando dos caracteristicas de JADE es posible desplegar una plataforma tolerante a fallas a nivel de la capa transversal de la arquitectura de OPTIMUS, que denominamos como *Capa JADE*. Gracias al *Main Replication* Service, se puede replicar el contenedor principal y el *AMS* que reside en él. De esta forma, si ocurre un falla y el contenedor principal deja de existir, las réplicas son capaces de detectar esto y comenzar las acciones de recuperación. Por otro lado, JADE proporciona el *Directory Facility* conocido como *DF* que provee información sobre los servicios que ofrecen los agentes en la plataforma, en el caso particular de este trabajo, se diferencian tres tipos de servicios para cada agente: *Info Agent*, *Mail Agent* y *Scheduler Agent*. Este catálogo también reside en el contenedor principal por lo que es importante proporcionar algún mecanismo que garantice su persistencia en caso de ocurrir alguna falla. Para lograr esto, JADE tiene la capacidad de vincular al *DF* con una Base de Datos de tal forma que se garantice la persistencia de la información contenida en este catálogo.

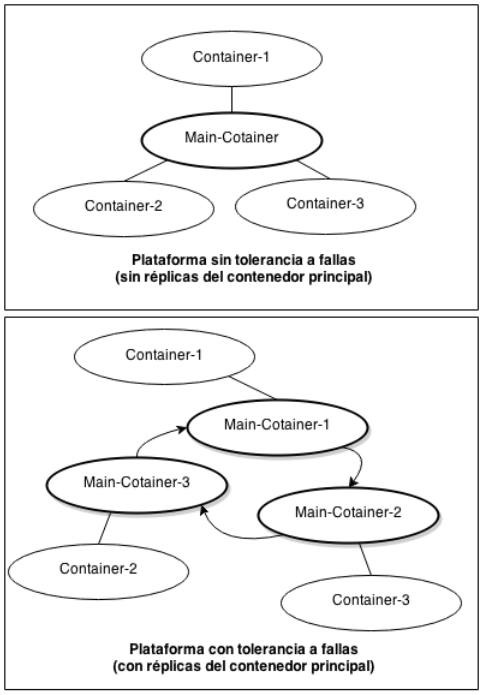


Figura 15: Tolerancia a Fallas JADE: Topología Estrella vs. Anillo [17]

La Figura 15 ilustra las topologías correspondientes a una plataforma en forma de estrella sin tolerancia y sin réplicas del contenedor principal, en contraposición a otra en forma de anillo que presenta réplicas del contenedor principal, las cuales monitorean al maestro y en caso de ocurrir una falla se reajustan, asignando un nuevo maestro. Por lo tanto, esta última provee tolerancia a fallas mientras que la primera no. La Capa JADE de OPTIMUS implementa la topología tolerante

a fallas.

La Tabla I muestra un resumen de las características que presenta cada uno de los agentes de OPTIMUS, en relación a los aspectos tratados en esta sección. Se puede observar que todas las capas de OPTIMUS ofrecen tolerancia a fallas y que las capas 1 y 2 son las encargadas del balance de carga de la plataforma.

Tabla I: Características de los Agentes de OPTIMUS

Característica	Capa 1	Capa 2	Capa 3	Capa JADE
Balance de Cargas	x	x		
Comunicación con Postfix			x	
Conexión con Base de Datos Global	x	x	x	
Conexión con Base de Datos Local		x	x	
Tolerancia a Fallas	x	x	x	x

IV. Evaluación Experimental

A. Evaluación del Rendimiento

Con la intención de evaluar el rendimiento de OPTIMUS frente a la versión original de Postfix, realizamos experimentos con lotes de correos de distintos tamaños y separamos las pruebas en dos escenarios: el primero, para medir el tiempo de inserción y procesamiento de los correos y el segundo, para medir el tiempo de envío de los mensajes.

1) ***Configuración de la Plataforma de Prueba****:* Para la implementación de OPTIMUS, se utilizó Redis 2.8.8 como Base de Datos, JADE 4.3 como plataforma P2P y se modificó la versión 2.11.0 de Postfix. Para las pruebas se utilizaron tres *PCs* con las siguientes especificaciones:

- Host 1: Intel(R) Pentium(R) D CPU 3.40GHz. SO: Debian 6 32bits. RAM: 1GB
- Host 2: Intel(R) Pentium(R) D CPU 3.40GHz. SO: Debian 6 32bits. RAM: 1GB
- Host 3: Intel(R) Pentium(R) D CPU 3.40GHz. SO: Debian 6 32bits. RAM: 1GB

2) ***Escenarios****:* La evaluación experimental se realizó utilizando las tres máquinas descritas anteriormente, de la siguiente manera: el *Host 1* desempeñó la función de enviador, el *Host 2* alojaba la Base de Datos Global y el *Host 3* se encargaba de recibir los mensajes.

Esta evaluación consistió en dos pruebas. En la primera se midió el desempeño de Postfix y de OPTIMUS insertando una cantidad determinada de correos en la cola *maildrop*. Por otra parte, en la segunda prueba se evaluó el desempeño tomando en cuenta el tiempo de envío de una cantidad determinada de correos. Ambas pruebas se realizaron utilizando lotes de 100, 1000, 10.000 y 100.000 mensajes, los cuales eran idénticos, y cada una de las prueba se repitió diez veces. Se reportan los promedios de las diez repeticiones.

3) ***Desempeño****:* La Tabla II muestra los resultados del desempeño de ambas arquitecturas insertando los mensajes en el *maildrop* y procesándolos para su posterior envío.

Tabla II: Tiempo Promedio de Inserción de Correos a *maildrop* de Postfix vs. OPTIMUS con Base de Datos Remota

Cantidad de correos	Tiempo de inserción (seg)		Porcentaje de mejora
	Postfix	OPTIMUS	
100	0.4997	1.83696	-267.6
1.000	5.2117	26.9182	-416.5
10.000	55.084	226.0528	-310.4
100.000	2654.62	2417.4916	8.9

Se observa que para un número pequeño de correos, el desempeño de OPTIMUS está muy por debajo del desempeño de Postfix, ya que nuestra arquitectura realiza una gran cantidad de operaciones, como distribución de cargas o accesos a Bases de Datos locales y remotas, lo cual ralentiza el procesamiento. No obstante, a medida que se aumenta el número de mensajes de manera significativa, OPTIMUS comienza a presentar leves mejoras en su desempeño.

Por otro lado, la Tabla III, refleja los resultados del desempeño de ambas arquitecturas para el envío de los correos.

Tabla III: Tiempo Promedio de Envío de Postfix vs. OPTIMUS

Cantidad de correos	Tiempo de inserción (seg)		Porcentaje de mejora
	Postfix	OPTIMUS	
100	1	1.1	-10
1.000	6.9	10.1	-46.4
10.000	110.7	101	8.8
100.000	1457.375	1022	30

En este caso, el desempeño de OPTIMUS continúa por debajo del desempeño de Postfix para cantidades de mensajes pequeñas, pero alcanza un 30% de mejora para el caso de los 100 mil correos.

Las Figuras 16 y 17 reflejan un comportamiento lineal del desempeño de OPTIMUS, tanto para el caso de procesamiento e inserción como para el de envío de correos, mientras que para Postfix, el desempeño empeora a medida que aumenta la cantidad de correos y no mantiene la linealidad.

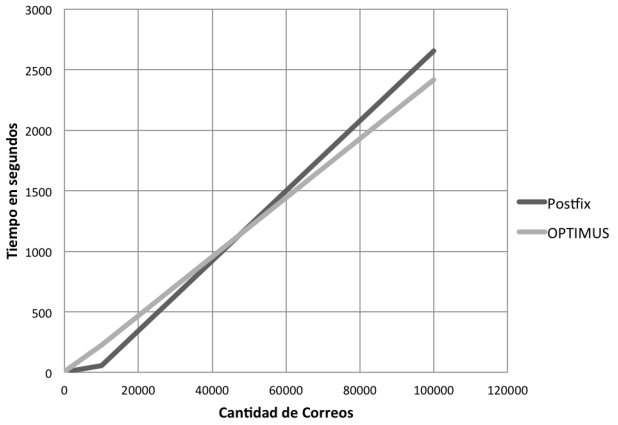


Figura 16: Inserción: Postfix vs. OPTIMUS con Base de Datos Remota

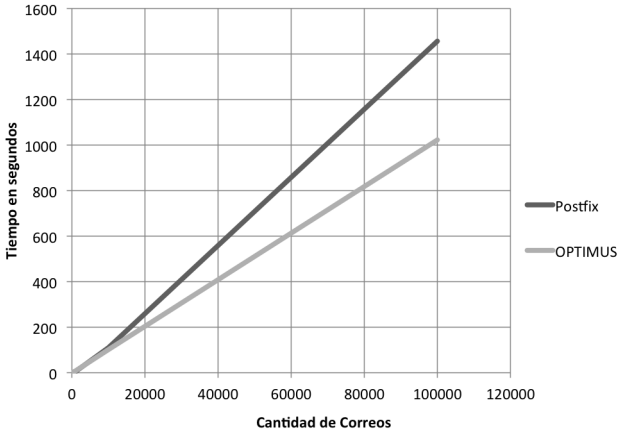


Figura 17: Envío: Postfix vs. OPTIMUS

B. Análisis del Impacto de la Tolerancia a Fallas sobre el Rendimiento

Una de los principales aspectos que caracteriza a OPTIMUS es la tolerancia a fallas que busca principalemente:

- Garantizar el envío de todos los correos electrónicos.
- Reasignar adecuadamente los correos que no pudieron ser entregados por motivo de alguna falla.

Estos objetivos se logran, aún en presencia de fallas, con la implementación de los algoritmos de tolerancia a fallas y recuperación. Realizamos un análisis experimental para determinar el nivel de intrusividad de la tolerancia a fallas, durante la actividad normal y el nivel de envío de correos duplicados ante una recuperación de una falla.

*1) **Nivel de Intrusividad**:* Cuando ocurre una falla de cualquiera de los agentes de OPTIMUS o JADE, ésta es detectada vía excepciones. Por lo tanto, no implica un *overhead* de tiempo durante la actividad normal del sistema (en ausencia de fallas). Esto implica que los algoritmos de tolerancia a fallas sólo son ejecutados en presencia de fallas. Un caso diferente ocurre cuando la falla se genera a nivel de Postfix. Para detectar estas fallas, los *Scheduler Agents* monitorean cada cierto tiempo que los lotes de sus *Mail Agents* se estén enviando. Este monitoreo períodico podría ocasionar intrusión en el rendimiento de la plataforma. Para medir este nivel de intrusión, realizamos dos pruebas con dos escenarios distintos, ambos con ausencia de fallas:

- Envío sin monitoreo: Se realizaron 10 envíos con lotes de 10 mil correos, sin el comportamiento que monitorea que los *Mail Agents* de un *Scheduler* estén enviando correos. Configuración de agentes: un *Mail Agent*, un *Scheduler* y un *Info Agent*.
- Envío con monitoreo: Se realizaron 10 envíos con lotes de 10 mil correos, con el comportamiento que monitorea que los *Mail Agents* de un *Scheduler* estén enviando correos. Configuración de agentes: un *Mail Agent*, un *Scheduler* y un *Info Agent*.

En la Tabla IV se observan los resultados de los experimentos realizados bajo los escenarios anteriormente descritos. Estos muestran que esta rutina de tolerancia a fallas incurre solo en un 11% de intrusión para una configuración con una sola instancia de cada tipo de agentes. Esta intrusión se debe a que la rutina se ejecuta muy seguido, y podría aminorarse si se amplía el período de tiempo entre una ejecución y otra.

Tabla IV: Intrusión del Comportamiento Monitor en el Desempeño de OPTIMUS

OPTIMUS sin monitoreo (segundos)	OPTIMUS con monitoreo (segundos)	% Intrusión
207	232.5	11

En resumen, tras llevar a cabo el experimento anterior, se observó que la implementación de la tolerancia a fallas no causa un impacto sustancial en el rendimiento de OPTIMUS en ausencia de fallas, dado que las rutinas de tolerancia únicamente se ejecutan una vez detectadas las fallas o, en el caso del monitoreo de los *Mail Agents*, el comportamiento se solapa con otras rutinas de envío y procesamiento y no genera un impacto negativo considerable en el rendimiento.

*2) **Correos Duplicados**:* Otra posible consecuencia de estos algoritmos de tolerancia y recuperación, es la aparición de correos duplicados. Esto se debe a que cuando ocurre una falla en algún *Mail Agent*, sus servidores Postfix pueden seguir procesando correos que no han sido aún eliminados de la Base de Datos global. Luego, cuando se activa una rutina de tolerancia a fallas y reasigne la carga que tenía asignado el *Mail Agent*, se redistribuyen correos que ya están siendo procesados por otro servidor de correo, causando que estos mensajes se envíen duplicados. Es decir, los correos duplicados son la consecuencia de no poder sincronizar de forma atómica la base de datos local, con la global.

Con la finalidad de obtener un porcentaje estimado de la cantidad de correos duplicados que podrían generarse en un envío de OPTIMUS, se llevaron a cabo 10 envíos de 10 mil correos, con dos *Mail Agents*, un *Scheduler Agent* y un *Info Agent*. En todas las iteraciones, se forzó una falla en uno de los *Mail Agents* y se esperó a que se reasignaran las cargas en cada caso. Al finalizar el experimento, se estimó que, en promedio, se duplicaron 8% de los correos enviados.

Es posible evitar los mensajes duplicados, pero esto requiere de un registro excesivo de todas las acciones que se realizan en la plataforma, lo que podría derivar en un cuello de botella y un consumo elevado de recursos. Por esta razón, OPTIMUS se centra en garantizar la entrega de la totalidad de los correos, más que en garantizar la unicidad de cada uno de los mensajes e ignora la posibilidad de que se duplique una pequeña cantidad de correos.

V. OPTIMUS Frente a Postfix Original

La principal característica de la arquitectura de OPTIMUS es la tolerancia a fallas y la distribución de tareas, lo cual garantiza robustez y el ahorro de recursos y tiempo durante el envío de correos. A continuación se listan las diferencias más importantes entre OPTIMUS y Postfix.

- Una ventaja de OPTIMUS sobre Postfix, es su capacidad de detectar y recuperar las fallas que puedan ocurrir en

sus servidores activos de correos, mientras que Postfix, a pesar de que es capaz de detectar alguna falla y mantener la persistencia de los correos, no presenta mecanismos de recuperación. Esto hace a OPTIMUS más robusto y tolerante a fallas que Postfix.

- Otro aspecto importante de OPTIMUS es la capacidad de modularizar los componentes y ubicarlos en distintas máquinas, lo que lo convierte en una plataforma escalable.
- La utilización de una Base de Datos NoSQL que comprime la información y la mantiene en memoria principal, minimiza de forma significativa los accesos a disco y la utilización del espacio de memoria, principalmente para los casos de cargas pesadas de correos. Esto permite que OPTIMUS presente un mejor desempeño con grandes cantidades de mensajes.
- En materia de seguridad, nuestra arquitectura tiene algunas debilidades a tomar en consideración. El hecho de no revisar la integridad de los correos, tras la constante transferencia de mensajes desde la Base de Datos global hasta las Bases de Datos locales o la misma existencia de una Base de Datos centralizada que al sucumbir ante algún ataque, ocasionaría el colapso de toda la plataforma. Por otro lado, Postfix es totalmente centralizado y presenta mecanismos como la ejecución de sus procesos en *chroot*, que elevan su nivel de seguridad.
- OPTIMUS permite distribuir equitativamente el balance de cargas entre un conjunto de servidores Postfix, con la finalidad de optimizar el envío de correos y obtener mejoras en el desempeño. Por esta razón OPTIMUS consigue un mejor rendimiento para cargas pesadas de mensajes, que una sola instancia de Postfix.

La Tabla V resume la comparación de las características antes mencionadas. Cabe destacar que las debilidades de seguridad están planteadas como trabajo futuro. Una solución factible es la utilización del protocolo SSL para la transferencia de los correos de la Base de Datos principal, hasta las locales y para la comunicación entre agentes.

Tabla V: Cuadro Comparativo entre Postfix y OPTIMUS

Características	Postfix	OPTIMUS
Tolerancia a fallas	bajo	alta
Robustez	mediana	alta
Escalabilidad	baja	alta
Desempeño con cargas pesadas	bajo	alto
Seguridad	alta	baja

VI. Conclusiones y Trabajo Futuro

En el presente trabajo se plantea la arquitectura de OPTIMUS, una plataforma distribuida para el envío masivo de correos electrónicos utilizando una adaptación del servidor de correo Postfix con un sistema multiagente P2P.

De acuerdo a los resultados experimentales y las comparaciones realizadas, se puede concluir que la arquitectura planteada es más eficiente en utilización de recursos y en desempeño que Postfix para grandes cantidades de mensajes, lo cual representa una ventaja contundente a la hora del envío masivo de correos. Por otra parte, al manejar múltiples servidores de correo Postfix, OPTIMUS muestra una mejora en el desempeño gracias al balance de cargas entre los distintos agentes y, por lo tanto, entre los distintos servidores de correo. Aunado a esto, OPTIMUS ofrece mecanismos de tolerancia a fallas que no incurren en intrusión significativa en el desempeño y que garantizan el envío total de los mensajes. De esta manera, se puede afirmar que esta plataforma es robusta y tolerante a fallas.

Como trabajo futuro se desea incorporar un componente de Inteligencia Artificial que permita decidir, de acuerdo a la configuración de las máquinas y su estado en tiempo real, la distribución más eficiente posible de los correos electrónicos entre los distintos servidores que existen en la plataforma, optimizando de esta manera el uso de los recursos. Además, se debe comprobar el comportamiento aparentemente lineal que presenta el desempeño de OPTIMUS con pruebas que consideren lotes de correos más grandes que los utilizados en este trabajo.

Por último, pretendemos realizar mejoras en materia de seguridad, mediante la utilización de SSL para el cifrado de la comunicación entre Bases de Datos y entre los agentes, así como otras herramientas de seguridad para proteger la integridad de la máquina donde se ejecuta la Base de Datos principal.

References

[1] N. Christenson, *Sendmail Performance Tunning*, 1st edition, Pearson Education, September 2002.

[2] *The Ultimate Mobile Email Statistics Overview*, http://www.emailmonday.com/mobile-email-usage-statistics.

[3] D. Chaffey, *Email Marketing Statistics 2014*, 2014.

[4] S. Radicati and J. Levenstein, *Mail Statistics Report, 2013-2017*, Palo Alto, California, USA, 2013.

[5] D. Nesset, *Massively Distributed Systems: Design Issues and Challenges*, in proceedings of the Workshop on Embedded Systems, Cambridge, Massachussets, USA, 1999.

[6] *Sendmail.Now*, http:www.sendmail.com/sm/open_source.

[7] *The Postfix Home Page*, http://www.postfix.org.

[8] F. Rosato, J. Argüello, and Y. Cardinale, *OPTIMUS: OPTImized Mail distribUtion System*, in proceedings of the XL Latin American Computing Conference (CLEI 2014), Montevideo, Uruguay, September 2014.

[9] N. Cox, *Electronic Messaging*, 1st edition, Auerbach Publications, November 1999.

[10] F. Avolio and P. Vixie, *Sendmail: Theory and Practice*, 1st edition, Digital Press, March 1995.

[11] R. Blum, *Postfix*, 1st edition, Sams Publishing, May 2001.

[12] K. Dent, *Postfix - The Definitive Guide: A Secure and Easy-to-Use MTA for Unix. Applications and Inter-Networking Technologies*, 1st edition, O'Reilly Media, 2003.

[13] C. Hunt, *Sendmail Cookbook*, 1st edition, O'Reilly Media, December 2003.

[14] J. Klensin, N. Freed, M. Rose, E. Stefferud, and D. Crocker, *SMTP Service Extensions*, RFC 1869, November 1995.

[15] J. Myers, *Local Mail Transfer Protocol*, RFC 2033, October 1996.

[16] T. Macedo and F. Oliveira, *Redis Cookbook*, 1st edition, O'Reilly Media, August 2011.

[17] F. Luigi, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*, 1st edition, Willey, April 2007.

Índice de Autores

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