Sociedad Venezolana de Computación

Revista Venezolana de Computación

REVECOM

Revista Venezolana de Computación

Sociedad Venezolana de Computación

ISSN: 2244-7040

Vol. 3, No. 2
Diciembre 2016

REVECOM

Revista Venezolana de Computación

Sociedad Venezolana
de Computación

Editores:
Eric Gamess, Wilmer Pereira, Yudith Cardinale

ISSN: 2244-7040

Vol. 3, No. 2
Diciembre 2016

Editorial

Las TICAs

Por el encabezado, se creería que fuésemos a hablar de las mujeres costarricenses, mujeres cultas y hermosas, de un país que ha logrado altos niveles de vida y una gran tranquilidad social, en el marco de una de las democracias más sólidas y largas de nuestra América, sin necesidad de ejércitos y militares, para priorizar la inversión social.

Pero no, quiero hablar aquí de las *Tecnologías de Información, Comunicación y Automatización*, es a eso que he abreviado como TICAs. Las TICAs son imprescindibles en los Ambientes Inteligentes, para posibilitar que los objetos, inteligentes o no, virtuales o físicos, en el ambiente, puedan interactuar entre sí de manera autónoma, y actuar sobre el ambiente, de tal manera de alcanzar los objetivos establecidos en el mismo.

Las TICAs nos permiten hacer converger con las *ciencias computacionales*, otro dominio imprescindible para los ambientes inteligentes, y que en otros ámbitos ya se habían dado la mano, los *sistemas de control y automatización*. La automatización permite introducir una capacidad importante en los sistemas informáticos: que puedan ejecutarse sin la necesidad de la intervención humana.

Para que sea inteligente, un ambiente debe poder actuar de manera autónoma, pudiéndose auto-organizar y hacer emerger nuevos comportamientos, en función de las necesidades del entorno. Eso es solo posible, si en el ambiente los avances de la automatización se dan la mano con la TICs. Estos ambientes deben considerar dos aspectos fundamentales, y en uno de ellos es imprescindible la automatización. El primero de ellos tiene que ver con *gestionar el indeterminismo y la ambigüedad* presente en esos ambientes, y allí paradigmas como la lógica difusa y las teorías estocásticas pueden ser usados. Además, los últimos avances en los sistemas emergentes permiten introducir nuevos mecanismos de adaptación en esto sistemas, basados en ideas como consenso social, asignación dinámica de tareas, programación emergente, entre otras.

Pero es en el otro ámbito donde es imprescindible la automatización, para posibilitar la *actuación autonómica de los objetos* en el ambiente, a partir de lo que van observando en él. En ese sentido, hablamos de la capacidad de los objetos en el ambiente para actuar de manera autónoma sobre el ambiente, según los objetivos requeridos en un momento dado. La *automación industrial* es un área bastante madura, y se basa en el uso de elementos computarizados y electromecánicos para la gestión autónoma de procesos industriales.

Las TICAs en los ambientes inteligentes, posibilitan conferirle a esos ambientes distribuidos un alto grado de autonomía, para lo cual se conjuga con la *Computación Autonómica*, de tal manera de posibilitar procesos de auto-gestión en esos ambientes, de forma dinámica, según sus necesidades. En este caso, hablamos de la introducción de la automatización en el software, para administrar contextos de alta complejidad, de manera integral con las TICs.

Editorial

Si bien es cierto que muchos podrían decir que esa sinergia entre las TICAs ya existía en el contexto de la automatización industrial, en la domótica, entre otros ámbitos, el impulso que le están dando los ambientes inteligentes, la computación autonómica, los sistemas emergentes, están replanteando a la propia disciplina de automatización, por los nuevos retos que están apareciendo:

- La automatización clásicamente se ocupa de mecanizar un proceso específico, de tal manera de seguir un comportamiento deseado óptimamente. En un ambiente inteligente se debe considerar el efecto agregado, no determinista y espontaneo, de las acciones/comportamientos de sus múltiples componentes/actores.
- La automatización clásicamente se refiere a mecanizar procesos de dimensiones fijas. En un ambiente inteligente típicamente el contexto es difuso, ambiguo.
- La automatización en un ambiente inteligente debe estudiar el efecto del proceso mecanizado, sobre las propiedades de los componentes/actores que conforman al ambiente inteligente.
- En la automatización clásica, la descripción matemática del sistema a mecanizar/controlar no cambia en respuesta a las señales de control. En los ambientes inteligentes, primero que nada, no pueden derivarse fácilmente los principios para modelar matemáticamente el sistema a mecanizar/controlar, por lo que son normalmente modelos basados en datos del mismo ambiente, y además, deben ser actualizados en tiempo real en función de la dinámica del ambiente.
- Tanto la automatización clásica como la automatización en los ambientes inteligentes, se dan en el marco de un “bucle cerrado”. Ahora bien, en la automatización para ambientes inteligentes, es imprescindible la incorporación de procesos de gestión del conocimiento y de generación de comportamientos inteligentes, basados en mecanismos de aprendizaje y razonamiento.

Las TICAs están empezando a aparecer en diferentes contextos, en los buses de gestión de servicios, en los ciclos de tareas de análisis de datos, en los esquemas de supervisión y control de procesos industriales, entre otros ámbitos, acercándonos a lo que la ciencia ficción nos había hecho soñar, pero que hoy los nuevos paradigmas teóricos y los avances tecnológicos, nos permiten empezar a plasmar.

El uso de las TICAs para construir vida, para posibilitar formas mejores de relación de la tecnología con la sociedad, es un reto, ante el acecho de los hacedores de guerra, que se aproximan a los avances tecnológicos para la manipulación, para construir nuevas esferas de poder. Sus posibilidades son inmensas para un mejor vivir, para hacer que las tecnologías puedan acercarse a los ciudadanos de este mundo de una manera no invasiva, no intrusiva, colaborando con ellos, no destruyéndolos. Está en nosotros los ciudadanos, velar por un uso adecuado de las TICAs.

Dr. Jose Aguilar
Profesor Titular de la Universidad de Los Andes
Mérida, Venezuela

Revista Venezolana de Computación

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

En octubre de 2016, se celebraron conjuntamente la Cuarta Conferencia Nacional de Informática, Computación y Sistemas (CoNCISa 2016) y la Cuarta Escuela Venezolana de Informática (EVI 2016), en el Colegio Universitario de Caracas, Caracas, Venezuela.

La edición de este quinto número de ReVeCom está dedicada a los mejores trabajos presentados en CoNCISa 2016. 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 selecciones de los mejores artículos de conferencias, 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. Leonid Tineo (Universidad Simón Bolívar)

Vicepresidente:
Dra. Yudith Cardinale (Universidad Simón Bolívar)

Secretario:
Dr. Wilmer Pereira (Universidad Católica Andrés Bello)

Tesorero:
MSc. Rosseline Rodríguez (Universidad Simón Bolívar)

Coordinadora de Educación e Investigación:
Dra. Dinarle Ortega (Universidad de Carabobo)

Coordinador de Publicaciones:
Dr. Eric Gamess (Universidad Central de Venezuela)

Coordinadora de Eventos:
MSc. Nataly Carmona (Universidad Marítima del Caribe)

Edición

Comité Editorial

Miembros del Comité Editorial

Tabla de Contenido

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 1-12, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Arquitectura para PostgreSQLf con Facilidades de Minería de Datos Descriptiva Difusa

Livia Borjas[1], Ana Aguilera[2], Rosseline Rodríguez[3], Francisca Losavio[4]
livacaro7@gmail.com, aaguilef@uc.edu.ve, crodrig@usb.ve, francislosavio@gmail.com

[1] Departamento de Informática, IUT FRP, Caracas, Venezuela
[2] Centro de Análisis, Modelado y Tratamiento de Datos CAMYTD, Universidad de Carabobo, Valencia, Venezuela
[3] Departamento de Computación, Universidad Simón Bolívar, Caracas, Venezuela
[4] Laboratorio de Modelos, Software y Tecnología (MoST), Universidad Central de Venezuela, Caracas, Venezuela

Resumen: En este trabajo se presenta una propuesta de arquitectura para PostgreSQLf, sistema gestor de bases de datos relacional difuso (SGBDRD) con capacidades de minería de datos descriptiva y representación difusa. Para la presente propuesta se aplicó el modelo de proceso de Diseño Arquitectónico Orientado a Metas, Aspectos y Calidad (DAOMAC), proceso que hace énfasis en la consideración explícita de los requisitos no funcionales desde el modelo de negocio y en diversos niveles de abstracción de todo el proceso de diseño arquitectónico hasta obtener una arquitectura inicial que garantice la trazabilidad entre componentes que representa funcionalidades y sus requisitos de calidad. Entre los resultados obtenidos, siguiendo DAOMAC, se destacan la descripción del modelo de negocio asociado al dominio del problema de implementación de sistemas gestores de bases de datos con requisitos funcionales de gestión de grandes volúmenes de datos y extracción de modelos con representación difusa. Además, se presenta la propuesta de una arquitectura general para PostgreSQLf, en su versión como arquitectura inicial orientada a metas, aspectos y calidad para esta familia de sistemas.

Palabras Clave: Arquitectura de Software; SGBDR; PostgreSQLf; Minería de Datos Difusa; Diseño Arquitectónico; DAOMAC.

Abstract: This work presents an architecture proposal for the fuzzy Relational Database Management System (RDBMS), PostgreSQLf, with capabilities of descriptive data mining and fuzzy representation. For the present proposal, the Architectural Design Process Oriented Design for Aspects and Quality (or Diseño Orientado a Metas, Aspectos y Calidad, DAOMAC) process model was applied; this is a process that emphasizes the explicit consideration of non-functional requirements from the business model; various levels of abstraction are considered during the whole architectural design process, until obtaining an initial architecture that guarantees the traceability of components representing functionalities and their quality requirements. Among the results obtained, according to DAOMAC, we highlight the description of the business model associated to the domain of the problem of implementation of database management systems with functional requirements for managing large volumes of data and extracting models with fuzzy representation. In addition, we present the proposal of a general architecture for PostgreSQLf, in its version as initial architecture, goals, aspects and quality-oriented, for this family of systems.

Keywords: Software Architecture; RDBMS; PostgreSQLf; Fuzzy Data Mining; Architectural Design; DAOMAC.

I. Introduction

Las Bases de Datos Relacionales, han facilitado la manipulación de datos en las organizaciones eficientemente. A pesar de sus grandes aportes en el manejo de la información, los Sistemas Gestores de Bases de Datos Relacionales (SGBDR) han presentado las siguientes situaciones: rigidez en la especificación y procesamiento de datos imperfectos, así como la acumulación de grandes volúmenes de datos, producto de la operación diaria de dichas bases de datos, sin ofrecer capacidades para su explotación rentable.

Por esta razón implementar SGBDR que provean capacidades de flexibilización en la especificación y manipulación de los datos, así como de mecanismos para la explotación de grandes volúmenes de datos, es un reto de investigación en la actualidad en la materia de gestión de datos [1][2][3].

Una manera de implementar este tipo de sistemas es integrando los SGBDR existentes con funcionalidades de gestión de datos difusos y de aplicación de las técnicas de Minería de Datos [4][5], que permiten extraer conocimiento implícito y novedoso con representación difusa. Existen tres (3)

arquitecturas de integración o acoplamiento [3] de estos SGBDR con estás funcionalidades especiales, algunas incorporadas a través de extensiones del lenguaje de consultas SQL con lógica difusa [1][6].

En los últimos veinte años [5], se han tenido diferentes experiencias en la implementación y aplicación de Sistemas Gestores de Bases de Datos Relacionales Difusos (SGBDRD), que usan SQLf y más recientemente con técnicas de minería de datos difusas utilizando diversos enfoques de integración [7]. Además, se han desarrollado diferentes aplicaciones que usan estas implementaciones [5].

Sin embargo, algunas de estas experiencias han usado procesos de desarrollo de software que no proveen en general mecanismos de conceptualización de requisitos funcionales (RF) difusos para sistemas o productos de minería de datos, ni de requisitos no funcionales (RNF) [8]. De igual manera no existe un marco arquitectónico de referencias que permita validar las características de calidad que estos productos presentan.

Por estas razones, en este trabajo se presenta una arquitectura inicial que sirva como marco de referencia para la implementación de estos SGBDR. Se espera que dicha arquitectura garantice la clara incorporación de requisitos funcionales y no funcionales en el proceso de construcción de un SGBDR Difuso (SGBDRD) desde el análisis del dominio, a fin de que provea mecanismos de trazabilidad entre sus modelos, vistas.

El presente trabajo se organiza de la siguiente forma: La Sección II expone los antecedentes de la investigación. La Sección III ofrece una breve explicación del modelo de proceso usado para el diseño arquitectónico. En la Sección IV se presenta la arquitectura inicial propuesta resultante de la aplicación de la metodología. Finalmente, en la Sección V se plantean las conclusiones y trabajos futuros.

II. ANTECEDENTES

Los antecedentes de la presente investigación incluyen: extensiones de SQL para incorporar gestión de datos difusos y capacidades de minería de datos; diferentes propuestas de arquitecturas según el tipo de acoplamiento usado, que permiten integrar las extensiones de SQL con los SGBDR; y finalmente, el proyecto PostgreSQLf, cuyo propósito es implementar una versión de PostgreSQL que permita la gestión de datos difusos.

A. SQLf

Se trata de una extensión de SQL que permite realizar consultas difusas sobre una base de datos tradicional. Las consultas difusas permiten especificar expresiones con lógica difusa en los lugares donde se usa la lógica clásica. La teoría de Conjuntos Difusos [9] es la base de la Lógica Difusa. En esta lógica, el valor de verdad de una condición está en el intervalo real [0,1]. El valor 0 se entiende como “completamente falso” y el valor 1 como “completamente cierto”. Los valores intermedios representan distintos grados de certidumbre o falsedad.

El uso de la lógica difusa permite gestionar datos de naturaleza vaga, imprecisa o incompleta [10], así como consultas difusas sobre datos tradicionales, los cuales están presentes en la lingüística humana y muchas veces expresan las preferencias de los usuarios [2][11]. Por ello han surgido varias extensiones del lenguaje de consultas SQL con lógica difusa, entre las cuales las más completas son: FSQL [12] y SQLf [1].

La presente investigación tiene especial interés en SQLf, por la variedad de consultas con condiciones flexibles, que permite especificar sobre Bases de Datos Relacionales. Los términos difusos de SQLf son incorporados en el sublenguaje de definición de datos DDL, del inglés Data Definition Language, y el sublenguaje de manipulación de datos DML, del inglés Data Manipulation Language incluyendo expresiones lingüísticas como: predicados, modificadores, comparadores, conectores y cuantificadores.

Por ejemplo, si se desea definir el término lingüístico “joven" sobre un dominio de edades comprendidas entre 0 y 120, se puede especificar un predicado difuso cuya función de membresía indique satisfacción completa (grado 1) para las edades menores que 30, para las edades en el intervalo comprendido de 30 a 60 años, un grado que decrece proporcionalmente (en línea recta) y para las edades mayores que 60 una completa insatisfacción (grado 0). Esta definición se expresa en SQLf mediante la instrucción CREATE FUZZY PREDICATE joven ON 0 .. 120 AS (0, 0, 30, 60). El gráfico de este tipo de definición, conocida como trapezoidal, puede observarse en la Figura 1.

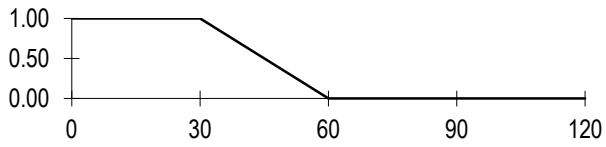


Figura 1: Definición Trapezoidal del Predicado Difuso *Joven*

La estructura básica de consulta de SQLf es el bloque multirelacional, cuya forma es SELECT <*attributes*> FROM <*relations*> WHERE <*fuzzy conditions*> WITH CALIBRATION [k|a|k,a]. Donde la cláusula WITH CALIBRATION es opcional, y permite la escogencia de las mejores respuestas. De esta forma, las condiciones (<*fuzzy conditions*>) de la cláusula WHERE pueden involucrar términos difusos definidos por el usuario, operadores y/o subconsultas difusas.

Entre sus características, SQLf permite la cuantificación lingüística sobre el conjunto de filas obtenidas en una consulta, a través del uso de estructuras de anidamiento y particionamiento con cuantificadores difusos. Así como también permite el uso de cuantificadores para calificar la cantidad de criterios de búsqueda satisfechos por filas.

Se han propuesto diversos mecanismos para la evaluación de las consultas en SQLf, destacándose el principio de derivación propuesto por Bosc y Pivert [13]. Este principio permite derivar consultas clásicas a partir de consultas difusas, de manera que el costo añadido por el procesamiento de un lenguaje difuso se mantiene lo más bajo posible.

En el presente trabajo se propone incorporar a SQLf capacidades para explotar los grandes volúmenes de datos presentes en almacenes históricos, producto de la operación de las bases de datos relacionales en diversas organizaciones. La

idea es que se puedan ejecutar consultas difusas sobre estos almacenes usando técnicas de minería de datos.

B. Extensiones de SQL para Minería de Datos

Algunas extensiones al lenguaje SQL se han realizado para soportar las tareas de minería de datos en el contexto de los SGBDR. Entre estas extensiones se destaca, con especial interés para la presente investigación, la propuesta de Timarán [3]. En ella se extiende SQL con primitivas de minería de datos en una arquitectura fuertemente acoplada sobre PostgreSQL. Los algoritmos de extracción de conocimiento forman parte del núcleo del SGBDR como operaciones primitivas, por lo que son ejecutados en el mismo espacio de direccionamiento que los datos, resultando en una extensión de SQL con muy buen desempeño en costo, y mostrando buen rendimiento y escalabilidad.

Otros esfuerzos se han realizado para incorporar reglas de asociación en lenguajes de consultas difusas a bases de datos tradicionales. Para el caso de SQLf, se incorporaron técnicas de análisis de datos a través del operador GROUP BY [14] para generar grupos difusos de una manera práctica y eficiente. Además, se han realizado desarrollos que comprueban la factibilidad de implementar estos operadores [15].

Otro grupo de extensiones [3] incluyen operadores unificados SQL-LIKE para soportar una tarea específica de minería de datos. También se proveen primitivas genéricas que ayudan al proceso de extracción de conocimiento sin soportar una tarea de minería de datos específica. Por otro lado, se han propuesto lenguajes de consulta para minería de datos con características similares al lenguaje SQL con la sintaxis SQL-LIKE. La debilidad de estas propuestas es que utilizan arquitecturas débiles y medianamente acopladas, produciendo bajo rendimiento y baja escalabilidad.

C. Arquitecturas de Acoplamiento

Para incorporar nuevas funcionalidades a un SGBDR se utilizan diferentes estrategias conocidas como arquitecturas de acoplamiento [5]. El tipo de arquitectura de acoplamiento usada en una extensión a un SGBDR es una decisión de diseño de mucha importancia, pues su selección podrá favorecer los RNF que definen la calidad de dicho producto. En la Figura 2 se pueden observar los tres tipos de arquitectura [3]: acoplamiento débil (*Loosely coupled*), acoplamiento medio (*Mildly coupled*) y acoplamiento fuerte (*Tightly coupled*). En trabajos previos se ha tenido diversas experiencias que muestran la factibilidad de implementar un SGBDRD con un desempeño adecuado [5].

Estas tres posibles arquitecturas de acoplamiento pueden ser usadas para integrar extensiones de SQL que incorporan el uso de requisitos funcionales considerados "novedosos", pues no son soportados por la mayoría de las herramientas conocidas, es decir la gestión de datos difusos y de minería de datos, que implican criterios no funcionales excluyentes entre sí, como son: rendimiento, escalabilidad y portabilidad [8].

Cada estrategia de integración tiene sus ventajas y desventajas [5][8]. En particular la alta portabilidad es la ventaja principal de los SGBDR débilmente acoplados, mientras en contraste el rendimiento es el mayor aporte del caso fuertemente acoplado, aunque se pierde en portabilidad. El acoplamiento medio ofrece la oportunidad de añadir a los SGBDR las características imprecisas de SQLf, sacrificando la migración. En la Tabla I, se resumen los RNF para estas tres arquitecturas que surgen ante la necesidad de gestionar requisitos funcionales "novedosos".

Figura 2: Arquitecturas de Acoplamiento

Tabla I: RNF para las Arquitecturas de Acoplamiento Derivados de RF "Novedosos"

Arquitectura de Acoplamiento	**Requisito No Funcional**	
	Ventaja	***Desventaja***
Débil	Portabilidad	Rendimiento y escalabilidad
Medio	Escalabilidad, Rendimiento	Portabilidad, mayor costo de desarrollo
Fuerte	Rendimiento y escalabilidad	sin portabilidad, alto costo de desarrollo

Para el presente trabajo se está interesado de manera especial en la arquitectura de acoplamiento fuerte.

D. Proyecto PostgreSQLf

Un antecedente directo a la presente investigación es el Proyecto PostgreSQLf, desarrollado en la Universidad de Carabobo, desde el año 2006, el cual agrega capacidades para expresar consultas flexibles a bases de datos al SGBD Objeto-Relacional, PostgreSQL, usando el lenguaje SQLf [15].

Entre las capacidades del lenguaje SQLf que se han incorporado están: predicados difusos, operadores difusos AND y OR, modificadores difusos, consultas particionadas difusas en las cláusulas GROUP BY y HAVING, vistas y operación de unión difusas, creación de cuantificadores difusos y consultas cuantificadas difusas de tipo bloque simple, comparadores difusos, subconsultas difusas en la cláusula FROM, bloques anidados EXISTS-LIKE y ANY-LIKE [15].

En [16] se realiza una extensión del motor de consultas del PostgreSQLf usando una arquitectura de acoplamiento fuerte pero no propone un diseño arquitectónico orientado a metas que considere los aspectos y la calidad. Tampoco se incluyeron operadores para el análisis de agrupamiento difuso ni la especificación de reglas de asociación difusas.

En el marco de este proyecto, se ha propuesto desarrollar una extensión de SQLf para realizar análisis descriptivo difuso desde PostgreSQLf, con una arquitectura fuertemente acoplada

[8]. Se pretende añadir al SGBDRD operadores para extraer Reglas de Asociación Difusas y Grupos Difusos. La idea es proveer primitivas para realizar la tarea del análisis de agrupamiento, así como, especificar reglas de asociación difusas, dentro de la instrucción SELECT.

En este trabajo se presenta una propuesta arquitectónica para el SGBDRD PostgreSQLf, en el cual se incorpora la extensión de SQLf con las nuevas primitivas que soportan análisis de asociación y de agrupamiento difuso, de manera que se aprovechen los beneficios de rendimiento (eficiencia) y escalabilidad propias de la arquitectura de acoplamiento fuerte.

III. Principios de Diseño

Una Arquitectura de Software [17][18] es el conjunto de componentes, conectores y relaciones que representan la estructura, comportamiento y propiedades esenciales de un sistema de software, de manera global, con un alto nivel de abstracción. Los componentes son los elementos donde se llevan a cabo los cómputos, las partes representan los roles, los puertos describen los puntos de interacción entre el entorno y las partes internas; y los conectores constituyen los medios por los cuales interactúan los componentes y sus partes. En conjunto, estos elementos conforman la estructura estática o lógica de un sistema de software, de tal manera que la arquitectura describe la organización del sistema de software, las relaciones con otros componentes, el ambiente y los principios que gobiernan su diseño y evolución [19].

El diseño de una arquitectura de software [17] es un tema de interés en la actualidad; contemplan aspectos de calidad que influye en la capacidad evolutiva de la arquitectura o base que soporta el sistema. Son varios los trabajos que tratan el problema de diseñar una arquitectura del software [20][21]. En particular existen esfuerzos [22] que se enfocan en el análisis de RNF para el diseño arquitectónico, por ser estos los que orientan dicho diseño.

En el presente trabajo se usa como punto de partida el modelo de proceso para Diseño Arquitectónico Orientado a Metas, Aspectos y Calidad (DAOMAC) [23]. Este proceso es una propuesta para el diseño de una arquitectura inicial, centrado en la especificación de los requisitos de calidad de producto de software, asociados a los RNF relativos al dominio, especificados según el estándar ISO/IEC 25010 [24]. DAOMAC garantiza la correspondencia o trazabilidad entre el modelo de negocio y el modelo de la arquitectura inicial del sistema de software, conforme a los RNF que intervienen y que dirigen la construcción de la misma. En este modelo es esencial considerar los aspectos no funcionales desde tempranos niveles de abstracción, de acuerdo al enfoque de orientación a aspectos [22], como lo es el modelo de negocio. La orientación a aspectos es un paradigma de la Ingeniería de Software que sugiere tratar "temprano" en el ciclo de desarrollo todas aquellas incumbencias que son aspectos del sistema no percibidos directamente por el usuario, como es el caso de los RNF. En DAOMAC se hace uso de este principio para garantizar la trazabilidad de los RNF entre los modelos construidos, en cada nivel de abstracción; el uso de una especificación estándar [24] de las propiedades de calidad relativas a los RNF facilita la comunicación entre los miembros del equipo de desarrollo.

Para comprender mejor los planteamientos del proceso DAOMAC, se presentará de manera general algunos conceptos relacionados. En primer lugar, se define como *Arquitectura Inicial* (*AI*), la estructura computacional que posee los componentes principales del sistema de software, a partir de la cual se articulará el resto del sistema [17][19]. Una consideración fundamental para diseñar y construir una AI es tomar en cuenta como componentes específicos de la AI las propiedades funcionales conocidas como RF y los RNF que debe cumplir el sistema desde tempranas etapas del ciclo de vida del software.

En segundo lugar, dentro de una organización se puede definir un proceso [25] como una colección de actividades, que toma una o varias clases de entradas y genera una salida de valor para el cliente. Por tanto, un *Proceso de Negocio* [26] es el conjunto vinculado y natural de actividades basadas en habilidades y competencias de trabajo, que incluyen interacciones entre actores, recursos utilizados y vínculos de dependencias entre estos. Dichas actividades son ejecutadas de acuerdo a reglas preestablecidas que permiten satisfacer ciertas metas [27].

Dado que en una organización las operaciones suelen ser automatizadas, resulta útil concebir el negocio como soporte al proceso de Ingeniería de Software [22]. Por tal razón, el *Modelo de Negocio* (*MN*) es considerado como una vista abstracta y simplificada, en términos de metas y factores de importancia, que refleja la lógica de la compleja realidad del funcionamiento del negocio de una organización o entidad [23]. El MN es un punto inicial para la elaboración del sistema de software de mucha utilidad para la especificación de los requisitos globales que el sistema debe satisfacer. Así, el *Modelo de Proceso de Negocio* (*MPN*) [28] se define como la representación gráfica de los procesos organizacionales, o red de actividades y controles de flujos, que describe su funcionamiento.

Una AI [22] es el modelo en el cual se identifican los componentes orientados a aspectos, derivados de RNF y los componentes funcionales del sistema, derivados de RF. En la Figura 3, se observa que el modelo de proceso DAOMAC, está conformado por un "paquete" con cuatro disciplinas, cada una de las cuales es descrita en otro paquete. Las cuatro disciplinas son: Construcción del Modelo de Negocios, Obtención del Modelo de Metas, Determinación de las Metas No Funcionales Transversales y Construcción Orientada a Aspecto de la Arquitectura Inicial. Estas disciplinas se interrelacionan entre sí, de acuerdo a un modelo iterativo y los modelos obtenidos en cada nivel de abstracción son la entrada y punto de partida para construir el modelo del nivel de abstracción siguiente.

Estos modelos son representados usando diferentes lenguajes de notación gráfica: MN debe ser expresado en el lenguaje BPMN, (*Business Process Model and Notation*) [30][31], el Modelo de Metas (MM) en el Lenguaje de Requisitos Orientado a Metas o GRL (*Goal-oriented Requirements Language)* [32], y AI en la notación estándar SPEM (*Software Process Engineering Metamodel*) [33] basada en el Lenguaje Unificado de Modelado, UML (*Unified Modelling Language*) [34].

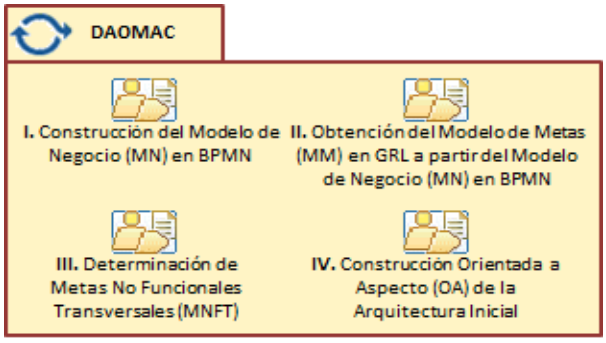


Figura 3: Proceso DAOMAC [29]

MM [23] incluye el conjunto de metas no funcionales transversales y su refinamiento en el diagrama o grafo jerárquico de interdependencias, ambos expresados como un *Softgoals Interdependency Graph* (SIG) [35][36]. Estas interdependencias son identificadas como operacionalizaciones concretas que permiten derivar componentes arquitectónicos; una *operacionalización* corresponde a un mecanismo arquitectónico específico que satisface o es una solución para el cumplimiento del *softgoal* (o meta no funcional). Estás metas están directamente relacionadas con los requisitos de calidad del sistema. El MM se basa en la Ingeniería de Requisitos Orientada a Metas [37]. MM se utiliza como enlace entre el modelo de negocio expresado en BPMN y el modelo de arquitectura inicial del sistema de software.

Esto se observa detalladamente en la Figura 4, la cual describe que el Modelo de Calidad del Producto, construido a partir del estándar ISO/EC 25010, el cual interviene en todos los niveles de abstracción del proceso DAOMAC.

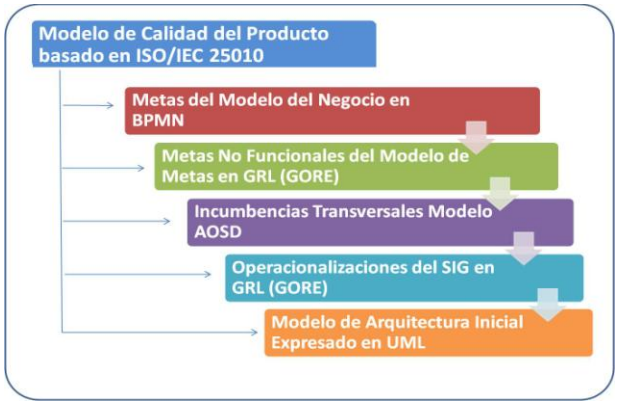


Figura 4: Niveles de Abstracción en el Proceso de DAOMAC

En DAOMAC, el MN es el punto de partida para construir la arquitectura inicial. Luego de aplicar varias transformaciones sobre el MN, se construye el modelo de metas, para finalmente obtener un modelo de arquitectura inicial. Este proceso presenta la siguiente secuencia transformacional:

1) En el MN, se identifican Tareas Funcionales y No Funcionales y los vínculos entre ellas. Las Tareas No Funcionales (TNF) pueden dar origen a Metas No Funcionales (MNF), denotadas por el estereotipo "MNF" del Lenguaje BPMN. Las MNF a nivel de BPMN indican objetivos de calidad asociados a una tarea [30][31].

2) Luego se construye el modelo de metas expresado en GRL, para hacer la transición entre las "MNF" del modelo de negocio y los requisitos no funcionales o *softgoals* del sistema y así identificar las Metas No Funcionales Transversales (MNFT), y su refinamiento en el SIG como operacionalizaciones concretas [23]. Estas MNFT del MM corresponden de manera directa con las Incumbencias Transversales (*Softgoals*) del Diagrama de Interdependencias de *Softgoal* o Modelo AOSD (*Aspect-Oriented Software Development*) [23]. Muy importante resaltar que la MNFT identificada como principal es el requisito no funcional transversal considerado de mayor prioridad para dicho sistema.

3) Seguidamente, las Operacionalizaciones del SIG expresadas en GRL, corresponden, en parte, con mecanismos o soluciones arquitectónicas [23], descritas y evaluadas de acuerdo a sus atributos o propiedades de calidad, para la selección de una opción adecuada entre varias alternativas. Finalmente, estos mecanismos/soluciones arquitectónicas se convertirán en "Componentes Aspectos", es decir, componentes orientados o tratados como aspectos (estereotipados en UML como «Aspect»), y en los componentes funcionales, obteniendo así el Modelo Arquitectónico Inicial expresado en UML como vista lógica de componentes y conectores [38].

A continuación, se detallan estas tres fases transformacionales del proceso DAOMAC.

B. Construcción del Modelo de Negocio en BPMN

El modelo de negocio expresado en BPMN identifica los vínculos entre tareas funcionales y no funcionales [39]. Para ello se realizan los siguientes pasos:

1) Crear los *Pools* (actores) y *Lanes* (roles de los actores) del Modelo de Negocio.
2) Definir las Tareas Funcionales y Subprocesos en los *Pools* y *Lanes*.
3) Incorporar Compuertas Exclusivas entre las diferentes Tareas Funcionales y Subprocesos que subyacen en los *Pools* y *Lanes*.
4) Identificar Tareas No Funcionales y asociarlas a Tareas Funcionales y subprocesos mediante el estereotipo «MNF».
5) Establecer los Flujos de Secuencia y de Mensajes.

C. Obtención del Modelo de Metas en GRL

En el MM expresado en GRL se identifican las metas no funcionales transversales y su refinamiento en el SIG como operacionalizaciones específicas [23]. Se construye después de hacer la transición entre las MNF del modelo de negocio y los requisitos no funcionales del sistema de software. Para construir el MM se realizan los siguientes pasos:

1) En base al MN, definir la Meta Funcional Medular o Principal asignando prioridades y vínculos de dependencias del MM.
2) En base a los *Pools* y *Lanes* del MN, definir los Actores y Roles en el MM.
3) Identificar *Pools*, *Lanes*, Subprocesos, Tareas y Asociaciones, Objetos de datos y Compuertas en BPMN y

hacerlas corresponder con Metas Funcionales, Metas No Funcionales y Tareas en GRL.

4) En base a los elementos de los *Pools* y *Lanes* en BPMN, establecer vínculos en GRL

D. Determinación de las Metas no Funcionales Transversales

Las MNFT son asociadas a mecanismos/soluciones arquitectónicas, las cuales son descritas de acuerdo a sus atributos o propiedades de calidad cumpliendo con el estándar ISO/EC 25010 [24] y evaluadas para la selección de una opción adecuada entre varias alternativas. Los pasos de esta fase son:

1) En base al MN, definir la Meta Funcional Medular o Principal asignando prioridades y vínculos de dependencias del MM.

2) En base a los *Pools* y *Lanes* del MN, definir los Actores y Roles en el MM.

3) Identificar *Pools*, *Lanes*, Subprocesos, Tareas y Asociaciones, Objetos de datos y Compuertas en BPMN y hacerlas corresponder con Metas Funcionales, Metas No Funcionales y Tareas en GRL.

4) En base a los elementos de los *Pools* y *Lanes* en BPMN, establecer vínculos en GRL.

E. Construcción Orientada a Aspectos (OA) de la Arquitectura Inicial

Para construir la arquitectura inicial se realizan los pasos:

1) Elegir la operacionalización de mayor relevancia desde el punto de vista arquitectónico, para cada MNFT, en orden de prioridad.

2) Describir los mecanismos/soluciones arquitectónicas asociados a la operacionalización seleccionada.

3) Comparar los mecanismos/soluciones arquitectónicas descritas considerando el modelo de calidad ISO/IEC 25010 [24], adaptado al dominio.

IV. ARQUITECTURA DE POSTGRESQL

PostgreSQL fue inicialmente desarrollado en el Departamento de Ciencias de la Computación de la Universidad de California, Berkeley [40]. A partir de 1996 deja de ser un proyecto académico de dicha universidad y pasa a manos de una comunidad desarrolladores de todo el mundo, quienes dotaron al sistema de una alta consistencia, uniformidad y seguridad. El grupo de desarrollo actual de este sistema, se conoce como *PostgreSQL Global Development Group*. Actualmente, PostgreSQL es un servidor de base de datos de código abierto muy avanzado cuya licencia permite ser usado, modificado y distribuido sin costo alguno.

En esta sección se describirá la arquitectura del SGBD Objeto-Relacional PostgreSQL, a fin de facilitar la presentación de la propuesta de extensión.

A. Estructura Interna de PostgreSQL

Una propuesta para modelar la arquitectura de PostgreSQL fue realizada en [32], la cual la presenta como una arquitectura Cliente-Servidor, que está dividida en tres subsistemas: el cliente (Client), el servidor (Server) y el gestor de almacenamiento (Storage Manager). En la Figura 5 se muestra la organización de tales subsistemas.

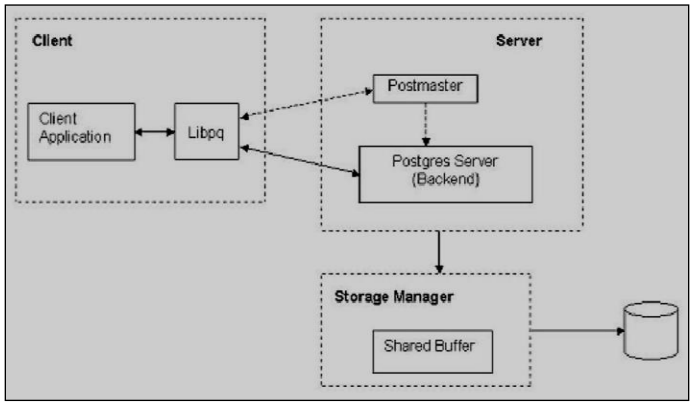


Figura 5: Estructura General de PostgreSQL [40]

El cliente consiste de cualquier aplicación que haga uso del SGBD, estableciendo una conexión al mismo por medio del componente Libpq. Este último es una interfaz en C nativo que gestiona la comunicación entre el SGBD y la aplicación cliente, enviando las consultas de este último al servidor. En la Figura 4 se muestra el funcionamiento de este subsistema.

El servidor lo integran dos componentes: el *Postmaster* y el *Postgres Server* o *backend*. El *Postmaster* está encargado de aceptar cada una de las peticiones del cliente que desee establecer una conexión, realiza las tareas de autenticación y control de acceso, y establece la conexión entre el cliente y un nuevo proceso o hilo del *Postgres Server*. De esta forma, cada cliente que haga uso del SGBD estará conectado a exactamente un proceso del servidor. Por otro lado, el *Postgres Server* maneja las consultas y tareas establecidas por el cliente.

El gestor de almacenamiento es responsable de toda la gestión y control de los recursos de almacenamiento en el SGBD, incluyendo *buffers* de memoria compartidos, así como, gestión de archivos, control de consistencia y manejo de *locks* sobre los archivos.

Dado que el servidor es quien representa el motor del SGBD, posteriormente, en [41] se estableció la estructura interna del *backend* de PostgreSQL. Los componentes que forman parte de esta estructura interna, así como el flujo de la comunicación entre ellos al momento de procesar una consulta, se pueden observar en la Figura 6.

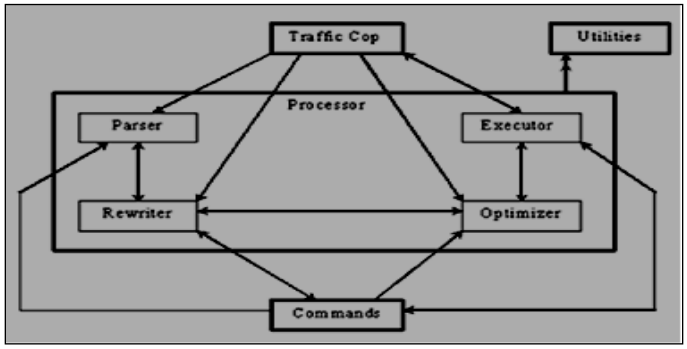


Figura 6: Arquitectura Interna del *Backend* de PostgreSQL [40]

El procesador (*processor*) está provisto de cuatro módulos: el *Parser*, el *Rewriter*, el *Optimizer* y el *Executor*. Cada una de

las tareas realizadas por estos módulos, son controladas por medio de la interacción de éstos con el *Traffic Cop*. La tarea fundamental del *Traffic Cop* (*T-Cop*) es controlar el flujo de operación de las consultas usuario en el subsistema procesador, cediendo el control operacional a cada uno de los cuatro módulos cuando corresponda.

B. *Ejecución de Operaciones en PostgreSQL*

El primer paso que se lleva a cabo para la ejecución de cualquier operación con PostgreSQL es establecer la conexión con el servidor. La misma es realizada entre el *Postmaster* y la aplicación cliente. La tarea principal de la aplicación es transmitir sentencias SQL del usuario al servidor, el cual se encargará de enviar de regreso los resultados obtenidos luego de la ejecución de dicha sentencia.

La sentencia SQL es enviada al módulo de *Parser* dentro del procesador para dar inicio a su ejecución. El *Parser* realiza el análisis gramatical y la transformación de la sentencia a una cadena de texto plano para validar su sintaxis. Luego de la validación se genera un Árbol de Traducción (*parser tree*), en caso de que la cadena sea válida, ó un mensaje de error en caso contrario. El *Parser* transforma el Árbol de Traducción en un Árbol de Consulta (*query tree*) para evaluar la existencia de las relaciones o atributos, a los cuales hace referencia dicho árbol, en el catálogo de tablas del SGBD. Si se detecta algún nombre desconocido, se devuelve un mensaje de error y se aborta el procesamiento de la sentencia. En caso contrario, el Árbol de Consulta se envía al *Rewriter*, el cual lo procesa, pues en caso de que exista alguna vista o regla que deba ser aplicada, se reescribe el árbol. El Árbol de Reescritura (*Rewritten Tree*) es enviado al *Optimizer*.

Una vez que se ha verificado que la sentencia del usuario cumple con todos los requisitos necesarios para la ejecución, el *Optimizer* refinar la ejecución, de tal manera que sea lo más económica posible para el servidor, en cuanto a consumo de recursos computacionales. Además, decide el plan de ejecución a llevar a cabo dependiendo de la estructura de la consulta. El *Executor* procesa dicho plan y se retornan las tuplas que cumplen con la condición de la consulta, o NULL cuando no existen tuplas que la cumplen.

V. Arquitectura Propuesta

La arquitectura de PostgreSQL sirve como punto de partida para construir una propuesta de arquitectura donde se incorpore al SGBD las capacidades para la gestión de consultas con minería de datos descriptiva difusa.

La propuesta arquitectónica aquí expuesta, describe el sistema donde se extiende el gestor PostgreSQLf incorporando al SQLf nuevas primitivas que añaden la semántica formal para soportar análisis de asociación y de agrupamiento difuso, de manera eficiente y escalable. Como se muestra en la Figura 7 de la Arquitectura Simplificada de PostgreSQLf, se incorporan las características novedosas directamente en los componentes ya existentes en el SGBD original.

Sin embargo, con el objetivo de garantizar las propiedades de calidad requeridas para PostgreSQLf con facilidades de minería de datos descriptiva difusa, esta propuesta de diseño arquitectónico se basa en una Arquitectura Inicial orientada a metas, aspectos y calidad. Las cualidades de calidad requeridas son rendimiento y escalabilidad, pues éstas garantizan una correcta gestión de consultas novedosas sobre grandes volúmenes de datos. En la propuesta solo se menciona el rendimiento pues la escalabilidad está incluida como parte de este requisito dentro de la norma ISO/IEC 25010.

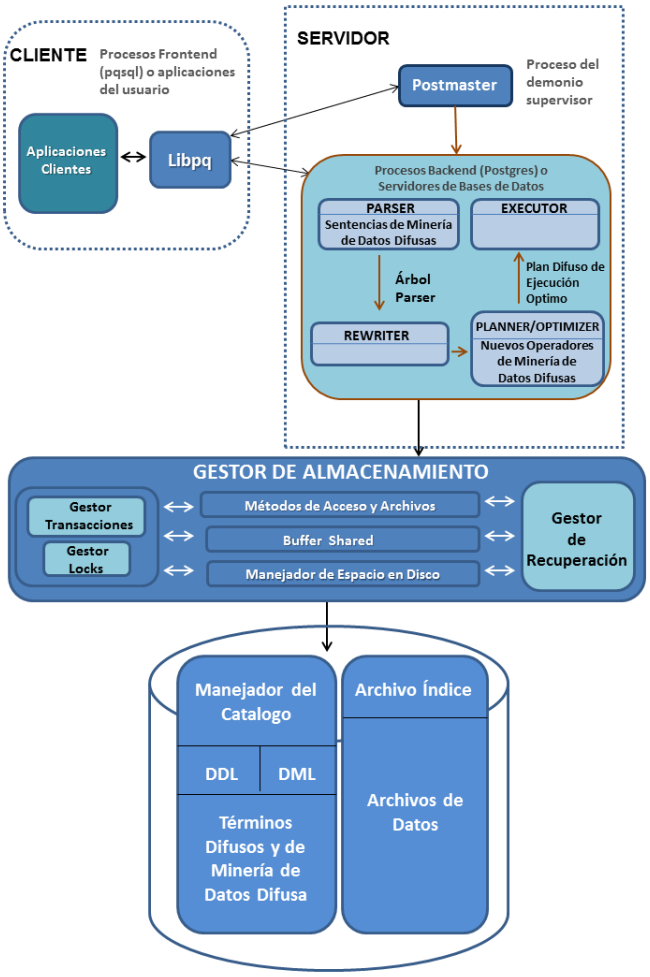


Figura 7: Arquitectura Simplificada de PostgreSQLf

Por otro lado, la gestión de grandes volúmenes de datos requiere que el diseño arquitectónico para tales capacidades se base en una estrategia de acoplamiento fuerte. Como ya se ha mencionado este tipo de acoplamiento garantiza tanto rendimiento como escalabilidad.

A continuación, se detalla la arquitectura propuesta, partiendo de una breve explicación del modelo de dominio que contiene las funcionalidades novedosas del sistema. Luego se explican las ventajas del enfoque de acoplamiento propuesto, para finalmente mostrar los resultados de la aplicación del proceso DAOMAC y de la Arquitectura Inicial obtenida.

A. *Modelo del Dominio*

Para ilustrar las funcionalidades novedosas que se desean incorporar a un SGBD, se describe el modelo del dominio a través de un ejemplo de aplicación: un sistema para la prevención de la deserción estudiantil en las universidades venezolanas.

Suponga que se tiene una tabla con los atributos de los ESTUDIANTES (ci, nombre, apellido, edad, EstratoSocial, carrera, PromedioNotas, nivel, estatus). Se quiere generar las

reglas de asociación difusas cuyo tamaño de regla sea igual a 4, con un soporte mínimo de 1 y una confianza mínima del 80%. Primero se obtienen todos los posibles subconjuntos de asociación conocidos como *itemsets*, los cuales quedan almacenados en la tabla ASSOCIATIONS. Para ello, se usa la sentencia SQL extendida:

SELECT nombre, apellido, edad, EstratoSocial, count(*) AS soporte INTO ASSOCIATIONS FROM ESTUDIANTES;

Posteriormente, tomando la tabla ASSOCIATIONS, se generan las reglas de asociación difusa con el nuevo operador propuesto DESCRIBE_FUZZY_ASSOCIATION_RULES, al cual se le indica el tamaño, el soporte y la confianza deseados:

SELECT * FROM DESCRIBE_FUZZY_ASSOCIATION_ RULES ('ASSOCIATIONS', 4, 1, 80);

Con estas reglas, si se desea contar la cantidad de estudiantes *pobres* que son *jóvenes*, *maduros* y *viejos*, cual se usa la sentencia extendida de bloque simple:

SELECT label(edad), count(*) FROM ESTUDIANTES WHERE EstratoSocial=pobre GROUP BY label(edad) USING PARTITION p(edad): {*joven*, *maduro*, *viejo*};

En esta consulta EstratoSocial=*pobre* es una condición difusa, donde pobre es un predicado difuso. Además, se tienen las etiquetas lingüísticas (*joven*, *maduro* y *viejo*) que generan una partición difusa del conjunto original de estudiantes para el atributo edad. De esta forma se generan los grupos difusos.

Aquí se observa que entre los requisitos funcionales "novedosos" se espera proveer consultas con capacidad de minería de datos descriptiva difusa, a fin de extraer conocimiento con representación en forma de reglas de asociación y de grupos difusos.

A partir de estos requisitos funcionales novedosos se infiere que los RNF esperados son: reusabilidad de los componentes de PostgreSQL, funcionamiento adecuado y preciso que induzca la corrección y pertinencia del sistema (idoneidad funcional) y finalmente, el rendimiento del SGBD en términos de velocidad de respuesta y uso de los recursos.

B. Enfoque de Acoplamiento

A manera general, el enfoque de integración usando una arquitectura fuertemente acoplada, es más sencillo debido a la totalidad de las tareas, componentes y funcionalidades a integrar, se implementan como operaciones primitivas. De esta forma, el SGBD podrá realizar las nuevas capacidades sin necesidad de conectarse o intercambiar datos con componentes adicionales, lo cual ocurre con los otros tipos de acoplamiento.

La ventaja principal de este enfoque es que se resuelven todos los problemas de escalabilidad y rendimiento presentados en los enfoques de integración para las arquitecturas débil y medianamente acopladas. Por otra parte, la limitación más relevante de la integración con una arquitectura fuertemente acoplada es la dificultad que representa mantener actualizadas oportunamente las nuevas funcionalidades que se hayan agregado al SGBD, cuando surjan nuevas versiones de éste o cuando ocurran avances teóricos en dichas funcionalidades. Esto siempre va a requerir cambios importantes en la estructura interna del código del SGBD.

C. Integración del Proceso DAOMAC

El proceso DAOMAC, descrito en la sección III, fue aplicado para extender la arquitectura general de PostgreSQLf con capacidades de Minería de Datos Descriptiva Difusa (MDDD), con el fin de construir su Arquitectura Inicial Orientada a Metas, Aspectos y Calidad.

La propuesta trata de abordar la implementación de consultas SQLf que permitan realizar un análisis descriptivo no supervisado para extraer el modelo de conocimiento con representación difusa, es decir, grupos y reglas de asociación difusas. Estas consultas se ejecutarán en el entorno de PostgreSQLf sobre los datos precisos.

Aplicando el proceso DAOMAC y partiendo del modelo del dominio descrito anteriormente, la Meta Funcional Primaria del Sistema es: Implementar (escribir, compilar y ejecutar) consultas en SQLf sobre bases de datos objeto relacionales para extraer conocimiento con representación difusa en forma de grupos y reglas de asociación, directo desde el entorno de PostgreSQLf.

Este proceso parte del modelo de negocio, en donde se describen los requisitos funcionales de minería de datos difusa. Se creó el *Pool o actor Usuario* con su respectivo *Lanes o rol de Analista de Datos*. Los pasos del proceso asociado a la meta funcional primaria se resumen en: establecer los requisitos funcionales de minería de datos difusa, selección y limpieza del conjunto de datos, selección del algoritmo de minería de datos que se va a aplicar, construcción de la vista minable, escribir las sentencias de SQLf, ejecutar la consulta y mostrar los resultados. El MN en estudio se muestra en la Figura 8, donde se observan las tareas funcionales y no funcionales necesarias para el PostgreSQLf.

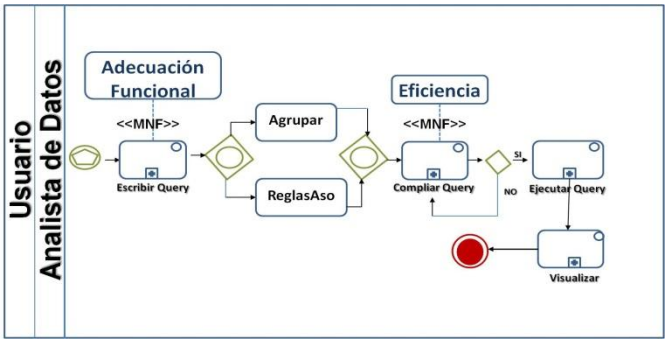


Figura 8: Modelo de Negocio de PostgreSQLf con MDDD

Este modelo de negocio está dirigido por el Modelo de Calidad del Sistema PostgreSQLf que se muestra en Figura 9. Las propiedades de calidad del modelo de calidad ISO/IEC 25010, corresponden con los RNF identificados en el modelo del dominio, donde se observó que la Reusabilidad, el Funcionamiento adecuado y el Rendimiento del sistema gestor son las cualidades no funcionales mínimas prioritarias.

A partir del MN y siguiendo las disciplinas del DAOMAC, se aplicaron una serie de transformaciones cumpliendo las reglas de correspondencia semántica entre BPMN y GRL [23], descritas en la Figura 10.

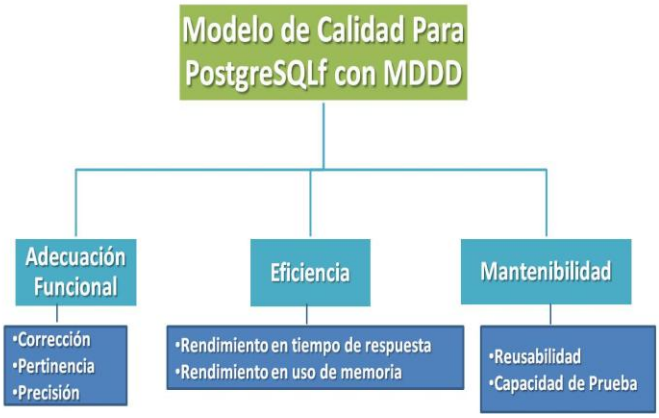


Figura 9: Adaptación del Modelo de Calidad ISO/IEC 25010 para PostgreSQLf con MDDD

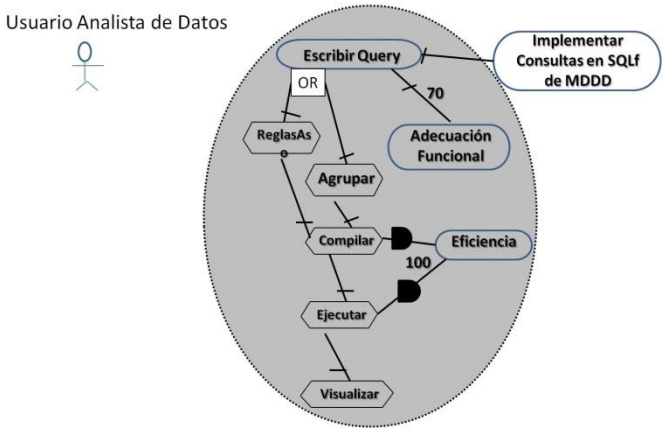


Figura 11: Modelo de Metas de PostgreSQLf en GRL

Una vez aplicadas las transformaciones se obtuvo el modelo de metas, mostrado en la Figura 11, para el *Pool* Usuario en su rol de Analista de Datos, correspondiente a la meta principal del MN, denominado Modelo de Metas Implementar Consultas para MDDD en SQLf. Se detectaron las Metas No Funcionales que estén asociadas a aspectos no funcionales del negocio en BPMN y se asocian con las Metas No Funcionales o *softgoals* de GRL, por lo qu e se asoció la tarea funcional EscribirQuery a la «MNF» *Adecuación Funcional* así como la «MNF» *Eficiencia* a la tarea funcional *Compilar*.

Luego el Modelo SIG se construyó para la Meta No Funcional Transversal Principal *Eficiencia*, el cual se observa en la Figura 12. En primer lugar se identificó las metas funcionals y no funcionales transversales en el modelo de metas anterior para el actor Usuario en su rol de Analista de Datos; se puede notar que la adecuación Funcional no es una Meta No Funcional Transversal ya que solo entrecruza a la Meta Funcional EscribirQuery. Adicionalmente se asignaron prioridades a las metas no funcionales, donde se evidenció que la Eficiencia tiene alta importancia, en vista que para el sistema que el tiempo de respuesta y uso de memoria es fundamental para implementar las consultas de mineria descriptiva con grandes

Regla	Lenguaje BPMN			Lenguaje GRL			Regla para la Correspondencia
	Termino	Definición	Notación Gráfica	Termino	Definición	Notación Gráfica	
1	*Process* o Proceso	Actividad realizada dentro o a través de empresas u organizaciones [48].	Texto	*Hardgoal* o Meta Funcional	Condición o estado de un asunto que a los *stakeholders* les gustaría lograr [30].		Un *Proceso* de BPMN representa una *Meta Funcional* en GRL.
2	*Subprocess* o Subproceso	Agregación de actividades que son incluidas dentro de un Proceso [48].					Un *Subproceso* de BPMN representa una *Meta Funcional* que ha sido descompuesta en GRL en otras metas de nivel inferior
3	*Pool*	Participantes (entidades o roles de negocio) [48].		*Actor*	Entidades activas que llevan a cabo ciertas acciones para lograr metas [30].	Actor	Un *Pool* de BPMN representa un *Actor* en GRL.
4	*Lane* o Carril	Organizan y categorizan actividades de un *Pool* [48].		*Role* o Papel	Caracterización abstracta del comportamiento de un actor [1].	Actor «Rol»	Un *Carril* de BPMN es descrito como el *Papel* de un Actor concreto en GRL.
5	*Task* o Tarea	Es una actividad atómica [48].	Tarea 1	*Task* o Tarea	Es una manera particular de hacer algo [30].		Una *Tarea* de BPMN representa una *Tarea* particular en GRL.
6	Labeled Association o Asociación Etiquetada	Muestra la asociación entre una tarea o subproceso funcional y una Meta No funcional, «MNF»	«MNF»	*Softgoal* o Meta No Funcional	Es una condición o estado de un asunto del mundo que a los *stakeholders* les gustaría lograr (ITU-T, 2012).		Una Asociación Etiquetada *de* BPMN relaciona una tarea funcional o subproceso de BPMN con *un Softgoal* en GRL.
7	*Data Object* u Objeto de Datos	Artefacto que suministra información a las actividades a ser realizadas [48].		*Resource* o Recurso	Entidad física o informacional que expresa necesidades [30].		Un *Objeto de Datos* en BPMN representa un *Recurso* de GRL.
8	*Gateway* o Compuerta	Controlan la interacción, convergencia o divergencia dentro de un proceso [48].		*Task decomposition link* o Vínculo de descomposición de tarea	Tipo de vínculo que descompone una tarea en un *hardgoal, subtask, resource* o *softgoal* [30].		Una *Compuerta* en BPMN representa un *Vínculo de descomposición de tareas* (And/Xor/Ior) de GRL.
9	*Sequence Flow* o Flujo de Secuencia	Muestra la secuencia de actividades que son realizadas en un proceso [48].		*Means-Ends link* o Vínculos Medio-Fin	Relación binaria entre un fin y el medio para lograrlos. El "medio" representa una tarea y el "fin" una meta [30].		Un *Flujo de Secuencia* en BPMN representa un *Vínculo Medio-Fin* en GRL.
10	*Message Flow* o Flujo de Mensaje	Muestra el flujo de mensaje entre dos entidades [48].		*Dependency link* o Vinculo de dependencia	Relación intencional entre dos actores [30].		Un *Flujo de Mensaje* en BPMN representa un *Vínculo de Dependencia* entre dos actores en GRL.

Figura 10: Reglas de Correspondencia Semántica entre BPMN y GRL [23][39]

volúmenes de datos.

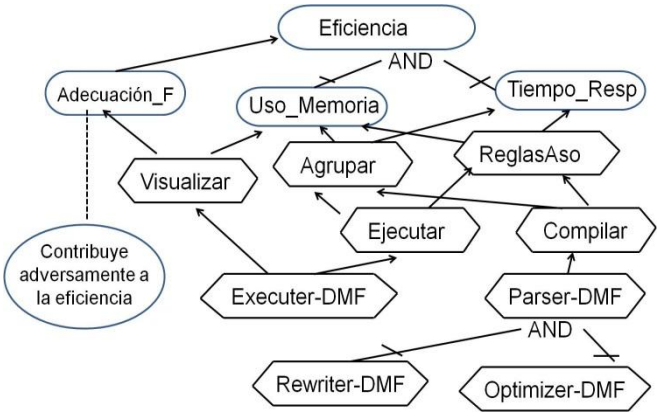


Figura 12: Modelo SIG para la Meta No Funcional Transversal Eficiencia

Finalmente, a partir de estos modelos se construye la Arquitectura Inicial orientada a metas, aspecto y calidad del Sistema PostgreSQLf con capacidades de Minería de Datos Difusa. En ella, las metas funcionales y las metas no funcionales transversales del Modelo de Metas, corresponden a Componentes Funcionales y Componentes de Aspectos (propiedades de calidad), respectivamente.

En la Figura 13, se muestra la AI, orientada a metas, aspectos y calidad, propuesta para PostgreSQLf. Ésta se obtuvo luego de aplicar el proceso de diseño arquitectónico DAOMAC, habiendóse aplicado técnicas orientada a metas, aspecto y calidad. La particularidad, de esta propuesta arquitectónica, son los componentes de aspecto relativos a las propiedades de calidad («Aspect») del sistema, denominados Eficiencia y Adecuación_Funcional, que corresponden a las metas no funcionales transversales del modelo de metas, que surgieron de los requisitos no funcionales. Los demás componentes, los cuales no son etiquetados con «Aspect», se refieren a las metas funcionales del sistema. Éstos pueden ser mecanismos, módulos, bases de datos, programas ejecutables, entre otros, que representan una solución arquitectónica para la meta funcional planteada.

Los círculos azules indican las interfaces, lazos de unión entre varios componentes ("*provide/require*"), reflejan las interacciones entre los diferentes componentes del diagrama. La existencia de una conexión muestra que un componente hace uso de la interfaz de otro, posiblemente a través de una solicitud de servicio, envío de un mensaje o comunicación de información.

La Adecuación Funcional sólo aparece conectada con los componentes Visualizar, Escribir_Query_MDDD y Executor_MDDD, pues este requisito no funcional o meta, sólo es asociado a las metas funcionales correspondientes a estos componentes. Este requisito se refiere a la completitud, pertinencia, precisión y corrección funcional durante el uso del producto de software. En este caso particular, dicha meta es importante para estos componentes pues corresponde a la especificación de las consultas y a la visualización de resultados asociados a la extensión MDDD. Así como también, a la ejecución de tales consultas. La complejidad intrínseca a este tipo de consultas demanda interfaces adecuadas que faciliten estos procesos. A través de estas interfaces, el usuario debe ser capaz de entender y gestionar las capacidades

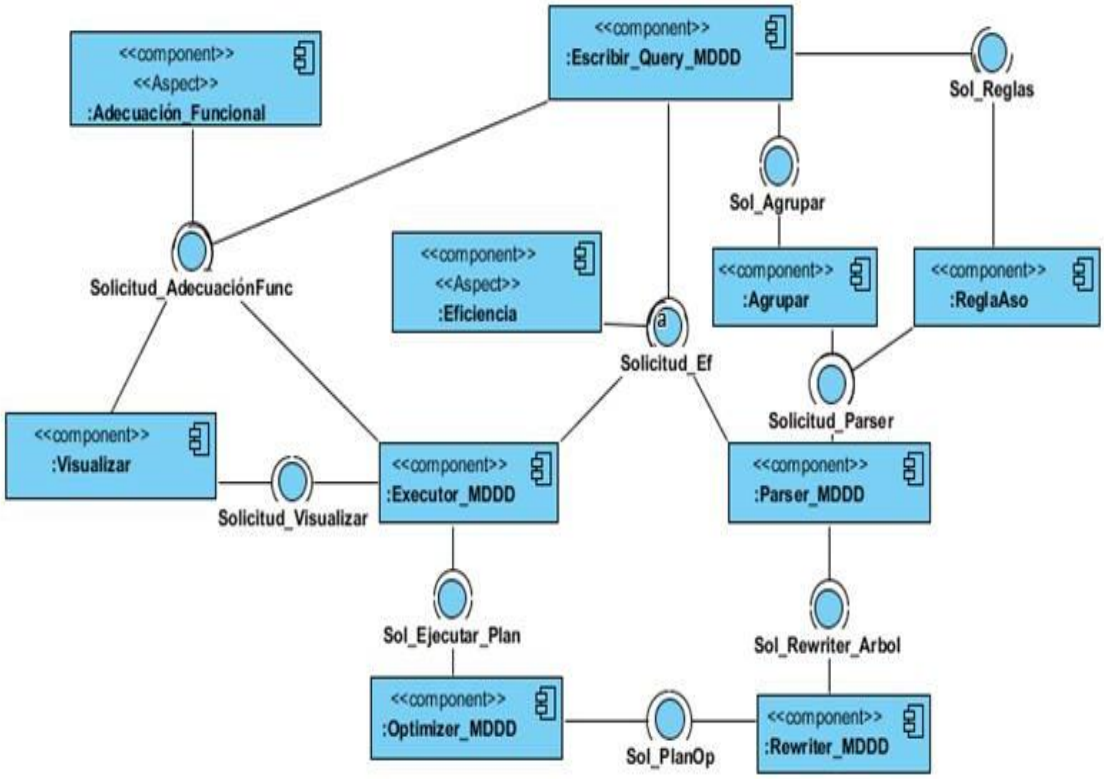


Figura 13: Arquitectura Inicial de PostgreSQLf en UML

vinculadas, a la minería de datos descriptiva difusa, que se hacen sobre los datos.

Durante la escritura de la consulta, por ejemplo, el conjunto de datos debe configurarse para poder aplicar los algoritmos de minería de datos difusa. El componente Escribir_Query_MDDD es entonces el ambiente para construir la sentencia a ser enviada a los demás componentes funcionales. Aquí, la eficiencia tiene que ver con el tiempo de respuesta y el uso de la memoria en la interfaz gráfica. Por otro lado, Visualizar es el componente que se refiere a la interfaz para observar el resultado final de una consulta MDDD. La eficiencia es un requisito de mayor importancia durante el análisis y ejecución de las consultas.

Los componentes Executor, Parser, Rewriter, Optimizer tienen una correspondencia directa con la estructura interna de PostgreSQLf (ver Figura 7), a los cuales se le agrega el sufijo "_MDDD", para indicar que han sido extendidos con capacidades de minería de datos descriptiva difusa.

ReglasAso es el componente que permite configurar una consulta que se va analizar usando reglas de asociación difusas. Mientras que Agrupar, permite configurar las consultas con agrupamiento difuso. En ambos se deben proveer mecanismos para especificar el tipo de algoritmo a usar durante la ejecución.

VI. Conclusiones y Trabajos Futuros

Se ha propuesto una arquitectura inicial o AI para el Sistema PostgreSQLf con capacidades de Minería de Datos Descriptiva Difusa, orientada a metas, aspectos y calidad, utilizando el modelo de diseño arquitectónico DAOMAC.

La AI construida considera el RNF de calidad Eficiencia, en términos de uso de recursos y tiempo de respuesta. Este requisito es un atributo de calidad primario (prioritario), basado en ISO/IEC 25010, relacionado con la meta funcional primaria de implementar consultas en SQLf para extraer grupos y reglas de asociación difusas, usando datos precisos almacenados en un Sistema Gestor de Bases de Datos.

Se propone extender el lenguaje SQLf con operadores para extraer reglas y grupos difusos, implementados en PostgreSQLf usando una arquitectura de acoplamiento fuerte que está basada en la arquitectura de referencia de PostgreSQL.

Se aplica el proceso de diseño arquitectónico DAOMAC, para la MNF de Eficiencia, con un resultado alentador. Se construye una Arquitectura Inicial y este resultado intermedio muestran la factibilidad de su aplicación.

Con esta arquitectura se espera beneficiar al proceso de análisis de asociación y de agrupamiento difuso con el rendimiento y escalabilidad deseados, considerando estos requisitos no funcionales desde altos niveles de abstracción como el Modelo de Negocio.

Como trabajos futuros, se han plateados los siguientes objetivos. Primero, refinar la Arquitectura Inicial construida hasta "valorar" la trazabilidad de los atributos de calidad propuestos desde el Modelo de Negocio. Segundo, aplicar la arquitectura para la totalidad de los requisitos funcionales y no funcionales del Sistema PostgreSQLf con facilidades de Minería de Datos Descriptiva Difusa. Tercero, comparar los resultados de aplicar la arquitectura a diversos casos de estudio que requieran otras funcionalidades novedosas.

Agradecimientos

Son muchas las personas que realizaron aportes de vital importancia para el logro de los objetivos planteados en el presente trabajo. En especial mención, extendemos nuestros agradecimientos al Dr. Leonid Tineo, Profesor Titular de la Universidad Simón Bolívar, Venezuela, por sus consejos y consideraciones técnicas ofrecidas oportunamente. Finalmente, agradecemos infinitamente al dador de toda ciencia y conocimiento, ¡El Señor creador de la vida! Quién nos ha dado los recursos y las fuerzas.

Referencias

[1] P. Bosc and O. Pivert, *SQLF: A Relational Database Language for Fuzzy Querying*. IEEE Transactions on Fuzzy System, vol. 3, no. 1, pp. 1-17, 1995.

[2] L. Tineo, *Extending RDBMS for Allowing Fuzzy Quantified Queries*. Lecture Notes in Computer Science, vol. 1873, pp. 407-416, 2000.

[3] R. Timarán, *Nuevas Primitivas SQL para el Descubrimiento de Conocimiento en Arquitecturas Fuertemente Acopladas con un Sistema de Gestion de Bases de Datos*. Tesis Doctoral. Universidad del Valle, Santiago de Cali, Colombia, 2005.

[4] P. Bosc and O. Pivert, *SQLf Query Functionality on Top of a Regular Relational Database Management System*. In: Knowledge Management in Fuzzy Databases, O. Pons, M. A. Vila, J. Kacprzyk (eds.), pp. 171-190, 2000.

[5] A. Aguilera, L. Borjas, R. Rodríguez, and L. Tineo, *Experiences on Fuzzy DBMS: Implementation and Use*, in proceedings of the XXXIX Latin American Computing Conference (CLEI 2013), vol. 1, pp. 478-485, Naiguatá, Venezuela, Octubre 2013.

[6] M. Goncalves and L. Tineo, *SQLf3: An Extension of SQLf with SQL3 Features*, in proceedings of the 10th IEEE International Conference on Fuzzy Systems (Fuzzy IEEE), Melbourne, Australia, December 2001.

[7] J. Ramírez and L. Tineo, *Un Mecanismo de Respuestas a Consultas en Presencia de Nulos*, Revista Venezolana de Computación (ReVeCom), ISSN: 2244-7040, vol. 2, no. 1, pp. 48-59, Diciembre 2015.

[8] A. Aguilera, L. Borjas, and R. Rodríguez, *Un Modelo de Proceso Heurístico para Implementación de SGDB Difusos*, Memorias de la Tercera Conferencia Nacional de Computación, Informática y Sistemas, ISBN: 978-980-7683-01-2, pp. 242-247, Valencia, Venezuela, Octubre 2015.

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

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

[11] C. Zapata, C. Jiménez, and J. Velásquez, *El Tratamiento de los Terminos Borrosos en Manejadores de Base de Datos Relacionales*, DYNA, vol. 70, no. 138, pp. 1-11, 2003.

[12] A. Urrutia, J. Galindo, and A. Sepúlveda, *Implementación de una Base de Datos Difusa con FIRST-2 y PostgreSQL*, XV Congreso Español sobre Tecnologías y Lógica Fuzzy, ESTYLF, pp. 199-204, Huelva, España, Febrero 2010.

[13] P. Bosc and O. Pivert, *On the Eficiency of the Alpha-cut Distribution Method to Evaluate Simple Fuzzy Relational Queries*, Advances in Fuzzy Systems-Applications and Theory, Fuzzy Logic and Soft Computing, Bouchon-Meunier, pp. 251-260, 1995.

[14] P. Bosc and O. Pivert, *On a Fuzzy Group-by Clause in SQLf*, in proceeding of the IEEE International Conference on Fuzzy Systems, pp 1-6, Barcelona, España, July 2010.

[15] R. Sabatino, *PostgreSQLf: Extensiones de Cuantificadores Difusos y Agrupamiento Difuso sobre el SGBD PostgreSQL*, Trabajo de Grado, Universidad de Carabobo, Valencia, 2010.

[16] A. Aguilera, J. Cadenas, and L. Tineo, *Fuzzy Querying Capability at Core of a RDBMS*, in Advanced Database Query Systems: Techniques, Applications and Technologies, L. Yan and Z. Ma (eds.), IGI Global. New York, USA, 2011, pp. 160-184.

[17] D. Garlan and M. Shaw, *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, 1996.

[18] C. Ghezzi, M. Jazayeri, and D. Mandrioli, *Fundamentals of Software Engineering*, Prentice Hall, 2002.

[19] IEEE, *IEEE Standard 1471-2000 – IEEE Recommended Practice for Architectural Description for Software-Intensive Systems, 2000*.

[20] 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.

[21] F. Losavio and Ch. Guillén. *Marco Conceptual para un Diseño Arquitectónico basado en Aspectos de Calidad*, Revista Universitaria Sapiens, vol. 7, no. 2, pp. 119-138, 2006.

[22] J. C. Guzmán, F. Losavio, and A. Matteo, *Comparación de Métodos para el Diseño Arquitectónico del Software que Consideran Metas, Aspectos y Estándares de Calidad*, Enl@ce: Revista Venezolana de Información, Tecnología y Conocimiento, vol. 10, no. 2, pp. 11-27, 2013.

[23] J. C. Guzmán, F. Losavio, and A. Matteo, *Del Modelo de Negocio a la Arquitectura del Sistema Considerando Metas, Aspectos y Estándares de Calidad*, Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software (RACCIS), vol. 3, no. 2, pp. 19-37, 2013.

[24] M. Mlitat and Z. Dai, *ISO/IEC 25010: Software Engineering-Software product Quality Requirements and Evaluation (SQuaRE)*, Quality models, 2011.

[25] M. Hammer and J. Champy, *Reengineering the Corporation: A Manifesto for Business Revolution*, Harper Collins, 1993.

[26] M. Hepp and D. Roman, *An Ontology Framework for Semantic Business Process Management*, in proceeding of the 8th International Conference Wirtschafts Informatik, pp. 1-18, Karlsruhe, Germany, 2007.

[27] A. Bluter, *Business Process Management Essentials*, Glintech, 2009.

[28] S. White, *Business Process Modeling Notation* (*BPMN*) 1.0. Business Process Management Initiative. 2004.

[29] P. Krutchen, *The Rational Unified Process: An Introduction*. Addison Wesley, 2000.

[30] T. Crusson, *Business Process Modeling Languag*, GLiNTECH, 2006.

[31] OMG. *Business Process Model and Notation* (*BPMN*) 2.0, http://www.omg.org/spec/BPMN/2.0. 2011.

[32] ITU-T, *ITU-T Z.151: User Requirements Notation* (*URN*)-*Language Definition, Recommendation*, 2012.

[33] OMG, *Software Process Engineering Metamodel* (*SPEM*) 2.0, http://www.omg.org/spec/SPEM/2.0/PDF. 2008.

[34] OMG, *Unified Modeling Language*™ (*UML*®) 2.0, http://www.omg. org/spec/UML, 2005.

[35] L. Chung, B. Nixon, and E. Yu, *Using Non-Functional Requirements to Systematically Select Among Alternatives in Architectural Design*, in proceeding of 1st Int. Workshop on Architectures for Software System, pp. 31-32, Washington, USA, 1995.

[36] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulus, *Non-Functional Requirements in Software Engineering*, International Series in Software Engineering, Springer, 2000.

[37] A. Lamsweerde, *Goal-Oriented Requirements Engineering: A Guided Tour*, in proceeding of the 5th Intl. Symp. Req. Eng. (RE'01), pp. 249-263, Toronto, Canada, August 2001.

[38] L. Chung, K. Cooper, and A. Yi, *Developing Adaptable Software Architectures Using Design Patterns: a NFR Approach*, Journal Computer Standards & Interfaces - 36 Special issue: Adaptable software architectures, vol. 25, no. 3, pp. 253-260, 2002.

[39] F. Losavio, J. C. Guzmán, and A. Matteo, *Correspondencia Semántica entre los Lenguajes BPMN y GRL*, Enl@ce Revista Venezolana de Información, Tecnología y Conocimiento, vol. 8, no. 1, pp. 11-29, 2011.

[40] H. Chen, H. Lim, and J. Xiao, (s.f.). *Concept architecture of PostgreSQL*.

[41] F. Yuan, J. She, Z. Fan, and J. J. Wu, *Concrete Architecture of PostgreSQL Backend*, http://www.cs.uwaterloo.ca/ f2yuan.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 13-25, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Diseño de Arquitecturas de Referencia para Líneas de Productos de Software Orientadas a Servicios: Revisión Documental Sistemática

Juan C. Herrera[1], Francisca Losavio[2], Oscar Ordaz[3]
jchr1982@gmail.com, francislosavio@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 Matemáticas, 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 general en la literatura y en particular en internet. El objetivo del trabajo es realizar una RDS en el tema de Líneas de Productos de Software Orientadas a Servicios (LPSOS), el cual integra los enfoques de Líneas de Productos de Software (LPS) y de Arquitecturas Orientadas a Servicios (SOA). La intención es que a partir de la RDS se identifiquen métodos (fases, etapas, actividades, artefactos y roles) para el diseño del artefacto principal de las LPS, la Arquitectura de Referencia (AR) siguiendo un modelo de arquitectura empresarial SOA y conformada por Servicios Web. Los resultados de la RDS servirán de guía para definir un proceso sistemático para el diseño de una AR en el contexto LPSOS. Esta arquitectura contemplará el tratamiento de requisitos de calidad en las etapas tempranas del desarrollo de LPSOS para garantizar su evolución, y el modelado de la variabilidad para la derivación de productos concretos, aspectos esenciales en el contexto de producción industrial de software.

Palabras Clave: Líneas de Productos de Software; Arquitectura Orientada a Servicios (SOA); Líneas de Productos de Software Orientadas a Servicios (LPSOS); Revisión Documental Sistemática (RDS); Arquitectura de Referencia; Servicios Web.

Abstract: A Systematic Review (SR) allows to extract knowledge about a research topic; this is done by formulating adequate questions from the large volume of information available in the literature in general and particularly searching on the Internet. The goal of the work is to perform an SR on the subject of Service Oriented Software Product Lines (SOSPL), which integrates the approaches of Software Product Lines (SPL) and Service Oriented Architectures (SOA). The intention is to use SR methods (phases, stages, activities, artifacts and roles) for the design the Reference Architecture (RA), the main artifact of LPS, following the SOA enterprise architecture model conformed by Web Services. The results of the SR will provide guidance to define a systematic process to design a RA in the SOSPL context. This architecture will cover the treatment of quality requirements at early stages of SOSPL development to ensure its evolution, and the variability modeling for the derivation of concrete products, which are essential issues in the context of software industrial production.

Keywords: Software Product Lines; Service Oriented Architecture (SOA); Service Oriented Software Product Lines (SOSPL); Systematic Review (SR); Reference Architecture (RA); Web Services.

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 *Ingeniería de Software (IS)*; surge entonces la tendencia al desarrollo basado en componentes reutilizables, para configurar nuevos sistemas con rapidez y de más bajo costo [1][2][3].

El enfoque de *Líneas de Productos de Software (LPS)* permite la construcción de soluciones individuales basadas en un repositorio de componentes de software reutilizables. El hecho de proporcionar soluciones individuales responde a diversas necesidades en el software con respecto a la funcionalidad, la tecnología subyacente 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, d) evolución, y e) disminución del tiempo de comercialización. Este enfoque representa un gran reto para los investigadores y desarrolladores, principalmente por la tarea de construir artefactos (*activos* o del inglés *assets*) suficientemente genéricos para poder ser reutilizados en la configuración de nuevos sistemas acordes con las necesidades de los clientes [2][3]. Para el abordaje adecuado de este enfoque, la *Ingeniería de Líneas de Productos de Software (ILPS)* ofrece métodos y técnicas eficaces para un desarrollo de software basado en la reutilización, con el fin de: a) apoyar arquitecturas de software configurables o instanciables 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 de la reutilización. La ILPS consta de dos ciclos de vida principales: *Ingeniería del Dominio (ID)* e *Ingeniería de la Aplicación (IA)* [8]. La ID abarca el análisis y la identificación del ámbito de aplicación de la LPS, 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 partes variables o puntos de variación (del inglés "variation points") que están presentes en la *Arquitectura de Referencia (AR)*, la cual es una arquitectura genérica basada en un esquema instanciable que dirige la derivación de productos concretos de la familia LPS.

Por otra parte, la *Arquitectura Orientada a Servicios* (del inglés *Service Oriented Architecture (SOA)*) [9][10][11] es un modelo de arquitectura empresarial distribuida, centrado en la comunicación en redes; su objetivo central es la integración y desarrollo rápido de sistemas empresariales en una organización o dominio específico. Un *servicio* se define como una unidad discreta de una funcionalidad del negocio, que está disponible a través de una *interfaz estándar* del servicio, mediante la cual el servicio se comunica con otros servicios [12]; el servicio puede ser visto como un componente de software reutilizable. Una implementación muy conocida de servicios son los Servicios Web, en el contexto de la WWW [9]. SOA especifica lineamientos para la construcción independiente de servicios alineados al negocio que pueden ser combinados a partir de procesos de negocio de alto nivel y trasladados a soluciones computacionales en el contexto empresarial. El valor real de SOA no es solo la provisión adecuada de servicios, sino más bien cuando los servicios reutilizables se combinan para "implementar" 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 [11][13][14], dando origen al enfoque de *Líneas de Productos de Software Orientadas a Servicios (LPSOS)*. La integración de los enfoques LPS y SOA requiere que las organizaciones de desarrollo de LPS apliquen la gestión de la variabilidad respecto a los servicios que son los componentes esenciales de SOA, y que los desarrolladores de sistemas orientados a servicios apliquen técnicas que usan los desarrolladores de LPS. Sin embargo esta tarea no es trivial; la necesidad de variación en cuanto a servicios, puede ser impulsada por las condiciones del mercado, oportunidades del negocio, las nuevas tecnologías y otros factores relacionados con la empresa. Esta necesidad puede expresarse en forma de objetivos de negocio establecidos por las organizaciones que desarrollan LPSOS [13]. Ahora bien, en un contexto de LPSOS, particularmente para la construcción de una AR, deben manejarse tres problemas principales: (1) la representación de la AR conformada por componentes y conectores [15], en donde los componentes son servicios y los conectores son mensajes entre servicios, incluyendo el traslado de servicios a componentes arquitecturales; (2) la gestión de las propiedades de calidad requeridas por los componentes que representan funcionalidades para asegurar la completitud y evolución de la AR; (3) la gestión de la variabilidad funcional y no funcional. Las soluciones a estos problemas determinarán los pasos a seguir para definir un método de diseño de una AR basada en servicios en un contexto LPS.

En los estudios anteriormente referenciados se han reflejado los problemas mencionados, aún no completamente resueltos, para algunos de los cuales este trabajo pretende proporcionar una guía para su solución. El objetivo de esta investigación, se centra principalmente en una Revisión Documental Sistemática de la literatura (RDS) [16] de trabajos recientes sobre diseño de una AR basada en servicios y configurada como una arquitectura de software [15] para LPS y no como una arquitectura empresarial, para identificar métodos sistemáticos de diseño que respondan a los tres problemas identificados anteriormente y que además contemplen:

- artefactos de entrada y de salida,
- enfoque de diseño utilizado para la AR,
- roles de quienes participan en las actividades del modelo de proceso del método.

Una RDS [16] 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 que aborde el tema de métodos de desarrollo de LPSOS, es indispensable para determinar los alcances y limitaciones de las propuestas encontradas. Para alcanzar este objetivo, nuestra RDS se apoya en un marco de evaluación conformado por la integración de los aspectos de interés de tres propuestas de evaluación [17][18][19], que abordan las siguientes temáticas: LPS, SOA y AR-SOA. Este marco de evaluación posteriormente será utilizado para comparar los "mejores" métodos encontrados por la RDS, es decir aquellos que satisfacen más del 50% del total de las 33 propiedades descritas en las Tablas VII, VIII, IX.

Este trabajo está estructurado de la siguiente manera, además de esta introducción: la Sección II presenta algunos aspectos a ser considerados para realizar la RDS; en la Sección III se define el marco integrado de evaluación; la Sección IV presenta el análisis de la RDS realizada; la Sección V discute los resultados obtenidos. Finalmente, la Sección VI presenta las conclusiones y las perspectivas.

II. Aspectos a Ser Considerados en la RDS

El enfoque LPSOS requiere considerar los siguientes elementos: la *Gestión de Procesos de Negocios (GPN)* o del inglés *Business Process Management (BPM)*, la gestión de la variabilidad funcional y no funcional, la representación de una

Arquitectura de Referencia Orientada a Servicios y la gestión de la calidad. Estos aspectos son considerados prioritarios en una RDS de dos Santos Rocha y Fantinato [7] que presenta enfoques para la aplicación conjunta de LPS y GPN, incluyendo el soporte de SOA para GPN, y en la cual se concluye que la combinación LPS, BPM y SOA es factible y beneficiosa para la construcción de un proceso sistemático integrado para elaborar una LPSOS. A continuación se describen brevemente cada uno de estos aspectos.

A. Gestión de Procesos de Negocios

GPN en [20] define un *proceso de negocio* como un conjunto de actividades que se realizan en coordinación con un entorno organizacional y técnico. Estas actividades se desempeñan de manera conjunta para realizar los objetivos del negocio. Cada proceso de negocio es generado por una sola organización, pero puede interactuar con los procesos de negocios generados por otras organizaciones. Las partes fundamentales de GPN son su ciclo de vida y el lenguaje de modelado de procesos de negocio del inglés *Business Process Management Language (BPML)* [20]. Existen varios BPMLs concebidos en la academia y la industria que se utilizan para representar el flujo de trabajo y los artefactos y/o producidos utilizados durante la realización de las actividades en el ciclo de vida. Algunos de estos son el Diagrama de Actividades (UML 2.0) y la Notación de Procesos de Negocios [21], los cuales fueron evaluados por un framework en [22]. Esta evaluación determinó que el *Business Process Modeling Notation (BPMN)* es uno de los BPMLs más utilizado en la fase de diseño de un ciclo de vida GPN. Representa los procesos como actividades centradas en el control de flujo. El lenguaje de modelado debe proporcionar elementos notacionales y mecanismos de apoyo a aspectos tales como la especificación de las actividades, el flujo de secuencia, el flujo de datos y los eventos. En cambio, no soporta especificaciones de aspectos o requisitos no funcionales. Por otra parte, la ejecución de los procesos de negocio, además de la selección de una plataforma de ejecución, como por ejemplo SOA, puede requerir de otros lenguajes. Por ejemplo el Lenguaje *Business Process Execution Language (BPEL)* es utilizado para la ejecución de los procesos de negocio a través de la composición de servicios por orquestación [20]. No obstante, la OMG[1] con la definición de la semántica de cada elemento de la notación en BPMN 2.0, permitió la posibilidad de ejecutar los modelos sin necesidad de algún otro lenguaje de apoyo. Por ejemplo la herramienta Activiti[2] que implementa BPMN. En la actualidad, la mayoría de los enfoques de GPN enfatizan el desarrollo de un proceso de negocio, y a partir de este mismo se derivan muchas variantes, que están especializadas según diversas necesidades de la organización. Los procesos de negocio pueden tener piezas comunes para un grupo (es decir, la familia) de diferentes casos de aplicación, implicando cierta variabilidad en la selección de las actividades de un proceso de negocio. Este aspecto podría ser utilizado para LPS, donde una funcionalidad común puede ser desarrollada como un activo reutilizable para la creación de nuevas variantes de los procesos de negocio en un dominio [13][20].

[1] Object Management Group
[2] http://activiti.org

B. Gestión de la Variabilidad

La *Gestión de la Variabilidad* (del inglés *Variation Management*) [13] se aplica tanto a SOA como a LPS; los puntos de variación pueden ser implementados ya sea por un único servicio (donde una interfaz de servicios puede ofrecer parametrización por *QoS (Quality of Service)* o algún otro mecanismo) o a través de servicios similares que traten cada variación. Mohabbati [11], indica que la variabilidad es gestionada teniendo varios servicios que tienen la misma funcionalidad, pero difieren en su QoS.

C. Arquitecturas de Referencia en LPS y SOA

Otro aspecto importante corresponde a la construcción de la AR, y en particular a las *Arquitecturas de Referencia Orientadas a Servicios (AROS)* que apoyen la reutilización sistemática de los activos principales para la configuración de productos de software concretos. En la RDS [23] se indica que existen diferencias entre las *Arquitecturas de Líneas de Productos (ALP)* y las AR: ALP tienden a ser menos abstractas que las AR, pero más abstractas que las arquitecturas de software concretas, es decir, las ALP son un tipo de AR [11][23]. Con respecto a nuestra RDS, AR y ALP son consideradas similares. Las AROS se han propuesto como un tipo de AR; lo que debe estar claro es que las AROS en el contexto SOA son un modelo de arquitecturas empresariales para la integración de aplicaciones en la empresa, no son arquitecturas de software "reales" en el sentido de Garlan y Shaw [15]. En cambio en el contexto LPS, AROS son arquitecturas de software genéricas e instanciables a partir de un modelo de variabilidad, para generar sistemas concretos orientados a servicios de la familia LPS. Una AROS-LPS será entonces para nosotros una AR en un contexto de LPSOS. Varias AROS bajo SOA han sido propuestas para sistemas de gobierno, entornos de trabajo colaborativo, o sistemas de atención de la salud; generalmente no son AROS "puras" solo conformadas por servicios, sino que incluyen componentes que no son servicios. Una RDS realizada sobre las AROS bajo LPS en [7], indica que faltan directrices sobre cómo definirlas, y en particular como definir las AROS que soporten un alto grado de variabilidad [11].

D. Gestión de la Calidad en Servicios

En los actuales momentos no se concibe un sistema de software que no tenga la calidad necesaria que satisfaga las expectativas de los usuarios finales. Más aun, las propiedades de calidad para una LPSOS son fundamentales para garantizar su completitud y carácter evolutivo. Por ello, es vital abordar las características de calidad de estos sistemas desde una etapa temprana en el ciclo de vida del desarrollo de software. Particularmente esto se debe hacer en el ciclo de vida de la ID en el proceso de una *ILPS Orientada a Servicios* (ILPSOS). Los servicios exponen propiedades de calidad de bajo nivel a las que responden en tiempo de ejecución, para intercomunicar con otros servicios; los valores de los QoS se ofrecen a través de parámetros. En general, lo que ocurre con las propiedades de calidad de alto nivel a las cuales también deben responder los servicios compuestos, cuando representan componentes funcionales en el caso de AROS, no se trata en detalles. Los *Servicios Web* (en inglés *Web Services (WS))* son tipos de servicios que pueden ser descubiertos, especificados y accedidos utilizando XML y protocolos Web estándar. El

elemento central de un WS está en la definición estándar de su interfaz en WSDL[3], que contiene entre otras informaciones, los QoS. La descripción de un WS tiene dos partes: - una definición abstracta que describe el servicio como un componente, con su interfaz y operaciones; - una definición que describe los enlaces concretos de las operaciones hacia puntos de acceso, que contienen una descripción de los datos, una dirección física y la información del protocolo de comunicación [24]. En un entorno de una aplicación que utiliza WS, se distinguen tres funciones: intermediario, proveedor y consumidor de servicios. El proveedor registra un servicio en el intermediario o "broker", por ejemplo en un repositorio UDDI[4]. El consumidor descubre un servicio gestionado por el intermediario y llama al servicio del proveedor. Todas las partes utilizan protocolos tipo SOAP[5], un protocolo estándar basado en XML para el manejo de respuestas y solicitudes de WS [24]. Está claro que un WS, como componente de una aplicación, también debe responder a propiedades de calidad de alto nivel, las cuales no necesariamente son observables en ejecución, por ejemplo persistencia de la información y políticas de confidencialidad.

III. Marcos para la Evaluación de Métodos para el Desarrollo de AR-LPS, SOA, AR-SOA

A. Framework para la Evaluación de los Métodos

El marco de evaluación o *framework*, constituido por las Tablas I, II y III, es utilizado como herramienta de análisis. La Tabla I muestra los criterios para evaluar métodos de diseño AR-LPS. La Tabla II muestra criterios para evaluar métodos de desarrollo de sistemas orientados a servicios bajo SOA. La Tabla III presenta criterios para evaluar métodos de desarrollo de AR-SOA. El objetivo de este framework de evaluación no es la comparación de métodos, sino proporcionar un panorama de métodos actuales para tomar decisiones en cuanto a un método general para el diseño de una arquitectura AROS-LPS.

AROS contempla tres estrategias de desarrollo:

- *bottom-up o ascendente (reactivo o extractivo):* derivar los servicios a partir de componentes existentes, es decir de aplicaciones heredadas o de una AR ya realizada, como por ejemplo la identificada a partir de un proceso "bottom-up" que considera la refactorización de productos existentes del mercado.
- *top-down o descendente o proactivo:* derivar los servicios a partir de la especificación del modelo del dominio expresado mediante procesos de negocios.
- *middle-out o meet-in-the-middle o híbrido:* partir de la especificación de los procesos de negocio como orquestaciones de servicios y relacionarlos con los componentes de una AR ya existente.

Para la RDS de este trabajo, se buscan los métodos o enfoques existentes (estado del arte) que describan de forma sistemática las etapas, actividades, roles y artefactos necesarios para realizar una AR en un contexto LPSOS (AROS-LPS), utilizando cualquiera de los tres frameworks mencionados [17][18][19].

La aplicación del marco de evaluación proporcionará la base para la definición de categorías detalladas de elementos o aspectos presentes en un criterio. Mediante un conjunto de preguntas, este estudio intenta abordar el alcance de los métodos para identificar las diferencias y similitudes que estos presentan. Como no existe un marco específico para la evaluación de enfoques LPSOS, se van a combinar los frameworks de las Tablas I, II y III, en un marco único que incluya todos los criterios de evaluación. A continuación se describen en detalles los criterios:

Criterios del marco de evaluación para métodos de desarrollo de AR-LPS [17]:

Hay cuatro criterios esenciales para la evaluación de estos métodos:

- *Contexto del método:* situación del problema, objetivo específico, ciclo de vida abordado.
- *Usuario del método:* personas que solucionan el problema, que rol desempeña, guía proporcionada para aplicar el método.
- *Componentes del método o proceso para la resolución del problema:* 1) modelos subyacentes, 2) lenguaje, 3) definición de los pasos, su secuencia, 4) artefactos, 5) vistas arquitecturales, como el modelo "4+1 vistas" de Kruchten [25]), 6) variabilidad (nivel de abstracción donde la variabilidad es manifiesta: a nivel de proceso de negocio, a nivel de los componentes), 7) herramientas que facilitan la realización del método.
- *Gestión de la calidad (nuevo criterio):* ¿Cómo el método soporta la especificación de las propiedades de calidad y/o requisitos no funcionales?; el aseguramiento de la calidad del producto es una actividad crucial para el éxito de la industria del software, pero es más importante cuando se trata del desarrollo de la LPS, dado que la reutilización masiva de activos de software hace que sus atributos de calidad (o propiedades medibles de un artefacto de software) impacten en la calidad de los productos de la LPS [26].
- *Validación del método:* casos de estudios donde se compruebe la efectividad del método, el estilo arquitectural elegido satisface las expectativas de los usuarios finales.

Criterios del marco de evaluación para métodos de desarrollo de sistemas bajo el modelo SOA [18]:

Hay cinco criterios esenciales para la evaluación de un método para desarrollo de sistemas o aplicaciones aisladas bajo el modelo SOA de arquitectura empresarial, que no involucran AR sino una arquitectura concreta de un producto:

- *Concepto de servicio:* para analizar si los conceptos de servicio, servicios de negocios y software, son soportados; un servicio de negocio es un conjunto específico de acciones que son llevadas a cabo por una organización, un servicio de software expone las funcionalidades de la

[3] Web Services Definition Language
[4] Universal Description, Discovery and Integration
[5] Simple Object Access Protocol

aplicación que pueden ser reutilizados y compuestos basados en las necesidades u objetivos del negocio, por lo tanto un servicio de software soporta la ejecución de un servicio de negocio.

- *Estrategia de realización o desarrollo:* top-down, bottom-up, middle-out o meet-in-the-middle que combina ambos; esta última estrategia es la más recomendada para implementar SOA.
- *Cobertura del ciclo de vida* o fases principales para el desarrollo bajo SOA; según SOMA [27] son: identificación, especificación y realización.
- *Grado de granularidad del análisis:* el nivel de detalle utilizado para describir el método, que va desde un nivel bastante descriptivo, con gran cantidad de detalles, hasta un nivel más abstracto, para describir un proceso más general, flexible y adaptable.
- *Accesibilidad y validez:* Indica que el método debe proveer documentación accesible y poder ser validado.

Criterios del marco de evaluación para métodos de desarrollo de AR-SOA [19]:

Las arquitecturas de software [15] juegan un papel importante en la determinación de la calidad del sistema, ya que forman la columna vertebral de cualquier sistema intensivo de software exitoso; el estilo arquitectónico garantiza por sí solo el cumplimiento de propiedades de calidad globales del dominio. En el contexto LPS, las AR reúnen el conocimiento del dominio y son los instrumentos que proveen esquemas reutilizables. Entre los principales beneficios de AR, se encuentran una mejor comprensión del dominio, el establecimiento de un vocabulario común, la reutilización de activos, la consistencia en la representación del sistema, y un menor tiempo de comercialización. SOA, como modelo de *arquitectura de referencia empresarial (AR-SOA)* ha tomado la atención en los últimos años. Facilita la integración mediante la interoperabilidad de servicios, la escalabilidad y la reutilización, ya que los sistemas basados en SOA son independientes respecto al lenguaje de programación y a la plataforma de ejecución utilizada. El estilo arquitectónico de sistemas de software empresariales orientados a servicios "puros", que no involucran otro tipo de sistemas no orientados a servicios, es basado en eventos [15], siguiendo un modelo cliente–servidor para la distribución y la comunicación en redes.

El marco de evaluación AR-SOA surge de responder la pregunta: ¿Cuáles características de SOA han sido consideradas durante el diseño y desarrollo de la AR-SOA? Se presentan tres criterios:

- *Publicación del servicio:* La AR-SOA debe permitir la publicación de la descripción de los servicios directamente a los consumidores, a través de intermediarios, por ejemplo UDDI.
- *Calidad del servicio (QoS):* La AR-SOA debe permitir utilizar mecanismos para capturar, controlar, registrar y señalar el incumplimiento de requisitos no funcionales descritos en los acuerdos de servicio por su interfaz. Estos acuerdos especifican las características de calidad que proporciona el servicio en ejecución, como el tiempo de respuesta, rendimiento, disponibilidad, entre otros.
- *Composición de los servicios:* es el proceso de construir un nuevo servicio a partir de servicios existentes [9]. Todos los servicios se comunican por paso de mensajes, a través de protocolos. La AR-SOA debe permitir la composición de los servicios: a través de los procesos de negocio, mediante una orquestación coordinada de servicios que enfatiza el paso de mensajes, o mediante la coreografía, más orientada hacia los aspectos funcionales que el servicio debe proporcionar.

B. Criterios y Elementos del Marco Único de Evaluación basado en los Tres Framework AR-LPS, SOA y AR-SOA

El objetivo del *Marco Único de Evaluación (MUE)* no es solo proporcionar una visión general de los actuales métodos de ingeniería para la LPSOS, sino además indagar cómo los métodos difieren en cuanto a criterios de diseño**.** Esto es especificado en la sección del informe de la RDS, donde se muestran los resultados de nuestra evaluación. La aplicación de MUE proporciona la base para la definición detallada de los elementos presentes en los criterios. Mediante preguntas, este marco intenta abordar por ejemplo, madurez, sentido práctico y alcance de los métodos para encontrar las similitudes y diferencias, además, de cómo se identifican los servicios a partir del modelo de procesos de negocio, y cuáles son los componentes que constituirán una AROS-LPS y que implementarán los servicios, incluyendo la relación entre estos componentes y el modelo de variabilidad.

A continuación, las Tablas I, II, III describen criterios y elementos que conforman el MUE, con sus respectivas preguntas de análisis. Estas preguntas ayudan a la captura de información relevante para cada uno de los criterios de evaluación. El MUE se deriva integrando todos los criterios de evaluación y será utilizado en la RDS para efectuar el análisis de los artículos seleccionados como *estudios primarios*: son aquellas investigaciones relevantes que quedan para el análisis, luego de haberse aplicado los criterios RDS de inclusión, exclusión y calidad:

Tabla I: Criterios y Elementos de Evaluación para Métodos de Desarrollo de AR-LPS, Adaptado de [17]

Criterios	Elementos	Preguntas
Dominio de la LPS	Ámbito	¿Cuál es/son el/los dominio(s) de la LPS en la que se enfoca el método? Ejemplo: dominio de los sistemas de información.
Contexto del método	Objetivo específico	¿Cuál es el objetivo específico del método?
	Fase(s) de línea de productos	¿Qué fases de la LPS cubre el método? Por ejemplo, el método abarca solamente las fases de análisis y diseño en el ciclo de vida de la ID.
	Entradas del método	¿Cuál es (son) la (s) entrada (s) del método?
	Salidas del método	¿Cuáles son la (s) salida (s) del método?
Usuario del	Grupo	¿Quiénes son los actores que

método	destinatario	intervienen en el método?
	Motivación	¿Cuáles son los beneficios que percibe el usuario al utilizar el método?
	Habilidades necesarias	¿Qué habilidades necesita el usuario para cumplir las tareas requeridas por el método?
	Orientación	¿Cómo el método guía al usuario mientras se aplica el método?
Componentes del método	Estructura del método	¿Cuáles son los pasos, fases y/o actividades del proceso de diseño que se utilizan para llevar a cabo un objetivo específico del método?
	Artefactos	¿Cuáles son los artefactos creados y gestionados por el método?
	Vistas arquitecturales	¿Cuáles son las vistas arquitecturales que se consideran en el método?
	Lenguaje/ notación	¿El método define un lenguaje o notación para representar los modelos, diagramas y otros artefactos que produce?
	Variabilidad	¿Cómo el método soporta la expresión de la variabilidad funcional y no funcional?
	Herramientas	¿Cuáles son las herramientas y/o técnicas que apoyan el método?
Gestión de la calidad en el método	Propiedades de calidad	¿Cómo el método soporta la especificación de las propiedades de calidad y/o requisitos no funcionales?
Validación del método	Madurez	¿Se ha validado el método en casos de estudios prácticos industriales?
	Calidad de la arquitectura	¿Cómo el método valida la calidad del diseño de la AR-LPS?

Tabla II: Criterios y Elementos de Evaluación para Métodos de Desarrollo de Sistemas bajo el Modelo SOA, Adaptado de [18]

Criterios	Elementos	Preguntas
Concepto de servicio	Servicios de negocios	¿Cuál es el concepto de servicio soportado?
	Servicios de software o WS	
	Ambos	
Estrategia de realización o desarrollo	Top-down	¿Cuál es el tipo de estrategia utilizada para el desarrollo bajo SOA?
	Bottom-up	
	Middle-out	
Cobertura del ciclo de vida	Identificación	¿Cuál es el grado de cobertura con relación al ciclo de vida SOA?
	Especificación	
	Realización	
Grado de granularidad del análisis	Bajo	¿Cuál es el grado de granularidad del nivel de abstracción del método?
	Moderado	
	Alto	
Accesibilidad y validez	Documentación no accesible	¿La documentación del método es apropiada y sirve de guía para su uso en la práctica?
	Parcialmente documentado	
	Bien documentado (Caso de estudio)	

Tabla III: Criterios y Elementos de Evaluación para Métodos de Desarrollo de AR-SOA, Adaptado de [19]

Criterios	Elementos	Preguntas
Publicación del servicio	Directamente a los consumidores	¿La AR-SOA permite la publicación de la descripción de los servicios?
	A través de intermediarios	
Composición de los servicios	Orquestación de servicios	¿La AR-SOA permite que el desarrollo de software pueda ser construido por medio de la composición de servicios?
	Coreografía de servicios	
Calidad del Servicio	Cumplimiento de los requisitos no funcionales de alto nivel	¿La AR-SOA permite el uso de mecanismos para validar que se cumplan los requisitos no funcionales de alto nivel requeridos por la funcionalidad que un servicio desempeña?
	Cumplimiento de los requisitos no funcionales de bajo nivel (QoS)	¿La AR-SOA permite el uso de mecanismos para validar que se satisfagan los requisitos no funcionales de bajo nivel (QoS) descritos en los acuerdos de su interfaz?

IV. Revisión Documental Sistemática (RDS)

Kitchenham [16] indica que la gestión del proceso de revisión sistemática se fundamenta en tres fases principales: Planificación, Realización e Informe de la revisión. En la Tabla IV se describen las fases, las etapas de cada fase y los artefactos generados en el proceso RDS.

Tabla IV: Proceso de Revisión Documental Sistemática

Fases	Etapas	Artefactos
Planificación de la revisión	Identificación 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 estudios relacionados.	Estudios aceptados
	Selección de estudios primarios.	
	Evaluación de la calidad de los estudios.	
	Extracción de los datos.	Modelo de datos (checklist)
	Síntesis de los datos.	
Informe de la revisión	Especificación de mecanismos de difusión.	Framework de evaluación
	Presentación de 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: Revisar los procesos de desarrollo actuales con el fin de proporcionar una guia para el diseño de una AR para LPSOS, en nuestro caso. Ademas, de identificar problemas pendientes por resolver y posibles áreas para la mejora de los procesos de desarrollo.

2) Desarrollo del Protocolo: El protocolo es el plan o conjunto de pasos a seguir en un estudio, constituido por las preguntas de investigación, las estrategias de búsqueda, los 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 y sintesis de los datos.

Las siguientes *Preguntas de Investigación (PI)* fueron consideradas en inglés; aquí las colocamos en castellano:

- PI1. ¿Cuáles métodos que siguen enfoques “bottom-up” y/o “top-down” existen (estado del arte) para pasar de (conversión) una arquitectura basada en componentes a una arquitectura basada en WS o como llevar a cabo esta correspondencia?

- PI2. ¿cómo se expresan los requisitos de calidad de las funcionalidades respecto a los QoS de los WS?

Con relación a la estrategia de búsqueda, las fuentes de búsqueda seleccionadas para encontrar los estudios primarios son los siguientes: Google Academic y ResearchGate.

Se identificaron las palabras claves de búsqueda (descriptores): Software Product Line, Service-Oriented Architecture, Methods and Tools Design, Reference Architecture, Web Services, Business Process, Non-Functional Properties, Quality of Service; combinándolas con operadores lógicos AND, OR; para capturar los estudios primarios.

Los *Criterios de Inclusión (CI)* utilizados para incluir los estudios relevantes en esta RDS son:

- CI1: Estudios publicados entre 2008 y 2015
- CI2: Que aborde la arquitectura para el desarrollo de Líneas de Productos de Software para Familias de Productos Orientados a Servicios utilizando SOA.
- CI3: Que en su contenido incluya al menos el ciclo de vida de la ID para LPSOS.
- CI4: Que en su contenido describa un método de desarrollo para LPSOS.

Los *Criterios de Exclusión (CE)* son:

- CE1: Que no aborde la integración de LPS y SOA como alternativa para el desarrollo de productos de software.
- CE2: Que esté escrito en un lenguaje diferente al inglés o castellano y que no esté disponible en formato PDF; o que tenga como contenido solo una presentación en un congreso o workshop realizado por los autores.
- CE3: Si existen trabajos repetidos, se elige uno de ellos.

En la selección inicial de las publicaciones entre los resultados de búsqueda, se considera la lectura de los títulos, palabras claves y resúmenes de las publicaciones encontradas. Las publicaciones serán almacenadas en carpetas independientes de acuerdo a la expresión de búsqueda utilizada, y de esta forma facilitar la ubicación de los estudios para realizar el análisis con mayor fluidez y comodidad.

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.

Para estandarizar la forma en que la información estará representada, se definió un formato (modelo de datos) para recopilación de los datos en los estudios seleccionados. Las Tablas VII, VIII, IX muestran la descripción de las propiedades de extracción o "checklist" que serán utilizadas a lo largo de la RDS. Las Tablas X y XI muestran los hallazgos encontrados de la extracción de los datos relevantes en los estudios.

El modelo de datos será utilizado en la fase de Realización de la revisión, para seleccionar los mejores estudios, y luego en la fase de Informe de la revisión, estos estudios serán analizados a través del MUE que se describe en la Sección III.

B. Realización de la Revisión

En esta fase se sigue el protocolo previamente establecido. Como resultado de la búsqueda, 331 estudios fueron localizados, de los cuales solo 21 fueron aceptados inicialmente para su análisis a través de tres "checklists", las Tablas VII, VIII y IX respectivamente. La Tabla V a continuación, muestra un resumen de los resultados indicando la cantidad de estudios localizados, aceptados y rechazados.

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

Fuentes de datos	**Artículos**		
	Aceptados	***Rechazados***	***Localizados***
Google Academic	12	283	295
ResearchGate	9	27	36
Total	**21**	309	331

De los estudios localizados se descartaron aquellos artículos cuyos contenidos eran idénticos, aun cuando sus identificadores eran diferentes. También, se identificaron estudios similares, que abordaban el mismo proceso y en los cuales participaban los mismos autores, aunque el titulo del estudio era distinto; para estos casos, se tomó la decisión de elegir el estudio más completo y actualizado. La Tabla VI muestra los estudios primarios aceptados según el tipo de evento y año de publicación.

Tabla VI: Relación de Estudios Primarios Relativos a Procesos de Desarrollo para LPSOS según Tipo y Año de Publicación

Año	**Tipo de evento**			**Total**
	Conferencia	***Revista***	***Tesis***	
2008	2	1		3
2009	2			2
2010	2	2	1	5
2011	5	1		6
2012	1			1
2013			1	1
2014	1	1		2
2015		1		1
Total	**13**	**6**	**2**	**21**

Los 21 estudios primarios describen un proceso para el desarrollo de una LPSOS. A continuación, un análisis más detallado se llevó a cabo en cada uno de ellos. Este análisis se realizó mediante la aplicación de dos "checklists" (Tablas X y XI) respecto a las propiedades que nos interesan evidenciar en los estudios (Tablas VII, VIII y IX). Esto permite caracterizarlos con relación a las propiedades presentes en sus respectivos contenidos. Se utiliza la nomenclatura (++), (+), (-), que significa que la calidad de la documentación fue buena, regular con un valor en ambos casos de 1 punto c/u o que carece de esa propiedad con un valor de 0 punto, respectivamente. Este aspecto también influyó en la decisión sobre cuales estudios fueron seleccionados para ser evaluados por el MUE.

Tabla VII: Descripción de las Propiedades: Métodos para AR-LPS

Propiedad	Descripción
ID	**Ciclo de Vida de Ingeniería del Dominio (ID)**
An	Fase de Análisis
Di	Fase de Diseño
Re	Fase de Realización
IA	**Ciclo de Vida de Ingeniería de la Aplicación (IA)**
Estrategia Desarrollo	**Estrategia utilizada para abordar el método [28]:**
Td	Top-down (proactivo)
Bu	Bottom-up (extractivo/reactivo)
Hi	Middle out o Meet-in-the-middle (híbrido)
Método	**El método describe un proceso sistemático:**
Act	Actividades a seguir
E/S	Entradas y salidas
Art	Artefactos producidos/utilizados
Rol	Roles de participación en el proceso
Arq	Vistas arquitecturales
CasoEst	Caso de estudio que valida el proceso
ModVariab	**Modelado de la variabilidad funcional y no funcional (calidad) [29]**
Car	En el modelo de características [46]
Arq	En el modelo arquitectural
PN	En el modelo de procesos de negocio
WS	En servicios: un servicio proporciona variantes mediante la parametrización; o varios servicios ofrecen la misma funcionalidad, pero con diferentes QoS (implementaciones)
Estd	Uso de un Modelo de Calidad estándar para especificar las propiedades de calidad [30]: ISO-9126, ISO-25010, McCall, Boehm, FUR PS, Dromey, etc.
EspArq	**El método considera la especificación de la AR**
Tech	**Enfoques y Tecnologías:** SCA[6]; MDA[7]; MDE[8]; SOC[9]
Ont	**Ontologías:** expresan el conocimiento del dominio; expresan la AR como medio para relacionar el dominio del problema con el dominio de la solución con una terminología común.
Herr	**Herramientas:** grado en que el método es automatizado; si una herramienta es propuesta o implementada.

Tabla VIII: Descripción de las Propiedades: Métodos para SOA

Propiedad	Descripción
GPN	**incluye el ciclo de vida de GPN [20]:**
NotacPN	Notación (BPMN, etc.) para el modelado de los procesos de negocios
LengEj	Utiliza el lenguaje BPEL para implementar (ejecutar) los procesos de negocios en la etapa de implementación del ciclo de vida de GPN
EstArq	**El estilo (modelo) arquitectural es SOA**
WS	**WS como tecnología para realizar SOA**
Métodos de conversión	**Método para identificar, especificar y realizar los servicios bajo un enfoque SOA**
Car/S	Actividades de como traducir del Modelo de Características a Servicios (“top-down”)
PN/S	Actividades de como traducir de un Modelo de Procesos de Negocios a Servicios (“top-down”)
WS/Arq	Actividades de como traducir los Servicios a Componentes arquitecturales (“top-down”)
Arq/WS	Actividades de como traducir una AR de software (componentes, conectores) a una AR cuyo componentes son Servicios (“bottom-up”)
Métodos de desarrollo	**Métodos de desarrollo SOA para las fases de Análisis y Diseño [28]:** IBM RUP/SOMA; SOAF; SOUP; SOMA; Erl methodology; Papazoglou methodology
ModRef/ ModConcept	**Modelos de Referencia/Modelos conceptuales SOA [31]:** SOA-RM[10]; SOA-RFA[11]; SOA Ontology[12]; SOMF[13]; PIM4SOA[14]; SoaMl[15]

[6] Service Component Architecture
[7] Model Driven Architecture
[8] Model Driven Engineering
[9] Service Oriented Computing
[10] OASIS Reference Model for SOA
[11] OASIS Architecture Foundation for SOA
[12] Open Group SOA Ontology
[13] Service-oriented Modeling Framework
[14] Platform-independent Model for SOA
[15] SOA Modeling Language

Tabla IX: Descripción de las Propiedades: Métodos para AR-SOA

Propiedad	Descripción
Pub	Permite la publicación de la descripción de los servicios directamente a los consumidores, a través de intermediarios
Comp	Permite que el software pueda ser construido por medio de la composición de los servicios
RNF	Cumplimiento de los requisitos no funcionales de alto nivel y/o de bajo nivel

Tabla X: Checklist de las Propiedades para Métodos AR-LPS Relevantes Identificadas en los Estudios Primarios sobre LPSOS

Estudios Primarios	AR-LPS																					Total
	ID			*IA*	*Estrategia Desarrollo*			*Método*						*ModVariab*				*Estd*	*Esp Arq*	*Tech*	*Ont*	*Herr*	
	An	Di	Re		Td	Bu	Hi	Act	E/S	Art	Rol	Arq	Caso Est	Car	Arq	PN	WS						
ZGCAL [32]	++	++	++	-	++	-	-	++	++	++	-	++	++	++	-	-	-	-	-	-	++	**11**
GB [24]	++	+	+	-	-	++	-	++	++	-	-	-	++	++	-	-	-	-	-	-	-	**8**
Istoan [2]	++	++	++	++	++	-	-	++	++	++	-	-	++	++	-	++	-	-	-	MDE[16]	-	**12**
RMALAAG [29]	+	-	++	-	++	-	-	++	++	++	-	-	-	-	-	-	++	-	-	-	-	**7**
CK [13]	++	-	-	-	-	++	-	++	-	-	-	-	++	-	-	-	++	-	-	-	-	**5**
GE [33]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
BGFF [34]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
EMA [35]	++	+	-	-	++	-	-	++	++	++	-	++	++	++	-	-	-	-	++	-	-	**10**
Med [3]	++	++	++	-	++	-	-	++	++	++	++	++	++	-	++	-	-	-	++	-	-	**12**
BMKARGBH [36]	++	++	++	++	++	-	-	++	++	++	++	-	++	++	-	-	-	-	-	-	++	**13**
DRHU [37]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
PJ [38]	++	++	++	++	-	++	-	++	++	++	++	++	++	++	-	-	-	-	-	-	++	**13**
SS [39]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
AG [40]	++	++	++	++	++	-	-	++	++	++	++	++	++	++	-	++	++	-	-	-	-	**14**
BCCMV [41]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
MAGL [6]	++	++	++	++	++	-	-	++	++	++	-	++	++	++	-	++	++	-	++	-	-	**14**
LMN [42]	++	++	++	++	++	-	-	++	++	++	-	++	++	++	-	++	-	-	-	-	-	**12**
Galster [43]	++	+	-	-	-	++	-	++	++	++	++	-	++	-	-	-	-	-	-	-	-	**8**
AMKGBH [20]	++	++	++	++	++	-	-	++	++	++	++	++	++	++	-	++	++	-	++	MDE	-	**17**
NBG [44]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
ALHG [45]	++	++	++	++	-	++	-	++	++	++	-	++	++	++	++	-	++	-	++	-	-	**14**
21	**21**	**13**	**12**	**8**	**16**	**5**	**0**	**21**	**14**	**13**	**6**	**9**	**14**	**11**	**2**	**5**	**6**	**0**	**5**	**2**	**1**	**4**	

Tabla XI: Checklist de las Propiedades para Métodos bajo SOA y AR-SOA Relevantes Identificadas en los Estudios Primarios sobre LPSOS

Estudios Primarios	SOA										AR-SOA			Total
	GPN		*EstArq*	*WS*	*Métodos de conversión a Servicios*				*Métodos de Desarrollo*	*ModRef/ModConcept*	*Pub*	*Comp*	*RNF*	
	NotacPN	LengEj			Car/S	PN/S	WS/Arq	Arq/WS						
ZGCAL [32]	++	++	++	++	++	++	++	-	-	-	-	++	-	**8**
GB [24]	-	-	++	-	++	-	-	-	-	-	-	-	-	**2**
Istoan [2]	++	-	++	-	++	++	-	-	-	-	-	++	-	**5**
RMALAAG [29]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
CK [13]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
GE [33]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
BGFF [34]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
EMA [35]	++	-	++	-	-	-	-	-	-	-	-	-	-	**2**
Med [3]	-	-	++	++	++	-	-	-	RUP/SOMA	-	+	++	++	**7**
BMKARGBH [36]	-	-	++	++	++	-	-	-	-	-	-	++	-	**4**
DRHU [37]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
PJ [38]	-	-	++	++	++	-	++	-	-	-	-	++	-	**5**
SS [39]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
AG [40]	++	-	++	-	++	++	-	-	-	SoaML[17]	-	++	-	**6**
BCCMV [41]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
MAGL [6]	++	-	++	-	-	++	-	-	-	-	-	++	++	**5**
LMN [42]	++	-	++	-	++	++	++	-	-	-	-	++	-	**6**
Galster [43]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
AMKGBH [20]	++	-	++	-	++	++	-	-	-	-	-	++	-	**5**
NBG [44]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
ALHG [45]	-	-	++	++	++	++	++	-	-	-	-	++	-	**6**
21	**7**	**1**	**21**	**5**	**10**	**7**	**4**	**0**	**1**	**1**	**1**	**10**	**2**	

[16] Model Driven Engineering
[17] http://www.omg.org/spec/SoaML

Luego de la captura de los datos, reflejados a través de las propiedades identificadas en las Tablas X, XI, aquellos estudios que obtuvieron el mayor número de propiedades consideradas, fueron seleccionados para ser evaluados utilizando las categorías, elementos y preguntas del MUE descrito en la sección III (Tablas I, II, III).

V. DISCUSIÓN

A continuación se presenta el informe de los resultados analíticos de la RDS, considerando cada pregunta de investigación y analizados a través de la aplicación del MUE, expresado a través de sus criterios y elementos de análisis. La Tabla XII identifica los 10 estudios más representativos relacionados con procesos de desarrollo de LPSOS, de los 21 estudios primarios inicialmente aceptados y que luego de ser analizados, son los que satisfacen el mayor número de propiedades encontradas en los métodos, este valor es obtenido de la sumatoria de los totales de las Tablas X, XI. MUE se aplicó a estos 10 estudios, que tratan explícitamente procesos de desarrollo para LPSOS.

Tabla XII: Estudios más Representativos entre los Primarios para LPSOS

Estudios	Propiedades
ZGCAL [32]	19
Istoan [2]	17
Med [3]	19
BMKARGBH [36]	17
PJ [38]	18
AG [40]	20
MAGL [6]	19
LMN [42]	18
AMKGBH [20]	22
ALHG [45]	20

A. Con Respecto a Métodos para AR-LPS (ver Tabla I)

1) Criterio Dominio: Los estudios tratan los dominios: gobierno electrónico o "e-government", comercio electrónico (e-business), y reservaciones (e-travel).

2) Criterio Contexto del Método: objetivos, fases, entradas, salidas

Objetivo: Se destacan dos propuestas para el desarrollo de LPSOS [2][20] que utilizan el paradigma de la ingeniería dirigida por modelos, en inglés *Model Driven Engineering (MDE)*. La principal diferencia entre ambos, es que aplican el enfoque de forma diferente: [20] lo aplica para desarrollar el ciclo de vida de ID para LPS completo; y [2] lo utiliza en la etapa final del ciclo de vida de la IA, para la generación de los productos de la línea.

Fases: Todas las propuestas incluyen el ciclo de vida de la ID para LPS.

Entradas: Todas las propuestas parten de que el dominio del problema es conocido.

Salidas: Modelo de características [4][20][32][38][42][46], modelo de variabilidad [8], modelo de procesos de negocio [6][7][20][40][41], modelo de servicios [2], modelo de componentes [3][42] y AR [11][19][20][45].

3) Criterio Usuarios del Método: grupo, motivación habilidades, orientación

Grupo: La mayoría de las propuestas señalan como participantes o actores a ingenieros del dominio, arquitectos e ingenieros de software y analistas SOA. Los *roles más representativos* son: *Business Analyst (Analista de Negocio)* que es el responsable de verificar que los candidatos a orquestación de servicios representen de forma precisa la lógica del negocio; *Domain Architect (Arquitecto del Dominio)* para la identificación de los componentes que proveen la implementación de las operaciones expuestas por los servicios; *SOA Architect (Arquitecto SOA)* para la identificación de los servicios candidatos; *Domain Designer (Diseñador del Domino)* and *Service Designer (Diseñador de Servicios)*. Para [3] los más importantes son: Analista de Negocio, el Arquitecto SOA y el Arquitecto del Dominio.

Orientación: Todas las propuestas indican al usuario como realizar las actividades en cada una de las fases del método.

Motivación y habilidades: Ninguna de las propuestas describe las habilidades requeridas por los usuarios de los métodos para poder realizar de forma efectiva las actividades presentadas.

4) Criterio Componentes del Método: estructura, artefactos, vistas, lenguajes, variabilidad y herramientas

Estructura: En cuanto a la *definición de los pasos y su secuencia*, existe consenso en determinar el alcance de la LPS mediante el análisis de sus características expresado a través del modelo de características, esto es desde el punto de vista estructural, y desde el punto de vista de su comportamiento, se realiza un análisis de los procesos de negocios utilizando por lo general BPMN para expresarlos, ambos conceptos son muy utilizados en la especificación de la variabilidad en la LPSOS.

En [20], para lograr la integración LPS-MDE-SOA, se comienza por la definición de los procesos de negocio y por la especificación del conjunto de actividades coordinadas para el cumplimiento de los objetivos organizacionales. Luego, los elementos de software necesarios para lograr estos objetivos (los servicios) son identificados, y para cada servicio, se identifican y desarrollan el conjunto de interfaces realizando la especificación del servicio, así como las correspondientes implementaciones de servicio.

Artefactos: el modelo de características [46] es el más utilizado, seguido del modelo de procesos de negocios [6][40], el modelo de variabilidad [32][38][42], y la AR [45].

Variabilidad: [3] indica que la gestión de la variabilidad es una actividad clave en ILPSOS, en la cual los servicios proporcionan la flexibilidad de las LPS y de sus productos, diferenciándolos en el momento de establecer el enlace de los servicios: en tiempo de diseño (estáticamente) y en tiempo en ejecución (dinámicamente). La variabilidad en la mayoría de los estudios analizados se realiza en el modelado de características, en algunos estudios ésta es realizada a través de los procesos de negocios [20], y en un solo estudio [45] es realizada a nivel de los servicios y de los componentes, expresados mediante una AR.

También en [20] se indica que la mayoría de los enfoques de gestión de procesos de negocio enfatizan el desarrollo de un proceso de negocio, y a partir del mismo, se derivan muchas variantes, especializadas según diversas necesidades de la organización. Los procesos de negocio pueden tener algunas piezas comunes para un grupo de diferentes casos de aplicación, donde implica cierta variabilidad en la selección de las actividades de un proceso de negocio. Una funcionalidad común puede ser desarrollada como un activo reutilizable para la creación de nuevas variantes de los procesos de negocio en

un dominio. Además, los puntos de variación, derivados de las propiedades no funcionales y tecnológicas de las actividades, proporcionan la funcionalidad del negocio con diferentes propiedades no funcionales y tecnológicas. Tal variabilidad facilita la configuración flexible del grupo de procesos de negocio dirigida por la calidad [20].

Vistas: Con relación a las *vistas arquitecturales*, la especificación de la arquitectura es la actividad, en la cual la arquitectura es documentada utilizando diferentes vistas y notaciones para representar los intereses de los diferentes actores involucrados. Existen cuatro estudios [3][32][40][42] que proponen la utilización de vistas arquitecturales similar al enfoque del 4+1 vistas [25].

En [40] se especifican 5 vistas: Vista de Características, Vista de Contrato de Servicio, Vista de Procesos de Negocio, Vista de Interface de Servicios y la Vista de Coordinación de Servicios. En [3] las 4 vistas arquitecturales consideradas son: Vista de Capas; Vista de Integración; Vista de Componente y Vista de Interacción. En [32] se especifican 3 vistas arquitecturales: Vista de Coordinación, Vista de Aplicación, Vista de Datos. En [42] se propone un estilo arquitectural heterogéneo, con tres niveles de descomposición asociados a tres estilos arquitecturales representados mediante 3 vistas: Vista Arquitectural de Servicios, Vista Arquitectural de Comunicación entre Procesos, Vista Arquitectural (en el ADL[18] C2). Solo los estudios [3][42] consideran la Vista de Componentes, donde existe una correspondencia entre las interfaces de los servicios y los componentes que las implementan.

Lenguajes: Los lenguajes utilizados por el método para el modelado arquitectural, [3][32][40][42] utilizan UML para expresar algunas de las vistas arquitecturales, especialmente [45] la utiliza para especificar la AR; para modelar los procesos de negocios, en [20][36] utilizan BPMN y en [30][40] el diagrama de actividades se expresa en UML.

Herramientas: En cuanto a las herramientas utilizadas solo cuatro trabajos mencionan alguna [20][32][36][38], que abordan parcialmente alguna parte del método propuesto; en [20] se utiliza FeatureMapper[19] para especificar el modelo de características y relacionar las características con los artefactos de software que las implementan; en [36] utilizan un enfoque basado en ontologías, donde utilizan el modelo de características [46] para la generación automática de composiciones de servicios ejecutables, para ello usan un lenguaje semántico de WS y utilizan un framework que soporta la composición de WS a nivel semántico, denominado *Web Service Modelling Ontology (WSMO)*, donde las composiciones de servicios son codificadas en el lenguaje WSML, este framework tiene una herramienta que ejecuta estas composiciones, denomina *Web Service Modelling eXecution environment (WSMX)*[20], que permite generar y ejecutar la orquestación y coreografía de los servicios. También utilizan *ATLAS*[21] *Transformation Language (ATL)* para traducir del modelo de características al WSML. En [32] *Feature Plug-in*[22] y [38] *FeatureIDE*[23] son utilizadas para especificar el modelo de características.

5) Criterio Gestión de la Calidad en el Método: propiedades de calidad

Propiedades de calidad: La calidad en general es muy poco abordada en la LPSOS y tratada tarde en la etapa de diseño, solo dos estudios [3][6] trataron este aspecto de forma más precisa. En [6] se indica que en la fase de diseño del dominio se debe incluir la especificación de las propiedades no funcionales, debido a que los NFRs están entrelazados y relacionados con los requisitos funcionales, y que puede haber varios servicios que proporcionan la misma funcionalidad, pero que difieren de la implementación de su QoS. Para ello, las características en el modelo de características tipo FODA [46] son etiquetadas con los rangos de calidad que serán soportados por la arquitectura de la línea de productos, ayudando al ingeniero de servicios y de software a evaluar el impacto de las características variantes seleccionadas acorde a las características de calidad que los servicios proveen. Por su parte [3] presenta una técnica para identificar servicios a partir de los atributos de calidad, analizando escenarios de atributos de calidad para identificar los servicios que ayuden al cumplimiento de los atributos de calidad arquitectural, para ello, dispone de un conjunto de patrones de diseño SOA que permiten satisfacer algunos atributos de calidad. Ninguna propuesta utiliza un estándar para especificar la calidad.

6) Criterio Validación del Método: madurez, calidad de la arquitectura

Madurez: Todas las propuestas analizadas presentaban un caso de estudio, donde se evaluaba su factibilidad.

Calidad de la arquitectura: Solo dos propuestas evalúan la calidad arquitectural: [6] lo hace desde el punto de vista de los usuarios finales del método, mediante la evaluación del desempeño y escalabilidad del método; [3] presenta métricas para medir la cohesión y el acoplamiento entre los componentes a nivel arquitectural.

B. Con Respecto a Métodos para SOA (ver Tabla II)

1) Criterio Concepto de Servicio: negocio, WS

En términos generales la mayoría de los métodos abordan ambas definiciones: servicios de negocios y WS, primero identifican cuales son los servicios necesarios para dar soporte a las funcionalidades del negocio y luego identifican cuales WS son requeridos.

2) Criterio Estrategia de realización o desarrollo: top-down, bottom-up, middle-out

"Top-down" fue la más utilizada, con un 76,2% del total de estudios analizados, seguido del "bottom-up" con un 23,8%; en ninguno de los métodos se abordó un enfoque hibrido.

3) Criterio Cobertura del ciclo de vida: Identificación, Especialización, Realización

Solo un estudio [45] abarcó en su análisis el ciclo de vida completo SOA, el resto de los estudios solo se enfocan en la presentación de propuestas para la fase de identificación de los servicios.

[18] Architecture Description Language
[19] http://featuremapper.org
[20] http://www.wsmx.org
[21] https://eclipse.org/atl
[22] http://gsd.uwaterloo.ca/fmp
[23] http://wwwiti.cs.uni-magdeburg.de/iti_db/research/featureide

4) Criterio Grado de granularidad del análisis: Bajo, Moderado, Alto

El nivel de abstracción en general es alto, la identificación de los servicios es la actividad más realizada en los estudios [3][20][36][38][40], la especificación es muy poco abordada y la realización casi inexistente. Sin embargo, dos estudios ameritan mencionarse: en [20], en el paradigma SOA, las actividades en un grupo de procesos de negocio pueden ser implementadas por servicios o delegadas a los usuarios. Cada actividad, puede tener uno o más servicios que pueden realizarla. Los servicios pueden ser comunes y reutilizados entre diversos procesos de negocios. Diferentes servicios pueden proporcionar funcionalidad del negocio para soportar distintos requisitos tecnológicos, como la comunicación a través de diversos medios de transporte. Además, pueden proporcionar funcionalidad del negocio con diferentes propiedades no funcionales, como seguridad y rendimiento, expuestos por diferentes implementaciones de interfaces de servicios y composición de servicios. En [45], la variabilidad de una LPS es realizada a nivel de los servicios y componentes, expresados en una AR. Ninguno de los trabajo revisados aborda tener una AR y luego adaptarla a servicios.

5) Criterio Accesibilidad y validez: Documentación no accesible, parciamente accessible, bien accessible

Todos los métodos analizados presentan una buena documentación y presentan un caso de estudio, donde se detallaba las actividades a seguir y se evaluaba su validez.

C. Con Respecto a Métodos para AR-SOA (ver Tabla III)

1) Criterio Publicación del servicio: directamente a consumidores, por intermediarios

Este criterio no fue abordado en los métodos analizados.

2) Criterio Composición de los servicios: orquestación, coreografía

La mayoría de las propuestas presentan como alternativa para la composición de los servicios la orquestación, implementada a través de los procesos de negocios, quienes coordinan el llamado a los servicios.

3) Criterio Calidad del Servicio: cumplimiento de requisitos no funcionales de alto nivel y de bajo nivel

Ninguno de los métodos evalúa la calidad de servicio a un alto nivel de abstracción en las arquitecturas presentadas, como por ejemplo desempeño, escalabilidad, interoperabilidad.

VI. Conclusiones

La principal contribución de este trabajo fue realizar una RDS para presentar una perspectiva detallada sobre las fases, actividades, artefactos y roles que participan en un proceso de desarrollo para LPSOS; en la literatura no se encontraron RDS similares. Aunque los beneficios de la orientación a servicios son frecuentes en la literatura, la revisión, análisis y evaluación de los 21 métodos presentados en esta RDS ha demostrado que falta un enfoque integral que incluya en las fases de un ciclo de vida ID de LPSOS, la consideración de las propiedades de calidad, para la identificación, especificación y realización de los servicios como componentes arquitecturales representados mediante una AROS-LPS, que muestre la clara trazabilidad entre los componentes de servicios funcionales y no funcionales. La gestión de calidad en los métodos fue muy poco abordado, y ningún estándar para especificar las propiedades de calidad fue considerado. Por su parte, la gestión de la variabilidad es clave en un contexto: SOA y LPS. En ambos casos, los puntos de variación deben ser implementados ya sea por un único servicio (donde una interfaz de servicios puede ofrecer parametrización) o a través de servicios similares que traten cada variación; estas variaciones pueden estar influenciadas por las tecnologías seleccionadas para satisfacer propiedades de calidad requeridas para alcanzar los objetivos empresariales, representados como componentes de servicios que resuelven funcionalidades. Los métodos que aplican la gestión de procesos de negocio enfatizan el desarrollo de un proceso de negocio, y a partir de él, se derivan las variantes, especializadas según las necesidades de la organización. Los procesos de negocio pueden tener algunas partes comunes para un grupo de diferentes casos de aplicación, donde implica cierta variabilidad en la selección de las actividades de un proceso de negocio. La RDS presentada provee a la comunidad de desarrolladores de LPSOS la visión general de métodos actuales para tomar decisiones en cuanto a la definición de un método general para el diseño de una AROS-LPS. La principal limitación del estudio realizado es que la RDS no es una comparación para determinar quién de los métodos de desarrollo es el mejor, la intención es de describir como los autores abordan el proceso de desarrollo de una AROS-LPS. Una perspectiva a ser considerada sería evaluar el alcance de la AROS-LPS respecto a la calidad y número de productos que se pueden derivar a partir de ella, para una familia de LPSOS determinada.

Agradecimientos

Al CDCH-UCV (Consejo de Desarrollo Científico y Humanístico), proyecto DARGRAF PG 03-8730-2013-2.

Referencias

[1] P. Istoan, G. Nain, G. Perrouin, and J. Jezequel, *Dynamic Software Product Lines for Service-Based Systems*, 9th IEEE International Conference on Computer and Information Technology (CIT´09), Xiamen, China, October 2009.

[2] P. Istoan, J. M. Jézéquel, and G. Perrouin, *Software Product Lines for Creating Service-Oriented Applications*, Ph.D. thesis, Irisa Rennes Research Institute, Rennes, France, June 2009.

[3] F. M. Medeiros, *SOPLE-DE: an Approach to Design Service-Oriented Product Line Architectures*, Master. thesis, Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Pernambuco, Recife, Brasil, 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. thesis, Department of Computer and Information Science, Linköping University, Linköping, Sweden, 2010.

[6] B. Mohabbati, M. Asadi, D. Gašević, and J. Lee, *Software Product Line Engineering to Develop Variant-Rich Web Services*, In Web Services Foundations, Springer, New York, 2014.

[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, no. 8, 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] World Wide Web Consortium, *Web Services Architecture Requirements*, W3C Working Group Note, February 2004.

[10] J. Nicolai, *SOA in Practice: the Art of Distributed System Design*, OReilly: Sebastopol, CA, 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] S. Cohen and R. Krut, *Managing Variation in Services in a Software Product Line Context* (No. CMU/SEI-2010-TN-007), Carnegie-Mellon University, Software Engineering Institute, Pittsburgh, PA, USA, 2010.

[14] 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.

[15] M. Shaw and D. Garlan, *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, 1996.

[16] 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.

[17] M. Matinlassi, *Comparison of Software Product Line Architecture Design Methods: COPA, FAST, FORM, KobrA and QADA*, In Proceedings of the 26th International Conference on Software Engineering (ICSE'04), pp. 127-136, Edinburgh, UK, May 2004.

[18] T. Kohlborn, A. Korthaus, T. Chan, and M. Rosemann, *Identification and Analysis of Business and Software Services - a Consolidated Approach*, Services Computing, IEEE Transactions, vol. 2, no. 1, pp. 50-64, 2009.

[19] L. B. R. de Oliveira, K. R. Felizardo, D. Feitosa, and E. Y. Nakagawa, *Reference Models and Reference Architectures Based on Service-Oriented Architecture: a Systematic Review*, In Software Architecture, pp. 360-367, Springer Berlin Heidelberg, 2010.

[20] M. Asadi, B. Mohabbati, N. Kaviani, D. Gaševic, M. Boškovic, and M. Hatala, *Model-driven Development of Families of Service-Oriented Architectures*, In Proceedings the 1th International Workshop on Feature-Oriented Software Development (FOSD´09), pp. 95-102, Denver, Colorado, USA, October 2009.

[21] Object Management Group (OMG). *Business Process Model and Notation (BPMN), Version 2.0*, 2011.

[22] B. List and B. Korherr, *An Evaluation of Conceptual Business Process Modelling Languages*, In Proceedings of the 2006 ACM symposium on Applied computing (SAC'06), pp. 1532-1539, Dijon, France, April 2006.

[23] J. Herrera, F. Losavio, and A. Matteo, *RDS de Enfoques y Técnicas para la Construcción de Arquitecturas en un Contexto de Líneas de Productos de Software*, Revista Venezolana de Ciencias de la Computación ReVeCom, vol. 1, no. 1, pp. 17-25, Junio 2014.

[24] S. Günther and T. Berger, *Service-Oriented Product Lines: Towards a Development Process and Feature Management Model for Web Services*, In SPLC, vol. 2, pp. 131-136, October 2008.

[25] P. Kruchten, *Architectural blueprints—The "4+1" View Model of Software Architecture*, In proceedings of the Conference on TRI-Ada'95, Anaheim, CA, USA, November 1995.

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

[27] A. Arsanjani, *Service-Oriented Modeling and Architecture: How to Identify, Specify, and Realize Services for your SOA*, IBM Software Group, November 2004.

[28] S. Svanidzaitė, *A Comparison of SOA Methodologies Analysis & Design Phases*, In Tenth International Baltic Conference on Databases and Information Systems (Baltic DB & IS 2012), Vilnius, Lithuania, July 2012.

[29] H. G. B. Ribeiro, S. R. de Lemos Meira, E. S. de Almeida, D. Lucredio, A. Alvaro, V. Alves, and V. Garcia, *An Assessment on Technologies for Implementing Core Assets in Service-Oriented Product Lines*, In 4th Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS´10), pp. 90-99, Bahia, Brazil, September 2010.

[30] J. P. Miguel, D. Mauricio, and G. Rodríguez, *A Review of Software Quality Models for the Evaluation of Software Products*, International Journal of Software Engineering & Applications (IJSEA), vol. 5, no. 6, November 2014.

[31] M. Mohammadi and M. Mukhtar, *A Review of SOA Modeling Approaches for Enterprise Information Systems*, Procedia Technology, vol. 11, pp. 794-800, 2013.

[32] F. Zaupa, I. M. de Souza Gimenes, D. D. Cowan, P. S. Alencar, and C. J. P. de Lucena, *A Service-Oriented Process to Develop Web Applications*, Journal UCS, vol. 14, no. 8, pp. 1368-1387, 2008.

[33] M. Galster and A. Eberlein, *Identifying Potential Core Assets in Service-Based Systems to Support the Transition to Service-Oriented Product Lines*, In 18th IEEE International Conference and Workshops on Engineering of Computer-Based SystemsIn Engineering of Computer Based Systems (ECBS), pp. 179-186, Las Vegas, Nevada, USA, April 2011.

[34] M. H. ter Beek, S. Gnesi, A. Fantechi, and J. L. Fiadeiro, *Variability and Rigour in Service Computing Engineering*, In 35th Annual IEEE Software Engineering Workshop (SEW), pp. 122-127, Limerick, Ireland, June 2011.

[35] A. Ezenwoke, S. Misra, and M. O. Adigun, *An Approach for E-Commerce On-Demand Service-Oriented Product Line Development*, Acta Polytechnica Hungarica, vol. 10, no. 2, pp. 69-87, 2013.

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

[37] N. C. Das, S. Ripon, O. Hossain, and M. S. Uddin, *Requirement Analysis of Product Line Based Semantic Web Services*, Lecture Notes on Software Engineering, vol. 2, no. 3, p. 210, 2014.

[38] C. Parra and D. Joya, *SPLIT: an Automated Approach for Enterprise Product Line Adoption Through SOA*, Journal of Internet Services and Information Security (JISIS), vol. 5, no. 1, pp. 29-52, 2015.

[39] H. Serajzadeh and F. Shams, *An Approach for Discovering Services for a Service-Oriented Product Line*, Global Journal on Technology, vol. 1, 2012.

[40] M. Abu-Matar and H. Gomaa, *Feature Based Variability for Service Oriented Architectures*, In WICSA'11 Proceedings of the 2011 Ninth Working IEEE/IFIP Conference on Software Architecture, pp. 302-309, Boulder, Colorado, USA, June 2011.

[41] N. Boffoli, D. Caivano, D. Castelluccia, F. M. Maggi, and G. Visaggio, *Business Process Lines to Develop Service-Oriented Architectures Through the Software Product Lines Paradigm*, In SPLC, vol. 2, pp. 143-147, 2008.

[42] 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.

[43] M. Galster, *Describing Variability in Service-Oriented Software Product Lines*, In Proceedings of the 4th European Conference on Software Architecture (ECSA'10), pp. 344-350, Copenhagen, Denmark, August 2010.

[44] M. Njima, M. H. Ter Beek, and S. Gnesi, *Product Line Architectures for SOA*, In Proceedings of the 11th Conference on Software Engineering Research and Practice (SERP´11), Las Vegas, Nevada, USA, July 2011.

[45] I. Achour, L. Labed, R. Helali, and H. B. Ghazela, *A Service Oriented Product Line Architecture for E-Government*, The 2011 International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government (EEE'11), Las Vegas, Nevada, USA, July 2011.

[46] K. Lee, K. Kang, and J. Lee, *Concepts and Guidelines of Feature Modeling for Product Line Software Engineering*, Lecture Notes in Computer Science, vol. 2319, pp. 62-77, 2002.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 26-37, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Diseño de una Arquitectura de Referencia para el Aprendizaje Electrónico basado en Modelo de Negocio y Calidad de Producto: un Caso de Estudio

Yuly Esteves[1], Francisca Losavio[2]
yulyesteves@gmail.com, francislosavio@gmail.com

[1] Departamento de Ciencias Naturales y Matemática, UPEL-IPMJMSM. Caracas, Venezuela
[2] Laboratorio MoST, Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: Una arquitectura de referencia para el dominio del aprendizaje electrónico o *e-learning* debe conciliar aspectos tecnológicos y pedagógicos relevantes, lo que implica la consideración de características del negocio educativo para garantizar la satisfacción de la calidad exigida por estos sistemas. En un trabajo previo de Esteves y Losavio (2016) se define el proceso de diseño arquitectónico MN-AR (*Modelo de Negocio y Arquitectura de Referencia)* basado en el modelado del negocio y dirigido por la calidad, especificada por el modelo de calidad estándar ISO/IEC 25010. MN-AR se inspira en el trabajo previo de Losavio, Ordaz y Esteller (2015) el cual sigue un enfoque de diseño ascendente extractivo, basado en la refactorización de arquitecturas de productos existentes en el mercado para un dominio. El objetivo de este trabajo es presentar un caso de estudio y aplicar MN-AR paso a paso para validarlo y también con el fin de que los elementos esenciales de la propuesta sean explícitamente explicados e ilustrados.

Palabras Clave: Aprendizaje Electrónico; Calidad del Software; Modelado del Negocio; Arquitectura de Referencia; IEEE-LTSA; ISO/IEC 25010.

Abstract: A reference architecture for the e-learning domain should combine relevant technological and pedagogical aspects; as a consequence, characteristics of the educational business must be considered to guarantee the satisfaction of the required quality of these systems. The previous work by Esteves and Losavio (2016) defined MN-AR (*Modelo de Negocio y Arquitectura de Referencia*), a quality-driven architectural design process based on business modeling, where software product quality is specified by the standard quality model ISO/IEC 25010. MN-AR is inspired in the previous work of Losavio, Ordaz and Esteller (2015) which follows a bottom-up extractive approach, based on the refactoring of architectures of existing market products in a particular domain. The goal of this work is to present a case study and apply MN-AR step by step to validate the process and also to explicitly illustrate and explain the main elements of the proposition.

Keywords: E-Learning; Software Quality; Business Modeling; Reference Architecture; IEEE-LTSA; ISO/IEC 25010.

I. Introducción

La presente investigación considera una adaptación del proceso de diseño arquitectónico para líneas de productos de software ascendente o "bottom-up" extractivo, centrado en la refactorización de productos existentes, definido en [1], como paso natural para obtener una *arquitectura de referencia* (AR) que permita generar a partir de ella, aplicaciones o sistemas concretos de calidad para el dominio del aprendizaje electrónico. La AR es una arquitectura genérica e instanciable que se reutiliza como plantilla o marco de referencia arquitectural, para derivar una familia de productos o sistemas de software similares en un determinado dominio [2]; es el eje central de la plataforma de aprendizaje electrónico sobre el cual se articula el sistema completo; en nuestro contexto, la *plataforma* contiene la AR.

Un *dominio* se define como el conjunto mínimo de propiedades que definen una familia de problemas para los cuales se requieren soluciones computacionales [3]. En este trabajo, se analiza el conjunto de problemas, características y tendencias actuales referidas al uso de las Tecnologías de Información y Comunicación (TIC) para mediar el proceso de aprendizaje, es decir, se trabaja en el contexto del *aprendizaje electrónico*, o del inglés *e-learning*; ambos términos se utilizarán indistintamente en esta investigación.

Por otra parte, en un trabajo previo de las mismas autoras [4] se define la adaptación del proceso de diseño de la AR propuesto en [1]; este nuevo proceso se denomina *MN-AR (Modelo de Negocio y Arquitectura de Referencia)* y considera explícitamente el modelo del negocio como entrada a la etapa de Análisis del Dominio; a diferencia del

proceso planteado en [1], MN-AR toma como entrada AR existentes definidas para el e-learning [5][6][7][8] y no productos concretos del mercado, y se considera la refactorización de estas arquitecturas en lugar del estudio de las arquitecturas de sistemas concretos e-learning. La AR obtenida en MN-AR es luego adecuada al estándar IEEE-LTSA [8], como práctica usual realizada para "validar" AR para el dominio e-learning. MN-AR contempla la trazabilidad entre *requisitos funcionales (RF)* y *no funcionales (RNF)*, garantizando de esta manera la calidad global de la AR y facilitando el cumplimiento de una de sus características prioritarias de calidad, su evolución en el tiempo; sin embargo en [4] el proceso completo no es aplicado a un caso de estudio. El objetivo principal de este trabajo es complementar lo realizado en [4] con la definición de un breve caso de estudio y aplicar MN-AR paso a paso, con el fin de ofrecer una primera validación de este proceso, resaltando su carácter sistemático y repetible.

Este artículo se estructura, además de esta *Introducción* y las *Conclusiones*, en tres secciones: *Contexto*, donde se presentan trabajos previos y bases conceptuales que dan soporte al estudio; *Proceso MN-AR*, donde se especifican las etapas del proceso propuesto; y finalmente *Caso de Estudio*, donde se aplica paso a paso el proceso MN-AR.

II. CONTEXTO

A. Trabajos Relacionados

Los trabajos relacionados con el tema en estudio, van en las siguientes direcciones:

- Hay enfoques para el proceso de diseño arquitectural que consideran relacionar los RF con los requisitos de calidad u objetivos de calidad a los cuales estos deben responder, entre estos se encuentra [9], donde se realiza una extensión al proceso propuesto en [10] llamado PEC (Proceso Extendido de Chung), incluyendo estos aspectos en la etapa inicial del análisis del dominio, para la reutilización del conocimiento sobre la arquitectura y la identificación de restricciones o propiedades globales sobre las familias de sistemas del dominio. Este análisis incluye la especificación de las propiedades de calidad derivadas de RF y RNF, es decir arquitecturales y otras restricciones extraídas de las reglas de negocio, especificadas mediante el estándar ISO/IEC 25010 [11]; una adaptación de este proceso mediante una etapa inicial de modelado del negocio se trató en [12]. El resultado de este análisis es el punto de partida para la justificación de los requisitos globales de la aplicación o sistema a ser construido utilizando estándares [9][11][13].
- Plataformas e-learning: aquí se destacan los trabajos de [14][15][16], quienes analizan AR para el aprendizaje electrónico y resaltan la importancia de garantizar la interoperabilidad entre plataformas, entre otros aspectos de calidad.
- Por otra parte, el trabajo presentado en [5] se relaciona directamente con esta investigación; su objetivo es obtener una AR para las plataformas e-learning uniendo dos AR existentes WbIS [7] y UkeU [17]; esta AR obtenida sin aplicar ningún proceso, es adecuada al IEEE-LTSA [8]; nuestro proceso MN-AR [4] es sistemático y repetible y también llega a especificar la AR, adecuándola al estándar IEEE-LTSA [8], lo cual es una práctica común en la mayoría de los trabajos que tratan del diseño de AR para este dominio. En [5] no se presenta ningún método específico de desarrollo, pero al igual que nosotros, se consideran aspectos de calidad y soluciones arquitecturales para satisfacerlas, pero no siguen ningún estándar ni técnicas para especificar dicha calidad.
- Otra referencia importante en esta investigación es el trabajo realizado en [1], donde se propone un proceso completo de diseño arquitectónico ascendente o "bottom-up" de refactorización de productos existentes en el mercado, para obtener una AR en el dominio de los Sistemas de Información Integrados de Salud, en un contexto de producción industrial de software. De este trabajo se tomó el sub-proceso para obtener automáticamente una Arquitectura Candidata (AC), la cual, utilizando el modelo de calidad estándar ISO/IEC 25010 junto con técnicas de orientación a metas para introducir también la variabilidad no funcional, permite obtener manualmente la AR a partir de la AC completada, agrupando en puntos de variación las variantes obtenidas que realizan tareas similares. El estudio en [4] incorpora a la propuesta de [1] el modelado del negocio como entrada a todo el proceso y el tratamiento de la calidad desde los procesos de negocio. Sin embargo, no se considera ahora el enfoque de metas, estableciéndose la trazabilidad de RF y RNF a través de relaciones entre componentes provee/ proporciona o "provides/requires", que determinan claramente que propiedad no funcional es requerida por cada funcionalidad.

B. Aprendizaje Electrónico

Las innovaciones desarrollada a finales del siglo XX y que han marcado el transcurso del siglo XXI, han redimensionado las estrategias de aprendizaje abierto y a distancia. Esta modalidad de estudio cada día tiene más aceptación "estimulado, en parte, por el creciente interés de educadores y tutores en las nuevas tecnologías vinculadas a internet y otras plataformas multimedia, y en parte debido al creciente consenso sobre la necesidad de apoyar las formas tradicionales de educación, valiéndose de medios más innovadores" [18].

En este sentido, cuando se habla de educación a distancia mediada por el computador, se hace referencia al aprendizaje electrónico, el cual permite crear ambientes de aprendizaje interactivos, eficientes, accesibles y distribuidos que, puede ser clasificado según los medios tecnológicos de los que hacen uso:

- El CBT (*Computer Based Training*) o CAI (*Computer Assisted Instruction*), aprendizaje basado en computador o instrucción asistida por computador, fue implantado en múltiples instituciones educativas y organizaciones.

Estaba basado en la lectura e incorporaba mecanismos de realimentación pregunta-respuesta, convirtiendo al alumno en un ente más activo dentro de su propio proceso formativo.

- El IBT (*Internet Based Training*) fue el siguiente paso evolutivo de los sistemas de aprendizaje basados en computador, CBT. Con la llegada de Internet los contenidos podían llegar a sus destinatarios a través de Internet o de intranet.
- El WBT (*Web Based Training*) consiste en el aprendizaje haciendo uso de la web, a través de la que se reciben los contenidos. En este último tipo se encuentra el Campus Virtual [19].

Así pues, puede decirse que el e-learning es una forma de utilizar la tecnología para mediar el proceso de enseñanza y de aprendizaje, utilizando modalidades abiertas y a distancia, capaz de llegar a un gran número de personas en el mundo, sin perder la capacidad de interacción entre profesores y estudiantes.

C. Características de Calidad

Algunos resultados obtenidos en [5], respecto a los sistemas de software educativos en general, sirven como base para analizar las características de calidad prioritarias, de acuerdo con el tipo de sistema educativo, entre los cuales están presentes los sistemas *e-learning*, junto con las soluciones arquitecturales requeridas por estos sistemas y dirigirán nuestro proceso de diseño de la AR el cual será presentada en detalle en la Sección III. Esta información, junto con las características de calidad que puedan derivarse de los objetivos y reglas del negocio especificadas en el modelo del negocio, permiten precisar un modelo de calidad para el dominio de los sistemas *e-learning* y expresarlo mediante una adaptación o instanciación del modelo de calidad estándar ISO/IEC 25010 [11], descrito en la Figura 1.

- Un *modelo de calidad del producto* según [11], está compuesto por ocho características de alto nivel de abstracción, que se subdividen en sub-características. Se refieren a propiedades *inherentes estáticas* (como usabilidad, disponibilidad, etc. y a las inherentes dinámicas (como eficiencia, fiabilidad, etc.) del sistema informático. También se habla de propiedades *asignadas*, que no son de calidad, como costo, tiempo de mercadeo, etc., pero influyen en las decisiones durante la construcción del software. El modelo es aplicable a los productos de software y en general a los sistemas informáticos, generalmente complejos, que también involucran componentes de software [11].
- Un *modelo de calidad en uso* que se relaciona con el resultado de la interacción con los usuarios finales, cuando un sistema de software se emplea en un contexto particular de uso. Este modelo no será utilizado en este trabajo que se centra en las etapas de análisis y diseño de una AR y no de un sistema concreto ya desarrollado.

Las características de calidad definidas son relevantes para todos los productos de software y sistemas informáticos. El modelo de calidad del producto, adaptado al dominio *e-learning* (ver Figura 1), es utilizado en MN-AR para dirigir el proceso de diseño completo de la AR.

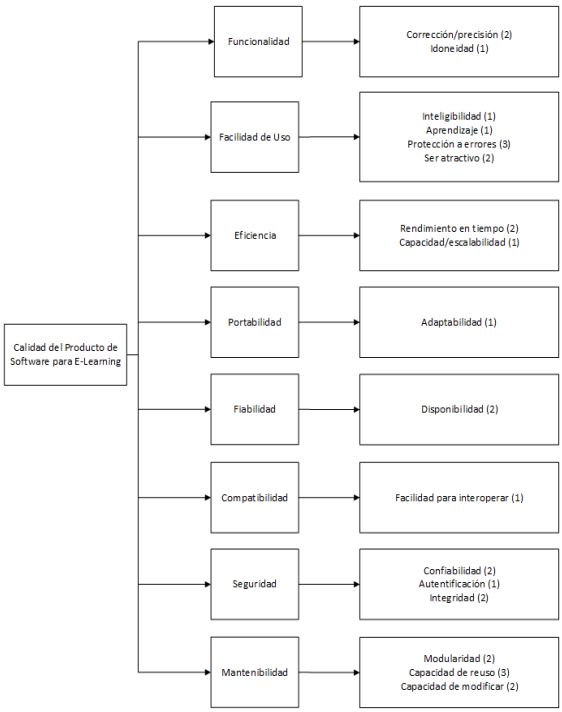


Figura 1: EL-DQM: Modelo de Calidad ISO/IEC 20510 [11] Adaptado al Dominio E-learning [5], con Prioridades (1≤p≤3, 1 Máxima Prioridad)

D. Modelado del Negocio para E-Learning

Para el modelado del negocio, en este estudio se asumirán las fases propuestas por [20] adaptándolas al dominio educativo (ver Figura 2), pues se considera que la incorporación de conceptos y estrategias de la "empresa" contribuiría con el desarrollo de software bajo el precepto de optimización de recursos y maximización de ganancias, entendida esta última como la satisfacción de un mayor número de usuarios a través de herramientas funcionales, fáciles de mantener, flexibles y fáciles de usar.

Para esto, se especificarán los elementos que deben considerarse en las Fases 1 y 2 del ciclo de vida de los procesos del negocio [20], así como la terminología propia al dominio, tomando como referencia el trabajo realizado por [21], quienes proponen un marco de referencia para el modelado del negocio en la industria del software. Las demás fases, por su nivel de abstracción, se basan en las fases 1 y 2.

- Fase de Descubrimiento. En esta fase se estudian los procesos del negocio en el caso del e-learning, como por ejemplo: Realizar Consulta, Añadir Conocimiento, Validar Respuestas [12], para lo cual, se deben tomar como referencia las características para las plataformas electrónicas propuestas en la literatura, de esta manera se estaría contribuyendo, en un futuro, a la determinación de

estándares. Los procesos deben ser especificados utilizando BPMN [22], que es la notación gráfica propuesta en BPMI *para describir la lógica de los pasos, funciones y actividades de un proceso de negocio.*

- Fase de Análisis. En esta fase deben analizarse los procesos que emergen de la fase anterior, para modelarlos con las características particulares del dominio. Es aquí, precisamente, donde se tiene la oportunidad de incorporar los elementos curriculares y de administración educativa, en general, que caracterice a la instancia para la cual se desarrolla. Hablar de "Educación" conlleva la consideración de políticas, teorías y concepciones que varían de un país a otro, es más, de una institución educativa a otra, por lo que esta fase brinda la oportunidad de considerar aquellas características particulares que son relevantes en el proceso [4].

Una vez realizadas las consideraciones presentadas en los párrafos precedentes, el ciclo de vida de los procesos del negocio, propuestos en [20], puede expresarse como se muestra en la Figura 2.

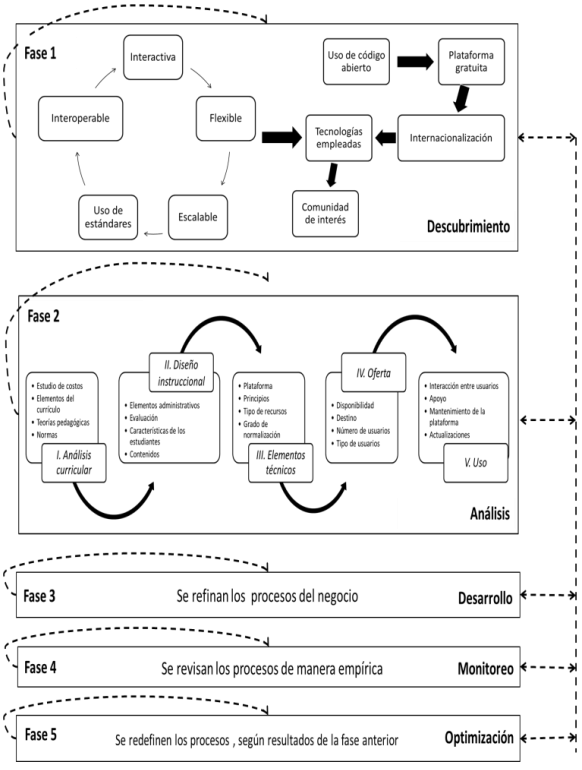


Figura 2: Adaptación de las Fases para el Establecimiento del Modelo del Negocio [4][20]

E. Arquitectura de Referencia para E-Learning

Una AR describe la esencia de la arquitectura de software de una familia de sistemas similares con sus aspectos más significativos y relevantes. El propósito de una arquitectura de referencia es proporcionar una guía para el desarrollo de sistemas concretos reutilizando la AR como un esquema instanciable.

Los sistemas de e-learning deben estar soportados por una arquitectura que cumpla con las siguientes características (sintetizadas en [23]):

- **Abierta**. Debe soportar la interoperabilidad entre distintos proveedores de soluciones y basada en los estándares de organizaciones como AICC, IMS, ADL e IEEE.
- **Escalable**. Sus funciones y uso de recursos deben poder ser ampliadas cuando sea necesario.
- **Global**. Debe poder ser utilizada en cualquier lugar del mundo y en cualquier momento con igual facilidad.
- **Integrada**. Debe integrarse con distintas infraestructuras de red y otras aplicaciones de seguridad, recursos humanos, etc.
- **Flexible o Evolutiva**. Debe poder adaptarse a nuevos requisitos y procesos, nuevas tecnologías y nuevos proveedores de soluciones.
- **Adaptable**. En un mundo en constante desarrollo, debe ser de rápida y fácil implantación en diferentes plataformas de hardware/software, organismos, empresas y entidades educativas.

Hay autores que consideran otros elementos para el diseño de arquitecturas de e-learning. En [24] se plantea la necesidad de la flexibilidad y adaptabilidad de estos sistemas a los cambios del entorno, de las tecnologías y de los contenidos. En este sentido, las últimas propuestas arquitecturales apuntan al diseño de aplicaciones basadas en servicios. Un *servicio* es un tipo de componente de software reutilizable que proporciona a otros componentes o aplicaciones, un conjunto de funciones u operaciones que se invocan a través de una interfaz de programación. La Arquitectura Orientada a Servicios (*Service Oriented Architecture – SOA*), facilita el desarrollo de nuevas aplicaciones reutilizables que resuelven problemas de integración, a través de aplicaciones independientes y distribuidas de manera que desde la red pueda accederse a sus funcionalidades, las cuales se ofrecen como servicios.

III. Proceso MN-AR

El proceso MN-AR propuesto en [4] es un proceso de diseño arquitectónico que permitirá obtener una AR para aplicaciones en el dominio del aprendizaje electrónico que satisfagan las características de calidad deseables en el dominio. En consecuencia, la principal adaptación de [1], además de considerar el modelo de negocio (MN) como paso inicial, viene dada por el hecho de que los productos existentes del dominio e-learning que se estudiaron son ya AR y no sistemas concretos. A continuación se especifica textualmente el proceso MN-AR:

A. Modelado del Negocio (ModN)

Se utiliza BPMI, de acuerdo a las recomendaciones de la Sección II-D, para establecer el MN educativo.

Entrada: la descripción textual del dominio.

Actividades:

- *Se realizan las fases de Descubrimiento y Análisis*, para obtener el MN
- *Se determina (n) el (los) estilo (s) arquitectural* (es) predominante (s) del dominio respectos a los sistemas computacionales existentes en la organización.
- *Se construye el Modelo de Calidad del Dominio*, respectos a los sistemas computacionales existentes en la organización y sensibles de la automatización de los procesos especificados; en nuestro caso EL-DQM (del inglés *E-Learning Domain Quality Model*), ver Figura 1.

Salida: reglas del negocio, MN en BPMN, representado como procesos del negocio, estilo arquitectural, EL-DQM.

B. Análisis del Dominio (AD)

Entrada: MN, estilo arquitectural, EL-DQM,

Actividades:

- *Convertir los procesos de negocios a procesos automatizables*, en términos de los principales componentes, correspondientes a RF de sistemas de software y sus propiedades de calidad o RNF prioritarias del dominio, dadas por el EL-DQM
- *Identificar características de sistemas existentes*, funcionalidades básicas del dominio (RF)
- *Describir las arquitecturas de productos existentes*: en nuestro caso los productos ya son AR, dadas por descripciones arquitecturales, generalmente informales y descritas en notaciones no estándar, proveniente de la práctica industrial y/o académica
- *Análisis de Similitud de componentes:* se realiza un análisis de similitud semántica entre los componentes de las AR; se identifican los componentes provenientes de RF y RNF con sus conexiones para cada AR.

Salida: Tabla de Similitud de componentes con conectores para cada AR.

C. Especificación de las AR en un Lenguaje de Descripción Arquitectónica (del inglés ADL: *Architecture Description Language*), en nuestro caso UML 2.0.

Salida: Representación UML de las AR.

D. Construcción automática de la Arquitectura Candidata (AC) inicial

Entrada: diagramas UML de las AR consideradas, Tabla de Similitud

- Se realiza la unión de los componentes sobre los diagramas UML, y se genera automáticamente la representación arquitectónica en UML, preservando las conexiones

Salida: AC representada como un diagrama UML

E. Construcción del Modelo de Calidad Extendido o "Extended Quality Model" (EQM), Tabla EQM.

Entrada: AC

- Se estudian los componentes arquitecturales de AC, analizando sus posibles metas no funcionales requeridas (propiedades o requisitos de calidad a las cuales los componentes deben responder), que podrían no estar presentes en el EL-DQM, si son derivadas de componentes funcionales
- Se asignan soluciones arquitecturales o mecanismos computacionales, que pueden ser requeridos por cada componente funcional para satisfacer la propiedad de calidad, provenientes de la tecnología o de catálogos y se asignan prioridades de acuerdo a su importancia respecto al dominio.
- Se determinan posibles condiciones de obligatoriedad u opcionalidad de las componentes.

Salida: EQM representado como una tabla

F. Completar AC

- A partir de la tabla EQM, se estudian las propiedades requeridas por las funcionalidades o componentes de AC y se seleccionan las soluciones arquitecturales que las pueden satisfacer, considerando todas las alternativas posibles
- Nuevos componentes pueden ser introducidos para realizar o agrupar varias soluciones arquitecturales, las cuales se denominarán variantes o <<variantes>>

Salida: AC completada

G. Diseño del Dominio (DD)

Entrada: AC completada

- *Construcción del Modelo de Variabilidad*. Se agrupan componentes que realizan tareas similares en componentes genéricos denominados *puntos de variación* o *<<variation points>>;* son los que serán instanciados en el momento de derivar un producto o sistema concreto a partir de la AR
- *Construcción de la AR*
- *Adecuación de la AR.* Se adecúa la AR a un estándar, si lo hay; en nuestro caso es la IEEE-LTSA [8].

Salida: AR representada por un diagrama UML

En resumen, el proceso MN-AR extiende la propuesta realizada en [1], incorporando el modelo del negocio, descrito en la Sección II.D, pues se considera que es allí donde se presentan las características relevantes del dominio; se utiliza la refactorización de AR existentes y no se utiliza el enfoque de metas como en [1][10][13], sino un enfoque basado en escenarios, representados por la Tabla EQM (del inglés *Extended Quality Model)* [4], donde se satisfacen las propiedades de calidad, exigidas por los componentes funcionales, por otros componentes que representan o agrupan mecanismos del mercado o soluciones arquitecturales.

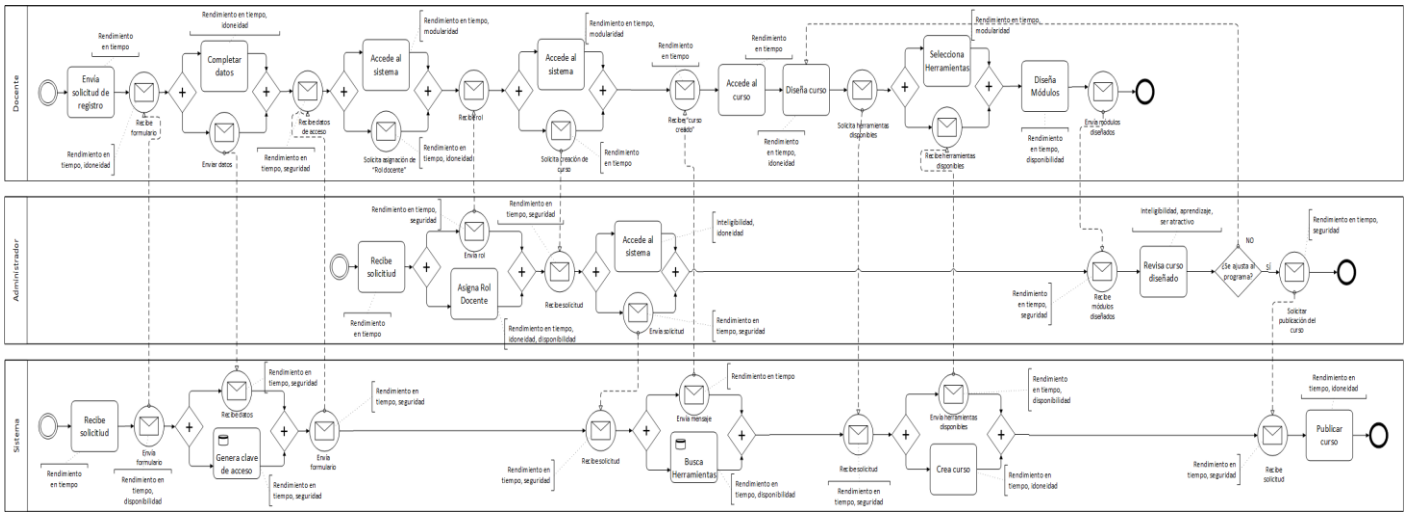


Figura 3: Modelo de Negocio Extendido: BPMN 2.0 con Características de Calidad. Proceso Crear Contenido (Autoras)

IV. CASO DE ESTUDIO

A continuación se iniciará la aplicación de MN-AR completo a un caso de estudio, con el fin de ofrecer una primera validación del proceso. Como parte esencial del modelo de negocio se dará la descripción de los flujos de dos procesos considerados relevantes para la concretización del e-learning: *Realizar Consulta* y *Crear Contenido.*

A. Modelado del Negocio (ModN)

Entrada: en el proceso *Crear Contenido* interviene el docente o grupo de docentes involucrados en el curso, el administrador del sistema y el sistema de software; mientras que en el proceso *Realizar Consulta* solo interactúa el estudiante con el sistema. Estos procesos no son necesariamente consecutivos, entre ellos existen otros procesos que no se discutirán en este artículo para simplificar su presentación.

Fase de Descubrimiento

- Proceso: *Crear Contenido*. Los actores a considerar son el sistema, el administrador y el docente.

  El proceso inicia cuando el docente *accede al sistema* para diseñar un nuevo curso virtual => el docente *se registra* como usuario => el administrador del sistema le *asigna el rol* de docente. Paralelamente, se *crea el curso* en el sistema => el docente *recibe estructura de módulos*; *diseña los módulos* del curso y las *estrategias de comunicación* a utilizar => el docente *sube recursos* al sistema: documentos, enlaces y demás recursos para el curso => el docente establece el *calendario de actividades.*

Estas actividades se integran a nivel de sistemas de software en los subsistemas o componentes de Infraestructura-Servicos Comunes y Servicios para el aprendizaje de la AR, según se plantea en [5][8] y veremos en la sección B. Analisis del Dominio. Debido a que BPMN 2.0 no posee una notación para expresar RNF, cada actividad en la Figuras 3 y 4 están relacionadas mediante un comentario, presentado en corchetes, con las propiedades de calidad del modelo EL-DQM de la Figura 1, que deben ser cumplidas por la respectiva actividad para tener la idoneidad funcional del componente, es decir que cada tarea se cumpla a cabalidad de acuerdo a lo especificado. Ellas son descritas en la fase de análisis.

- Proceso: *Realizar Consulta*. Los actores a considerar son el sistema y los estudiantes.

  El proceso inicia cuando los estudiantes solicitan *acceder al sistema* para consultar un tema en específico => el sistema *recibe la solicitud y analiza* la lista de temas => el sistema *muestra los contenidos* disponibles => el estudiante *selecciona el contenido* con el que va a trabajar => el sistema *muestra el contenido y las preguntas* relacionadas => el estudiante *responde la pregunta* => el sistema *indica si es o no correcta* y presenta la opción de responder otra pregunta => finaliza el proceso cuando el estudiante *recibe la estadística de respuestas acertadas*.

Al igual que en el proceso *Crear Contenido*, estas actividades se integran en los subsistemas o componentes de *Infraestructura-Servicios Comunes* y *Servicios para el aprendizaje* [5] y [8]; igualmente se especifican en BPMN las propiedades de calidad referentes a estas actividades que deben ser cumplidas. Pueden verse en la Figura 1.

Fase de Análisis

- Proceso: *Crear Contenido*.

  RF: *Registrar Usuario, Asignar Rol, Recibir estructura de módulos, Diseñar Módulos, Subir Recursos, Establecer Calendario de Actividades, Publicar Curso* (Creación y Diseño de Curso)

  RNF o metas de calidad asociadas a las actividades del proceso: *rendimiento en tiempo, idoneidad, inteligibilidad, aprendizaje, ser atractivo, seguridad de*

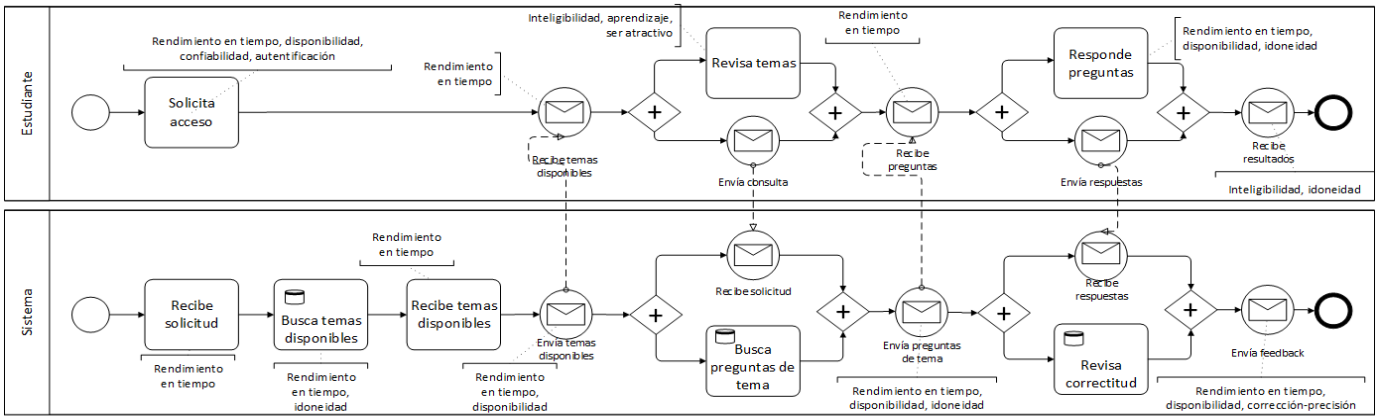


Figura 4: Modelo de Negocio Extendido: BPMN 2.0 con Características de Calidad. Proceso Realizar Consulta (Autoras)

acceso, disponibilidad del sistema y modularidad (ver Figura 1). En la Tabla II. EQM, se justifica el cumplimiento de estas propiedades relativas a los componentes de *Infraestructura y Servicios para el Aprendizaje*.

Reglas del negocio que pueden afectar el proceso: proporcionar contenido adecuado según programa analítico del curso. Será verificado por el Administrador en la actividad *Revise Curso Diseñado*.

- Proceso: *Realizar Consulta*

  RF: *acceder al sistema, recibir solicitud, analizar lista de temas, mostrar contenidos, mostrar preguntas, responder preguntas, indicar corrección de respuesta, recibir estadísticas de respuestas.*

  RNF o metas de calidad asociadas a las actividades del proceso: *disponibilidad* del sistemas, *seguridad de acceso (confiabilidad, autentificación), rendimiento en tiempo, corrección-precisión* respecto a las respuestas, *idoneidad* o completitud de las tareas realizadas, *inteligibilidad, aprendizaje, ser atractivo* para el estudiante. También se justifican en la Tabla II. EQM.

  Reglas de negocio que pueden afectar al proceso: Proporcionar acceso adecuado al contenido. El sistema debe garantizar seguridad y tiempo en las actividades *enviar solicitud de acceso y recibir petición de curso*.

Al culminar con la fase de análisis, se refinan los procesos del negocio (Fase 3), para luego revisarlos (Fase 4), y eventualmente optimizarlos (Fase 5), hasta obtener todos los procesos del negocio definitivos (ver Figuras 3 y 4). En este trabajo se muestran esencialmente las fases de Descubrimiento y Análisis.

Estilos arquitecturales del Dominio

*Los estilos arquitecturale*s predominantes del dominio del aprendizaje electrónico son los estilos híbridos SOA/Capas, bajo plataforma tipo LAMP [4][5] (Figura 5). Nótese que las propiedades de calidad *escalabilidad, adaptabilidad e interoperabilidad* (ver Tabla II. EQM) se satisfacen debido al uso de estilos arquitecturales basados en servicios; estas propiedades no se reflejan de los componentes funcionales, sino que son satisfecho por el estilo arquitectural utilizado.

Modelo de Calidad del Dominio

El modelo de calidad del producto de software, en el caso del e-learning (EL-DQM) presentado en la Figura 1, es extraído de las características del dominio y adaptadas de acuerdo al estándar ISO/IEC 25010 [4][5][12].

Salida: reglas del negocio, MN en BPMN, representado como procesos del negocio (Figuras 3 y 4), estilo arquitectural (Figura 5) y EL-DQM (ver Figura 1).

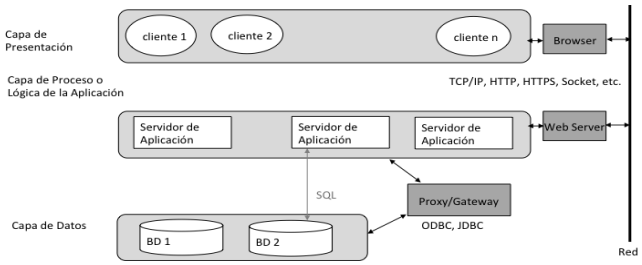


Figura 5: Arquitectura Hibrida basada en Eventos, SOA/Capas que se Adapta al Dominio E-learning [4]

B. Análisis del Dominio (AD)

Convertir los Procesos de Negocio

Las funcionalidades básicas del dominio e-learning en la fase de análisis de MN, según la IEEE-LTSA [8], deben ser identificadas a partir de los procesos de negocio (Figuras 3 y 4); para este trabajo, se identificaron los siguientes requisitos funcionales, comunes para ambos procesos, que dan origen a las funcionalidades básicas:

- Crear, operar y administrar actividades de aprendizaje en línea => sistemas para gestión y creación de contenidos LMS, LCMS (Componentes de Servicios de Aprendizaje)
- Soportar la colaboración entre los usuarios => sistemas colaborativos COLLS (Componentes de Servicios de Aprendizaje)
- Crear y proporcionar preguntas y pruebas para la evaluación del aprendizaje del estudiante => sistemas de evaluación ASSES (Componentes de Servicios de Aprendizaje)

- Organizar recursos humanos y financieros => sistemas de recursos humanos HRS (Componentes de Servicios Comunes)
- Administrar experiencias de aprendizaje virtual y distribuido vía Internet a estudiantes geográficamente distantes => sistemas de mails, gestión de usuarios y portales MAILS, UMS y Portal (Componentes de Servicios Comunes e Infraestructura)

Por lo tanto estos componentes funcionales formarán parte de nuestra arquitectura AC inicial, AC completada y finalmente de la AR para e-learning como se verá en los pasos siguientes del proceso MN-AR.

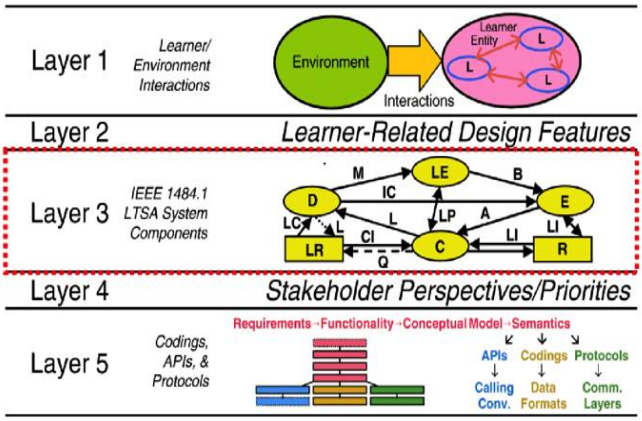


Figura 6: LTSA: IEEE P1484-1 Standard [8]

Identificar características de sistemas existentes

Una AR para e-learning debería adaptarse al estándar IEEE-LTSA [8], el cual describe una arquitectura en cinco capas de alto nivel para sistemas de aprendizaje basados en tecnología de información. Describe una solución en términos de una perspectiva de información tecnológica y es por lo tanto independiente respecto a aspectos pedagógicos, de contenido, culturales y de plataforma [8] (ver Figura 6).

La capa 3 de esta arquitectura es obligatoria para este estándar y debe estar presente en cualquier AR que se diseñe; identifica los componentes de la arquitectura (funcionalidades principales que deben estar presentes) en cuatro procesos:

D: Delivery (Entrega), LE: Learning Entity (Entidad de Aprendizaje), E: Evaluation (Evaluación) y C: Coach (Monitoreo); y dos almacenamientos de aprendizaje: Recursos y Registros.

Las relaciones son M: Multimedia, B; Behavior (Comportamiento): IC: Interaction Context (Contexto de Interacción), LP: Learning Parameters (Parámetros de Aprendizaje), L: Locator (Localizador), A: Assessment (Evaluación), LC: Learning Content (Contenidos), CI: Catalog Info (Información del Catálogo), Q: Query.

Describir las arquitecturas de productos existentes

En [4] se consideró la proposición de [5], cuya AR integra los frameworks UKeU [17] y WbIS [7] y que además se adecúa a la IEEE-LTSA (Figura 6) [8]; se consideran tres subsistemas o componentes principales:

- *Infraestructura & Servicios Comunes o Infrastructure & Common Services:* Portal (UI), Servicios Comunes (UMS: User Management System, MAILS: Mail, HRS: Human Rsources), Base de Datos (DBMS: Data Base Management Systems).
- *Servicios para el aprendizaje o e-learning services*: LMS, LCMS, sistema de evaluación (ASSES: Assessment System), sistema colaborativo (COLLS: Collaborative System), sistema de planificación (no fue considerado en esta AC como sistema aislado, es incluido con frecuencia en los sistemas ASSES).
- *Recursos para el aprendizaje o e-learning resources*: material en línea, material impreso, CD, DVD; estos aspectos se manejan en los sistemas LMS y LCMS.
- *Recursos Humanos o Human Resources*: Usuarios (estudiante, instructor o docente, administrador de la comunicación en línea, administrador del entrenamiento, autor, administrador del sistema), empresa u organización cliente, equipo de desarrollo, equipo de mantenimiento y soporte. Estos aspectos se consideran en los sistemas UMS y HRS.

Tabla I: Componentes de la AR Obtenida Aplicando MN-AR, con sus Conectores

Componentes	*Conectores*
P1. Portal	P1RC1, P1RC2
L1. Learning Services	L1RL2, L1RC11, L1RC12
L11. LMS	L11RL1, L11RL12, L11RL13, L11RL14, L11RL1
L12. LCMS	L12RL11, L12RL1
L13. COLLS	L13RL11, L13RL1
L14. ASSES	L14RL11, L14RL1
L2. Common Services	L2RL1, L2RC11, L2RC12
L21. HRS	L21RL22, L21RL2, L21RL2
L22. UMS	L22RL21, L22RL2, L22RL2
L23. MAILS	L23RL2
D1. DBMS	D1RL1, D1RL2
C1. Networks	C1RP1, C1RC11, C1RC12
C11. Internet	C11RP1, C11RL1, C11RL2
C12. Satellite	C12RP1, C12RL1, C12RL2

Estos componentes (ver Tabla I), de alto nivel, deben ser articulados de acuerdo al estilo arquitectónico considerado para el dominio e-learning en la etapa del Modelado del Negocio, es decir un estilo hibrido SOA/capas típico para sistemas o aplicaciones Web, siguiendo una plataforma tipo LAMP (ver Figura 5). Nótese que las arquitecturas son grafos conexos, es decir sin componentes aislados.

- *Análisis de Similitud de Componentes*. Este paso no es necesario ya que la AR considerada toma en cuenta ya dos AR y es conforme a la IEEE-LTSA.

C. Especificación de las AR en UML

Salida: Ver Figura 7 donde se muestra una arquitectura en tres capas, más la capa de comunicación o transmisión, constituida por las redes (Internet/intranet, satelital, etc.), que entrecruza a las tres capas. Nótese que los nombres de los

Tabla II: Tabla EQM (Extended Quality Model) – Adaptado de [4] al Caso de Estudio

Componentes	*Variantes*	*Descripción*	*Propiedad de Calidad con prioridad en ()* *Requiere*	*Proporciona*	*Restricciones*
P1. Portal	No	Es la componente Interfaz Usuario (UI) a través de la cual se tiene acceso al sistema	- inteligibilidad (1), - aprendizaje (1), - ser atractiva (2), - protección de errores (3) - confiabilidad (2), autentificación (1) - integridad (2)	P1: técnicas de diseño de páginas, no se evalúan a nivel arquitectural P11: mecanismo corrector C1: SOAP/HTTP/ LDAP/HTTPS/Web service	CC: Componente Común funcional, obligatoria
P11. AutoCorrector	Si (*)	Provee autocorrección al introducir texto	Servicio Web	Mecanismo	Opcional
L1. Learning Services	No	Agrupa sistemas que ofrecen servicios pare el proceso de aprendizaje	Por cada sistema componente:	Proporcionadas para cada sistema componente:	CC funcionales opcionales u obligatorios
L11. LMS	Si (*)	Sistemas de gestión de aprendizaje: permite administrar usuarios, recursos, contenidos (importar/exportar) y actividades de formación, administrar acceso, controlar el proceso de aprendizaje, generar informes, administrar servicios de comunicación; no incluyen autoría;	- idoneidad (1), - corrección-precisión (2), - rendimiento en tiempo (2) - capacidad-escalabilidad (1) - adaptabilidad (1) - disponibilidad-persistencia (2) -interoperabilidad (1)	L1: por construcción L1: Comptuation modules C1: HTTP/SOAP/FTP Messaging/SOAP/RPC/ Web service API ODBC/JDBC L1: LMS copy L1: AICC/SCROM	CC funcional, obligatorio variantes: SumTotal, Saba, OLAT, Sakai CLE, ATutor por ser sistema crítico
L12. LCMS	Si (*)	Sistemas de autoría: creación de contenidos; permite a los autores registrar, ensamblar, administrar y publicar/exportar contenidos para ser entregados vía Web; Variantes: Evolution, ForceTen	- idoneidad (1), - corrección-precisión (2), - comportamiento en tiempo (2) - capacidad-escalabilidad (1) - adaptabilidad (1) - disponibilidad-persistencia (2) -interoperabilidad (1) - modificabilidad (2), - reuso (3)	L1: por construcción L1: Comptuation modules C1: HTTP/SOAP /Messaging/SOAP/RPC API ODBC/JDBC L1: LCMS copy L1: AICC/SCROM L1: por construcción L1: por construcción	CC funcional, obligatorio variantes: *Evolution, ForceTen, Eduslide* por ser sistema crítico
L13. COLLS	Si (*)	Sistema colaborativo: proporciona funcionalidades para crear y administrar sesiones colaborativas incluyendo comunicación síncrona (video-conferencias, salón virtual) y asíncrona (blog, wiki, grupos, chats)	- comportamiento en tiempo (2), - disponibilidad-persistencia (2), - adaptabilidad (1), - capacidad-escalabilidad (1) - autentificación (1), - confidencialidad (2) - integridad (2)	C1: protocolos HTTP/SOAP T.120/H.323 LDAP/HTTPS/Web service	CC funcional, variantes: Alfresco, Asana, Box.net, Clearspace, Drupal, Google Drive, Huddle
L14. ASSES	Si (*)	Sistema de evaluación: permite a los autores crear “surveys”, evaluación formativa y sumativa , exportar una evaluación vía AICC o SCORM, para ser entregada vía Web	- idoneidad (1), - corrección-precisión (2), - comportamiento en tiempo (2) - capacidad-escalabilidad (1) - adaptabilidad (1) - disponibilidad-persistencia (2) - autentificación (1), - confidencialidad (2) - integridad (2)	L1: por construcción L1: Computation modules C1: HTTP/SOAP/ Messaging/RPC API ODBC/JDBC LDAP/HTTPS/Web service	CC funcional, obligatorio variantes: sistemas LMS (son los mismos LMS)
L15. Computation modules	Si (*)	Garantiza la precisión en los sistemas que requieren cómputos	Componente	Módulos	variantes: pueden haber muchos algoritmos Obligatorios
L16. LMS copy	Si	Garantiza la disponibilidad en caso de falla	Subsistema	Componentes	Opcional
L17. LCMS copy	Si	Garantiza la disponibilidad en caso de falla	Subsistema	Componentes	Opcional
L18. AICC	Si	Garantiza importación/exportación	Componente	API	Obligatorio
L19. SCROM	Si	Garantiza importación/exportación	Componente	API	Obligatorio
L2. Common Services	No	Agrupa sistemas que ofrecen servicios a los que todos pueden acceder	Por cada sistema componente:	Proporcionadas para cada sistema componente:	CC funcionales, obligatorios
L21. HRS	Si (*)	Sistemas de recursos humanos:	- disponibilidad-	C1: Messaging/SOAP/FTP/	CC funcional,

Tabla II: Tabla EQM (Extended Quality Model) – Adaptado de [4] al Caso de Estudio

Componentes	*Variantes*	*Descripción*	*Propiedad de Calidad con prioridad en ()* *Requiere*	*Proporciona*	*Restricciones*
		permite administrar el perfil del usuario final incluyendo habilidades, competencias y tipo de trabajo; crear y mantener los registros de planes personales de desarrollo del usuario	persistencia (2), - capacidad-escalabilidad (1) - adaptabilidad (1), - autentificación (1), - confidencialidad (2)	API ODBC/JDBC LDAP/HTTPS, Web Service	obligatorio variantes: Orange HRM, Centrifugo, SimpleHRM
L22. UMS	Si (*)	Sistemas de gestión de usuarios; administra usuarios, grupos y roles a partir de todos los componentes Involucrados en la solución; maneja el registro y cuenta del usuario, la autorización y la autentificación	- disponibilidad-persistencia (2), - capacidad-escalabilidad (1) - adaptabilidad (1), - autentificación (1), - confidencialidad (2)	C1: Messaging/ SOAP/FTP API ODBC / JDBC LDAP/HTTPS, Web Service	CC funcional, obligatorio variantes: Apache Syncope, OpenIAM
L23. MAILS	Si (*)	Sistema e-mail: es responsable de enviar, recuperar y reenviar e-mails a los diferentes componentes involucrados en la solución.	- comportamiento en tiempo (2), - disponibilidad-persistencia (2), - adaptabilidad (1) - capacidad-escalabilidad (1) - autentificación (1), - confidencialidad (2)	C1: SOAP/ HTTP/Messaging, /FTP/ T.120,/H.323 API ODBC/JDBC LDAP, HTTPS, Web service	CC funcional, obligatorio variantes: gmail, hotmail, yahoo, cantv.net,
C1. Networks	No	Agrupa los tipos de redes	Requeridas por cada componente:	Proporcionadas para cada componente:	CC funcionales, obligatorios
C11. Internet	Si	conjunto descentralizado de redes de comunicación interconectadas, basadas en protocolos TCP/IP, que hace ver las redes físicas heterogéneas que la componen como una red lógica única de alcance mundial.	- comportamiento en tiempo (2), - disponibilidad-persistencia (2), - adaptabilidad (1), - capacidad-escalabilidad (1) - autentificación (1), - confidencialidad (2) - integridad (2)	C4. Network protocols: SOAP/HTTP/FTP/T.120/H.323 /Messaging (Publisher-Subscriber) LDAP/HTTPS/Web Services	CC, obligatorio variante: Internet
C12. Satellite	Si	es un método de conexión a redes de comunicación (ej. Internet) utilizando como medio de enlace un satélite.	- comportamiento en tiempo (2), - disponibilidad-persistencia (2), - adaptabilidad (1), - capacidad-escalabilidad (1) - autentificación (1), - confidencialidad (2) - integridad (2)	C4. Network protocols: SOAP/HTTP/FTP/T.120/H.323 /Messaging (Publisher-Subscriber) C2. ODBC C3. JDBC C5. Security protocols: LDAP/HTTPS/Web Services	CC funcional, opcional Messaging is implementado por un canal Publisher-Subscriber variante: Satellite
C2. ODBC	Si	Garantiza portabilidad	API	Mecanismo	obligatoria
C3. JDBC	Si	Garantiza portabilidad	API	Mecanismo	obligatoria
C4. Network protocols	Si (*)	Garantizan propiedades de la comunicación	Protocolos	Mecanismos, servicios Web	obligatorias variante: múltiples dependen del mercado
C5. Security protocols	Si (*)	Garantizan las propiedades para el control de acceso al sistema y la consistencia de mensajes	Protocolos	Mecanismos, servicios Web, sistemas	obligatorias variante: múltiples dependen del mercado
D1. DBMS	Si (*)	Sistemas Gestores de Bases de Datos; las diferentes bases de datos utilizadas por los subsistemas son consideradas un almacenamiento persistente compartido de datos: D11. LMSDB, D12. LCMSDB, D13. CALLSDB, D14. ASSESDB, D15. HRSDB, D16. UMSDB	- capacidad-escalabilidad (1) - disponibilidad-persistencia (2) -interoperabilidad(1) - integridad (2) - adaptabilidad (1)	D1. módulos, mecanismos propios de las BD C1: API ODBC/JDBC	CC funcional, obligatorios; cada base de datos es obligatoria. Solo BD relacionales son consideradas; variantes: Oracle, MySQL, PostgreSQL, Firebird
D17. DB mechanisms	Si (*)	Garantizan propiedades de los BDMS	DBMS	Componentes	variantes: específicas para cada DBMS

(*) indica varios sistemas diferentes; cada uno es una variante

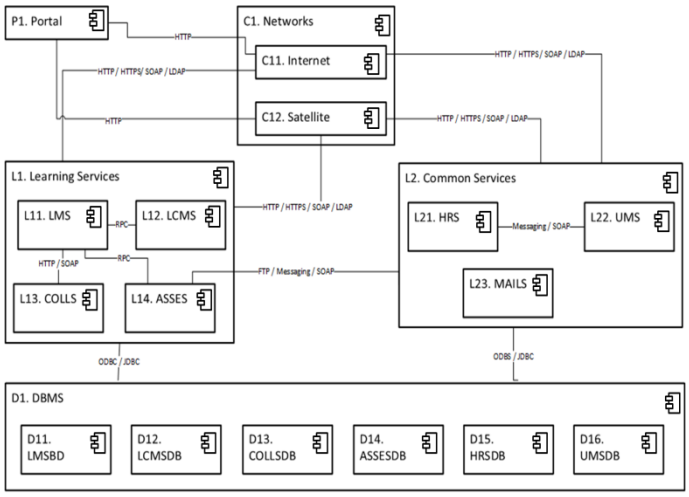


Figura 7: Arquitectura Candidata (AC) en UML Generada Automáticamente a Partir de la Especificación de las AR Existentes (Autores)

componentes son numerados secuencialmente de acuerdo a la capa en donde están situados, P: Presentación, L: Lógica o Proceso, C: Comunicación, y D: Datos y se han mantenido los nombres en inglés para efectos de divulgación internacional.

D. Construcción automática de la AC inicial.

Nuestra AC es la que se muestra la Figura 7. Solo las funcionalidades están presentes en esta AC inicial, pues en la AR que se tomó de [7] no se incluían componentes no funcionales, derivados de las propiedades de calidad; estas son especificadas en la Tabla EQM a continuación.

E. Construcción de la Tabla EQM

En la Tabla II se analizan los componentes de la AC, sus variantes, descripción, las propiedades de calidad requeridas y proporcionadas, indicando prioridad y restricciones asociadas.

F. Completar AC

A partir de la Tabla EQM (Tabla II) se completa la AC inicial obtenida en D, la cual se muestra en la Figura 8; las variantes son denotadas por el estereotipo <<variant>>. Los componentes se han introducido para satisfacer propiedades de calidad requerida por algún componente funcional ya existente lo que garantiza la trazabilidad entre los RF y los RNF.

G. Diseño del Dominio (DD)

- *Construcción del Modelo de Variabilidad*

  A partir de la AC completada se obtienen los puntos de variación, denotados por <<vp>>: <<C1>> Networks que agrupa C11, C12; <<C6>> SecurityProt agrupa los mecanismos para la seguridad, <<C7>> AdaptabAPI agrupa los mecanismos para la portabilidad C4. JDBC y C5. ODBC, <<C8>> NetProt agrupa otros protocolos de red para tratar tiempo de respuesta, disponibilidad y capacidad en la transmisión, y <<D18>> RDBMS que agrupa DBMS relacionales y <<D19>> DBMech que agrupa mecanismos propios a los DBMS para satisfacer las propiedades de calidad capacidad, disponibilidad (por ejemplo bases de datos espejo o replicación), interoperabilidad e integridad; a efectos de legibilidad y abreviar la presentación, las soluciones se dejaron agrupadas en un solo <<vp>>. Todos los subsistemas de L1 y L2 son renombrados como estereotipos por desempeñar las mismas tareas, <<L111>> LMS, <<L121>> LCMS, <<L131>> COLLS, <<L141>> ASSES, <<L211>> HRS, <<L221>> UMS, <<L231>>> MAILS; <<L161>> Avail agrupa L16. LMS copy y L17. LCMS copy, <<L151>> AlgorPrec agrupa algoritmos de cálculo, <<L181>> Interop agrupa L18. AICC y L19. SCROM; finally <<P111>> ErrorProt agrupa variantes de sistemas de auto corrección.

Tabla III: Adecuación a IEEE-LTSA de la AR para E-learning Obtenida Aplicando MN-AR al Caso de Estudio

AR LTSA capa 3	*Infraestructura Técnica*	*Recursos Humanos*	*Recursos de Aprendizaje*
L: Learning Entity	Portal, LMS, LCMS, ASSES, UMS	Estudiante	Material de aprendizaje
D: Delivery	LMS, LCMS	Estudiante, Instructor/Autor	LMSDB, LCMSDB
E: Evaluation	ASSSES, UMS	Instructor	ASSESDB, UMSDB
C: Coach	ASSES, COLLS, HRS, MAILS	Instructor, Supervisor	HRSDB, COLLSDB, ASSESDB
LR: Learning Resources	DBMS	Administrador	LMSDB, LCMSDB, COLLSDB
R: Register	DBMS	Administrador	UMSDB, HRSDB

- *Construcción de la AR.* Como se observa de la Figura 9, la AR cumple con las propiedades de calidad especificadas en EL-DQM; en cuanto a los conectores que relacionan los <<vp>>, estos pueden ser también un conjunto de variantes. La AR así obtenida se adapta perfectamente al caso de estudio aquí presentado
- *Adecuación con la IEEE-LTSA.* La Tabla III muestra la adecuación o conformidad de nuestra AR con la capa 3 obligatoria de la IEEE-LTSA [8] (ver Figura 6).

V. CONCLUSIONES

En este trabajo se ha realizado, a través de un caso de estudio sencillo compuesto por dos procesos de negocio relevantes al dominio e-learning *Crear Contenido* y *Realizar Consulta*, la aplicación paso a paso del proceso MN-AR [4] que permite la obtención de una AR de carácter evolutivo para el dominio e-learning, considerando sus funcionalidades básicas y la trazabilidad con las propiedades de calidad que son requeridas para que estas funcionalidades se desempeñen adecuadamente. Esta propuesta sugiere, como trabajo futuro a corto plazo, una representación más formal, como un modelo conceptual en UML, de los elementos curriculares expresados como reglas de negocio que deben estar presentes para garantizar la obtención de sistemas de aprendizaje electrónicos adecuados, así como el diseño de herramientas de soporte.

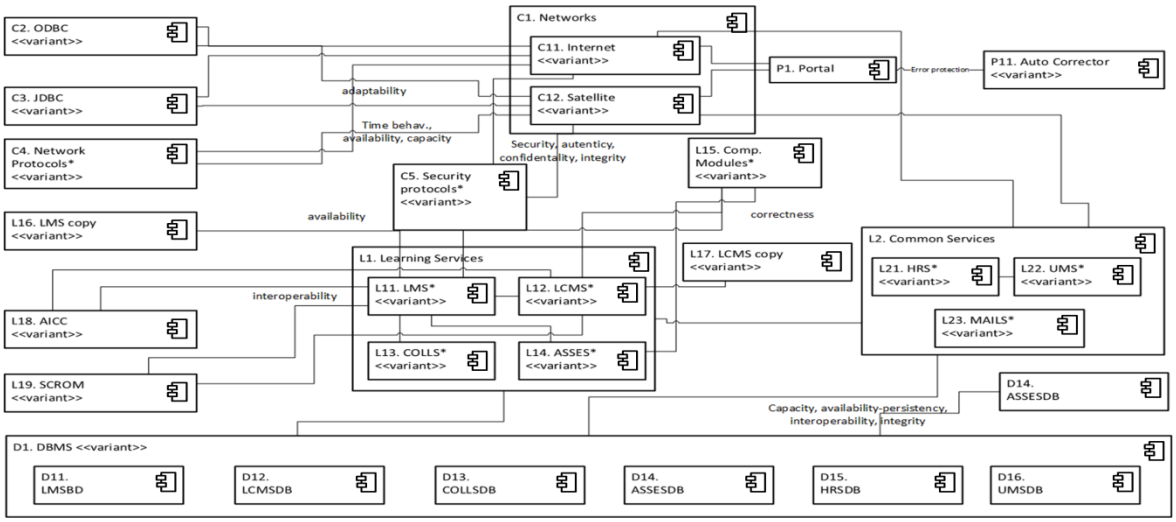


Figura 8: Arquitectura Candidata (AC) Completada en UML (Autores)

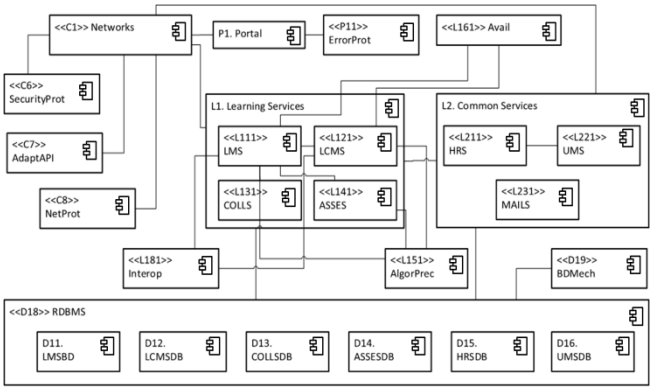


Figura 9: AR en UML para el Dominio E-learning (Autores)

AGRADECIMIENTOS

Al Postgrado en Ciencias de la Computación, Facultad de Ciencias y CDCH (Consejo de Desarrollo Científico y Humanístico) de la UCV, DARGRAF PG 03-8730-2013-2.

REFERENCIAS

[1] F. Losavio, O. Ordaz, and V. Esteller. *Refactoring-Based Design of Reference Architecture*. RACCIS 5(1), pp. 32-48, 2.2015

[2] E. Nakagawa, P. Antonio, and M. Becker. *Reference Architecture and Product Line Architecture: a Subtle but Critical Difference*, Crancov ic V., Grhun, and M. Book (Eds.): ECSA 2011, LNCS 6903, pp. 207-211, Springer-Verlag, Berlin, Heidelberg.

[3] E. Bérard. *Testing of Object-Oriented Software*. Proceedings of the eighth international conference on Technology of object oriented languages and systems. EEUU: Prentice-Hall, Inc. 1992.

[4] Y. Esteves and F. Losavio. *Modelado del Negocio Dirigido por la Calidad para una Arquitectura de Referencia en el Dominio del Aprendizaje Electrónico*. RACCIS 6(1), pp. 59-76, 2016.

[5] J. Habraken. *Reference Architecture for E-learning Solutions*, Master Thesis, Open Univ. Faculty Comp. Science, Jan 28, 2008.

[6] ForceTen Architecture version 4.1; whitepaper from EEDO

[7] S. Retalis and P. Avgeriou. *Modelling Web-based Instructional Systems*, Journal of Information Technology Education, 1(1), 2002.

[8] IEEE. *Draft Standard for Learning Technology – Learning Technology Systems Architecture*, Technical Report IEEE-Std Draft-P1484-1, 2002.

[9] F. Losavio, A. Matteo, and I. Pacilli. *Unified Process for Domain Analysis Integrating Quality, Aspects and Goals***.** CLEI Electronic Journal, 17(2), paper 1, 21 pages, August, 2014.

[10] S. Supakkul and L. Chung. *Integrating FRs and NFRs: A Use Case and Goal Driven Approach*, 2nd ICSE, 30-37, S. Fco. USA, 2004.

[11] ISO/IEC 25010. *Systems and Software Engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE). Systems and Software Quality Models*, ISO/IEC JTC1/SC7/WG6. Ginebra, 2011.

[12] F. Losavio and Y. Esteves. *Modelo del Negocio para Análisis del Dominio del Software Educativo: un Enfoque Centrado en la Calidad del Producto*. Por aparecer en el No. 16, Revista Sapiens UPEL, 2015.

[13] F. Losavio, L. Chirinos, A. Matteo, and L. Ramdane-Cherif. *Designing Quality Architecture: Incorporating ISO Standards into the Unified Process*. Journal of ISYM, 21(1), 27-44. Johannes Gutenberg: Universität Mainz, 2004.

[14] S. Otón. *Propuesta de una Arquitectura Software Basada en Servicios para la Implementación de Repositorios de Objetos de Aprendizaje Distribuidos*, Tesis Doctoral, Universidad de Alcalá, España, 2006.

[15] C. Pagés, J. J. Martínez, and R. Barchino. *Situación Actual de Estandarización de Procesos de Aprendizaje y su Relación con la Información sobre el Alumno*. White Paper: Universidad de Alcalá, España, 2004.

[16] R. Peredo, A. Canales, L. Balladares, and A. Menchaca. *Arquitecturas para Sistemas de Educación basada en Web usando Programación Orientada a Componentes*. Revista Diálogo Educ., Curitiba, 8(24), pp. 485-502, mayo/agosto 2008.

[17] UK e-Universities Worldwide. *Principles and Practice in E-Learning Platform Architecture*, UKeU, 2002.

[18] UNESCO. *Aprendizaje Abierto y a Distancia. Consideraciones sobre Tendencias, Políticas y Estrategias.* Ediciones Trilce. Uruguay, 2002.

[19] J. Boneu. *Plataformas Abiertas de E-learning para el Soporte de Contenidos Educativos Abiertos.* Revista de Universidad y Sociedad del Conocimiento, 2007.

[20] J. Berrocal, J. García, J. M. Murillo. *Hacia una Gestión del Proceso Software Dirigida por Procesos de Negocio*, 2005.

[21] M. Schief, A. Pussep, and P. Buxmann. *Performance of Business Models: Empirical Insights from the Software Industry*. En: ECIS, Barcelona, Spain, Junio 2012.

[22] OMG. *Business Process Model and Notation (BPMN) 2.0*. 2011.

[23] Cisco. *Model of an E-Learning Solution Architecture for the Enterprise*. Cisco Systems, 2001. http://www.cisco.com/warp/public/10/wwtraining/elearning/learn/whitepaper_docs/solution_architecture_wp.pdf

[24] S. Rosenberg. *Beyond e-Learning*, Pfeiffer, 2006.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 38-46, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Extensiones a Metrópolis para una Emergencia Fuerte

Jose Aguilar[1, 2], Junior Altamiranda[1], Danilo Chavez[2]
aguilar@ula.ve, altamira@ula.ve, danilo.chavez@epn.edu.ec

[1] CEMISID, Universidad de Los Andes, Mérida, Venezuela
[2] Escuela Politécnica Nacional, Quito, Ecuador

Resumen: Este trabajo estudia las características y tipos de comportamiento que se pueden ver en los juegos emergentes, y a partir de esto propone una extensión a Metrópolis. Metrópolis es un juego que permite la construcción de una ciudad, a través de un proceso de toma de decisiones colectivas de los habitantes de la ciudad, sobre los tipos de edificios a construir y ubicaciones de los mismos. En particular, el objetivo de este artículo es estudiar los tipos de emergencia fuerte en Metrópolis, y definir los mecanismos que le permiten adaptarse a las características de sus jugadores, en base a una emergencia fuerte. Esto se hace, manteniendo las capacidades de emergencia débil originales de Metrópolis, que permiten la aparición de patrones urbanos. Una emergencia fuerte, permite la auto-modificación del juego para adaptarse a sus jugadores, mientras que una emergencia débil simplemente muestra la construcción de patrones urbanísticos que surge de la ciudad, derivado de las decisiones colectivas de los jugadores. Particularmente, en este artículo se propone un mecanismo de emergencia fuerte para Metrópolis basado en el aprendizaje de los umbrales de toma de decisión de él. Los resultados sobre la adaptación del juego a los jugadores, derivados de la emergencia fuerte, son muy alentadores.

Palabras Clave: Juegos Emergentes; Emergencia Computacional; Máquinas Inteligentes; Juegos de Ciudades; Sistemas Adaptativos.

Abstract: This work studies the characteristics and types of behavior that can be seen in emerging games, and from this proposes an extension to Metropolis. Metropolis is a game that allows the construction of a city, through a process of collective decision making of the inhabitants of the city, about the types of buildings to be built and their locations. In particular, the objective of this article is to study the types of strong emergencies in Metropolis, and define the mechanisms that allow it to adapt to the characteristics of its players, based on a strong emergency. This is done, maintaining the original weak emergency capacities of Metropolis, which allow the emergence of urban patterns. A strong emergency allows for self-modification of the game to suit its players, while a weak emergency simply shows the construction of urban patterns arising from the city, derived from the collective decisions of players. In particular, in this article we propose a strong emergency mechanism for Metropolis based on learning the decision-making thresholds of it. The results on the adaptation of the game to the players, derived from the strong emergence, are very encouraging.

Keywords: Emerging Games; Computational Emergency; Intelligent Machines; City Games; Adaptive Systems.

I. Introducción

Los recientes avances en inteligencia artificial son fundamentales para el diseño de videojuegos. En particular, el concepto de juego emergente marca el comienzo de una nueva era en los videojuegos [1][2][3]: la capacidad de los juegos para adaptarse al jugador, aunque el hilo central del juego no se vea alterado. En general, la emergencia puede ser definida como un "Comportamiento de un sistema, que emerge de las interacciones entre sus componentes, difíciles o imposibles de predecir" [4].

Existen varios tipos de emergencia [4][5], la débil solamente analiza lo que surge del proceso, mientras que la fuerte estudia los elementos que conllevan a la emergencia. Es decir, la emergencia débil se refiere solamente a observar en el sistema bajo estudio como emergen comportamiento y patrones colectivos (por ejemplo, urbanísticos), derivados de las decisiones locales de sus componentes. Por otro lado, la emergencia fuerte analiza esos comportamientos que emergen, para incluir un proceso de retroalimentación (aprendizaje) en el sistema, que le permita irse adaptando a su contexto.

En los juegos emergentes, la emergencia se basa en la creación dinámica de un mundo, derivado de las acciones del jugador, generando comportamientos en el juego basado en lo que está sucediendo en él. Ambos tipo de emergencia se pueden dar en un juego.

En trabajos anteriores se ha propuesto un juego, llamado Metrópolis, donde patrones urbanísticos en la ciudad emergen de las decisiones que sus habitantes toman [6]. El objetivo de Metrópolis es permitir la auto-gestión de la ciudad, basada en reglas que gobiernan las interacciones entre los habitantes de la misma (los jugadores). De esa manera, Metrópolis parte de

la premisa que las ciudades son auto-gestionadas por decisiones tomadas en conjunto por sus habitantes (jugadores), sin que exista un habitante con un papel más importante, ni una autoridad local (alcalde, gobernador, entre otros.). El juego tiene como objetivo, planificar colectivamente el crecimiento de la ciudad, y en función de ese proceso, observar y medir las cualidades de la ciudad desarrollada, basada en las personalidades de sus habitantes.

Como ya se comentó, en la actual versión de Metrópolis emergen patrones urbanísticos en la ciudad, como resultado de la dinámica del juego [6]. Ese es un tipo de emergencia débil. En este artículo se extiende la noción de emergencia, introduciéndole mecanismos emergentes a la dinámica del juego, para que se adapte a sus jugadores. Para ello, inicialmente se analiza los posibles tipos de emergencia a introducir en Metrópolis. Basado en ello, en este trabajo se presenta una modificación al juego, que le permite a partir de ella, la *emergencia fuerte* de comportamientos.

Este artículo se organiza de la siguiente manera, en la siguiente sección se presentan el concepto de juego emergente y a Metrópolis, luego se analizan los posibles mecanismos emergentes para Metrópolis y se implementa uno en el juego, para culminar con el análisis de las capacidades adaptativas conferidas al mismo provista por dicho mecanismo emergente.

II. Aspectos Teóricos

A. Juegos Emergentes

Emergencia [4]: "es lo que ocurre cuando un sistema compuesto por elementos relativamente simples, se organiza espontáneamente, y sin leyes explícitas, para dar lugar a un comportamiento inteligente". Existen varios tipos de emergencia:

- En la *emergencia débil*, características sistémicas en el nivel "superior" no se pueden predecir, a pesar del conocimiento que se tenga de las características y leyes que rigen los componentes de ese sistema. Así, normalmente uno se limita a observar ese comportamiento que emerge. Por ejemplo, viendo a una ciudad (nivel superior) como un sistema emergente débil, solo se constatarían los patrones urbanísticos que emergen en ella.
- En la *emergencia fuerte*, fenómenos emergentes pueden obtener nuevas capacidades causales que hace posible que los sistemas puedan ejercer una influencia de arriba hacia abajo. Un sistema *fuertemente emergente* es aquel en el que los niveles superiores de complejidad poseen características genuinas, que están ausentes de las partes constitutivas, pero que los pueden enriquecer. Por ejemplo, viendo a una ciudad (nivel superior) como un sistema emergente fuerte, ella tiene propiedades específicas que configuran un proceso de retroalimentación que permitirán amplificar o disipar los patrones urbanísticos de la ciudad (por ejemplo, las migraciones de actividades económicas o sociales en zonas de una ciudad).

En particular, un juego emergente responde a las acciones del jugador, lo que añade una nueva dimensión al juego [2][3][7][8]. Esto es lo que hacen juegos como SimCity, Lincity y Los Sims [9][10][11].

Ahora bien, en la literatura existen varias definiciones de lo que es un juego emergente. Por ejemplo, en [7][8] indican que un juego emergente aparece cuando un conjunto relativamente simple de reglas conducen a estrategias de juego complejos. De manera complementaria, [4] indica que un juego emergente significa que el juego tiene su propia vida de acuerdo a lo que el jugador lleva a cabo, de manera que el juego puede responder a sus acciones, para construir una historia que siempre será esencialmente la misma, pero adaptada a las preferencias del jugador.

En general, un juego emergente requiere un alto nivel de interactividad. La creatividad humana para encontrar soluciones a los múltiples problemas en varios dominios, es lo que busca explotar los juegos emergentes. La idea de "emergencia" es lo opuesto a la "creación", en el sentido de que si algo emerge se lleva a cabo por sí mismo, sin la ayuda de nadie [4][5]. La emergencia en los juegos es posible gracias a la definición de reglas simples globales sobre el comportamiento y la dinámica del mismo, a las propiedades dinámicas de los objetos del juego, así como a las interacciones entre el mundo del juego y el jugador. La emergencia en un juego permite que el mundo del juego se vuelva más interactivo y reactivo, creando una gama más amplia de posibilidades para las acciones, estrategias, etc. En general, las formas de emergencia en un juego se pueden expresar de diferentes maneras [6][7][8]:

- Por la aparición de nuevos comportamientos en el juego.
- Por la aparición de nuevas secuencias/escenarios, tramas o temáticas en los juegos.
- Por el surgimiento de nuevas propiedades en los objetos que los componen. Por ejemplo, en juegos como los Sims, los personajes que van apareciendo tienen sus propias personalidades.
- Por la aparición de patrones que reflejan los resultados finales de los juegos, derivados por las decisiones tomadas por los jugadores. Por ejemplo, en Metrópolis aparecen patrones urbanísticos.
- Por el surgimiento de modelos de negocios alrededor de los juegos. Por ejemplo, en algunos juegos aparece un sistema de comercio para comprar e intercambiar personajes, herramientas, entre otras cosas.

Todo lo anterior surge producto de un conjunto de reglas simples, de las propiedades dinámicas de los objetos del juego, de las decisiones de los jugadores, de las interacciones de los jugadores, entre otras. Los tres primeros tipos de apariciones del listado anterior se corresponden con la *emergencia fuerte*, y son los que explotaremos en este trabajo.

B. Metropolis

Metrópolis es un Simulador de Ciudades Auto-gestionadas [6][12]. Se basa en la construcción de una ciudad, mediante la toma de decisiones colectiva de sus habitantes sobre las edificaciones a construir en la misma, haciendo uso de un presupuesto comunitario limitado. La toma de decisiones de cada jugador se basa en sus personalidades. De acuerdo a las decisiones tomadas colectivamente, tanto los índices de felicidad de la ciudad, como de cada jugador, se verán afectados.

En la literatura no existen juegos basado en la idea de Ciudades Auto-gestionadas del tipo propuesto en este trabajo. Los juegos anteriormente nombrados (como SimCity) son juegos enfocados en la creación y gestión de una ciudad, comportándose el jugador como un alcalde; a diferencia de Metrópolis, donde un grupo de jugadores (que representan los ciudadanos) gestionan autónoma y colectivamente la ciudad.

1) Filosofía del Juego: El juego busca beneficiar a cada jugador. Pero inspirado en el dilema del prisionero [13], la cooperación es crucial en el juego; permite el equilibrio entre los deseos de todos, lo cual lleva a la felicidad colectiva. Las ciudades son castigadas cuando las personas no cooperan, porque la felicidad de la ciudad será peor. Esto es así, ya que la felicidad de la ciudad es el promedio de la de sus habitantes.

El elemento fundamental del juego es la toma de decisiones colectiva, la cual se realiza en el consejo que gobierna la ciudad. En el consejo, cada jugador tiene la oportunidad de votar a favor o en contra de ciertos tipos de construcciones de edificios, utilizando el presupuesto de la ciudad. El consejo toma decisiones basándose en lo que cada jugador vota. La decisión será determinada por el voto de la mayoría. En el juego hay dos tipos de habitantes, uno en representación de cada jugador, otros generados al azar. Cuando alguien quiere construir en la ciudad, el proyecto se somete al voto popular en el consejo. Cada habitante tiene su propia personalidad, la cual es usada para determinar los tipos de construcción de más interés para él, tanto para construir como para votar en el consejo. La ciudad se va desarrollando a través de las decisiones colectivas sobre construir o no construir.

El juego tiene un sistema de puntuación, que se calcula cada año. La puntuación se basa en la felicidad de su población (la felicidad de todos los agentes afecta el índice de felicidad de la ciudad). Por otra parte, la felicidad de los jugadores depende de las construcciones realizadas en la ciudad, y su relación con sus personalidades.

Cada ciudad cuenta con 10 habitantes, en los que al menos 5 de ellos son artificiales, con el fin de simular la parte de los ciudadanos que no pueden ser controlados por los jugadores. Estos habitantes son generados al azar, y toman decisiones de la misma manera que los otros habitantes. Los habitantes no-artificiales pueden tener personalidades creadas al azar, o predefinidas por los jugadores. Los jugadores pueden pedir construir el edificio que ellos quieran, en sus zonas preferidas. Las edificaciones posibles en una ciudad son en las siguientes áreas: salud, educación, medio ambiente, comercio, industria y tecnología.

2) Caracterización de los Edificios: Existen dos tipos de edificios, los edificios primarios pueden añadir o quitar puntos a los índices de felicidad de cada área de la ciudad (más adelante se definen), y los edificios secundarios añaden/quitan un porcentaje del valor de dichos índices. Una edificación primaria es la primera que se construye en un sitio dado, y las secundarias son las que se colocan a su alrededor. De esta manera, cada tipo de construcción tendrá un cierto efecto en el juego. La Tabla I muestra los detalles de cada tipo de construcción: el radio de cobertura de cada edificación, el costo de la construcción, su grupo, y la bonificación o penalización que aporta a los índices de felicidad de cada área de la ciudad.

Como se ha indicado anteriormente, cada edificio tiene un radio de acción (zona de cobertura), que por simplicidad es un diamante, cuya diagonal es dos veces el número dado en la Tabla I. Mientras dos edificios del mismo tipo (primarios/ secundarios) no se intercepten, las edificaciones proveerán su puntuación total. Cuando hay una intercepción, la bonificación positiva disminuye un 50%, mientras que la negativa se mantendrá sin cambios.

Tabla I: Radios, Bonificaciones y Penalizaciones, para Cada Tipo de Construcción

Número	Tipo	Grupo	Radio	Bono	Penalización	Costo
0	Vacío	-	-	-	-	-
1	Árbol	-	-	+10Ambiente.	-	20
2	Casa	-	-	-	-	500
3	Calle	-	-	-	-	40
4	Institución de Salud	Primaria	7	+250 Salud	-	7000
5	Institución Educativa	Primaria	4	+150Educación	-	4000
6	Institución Ambiental	Primaria	7	+100 Ambiente	-	4000
7	Institución Comercial	Primaria	2	+50Comercio	-	5000
8	Industria	Primaria	7	+200Industria	-75 Ambiente	7000
9	Institución Tecnológica	Primaria	3	+170Tecnología	-	3000
10	Institución de Salud	Secundaria	2	+15%Salud	-	1000
11	Institución Educativa	Secundaria	2	+10%Educación	-	800
12	Institución Ambiental	Secundaria	2	+12%Ambiente	-	800
13	Institución Comercial	Secundaria	1	+7%Comercio	-	600
14	Industria	Secundaria	3	+18%Industria	-8% Ambiente	1000
15	Institución Tecnológica	Secundaria	2	+10%Tecnología	-	800

Por otro lado, cuando dos tipos diferentes de edificaciones están próximas, ellas reciben sanciones o bonificaciones por dicha proximidad (ver Tabla II). Por ejemplo, si un hospital está próximo a una industria se le daría una penalización de -6 puntos en ambos edificios porque están cerca. Si construimos un centro de investigación (institución tecnológica) cerca de una escuela o universidad (instituciones educativas), se le daría una bonificación de 3 puntos a cada uno. Para determinar la proximidad se considerada la zona de cobertura para cada tipo de edificio (véase la Tabla I).

Tabla II: Bonificación y Sanción en Función de la Proximidad de los Tipos de Construcción

	P Salud	P Ed	P Am.	P Com.	P Ind.	P Tec.	S Sal	S Ed	S Am	S Com.	S Ind.	S. Tec.
P Salud	-4			-2	-6		+3				-4	
P Ed					-4	+3	+3	+3			-3	+3
P Am.					-3	+2			+3		-2	
P Co.	-2				-3					+3	-2	
P Ind.	-6	-4	-3	-3		-3	-5	-3	-3	-3	+3	-3
P Tec.	-2	+3	+2		-3			+3	+2		-2	+3
S Sal	+3	+3			-5			+3			-3	
S Ed		+3			-3	+3	+3		+2		-3	+3
S Am			+3		-3	+2		+2			-2	+2
S Co.				+3	-3						-2	
S Ind.	-4	-3	-2	-2	+3	-2	-3	-3	-2	-2		
S. Tec.		+3			-2	+3		+3	+2			

3) Personalidades de los Agentes: A principios del juego, los jugadores deben elegir cuatro tipos de personalidades, mientras que a los habitantes artificiales se les puede elegir al azar, o seleccionar (tanto las personalidades elegidas por los usuarios, como las seleccionadas a los habitantes artificiales, son las personalidades activas de cada uno). La personalidad de los jugadores se utiliza para calcular su felicidad, y en los habitantes artificiales es usada para establecer sus comportamientos durante el juego (por ejemplo, un habitante artificial ambientalista siempre vota en contra de las propuestas que incluyen daños al ambiente).

Así, las personalidades de un jugador determinan sus necesidades básicas (felicidad). Las personalidades activas de un jugador aportan entre 1250 y 750 puntos al índice de felicidad en una área dada, mientras que las que no están activas permanecen en un nivel normal (1000 puntos, ver Tabla III).

Tabla III: Tipo de Personalidad e Índices de Felicidad

Personalidad	Salud	Educación	Ambiente.	Comercio	Industria	Tecnología
Hipocondr.	1250	1000	1000	1000	1000	1000
Con salud	750	1000	1000	1000	1000	1000
Con dificult.	1000	1250	1000	1000	1000	1000
Auto-aprende	1000	750	1000	1000	1000	1000
Indiferent	1000	1000	1250	1000	1000	1000
Ambiental.	1000	1000	750	1000	1000	1000
Ahorrador	1000	1000	1000	750	1000	1000
Comprador	1000	1000	1000	1250	1000	1000
Industrial	1000	1000	1000	1000	1250	1000
Contra Indust	1000	1000	1000	1000	750	1000
Anticuado	1000	1000	1000	1000	1000	750
Pro-tecnol	1000	1000	1000	1000	1000	1250

4) Índice de Felicidad de una Ciudad: El índice de felicidad del área j de la ciudad se calcula como:

$$PT_j = \sum_l \sum_i (PC_{ij}^l + \sum_s CER_{si}) + \sum_m \sum_k (PC_{kj}^m + \sum_s CER_{sk}) \quad (1)$$

Donde l representa los edificios principales y m los edificios secundarios de la ciudad, respectivamente, PC_{ij} es la puntuación total que da el edificio primario i al índice de felicidad del área j (ver Tabla I), tomando para ello en cuenta las intercepciones con otras edificaciones, y CER_{ik} son las bonificaciones o penalizaciones en función de la cercanía de los edificios i y k (ver Tabla II). La ecuación 1 determina cual es la afinidad de las edificaciones cercanas según la Tabla I. Un área "feliz", es aquella donde las edificaciones en ella tienen cierta relación (por ejemplo, instituciones educativas y tecnológicas).

El cálculo del índice de felicidad para cada jugador p es:

$$VI_p = \sum_j \frac{1000 * PT_j}{TRJ_{pj}} \quad (2)$$

Donde, TRJ_{pj} es el total requerido por el jugador p para el índice j según su personalidad (ver Tabla III). En este caso, el índice de felicidad del jugador tiene que ver con la relación que existe entre sus personalidades activas y las edificaciones en la ciudad. Por ejemplo, si una de las personalidades de un jugador es ecológica, la existencia de instituciones ambientales aumentarán su índice de felicidad, y la existencia de instituciones industriales lo disminuirán.

Luego de obtener los índices de cada agente, se promedian todos, para generar los índices generales de felicidad de la ciudad.

La Figura 1 muestra una de las pantallas de Metrópolis, en la que se puede observar la cantidad de dinero en el presupuesto actualmente (ver flecha 1), la edificación selecciona y su costo (ver flecha 2), las coordenadas X, Y del mapa (ver flechas 3 y

4), los índice de felicidad en cada área (salud, educación, medio ambiente, comercio, tecnología e industria, ver flecha 5), las coordenadas en el mapa (ver flecha 6), el botón para configurar el juego al inicio (ver flecha 7), el mapa de la ciudad (ver flecha 8), y la leyenda con los iconos para cada tipo de construcción (ver flecha 9).

Además, la Figura 2 muestra un ejemplo de cómo la ciudad se ha ido organizando, en función de las decisiones de sus habitantes (jugadores).

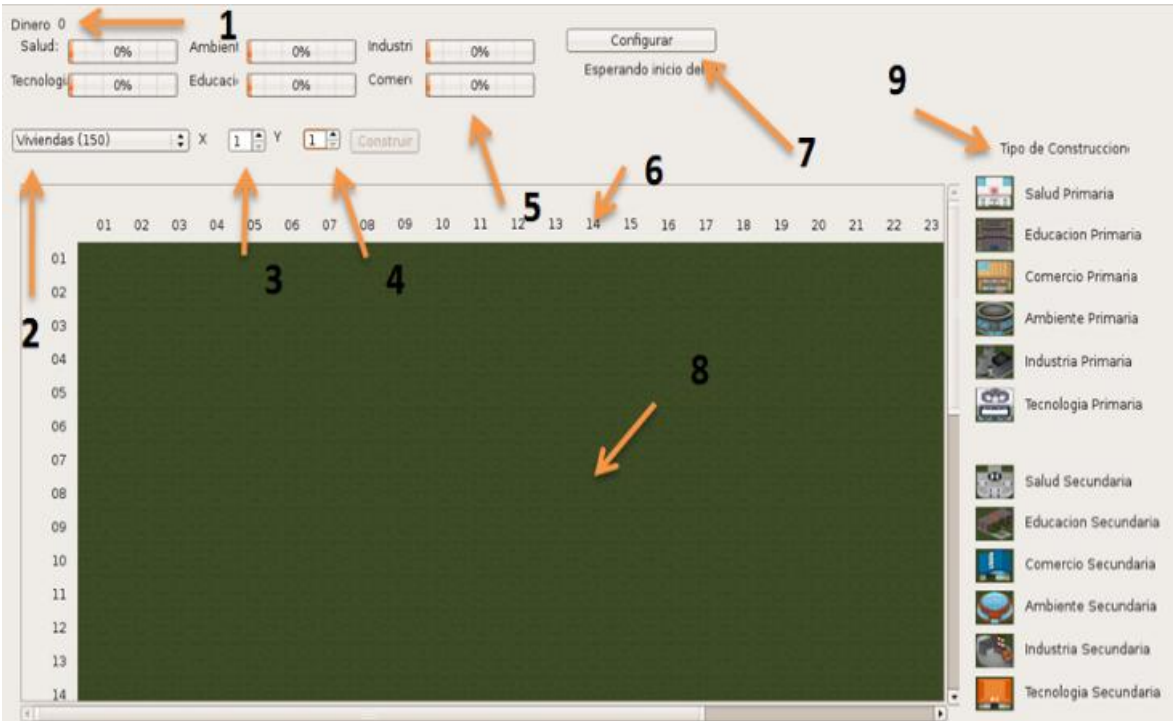


Figura 1: Pantalla Principal de Metrópolis [9]

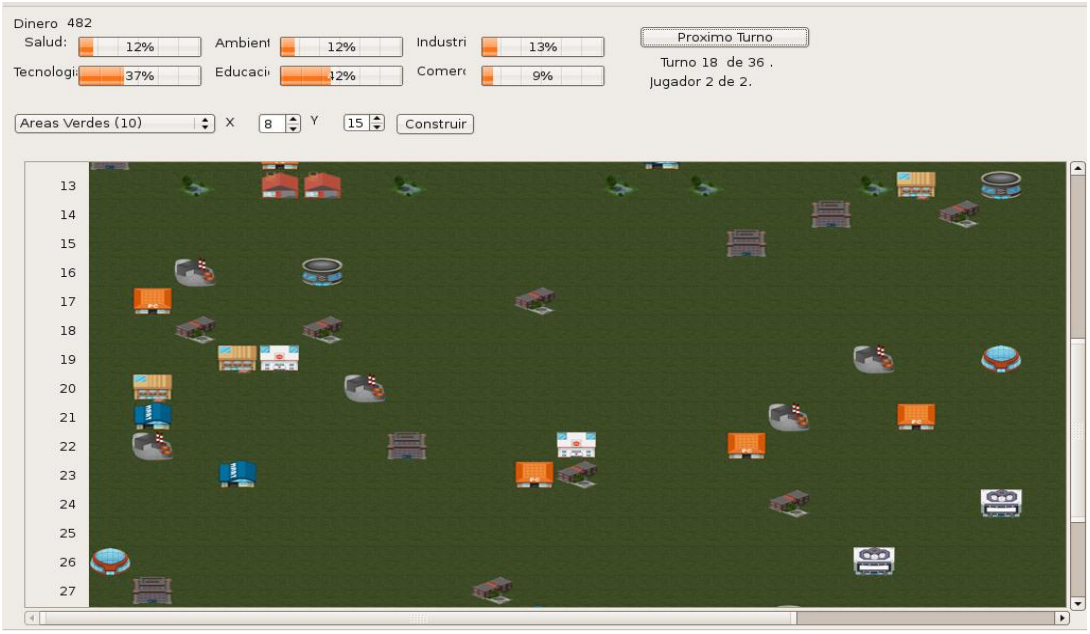


Figura 2: Ejemplo de una Ciudad en Metrópolis [9]

III. Análisis de la Emergencia Fuerte para Metropolis

A. *Caracterización de la Emergencia Fuerte en Metropolis*

Para evaluar dónde introducir mecanismos que posibiliten comportamientos emergentes en Metrópolis, se deben considerar los siguientes aspectos:

- Determinar aquellos elementos que repercuten en los procesos de toma de decisión en el juego que se puedan parametrizar.
- Determinar cómo desde la interacción de los jugadores de Metrópolis, se pueden establecer mecanismos de aprendizaje de esos parámetros.

En el caso de Metrópolis, los procesos de toma de decisión determinan los índices de felicidad. Ellos son afectados por los siguientes elementos:

- Las zonas de cobertura (ver Tabla I)
- Las bonificaciones/penalizaciones que aporta cada tipo de edificación (ver Tabla I)
- Las penalizaciones por las proximidades de las edificaciones de diferentes tipos (ver Tabla II)
- Las penalizaciones por las intercepciones de las edificaciones del mismo tipo (ver Tabla I)
- Los costos de cada tipo de construcción (ver Tabla I)
- Los puntos que aportan las personalidades a los índices de felicidad (ver Tabla III)

Esos son potenciales parámetros, que se pueden adecuar en el juego (que se pueden aprender), para lo cual se requieren establecer los mecanismos necesarios para que el juego los vaya modificando internamente por procesos de aprendizaje, cuando juegan los usuarios.

En este trabajo vamos a analizar el comportamiento de uno de ellos en la emergencia en el juego, el efecto por aprender el radio de cobertura de cada edificación. Este parámetro es fundamental, porque posibilita dos aspectos:

- La aparición de patrones urbanísticos, ya que determina los posibles agrupamientos que se dan por tipo de edificaciones (ver Tablas I y II).
- La adecuación de la ciudad a los ciudadanos, tal que sus índices de felicidad se logren mejorar.

En la siguiente subsección se analiza con más detalles este último aspecto, ya que inspirado en esa mejora, es que se propone el mecanismo de aprendizaje del radio de cobertura.

B. *Ejemplo de Mecanismo Emergente Fuerte en Metropolis*

En Metrópolis, el radio que cubre una edificación determina cuando tiene solapamiento con otras edificaciones. El radio indica el entorno cercano de influencia de una edificación en la ciudad. Metrópolis usa en cada iteración, los valores de radio para penalizar o bonificar cada vez que exista un solapamiento entre construcciones (sin importar si son primarias o secundarias).

El valor de radio, tanto para las construcciones primarias como secundarias, se puede observar en la Tabla I. Además, hay construcciones que no se les asigna valor del radio, tales como las zonas residenciales y calles, ya que éstas solo afectan al sitio de la ciudad donde están. Es decir, en estos casos sus construcciones no tienen ningún impacto en sus vecindades.

El radio es un parámetro que se podría adecuar a los usuarios. Su influencia en el índice de felicidad es importante, ya que determina las penalizaciones, tanto de proximidad como de intercepción (ver ecuación 1). En ese sentido, establecer valores de radio que maximicen los índices de felicidad de los jugadores sería el objetivo a alcanzar. Por ejemplo, un valor de radio que minimice las penalizaciones entre las edificaciones importantes para las personalidades e intereses de los jugadores, sería el objetivo del proceso de aprendizaje.

En este trabajo se modificarán los valores de radio, usando como radio inicial por defecto el mismo de la Tabla I, considerando un valor adicional, denominado rX, que podrá ser negativo o positivo. En particular, consideraremos:

- Variables adicionales por tipo de área para el ajuste de sus radios (salud, educación, ambiente, comercio, industria, tecnología): rS, rE, rA, rC, rI, rT.
- Además, se calculará el valor promedio de felicidad por área de la ciudad, considerando a todos los habitantes de la ciudad.

En función de esas variables, se actualizan los radios de cada edificación, usando los siguientes criterios (ver Figura 3):

- Si el valor del índice se encuentra por encima del promedio en un rango inferior al 10%, se deja igual (rX=0).
- Si el valor del índice se encuentra por encima del promedio, entre el 10% y el 90%, se decrementa uno (rX=-1).
- Si el valor del índice se encuentra por encima del promedio, en un rango superior al 90%, se decrementa dos (rX=-2).

Para el caso contrario,

- Si el valor del índice se encuentra por debajo del promedio en un rango inferior al 10%, se deja igual.
- Si el valor del índice se encuentra por debajo del promedio entre el 10% y el 90%, se incrementa uno (rX=+1).
- Si el valor del índice se encuentra por debajo del promedio en un rango superior al 90%, se incrementa dos (rX=+2).

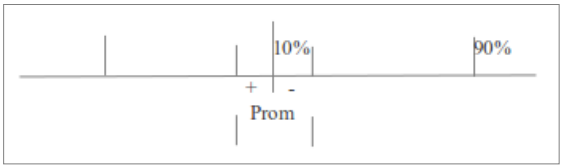


Figura 3: Promedio y Rangos de Cercanía

Las reglas antes descritas, se implementan en un macro-algoritmo que aprende a modificar los radios de cada tipo de edificación, haciendo al juego auto-modificable. Ese macro-

algoritmo es el siguiente, para el caso Salud (es similar para el resto de áreas):

```
if (salud >= fprom)  {
     If (salud-fprom < fprom*0.10)
        rS=rS;
     else if (salud-fprom >= fprom*0.10 && salud-fprom < fprom*0.90)
        rS--;
     else if (salud-fprom >= fprom*0.90)
        rS-=2;
  }
else if (salud < fprom)  {
     if (fprom-salud < fprom*0.10)
        rS=rS;
     else if (fprom-salud >= fprom*0.10 && fprom-salud <f prom*0.90)
        rS++;
     else if (fprom-salud >= fprom*0.90)
        rS+=2;
  }
```

IV. EXPERIMENTOS

Se realizó una serie de pruebas, garantizando en cada una de ellas que las decisiones tomadas por los jugadores fuesen las mismas, tanto al jugar Metrópolis sin cambio de radio como Metrópolis auto-modificable.

A. Comparación General de Metropolis con Modificación y sin Modificación de Radios

En esta prueba se definen los habitantes de la ciudad aleatoriamente, tanto para el caso de Metrópolis sin cambio de radio y Metrópolis auto-modificable. Los resultados obtenidos de los índices de felicidad por área se observan en la Tabla IV para Metrópolis auto-modificable, y en la Tabla V para Metrópolis sin modificar sus radios, para los años 3, 4 y 5. Se muestran para 5 años, porque es el periodo de tiempo suficiente habitualmente, para empezar a ver comportamientos emergentes en la ciudad.

Al hacer un análisis de los resultados, se observa que con Metrópolis adaptativo se obtiene:

- Que el valor de los promedios finales de los índices de felicidad por área son siempre superiores.
- Que el valor de los promedios generales del índice de felicidad de la ciudad por iteración (año) son siempre superiores.

Tabla IV: Metrópolis con Radios Modificados por Área

Área vs Iteración	3	4	5	Promedio
Salud	24	100	85	69,67
Educación	62	99	100	87
Ambiente	49	57	100	68,67
Tecnología	21	99	37	52,33
Industria	56	54	65	58,33
Comercio	46	18	77	47,33
Promedio	43	71,33	77,33	

Tabla V: Metrópolis sin Radios Modificados por Área

Área vs Iteración	3	4	5	Promedio
Salud	13	34	78	41,67
Educación	35	36	83	51,33
Ambiente	31	37	95	54,33
Tecnología	37	39	52	42,67
Industria	37	55	37	43
Comercio	19	30	40	29,67
Promedio	28,67	38,5	64,17	

B. Comparación de Metropolis con y sin Modificación de Radios, para una Específica Personalidad

En esta prueba se define un perfil de usuario hipocondriaco para todos los agentes (habitantes de la ciudad), y se ejecutan las dos versiones de Metrópolis (con modificación y sin modificación de radio) por 4 años. Los resultados obtenidos se observan en la Tabla VI, los cuales indican que:

- Metrópolis adaptativo alcanza un mejor valor en cada uno de los índices de felicidad por área.
- En particular, en el área de salud, que es fundamental para esa personalidad, se alcanza una mejora notable. Además, en otras áreas también se alcanzan mejoras importantes, como por ejemplo, en educación.

Tabla VI: Resultados con Modificación y sin Modificación de Radio, para la Personalidad Hipocondriaca

Área vs Iteración	Con Modificación	Sin Modificación
Salud	83	69
Educación	60	34
Ambiente	38	23
Tecnología	56	48
Industria	53	47
Comercio	61	27
Promedio	58,5	41,33

C. Comparación de Metropolis con y sin Modificación de Radios, para Habitantes con Personalidades Diversas

En esta prueba suponemos que los agentes tienen personalidades diversas, pero definidas por nosotros. En este caso, suponemos las siguientes personalidades, distribuidas entre los habitantes de la ciudad: hipocondriaca, autodidacta, ahorrador y pro-tecnología. Los resultados se muestran en la Tabla VII, determinándose los siguientes aspectos:

- Los resultados son mejores cuando el radio se auto-modifica.
- Aunque las personalidades de los habitantes son diversas, Metrópolis adaptativo logra satisfacer todos los intereses para el grupo de agentes presentes (áreas de tecnología, salud, comercio, educación).
- Un caso interesante es en la área de comercio, donde Metrópolis adaptativo logra reducir ese índice, por ser de interés hacerlo por una de las personalidades presentes (ahorrador).
- También, es importante observar la notables mejorarías que realiza Metrópolis adaptativo en las áreas de educación y tecnología.

Tabla VII: Resultados con y sin Modificaciones de Radio para la Personalidades Hipocondriaca, Autodidacta, Ahorrador y Pro-Tecnología

Área vs Iteración	Con Modificación	Sin Modificación
Salud	74	42

Educación	96	54
Ambiente	59	26
Tecnología	100	42
Industria	42	38
Comercio	20	35
Promedio	65,17	39,5

V. CONCLUSIONES

En este trabajo se ha propuesto una modificación a Metrópolis, para conferirle características de emergencia fuerte. En específico, se le permite aprender cual debe ser el radio de cobertura de las edificaciones. Esto lo va haciendo en el tiempo, sin perder sus capacidades para hacer emerger patrones (emergencia débil).

Particularmente, para mostrar la importancia de cambiar el radio, en vez de dejarlo constante, como la primera versión del juego, se realizaron las mismas pruebas. Basada en los resultados obtenidos, vemos como el aprendizaje del valor del radio logra una mejor adecuación de los índices de felicidad de la ciudad, derivado de una emergencia de patrones urbanístico que siguen los intereses de sus ciudadanos.

En específico, la versión adaptativa permite alcanzar niveles de felicidad más adecuado a las personalidades de los agentes, esto se debe a que la modificación del radio de alcance que se realiza adapta el juego a las necesidades de los agentes. Modificar el radio es modificar el impacto que tiene un edificio sobre sus zonas adyacentes, y por lo tanto, las necesidades que cubre. Esta primera experiencia abre un abanico de posibilidades alrededor del proceso de emergencia fuerte en Metrópolis, modificando otras variables que inciden en la dinámica del juego.

En Metrópolis adaptativo la ciudad se transforma en un sistema dinámico, cuya evolución depende de las emergencias fuertes que se producen en ella producto de las decisiones colectivas de los agentes, y de las auto-modificaciones que él va realizando en sus parámetros, que producen fuerzas ascendentes que afectan directamente su estructura general. En Metrópolis adaptativo se acelera el comportamiento emergente. En específico, los comportamientos emergentes que se aceleran en la nueva versión de Metrópolis, basados en la emergencia fuerte, son:

- Patrones urbanos en la ciudad, con edificios similares en ciertas zonas de ella (edificios atraen a otros edificios similares, que se construirán cerca de ellos).
- Patrón de felicidad en la ciudad, que sigue las personalidades de los agentes que alberga (combina de alguna manera, las necesidades de sus ciudadanos).

Metrópolis adaptativo acelera dichas emergencias, porque logra una adecuación más rápida de los índices de felicidad de sus ciudadanos, por el proceso adaptativo incorporado por la emergencia fuerte.

Por otro lado, en la Sección III se han identificado otros elementos que pueden coadyuvar a la emergencia fuerte, como las bonificaciones/penalizaciones que aportan cada tipo de edificación, o las penalizaciones por las proximidades de las edificaciones de diferentes tipos, entre otras. Cada una de ellas coadyuva de una manera diferente a propiedades emergentes en Metrópolis, ya sea para definir patrones urbanísticos, o hacer emerger un comportamiento global de la ciudad que siga las personalidades de sus ciudadanos. En ese sentido, próximos juegos deberán evaluar la sensibilidad de esas métricas, para facilitar esos procesos emergentes en Metrópolis.

Podemos ver que hasta ahora se han estudiado sobre Metrópolis dos tipos de emergencia, una emergencia débil, desarrollada en la primera versión [2], para hacer emerger patrones urbanísticos y comportamiento general de la ciudad, parecido a la de sus ciudadanos. También, una emergencia fuerte, derivada por la propia auto-modificación del juego según como van evolucionando los índices de felicidad de los jugadores (que es el enfoque planteado en este trabajo). Se deberán explorar otras formas de emergencia fuerte, derivadas por la aparición/surgimiento de nuevos comportamientos o temáticas en el juego, o por la aparición de nuevas propiedades en los objetos que lo componen (por ejemplo, nuevas personalidades o tipos de edificaciones). Al respecto, se deberán hacer dos tipos de trabajos:

- De adaptación a Metrópolis para posibilitar esa emergencia fuerte.
- De análisis de los patrones que ahora surjan como resultados finales del juego, derivado de esos nuevos comportamientos emergentes.

AGRADECIMIENTOS

El Dr. Aguilar ha sido parcialmente financiado por el Proyecto Prometeo del Ministerio de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) de la República de Ecuador.

REFERENCIAS

[1] A. Rollings and E. Adams. *Fundamentals of Game Design*. Prentice Hall, 2006.

[2] J. Juul, *The Open and the Closed: Games of Emergence and Games of Progression*, http://www.digra.org/wp-content/uploads/digital-library/05164.10096.pdf.

[3] J. Dormans, *Simulating Mechanics to Study Emergence in Games*. *in proceedings of the Seventh Artificial Intelligence and Interactive Digital Entertainment Conference*, 2009.

[4] J. Aguilar, *Introducción a los Sistemas Emergentes*. Talleres Gráficos, Universidad de Los Andes, 2014.

[5] N. Perozo, J. Aguilar, O. Terán, and H. Molina, *Self-organization and Emergence Phenomena in Wikipedia and Free Software Development using MASOES*, Publicaciones en Ciencias y Tecnología, vol. 7, pp. 51-72, 2013.

[6] J. Aguilar, J. Cardozo, C. González, and B. Rengifo. *Una Aproximación a los Juegos Emergentes. Metropolis, Simulador de Ciudades Autogestionadas*. in proceedings of the XXXXVII Conferencia Latinoamericana de Informática (CLEI 2011), Quito, Ecuador, October 2011.

[7] P. Sweetser. *Emergence in Games*. *Charles River Media*, 2007.

[8] P. Sweetser and J. Wiles. *Scripting versus Emergence: Issues for Game Developers and Players in Game Environment Design*. International Journal of Intelligent Games and Simulations, vol. 4, pp. 1-9, 2005.

[9] P. Adams. *Teaching and Learning with Simcity 2000*. Journal of Geography, vol. 97, pp. 47-55, 1998.

[10] A. Ferré. *SimCity*, in Verb, Architecture Boogazine: Connection. Actar, 2004.

[11] *Lincity - A City Simulation Game*. http://lincity.sourceforge.net.

[12] *Video de Metropolis*, http://www.ing.ula.ve/~aguilar/desarrollo-software/desarrollo-software.htm.

[13] K. Binmore, *La Teoría de Juegos: una Breve Introducción*, Alianza Editorial, Madrid, 2011.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 47-56, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Arquitectura para la Gestión de Datos Imperfectos en la Era de Big Data

Kity Álvarez[1], Betzaida Romero[1], José Tomás Cadenas[1], David Coronado[1], Rosseline Rodríguez[1]
kjalvarez@usb.ve, betzaidaromero@usb.ve, jtcadenas@usb.ve, dcoronado@usb.ve, crodrig@usb.ve

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

Resumen: La gran cantidad de datos que se maneja hoy en día y la manera como estos datos se extraen del contexto obligan a utilizar sistemas de información capaces de tomar decisiones, adecuarlos a la empresa y al usuario final. La digitalización da lugar a distintos tipos de datos en tiempo real de acuerdo al escenario que se plantee. Una gran cantidad de estos datos, en ocasiones se presentan de forma no normalizada: datos en *streaming*, geoespaciales o generados por diferentes tipos de sensores, que no encajan bien en un esquema relacional, tradicional o estructurado. Mucha de esta información puede ser vaga, imprecisa, ambigua o incompleta, muy parecida al lenguaje humano con respecto a términos cualitativos y cuantitativos. El objetivo del presente trabajo es estudiar acerca de la gestión de datos imperfectos sensibles al contexto integrado a la utilización de tecnologías asociadas a *Big Data*. La teoría de conjuntos difusos, en el marco de *Soft Computing*, aporta mecanismos para modelar y representar datos posibilísticos en bases de datos difusas, se propone una arquitectura que gestione este tipo de datos integrada con tecnologías para el manejo de datos masivos. Esta arquitectura incorpora un Módulo de Interacción con una Base de Datos Difusa que permita el almacenamiento y recuperación de datos sensibles al contexto. El propósito es proporcionar una herramienta útil para enfrentar el reto que tienen las organizaciones de obtener un mayor provecho de la gran cantidad de información proporcionada por las tecnologías del mundo actual. Además, se espera obtener los beneficios que agrega la gestión de datos difusos sensibles al contexto en el almacenamiento de datos imperfectos y consultas flexibles que no pueden ser ofrecidas en sistemas de bases de datos tradicionales.

Palabras Clave: Datos Imperfectos; Bases de Datos Difusas; Bases de Datos Sensibles al Contexto; Soft Computing; Big Data.

Abstract: The large amount of data that is handled today and the way these data are extracted from the context require the use of information systems capable of making decisions, adapting them to the company and the end user. Digitization gives rise to different types of data in real time according to the scenario that arises. A large amount of this data, sometimes, is presented in a non-normalized way: data streaming, geospatial or generated data by different types of sensors that do not fit well into a relational, traditional or structured scheme. Much of this information may be vague, imprecise, ambiguous or incomplete, very similar to human language with respect to qualitative and quantitative terms. The objective of the present work is to study of context-aware imperfect data management integrated to the use of technologies associated with Big Data. The fuzzy sets theory, in the framework of Soft Computing, provides mechanisms to model and represent possibilistic data in fuzzy databases. We propose an architecture that manages this data type, integrated with technologies for big data management. This architecture incorporates an Interactive Module with a Fuzzy Database that allows the storage and retrieval of context-aware data. The purpose is to provide a useful tool to face the challenge that has the organizations to make the most of the vast amount of information provided by today's technologies. Also, we expect to obtain the added benefits of context-aware fuzzy data management in storage imperfect data and flexible queries that cannot be offered in traditional database systems.

Keywords: Imperfect Data; Fuzzy Databases; Context-Aware Databases; Soft Computing; Big Data.

I. Introducción

En la actualidad, el tratamiento de los datos ha ido cambiando debido a que el alcance de las aplicaciones va en crecimiento y los retos empresariales al igual que la cantidad de datos disponibles son cada vez mayores. Estos datos provienen de diferentes fuentes y son de naturaleza distinta.

Debido a los avances en las Tecnologías de la Información y Comunicación (TIC), la gran cantidad de información disponible, denominada Big Data [1], constituye una oportunidad sin precedentes, sin embargo, los usuarios de esta información corren el riesgo de verse abrumados, perdiendo la oportunidad de utilizar el valioso conocimiento que puede ser extraído de estos datos. Es así como, las aplicaciones que usan

estas bases de datos, requieren de una gestión inteligente de la información para satisfacer las demandas de almacenamiento y consultas. Esto representa un gran reto para los sistemas informáticos modernos pues los gestores de bases de datos tradicionales no son capaces de satisfacer las necesidades actuales de información de los usuarios.

Con la evolución de los datos masivos (Big Data) [2], las empresas de todo el mundo están descubriendo nuevas formas de competir y ganar. Esto obliga a una comprensión exhaustiva de los mercados, los clientes, los productos, las normativas, los competidores, los proveedores, los empleados, entre otros. Esta comprensión exige un uso eficaz y analítico de la información, que debe ir de la mano de las tecnologías y herramientas adecuadas para el procesamiento del mismo.

Las aplicaciones en el área de Big Data, requieren administrar un volumen de muchos Terabytes de datos cuyo tamaño, complejidad y diversidad, impiden que los Sistemas Gestores de Bases de Datos (SGBD) tradicionales puedan manejarlos eficientemente. Por eso han surgido las Bases de Datos bajo el paradigma NoSQL (No sólo SQL), que permiten resolver problemas de escalabilidad y rendimiento que presentan estos tamaños, así como su gran diversidad, usando nuevos entornos de datos distribuidos y escalables de forma horizontal [1].

Además, con el surgimiento de tecnologías asociadas a Internet de las Cosas (IoT) se puede obtener una gran cantidad datos de diversas fuentes, de forma rápida, confiable y segura. La integración de áreas como sistemas de información, inteligencia artificial, base de datos, sistemas expertos y extracción de conocimiento ha evidenciado la necesidad de gestionar datos imperfectos [3]. Este se considera un tema de investigación importante ya que los datos imperfectos están presentes en muchas aplicaciones del mundo real, la gestión de estos datos contribuyen a comprender y predecir mejor el comportamiento de los usuarios y mejorar sus experiencias [4].

En cuanto a la naturaleza de los datos, se consideran varios tipos de información imperfecta que debe ser incluida en una base de datos, clasificándola como imprecisa, incierta o vaga [5]. La gestión de datos imperfectos es importante en la exploración de información proveniente de las redes sociales y sus tendencias, de fuentes datos sindicados como las tarjetas de fidelidad (RFID), además de información relacionada a la toma de decisiones para la calidad de los servicios prestados a los usuarios.

Hoy en día se considera de vital importancia disponer de técnicas y herramientas de análisis de información que usen grandes repositorios de datos, así como representa un gran reto la gestión de datos imperfectos sensibles al contexto en los Sistemas de Bases de Datos. Por tal razón, el presente artículo se enfoca en integrar la gestión de datos imperfectos, con técnicas que ya han sido empleadas en Bases de Datos Difusas, que tomen en cuenta las nuevas tecnologías que han surgido en el área de Big Data.

La estructura del presente artículo es la siguiente: la sección II trata sobre la Gestión de Datos Imperfectos que constituye el marco teórico de este trabajo. La sección III muestra los retos que presenta el área del Big Data. La sección IV, presenta los antecedentes del trabajo. La sección V, plantea la propuesta de arquitectura para gestión de datos imperfectos sensibles al contexto integrada con tecnologías de datos masivos. Finalmente, en la sección VI, se exponen las conclusiones y trabajos futuros.

II. Gestión de Datos Imperfectos

En esta sección se presentan los fundamentos teóricos de la investigación realizada, que incluye la noción de datos imperfectos y las diversas formas como han sido tratados, la teoría de conjuntos difusos como una alternativa posible y el principio de información planteado por Zadeh.

A. Datos Imperfectos

Debido a la gran variedad de imperfecciones que pueden afectar a los datos, se puede enfocar el análisis de los mismos separando aquellos que tienen una apariencia flexible (*soft*) de aquellos que tienen especificaciones precisas y completamente ciertas. Estos últimos corresponden a la información almacenada en Bases de Datos tradicionales. En [5], se definen tres términos relacionados a la imperfección de los datos: imprecisión, incertidumbre y vaguedad.

La *imprecisión* denota la carencia de exactitud en la expresión de la información, ocurre cuando el dato es desconocido o el dominio del atributo es impreciso por su naturaleza, por ejemplo “la imagen es *vieja*”, “Luis es *joven*”, o “el paciente es *obeso*”. La *incertidumbre* revela una situación en donde no se está seguro acerca de la veracidad de la información, tal como “es *posible* que el carro tenga cinco metros de longitud”, “Luis *probablemente* vive en Maracay”, o “*Creo* que la placa del automóvil que produjo el accidente era DBW50S”.

La *vaguedad* sucede cuando la información es afectada por imprecisión, incertidumbre o ambos; pues a veces es difícil establecer los límites entre imprecisión e incertidumbre. Cuando se usa la expresión “el carro es *grande*” puede ser que la persona no conozca el tamaño del carro con certeza, aunque no lo exprese explícitamente. Algunos autores usan este término como sinónimo de imprecisión.

Es de hacer notar que la imperfección puede incluir otras deficiencias, tal como datos erróneos o inconsistentes debido a: falta de precisión de dispositivos sensores, datos que provienen de fuentes heterogéneas o información resultante de técnicas como la consolidación.

La imperfección en los datos está presente en una gran variedad de aplicaciones emergentes, tales como: Servicios basados en Localización, Sistemas de Información Geográficos (GIS), Redes de Sensores Inalámbricos, Flujo de Datos (*data streaming*), Bases de Datos de uso específico (espaciales, biológicas y biométricas), Extracción de Conocimiento (Minería de Datos, Conocimiento Experto y Ontologías), Inteligencia Ambiental, además de Sistemas de Recomendación y Recuperación de Información. En estas aplicaciones es importante gestionar la imperfección de los datos adecuadamente, a fin de facilitar la toma de decisiones de los usuarios y la dotación de servicios de alta calidad.

La gestión de datos imperfectos ha sido de interés debido a la gran cantidad de áreas donde se usan, tales como: minería de datos preservando privacidad, datos probabilísticos inferidos por métodos de predicción estadística, incertidumbre en bases de datos [1]. Debido a esto, se han diseñado e implementado diferentes propuestas de bases de datos que permitan la

representación y gestión de la imperfección de los datos a través de los Sistemas Gestores de Bases de Datos (SGBD).

Las Bases de Datos tradicionales sólo manejan datos y condiciones precisas que en muchas ocasiones no representan las necesidades reales de información de los usuarios. Los datos faltantes son normalmente manejados a través de un pseudovalor denominado *null*. El uso de este pseudovalor puede deberse a varias razones tales como: el dato es desconocido, es impreciso, incierto o puede que no aplique para en una determinada tupla, entre otras [3].

Una estrategia utilizada regularmente por los manejadores de Bases de Datos tradicionales es la llamada imputación, que consiste en cambiar los valores *null* en la Base de Datos por valores sustitutos elegidos con algún tipo de criterio establecido. En [6] se propone el uso de la técnica de Reglas de Asociación Difusa, para obtener un conjunto de reglas que permitan estimar valores nulos en función de los demás valores presentes en los registros y cuya aplicación se extiende a atributos con valores cuantitativos y no solo categóricos.

Según De Tré y Zadrozny [3] para que las organizaciones sean competitivas tienen que utilizar toda la información disponible incluyendo la imperfecta, la cual no debe ser descartada. Por lo tanto, almacenar de manera eficiente y consultar datos imperfectos, sin introducir errores o causar perdida de datos, es considerado actualmente, como uno de los principales retos para gestión de la información.

Una de las áreas que ofrece formalismos y técnicas matemáticas para enfrentar la gestión de datos imperfectos es *Soft Computing* [7]. Su principal objetivo es aprovechar la tolerancia a la imprecisión, incertidumbre y verdad parcial, con el fin de obtener soluciones computacionales que sean tratables, robustas y de bajo costo. Sus técnicas incluyen Lógica Difusa, Neuro-computación, Razonamiento Probabilístico, Computación Evolutiva, además de otras técnicas como Redes de Creencia, Sistemas Caóticos y Aprendizaje Automático (*Machine Learning*).

Dado que la información imperfecta aparece en dominios y situaciones reales, debido a errores instrumentales o la corrupción por ruido durante la recogida de datos, se puede dar lugar a información incompleta para atributos específicos [8]. Además, la extracción de información exacta puede ser excesivamente costosa, de allí la necesidad de aplicar técnicas de *Soft Computing*.

Motro [8] indica que la imprecisión en los valores de los datos toma diferentes formas. Puede que el valor real del dato pertenezca a un conjunto específico de valores, por lo que es llamado dato disyuntivo. Si el conjunto al cual pertenece es el dominio completo entonces es indisponible, desconocido o perdido. Si cada uno de los valores candidatos está acompañado por un número describiendo la probabilidad que el valor sea verdadero (y la suma de todos ellos es uno) entonces se denomina probabilístico. Ocasionalmente, la información disponible en la ausencia de datos precisos es un término descriptivo, estos datos son denominados difusos.

La incertidumbre debido a ambigüedad o inconsistencia en los datos, asociada con situaciones en la cual se debe escoger entre diversas alternativas precisas, puede ser modelada utilizando la teoría de probabilidad [9]. Este es el caso de la medición de datos que pueden ser obtenidos a través de sensores o como resultado de un proceso de consolidación de datos provenientes de diversas fuentes. Es por ello que se han desarrollado una serie de modelos probabilísticos que pueden ser revisados en profundidad en [10].

Por otro lado, la incertidumbre debida a la vaguedad, es modelada con herramientas como la Teoría de Conjuntos Difusos [11], en la cual se hace énfasis en el significado de los términos imprecisos dados por el ser humano en un contexto específico. Zadeh [12] propuso la Teoría de Posibilidad donde se definen distribuciones de posibilidad como conjuntos difusos que permiten hacer limitaciones flexibles sobre los valores que pueden asignarse a una variable. La importancia de esta teoría reside en el hecho de que la intrínseca borrosidad de los lenguajes naturales, como consecuencia lógica de las expresiones utilizadas es *posibilística* y no probabilística, representando cuantiosa información sobre la cual se basan las decisiones del ser humano en un contexto.

Además, Zadeh [13] afirma que una de las más importantes facetas del pensamiento humano es la habilidad de resumir información en etiquetas de conjuntos difusos (denominadas *etiquetas lingüísticas*), que proporcionan una relación aproximada con los datos originales. Los postulados de Zadeh [13] permiten modelar la percepción y los procesos de pensamiento humano.

Técnicas de computación flexible (*Soft Computing*) hacen que sea posible gestionar la información imperfecta sobre una parte modelada del mundo real y representarla directamente en una base de datos. Si se usa la teoría de conjuntos difusos o la teoría de posibilidad para modelar datos imperfectos en una base de datos, ésta se denomina *difusa*. Otros enfoques incluyen aquellos que se basan en la teoría de conjuntos aproximados (*Rough sets*) y en la teoría de probabilidad, por lo que las bases de datos resultantes son denominadas *aproximadas* y *probabilísticas*, respectivamente.

Las técnicas más importantes de computación flexible para el tratamiento de la información imperfecta, debido a la investigación previa que han realizado diversos autores en el área de bases de datos, están basadas en la Teoría de Conjuntos Difusos y la Teoría de Posibilidad [3]. Es por ello que en la próxima sección se introduce este tema.

B. Teoría de Conjuntos Difusos

La teoría de conjuntos difusos [11] proporciona un marco matemático y computacional formal para representar las nociones de naturaleza vaga o imprecisa. Los conjuntos difusos, extienden el concepto de conjunto clásico, por lo que se caracterizan por una función de pertenencia, sobre un universo, cuyo rango está en el intervalo real [0,1]. Cuanto más se acerca a 1 el grado de pertenencia de un elemento, el mismo está más posiblemente (o certeramente) incluido en el conjunto. Así, 0 es la medida de completa exclusión y 1 la de completa inclusión o pertenencia total. El *borde* se define como el conjunto formado por los elementos parcialmente incluidos, el *núcleo*, como el conjunto de los elementos completamente incluidos y el *soporte*, como el conjunto de los elementos que no están completamente excluidos.

Según Zadeh [7], el término de lógica difusa es utilizado en dos sentidos diferentes. En un sentido limitado es visto como

una extensión de lógica multivaluada con el propósito de servir para el razonamiento aproximado. En su sentido más amplio, es un sinónimo de la teoría de conjuntos difusos. Es importante reconocer que el término lógica difusa es usado predominantemente en su sentido más amplio, así cualquier campo puede ser *fuzzificado*, reemplazando valores de un conjunto preciso por valores de un conjunto difuso.

La lógica difusa es una técnica de inteligencia computacional que permite trabajar con la información con alto grado de imprecisión, en esto se diferencia de la lógica convencional que trabaja con información bien definida y precisa. La lógica difusa permite que haya valores flexibles como: *cerca/lejos*, *grande/pequeño*, *fuerte/débil*, entre otros.

En el conjunto de datos se debe tener en cuenta la imperfección de la información para poder extraer modelos más cercanos a la realidad de acuerdo al contexto. Sin embargo, a pesar de la gran cantidad de técnicas existentes para dicho proceso, es común observar que se siguen utilizando Bases de Datos tradicionales para información clásica, y muy pocas toman en cuentan los datos imperfectos para su posterior análisis.

Por otro lado, la Teoría Computacional de Percepciones (CTP por sus siglas en inglés) propuesta por Zadeh [7], proporciona una base teórica para modelar sistemas complejos que gestionen información imperfecta, convirtiéndose en una poderosa herramienta que permite formalizar procesos de naturaleza humana (tal como conversar, razonar y tomar decisiones). Mediante el uso de etiquetas lingüísticas se pueden representar valores imperfectos para ser almacenados en la base de datos o para hacer posible el uso de términos lingüísticos en las consultas que se asemejan más al lenguaje natural utilizado por los seres humanos, es decir, utilizar el enfoque de representar la información en forma lingüística (computación por palabras) basado en la percepción más que en forma de medidas basadas en números.

Además, es necesario resaltar la habilidad extraordinaria del cerebro humano para manipular percepciones con respecto a diversos aspectos tales como: distancia, tamaño, peso, color, velocidad, tiempo, dirección, fuerza, verdad, probabilidad y otras características de objetos físicos y mentales; jugando un papel crucial en el reconocimiento de patrones, ejecución de actividades y la toma de decisiones.

Un ejemplo del uso de la lógica difusa, en la toma de decisiones, es el escenario en el que existe un entrenador de básquet que debe seleccionar a los candidatos para su equipo [11]. En la solución clásica de este ejemplo, la altura debe ser mayor a 185 cm y debe haber encestado al menos 13 de 16 tiros a cesta.

En la Tabla I se muestran los resultados de la solución clásica, donde sólo se seleccionarían a los candidatos F e I. Sin embargo, se observa que el candidato E tuvo 16 aciertos. La solución que utiliza conjuntos difusos es diferente. Se definen términos lingüísticos para cada variable (estatura y aciertos) y se da una respuesta que usa lógica difusa. Por ejemplo, el candidato exitoso es el que reúne los criterios: estatura *alta* y encestador *bueno*. El resultado se muestra en la Tabla II.

Utilizando lógica difusa se puede lograr una selección con discriminación entre los candidatos (*ranking* valorado) como se observa en la Tabla III. Esto evita dejar fuera del equipo a un buen encestador que mide 183 cm, tal como haría un entrenador. En lugar de tomar una decisión limitada a sólo dos alternativas "se rechaza" o "se acepta" (como el 0 ó 1 de la lógica clásica), se toma en cuenta la gradualidad.

Tabla I: Candidatos para el Equipo de Básquet (Solución Clásica)

CANDIDATO	ESTATURA (CMS)	ACIERTOS (16 TIROS)	SOLUCIÓN (CLÁSICA)
A	167	12	0
B	169	6	0
C	175	15	0
D	179	12	0
E	183	16	0
F	186	13	1
G	187	12	0
H	190	10	0
I	200	13	1

Tabla II: Candidatos para el Equipo de Básquet (Solución Difusa)

CANDIDATO	ESTATURA (CMS)	ACIERTOS (16 TIROS)	LÓGICA DIFUSA
A	167	12	0
B	169	6	0
C	175	15	0,33
D	179	12	0,50
E	183	16	0,87
F	186	13	0,75
G	187	12	0,50
H	190	10	0
I	200	13	0,75

Tabla III: Candidatos para el Equipo de Básquet (*Ranking* Valorado)

CANDIDATO	ESTATURA (CMS)	ACIERTOS (16 TIROS)	LÓGICA CLÁSICA	LÓGICA DIFUSA
E	183	16	0	0,87
F	186	13	1	0,75
J	200	13	1	0,75
D	179	12	0	0,50
G	187	12	0	0,50
C	175	15	0	0,33
A	167	12	0	0
B	169	6	0	0
H	190	10	0	0

La Teoría de Conjuntos Difusos es más adecuada que la Lógica Clásica para representar el conocimiento humano, ya que permite que los fenómenos y observaciones tengan más de dos estados lógicos. La Lógica Difusa produce resultados exactos a partir de datos imprecisos, por lo cual es particularmente útil en aplicaciones electrónicas o computacionales.

Los conjuntos difusos permiten representar y manipular información imperfecta a nivel de atributos [14]. Para ello, se definen dominios difusos de diferentes naturalezas que permiten especificar los diversos atributos de una entidad que presentan imprecisión. También, se definen diferentes operadores difusos para cada uno de estos dominios, tal como la igualdad difusa (*Fuzzy Equal* o FEQ) para comparar datos difusos.

La imperfección inherente al valor de un atributo está dada por la naturaleza del dominio del atributo. Los valores imperfectos en el modelo pueden estar representados por una etiqueta lingüística que dispondrá de diferentes interpretaciones dependiendo de la semántica en el contexto del dominio donde se defina. En base a la cardinalidad se pueden distinguir dos tipos de dominios difusos: *atómicos* o *conjuntivos*.

Los dominios *atómicos* son aquellos donde el atributo puede tomar un único valor que puede o no tener representación semántica asociada. Si no tienen representación semántica se conocen como categóricos. Por ejemplo, cuando se habla de la calidad de un producto el dominio se puede componer de etiquetas lingüísticas tales como *buena*, *regular* y *mala*.

En caso de tener una representación semántica, pueden ser de dos tipos dependiendo del universo sobre el cual estén definidos: *discreto* o *continuo*. Por ejemplo, se puede definir un dominio atómico continuo para el atributo estatura, formado por las etiquetas lingüísticas *gigante*, *alto*, *mediano*, *bajo* y *enano*. Cada una de estas etiquetas es asociada a un conjunto difuso cuyo universo o conjunto soporte es continuo (los números reales).

Otro ejemplo, es el atributo dureza de un mineral, la cual puede ser medida en una escala discreta en el rango de 1 a 10, pero normalmente se usan etiquetas lingüísticas, tales como, *muy dura*, *dura*, *media*, *blanda* y *muy blanda*. En este tipo de dominio, las etiquetas lingüísticas se representan con conjuntos difusos por extensión que indican la asociación de éstas a la escala del 1 al 10, el cual es un universo discreto.

Los dominios difusos donde el atributo puede tener diversos valores representados por un conjunto difuso, se denominan *conjuntivos*. Por ejemplo, para indicar que un investigador domina tres idiomas, se puede representar con un conjunto difuso tal como: {1.0/español, 0.8/inglés, 0.5/italiano}. Aquí el valor que acompaña al idioma determina nivel de experticia que tiene el investigador (mientras más cercano al 1 es más alto, más cercano al cero es más bajo). El universo es el conjunto de idiomas posibles {inglés, francés, ruso, italiano, español} y el valor del atributo es la conjunción de los idiomas que conforman el conjunto difuso.

Debido a la importancia que tiene actualmente la gestión de datos imperfectos en la siguiente sección se estudia el Principio de la Información postulado por Zadeh [13].

C. El Principio de la Información

El Principio de la Información se basa en identificar información con restricción, donde una restricción es una limitación a los valores que una variable puede tomar. Este principio consta de tres partes, a saber:

1) La información es una restricción.

2) Hay tres tipos principales de información, que provienen de los tres tipos principales de restricciones:

- *Información posibilística*: La variable toma el valor de un conjunto difuso.
- *Información probabilística*: La variable toma un valor con cierta probabilidad.
- *Información bimodal*: combinación de información posibilística y probabilística.

3) La información posibilística y la información probabilística no son derivables (son ortogonales), linealmente independientes, en el sentido de que ninguna se puede deducir de la otra.

Zadeh [13] afirma que, a partir de observaciones empíricas, se puede concluir que la mayoría de la información que se maneja a diario es posibilística o bimodal. Además, reconoce que la mayoría de los sistemas existentes no están capacitados para trabajar con la información bimodal. La diferencia entre la información posibilística y la probabilística, es que la posibilidad es la respuesta a la pregunta "¿puede ocurrir?", mientras que la probabilidad responde a "¿con qué frecuencia?". Para que una información sea probable, primero debe ser posible, por lo tanto, se tiene información bimodal.

Además de ser posible, probable y bimodal, la información puede ser precisa o difusa de acuerdo con el tipo de restricción. Por ejemplo, la probabilidad de que la estatura del candidato A está entre 1,65 y 1,69 cm es 0,8, es información bimodal precisa (CBI por sus siglas en inglés *Crisp Bimodal Information*). Por otro lado, decir que es muy probable que el candidato A no sea alto, es información bimodal difusa (FBI, *Fuzzy Bimodal Information*).

A continuación se presenta un ejemplo que permite observar cómo representar la información bimodal. Se quiere formar un equipo pequeño de baloncesto (tres contra tres), con los jugadores de la Tabla I, de la sección anterior. El universo estaría dado por el conjunto U = {A, B, C, D, E, F, G, H, I}. Durante el torneo, después de cada partido, los jugadores pueden ser cambiados, tal que a cada jugador X se le asocian dos números μ_X que representa la posibilidad de que X sea seleccionado y p_X la probabilidad de que X sea escogido. Ambos valores están en el intervalo real [0,1]. Si T es el conjunto de los jugadores seleccionados, estos valores se representan con la notación μ_X = Posibilidad ($X \in T$) y p_X = Probabilidad ($X \in T$). Así, se puede representar la información bimodal para el conjunto T, con las siguientes distribuciones de posibilidad y probabilidad:

$$\text{Poss}(T) = \mu_A/\text{A} + \mu_B/\text{B} + \ldots + \mu_I/\text{I}$$

$$\text{Prob}(T) = p_A/\text{A} + p_B/\text{B} + \ldots + p_I/\text{I}$$

Donde el símbolo + funciona como un separador. En particular, Poss(T) coincide con la representación de un dominio difuso conjuntivo. La información bimodal del equipo estaría dada entonces por Poss(T) y Prob(T).

III. Retos que Presenta la Era de *Big Data*

El constante uso de las tecnologías ha traído consigo un crecimiento explosivo en la cantidad de datos, los cuales son generados a grandes velocidades y en distintos formatos. A partir de este aumento de información, se da la necesidad de

extraer de ella, patrones y/o conocimientos, de forma rápida y eficiente, para lo cual, los métodos tradicionales han tenido que evolucionar en búsqueda de rendimiento y escalabilidad. El gran contenido de valor que genera este tipo de información está permitiendo a las organizaciones una mejora en la toma de sus decisiones, lo que conlleva a la obtención de ventajas competitivas en los diferentes campos de acción.

En un principio, las tecnologías informáticas apoyaron las funciones operativas de compañías y organismos mediante sistemas transaccionales internos, siempre basados en datos perfectamente normalizados, dotados de un formato sistemático y común. Posteriormente, los datos procedentes de los procesos operativos, generalmente almacenados en bases de datos relacionales, se usaron para sustentar los procesos de toma de decisión, siendo la fuente más importante de los sistemas estratégicos que constituían la estructura de la empresa.

Más allá de estos datos críticos de formato fijo, hay un tesoro escondido en otros tipos de datos menos tradicionales o aparentemente menos susceptibles de ser tratados de un modo automatizado, pero con los que también cuentan las empresas y entidades públicas. Los datos no tradicionales proceden de fuentes tales como portales, canales de acceso y de relación con clientes, redes sociales, correo electrónico, fotografías, dispositivos, sensores, objetos, medidores electrónicos, posicionamiento geográfico, entre otros.

Big Data [15] es un término que se ha acuñado para referirse a la manipulación de gran cantidad de datos. El volumen masivo, variedad y velocidad que ahora toma la información hace imprescindible capturar, almacenar y analizar todo este complejo repositorio.

Big Data aplica para toda aquella información que no puede ser procesada o analizada utilizando procesos o herramientas tradicionales. Entre los beneficios que se obtienen cuando se asume un proyecto de gestión de grandes volúmenes de datos, está la toma de decisiones más asertivas. Así, entre otras cosas, se obtiene un mercadeo personalizado acorde con las características de la empresa, como es el caso de *Twitter*, *Facebook*, *Instagram* y *Amazon*, con grandes ventajas competitivas. En el ámbito de seguridad, el uso de *Big Data* permite descifrar vulnerabilidades computacionales existentes y protegerse a tiempo de cualquier eventualidad.

Big Data también está emparentado con el área de Minería de Datos, o en forma más general, Extracción de Conocimiento, donde se intenta descubrir patrones de comportamiento en grandes volúmenes de datos.

Dentro de los tipos de datos masivos se pueden mencionar los siguientes [15]:

1) Empresariales tradicionales que incluyen información de clientes proveniente de sistemas como Relaciones con el Cliente (CRM), Sistemas Empresariales (ERP), transacciones realizadas en la Web, inventarios de ventas, entre otros.

2) Detallados que son datos o hallazgos derivados de actividades como campañas de atención telefónica, históricos (*logs*) de equipos, medidores inteligentes, información de planta o producción, sistemas de ventas y comercio.

3) Sociales, que son datos obtenidos a través del comportamiento e interacción de los usuarios con las redes sociales, los cuales incluyen retroalimentaciones (*feedbacks*), opiniones y tendencias.

Esta contribución a la acumulación masiva de datos se puede encontrar en diversas industrias. Las organizaciones mantienen grandes cantidades de datos transaccionales, reuniendo información acerca de sus clientes, proveedores y operaciones. En muchos países [16], se administran enormes bases de datos que contienen datos de censo de población, registros médicos, impuestos, ubicación geográfica mediante coordenadas *GPS*, además de transacciones financieras realizadas en línea o por dispositivos móviles.

Por otro lado, están los análisis de redes sociales. Sólo en Twitter son cerca de 12 terabytes de *tweets* creados diariamente. *Facebook* almacena alrededor de 100 *petabytes* de fotos y videos [16]. Si se cuentan todas aquellas actividades, que la mayoría de las personas realizan varias veces al día con los teléfonos inteligentes, se está hablando de alrededor de 2.5 quintillones de *bytes* generados diariamente en el mundo. Por tal razón, en la actualidad, los sistemas computacionales deben trabajar en el orden de los *terabytes* [17].

En la Figura 1 se observa cómo ha sido la evolución de los datos en cuanto a su tamaño, de acuerdo a las operaciones realizadas y los sistemas utilizados. Además, se evidencia el paso con respecto a la tecnología que se empleará, desde un simple dato transaccional, hasta un significativo aumento de los datos generados por los sitios web, los mensajes de texto, el contenido multimedia y las aplicaciones que se encuentran en las populares redes sociales.

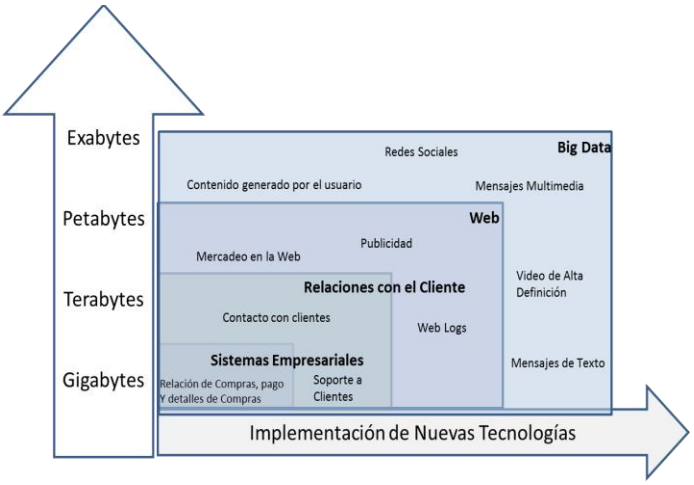


Figura 1: Evolución del Big Data [15]

Big Data incluye una nueva generación de tecnologías y arquitecturas, diseñadas para extraer valor económico de grandes volúmenes de datos heterogéneos, habilitando la captura, identificación y/o análisis a alta velocidad. *Big Data* se caracteriza por tener cinco dimensiones [18]: volumen, variedad, velocidad, veracidad y valor. Estas, son realmente las que definen la frontera entre si se debe o no utilizar *Big Data*. También, dan parámetros a considerar cuándo se piensa contar con el uso tradicional de las bases de datos relacionales. Además, ayudan a decidir si se debe utilizar herramientas existentes para el manejo de datos que no pueden ser tratados

bajo el esquema de tablas normalizadas, facilitando la elección de la herramienta o la aplicación que corresponda [17].

El Volumen es la dimensión que genera la necesidad de procesamiento intensivo y complejo de datos masivos que contienen información de valor para una organización. Se habla de al menos más de un millón de registros [19].

La Variedad indica que la información de valor es el resultado de la combinación de datos de múltiple origen y tipología, presente en forma estructurada, semiestructurada o no estructurada [17].

La Velocidad está asociada a los requerimientos de los procesos y los usuarios [18]. Hoy en día, los datos se generan de forma continua a una velocidad que los sistemas tradicionales no pueden captar, almacenar y/o analizar. Se espera que los sistemas tarden 5 segundos en dar respuesta al procesamiento de los datos.

El Valor hace referencia a los beneficios que se obtienen por el uso de *Big Data*, como, reducción de costes, eficiencia operativa, mejoras del negocio [20].

La Veracidad se refiere al nivel de fiabilidad asociado a ciertos tipos de datos. Conseguir datos de alta calidad es un requisito importante y un reto fundamental de *Big Data*. Sin embargo, aún los mejores métodos de limpieza de datos no pueden eliminar lo imprevisible de algunos datos, como el tiempo, la economía o las futuras decisiones [20].

Fundamentalmente, los sistemas gestores de grandes volúmenes de datos NoSQL están diseñados para aprovechar las nuevas arquitecturas de computación en la nube que han surgido en la última década para permitir cálculos masivos que se ejecuten con bajo costo y de manera eficiente. Esto hace que la gestión de grandes cargas de trabajo sea más fácil, más económica y más rápida de implementar.

La Figura 2 muestra las etapas de la gestión de *Big Data*. La captura es la recolección de datos, procedentes principalmente de la Web, las redes sociales, la interconexión de objetos Máquina a Máquina (M2M), sensores presentes en el Internet de las Cosas (IoT), biometría o directamente de las personas.

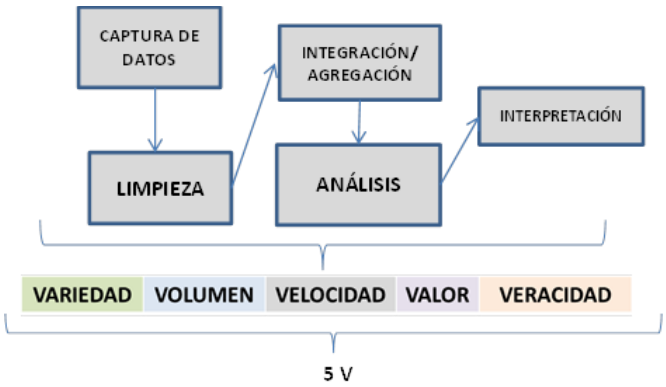


Figura 2: Etapas del Big Data [22]

Boyd y Crawford [21] consideran que la novedad de *Big Data* se encuentra en las capacidades de búsqueda y agregación de grandes cantidades de conjuntos de datos relacionados, más que en el volumen.

La etapa de filtrado y limpieza de los datos puede demandar hasta el 80% del trabajo de análisis [23]. Actualmente, esta etapa se considera una fase previa y separada de los procesos ETL (extraer, transformar y cargar datos). Sin embargo, esto no significa que sea de menor importancia, pues ella asegura la calidad de los datos que se van a procesar, evita información errónea y ayuda en la toma de decisiones correctas.

IV. Antecedentes

En [24], se muestran los avances de la epidemiología gracias a la tecnología digital y al procesamiento de datos masivos (*Big Data*), obtenidos a tiempo real de las poblaciones. Se buscan correlaciones que de forma rápida permitan establecer estadísticas precisas para inferir realidades o causas de fenómenos epidemiológicos, con al menos cierta probabilidad de resultados confiables. Este tipo de datos son muy útiles en investigaciones médicas, para esclarecer la evolución de una posible pandemia, así como, la toma de decisiones relacionadas a los resultados de tales investigaciones. En esas investigaciones no se han considerado uso de datos difusos para modelar las preferencias de los usuarios en la definición de términos médicos, ni tampoco con semántica definida de acuerdo al contexto del usuario.

En el Hospital Birgmingham, de Reino Unido [25], los médicos usaron un sistema conocido como Telemetría para monitorear la recuperación de un niño de un paro cardíaco. Este sistema, normalmente es empleado para evaluar el desempeño del monoplaza en las pistas de carrera de la Fórmula 1. Es la primera vez que se adapta este tipo de sistemas en los seres humanos. Los Laboratorios de la *MCLaren Electronics*, tomaron esta tecnología que recaba una enorme cantidad de información a tiempo real, ya que en cada auto se evalúan cerca de 130 parámetros aproximadamente, que además es capaz de procesarla en corto tiempo.

Estos antecedentes muestran la utilidad del uso de la tecnología de *Big Data* en un área como la médica donde normalmente se presentan datos imperfectos. En [2], se presenta un trabajo de investigación que muestra la gestión de datos imperfectos haciendo uso de tecnologías en el entorno de *Big Data*. El trabajo muestra un caso de estudio en EEUU sobre la atención médica de lesiones de cadera y de antebrazo. Además, exhiben una comparación con el método tradicional de base de datos relacionales.

Desde hace varios años, muchos investigadores han utilizado la Lógica Difusa para modelar e implementar la imperfección de los datos en los sistemas gestores de bases de datos como puede verse en la literatura [26]. La mayoría de estas propuestas están desarrolladas para ser implementadas en Sistemas Gestores de Bases de Datos Relacionales (SGBDR) [27]. Por otra parte, se han abordado diversas aproximaciones teóricas de modelos de datos conceptuales difusos [28].

Si bien la utilización de la Lógica Difusa ha tenido mucho éxito en Ingeniería de Control de la industria manufacturera, existen también un gran número de desarrollos en el área de la Informática, ya que es uno de los pilares para el desarrollo de la Computación Flexible (*Soft Computing*) y Sistemas Inteligentes.

La influencia de la Lógica Difusa sobre sistemas informáticos y en bases de datos, se apoya en que representa un área muy

activa de investigación ya que permite una adaptación al mundo real en que vivimos [28][29]. Así, se han desarrollado trabajos en áreas tales como: Bases de datos Difusas Temporales [30], gestión de datos semánticos [31], recuperación flexible de imágenes médicas [32], citas médicas [33], recursos humanos [34], evaluación de profesores [35].

Un antecedente importante a este artículo lo presentan Cadenas, Marín y Vila [36] quienes proponen una arquitectura para el desarrollo de Sistemas de Bases de Datos Difusas Sensibles al Contexto que sirve de marco para el desarrollo de sistemas inteligentes, flexibles y personalizados al usuario. Una adaptación de esta arquitectura será parte de la propuesta presentada en la próxima sección.

V. Propuesta de Arquitectura para la Gestión de Datos Imperfectos usando Big Data

En este artículo se propone integrar las etapas de desarrollo que considera datos masivos y el trabajo realizado por Cadenas, Marín y Vila [36], que gestiona datos imperfectos sensibles al contexto utilizando una base de datos difusa. En la Figura 3, se observa la arquitectura propuesta que se compone de la gestión de datos imperfectos sensibles al contexto y la tecnología utilizada en *Big Data*. Este modelo mixto plantea el desarrollo de un Módulo de Interacción (MI). En el modelo se muestran las tres fases de administración de *Big Data* que se integran a la arquitectura adaptada de [36], la cual aparece en el lado izquierdo de la Figura 3. Del lado derecho se encuentran esas fases correspondientes al uso de herramientas y tecnologías para datos masivos [37].

La primera parte de la arquitectura corresponde a un Sistema de Bases de Datos Difusas Sensible al Contexto. Como ejemplo, de este tipo de sistema, en [14] se presenta una aplicación para el Examen Físico Articular (EFA) realizado por el Laboratorio de Marcha del Hospital Ortopédico Infantil. Esta aplicación contiene definiciones de etiquetas lingüísticas para diferentes atributos del 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. Tales definiciones se adecuan a las preferencias de cada usuario (médicos o fisioterapeutas).

Los usuarios a través de los módulos de sensibilidad al contexto interactúan con el sistema para aportar instrucciones e información de contexto de manera explícita, como la que recoge su perfil. La interacción puede ser efectuada a través de instrucciones del lenguaje estándar de un Sistema Gestor de Base de Datos Objeto Relacional, o a través de aplicaciones sensibles al contexto diseñadas para usuarios inexpertos, tal como el ejemplo mencionado para el EFA.

Los módulos TCP de Entrada/Salida, se encargan del almacenamiento (ASC) y recuperación (RSC) de los datos sensibles al contexto en la base de datos difusa. Estos módulos se comunican con el Gestor de Apoyo de Sensibilización al Contexto para que sean adaptadas las entradas (o salidas) de datos de acuerdo al contexto, antes de ser almacenadas (o recuperadas) en (o de) la base de datos.

El Gestor de Apoyo de Sensibilización al Contexto (GASC) es el encargado de gestionar los datos contextuales en el Catalogo Contextual y las reglas que permitan inferir comportamiento de acuerdo al contexto. Este sistema se convierte en la meta-base de conocimiento que permite a los Módulos TCP hacer su trabajo de transformación.

En la base de datos se almacenan todos los datos que dan soporte a la información de los dominios contextuales, así como, sus metadatos, además de los datos difusos.

La segunda parte, iniciada en el Módulo de Interacción (MI), corresponde a una capa de integración entre las otras dos partes de la arquitectura. Este módulo se encarga de interpretar los datos sensibles al contexto, que se obtienen a través de la "interacción" con los módulos TCP de entrada y salida, para que puedan ser entendidos por los módulos de Análisis y

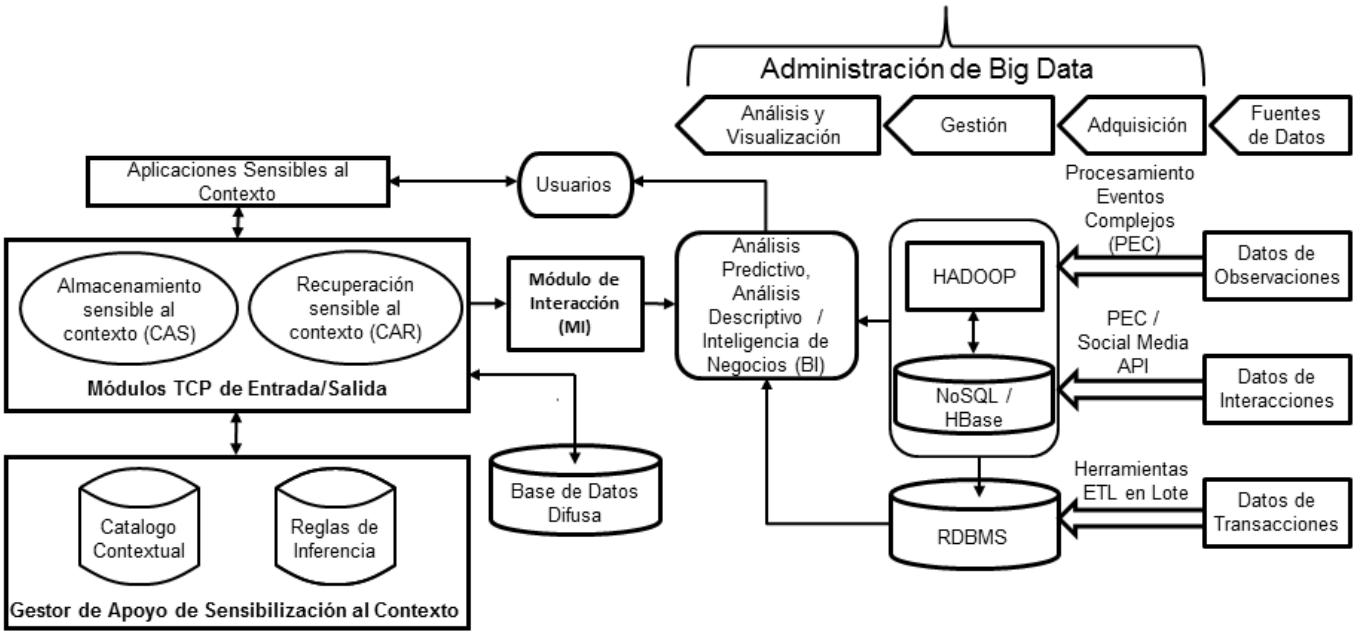


Figura 3: Arquitectura para la Gestión de Datos Imperfectos con *Big Data*

Visualización de la administración *Big Data*. Es importante destacar que los gestores de bases de datos relacionales (SGBDR) no tienen la capacidad de representar y manipular datos imperfectos. Por tal motivo, es necesaria la presencia de una capa que sirva de intérprete, función que en esta arquitectura la haría el Módulo de Interacción.

La tercera parte de la arquitectura corresponde a la administración *Big Data*, que inicia con la fase de adquisición de datos provenientes de diversas fuentes, como son: los datos de observaciones realizadas en procesamientos de eventos complejos (PEC), datos de interacciones obtenidos principalmente de las redes sociales y datos transaccionales. Luego, aparece la fase de gestión, donde no sólo se considera el tratamiento de datos a través de operaciones SQL realizadas en un SGBDR, sino también de datos no estructurados (No sólo SQL ó NoSQL).

Las Bases de Datos NoSQL, como tendencia [38], han venido ganando espacio especialmente por la escalabilidad y velocidad en sus tiempos de respuestas, superiores a los de los sistemas relacionales. En el caso de PostgreSQL, siendo un SGBD Objeto Relacional, ha adquirido varias características de tipo NoSQL, como el almacenamiento momentáneo o fugaz y el manejo de datos en notación de objetos JavaScript (JSON). En [39], dichas características fueron evaluadas con respecto al gestor NoSQL MongoDB, dando como resultado, mejores tiempos de respuestas en todas las operaciones realizadas. Entre las plataformas existentes de tratamiento masivo de datos se pueden mencionar, HADOOP, CouchDB, Neo4J, MapReduce, MongoDB, entre otras.

Finalmente, la administración *Big Data* culmina con la fase de Análisis y Visualización. Aquí se realiza el análisis con técnicas de Inteligencia de Negocio (*Business Intelligence*), predictivas o descriptivas, a fin de producir los resultados o reportes (visualización) que servirán como estrategia para la toma de decisiones de la organización, los cuales están disponibles al usuario de la arquitectura. Es en esta fase donde el módulo de interacción le proporciona los datos difusos y contextuales transformados para que sean considerados durante el análisis. De tal forma que el usuario dispondrá de resultados en base a sus preferencias y contexto en que se ubica.

En resumen, se observan tres partes en esta arquitectura. La primera parte, que trata de la gestión de datos imperfectos sensibles al contexto. La segunda parte, corresponde al módulo de interacción, que sirve de mediador entre esta gestión y la tecnología *Big Data* a utilizar. Finalmente, la tercera parte, se refiere a las etapas del desarrollo *Big Data*.

VI. Conclusiones y Trabajos Futuros

El manejo de grandes cantidades de datos ha ayudado a los investigadores a hacer descubrimientos que les podrían haber tomado años en descifrar por si mismos sin el uso de herramientas para procesar y analizar dichos datos. Debido a la abundancia de sensores, micrófonos, cámaras, escáneres médicos, imágenes, entre otros; presentes en la vida cotidiana, los datos generados a partir de estos elementos serán dentro de poco el segmento más grande de toda la información disponible.

Desde hace varios años, los investigadores han utilizado la teoría de los conjuntos difusos y la lógica difusa, permitiendo modelar la incertidumbre en las Bases de Datos, así como representar las diferentes preferencias de los usuarios. A través de los conjuntos difusos, se modelan los datos posibilísticos, conceptos comúnmente utilizados en el lenguaje natural. Por otro lado, se ha usado esta teoría para representar la información contextual obteniendo aplicaciones inteligentes que gestionan datos imperfectos sensibles al contexto.

Los resultados obtenidos en estas investigaciones previas se han considerado en la propuesta presentada en este trabajo a fin de obtener una arquitectura de un sistema de Base de Datos que gestiona la imperfección, así como, la información del contexto del usuario, con tecnología para la gestión de datos masivos. Se plantea el uso de un Módulo de Interacción (MI), el cual se encarga de la integración entre el módulo de Bases de Datos Difusas Sensibles al Contexto y la fase de análisis y visualización de la administración *Big Data*.

De esta forma, en esta propuesta se incluyen avances ya logrados en cuanto a la gestión de datos imperfectos mediante su almacenamiento y recuperación en Sistemas Gestores de Bases de Datos flexibles. La gestión de datos imperfectos, que incluye ingreso, actualización, eliminación, consulta y extracción de conocimiento, proporciona una mejor aproximación de la información manejada en el mundo virtual con respecto al universo real de los usuarios. Por otro lado, se abarca también las funcionalidades referentes a la formulación de consultas flexibles, por parte de los usuarios, que utilicen términos imperfectos a pesar de que los datos sean precisos.

En este trabajo, se pudieron identificar las características más importantes de la gestión de datos masivos, que incluyen los diferentes formatos, existentes hoy en día, manejados por los usuarios; así como las tecnologías necesarias para convertir datos no estructurados en información y conocimiento que beneficie tanto a personas como a empresas en la toma de decisiones.

Parte de la investigación arrojó que existe una gran variedad de herramientas tecnológicas que permiten el análisis de datos. En su mayoría, dichas herramientas están basadas en Apache HADOOP, disponibles online para ser trabajadas en ambiente web, otras para ser instaladas en computadores personales y algunas para ser usadas en la nube.

Como trabajos futuros, se plantea el desarrollo del Módulo de Interacción propuesto, según los requerimientos descritos. Además, estudiar la utilización de PostgreSQL como Sistema Gestor de Bases de Datos, pues posee las bondades de ser código abierto, incluye características Objeto Relacional e incorpora propiedades del paradigma NoSQL.

También, se plantea la aplicación de la arquitectura propuesta en un caso de estudio. Finalmente, sería deseable lograr incorporar otros tipos de datos imperfectos a esta arquitectura, además de los posibilísticos, tales como los probabilísticos y bimodales tal como lo plantea Zadeh en el Principio de la Información [13].

Agradecimientos

Los autores de esta investigación expresamos agradecimiento por el apoyo incondicional al Prof. Leonid Tineo, Prof. Orlando Luna, Prof. Josué Ramírez y a nuestros familiares por su consideración y aliento. “Te damos gracias, oh Dios, te

damos gracias, pues cercano está tu nombre; los hombres declaran tus maravillas" (Salmos 75:1).

REFERENCIAS

[1] J. Torres, *Big Data: Por qué las Bases de Datos no Sirven para Todos*, TecnoNews, http://www.tecnonews.info/opiniones, Mayo 2013.

[2] M. Hilbert, *Big Data for Development: from Information to Knowledge Societies*, Social Science Research Network, http://www.martinhilbert.net/wp-content/uploads/2015/01/BigData4Dev _Hilbert2014.pdf, 2013.

[3] G. De Tré and S. Zadrozny, *The Application of Fuzzy Logic and Soft Computing in Information Management*, Fuzzy Sets and Systems, vol. 160, no. 15, pp. 2117-2119. 2009.

[4] L. Yan and Z. M. Ma, *Modeling Fuzzy Information in Fuzzy Extended Entity-Relationship Model and Fuzzy Relational Databases*, Journal of Intelligent & Fuzzy Systems, vol. 27, no. 4, pp. 1881–1896, 2014.

[5] M. A. Vila, J. C. Cubero, J. M. Medina, and O. Pons, *A Conceptual Approach for Dealing with Imprecision and Uncertainty in Object based Data Models*, Int. J. Intell. Syst, vol. 11, no. 10, pp. 791-806, 1996.

[6] J. Ramirez and L. Tineo, *Un Mecanismo de Respuesta a Consultas en Presencia de Nulos*, Revista Venezolana de Computación (ReVeCom), vol. 2, no. 1, pp. 48-59, 2015.

[7] L. Zadeh. *Soft Computing and Fuzzy Logic*. IEEE Software, vol. 11, no. 6, pp. 48-56, 1994.

[8] A. Motro, *Accommodating Imprecision in Database Systems: Issues and Solutions*, ACM Sigmod Record, vol. 19, no. 4, pp. 69-74, 1990.

[9] S. Sarkar and D. Dey, *Relational Models and Algebra for Uncertain Data*, In Managing and mining Uncertain Data, C. Aggarwal (ed.), vol. 35, series Advances in Database Systems, Springer, USA, 2009.

[10] C. Aggarwal, *Managing and Mining Uncertain Data*, vol. 35 of the series Advances in Database Systems, Springer, USA, 2009.

[11] L. Zadeh, *Fuzzy Sets*, Information and Control, vol. 8, no. 3, pp. 338-353, 1965.

[12] L. Zadeh, *Fuzzy Sets as a Basis for a Theory of Possibility*, Fuzzy Sets and Systems, vol. 1, pp. 3-28, 1978.

[13] L. Zadeh, *The Information Principle*, Information Sciences, vol. 294, pp. 540–549, 2015.

[14] J. T. Cadenas, *Sistemas de Bases de Datos Difusas Sensibles al Contexto*, Disertación Doctoral, Universidad de Granada, España, 2015.

[15] M. L. Tascon, *Introducción al Big Data. Pasado, Presente y Futuro*, Revista Telos, no. 95, https://telos.fundaciontelefonica.com/seccion=1268&idioma=es_ES&id=2013062110090002&activo=6.do, 2013.

[16] A. Rattinger, *Mejora tu Marketing con Big Data*, Revista Mercado 2.0, vol. 6, no. 3, http://www.merca20.com/big-data, 2014.

[17] J. Curto, *Big Data: Un Mercado Emergente*, Madrid, España, 2012.

[18] IBM Big Data and Analytics Platform, http://www-01.ibm.com /software/data/bigdata, 2012.

[19] L. Joyanes, J. F. Camargo, and J. J. Camargo, *Conociendo Big Data*, Revista Facultad de Ingeniería, vol. 24, no. 38, pp. 63-77, 2014.

[20] M. Chroeck, R. Shockley, J. Smart, D. Romero-Moreno, and P. Tufano, *Analytics: El Uso de Big Data en el Mundo Real*, IBM Global Business Services, IBM Institute for Business Value, 2012.

[21] D. Boyd and K. Crawford, *Critical Questions for Big Data: Provocations for a Cultural, Technological, and Scholarly Phenomenon*, Information, communication & society, vol. 15, no. 5, pp. 662-679, 2012.

[22] F. Malvicino and G. Joguel, *Big Data: Avances Recientes a Nivel Internacional y Perspectivas para el Desarrollo Local*, CIETI, Universidad Nacional General Sarmiento, Buenos Aires, Argentina, Agosto 2015.

[23] E. Dumbell, *Plannig for Big Data*, A CIO's Handbook to the Changing Data Landscape, 1st Edition, O'Really Radar Team, USA, March 2012.

[24] D. Casacuberta, *Innovación, Big Data y Epidemiología*, Revista Iberoamericana de Argumentación, no.7, pp. 1-12, 2013.

[25] H. Campos, *La Tecnología de MCLaren Salva Vidas en un Hospital Infantil de Inglaterra*, http://www.caranddriverthef1.com, Madrid, España, 2016.

[26] F. E. Petry, *Fuzzy Set Theory Utility for Database and Information Systems*, On Fuzziness, R. Seising, E. Trillas, C. Moraga and S. Termini (eds.), vol. 299, pp. 547-552, series Studies in Fuzziness and Soft Computing, Springer Berlin Heidelberg, 2013.

[27] J. Galindo, *Handbook of Research on Fuzzy Information Processing in Databases*, 1st edition, IGI Global, May 2008.

[28] Z. M. Ma and Y. Li, *A Literature Overview of Fuzzy Conceptual Data Modeling*, J. Inf. Sci. Eng, vol. 26, no. 2, pp. 427-441, 2010.

[29] L. Borjas, J. Ramírez, R. Rodríguez, and L. Tineo, *Automated System for Tests Preparation and Configuration Using Fuzzy Queries*, Computational Intelligence, K. Madani, A. D. Correia, A. Rosa and J. . Filipe (eds.), pp. 199-212, 1st edition, Springer International Publishing, 2015.

[30] J. M. Medina, J. E. Pons, C. D. Barranco, and O. Pons, *A Fuzzy Temporal Object‐Relational Database: Model and Implementation*, International Journal of Intelligent Systems, vol. 29, no. 9, pp. 836-863, 2014.

[31] J. R. Campaña, J. M. Medina, and M. A. Vila, *Semantic Data Management Using Fuzzy Relational Databases*, Flexible Approaches in Data, Information and Knowledge Management, O. Pivert and S. Zadrozny (eds.), vol. 497, pp. 115-140, Serie Studies in Computational Intelligence, Springer International Publishing, September 2013.

[32] J. M. Medina, S. Jaime-Castillo, C. D. Barranco, and J. R. Campana, *On the Use of a Fuzzy Object-Relational Database for Flexible Retrieval of Medical Images*, IEEE Transactions on Fuzzy Systems, vol. 20, no. 4, pp. 786-803, 2012.

[33] T. Toshiro, K. Asai, and M. Sugeno, *Applied Fuzzy Systems*, 1st edition, Academic Press, San Diego, USA, 1994.

[34] L. Lien-Fu, W. Chao-Chin, H. Yi-Ta, and H. Liang-Tsung, *A Fuzzy Query Approach to Human Resource Web Services*, in proceeding of the 10th International Conference on e-Business Engineering (ICEBE), pp. 461-466, Coventry, United Kindom, September 2013.

[35] A. Aguilera, L. Borjas, R. Rodríguez, and L. Tineo, *Experiences on Fuzzy DBMS: Implementation and Use*, in proceeding of the XXXIX Latin American Computing Conference (CLEI 2013), pp. 1-8, Naiguatá, Venezuela, October 2013.

[36] J. T. Cadenas, N. Marín, and M. A. Vila, *Context-Aware Fuzzy Databases*, Applied Soft Computing, vol. 25, pp. 215-233, 2014.

[37] H. Cheng, R. Chiang, and V. Storey, *Business Intelligence and Analytics: from Big Data to Big Impact*, MIS Quarterly, vol. 36, no. 4, pp. 1165-1188, December 2012.

[38] N. Leavitt, *NoSQL Databases Live Up to Their Promise?*, Journal Computer, vol. 43, no. 2, pp.12-14, USA, 2010.

[39] A. Sotolongo, L. Vasquez, and Y. Vazquez, *Evaluación de Características NoSQL en PostgreSQL*, http://es.slideshare.net/asotolongo/caractersticas -nosql-de-postgresql, La Habana, Cuba, 2013.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 57-67, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Ontología para las Manifestaciones Rupestres en Venezuela: Hacia el Desarrollo de una Plataforma para la Preservación Digital

Alejandro Amaro[1], Federico Flaviani[2], Alejandro Figueroa[3], Ruby De Valencia[4], Yudith Cardinale[2]
aamaro@cuc.edu.ve, fflaviani@usb.ve, figueroa02@gmail.com, anarvenezuela@gmail.com, ycardinale@usb.ve

[1] Departamento de Investigación, Colegio Universitario de Caracas, Caracas, Venezuela
[2] Departamento de Computación, Universidad Simón Bolívar, Caracas, Venezuela
[3] Escuela de Informática, Universidad Católica Andrés Bello, Caracas, Venezuela
[4] ANAR, Fundación Venezolana para la Preservación del Hábitat, Promoción y la Defensa de las Culturas, Caracas, Venezuela

Resumen: En este artículo se estudia el dominio de las Manifestaciones Rupestres en Venezuela y se presentan los primeros resultados de extraer y representar, en una ontología, la *semántica implícita* en formularios desarrollados por el equipo arqueológico del Archivo Nacional de Arte Rupestre (ANAR), encargado de llevar el registro manual de los hallazgos en un total de 650 yacimientos, distribuidos en los 24 estados de Venezuela. La información plasmada en dichos formularios físicos, representa una inmensa e invalorable fuente de datos, a los que se podría acceder a través de servicios abiertos vía Internet. Un requisito previo para proporcionar estos servicios es la construcción de una descripción del contenido semántico del dominio. Lo siguiente sería el desarrollo de una plataforma tecnológica para su preservación digital. Este trabajo se concentra en el primer paso hacia esa plataforma de apoyo a los expertos del dominio que proporcionará un rico repositorio de información en el tema. En este contexto, se presenta una ontología para las Manifestaciones Rupestres en Venezuela que permite representar los datos, serializarlos y utilizarlos para desarrollar servicios *endpoint* basados en las tecnologías de la Web Semántica. Como una prueba de conceptos, se muestra el desarrollo de un servicio *endpoint* para consultar contenidos multimedia, anotados con la ontología propuesta. Esta experiencia permite, además, proponer un diseño preliminar de la plataforma de preservación digital de las Manifestaciones Rupestres y una estrategia para desarrollarla.

Palabras Clave: Manifestaciones Rupestres; Web Semántica; Ontologías; Patrimonio Cultural; Documentación.

Abstract: In this article we study the domain of the Rock Art Manifestations in Venezuela and present the first results of extracting and representing, in an ontology, the semantics implicit in forms developed by the archaeological team of the National Rock Art Archives (ANAR), in charge of keeping the manual registry of the findings in a total of 650 sites, distributed across the 24 states of Venezuela. The information contained in these physical forms represents an immense and invaluable source of data, which could be accessed through open services via the Internet. A prerequisite for providing these services is the construction of a semantic content description for this domain. The following would be the development of a technological platform for its digital preservation. This work focuses on the first step towards that platform to support domain experts and to provide a rich repository of information on the topic. In this context, we present an ontology for the Rock Art Manifestations in Venezuela that allows us to represent the data, serialize them and use them to develop endpoint services based on Semantic Web technologies. As a test of concepts, the development of an endpoint service to consult multimedia contents, annotated with the proposed ontology, is shown. This experience also allows to propose a preliminary design of the platform for the digital preservation of the Rock Art Manifestations and a strategy to develop it.

Keywords: Rock Art Manifestations; Semantic Web; Ontology; Cultural Heritage; Documentation.

I. Introducción

El Archivo Nacional de Arte Rupestre (ANAR) [1], es un proyecto desarrollado por FUNDABITAT (Fundación Venezolana para la Preservación del Hábitat, Promoción y Defensa de las Culturas), en el marco de su Programa de Formación y Capacitación FACILITADORES EN ESCUELA ACTIVA, cuyo objetivo principal es: Servir de Centro de Referencia y Servicio de Información para el conocimiento y protección de las Manifestaciones Rupestres en Venezuela, como parte de la Conservación de la Memoria Histórica y Documental de

este Patrimonio Cultural Arqueológico del país. Entre otras actividades realiza el registro de los sitios arqueológicos donde se encuentran las Manifestaciones Rupestres venezolanas. El ANAR cuenta actualmente de los siguientes documentos:

- Documentos relacionados a las Manifestaciones Rupestres en Venezuela, que consisten de: (a) fichas de trabajo de campo que describen cada una de las 650 estaciones rupestres (contentivas de 1 a 30 piedras por estación) registradas hasta la fecha, con las respectivas fotografías o ilustraciones tanto del yacimiento como de las piedras; (b) fichas cartográficas en las que se indica la ubicación de cada estación rupestre registrada, así como los exploradores que la han reseñado, dibujado o fotografiado, el año y la página en que se publicaron los hallazgos; (c) carpetas de documentos bibliográficos que consisten del compendio hasta la fecha de un total de documentos bibliográficos y hemerográficos referentes a los yacimientos con Manifestaciones Rupestres venezolanas registradas en el ANAR; (d) material no bibliográfico, donde se encuentran fotografías e ilustraciones de cada una de las estaciones rupestres y piedras; (e) carpetas de documentos bibliográficos de otros países, que consisten del compendio de documentos bibliográficos y hemerográficos referentes a las Manifestaciones Rupestres en otros países; (f) fichero bibliográfico que integra más de 1000 referencias bibliográficas relativas a las Manifestaciones Rupestres de Venezuela en general, así como una sección de tópicos relacionados (navegación primitiva, pintura corporal, técnicas modernas de prospección arqueológica, entre otros) y Arte Rupestre de otros países, con una cantidad similar de entradas; (g) material no clasificado, que corresponde a un extenso número de materiales (acumulados en fecha posterior a la publicación en 1987 del libro acerca del diseño de los petroglifos venezolanos [2]), los cuales están siendo permanentemente recolectados y procesados para su integración al ANAR. Todos los documentos referidos en esta sección están codificados de manera cruzada, lo que permite una infinita variedad de relaciones que facilitan la labor del investigador.
- Documentos bibliográficos sobre antropología general. Las carpetas de documentos de este tipo están agrupadas en diferentes categorías antropológicas tales como Paleontología, Arqueología, Etnología, entre otras, y subdivididas luego por países.
- Documentos hemergráficos sobre antropología general Las carpetas de documentos de este tipo se agrupadan siguiendo aproximadamente la misma clasificación de los documentos bibliográficos.

Este artículo presenta los primeros resultados de una serie de esfuerzos para extraer la *semántica implícita* en dichas fichas, formularios y carpetas, y plasmarla en una ontología del dominio que permita digitalizar fácilmente toda esta información. La necesidad de hacer disponible y reutilizable esta rica e invaluable información lleva a pensar en una implementación más allá de la tecnología de las bases de datos convencionales. No se trata sólo de hacer una aplicación alrededor de los datos, sino más bien de exponer los datos para permitir su reutilización por todos aquellos interesados en construir aplicaciones.

Un requisito previo para proporcionar este servicio es la construcción de una descripción del contenido semántico del dominio con el cual se podría iniciar el trabajo de ingeniería para crear soluciones a las necesidades de información que se presentan en el campo cultural por parte de entes cuya actividad se relaciona con este tema. Esta descripción semántica dará paso al desarrollo de una plataforma tecnológica completa para la preservación digital de este arte, lo cual complementa las estrategias adoptadas para preservar el Patrimonio Rupestre presente en estos yacimientos. Es decir que la plataforma, por un lado, apoyaría a los expertos del dominio proporcionando un repositorio de información, que puede convertirse en una referencia a nivel nacional y latinoamericano al ofrecer un conjunto detallado de datos abiertos y enlazados acerca de las manifestaciones rupestres. Por otro lado, también serviría para promover el conocimiento y la preservación de la riqueza cultural, haciendo que la información esté al alcance de todos a través de servicios Web y ayudaría a la preservación de este invaluable patrimonio para la posteridad.

En contexto, en este trabajo de investigación se propone una ontología de dominio para las Manifestaciones Rupestres venezolanas. Se utiliza en primer lugar la metodología de construcción de ontologías de Uschold y King [3] para describir el contenido semántico del dominio y luego se utiliza la herramienta Protégé [4] para obtener una representación de la ontología en el lenguaje OWL [5]. Con esta ontología se logra una descripción del contenido semántico del dominio que sirve para representar los datos, serializarlos y utilizarlos para generar un servicio *endpoint* que permita desarrollar aplicaciones construidas con las tecnologías de la Web Semántica. Como una prueba de conceptos, se presenta en este documento el desarrollo de un servicio *endpoint* para consultar contenidos multimedia, anotados con la ontología propuesta. Además, esta experiencia permite proponer un diseño preliminar de la plataforma para preservación digital de las Manifestaciones Rupestres y una estrategia para desarrollarla.

El resto de este artículo se organiza de la siguiente manera. La Sección II contiene una breve relación de algunos trabajos recientes vinculados al modelado del patrimonio cultural. En la Sección III se describe el ANAR y el tipo de información que maneja. La Sección IV explica el enfoque, la metodología y las herramientas utilizadas para resolver el problema. En la Sección V se presenta la ontología propuesta y en la Sección VI se describe el servicio de consulta desarrollado y se propone un diseño preliminar de la plataforma y una estrategia para continuar este trabajo. Finalmente, la Sección VII contiene las conclusiones de esta investigación.

II. Trabajos Relacionados

El modelado del dominio del patrimonio cultural es importante porque permite el registro de los elementos y manifestaciones del patrimonio y la posterior difusión del conocimiento con el fin de ayudar a la preservación de un legado que pertenece

a las generaciones futuras y que forma parte de la identidad de cada nación. Las Manifestaciones Rupestres son un subdominio del dominio del patrimonio cultural y representa una parte del aspecto tangible de este patrimonio. Un desarrollo que antecede a este trabajo es la metodología para el registro y documentación de las Manifestaciones Rupestres venezolanas generada como parte del proyecto del ANAR [6]. Esta metodología contiene una especificación de los elementos de información que deben ser recolectados durante el trabajo de campo en las expediciones arqueológicas que tienen como objeto de estudio los yacimientos de las Manifestaciones Rupestres venezolanas. El ANAR cuenta con formularios o fichas diseñados especialmente para la documentación de los hallazgos, un tipo de ficha para registrar los yacimientos y otro tipo de ficha para registrar las rocas que están en el yacimiento. Toda esta información constituyó la base y el conocimiento experto para la definición de la ontología.

Entre los trabajos recientes, basados en el modelado de ontologías, vale la pena mencionar los esfuerzos por registrar, analizar y comparar las manifestaciones rupestres presentes en Monte Bego [7][8], al sur occidente de Francia y cerca de la frontera con Italia. En este caso se ha desarrollado un Sistema Multi-Agente que utiliza ontologías basadas en la investigación de campo de los arqueólogos para llevar a cabo la integración de los datos, su análisis, normalización, interpretación y enriquecimiento semántico, el cual tiene como objetivo a largo plazo la creación de un repositorio a nivel de toda Europa de los petroglifos de la Edad del Bronce. Una de las habilidades importantes de este sistema es la capacidad de recuperación de contenidos basada en los dibujos de los petroglifos. En vista de que se desea lograr una integración de los datos para toda Europa, es necesario el desarrollo futuro de un modelo generalizable capaz de integrar progresivamente los datos de otras colecciones como los petroglifos de la región de Valcamonica [9]. En la misma línea de trabajo debemos incluir la creación de la Ontología Indiana [10] para el modelado del conocimiento acerca del Arte Rupestre de Monte Bego [11], [12], cuyo diseño no incluye el uso de otras ontologías creadas para el procesamiento de objetos del patrimonio cultural, sino que se enfoca en una estrecha integración con el sistema IndianaMAS, es decir, los agentes de software inteligentes y la biblioteca digital utilizada para clasificar y almacenar objetos multimedia y multilinges acerca del Arte Rupestre de Monte Bego. La ontología IndianaMAS fue implementada en OWL utilizando Protégé.

La Base de Datos Global del Arte Rupestre (The Global Rock Art Database) [13] es una organización virtual orientada a construir una comunidad global del Arte Rupestre. Este proyecto reúne a profesionales y entusiastas del Arte Rupestre de todo el mundo a través del RADB Management System el cual está disponible públicamente. Esta plataforma en línea utiliza el sector de patrimonio australiano y el Arte Rupestre australiano como modelo de prueba para desarrollar un método para el diseño del sistema RADB. La idea original surgió a partir de un proyecto doctoral, la prueba del modelo conceptual se llevó a cabo en 2014 y se hizo disponible para el público general, pero el acceso completo se encuentra limitado actualmente a un pequeño grupo de usuarios de prueba seleccionados.

Al igual que estos trabajos, pretendemos que la ontología propuesta permita, por un lado, ser usada y extendida para la descripción de las Manifestaciones Rupestres en Latino América y por otro lado, ser la base para el desarrollo de servicios interconectados que provean un gran repositorio de Manifestaciones Rupestres latinoamericanas. La idea además, es que estos servicios puedan integrarse con servicios ya existentes como los descritos anteriormente.

En Chile, un equipo de investigación ha trabajado recientemente en un método para obtener conocimiento a partir de bases de datos existentes con la finalidad de crear un modelo ontológico para el patrimonio cultural intangible [14]. Si bien las Manifestaciones Rupestres son un Patrimonio Cultural tangible, este trabajo ha sido motivo de inspiración para emprender la tarea del modelado del patrimonio cultural venezolano con base en los registros existentes, realizados en la geografía local venezolana por especialistas del dominio, como los miembros del equipo arqueológico del proyecto ANAR. Al mismo tiempo se presenta la oportunidad de intercambiar experiencias con los grupos de investigación que realizan trabajos semejantes en nuestro continente, y realizar aprendizajes a través del estudio y la comparación de las características comunes y divergentes que presenta el dominio del patrimonio cultural en distintas localizaciones geográficas. Son estos intercambios los que pueden conducir a una solución de ingeniería que permita la integración de los datos y los servicios a escala latinoamericana. El punto de partida en este caso es el subdominio de las Manifestaciones Rupestres, pero manteniendo la vista puesta sobre la necesidad de abordar también el modelado de los otros subdominios del patrimonio cultural tangible e intangible.

III. Manifestaciones Rupestres en Venezuela: un Problema de Información

El ANAR es el miembro N° 39 de los 50 en el mundo y único miembro en Venezuela de IFRAO (International Federation of Rock Art Organizations) adscrito a UNESCO. Mantiene una permanente relación con la comunidad a través de conferencias audiovisuales dictadas a colegios, universidades y otras entidades que lo soliciten. Su participación en el medio académico es constante. Viene desarrollando, en el marco de su Programa de Educación, el tema de las manifestaciones rupestres venezolanas de forma inter y transdisciplinaria, utilizando las TIC, con el objeto de promover su conocimiento, difusión y preservación en las diversas áreas de aprendizaje, de manera que se involucre en el pensum oficial de los subsistemas de educación primaria y secundaria. Para el ANAR la educación, capacitación y toma de conciencia, es un proceso permanente, progresivo y coherente, dirigido a la formación de valores, conocimientos y conductas, cónsonas con la preservación del patrimonio cultural, la conservación ambiental y el desarrollo sostenible [2][15].

Las Manifestaciones Rupestres constituyen los indicios más antiguos conocidos de un intento de comunicación humana.

En ausencia de testimonios de la expresión oral hasta la aparición de la escritura, tenemos en cambio fragmentos que representan en imágenes el mundo de las primeras sociedades. En Venezuela, se encuentran repartidas prácticamente por todo el territorio: en las sabanas, en las altas montañas, a las orillas y en medio de los ríos y también en las selvas. La Figura 1 presenta fotos de cuatro Manifestaciones Rupestres presentes en Venezuela. La existencia de un muy rico y variado patrimonio rupestre, convierte a Venezuela en uno de los países donde están presentes casi todas las posibles manifestaciones de este Patrimonio Cultural Arqueológico:

- Petroglifos (grabados en rocas – ver Figura 1a),
- Pinturas Rupestres o Pictografías (dibujos realizados sobre las rocas, mediante la aplicación de pigmentos – ver Figura 1b),
- Monumentos Megalíticos (Menhires: grandes piedras alargadas, colocadas verticalmente en el suelo – ver Figura 1c; Monolitos: bloques de piedra de gran tamaño, con grabados; Dólmenes: Piedras verticales, cubiertas con losas horizontales de gran tamaño, conformando paredes y techo),
- Geoglifos (grabados gigantes en tierra – ver Figura 1d),
- Micropetroglifos (pequeños guijarros o lajas líticas con grabados),
- Amoladores líticos (depresiones de formas diversas para amolar instrumentos líticos),
- Cúpulas y Puntos Acoplados (grabados de formas semicirculares de diverso tamaño, en la roca),
- Bateas (depresiones de formas rectangulares o circulares cortadas en la rocas),
- Piedras y Cerros Míticos Naturales (Piedras o Cerros no trabajados por el hombre, con deformaciones naturales por las que se les atribuyen explicaciones mitológicas generalmente relacionadas con el origen de los Petroglifos).

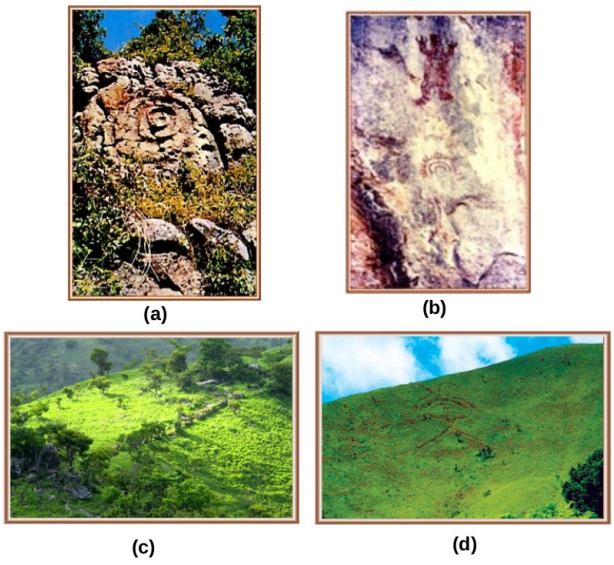


Figura 1: Ejemplos de Manifestaciones Rupestres en Venezuela

Aún cuando FUNDABITAT, en colaboración con personal de la Universidad Simón Bolívar, ha adelantado un esfuerzo para llevar a cabo el registro digital de todas las Manifestaciones Rupestres encontradas en cada uno de los 24 estados venezolanos, su importancia es tanta (dentro del contexto del nutrido Patrimonio Cultural de Venezuela, y como legado de culturas originarias en buena medida desaparecidas) como grande es el riesgo de su deterioro, debido a la ignorancia en la que se encuentran los habitantes del país en relación con su presencia y su relevancia. Por absurdo que pudiera parecer, son los mismos pobladores del país los que representan este riesgo de daño y destrucción.

Las actividades y proyectos del ANAR, se enmarcan en el área de la CONSERVACIÓN PREVENTIVA, basándose en el planteamiento que no se valora lo que no se conoce, impulsan la protección de este Patrimonio Cultural a través de su conocimiento y difusión. para la transmisión de sus contenidos, con estrategias pedagógicas de divulgación, que involucren a la comunidad educativa y público en general de forma lúdica, y a través de recorridos temáticos, que formen parte de un proceso de enseñanza-aprendizaje que informe, dando la posibilidad de interpretar, conservar y preservar estas riquezas ecológico-arqueológicas.

El ANAR está involucrado en este esfuerzo de colocar todo el conocimiento que posee en un sistema de información Web [15]. Esto permite que las personas puedan consultar la información y tener acceso a otros servicios y productos que ofrece ANAR. Sin embargo, esta labor no es suficiente para lograr que se desarrolle software capaz de obtener la información y procesarla para su uso por parte de las instituciones del estado que requieren esa información como insumo para planificar acciones en resguardo del Arte Rupestre o actividades formativas para la población.

El almacenamiento de la información mediante el uso de tecnologías de almacenamiento y recuperación típicos, como las bases de datos relacionales, tiene el inconveniente de que los datos no están disponibles abiertamente para los especialistas en el ámbito nacional o internacional ni para las máquinas que ejecutan consultas de datos (usando, por ejemplo, el lenguaje SPARQL) sobre los recursos de la Web. Esto limita la posibilidad de desarrollar y expandir la literatura bajo la forma de estudios académicos, artículos en prensa escrita y revistas y todas las valoraciones que pudieran hacerse de esta información desde aproximaciones de diversa índole (históricas, antropológicas, sociológicas, entre otras) y por diferentes medios. Todo esto es esencial para la difusión del conocimiento acerca de las Manifestaciones Rupestres venezolanas.

El problema es, entonces, la falta de una fuente o servicio accesible de datos abiertos y enlazados para el dominio de las Manifestaciones Rupestres en Venezuela. Un requisito previo para poder proporcionar esta fuente de datos es la construcción de una descripción o diseño del contenido semántico del dominio de las Manifestaciones Rupestres.

Así, el objetivo principal de este trabajo consiste en crear las condiciones para que los datos estén disponibles para cualquier tipo de procesamiento por parte de cualquier persona

u organización que la requiera. La solución futura en su última fase deberá ayudar a responder preguntas tan sencillas como: *Cuáles son los lugares donde están presentes las Manifestaciones Rupestres venezolanas? Cuántos son? Cuáles son en cada estado, en cada municipio? Cuáles son las características de este yacimiento? Hay una fotografía? Existe documentación de este sitio (artículos en prensa, revistas, documento en ANAR)?* Así como consultas más avanzadas que impliquen búsquedas de relaciones de datos más complejas, descubrir relaciones e inferir en la información disponible.

La importancia de una solución en este contexto radica en que las Manifestaciones Rupestres no son solo Patrimonio venezolano, también lo son del resto de la humanidad. El rastro de la cultura de cada grupo humano que ha existido sobre la Tierra contribuye, igual que una pieza en un rompecabezas, a la comprensión global que se puede tener hoy acerca de la cultura humana y sus raíces como un todo.

Pero sobre todo se debe comprender, que las Manifestaciones Rupestres son un Patrimonio NO RENOVABLE.

Al proponer una noción unitaria de Patrimonio Cultural y Patrimonio Natural, tradicionalmente considerados como aspectos separados e independientes entre sí, se introduce la idea concreta de PATRIMONIO MUNDIAL, cuya importancia trasciende las fronteras políticas y geográficas, y se establece además, como otro objetivo esencial: "Crear una conciencia general más clara y activa del valor de este Patrimonio" y de los graves peligros que lo amenazan. No obstante, en última instancia, corresponde a cada país y región asumir la responsabilidad de la Conservación de su propio PATRIMONIO.

IV. Enfoque Metodológico para la Creación de una Ontología para las Manifestaciones Rupestres en Venezuela

Las tecnologías de la Web Semántica basadas en ontologías permiten superar las limitaciones semánticas y de acceso que acompañan a las bases de datos y al modelado convencional de los datos, ya que las ontologías incorporan ventajas tales como el vocabulario y la capacidad de utilizar la inferencia en el procesamiento de la información, además de la capacidad para entregar la información como un servicio Web de datos abiertos y enlazados mediante la creación de *endpoints* que permitan el procesamiento de consultas y el aprovechamiento de los datos por parte de máquinas además de humanos.

Para resolver el problema se propone crear una descripción del contenido semántico del dominio de las Manifestaciones Rupestres venezolanas mediante la construcción de una ontología de dominio con el apoyo del *framework* Protégé [4], herramienta de uso común en la Web Semántica que permite generar el código en lenguaje OWL [5]. Protégé podrá ser utilizado posteriormente para implantar la ontología en un servicio Web del tipo *endpoint* mediante alguna plataforma como Jena [16] o Virtuoso [17], que se utilizará para poblar la ontología y proveer los datos a las aplicaciones remotas.

A. Metodología para el Desarrollo de la Ontología

De acuerdo a lo propuesto por Upschold et al. [3], los pasos para el desarrollo de esta ontología son los siguientes:

- Identificación de los conceptos y relaciones claves en el dominio de interés.
- Producción de definiciones en forma de texto, precisas y sin ambiguedades.
- Identificación de términos para referirse a esos conceptos y relaciones.
- Alcanzar acuerdos a nivel de la comunidad acerca de todo lo anterior.

La existencia de un trabajo previo llevado a cabo como parte del proyecto ANAR para sistematizar el trabajo arquelógico de campo, mediante el diseño de formularios que resumen la práctica profesional de los expertos en el dominio, facilita la realización de estos pasos, puesto que buena parte de las definiciones ya existen y están presentes en los instrumentos de recolección de información. Para ilustrar esta correspondencia, la Figura 5, que aparece más adelante, exhibe en su lado izquierdo parte de un formulario del proyecto ANAR, mientras que el lado derecho de la misma imagen se muestra una parte de la ontología cuyas categorías taxonómicas corresponden a los conceptos que contiene el formulario del ANAR utilizado por el equipo arqueológico para el registro de hallazgos en actividades de campo. Esto se hace siempre que es posible en el modelado para hacer coincidir la ontología con los conceptos que manejan los expertos del dominio.

En apoyo de esta metodología, se cuenta con la existencia de una herramienta como Protégé, la cual permite acelerar el desarrollo de las ontologías, ya que posee funcionalidades para: (i) representar el diseño de la ontología; (ii) registrar las definiciones en forma de texto, y así generar documentación de manera asistida; y (iii) realizar la ingeniería hacia adelante, generando automáticamente el código en el lenguaje OWL.

B. La Herramienta Protégé

Protégé es quizás la herramienta de ingeniería de ontologías más conocida, es un software de código abierto desarrollado en Stanford University [18]. Fue creado inicialmente como un Sistema de Aprendizaje aplicado a la Salud para traducir datos biomédicos primarios a datos legibles por máquinas para la toma de decisiones [19]. Proporciona una estructura interna denominada "modelo" para la representación de las ontologías y una interfaz para mostrar y manipular el modelo subyacente. El modelo de Protégé se utiliza para representar elementos de la ontología tales como clases, propiedades (slots), características de las propiedades (por ejemplo, facetas y restriciones), así como también instancias. La interfaz gráfica de usuario de Protégé puede utilizarse para crear clases e instancias, y para establecer propiedades de las clases y restricciones sobre las facetas de las propiedades. Protégé presenta al usuario varias funcionalidades para el acceso, la visualización gráfica y para realizar consultas sobre las ontologías. Además, posee funciones para cargar, modificar y guardar las ontologías en formatos diferentes, que incluyen XML, RDF, UML y OWL.

C. Lenguajes de la Web Semántica para la Especificación de Ontologías

La especificación de una ontología para la Web Semántica se sirve de algún lenguaje lógico descriptivo que permite representar el contenido semántico y describir la ontología formalmente bajo una sintaxis basada en la lógica. En la Web Semántica se utilizan principalmente lenguajes tales como XML, XML Schema (XMLS), RDF, RDF Schema (RDFS) y OWL para definir las ontologías. En este trabajo se utiliza el lenguaje OWL para obtener una descripción formal de la ontología para las Manifestaciones Rupestres venezolanas.

OWL fue desarrollado para superar algunas debilidades expresivas del lenguaje RDF y del lenguaje RDFS. OWL amplía la expresividad del RDF y del RDFS con herramientas para describir las relaciones entre las clases, definir las características de las propiedades, las restricciones de cardinalidad y las restricciones sobre los valores de las propiedades. Herramientas tales como Protégé permiten ejecutar un paso de ingeniería hacia adelante, proporcionando una interfaz gráfica amigable para que el usuario pueda especificar el contenido semántico de las ontologías y luego, al final, la herramienta genera el código OWL necesario.

V. Ontología para el Dominio de las Manifestaciones Rupestres Venezolanas

De acuerdo a los formularios que han sido examinados, en este dominio el punto de partida es el modelado de los conceptos de `Yacimiento (Site)` y `Piedra (Rock)`. Es precisamente de esos dos tipos de objeto que están constituidos los hallazgos arqueológicos en materia de Manifestaciones Rupestres. Una característica de un `Yacimiento` es que contiene $N \geq 1$ `Piedras` en cada una de las cuales está presente alguna `Imagen (Figure)` grabada. Las `Imágenes` a su vez poseen una clasificación preestablecida por los expertos del dominio. Otro concepto importante en el dominio que está asociado a `Yacimiento` es la `Manifestación (Find)` y se refiere al tipo de Manifestación Rupestre que está presente en el yacimiento. Las manifestaciones están clasificadas jerárquicamente y forman una taxonomía. Estos conceptos principales del dominio corresponden a entidades o clases en la ontología y buena parte del trabajo del modelado consiste en identificar estos conceptos y representarlos dentro de la ontología. Algunos conceptos ayudan a establecer la relación del `Yacimiento` con su entorno. Por ejemplo, el concepto de `Ubicación (Location)` incluye una clasificación que sirve para documentar el tipo de lugar donde está el `Yacimiento: Cerro (Hill), Valle (Valley), Río (River), Costa (Coast).`

En las Figuras 2, 3 y 4, se muestran las representaciones de la Ontología en Protégé a un alto nivel para los conceptos de `Yacimiento (Site), Manifestación (Find)` e `Imagen (Figure)`, respectivamente.

En la Figura 2, correspondiente a la taxonomía de `Yacimiento`, están presentes las siguientes categorías en el primer nivel: `SurfaceStrata` (Terreno Superficial), `RockWall` (Pared Rocosa), `RockSite` (Yacimiento de Rocas), `Shelter` (Abrigo), `DeepStrata` (Terreno Profundo), `AccidentalCave` (Cueva de Recubrimiento), `Cave` (Cueva), `NaturalDolmen` (Dolmen Natural).

Del mismo modo, en el primer nivel de la taxonomía de `Manifestación`, según la Figura 3, están presentes las categorías: `Geoglyph` (Geoglifo), `RockPainting` (Pintura Rupestre), `Petroglyph` (Petroglifo), `NaturalMythicRock` (Piedra Mítica Natural), `NaturalMythicHill` (Cerro Mítico Natural), `MegalithicMonuments` (Monumentos Megalíticos), `Grindstone` (Amolador o Piedra de Amolar), `Troughs` (Bateas), `CoupledPoints` (Puntos Acoplados), `Dome` (Cúpula), `Mortar` (Mortero o Metate).

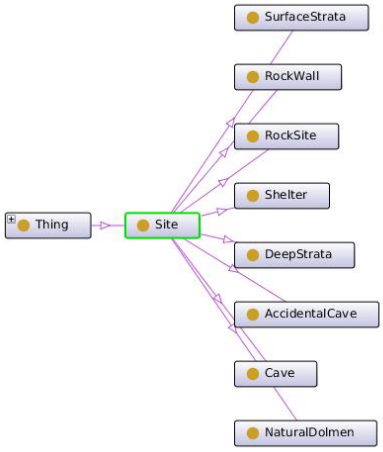


Figura 2: Taxonomía del Concepto $Yacimiento$ $(Site)$

En la Figura 4, se muestran las categorías que permiten clasificar las `imágenes` que se encuentran sobre algunas Manifestaciones Rupestres halladas en el trabajo de campo. éstas son: `TroughsFigure` (Bateas, piedras con cortes de forma rectangular o semiesférica), `ZooAntropmorphFigure` (Imágenes de animales con rasgos humanos) `AntropogeometricFigure` (Figuras humanas, realizadas con componentes geometricos), `ZoogeometricFigure` (Figuras de animales o con sus rasgos, realizadas con elementos geométricos), `CupolaFigure` (Domo, grabados semiesféricos de 2 a 4 cms. de diámetro x 1 o 2 de profundidad), `GrindstonesFigure` (Surcos realizados por abrasión, no necesariamente tienen formas determinadas), `AntropomorphicFigure` (Imagen con forma humana), `CoupledPointsFigure` (Grabados de punteados pequeños semiesféricos), `ZoomorphicFigure` (Imágenes con forma o rasgos de animales), `GeometricFigure` (Figuras con elementos geométricos).

La Figura 5 muestra la correspondencia entre los elementos de la ficha original de ANAR y las categorías de la taxonomía dentro de la ontología para la clase `Manifestación (Find)`. El lado izquierdo de la Figura 5 muestra una parte de un formulario del proyecto ANAR, mientras que el lado derecho muestra parte de la ontología cuyas categorías taxonómicas corresponden a los conceptos que contiene el formulario del ANAR.

La Figura 6 representa las características de un surco de petroglifo según la ficha original de ANAR, correspondiente a la clase `PetroglyphGroove (Petroglifo-Surco)` de la ontología. Estas características se convierten en los atributos (listados en la Tabla I) de la clase `PetroglyphGroove`.

La Figura 7 muestra la representación de las clases `Petroglyph` y `PetrogliphGroove` y la relación que existe entre ambas. En este caso *hasPetroglyphGroove* es un propiedad (*Object property* en OWL) que relaciona ambas clases.

Así, partiendo de la información contenida en las fichas y formularios de ANAR para el registro manual de todas las manifestaciones rupestres en Venezuela y siguiendo la metodología mencionada en la Sección IV, se logró construir

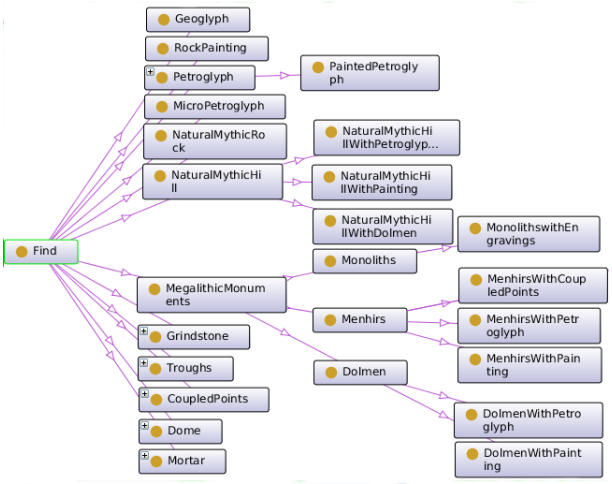


Figura 3: Taxonomía del Concepto *Manifestacion* (*Find*)

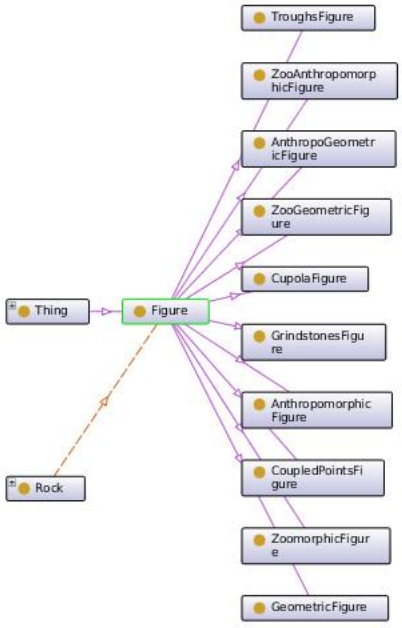


Figura 4: Taxonomía del Concepto *Imagen* (*Figure*)

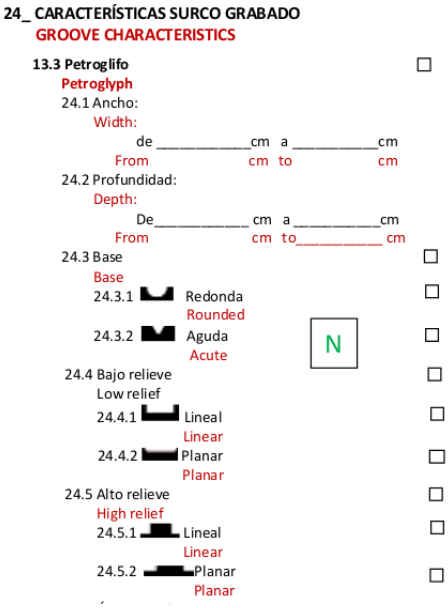


Figura 6: Características de un Surco para un Petroglifo, Correspondiente a la Clase *PetrogliphGroove* en la Ontología

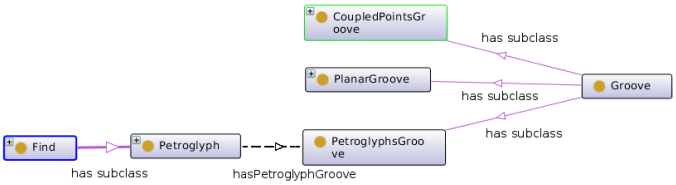


Figura 7: Representación de las Clases *Petroglyph* y *PetrogliphGroove*

la primera versión de la ontología para las Manifestaciones Rupestres venezolanas. La Figura 8 muestra una parte de las clases que conforman la ontología al nivel más alto, así como también algunas de las propiedades de objeto que están presentes entre esas clases.

El rasgo principal del dominio es la complejidad que aparece en la medida en la cual se avanza en la profundidad de las diferentes Manifestaciones Rupestres, pues en Venezuela se encuentran presentes la mayoría de las que se encuentran en diferentes lugares del mundo. Sin embargo, a partir de los instrumentos de recolección de información arquelógica, a través de la colaboración entre personal de sistemas y del área arqueológica y haciendo uso de la documentación disponible, ha sido posible realizar una conceptualización en un primer nivel para representar digitalmente los datos que dispone el ANAR.

El hecho de que Venezuela sea uno de los países en que se encuentran presentes la mayoría de las Manifestaciones Rupestres, hace vislumbrar una ontología completa del dominio que podrá ser fácilmente adoptada para representar las Manifestaciones Rupestres de cualquier otro país de Latino América. La idea es completar, extender y complementar a nivel más detallado esta ontología, no sólo con respecto a las Manifestaciones Rupestres en sí, sino con respecto a

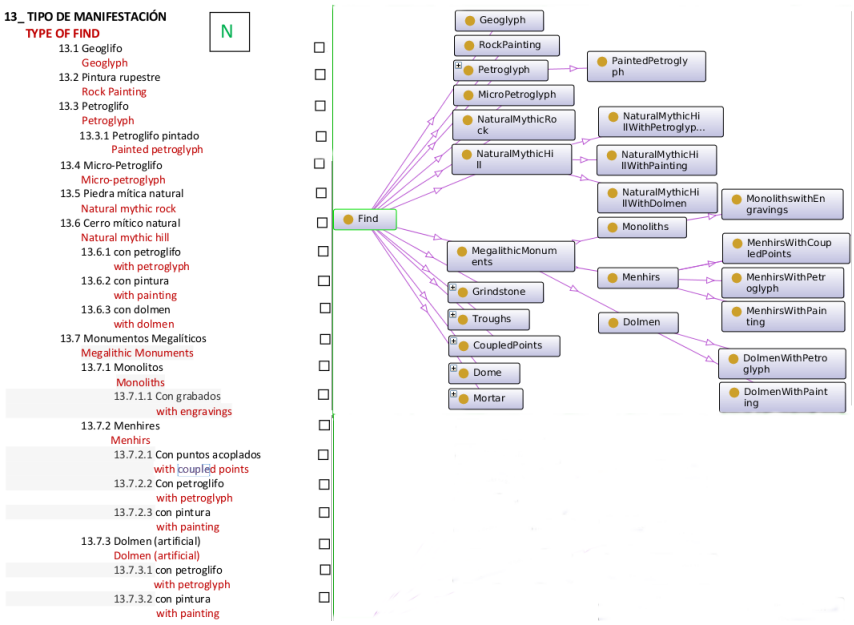


Figura 5: Correspondencia entre las Categorías en las Fichas de ANAR y las Categorías de la Taxonomía del Concepto $Manifestacion\ (Find)$

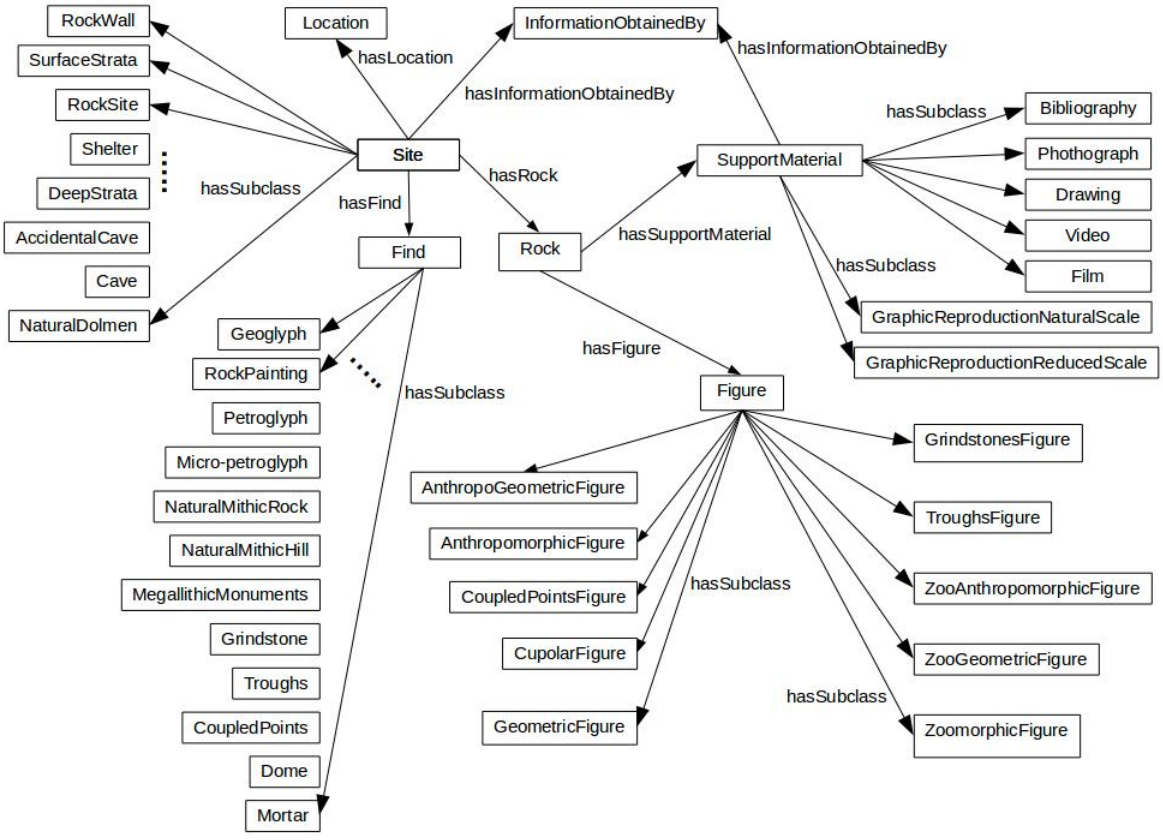


Figura 8: Parte de la Ontología para las Manifestaciones Rupestres

otros conceptos relacionados tales como información de geolocalización, multimedios (fotos, videos, figuras, documentos, entre otros) o roles y usuarios (para permitir el acceso a la información de acuerdo al perfil del usuario – público gen-

Tabla I: Lista de Atributos de la Clase *PetrogliphGroove*

Data Property	Range
widthFrom	float
widthTo	float
depthFrom	float
depthTo	float
base	{"acute","rounded"}
lowRelief	{"linear","planar"}
highRelief	{"linear","planar"}
areasBetweenLines	{"pulished","recessed"}
overlaidEngravings	boolean
recessedEngravings	boolean

eral, turista, estudiante, arqueologo, historiador, antropólogo, científico, entre otros). Esto permitirá construir posteriormente servicios que puedan ofrecer los datos de manera abierta y con la mayor versatilidad posible.

VI. Hacia el Desarrollo de una Plataforma para la Preservación Digital

Además de la ontología para las Manifestaciones Rupestres propuesta, este equipo de trabajo desarrolló un sencillo servicio *endpoint* accesible desde plataformas móviles para acceder a material multimedia de ANAR, como prueba de conceptos y para ejemplificar los beneficios que se pueden obtener de una plataforma tecnológica completa para la preservación digital de las Manifestaciones Rupestres en Venezuela (extensible a toda Latino América). Además, también se propone un diseño preliminar de la plataforma y una estrategia para concretar el objetivo final.

A. Servicio de Acceso a Datos Multimedios de ANAR

Basado en la ontología descrita en la Sección V se anotaron semánticamente un conjunto de datos multimedios provistos por ANAR y se implementó un servicio *endpoint* para acceder a dichos contenidos multimedia. La interfaz propuesta del servicio permite: (i) interrogar al repositorio RDF de datos multimedia de ANAR, sin que el usuario requiera conocer el lenguaje de interrogación (SPARQL); (ii) agregar contenido multimedia a dicho repositorio sin que el usuario deba conocer detalles de la representación de bajo nivel (tuplas RDF); y (iii) desarrollar un mecanismo de acceso al servicio a través de clientes livianos instalados en los dispositivos móviles de los usuarios.

Actualmente el proceso de registro de usuarios es muy sencillo, posteriormente se complementará con la ontología que permita representar perfiles y roles, de forma que se pueda proveer un acceso de acuerdo a perfiles, preferencias e incluso reglas de compartimiento de los usuarios y, además, proteger información importante que no puede mostrarse al público en general. La Figura 9 muestra algunos *screenshots* para este proceso.

Luego del proceso de registro, el usuario podrá acceder al servicio, a través del cliente liviano en su dispositivo móvil, para cargar cualquier contenido multimedia relacionado que desee compartir o consultar contenidos ya existente. Cuando un usuario va a realizar una consulta (por ejemplo, Monolitos

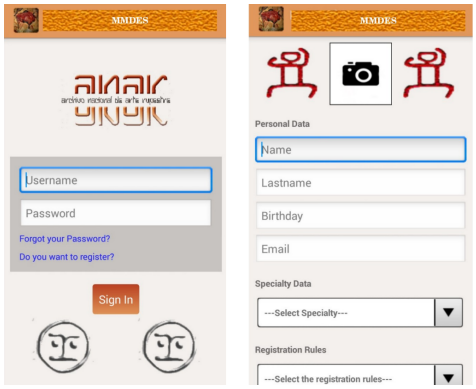


Figura 9: *Screenshots* para el Registro de Usuarios sobre un Móvil

en Venezuela), el cliente móvil pasa un archivo XML con la información del perfil del usuario (por ejemplo, arqueologo) al servicio y devuelve como resultado de la consulta, todos aquellos contenidos que concuerden con la consulta (e.g., fotos de monolitos, videos, información georeferencial, documentos de clasificación). La Figura 10 muestra gráficamente el esquema de consulta y la Figura 11 muestra un *screenshot* después de una consulta.

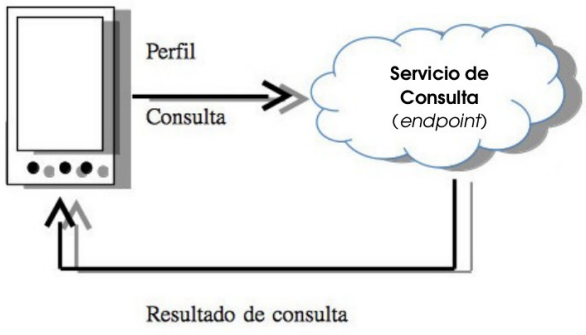


Figura 10: Esquema de Consulta

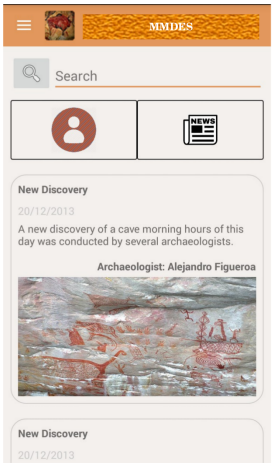


Figura 11: Pantalla para Ejecutar Consultas y Recibir Noticias

Para este servicio, las tuplas RDF son manejadas con el API de Jena RDF [16] y el *framework* de Apache Jena, el

cual está implementado en Java para desarrollar aplicaciones basadas en Web Semántica y datos enlazados (*Linked Data*). El *framework* de Jena ofrece varios APIs para implementar sevicios *endpoints* que manejen y procesen modelos RDFs. Para el acceso local de los datos, el sevicio *endpoint* usa ARQ[1], una máquina de consultas de Jena que soporta el lenguaje SPARQL para RDF.

El cliente liviano para las plataformas móviles fue desarrollado en Java y maneja una pequeña base de datos (desarrollada con SQLite[2]) para mantener información relacionada a los contenidos recuperados y al perfil del usuario.

B. Diseño Preliminar y Estrategia de Desarrollo

La Figura 12 muestra lo que, en primera instancia, debería ser la plataforma para el servicio de datos abiertos y enlazados para las Manifestaciones Rupestres venezolanas. El módulo de Manifestaciones Rupestres recibe contenidos diversos. Los objetos de datos espaciales formulados como tripletas RDF representan los elementos de la toponimia, además de otros hechos geográficos y su relación con la división político territorial venezolana con el propósito de integrar cada elemento de las Manifestaciones Rupestres presente en el repositorio con su referencia espacial. El repositorio admite datos de naturaleza multimedia y multilinge y, para el caso de las imágenes, al igual que el sistema IndianaMAS [10], es capaz de reconocer imágenes de las Manifestaciones Rupestres y localizar los datos mediante un sistema de reconocimiento posiblemente basado en agentes inteligentes, esto puede ayudar a realizar anotaciones semánticas sobre las imágenes de manera semiautomática, ya que una parte de las anotaciones deberán hacerse con asistencia de los expertos del dominio. La arquitectura de este sistema utiliza en lo posible la base de conocimiento de cada uno de los otros subdominios del patrimonio cultural con el propósito de ofrecer un servicio de datos abiertos y enlazados integrado con todo el dominio.

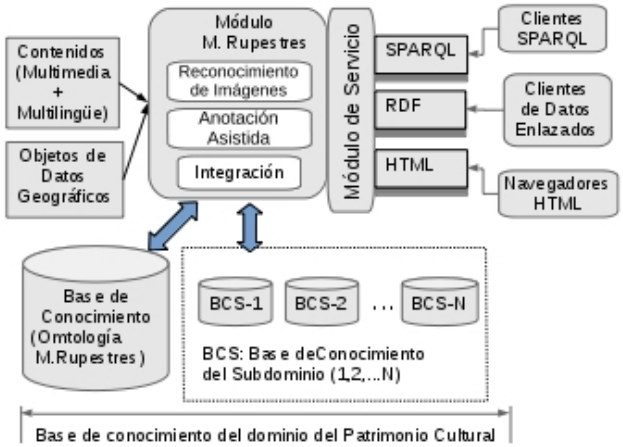


Figura 12: Diseño Preliminar de la Plataforma de Preservación Digital de las Manifestaciones Rupestres

[1]ARQ: A SPARQL Processor for Jena, The Apache Software Foundation, https://jena.apache.org/documentation/query

[2]https://sqlite.org

Partiendo de la ontología propuesta y del diseño preliminar, la estrategia para desarrollar la plataforma para la preservación digital de las Manifestaciones Rupestres consta de las siguientes tareas:

- Extender las características de la ontología actual hasta completar toda la especificación para el dominio de las Manifestaciones Rupestres venezolanas.
- Una vez completada la especificación de toda la semántica del dominio, realizar la implementación mediante alguna plataforma de servicio como Jena o Virtuoso. Hará falta la preparación de un conjunto de datos, con base a la información manejada por el ANAR, y luego se deberá poblar la ontología utilizando este conjunto de datos. Todo esto con el fin de construir un *endpoint* que permita servir los datos de manera abierta y enlazada.
- Una tarea que se debe llevar a cabo, luego de la serialización y la implementación del conjunto de datos es la dereferenciación de los URIs, que son los identificadores primarios de las entidades presentes en el conjunto de datos. Esto es necesario para garantizar que el servicio cumpla con los estándares de la Web Semántica, específicamente en lo relacionado con las cinco estrellas enunciadas por Berners-Lee [20].
- Estudiar en profundidad los desarrollos de otros modelos basados en ontologías desarrollados específicamente para el dominio de las Manifestaciones Rupestres en general. Varios esfuerzos mencionados en la Sección de Trabajos Relacionados (Sección II) están en proceso de consolidación en el presente. El dominio de las Manifestaciones Rupestres no es generalizable fácilmente, algunas categorías taxonómicas pueden variar de una región a otra dadas las diferencias culturales, y en las ideas y conceptos que están presentes en el origen de este Patrimonio Cultural, pero el desarrollo de una ontología propia permite conocer las particularidades y las generalidades que poseen las Manifestaciones Rupestres venezolanas.

Lo más importante para el futuro es asegurar el desarrollo de una plataforma tecnológica completa y coherente para la preservación digital de las Manifestaciones Rupestres. La plataforma apoyará a los expertos del dominio en la creación de un repositorio, que puede convertirse en un referente a nivel nacional y latinoamericano como un conjunto detallado de datos abiertos y enlazados acerca de las Manifestaciones Rupestres venezolanas, y sus posibles interpretaciones con vocabulario propio y enlazado con otros vocabularios presentes en la Web Semántica.

Esto también servirá para promover el conocimiento y la preservación de la riqueza cultural, haciendo que la información cultural esté al alcance de todos a través de Internet y contribuir a la preservación de este patrimonio para las generaciones futuras. Con este fin, la plataforma permitirá la preservación de todo tipo de datos disponibles acerca de las Manifestaciones Rupestres, que incluye documentos en texto, multimedios (imágenes, videos, fotos, etc.), información de geolocalización, entre otros. Además, proporcionará los medios para organizar y estructurar estos datos en una herramienta de

colaboración.

VII. Conclusiones

Este trabajo presenta una conceptualización en un primer nivel para representar digitalmente los datos del ANAR-Archivo Nacional de Arte Rupestre venezolano a través de una ontología de dominio. Se realizó una prueba de conceptos desarrollando un servicio *endpoint* para consultar contenidos multimedia anotados con la ontología propuesta y se propone un diseño preliminar de la plataforma para la preservación digital de las Manifestaciones Rupestres, una estrategia de desarrollo. A pesar de que existen trabajos realizados bastante recientes en el subdominio del patrimonio cultural denominado Manifestaciones Rupestres, no existen aún ontologías definitivas para representar todas las Manifestaciones Rupestres a nivel mundial. La base de datos global del Arte Rupestre, por ejemplo, todavía está bajo prueba dentro de los límites de las Manifestaciones del Arte Rupestre australiano. El modelado de las Manifestaciones Rupestres venezolanas tiene sentido debido a que permitirá conocer las particularidades y las generalidades de los elementos de información que se requieren en el diseño, lo cual será necesario a la hora de integrar y anotar los datos para su uso a escala global internacional. El hecho de que Venezuela sea uno de los países en el que se encuentran presentes todas las Manifestaciones Rupestres posibles, hace vislumbrar una ontología del dominio amplia y completa que podrá ser fácilmente adoptada para representar las Manifestaciones Rupestres de cualquier otro país de Latinoamérica.

El equipo de trabajo dará continuidad a este proyecto hasta completar la plataforma de preservación digital de de las Manifestaciones Rupestres venezolanas.

Referencias

[1] FUNDABITAT, *ANAR- Archivo Nacional de Arte Rupestre*, 2006, http://www.anar.org.ve.

[2] R. D. Valencia and J. Sujo, *El Diseño de los Petroglifos Venezolanos*. Caracas, Venezuela: Fundación Pampero, 1987.

[3] M. Uschold and M. King, *Towards a Methodology for Building Ontologies*, in Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI, 1995.

[4] Stanford University, *Protégé*, 2016, http://protege.stanford.edu.

[5] W3C, *Owl- Web Ontology Language*, 2016, https://www.w3.org/OWL.

[6] R. D. Valencia, *Las Manifestaciones Rupestres y la Escuela*, ANAR, Caracas, Venezuela, Enero 2012.

[7] L. Papaleo, G. Quercini, V. Mascardi, M. Ancona, A. Traverso, and H. de Lumley, *Agents and Ontologies for Understanding and Preserving the Rock Art of Mount Bego*, in The 3rd International Conference on Agents and Artificial Intelligence (ICAART), vol. 2 - Agents, Rome, Italy, January 2011, pp. 288-295.

[8] Le Orme dell'Uomo, *Mount Bego Petroglyphs - Rock Art in the Alps*, Rupestre.net, 2016, http://www.rupestre.net/alps/bego.html.

[9] Archeocamuni, *Intro to Valcamonica Rock Art*, Archeocamuni.it, 2016, http://www.archeocamuni.it/rock_art_en.html.

[10] A. Locoro, V. Mascardi, D. Briola, M. Martelli, M. Ancona, V. Deufemia, L. Paolino, G. Tortora, G. Polese, and R. Francese, *The Indiana MAS Project: Goals and Preliminary Results*, in The 13th Workshop on Objects and Agents, vol. CEUR- 892, Milano, Italy, September 2012. http://ceur-ws.org/Vol-892/paper10.pdf.

[11] V. Mascardi, V. Deufemia, D. Malafronte, A. Ricciarelli, N. Bianchi, and H. de Lumley, *Rock Art Interpretation within Indiana MAS*, Springer Berlin Heidelberg, 2012, pp. 271-281. http://dx.doi.org/10.1007/978-3-642-30947-2_31

[12] D. Briola, V. Deufemia, V. Mascardi, L. Paolino, and N. Bianchi, *Ontology-Driven Processing and Management of Digital Rock Art Objects in IndianaMAS*. Limassol, Cyprus: Springer International Publishing, November 2014, pp. 217-227. http://dx.doi.org/10.1007/978-3-319-13695-0_21.

[13] R. A. Haubt, *The Global Rock Art Database: Developing a Rock Art Reference Model for the RADB System using the CIDOC CRM and Australian Heritage Examples*, in Annals of the Photogrammetry, Remote Sensing ans Spatial Information Sciences. 25th International CIPA Symposium, vol. II-5/W3, 2015, pp. 89-96.

[14] R. Stanley, H. Astudillo, V. Codocedo, and A. Napoli, *A Conceptual-kdd Approach and its Application to Cultural Heritage*, in CLA, ser. CEUR Workshop Proceedings, vol. 1062. CEUR-WS.org, 2013, pp. 163-174 http://dblp.unitrier.de/db/conf/cla/cla2013.html#StanleyACN13

[15] R. D. Valencia, N. Manrique, J. González, and Y. Cardinale, *Manifestaciones Rupestres Venezolanas: La Comunicación, del Pasado al Presente en Realidad Aumentada*, en CoNCISa 2013, Naiguatá, Venezuela, Octubre 2013, pp. 85-90.

[16] The Apache Software Foundation, *Apache Jena*, 2016. https://jena.apache.org

[17] OpenLink Software, *Virtuoso Universal Server*, 2016. http://virtuoso.openlinksw.com

[18] M.-A. Sicilia, *Handbook of Metadata, Semantics and Ontologies*. Singapore: World Scientific Publishing Co, 2014.

[19] L. F. Sikos, *Mastering Structured Data on the Semantic Web*. New York: Apress, 2015, distributed by Springer Science+Business Media.

[20] T. Berners-Lee, *Linked Data - Design Issues*, 2016. https://www.w3.org/DesignIssues/LinkedData.html

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 68-80, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Cálculo de Precondiciones más Débiles

Federico Flaviani[1]
fflaviani@usb.ve

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

Resumen: El cálculo comenzó a desarrollarse en el siglo XVII con la intención de conseguir técnicas estándar para resolver problemas referentes a derivadas, integrales y ecuaciones diferenciales. Gracias al desarrollo de esta ciencia y la estandarización de las técnicas de cálculo integral, hoy en día es posible que un estudiante de primer año de universidad pueda aprender a resolver una cantidad enorme de integrales en poco tiempo y aunque el estudiante tenga la sensación de que esta usando el sentido común para hacer dichos cálculos, realmente puede hacerlo porque esta siguiendo un algoritmo de resolución, muy complejo, pero algoritmo en fin. Este algoritmo es el que hace posible que hoy en día, aplicaciones como Mapple o Mathematica puedan resolver simbólicamente integrales y ecuaciones diferenciales. Es decir, el mayor porcentaje del crédito por el éxito de estas aplicaciones, se debe a la existencia de los algoritmos de resolución del cálculo, que han sido desarrollados durante cuatro siglos. Por otro lado en el mundo de la demostración formal de programas, también existen software como en [1][2][3] que intentan hacer cómputos simbólicos para la corrección de programas. Estas aplicaciones para la corrección de programas, no usan un algoritmo de cuatro siglo de desarrollo que estandarizan técnicas del cálculo de los invariantes y obligaciones de prueba, por lo que tienen una desventaja natural con respecto a las aplicaciones de resolución simbólica para el cálculo integral o diferencial.

Por lo argumentado anteriormente se propone que para lograr un avance significativo en las aplicaciones para la corrección de programas, es conveniente crear ahora, en la ciencia de la matemática, un cálculo (al igual como en su tiempo se creó el cálculo integral), para estandarizar los métodos que permiten computar invariantes, obligaciones de prueba y precondiciones más débiles. Este articulo muestra algunos teoremas que tienen la intención de dar un pequeño paso en el desarrollo de este cálculo pero para computar precondiciones más débiles.

Palabras Clave: Precondición más Débil; Cálculo; Corrección Formal de Programas; GCL; Inducción.

Abstract: The calculus began to develop in the seventeenth century with the intention of obtaining standard techniques for solve problems related to derivatives, integrals and differential equations. Thanks to the development of this science and the standardization of the techniques of integral calculus, today it is possible that a first-year university student can learn to solve a huge amount of integrals in a short time. Even if the student has the sensation that he is using common sense to do such calculations, he can actually do it because he is following a Algorith resolution, very complex, but algorithm. This algorithm is what makes it possible today, that applications such as Mapple or Mathematica can symbolically solve integrals and differential equations. That is to say, the highest percentage of credit for the success of these applications is due to the existence of resolution algorithms of calculus, which have been developed over four centuries. On the other hand, in the world of formal verification of programs there are also software like [1][2][3] which try to make symbolic computation for formal verification of programs. These applications for formal verification do not use a four-century algorithm of development that standardize techniques for calculating invariants, so it have a natural disadvantage with respect to applications of symbolic resolution for integral or differential calculus.

In order to achieve a significant advance in the applications for the formal verification of programs, it is convenient to create, in the science of mathematics, a calculus (just as in its time the integral calculus was created) for standardize the methods that allow computing invariants, and weakest preconditions. This article shows some theorems that are intended to take a small step in the development of this calculus but to compute weakest preconditions.

Keywords: Weakest Precondition; Calculus; Formal Verification; GCL; Induction.

I. Introducción

Todos los algoritmos planteados en este trabajo serán escritos en pseudolenguaje GCL (Guarded Command Language) [4], que es un pseudolenguaje definido por Dijkstra, que admite la escritura de algoritmos no determinísticos y su diseño, admite una lógica de Hoare y fórmulas para precondiciones más débiles, relativamente simples, que facilitan la actividad de corrección de un programa.

Si $B_0, \ldots, B_n$ son expresiones booleanas del lenguaje GCL y definiendo DG como una abreviación de $domain(B_0, \ldots, B_n)$ (predicado que expresa que todas las expresiones están bien definidas [5]), tenemos que la lógica de Hore [6] se basa en las siguientes reglas de inferencia:

$$\{A\}SKIP\{A\}$$

$$\{domain(\overline{E}) \wedge B[\overline{x} := \overline{E}]\}\overline{x} := \overline{E}\{B\}$$

$$\frac{\{A\}S_0\{B\} \qquad \{B\}S_1\{C\}}{\{A\}S_0; S_1\{C\}}$$

$$\frac{A \Rightarrow DG \wedge (B_0 \vee .. \vee B_n) \qquad \{A \wedge B_0\}S_0\{B\} .. \qquad ..\{A \wedge B_n\}S_n\{B\}}{\{A\}if\ B_0 \rightarrow S_0[]..[]B_n \rightarrow S_n fi\{B\}}$$

$$\frac{I \Rightarrow domain(B_0) \qquad \{I \wedge B_0\}S_0\{I\}}{\{I\}do\ B_0 \rightarrow S_0\{I \wedge \neg B_0\}}$$

$$\frac{A \Rightarrow A' \qquad \{A'\}S_0\{B'\} \qquad B' \Rightarrow B}{\{A\}S_0\{B\}}$$

Donde $S_0, \ldots, S_n$ son instrucciones A, A', B, B' y C son predicados, I es un predicado que lleva el nombre de "invariante del ciclo" y $\overline{E}$ es una lista de expresiones, que una a una, son del mismo tipo que la lista de variables $\overline{x}$ (la barra superior tanto en E como en x, es una notación que indica que se tiene una lista de expresiones y variables respectivamente).

De las reglas de inferencia de arriba se concluye que una derivación en esta lógica tiene forma de árbol. Adicionalmente una demostración de que un algoritmo es correcto usando la lógica de Hoare, no garantiza la terminación del programa, por lo que el árbol de derivación usando las reglas de Hoare, debe ir acompañado de un argumento de función de cota en cada ciclo Do, si se quiere tener una demostración completa de que el algoritmo es correcto con respecto a la especificación.

Por otro lado la lógica de Dijkstra [4] para la corrección de programas se basa en el transformador de predicados wp (weakest precondition), que es básicamente una función sintáctica de dos variables que devuelve de forma simbólica la precondición más débil de una instrucción $inst$ dado una postcondición $Post$ (usando la notación clásica de funciones de dos variables, la notación $wp(inst, Post)$ se refiere al resultado de aplicarle a la función wp, los argumentos $inst$ y $Post$, este resultado es la precondición más débil, simbolicamente hablando, de la instrucción $inst$ con la instrucción $Post$). El uso sucesivo de wp permite ir calculando precondiciones más débiles entre instrucción e instrucción, desde el final del programa hasta el inicio como se muestra en la Figura 1.

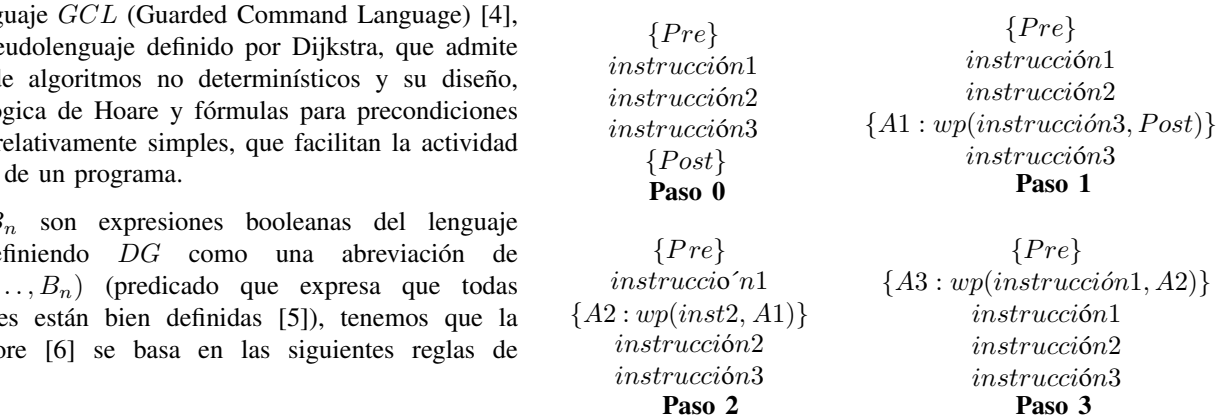


Figura 1: Uso Sucesivo de wp

Usando el transformador wp de Dijkstra sucesivamente, se puede calcular la precondición más débil de todo el programa, de modo que para demostrar la corrección de un algoritmo, basta con demostrar que la precondición Pre implica la precondicción más débil calculada (En el caso del ejemplo de la figura se debe demostrar $Pre \Rightarrow A3$)

La corrección de programas usando la lógica de Dijkstra garantiza terminación y no necesita el argumento de función de cota en cada iteración, por lo que induce un método limpio para decidir la corrección de un programa.

Calcular una precondición más débil usando el transformador wp para cualquier postcondición e instrucción es fundamental, porque permite ir consiguiendo aserciones que permitan subir entre instrucción e instrucción como en la figura. Para lograr esto Dijkstra en [4] definió las reglas que definen la función de transformación sintáctica wp de la siguiente manera:

- $wp(SKIP, Post) := Post$
- $wp(y_{i_1}, \ldots, y_{i_k} := Exp_1, \ldots, Exp_k, Post) :=$ $domain(Exp_1, \ldots, Exp_k) \wedge$ $Post[y_{i_1}, \ldots, y_{i_k} := Exp_1, \ldots, Exp_k]$
- $wp(S_0; S_1, Post) := wp(S_0, wp(S_1, Post))$
- $wp(IF, Post) := domain(B_0, \ldots, B_n) \wedge$ $(B_0 \vee \cdots \vee B_n) \wedge (B_0 \Rightarrow wp(S_0, Post)) \wedge \ldots$ $\wedge (B_n \Rightarrow wp(S_n, Post))$
- $wp(DO, Post) := (\exists k | k \geq 0 : H_k(Post))$

  en donde $H_k(Post)$ es un predicado que satisface las ecuaciones:

$$H_0(Post) \equiv domain(BB) \wedge \neg(B_0 \vee \cdots \vee B_n) \wedge Post$$

$$H_k(Post) \equiv H_0(Post) \vee wp(IF, H_{k-1}(Post))$$

  para $k \geq 1$

Lamentablemente la fórmula que define a wp en general para ciclos DO, está escrita en lógica de segundo orden y por lo tanto no es aplicable directamente. Tampoco es útil tener aserciones escritas en lógica de segundo orden,

ya que si una aserción de segundo orden fuese la post-condición de una instrucción de asignación $y_{i_1}, \ldots, y_{i_k} := Exp_1, \ldots, Exp_k$, entonces no se pudiera aplicar la segunda de las reglas listadas anteriormente, porque el operador de sustitución $[y_{i_1}, \ldots, y_{i_k} := Exp_1, \ldots, Exp_k]$ no esta definido para fórmulas escritas en segundo orden.

Esta limitante de la descripción de wp de Dijkstra lo convierte, en el caso de programas que contienen ciclos DO, en una herramienta teórica sin pragmática alguna. Sin embargo, esta demostrado en [7] que para cada algoritmo en particular, con una postcondición particular, siempre existe un predicado escrito en lógica de primer orden equivalente a la definición de wp para el caso en cuestión. El único problema es que la demostración que se encuentra en [7] no es constructiva y no induce un algoritmo para computar dicho predicado.

La idea que se propone en este artículo es la de dar técnicas de cálculo que permitan construir una fórmula escrita en lógica de primer orden, que sea equivalente a la fórmula de Dijkstra para los ciclos DO, de modo tal de rescatar la idea original del transformador de predicado wp y convertirla en una herramienta práctica para poder decidir si un programa es correcto. Al igual como en el cálculo integral, en donde cada técnica de integración depende del tipo de la integral (si es polinómica, trigonométrica, exponencial, o multiplicación y división de las combinaciones de las tres categorías anteriores) las técnicas que se proponen dependerán del tipo de ciclo.

El resultado será el de caracterizar algunos tipos de ciclos en los que se pueda obtener una plantilla de predicado equivalente a wp en lógica de primer orden, de manera que la corrección del algoritmo consistirá simplemente en la aplicación de la plantilla instanciada al caso particular que se encuentre (sumándole inevitablemente algún toque mínimo del sentido común humano).

A continuación se presentan cuatro secciones de las cuales, en la primera de ellas se hace un repaso de las fórmulas de segundo orden que propuso Dijkstra para definir wp en ciclos DO en general. Luego se caracterizan que fórmulas de primer orden se deben usar para calcular wp en ciclos for. Seguidamente se presentan ejemplos del uso de los teoremas presentados y por último se da una sección de teoremas complementarios para el cálculo de precondiciones más débiles.

II. wp PARA LA INSTRUCCIÓN Do

El lenguaje GCL de Dijkstra fue diseñado para poder hacer algoritmos no determinísticos. Tanto la instrucción condicional como la de iteración tienen versiones no determísticas en GCL, la versión no determinística de la instrucción de iteración se denota DO, mientras la versión determinística se denota Do. La definición de wp que se listó anteriormente para la instrucción de iteración corresponde a la versión no determinística DO de la misma. Se puede deducir una fórmula de wp para la instrucción Do a partir de la anterior, sin embargo en [8] se encuentra dicha fórmula y es:

$$(\exists k | k \geq 0 : H_k(Post))$$

donde $H_k(Post)$ es un predicado definido recursivamente como

$$H_0(Post) := domain(B_0) \wedge \neg B_0 \wedge Post$$

$$H_k(Post) :=$$

$$H_0(Post) \vee (domain(B_0) \wedge B_0 \wedge wp(S_0, H_{k-1}(Post)))$$

$$para\ k \geq 1$$

La fórmula anterior está escrita en lógica de segundo orden, pero es conocido (ver por ejemplo [7]) que si restringimos el lenguaje de las aserciones a cierto tipo de lenguaje de primer orden, es siempre posible que para cada instancia particular de S_0, B_0 y $Post$, se puede escribir la fórmula de wp del párrafo anterior, de una forma equivalente usando estos lenguajes de primer orden.

Este resultado sigue siendo cierto si el lenguaje que se usa para escribir las aserciones es el de la teoría de conjuntos de Zermelo-Fraenkel-Skolem. A continuación se muestra una demostración.

Teorema 1. *Si el lenguaje que se usa para escribir los predicados de las aserciones en GCL, es el lenguaje de primer orden de la teoría de conjuntos de Zermelo-Frankel-Skolem, entonces por cada caso particular de predicado $Post$, de expresión B_0 e instrucción S_0, existe una fórmula escrita en el lenguaje de primer orden de la teoría de conjuntos de Zermelo-Framkel-Skolem, que es equivalente a $wp(do\ B_0 \rightarrow S_0\ od, Post)$*

Demostración: Se considerará la semántica denotacional de GCL que se propuso en [8].

Sea S_0 una instrucción de GCL, entonces se denota como $\mathcal{C}[\![S_0]\!]$ a la relación del espacio de estados al espacio de estados del programa, que corresponde a la interpretación con respecto a la semántica denotacional de [8], de la instrucción S_0 (en [8] se usó la notación R_{S_0} en lugar de $\mathcal{C}[\![S_0]\!]$).

Se definen recursivamente las siguientes instrucciones

$$If := if B_0 \rightarrow S_0 [] \neg B_0 \rightarrow SKIP fi$$

$$Do_0 := if \neg B_0 \rightarrow SKIP fi$$

$$Do_{k+1} := If; Do_k.$$

Como la interpretación de una secuenciación de instrucciones es la composición de las interpretaciones, entonces la interpretación de la instrucción Do_k satisface la siguiente recurrencia:

$$\mathcal{C}[\![Do_0]\!] := \mathcal{C}[\![if \neg B_0 \rightarrow SKIP fi]\!]$$

$$\mathcal{C}[\![Do_{k+1}]\!] := \mathcal{C}[\![Do_k]\!] \circ \mathcal{C}[\![If]\!]$$

Por otro lado la fórmula definida por

$$\varphi(k, F, y) :=$$

$$(k = 0 \wedge y = \mathcal{C}[\![Do_0]\!]) \vee$$

$$(k \neq 0 \wedge k \in \omega \wedge esFuncion(F) \wedge y = F(k-1) \circ \mathcal{C}[\![If]\!]) \vee$$

$$(\neg(k = 0 \vee (k \neq 0 \wedge k \in \omega \wedge esFuncion(F))) \wedge y = F)$$

satisface que:

$$(\forall k, F| : (\exists! y| : \varphi(k, F, y))).$$

Si se define a $G(k, F)$ como el único y que satisface $\varphi(k, F, y)$, entonces por el teorema de recursión transfinita [9] aplicado a los ordinales finitos ω, se tiene que existe una formula ψ que satisface que:

1) $(\forall k| : (\exists! y| : \psi(k, y)))$, es decir ψ define una función F tal que $\psi(k, F(k))$
2) $(\forall k|k \in \omega| : F(k) = G(k, F \restriction_{k-1}))$

Como $G(k, F)$ en notación de llaves es la función a trozos

$$G(k,F) = \begin{cases} \mathcal{C}[\![Do_0]\!] & si \quad k = 0 \\ F(k-1) \circ \mathcal{C}[\![If]\!] & si \quad k \in \omega \ y \\ & \quad esFuncion(F) \\ F & sino \end{cases}$$

entonces la fórmula

$$(\forall k|k \in \omega| : F(k) = G(k, F \restriction_{k-1}))$$

es equivalente a la recurrencia que define a $\mathcal{C}[\![Do_k]\!]$ arriba y por lo tanto, $F(k)$ debe ser igual a $\mathcal{C}[\![Do_k]\!]$. Por esta razón, como se tiene que F es una función tal que $F(k)$ es el único valor en el que $\psi(k, F(k))$ es verdad, entonces cuando el predicado

$$\psi(k, R)$$

sea cierto, debe ocurrir que $R = \mathcal{C}[\![Do_k]\!]$.

Por otro lado usando las notaciones de [8], las cuales son: $\vec{x}$ para indicar el vector de constantes y variables declaradas en un algoritmo, Esp para indicar el espacio de estados del algoritmo y $Rgo_{\overline{Y}}$ para referirse al conjunto $\{\vec{x} \in Esp|Post(\vec{x}, \overline{Y})\}$, se demostró que

$$(\exists k|k \geq 0 : \mathcal{C}[\![Do_k]\!](\{\vec{x}\}) \subseteq Rgo_{\overline{Y}})$$

es un predicado equivalente a $wp(do\ B_0 \rightarrow S_0\ od, Post)$. Sin embargo, usando el predicado ψ se puede reescribir la fórmula anterior como

$$(\exists k|k \geq 0 : \psi(k, R) \wedge R(\{\vec{x}\}) \subseteq Rgo_{\overline{Y}})$$

la cual está escrita en el lenguaje de primer orden de la teoría de conjuntos. ■

Según el teorema anterior, si se usa el lenguaje de la teoría de conjuntos de Zermelo-Frankel-Skolem para escribir las aserciones, entonces se tiene garantía de que toda precondición más débil se puede escribir dentro del lenguaje. El siguiente corolario habla sobre algunos tipos de lenguajes, de la lógica de primer orden, que son convenientes para escribir las aserciones de los algoritmos.

Corolario 1. *Sea L un lenguaje de la lógica de primer orden para escribir predicados en las aserciones de GCL. Si L es un lenguaje tal que, por cada fórmula en L, existe una fórmula equivalente en el lenguaje de la teoría de conjuntos de Zermelo-Frankel-Skolem y por cada fórmula en la teoría de conjuntos de Zermelo-Frankel-Skolem, existe una fórmula equivalente en L, entonces por cada caso particular de predicado $Post$, de expresión B_0 e instrucción S_0, existe una fórmula escrita en L, que es equivalente a $wp(do\ B_0 \rightarrow S_0\ od, Post)$ y por lo tanto el transformador de predicados wp esta bien definido en todo L*

El corolario y teorema anterior sugieren que con el lenguaje adecuado, siempre es posible encontrar un predicado de primer orden para la precondición más débil de un ciclo, sin embargo como la demostración no es constructiva, no se tiene un procedimiento para obtener dicho predicado. El objetivo de este trabajo, es el de mostrar un método que permite conseguir la precondición más débil (escrita en lógica de primer orden) de cierto tipos de ciclos.

En las siguientes secciones usaremos las siguientes propiedades del transformador de predicado wp:

- $wp(S_0, False) \equiv False$
- $wp(S_0, Post_1 \wedge Post_2) \equiv wp(S_0, Post_1) \wedge wp(S_0, Post_2)$
- Si $Post_1 \Rightarrow Post_2$ entonces $wp(S_0, Post_1) \Rightarrow wp(S_0, Post_2)$
- Si S_0 es una instrucción determinística, entonces $wp(S_0, Post_1 \vee Post_2) \equiv wp(S_0, Post_1) \vee wp(S_0, Post_2)$

Adicionalmente para demostrar los teoremas que vienen a continuación vamos a usar dos propiedades del transformador de predicados wp que se deducen de la semántica denotacional de GCL.

Lema 1. *Sean $\overline{x}$ y $\overline{y}$ la lista de constantes y variables declaradas en un algoritmo respectivamente y $\overline{Y}$ una lista de variables de especificación. Sea S una instrucción que no modifica los valores de las variables $y_{i_1}, \ldots, y_{i_k}$ de la lista $\overline{y}$. Sean $P(\overline{x}, y_{i_1}, \ldots, y_{i_k}, \overline{Y})$ y $Q(\overline{x}, \overline{y}, \overline{Y})$ predicados donde $P(\overline{x}, y_{i_1}, \ldots, y_{i_k}, \overline{Y})$ sólo depende de $\overline{x}, y_{i_1}, \ldots, y_{i_k}$, entonces*

$$wp(S, P(\overline{x}, y_{i_1}, \ldots, y_{i_k}, \overline{Y}) \wedge Q(\overline{x}, \overline{y}, \overline{Y})) \equiv$$

$$P(\overline{x}, y_{i_1}, \ldots, y_{i_k}, \overline{Y}) \wedge wp(S, Q(\overline{x}, \overline{y}, \overline{Y}))$$

y

$$wp(S, P(\overline{x}, y_{i_1}, \ldots, y_{i_k}, \overline{Y}) \vee Q(\overline{x}, \overline{y}, \overline{Y})) \equiv$$

$$soporte(S) \wedge (P(\overline{x}, y_{i_1}, \ldots, y_{i_k}, \overline{Y}) \vee wp(S, Q(\overline{x}, \overline{y}, \overline{Y})))$$

donde $soporte(S)$ es un predicado tal que un estado lo satisface si y sólo si la instrucción S no aborta al ser ejecutado en dicho estado.

Demostración: En [8] se demostró que la precondición más débil de una instrucción S y la postcondición $Post$ es equivalente a

$$(\overline{x}, \overline{y}) \in supp(\mathcal{C}[\![S]\!]) \wedge$$

$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x}, \overline{y})\}) \Rightarrow$$

$$(\exists \overline{y'}|y = (\overline{x}, \overline{y'}) : Post(\overline{x}, \overline{y'}, \overline{Y})))$$

donde $\mathcal{C}$ es la función de interpretación, que interpreta una instrucción S en una relación $\mathcal{C}[\![S]\!]$ del espacio de estados Esp a el espacio de estados Esp y donde:

$$supp(\mathcal{C}[\![S]\!]) = \{(\overline{x},\overline{y}) \in Esp | abort \notin \mathcal{C}[\![S]\!]\{(\overline{x},\overline{y})\}\}.$$

Como la interpretación $\mathcal{C}[\![S]\!]$ de la instrucción S no modifica las variables $y_{i_1},\ldots,y_{i_k}$, entonces el predicado anterior es equivalente a:

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists y'_1,\ldots,y'_{i_1-1},y'_{i_1+1},\ldots,y'_{i_k-1},y'_{i_k+1},\ldots,y'_m|$$
$$y = (\overline{x},y'_1,\ldots,y'_{i_1-1},y_{i_1},y'_{i_1+1},\ldots,y'_{i_k-1},y_{i_k},y'_{i_k+1},\ldots,y'_m) :$$
$$Post(\overline{x},y'_1,\ldots,y_{i_1},y'_{i_1+1},\ldots,y_{i_k},y'_{i_k+1},\ldots,y'_m,\overline{Y})))$$

Si la postcondición $Post$ es $P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})op\ Q(\overline{x},\overline{y},\overline{Y})$ con $op \in \{\wedge,\vee\}$, entonces la precondición más débil de la intrucción S con esta postcondición es equivalente a:

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists y'_1,\ldots,y'_{i_1-1},y'_{i_1+1},\ldots,y'_{i_k-1},y'_{i_k+1},\ldots,y'_m|$$
$$y = (\overline{x},y'_1,\ldots,y'_{i_1-1},y_{i_1},y'_{i_1+1},\ldots,y'_{i_k-1},y_{i_k},y'_{i_k+1},\ldots,y'_m) :$$
$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})op$$
$$Q(\overline{x},y'_1,\ldots,y_{i_1},y'_{i_1+1},\ldots,y_{i_k},y'_{i_k+1},\ldots,y'_m,\overline{Y})))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})op$$
$$(\exists y'_1,\ldots,y'_{i_1-1},y'_{i_1+1},\ldots,y'_{i_k-1},y'_{i_k+1},\ldots,y'_m|$$
$$y = (\overline{x},y'_1,\ldots,y'_{i_1-1},y_{i_1},y'_{i_1+1},\ldots,y'_{i_k-1},y_{i_k},y'_{i_k+1},\ldots,y'_m) :$$
$$Q(\overline{x},y'_1,\ldots,y'_{i_1-1},y_{i_1},y'_{i_1+1},\ldots,y'_{i_k-1},y_{i_k},y'_{i_k+1},\ldots,y'_m,\overline{Y}))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})op(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

Si $op = \vee$, entonces:

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \notin \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\})\vee$$
$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}) \vee (\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})\vee(\forall y| : y \notin \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\})\vee$$
$$(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y}))))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})\vee(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y}))))$$

$\equiv <\ p \wedge (q \vee r) \equiv p \wedge (q \vee (p \wedge r))\ >$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!]) \wedge (P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})\vee$$
$$((\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))))$$

$\equiv$

$$soporte(S) \wedge (P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}) \vee wp(S,Q(\overline{x},\overline{y},\overline{Y})))$$

Por otro lado si $op = \wedge$, entonces la precondición más débil sería:

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}) \wedge (\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : (y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}))$$
$$\wedge(y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y}))))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}))$$
$$\wedge(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y| : y \notin \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\})\vee P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}))$$
$$\wedge(\forall y| : y \in \mathcal{C}[\![S]\!] \restriction_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists\overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

$\equiv$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})\vee(\forall y|: y \notin \mathcal{C}[\![S]\!] \upharpoonright_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\})))$$
$$\wedge(\forall y|: y \in \mathcal{C}[\![S]\!] \upharpoonright_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists \overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$
$$\equiv< \mathcal{C}[\![S]\!] \upharpoonright_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \neq \emptyset >$$

$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y}) \vee false)\wedge$$
$$(\forall y|: y \in \mathcal{C}[\![S]\!] \upharpoonright_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists \overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

$\equiv$

$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})\wedge$$
$$(\overline{x},\overline{y}) \in supp(\mathcal{C}[\![S]\!])\wedge$$
$$(\forall y|: y \in \mathcal{C}[\![S]\!] \upharpoonright_{supp(\mathcal{C}[\![S]\!])} (\{(\overline{x},\overline{y})\}) \Rightarrow$$
$$(\exists \overline{y'}|y = (\overline{x},\overline{y'}) : Q(\overline{x},\overline{y'},\overline{Y})))$$

$\equiv$

$$P(\overline{x},y_{i_1},\ldots,y_{i_k},\overline{Y})\wedge$$
$$wp(S,Q(\overline{x},\overline{y},\overline{Y}))$$

■

III. Instrucción for

Una instrucción for es equivalente a un ciclo Do de la siguiente forma:

do $i \neq N \rightarrow$
  S_0;
  $i := i + 1$
od
$\{Post\}$

Donde S_0 es un bloque de instrucciones que no modifica las variable i y N. Para poder calcular la precondición más débil de un ciclo de este estilo, presentamos el siguiente teorema y Corolario.

Teorema 2. *Sea un algoritmo con la estructura que se presentó anteriormente y k, k' son variables que no ocurren en S_0 y $Post$, entonces:*

1) *Si existe un predicado inv tal que $inv[i := N] \equiv Post$ y $(\forall i|N - k \leq i < N : wp(S_0; i := i + 1, inv) \equiv inv)$, donde las variables k y k' no ocurren en inv, entonces*

$$H_{k'}(Post) \equiv N - k' \leq i \leq N \wedge inv$$

   para todo k' tal que $0 \leq k' \leq k$.
2) *Adicionalmente si $wp(S_0; i := i + 1, inv) \equiv False$ cuando $i = N - (k + 1)$, entonces*

$$H_{k+1}(Post) \equiv H_k(Post)$$

Demostración: Por ser este un teorema sobre una fórmula cuyas instancias son fórmulas, entonces se usará un sistema de derivación formal de predicados para asegurar un resultado correcto. El lector debe entender la siguiente demostración como una familia de demostraciones (una por cada instancia del predicado inv), que resulta de aplicar cada una de las derivaciones siguientes en el orden que se presentan. Las reglas de inferencia que se usan en este trabajo son las de la lógica ecuacional presentada en el libro de Gries [10].
Se demostrará por inducción sobre k' suponiendo que $k' \leq k$ y que k' y k son variables que no ocurren en inv, S_0 y $Post$.

Caso 1 $k' = 0$

$H_{k'}(Post)$
$\equiv< k' = 0 >$
$H_0(Post)$
$\equiv$
$i = N \wedge Post$
$\equiv$
$i = N \wedge inv[i := N]$
$\equiv$
$i = N \wedge inv[i := i]$
$\equiv$
$N \leq i \leq N \wedge inv$
$\equiv< k' = 0 >$
$N - k' \leq i \leq N \wedge inv$

Se supone ahora que el teorema es cierto para $k' - 1$ y se demostrará para k'

$H_{k'}(Post)$
$\equiv$
$H_0(Post) \vee (i \neq N \wedge wp(S_0; i := i + 1, H_{k'-1}(Post)))$

$\equiv$<hipótesis inductiva>

$H_0(Post) \vee (i \neq N \wedge wp(S_0; i := i + 1,$
$N - (k' - 1) \leq i \leq N \wedge inv))$

$\equiv$<hipótesis>

$H_0(Post) \vee (i \neq N \wedge wp(S_0; i := i + 1,$
$N - (k' - 1) \leq i \leq N) \wedge wp(S_0; i := i + 1, inv))$

$\equiv<$ i y N no se modifican en S_0 y k' no ocurre en S_0 y lema 1>

$H_0(Post) \vee (i \neq N \wedge N - (k' - 1) \leq i + 1 \leq N \wedge$
$wp(S_0; i := i + 1, inv))$
$\equiv$
$H_0(Post) \vee (i \neq N \wedge N - k' \leq i \leq N - 1 \wedge$
$wp(S_0; i := i + 1, inv))$

$\equiv$<hipótesis>

$H_0(Post) \vee (i \neq N \wedge N - k' \leq i \leq N - 1 \wedge inv)$
$\equiv$

$H_0(Post) \vee (N - k' \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \wedge Post) \vee (N - k' \leq i \leq N - 1 \wedge inv)$

$\equiv$<hipótesis>

$(i = N \wedge inv[i := N]) \vee (N - k' \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \wedge inv[i := i]) \vee (N - k' \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \wedge inv) \vee (N - k' \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \vee N - k' \leq i \leq N - 1) \wedge inv$
$\equiv$
$(N - k' \leq i \leq N) \wedge inv$

Por otro lado

Si $wp(S_0; i := i + 1, inv) \equiv False$ cuando $i = N - (k + 1)$ entonces

$H_{k+1}(Post)$
$\equiv$
$H_0(Post) \vee (i \neq N \wedge wp(S_0; i := i + 1, H_k(Post)))$
$\equiv$
$H_0(Post) \vee (i \neq N \wedge wp(S_0; i := i + 1,$
$N - k \leq i \leq N \wedge inv))$
$\equiv$
$H_0(Post) \vee (i \neq N \wedge wp(S_0; i := i + 1,$
$N - k \leq i \leq N) \wedge wp(S_0; i := i + 1, inv))$

$\equiv <$ i y N no se modifican en S_0 y k no ocurre en S_0 y lema 1$>$

$H_0(Post) \vee (i \neq N \wedge N - k \leq i + 1 \leq N \wedge$
$wp(S_0; i := i + 1, inv))$

$\equiv$<hipótesis>

$H_0(Post) \vee (i \neq N \wedge N - k - 1 \leq i \leq N - 1 \wedge$
$wp(S_0; i := i + 1, inv))$
$\equiv$
$H_0(Post) \vee (i \neq N \wedge N - (k + 1) \leq i \leq N - 1 \wedge$
$wp(S_0; i := i + 1, inv))$
$\equiv$
$H_0(Post) \vee (N - (k + 1) \leq i \leq N - 1 \wedge$
$wp(S_0; i := i + 1, inv))$

$\equiv < k \geq 0 >$

$H_0(Post) \vee ((i = N - (k + 1) \vee N - k \leq i \leq N - 1) \wedge$
$wp(S_0; i := i + 1, inv))$
$\equiv$
$H_0(Post) \vee (i = N - (k + 1) \wedge wp(S_0; i := i + 1, inv)) \vee$
$(N - k \leq i \leq N - 1 \wedge wp(S_0; i := i + 1, inv))$

$\equiv$<hipótesis>

$H_0(Post) \vee (i = N - (k + 1) \wedge False) \vee$
$(N - k \leq i \leq N - 1 \wedge inv)$
$\equiv$
$H_0(Post) \vee False \vee (N - k \leq i \leq N - 1 \wedge inv)$
$\equiv$
$H_0(Post) \vee (N - k \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \wedge Post) \vee (N - k \leq i \leq N - 1 \wedge inv)$

$\equiv$<hipótesis>

$(i = N \wedge inv[i := N]) \vee (N - k \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \wedge inv[i := i]) \vee (N - k \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \wedge inv) \vee (N - k \leq i \leq N - 1 \wedge inv)$
$\equiv$
$(i = N \vee N - k \leq i \leq N - 1) \wedge inv$
$\equiv$
$N - k \leq i \leq N \wedge inv$
$\equiv$
$H_k(Post)$ ■

Corolario 2. *Si un predicado inv cumple con las hipótesis del caso 1 del teorema 2 para todo k y no cumple con la hipótesis del caso 2 del mismo teorema, entonces*

$$wp(do\ B_0 \rightarrow S_0; i := i + 1\ od, Post) \equiv i \leq N \wedge inv$$

por otro lado, si inv cumple con las hipótesis de los casos 1 y 2, entonces

$wp(do\ B_0 \rightarrow S_0; i := i+1\ od, Post) \equiv N-k \leq i \leq N \wedge inv$

Demostración: Si inv cumple con las hipótesis del caso 1 del teorema 2 para todo k y no se cumple la hipótesis del caso 2, entonces

$wp(do\ B_0 \rightarrow S_0; i := i + 1\ od, Post)$
$\equiv$
$(\exists k | k \geq 0 : H_k(Post))$
$\equiv$
$(\exists k | k \geq 0 : N - k \leq i \leq N \wedge inv)$
$\equiv$
$(\exists k | k \geq 0 : N - k \leq i \wedge i \leq N \wedge inv)$
$\equiv$
$i \leq N \wedge inv \wedge (\exists k | k \geq 0 : N - k \leq i)$
$\equiv$
$i \leq N \wedge inv \wedge True$
$\equiv$
$i \leq N \wedge inv$

Si se cumplen las hipótesis de los casos 1 y 2 del teorema 2, entonces asumiendo que la variable k del teorema es mayor o igual a cero, se tiene que

$(\exists k'' | k'' \geq 0 : H_{k''}(Post))$
$\equiv$
$(\exists k'' | 0 \leq k'' \leq k : H_{k''}(Post)) \vee$

$(\exists k''|k < k'' : H_{k''}(Post))$

$\equiv$<caso 1 del teorema 2>

$(\exists k''|0 \leq k'' \leq k : N - k'' \leq i \leq N \wedge inv) \vee$
$(\exists k''|k < k'' : H_k(Post))$
$\equiv$
$(\exists k''|0 \leq k'' \leq k : N - k'' \leq i \wedge i \leq N \wedge inv) \vee$
$(\exists k''| : k < k'' \wedge H_k(Post))$
$\equiv$
$(i \leq N \wedge inv \wedge (\exists k''|0 \leq k'' \leq k : N - k'' \leq i)) \vee$
$(H_k(Post) \wedge (\exists k''| : k < k''))$
$\equiv$
$(i \leq N \wedge inv \wedge N - k \leq i) \vee (H_k(Post) \wedge True)$
$\equiv$
$(N - k \leq i \wedge i \leq N \wedge inv) \vee (H_k(Post))$
$\equiv$
$(N - k \leq i \leq N \wedge inv) \vee (N - k \leq i \leq N \wedge inv)$
$\equiv$
$(N - k \leq i \leq N \wedge inv)$

■

El teorema anterior tiene la misma limitante que el teorema de la invariancia, que pretende que se busque un predicado invariante inv sin ningún método o heurística particular. A continuación se dará un teorema que sugiere un método que permite conseguir un predicado como el que establece el teorema anterior

Teorema 3. *Sea un algoritmo con la estructura mostrada anteriormente, ϵ una variable distinta de N que no ocurre en $S_0; i := i+1$ ni en $Post$, i no ocurre en $Post$ y definiendo el predicado $PostG$ tal que satisfaga las siguientes ecuaciones recursivas:*

- $PostG[\epsilon := 0] \equiv Post$
- $PostG \equiv wp(S_0; i := i + 1, PostG[\epsilon := \epsilon - 1])$ *para* $0 < \epsilon \leq k$ *y* $N - k \leq i < N$

entonces el predicado $PostG[\epsilon := N - i]$ es un predicado, que satisface las hipótesis del caso 1 del teorema 2.

Demostración: Instanciemos el predicado $PostG[\epsilon := N - i]$ con $i := N$

$PostG[\epsilon := N - i][i := N]$

$\equiv <$ i puede ocurrir en $PostG >$

$PostG[i := N][\epsilon := N - N]$
$\equiv$
$PostG[i := N][\epsilon := 0]$

$\equiv <$ ϵ es una variable fresca$>$

$PostG[\epsilon := 0][i := N]$

$\equiv$<hipótesis>

$Post[i := N]$

$\equiv <$ i no ocurre en $Post >$

$Post$

Por otro lado tenemos que:

$(\forall i|N - k \leq i < N : wp(S_0; i := i + 1, PostG[\epsilon := N - i]))$
$\equiv$
$(\forall i|N - k \leq i < N : wp(S_0; i := i + 1,$
$(\forall \epsilon|\epsilon = N - i : PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : wp(S_0; i := i + 1,$
$(\forall \epsilon| : \epsilon = N - i \Rightarrow PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : (\forall \epsilon| : wp(S_0; i := i + 1,$
$\epsilon = N - i \Rightarrow PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : (\forall \epsilon| : wp(S_0; i := i + 1,$
$\epsilon \neq N - i \vee PostG)))$

$\equiv <$ i y N no se modifican en S_0 y lema 1$>$

$(\forall i|N - k \leq i < N : (\forall \epsilon| : soporte(S_0) \wedge (\epsilon \neq N - (i+1) \vee$
$wp(S_0; i := i + 1, PostG))))$
$\equiv$
$(\forall i|N - k \leq i < N : soporte(S_0) \wedge$
$(\forall \epsilon| : \epsilon \neq N - (i + 1) \vee wp(S_0; i := i + 1, PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : soporte(S_0) \wedge$
$(\forall \epsilon| : \epsilon = N - (i + 1) \Rightarrow wp(S_0; i := i + 1, PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : soporte(S_0) \wedge$
$(\forall \epsilon|\epsilon = N - (i + 1) : wp(S_0; i := i + 1, PostG)))$

$\equiv$<Existe ϵ tal que $\epsilon = N - (i + 1) >$

$(\forall i|N - k \leq i < N :$
$(\forall \epsilon|\epsilon = N - (i + 1) : soporte(S_0) \wedge$
$wp(S_0; i := i + 1, PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N :$
$(\forall \epsilon|\epsilon = N - (i + 1) : soporte(S_0) \wedge$
$wp(S_0, wp(i := i + 1, PostG))))$

$\equiv <$ $wp(S_0, P) \Rightarrow soporte(S_0)$ para cualquier $P >$

$(\forall i|N - k \leq i < N :$
$(\forall \epsilon|\epsilon = N - (i + 1) : wp(S_0; wp(i := i + 1, PostG))))$
$\equiv$
$(\forall i|N - k \leq i < N :$
$(\forall \epsilon|i = N - (\epsilon + 1) : wp(S_0; i := i + 1, PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : (\forall \epsilon|i = N - (\epsilon + 1) \wedge N - k \leq i \wedge$
$i < N : wp(S_0; i := i + 1, PostG)))$
$\equiv$
$(\forall i|N - k \leq i < N : (\forall \epsilon|i = N - (\epsilon + 1) \wedge$
$N - k \leq N - (\epsilon + 1) \wedge N - (\epsilon + 1) < N :$
$wp(S_0; i := i + 1, PostG)))$

$\equiv$
$(\forall i | N-k \leq i < N : (\forall \epsilon | i = N-(\epsilon+1) \wedge \epsilon \leq k-1 \wedge$
$-1 < \epsilon : wp(S_0; i := i+1, PostG)))$
$\equiv$
$(\forall i | N-k \leq i < N : (\forall \epsilon | i = N-(\epsilon+1) \wedge 0 \leq \epsilon < k :$
$wp(S_0; i := i+1, PostG)))$

$\equiv$<hipótesis>

$(\forall i | N-k \leq i < N : (\forall \epsilon | i = N-(\epsilon+1) \wedge 0 \leq \epsilon < k :$
$PostG[\epsilon := \epsilon+1]))$
$\equiv$
$(\forall i | N-k \leq i < N : (\forall \epsilon | i = N-(\epsilon+1) :$
$PostG[\epsilon := \epsilon+1]))$
$\equiv$
$(\forall i | N-k \leq i < N : (\forall \epsilon | \epsilon = N-(i+1) :$
$PostG[\epsilon := \epsilon+1]))$
$\equiv$
$(\forall i | N-k \leq i < N : PostG[\epsilon := \epsilon+1][\epsilon := N-(i+1)])$
$\equiv$
$(\forall i | N-k \leq i < N : PostG[\epsilon := N-(i+1)+1])$
$\equiv$
$(\forall i | N-k \leq i < N : PostG[\epsilon := N-i])$ ■

IV. Ejemplos

El teorema anterior sugiere un método para calcular la precondición más débil de un ciclo del tipo que se presentó anteriormente. La técnica consiste en aplicar el transformador de predicados al cuerpo del ciclo y la postcondición ϵ veces hasta deducir el predicado $PostG$. Se muestra a continuación un ejemplo:

A. Factorial

do $i \neq N \rightarrow$
  $fact := fact * i;$
  $i := i+1$
od
$\{Post : N > 0 \wedge fact = (N-1)!\}$

La instrucción de asignación en paralelo $fact, i := fact*i, i+1$ es equivalente a las dos instrucciones del bloque interno del ciclo, por lo que para resumir se va a usar la instrucción de asignación en paralelo en los cálculos de este ejemplo.

Se aplica wp una vez al cuerpo del ciclo y a la postcondición:

$wp(fact, i := fact*i, i+1, N > 0 \wedge fact = (N-1)!)$
$\equiv$
$(N > 0 \wedge fact = (N-1)!)[fact, i := fact*i, i+1]$
$\equiv$
$N > 0 \wedge fact * i = (N-1)!$

Ahora al resultado anterior se le vuelve aplicar wp obteniendo

$wp(fact, i := fact*i, i+1, N > 0 \wedge fact * i = (N-1)!)$
$\equiv$
$(N > 0 \wedge fact * i = (N-1)!)[fact, i := fact*i, i+1]$
$\equiv$
$N > 0 \wedge fact * i * (i+1) = (N-1)!$

Si al resultado anterior se le vuelve aplicar wp se obtiene:

$wp(fact, i := fact*i, i+1, N > 0 \wedge fact*i*(i+1) = (N-1)!)$
$\equiv$
$(N > 0 \wedge fact*i*(i+1) = (N-1)!)[fact, i := fact*i, i+1]$
$\equiv$
$N > 0 \wedge fact * i * (i+1) * (i+2) = (N-1)!$

y si se define

$$i^{\overline{\epsilon}} := \begin{cases} i(i+1)(i+2)\cdots(i+\epsilon-1) & si \quad \epsilon > 0 \\ 1 & si \quad \epsilon = 0 \end{cases}$$

entonces es fácil demostrar por inducción, que el resultado de aplicar wp al cuerpo de éste ciclo y a la postcondición $N > 0 \wedge fact = (N-1)!$ un número de ϵ de veces es igual a

$$N > 0 \wedge fact * i^{\overline{\epsilon}} = (N-1)!$$

Por lo tanto este predicado satisface la recurrencia que define a $PostG$ en el último teorema de la sección anterior para $0 \leq \epsilon \leq N-1$, ya que si $\epsilon = N$ entonces el predicado es insatisfacible porque la recurrencia esta limitada a que $i < N$ por hipótesis.

De esta forma usando el teorema recién referenciado, tenemos que

$(N > 0 \wedge fact * i^{\overline{\epsilon}} = (N-1)!)[\epsilon := N-i]$
$\equiv$
$N > 0 \wedge fact * i^{\overline{N-i}} = (N-1)!$
$\equiv$
$N > 0 \wedge fact * i^{\overline{N-i}} = i^{\overline{N-i}} * (i-1)!$
$\equiv <$ *regla de cancelación* $>$
$N > 0 \wedge fact = (i-1)!$

Es un predicado que satisface las hipótesis del caso 1 del teorema 2, es decir

$(N > 0 \wedge fact = (i-1)!)[i := N] \equiv Post$

y

$(\forall i | N-(N-1) \leq i < N :$
  $wp(fact, i := fact*i, i+1, fact = (i-1)!) \equiv$
  $fact = (i-1)!).$

Adicionalmente si $i = 0 = N-((N-1)+1)$, entonces

$wp(fact, i := fact*i, i+1, N > 0 \wedge fact = (i-1)!)$
$\equiv$
$N > 0 \wedge fact * i = ((i+1)-1)!$
$\equiv$
$N > 0 \wedge fact * i = i!$
$\equiv$
$N > 0 \wedge fact * 0 = 0!$

$\equiv$
$N > 0 \wedge 0 = 1$
$\equiv$
$N > 0 \wedge False$
$\equiv$
$False$

Por lo que se cumple la hipótesis del caso 2 del teorema 2 y por lo tanto la precondición más débil del ciclo de éste ejemplo, es la siguiente:

$$N - (N-1) \leq i \leq N \wedge N > 0 \wedge fact = (i-1)!$$

que es equivalente a:

$$1 \leq i \leq N \wedge N > 0 \wedge fact = (i-1)!$$

B. Sumatoria de los Primeros N Enteros al Cuadrado

Para ejemplificar el uso de los teoremas anteriores, se considerará ahora el siguiente algoritmo para calcular la sumatoria de los primeros N enteros positivos al cuadrado.

do $i \neq N \rightarrow$
$suma := suma + i * i;$
$i := i + 1$
od
$\{Post : N \geq 0 \wedge suma = \sum_{j=0}^{N-1} j * j\}$

La instrucción de asignación en paralelo $suma, i := suma + i * i, i+1$, es equivalente a las dos instrucciones que se encuentran dentro del ciclo anterior, por lo tanto para simplificar los cálculos siguientes, se usará la instrucción de asignación en paralelo en lugar del cuerpo del ciclo anterior.

Se Aplica el transformador wp al cuerpo del ciclo y la postcondición:

$wp(suma, i := suma + i * i, i+1, N \geq 0 \wedge suma = \sum_{j=0}^{N-1} j^2)$
$\equiv$
$(N \geq 0 \wedge suma = \sum_{j=0}^{N-1} j^2)[suma, i := suma + i * i, i+1]$
$\equiv$
$N \geq 0 \wedge suma + i * i = \sum_{j=0}^{N-1} j^2$

Se aplica ahora el transformador wp al cuerpo del ciclo y al resultado anterior

$wp(suma, i := suma + i * i, i+1, N \geq 0 \wedge suma + i * i = \sum_{j=0}^{N-1} j^2)$
$\equiv$
$(N \geq 0 \wedge suma + i^2 = \sum_{j=0}^{N-1} j^2)[suma, i := suma + i * i, i+1]$
$\equiv$
$N \geq 0 \wedge suma + i * i + (i+1)^2 = \sum_{j=0}^{N-1} j^2$
$\equiv$
$N \geq 0 \wedge suma + \sum_{j=i}^{i+1} j^2 = \sum_{j=0}^{N-1} j^2$

Por inducción se puede demostrar que aplicar un número ϵ de veces, el transformador wp al cuerpo del ciclo y la postcondición se obtiene la siguiente fórmula:

$$N \geq 0 \wedge suma + \sum_{j=i}^{i+\epsilon-1} j^2 = \sum_{j=0}^{N-1} j^2$$

Es facil demostrar despejando la variable $suma$, que la fórmula es siempre satisfacible para cualquier valor de ϵ, por lo tanto el predicado:

$(N \geq 0 \wedge suma + \sum_{j=i}^{i+\epsilon-1} j^2 = \sum_{j=0}^{N-1} j^2)[\epsilon := N - i]$
$\equiv$
$N \geq 0 \wedge suma + \sum_{j=i}^{i+N-i-1} j^2 = \sum_{j=0}^{N-1} j^2$
$\equiv$
$N \geq 0 \wedge suma + \sum_{j=i}^{N-1} j^2 = \sum_{j=0}^{N-1} j^2$

Satisface las hipótesis del caso 1 del teorema 2 para cualquier valor de k y por lo tanto la precondición más débil de este ejemplo es $i \leq N \wedge N \geq 0 \wedge suma + \sum_{j=i}^{N-1} j^2 = \sum_{j=0}^{N-1} j^2$

Esta precondición puede refinarse aún más haciendo lo siguiente:

$i \leq N \wedge N \geq 0 \wedge suma + \sum_{j=i}^{N-1} j^2 = \sum_{j=0}^{N-1} j^2$
$\equiv$
$i \leq N \wedge N \geq 0 \wedge suma = \sum_{j=0}^{N-1} j^2 - \sum_{j=i}^{N-1} j^2$
$\equiv$
$i \leq N \wedge N \geq 0 \wedge (i \geq 0 \Rightarrow suma = \sum_{j=0}^{i-1} j^2)$
$\wedge (i < 0 \Rightarrow suma = -\sum_{j=i}^{0} j^2)$
$\equiv$
$i \leq N \wedge N \geq 0 \wedge suma = \sum_{j=0}^{i-1} j^2 - \sum_{j=i}^{0} j^2$

Esta precondición dice que el ciclo puede empezar desde un índice i con valor negativo, pero para eso la variable sum debe ser inicializada en el ciclo con el valor negativo

$$-\sum_{j=i}^{0} j^2,$$

para que en cada iteración le sea sumada una cantidad positiva, hasta que su valor llegue a ser positivo e igual al valor que indica la postcondición.

V. Teoremas Complementarios

A continuación se presenta un teorema y corolario cuyas demostraciones son análogas a las que se presentaron anteriormente.

Sea un algoritmo con la estructura

do $i \neq N \rightarrow$
  S_0;
  $i := i - 1$
od
$\{Post\}$,

donde las variables i y N no son modificadas en S_0.

Teorema 4. *Sea un algoritmo con la estructura que se presentó arriba y k, k' variables que no ocurren en S_0 y $Post$, entonces:*
si existe un predicado inv tal que $inv[i := N] \equiv Post$ y $(\forall i | N < i \leq N + k : wp(S_0; i := i - 1, inv) \equiv inv)$, donde las variables k, k' no ocurren en inv, entonces

$$H_{k'}(Post) \equiv N \leq i \leq N + k' \wedge inv$$

para todo k' tal que $0 \leq k' \leq k$.
Adicionalmente si $wp(S_0; i := i - 1, inv) \equiv False$ cuando $i = N + (k + 1)$, entonces

$$H_{k+1}(Post) \equiv H_k(Post)$$

Corolario 3. *Si un predicado inv cumple con las hipótesis del caso 1 del teorema 4 para todo k y no cumple la hipótesis del caso 2 del mismo teorema, entonces:*

$$wp(do\ B_0 \rightarrow S_0; i := i - 1\ od, Post) \equiv N \leq i \wedge inv$$

por otro lado, si cumple con las hipótesis de los casos 1 y 2, entonces:

$$wp(do\ B_0 \rightarrow S_0; i := i+1\ od, Post) \equiv N \leq i \leq N+k \wedge inv$$

Definición. *La función rampa se define de la siguiente manera:* $r^{(N)}(i) = \begin{cases} i & si \quad i \leq N \\ N & si \quad i > N \end{cases}$

La guardia de todos los ciclos que se han estudiado hasta ahora es $i \neq N$, si la intercambiamos por $i < N$, la precondición más débil es ligeramente distinta. El siguiente teorema proporciona un resultado al respecto.

Teorema 5. *Sea el algoritmo con la estructura*

do $i < N \rightarrow$
  S_0;
  $i := i + 1$
od
$\{Post\}$,

donde la variable i y N no son modificadas en S_0.
Si k, k' son variables que no ocurren en S_0 y $Post$, entonces:

1) *si existe un predicado inv tal que $inv[i := N] \equiv Post$ y $(\forall i | N - k \leq i < N : wp(S_0; i := i + 1, inv) \equiv inv)$, donde las variables k, k' no ocurren en inv, entonces*

$$H_{k'}(Post) \equiv N - k' \leq i \wedge inv[i := r^{(N)}(i)]$$

para todo k' tal que $0 \leq k' \leq k$.
2) *Adicionalmente si $wp(S_0; i := i + 1, inv) \equiv False$ cuando $i = N - (k + 1)$, entonces*

$$H_{k+1}(Post) \equiv H_k(Post)$$

Demostración: Se demuestra por inducción sobre k' suponiendo que $k' \leq k$ y que k' y k son variables que no ocurren en inv, S_0 y $Post$.

Caso 1 $k' = 0$

$H_{k'}(Post)$
$\equiv< k' = 0 >$
$H_0(Post)$
$\equiv$
$i \geq N \wedge Post$
$\equiv$
$(i = N \vee i > N) \wedge Post$
$\equiv$
$(i = N \wedge Post) \vee (i > N \wedge Post)$

$\equiv$<hipótesis>

$(N \leq i \leq N \wedge inv[i := N]) \vee (i > N \wedge Post)$

$\equiv$<hipótesis>

$(N \leq i \leq N \wedge inv[i := N]) \vee (i > N \wedge inv[i := N])$

$\equiv< r^{(N)}(i) = N$ por definición>

$(N \leq i \leq N \wedge inv[i := r^{(N)}(i)]) \vee$
$(i > N \wedge inv[i := r^{(N)}(i)])$
$\equiv< k' = 0 >$
$(N - k' \leq i \leq N \wedge inv[i := r^{(N)}(i)]) \vee$
$(i > N \wedge inv[i := r^{(N)}(i)])$
$\equiv$
$(N - k' \leq i \leq N \vee i > N) \wedge inv[i := r^{(N)}(i)]$
$\equiv$
$N - k' \leq i \wedge inv[i := r^{(N)}(i)]$

A continuación se supone que el teorema es cierto para $k' - 1$ y se demuestra para k'

$H_{k'}(Post)$
$\equiv$
$H_0(Post) \vee (i < N \wedge wp(S_0; i := i + 1, H_{k'-1}(Post)))$

$\equiv$<hipótesis inductiva>

$H_0(Post) \vee (i < N \wedge wp(S_0; i := i + 1,$
$N - (k' - 1) \leq i \wedge inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge wp(S_0; i := i + 1,$
$N - (k' - 1) \leq i) \wedge wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$

$\equiv< i$ y N no se modifican en S_0 y k' no ocurre en S_0 y

lema 1>

$H_0(Post) \vee (i < N \wedge N - (k' - 1) \leq i + 1 \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge N - k' \leq i \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$

$\equiv <$ $r^{(N)}(i) = i$ por definición>

$H_0(Post) \vee (i < N \wedge N - k' \leq i \wedge wp(S_0; i := i + 1, inv))$

$\equiv$<hipótesis>

$H_0(Post) \vee (i < N \wedge N - k' \leq i \wedge inv)$

$\equiv <$ $r^{(N)}(i) = i$ por definición>

$H_0(Post) \vee (i < N \wedge N - k' \leq i \wedge inv[i := r^{(N)}(i)])$
$\equiv$
$(i \geq N \wedge Post) \vee (i < N \wedge N - k' \leq i \wedge inv[i := r^{(N)}(i)])$

$\equiv$<hipótesis>

$(i \geq N \wedge inv[i := N]) \vee (i < N \wedge N - k' \leq i \wedge$
$inv[i := r^{(N)}(i)])$

$\equiv <$ $r^{(N)}(i) = N$ cuando $i \geq N >$

$(i \geq N \wedge inv[i := r^{(N)}(i)]) \vee (i < N \wedge N - k' \leq i \wedge$
$inv[i := r^{(N)}(i)])$
$\equiv$
$(i \geq N \vee (i < N \wedge N - k' \leq i)) \wedge inv[i := r^{(N)}(i)]$
$\equiv$
$(N - k' \leq i) \wedge inv[i := r^{(N)}(i)]$

Si $wp(S_0; i := i + 1, inv) \equiv False$ cuando $i = N - (k + 1)$ entonces

$H_{k+1}(Post)$
$\equiv$
$H_0(Post) \vee (i < N \wedge wp(S_0; i := i + 1, H_k(Post)))$

$\equiv$<caso 1 del teorema>

$H_0(Post) \vee (i < N \wedge wp(S_0; i := i + 1, N - k \leq i \wedge$
$inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge wp(S_0; i := i + 1, N - k \leq i) \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$

$\equiv <$ i y N no se modifican en S_0, k no ocurre en S_0 y lema 1>

$H_0(Post) \vee (i < N \wedge N - k \leq i + 1 \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge N - k - 1 \leq i \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge N - (k + 1) \leq i \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge (i = N - (k + 1) \vee N - k \leq i) \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (((i < N \wedge i = N - (k + 1)) \vee$
$(i < N \wedge N - k \leq i)) \wedge wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$
$\equiv$
$H_0(Post) \vee (i < N \wedge i = N - (k + 1) \wedge$
$wp(S_0; i := i + 1, inv[i := r^{(N)}(i)])) \vee$
$(i < N \wedge N - k \leq i \wedge wp(S_0; i := i + 1, inv[i := r^{(N)}(i)]))$

$\equiv <$ $r^{(N)}(i) = i$ cuando $i < N >$

$H_0(Post) \vee (i < N \wedge i = N - (k + 1) \wedge$
$wp(S_0; i := i + 1, inv)) \vee$
$(i < N \wedge N - k \leq i \wedge wp(S_0; i := i + 1, inv))$

$\equiv$<hipótesis>

$H_0(Post) \vee (i < N \wedge i = N - (k + 1) \wedge False) \vee$
$(i < N \wedge N - k \leq i \wedge inv)$
$\equiv$
$H_0(Post) \vee False \vee (i < N \wedge N - k \leq i \wedge inv)$
$\equiv$
$H_0(Post) \vee (i < N \wedge N - k \leq i \wedge inv)$
$\equiv$
$(i \geq N \wedge Post) \vee (i < N \wedge N - k \leq i \wedge inv)$

$\equiv$<hipótesis>

$(i \geq N \wedge inv[i := N]) \vee (i < N \wedge N - k \leq i \wedge inv)$

$\equiv <$ $r^{(N)}(i) = N$ si $i \geq N$ y $r^{(N)}(i) = i$ si $i < N$ $>$

$(i \geq N \wedge inv[i := r^{(N)}(i)]) \vee$
$(i < N \wedge N - k \leq i \wedge inv[i := r^{(N)}(i)])$
$\equiv$
$(i \geq N \vee (i < N \wedge N - k \leq i)) \wedge inv[i := r^{(N)}(i)]$
$\equiv$
$N - k \leq i \wedge inv[i := r^{(N)}(i)]$
$\equiv$
$H_k(Post)$ ■

La demostración del siguiente corolario es análoga a la del corolario 2.

Corolario 4. *Si un predicado inv cumple con las hipótesis del caso 1 del teorema 5 para todo k y no cumple con la hipótesis del caso 2 del mismo teorema, entonces:*

$$wp(do\ B_0 \rightarrow S_0; i := i + 1\ od, Post) \equiv$$
$$inv[i := r^{(N)}(i)]$$

por otro lado, si inv cumple con las hipótesis de los casos 1 y 2, entonces:

$$wp(do\ B_0 \rightarrow S_0; i := i + 1\ od, Post) \equiv$$

$$N - k \leq i \wedge inv[i := r^{(N)}(i)]$$

Nota. *Como las hipótesis de los casos 1 y 2 del teorema 5 son las mismas que la de los casos 1 y 2 del teorema 2, y para el caso del ejemplo de factorial, el predicado:*

$$N > 0 \wedge fact = (i-1)!$$

satisfacía dichas hipótesis, entonces según el corolario 4 se tiene que para el mismo algoritmo pero sustituyendo la guardia por $i < N$, la precondición más débil del ciclo sería:

$$1 \leq i \wedge fact = (r^{(N)}(i) - 1)!$$

VI. Conclusiones

Los teoremas aquí presentados son un pequeño aporte para el desarrollo de un cálculo efectivo para la corrección de programas. Desarrollar técnicas de cálculo que simplifiquen el trabajo en el área de corrección de algoritmos, es una tarea difícil. Sin embargo, los ejemplos aquí presentados muestran que usando los teoremas adecuados, es posible conseguir la precondición más débil de ciertas instrucciones Do de una forma rápida y formal. Por esta razón, si los esfuerzos de los investigadores se concentraran en desarrollar teoremas como estos, la corrección de programas pasaría a ser en la práctica, un tema de interés a nivel de ingeniería, que impulsaría a la ciencia de la computación a niveles de exactitud y robustez mucho mayores de los de hoy en día.

Referencias

[1] S. Magill, A. Nanevski, E. Clarke, and P. Lee. *Inferring Invariants in Separation Logic for Imperative List-processing Programs*. SPACE, 1(1):57, 2006.

[2] J. Berdine, B. Cook, and S. Ishtiaq, *SLAyer: Memory Safety for Systems-Level Code*. Gopalakrishnan, Springer, Heidelberg 6806:178183, 2011.

[3] C. Varming and L. Birkedal. *Higher-order Separation Logic in Isabelle/holcf*. Electronic Notes in Theoretical Computer Science, 218:371389, 2008

[4] E. W. Dijkstra. *Guarded Commands, Nondeterminacy and Formal Derivation of Programs*. Communications of the ACM, 18(8):453457, 1975.

[5] D. Gries. *The Science of Programming*. New York, New York: Springer, 1981.

[6] C. A. R. Hoare. *An Axiomatic Basis for Computer Programming*. Communications of the ACM, 12(10):576580, 1969.

[7] G. Winskel. *The Formal Semantics of Programming Languages: An Introduction*. MIT Press, 1993.

[8] F. Flaviani. *Modelo Relacional de la Teoría Axiomática del Lenguaje GCL de Dijkstra*. CoNCISa 2015, Valencia, Venezuela, Noviembre 2015, pp. 153-164.

[9] K. Kunen. *Set Theory*. College Publications, London, UK, 2013

[10] D. Gries and F. B. Schneider. *A Logical Approach to Discrete Math*. New York, New York: Springer, 1993.

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 81-93, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

k-Inpainting: Un Enfoque Híbrido para Inpainting

Esmitt Ramírez[1], Karina Pedrique[1]
esmitt.ramirez@ciens.ucv.ve, kpedrique@gmail.com

[1] Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: En el procesamiento digital de imágenes se aplican técnicas que tienen como objetivo mejorar la calidad de la imagen o extraer algún tipo de información. Entre las técnicas utilizadas se encuentran la segmentación, corrección de imagen, análisis de la imagen, reconstrucción de partes perdidas, entre otras. En la reconstrucción de partes de una imagen, existe una técnica llamada Inpainting, la cual modifica una imagen para reconstruir áreas deterioradas o eliminadas de ésta. En la actualidad, los algoritmos de Inpainting ocupan un amplio campo de investigación y desarrollo. En este trabajo se plantea la implementación de un novedoso algoritmo de Inpainting llamado k-Inpainting, el cual está basado en un proceso iterativo para lograr la reconstrucción total de la imagen. El algoritmo k-Inpainting se presenta como una técnica híbrida que une dos enfoques distintos de investigación. Así, se demuestra su utilización al implementar una versión del algoritmo de forma secuencial y una versión paralela en la GPU. Se realizaron diversas pruebas variando los parámetros de entrada del algoritmo, para obtener valores óptimos. Las pruebas comprueban su eficiencia y efectividad en la reconstrucción de segmentos de imágenes a recuperar.

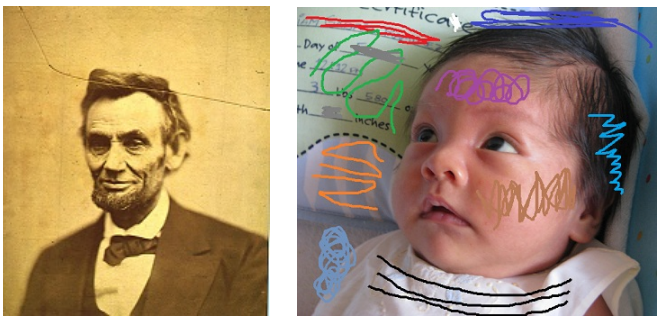
Palabras Clave: Inpainting; Reconstrucción de Imágenes; Síntesis de Textura; Difusión de Textura; Propagación de Textura; Auto-Similitud de Textura.

Abstract: In the digital image processing, techniques are applied in order to improve the quality of the image or to extract some kind of information. Among techniques used are the segmentation, image correction, image analysis, reconstruction of missing parts, and others. In the image reconstruction of missing parts, there is a technique called Inpainting, which modifies an image to reconstruct deteriorated areas of removed from it. At the present, Inpainting algorithms are studied in several research and development areas. In this paper, we present an implementation of a novelty algorithm named k-Inpainting, which is an iterative process to perform the total reconstruction of the image. The k-Inpainting algorithm is presented as a hybrid technique joining two different approaches. Thus, the use of k-Inpainting is probe with a sequential (using the CPU) and parallel version (using the GPU). We performed several tests varying the input parameters of algorithm, to obtain the optimal values. Tests checked its efficiency and effectiveness in the reconstruction of missing segments on images.

Keywords: Inpainting; Image Reconstruction; Texture Synthesis; Texture Diffusion; Texture Propagation; Texture Self-Similarity.

I. Introducción

La restauración de imágenes consiste en recuperar ciertas zonas en imágenes donde visualmente existe una interrupción de la continuidad estética. Un claro ejemplo consiste en el proceso de retoque de una fotografía antigua que ha sido digitalizada, removiendo artefactos notables u objetos, reconstruyendo áreas de un objeto en específico dentro de una imagen. El conjunto de técnicas asociadas con esta restauración se conoce como *Inpainting*. La Figura 1 muestra dos ejemplos de imágenes a ser recuperadas.



(a) (b)

Figura 1: Ejemplos de Restauración de Imágenes con (a) Fotografías Antiguas y (b) Fotografía con Interrupciones Visuales

Remover a un turista de una fotografía en un lugar histórico, o una señal de tránsito, o un ente inesperado en una imagen, es posible hacerlo con la técnica de *Inpainting*. De manera formal, el problema de remoción o restauración de imágenes

consiste en que dada una región Ω a ser restaurada, se emplea la información conocida que rodea dicha región, para regenerar de la manera más aceptable los datos en Ω [1].

El algoritmo de *Inpainting* debe buscar el mejor píxel q en Ω^c que sustituya el valor desconocido de un píxel p en Ω, tal como se muestra en la Figura 2. Es claro que $\Omega \cup \Omega^c = I$, donde I representa a la imagen.

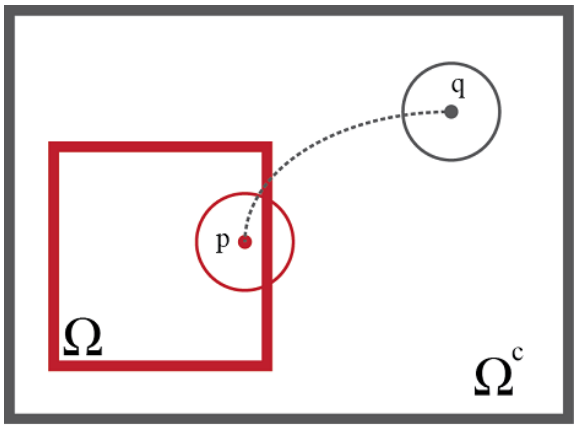


Figura 2: Representación de las Variables Involucradas en el Proceso de Inpainting. El Recuadro Rojo Representa el Área a Restaurar

Las diferentes aplicaciones de procesamiento digital de imágenes que se encargan de mejorar una imagen, tienen en común que los píxeles contienen información acerca de los datos reales y el ruido mientras que en *Inpainting*, no hay información significativa en la región a ser reconstruida. La información se encuentra principalmente alrededor de las áreas a ser tratadas [2]. En consecuencia, se han desarrollado técnicas específicas para estudiar este problema teniendo en común el sistema de vecindad de los píxeles [3].

El proceso de *Inpainting* surge a partir de la importancia de restaurar y modificar imágenes y videos, pero también es empleado para entender la validez de diferentes modelos de imagen (e.g. modelos de imagen estocásticos o determinísticos). Considerando estos modelos de imágenes, en [4] se plantea que la técnica de *Inpainting* puede ser dividida en tres grupos básicos: auto-similitud y síntesis de textura; difusión y propagación (basada en emplear ecuaciones diferenciales parciales de orden superior); y coherencia. Además, se puede incluir un nuevo grupo [5] donde se contemplan métodos basados en la continuación de bordes. Adicionalmente, existen diversos trabajos que combinan técnicas y crean nuevos algoritmos para *Inpainting*. Un resumen de las técnicas principales de *Inpainting* para la reconstrucción, fue presentado por Suthar y Patel [6], y por Joshua y Darsan [7].

En este trabajo, se presenta *k-Inpainting* como una propuesta híbrida que resulta de la combinación de dos trabajos de investigación: *A Comprehensive Framework for Image Inpainting* [4] y *The Generalized PatchMatch Correspondence Algorithm* [8], en conjunto con diferentes aportes presentes en este trabajo. Nuestro algoritmo tiene como entrada dos imágenes y una serie de parámetros: la primera imagen I de entrada es la destinada a ser procesada, la cual tiene una región desconocida Ω a ser reconstruida y la segunda imagen es una máscara M que permite diferenciar en coordenadas cartesianas los píxeles que deben ser procesados. Esta propuesta es llamada híbrida, ya que usa una aproximación del algoritmo *k-Nearest Neighbor* de [8] para una preselección de N candidatos para cada píxel p y aplicar la combinación de energías en [4] entre los N candidatos escogidos previamente para cada píxel p.

Este artículo se organiza de la siguiente forma: la Sección II presenta una visión general de nuestra propuesta. En la Sección III se muestra el proceso de interacción del usuario con la zona a reconstruir, para poder aplicar el algoritmo de *k-Inpainting* que se explica detalladamente en la Sección IV. Dos etapas de particular importancia, *k-Propagación* y *Búsqueda Aleatoria* son presentadas en la Sección V. La Sección VI presenta la experimentación aplicada a nuestra propuesta. Finalmente, en la Sección VII se muestran las conclusiones y trabajos a futuro.

II. Enfoque Global del Algoritmo

La Figura 3 muestra el proceso de reconstrucción aplicado en *k-Inpainting*. La región Ω es reconstruida de afuera hacia adentro, siguiendo un esquema de capas. El color de cada píxel p en Ω es reemplazado por el color de algún píxel q en Ω^c. Este píxel q es escogido aplicando la técnica de mapa de correspondencia junto a otros algoritmos iterativos. Además, se utilizan diferentes métricas para lograr un mejor resultado.

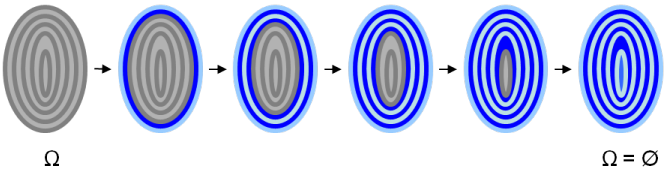


Figura 3: Proceso de Reconstrucción: La Región Ω es Reconstruida de Afuera Hacia Adentro Siguiendo un Sistema de Capas

El enfoque basado en capas permite que el conjunto Ω^c vaya siendo mayor iteración a iteración, de forma tal que exista una mayor cantidad de píxeles q, para un píxel p. El algoritmo *k-Inpainting* puede ser implementado en un enfoque secuencial y en ambientes paralelos. Las métricas presentadas en este trabajo están aplicadas sobre el espacio de color L*a*b [10].

En la Figura 4 se muestra la estructura general del algoritmo de *k-Inpainting* en seis pasos, donde los pasos 4-6 se ejecutan en diversas iteraciones hasta que se converja al resultado final. Primero, se obtienen los datos se selecciona la región a ser reconstruida. A partir de la selección, se aplica el algoritmo DFS (*Depth-First Search*), y se construye una pirámide de Gauss de dos niveles conformada por las imágenes I e I'. La imagen I' constituye una imagen de menor tamaño para poder operar de forma rápida sobre una aproximación de I.

Sobre la imagen I' se aplica la etapa de inicialización que proporciona valores preliminares a Ω a partir del procesamiento de los píxeles recolectados mediante un muestreo. Así, sobre I' se aplica el procesamiento de textura para refinar la zona a

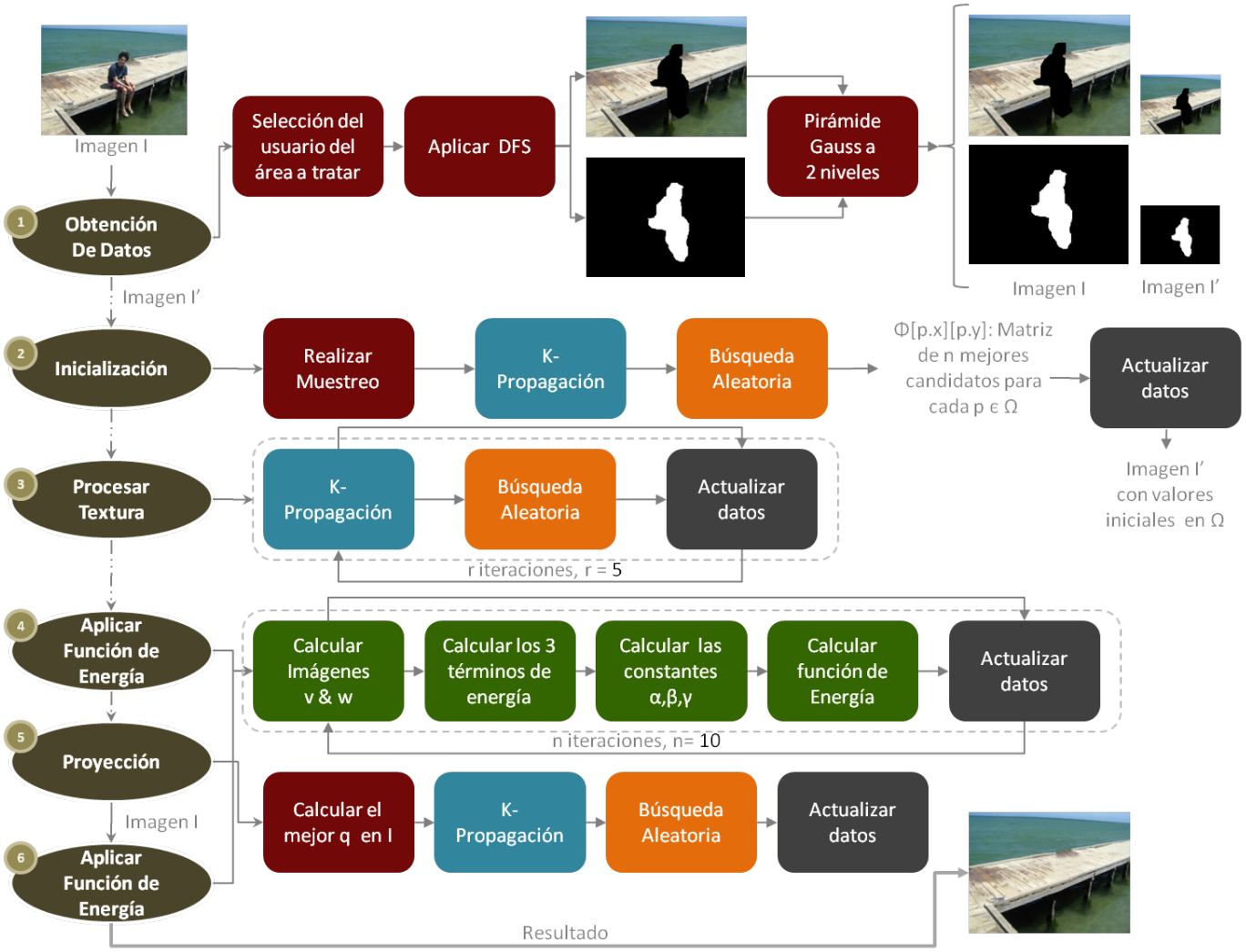


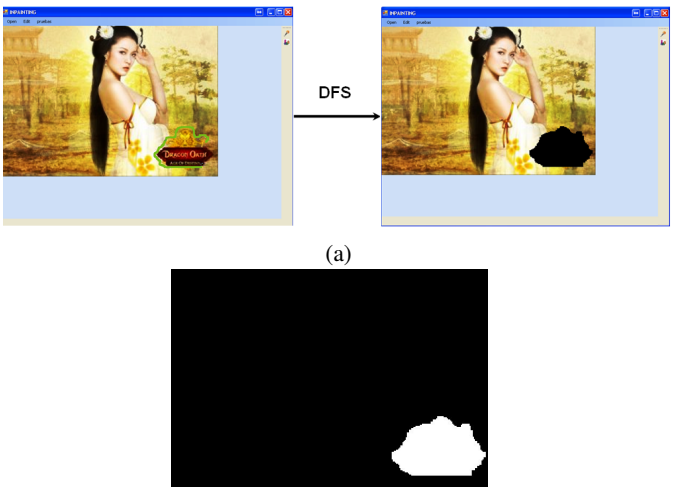
Figura 4: Estructura General del Algoritmo Propuesto

tratar y la prepara para la siguiente etapa, donde se aplica una función de energía que considera las tres técnicas de *Inpainting*. Después, se realiza una proyección de los píxeles procesados en I' a la imagen original I. Finalmente, se aplica la función de energía sobre I y se obtienen el resultado. Realizar el procesamiento en la textura de menor tamaño I' y luego realizar la proyección a a I es menos costoso en tiempo y cómputo que sobre la imagen I de forma directa.

Las sub-etapas *K-Propagación* y *Búsqueda Aleatoria* (etapa 2, 3 y 5) son algoritmos que trabajan en conjunto y buscan encontrar los mejores candidatos para cada píxel que constituye la parte desconocida de la imagen. Se denomina candidato a cualquier píxel $q \in \Omega^c$ que pueda reemplazar el valor incógnita de algún píxel $p \in \Omega$. Estos algoritmos emplean métricas diferentes dependiendo de la etapa donde se ejecuten.

III. Obtención de Datos

El usuario selecciona un área a tratar dentro de la imagen, y se busca un punto dentro del área a reconstruir para aplicar el algoritmo de DFS para llenar dicha área (ver Figura 5a). También se construye una máscara binaria (ver Figura 5b).



(a)

(b)

Figura 5: (a) Selección del Área a Reconstruir en la Imagen y (b) Máscara de la Imagen que Permite Diferenciar el Área a ser Tratada (Ω)

En *k-Inpainting* se utiliza un esquema de multi-resolución [11][12][13], donde se construye una pirámide gaussiana [14] de dos niveles, es decir, se obtiene una nueva imagen I' dos veces más pequeña que la imagen I. Para cada imagen se recopila la máscara y puntos máximos y mínimos del área a reconstruir, así como los puntos máximos y mínimos con respecto a la imagen anterior.

Para el paso de proyección es importante ubicar correctamente la correspondencia de los puntos entre las imágenes, especialmente para las coordenadas ubicadas dentro de Ω. En algunos casos las dimensiones de la imagen no son múltiplo de 2, por ello debe almacenarse el resto de la división. El proceso de corresponder un punto de I a I' se estudiará posteriormente.

IV. Algoritmo *k-Inpainting*

Como se mencionó anteriormente, para aplicar *k-Inpainting* se requiere realizar un muestreo sobre la parte conocida de la imagen para seleccionar píxeles candidatos que pueden sustituir los valores desconocidos de los píxeles que constituyen a Ω. Del mismo modo, se debe emplear una estructura de *heap* con algunas variaciones que permita almacenar los mejores candidatos para cada píxel $p \in \Omega$.

En las diferentes etapas de *k-Inpainting* se emplean diferentes métricas, algunas denominadas términos de energía, mientras que en otras sub-etapas se emplean una ecuación que acopla dichos términos. Así, se deben conocer tales métricas antes de explicar con detalle cada etapa que conforma el algoritmo. A continuación, se describirá la forma como se efectúa el muestreo, así como las características del heap. También se discutirá los términos de energía y de la combinación de los mismos en una función de energía.

1) Implementación del Heap: Para la aplicación del algoritmo, es necesario realizar un muestreo sobre la parte conocida de la imagen para realizar una selección inicial de los candidatos que pueden sustituir los valores desconocidos de los píxeles que constituyen a Ω; por ello se emplea un heap con algunas variaciones que permita almacenar los mejores candidatos para cada píxel $p \in \Omega$.

La formulación de esta estructura está basada en [8], donde implementan un *max-heap* que almacena la distancia D entre dos vectores de color, teniéndose la mayor distancia en el tope de la estructura. Cuando se examinan los candidatos, se construye una tabla *hash* que verifica rápidamente si el candidato ya ha sido almacenado en la lista.

Se presenta la implementación de un *max-heap* sin repetición basado en arreglos con variaciones. La estructura almacena dos datos, un tipo de dato *punto* que identifica cual píxel $q \in \Omega^c$ es candidato a reemplazar algún $p \in \Omega$ y un tipo de dato *float* que es la diferencia entre las vecindades de p y q para alguna métrica. Además, se dispone de una tabla *hash* para guardar los puntos que ya han sido escogidos. El ordenamiento de los puntos viene dado la distancia entre los colores de la vecindad de p y q (ver Figura 6).

Antes de actualizar los datos en la imagen en cada paso del algoritmo, se debe conocer cuál de los N candidatos es más

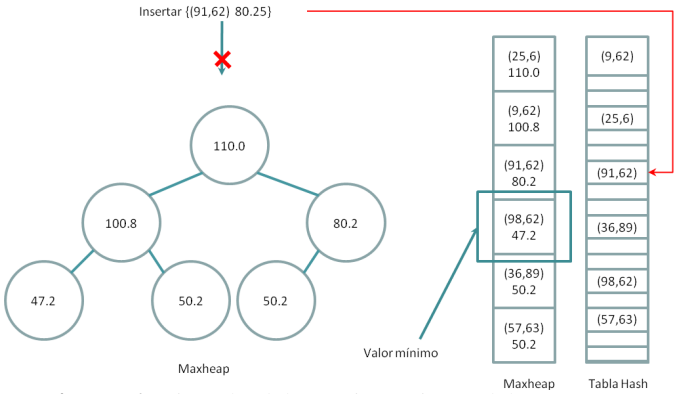


Figura 6: Ejemplo del Funcionamiento del *Max-Heap* Mostrando el Proceso de Inserción del Nodo con Distancia 80.2 y Posición 91,62. Sin embargo, Antes de Insertar se Compara con la Tabla *Hash*, Como la Posición 91,62 Existe, Entonces no se Inserta

conveniente para cada píxel p. El mejor candidato es el píxel que tiene menor diferencia entre su vecindad y la vecindad del píxel para el cual es candidato (i.e. menor valor flotante).

Se estimó que cuando el parámetro N corresponde al 0.05% de los píxeles totales de la imagen, se obtienen resultados satisfactorios. El algoritmo maneja un atributo $16 \leq K \leq 32$ para dos algoritmos que aquí hemos llamado *K-Propagación* y *Búsqueda Aleatoria*. Si el valor de N es menor que el máximo valor que toma K, entonces N toma el máximo valor de K. Dicho esto, N es definido como lo indica la Ecuación 1, dónde $\rho = (I.Width * I.Height * 0.05)/100$.

$$N = \begin{cases} \rho & si \quad \rho \geq 2 \times K \\ 2 \times K & \text{de otro modo} \end{cases} \tag{1}$$

La definición de N es formulada de esta manera para que la dimensión del *heap* no sea muy pequeña con respecto a una selección previa de K candidatos.

2) Muestreo: El paso de inicialización (ver Figura 4) realiza un muestreo uniforme sobre la imagen a procesar, para escoger aleatoriamente los posibles candidatos de cada píxel $p \in \Omega$. Para llevar a cabo tal muestreo se requiere de un generador de números pseudo-aleatorios con distribución uniforme, en los que cada valor tiene la misma probabilidad. En *k-Inpainting* se emplea el método de congruencia lineal [15] y un algoritmo de distribución uniforme [16].

Es importante realizar un muestreo sobre la parte conocida de la imagen, porque realizar un algoritmo de fuerza bruta (cada píxel desconocido por cada píxel de la parte conocida de la imagen) tiene un alto costo computacional, lo cual implica horas de procesamiento en PCs convencionales.

3) Combinación de los Términos de Energía: Los términos de energía se pueden clasificar en técnicas de auto-similitud y síntesis de textura; difusión y propagación; y coherencia. En [4] se presenta una combinación de dichas técnicas que seleccionan un candidato no solo por la similitud de sus vecindades en cuanto a los colores que la constituyen, sino

a la forma como están organizados los colores (coherencia espacial) y el cambio del tono del color de un vecino a otro (difusión), lo que hace que los resultados sean exitosos. Cada uno de estos métodos fue planteado como un término de energía, que serán explicados a continuación.

Auto-similitud y Síntesis de Textura

En [17] [1][4] se presenta una formulación variacional del método de síntesis y textura propuesto por Efros y Leung [18]. Esta variación consiste en calcular el mapa de correspondencia φ que minimiza la función de energía expresada en [17], para un píxel $p \in \Omega \subset \iota$, como:

$$E(\varphi) = \sum_{\tau \in N_0} \|\iota(\varphi(p+\tau)) - \iota(q+\tau)\|^2 \quad (2)$$

La síntesis de textura suele ser un proceso de refinamiento iterativo, y la Ecuación 2 debe ser expresada en función a las r iteraciones aplicadas para procesar la textura (ver Ecuación 3).

$$E_1(\varphi, \varphi^{r-1}) = \sum_{\tau \in N_0} \left\|\iota(\varphi^{r-1}(p+\tau)) - \iota(q+\tau)\right\|^2 \quad (3)$$

Difusión y Propagación

Se toma la difusión Laplaciana como segundo término de energía, entonces dado un píxel $p \in \Omega$, la difusión Laplaciana de p se define como:

$$\upsilon(p, \varphi) = \frac{1}{4}\left[\iota(\varphi(p^N)) + \iota(\varphi(p^S)) + \iota(\varphi(p^E)) + \iota(\varphi(p^O))\right] \quad (4)$$

Basados en la Ecuación 4, el segundo término de energía es:

$$E_2(\varphi, \varphi^{r-1}) = \sum_{\tau \in N_0} \left\|\upsilon(p+\tau, \varphi^{r-1}) - \iota(q+\tau))\right\|^2 \quad (5)$$

Coherencia

El tercer y último termino de energía se define como:

$$E_3(\varphi, \varphi^{r-1}) = \sum_{\tau \in N_0} \left\|\omega(p, \tau, \varphi^{r-1}) - \iota(q+\tau))\right\|^2 \quad (6)$$

, siendo ω el cálculo que favorece la similitud de los parches correspondiente a los píxeles vecinos. Para hacer esta estimación menos sensible a valores que no se encuentran en la imagen (valores aberrantes), en [4] proponen emplear la mediana de los valores generados tal como sigue:

$$\omega(p, \tau, \varphi^{r-1}) = median_{l \in N_0}\left\{\iota(\varphi^{r-1}(p+l) - l + \tau)\right\} \quad (7)$$

El cálculo de la mediana en el espacio L*a*b se hace comparando primero el componente de luz, en caso que para dos colores estos valores sean iguales, se ordena con respecto al componente *a, y por último por el componente *b. Se ordenaron de esta forma, debido a la sensibilidad del ojo humano a la luz. Nótese que todos los términos de energía se basan en la suma de diferencias cuadradas.

4) Combinación de los Tres Términos de Energía: Se deben escoger los N mejores candidatos para cada $p \in \Omega$, los cuales son almacenados en un conjunto $\Phi(p)$. Para cada conjunto $\Phi(p)$ se aplican los tres términos de energía combinados con respecto a cada píxel p en cada iteración r, quedando como:

$$\varphi^{r-1}(p) = argmin_{q \in \Phi(p)}(\alpha E_1 + \beta E_2 + \gamma E_3) \quad (8)$$

Las constantes α, β y γ definen la influencia de cada uno de los tres términos en el resultado final. Estos pesos han sido definidos dependiendo de las propiedades de cada píxel en la imagen según las métricas determinadas. El cálculo de dichos pesos viene dado en primer lugar por el valor mínimo de cada uno de los términos de energía (ver Ecuación 9) En [4] estos pesos han sido definidos dependiendo de las propiedades de cada píxel en la imagen según las métricas determinadas. El cálculo de dichos pesos viene dado en primer lugar por el valor mínimo de cada uno de los términos de energía (ver Ecuación 9)

$$m_i(p) = min_{q \in \Phi(p)} E_i \quad (9)$$

Estos mínimos son considerados en la estimación de las constantes, ya que se asume que los valores de sus pesos podrían depender de la validez del mejor parche candidato (píxel q con su vecindad definida según el parámetro L). Esto es porque si el parche candidato para un determinado píxel en un tiempo t no es suficientemente bueno, el valor del correspondiente peso podría ser pequeño, es decir, se disminuirá en función del valor $m_i(p)$. En relación a lo antes descrito, siendo α, β y γ definidos como:

$$\alpha(p) = \epsilon^{\frac{-m_1}{\sigma}}, \beta(p) = \epsilon^{\frac{-m_2}{\sigma}}, \gamma(p) = \epsilon^{\frac{-m_3}{\sigma}}, \quad (10)$$

donde $\sigma = \frac{m_1+m_2+m_3}{3}$.

Investigaciones previas de *Inpainting* han demostrado que usar el filtro de la ecuación laplaciana arroja mejores resultados que otros tipos de filtros, debido a ello, hemos usado la ecuación laplaciana aplicada sobre cada píxel $p \in \omega \subset \iota$ para cenerar genera una imagen denominada imagen de difusión v y el cálculo de ω sobre el mismo conjunto de píxeles produce una imagen de coherencia w. Por ello es conveniente construir la imagen coherencia y difusión con las ecuaciones 7 y 4 respectivamente, y luego realizar todos los cálculos pertinentes para cada término de energía.

V. K-Propagación y Búsqueda Aleatoria

En varias etapas del algoritmo que se muestra en el esquema de la Figura 4, se aplican los algoritmos de *K-Propagación y Búsqueda aleatoria* para obtener los candidatos de cada píxel desconocido de la imagen. A continuación, se describe cada uno de estos.

A. Obtener Candidatos

Dependiendo de la etapa del algoritmo que se esté aplicando, se puede tener una estructura Υ (lista de muestras) o Φ (matriz de *heaps*) con candidatos. Dichas estructuras son datos de entrada a dos algoritmos que encuentran los mejores N píxeles candidatos a sustituir el valor desconocido de cada píxel p.

Estos dos algoritmos reciben el nombre de *K-Propagación* y *Búsqueda Aleatoria*, el primero toma cada muestra y la compara con K vecinos en una dirección x y en una dirección y. El segundo algoritmo hace una búsqueda aleatoria en varias direcciones a diferentes radios de distancia del píxel escogido y lo compara con sus vecinos. Cabe destacar que estas funciones forman parte del algoritmo *NNF* (*Nearest Neighbors Field*) y *K-NNF* (*K-Nearest Neighbors Field*) presentados en [19] y [8] respectivamente, los cuales están enfocados en la síntesis de textura basada en parches. En esta investigación, éstos se modificaron para sintetizar la textura basada en píxeles.

Las métricas que permiten conocer cuál píxel es mejor candidato que otro, están basadas en la suma de las diferencias cuadradas de las vecindades del píxel incógnita y el píxel seleccionado. Suponiendo la utilización de alguna métrica que calcula una distancia D, se muestra el funcionamiento de estos algoritmos.

1) K-Propagación: Dado un píxel desconocido $p \in \Omega$, *K-Propagación* es aplicado a todos los píxeles $q \in \Upsilon$ ($q \in \Phi$ dependiendo de la etapa que esté ejecutando *k-Inpainting*). Sin embargo, antes se deben escoger las direcciones en el eje x y y. En [19] (algoritmo base de *K-Propagación*) se establece que los candidatos $f(x, y)$ tratan de ser mejorados haciendo desplazamientos $f(x-1, y)$ y $f(x, y-1)$ y en algunas iteraciones los examinan de forma inversa $f(x-1, y)$ y $f(x, y-1)$. En esta propuesta, las direcciones son escogidas aleatoriamente como $q_{kx} = (x \pm k, y)$ y $q_{ky} = (x, y \pm k)$, donde $0 \leq k < K$.

La aleatoriedad de las direcciones viene dada por la elección del signo. En x, si el número aleatoriamente seleccionado es par, el signo es negativo y es positivo en caso que el número sea impar. Para y se sigue la misma política.

Se debe acotar que K es un parámetro global del algoritmo. Los estudios arrojados en [8] del algoritmo *k-PatchMatch*, indican que el valor ideal para K es el valor constante 16. Por lo que se tomará un rango de valores [8,32) para esta variable global en el algoritmo *k-Inpainting*.

Una vez comprendido como se escogen las direcciones x y y y la influencia de la variable K, se puede describir el funcionamiento completo de la subetapa *K-Propagación*, que se resume en el Algoritmo 1.

Algorithm 1 K-Propagación

```
1: procedure KPROPAGACION(Point p ∈ Ω, Point q ∈ Υ,
   Int K)                                  ▷ Siendo Υ ⊂ Ω^c
2:     MaxHeap H(K) ▷ H máximo almacena K candidatos
3:     Escoger una dirección aleatoria en el eje de las x
4:     Escoger una dirección aleatoria en el eje de las y
5:     Evaluar y obtener D para K candidatos electos y
   almacenarlos en H
6:     List λ = H.List()
7:     return λ
8: end procedure
```

2) Búsqueda Aleatoria: En *K-Propagación* se obtuvo la lista λ con los K mejores candidatos. Para evitar un mínimo local, se aplica el algoritmo de *Búsqueda Aleatoria* propuesto en [19], en donde para cada $q \in \lambda$, se aplica que dado $v_0 = \varphi(p)$, se intenta mejorar tal valor probando una secuencia de desplazamientos de candidatos en un decremento de distancia exponencial desde v_0, tal como se muestra en la Figura 7, descrito como $u_i = v_0 + w\alpha^i R_i$.

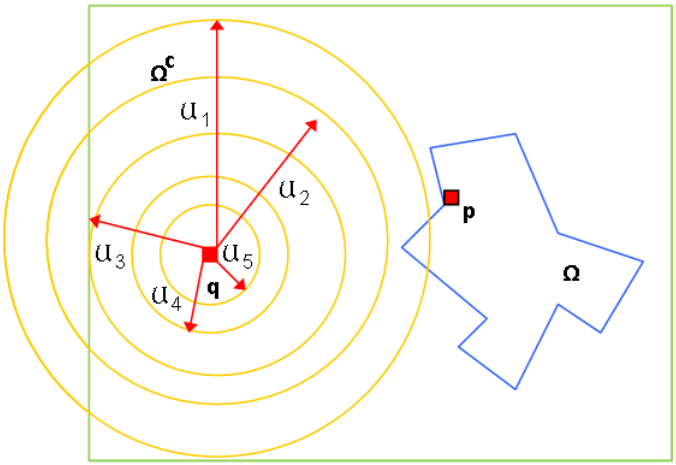


Figura 7: Búsqueda Aleatoria de Puntos Dada una Posición q

La variable R_i es una variable aleatoria uniforme sobre la ventana $[-1, 1] \times [-1, 1]$, w es el máximo radio de búsqueda y α es un valor fijo que varía la distancia del radio de búsqueda en alguna dirección. Se examinan las vecindades de los nuevos puntos generados para i = 0,1,2, ..., hasta que $w\alpha^i < 1$. Entonces, w es la máxima dimensión de la imagen y $\alpha = 1/2$. Sin embargo cuando un nuevo punto generado está fuera de la imagen, este se descarta y se realiza la siguiente iteración.

B. Inicialización

El principal objetivo de este paso es encontrar desde un punto de vista espacial los mejores candidatos para los valores incógnitas de I. Se busca el argumento mínimo producto de aplicar los tres términos de energía.

Una forma de encontrar este argumento, que es empleado por diferentes trabajos [17], [12], [11], [20], es aplicar un modelo basado en los Campos Aleatorios de Markov (MRF). Sin embargo, este mecanismo puede ser costoso en el tiempo para nuestro propósito, por ello en [12] se agiliza el tiempo de respuesta usando esta técnica con el algoritmo de Label Pruning.

En el 2009, Barnes et al. [19] propusieron un algoritmo que busca rápidamente $\varphi(p)$ que recibe el nombre de *Nearest Neighbors Field* (NNF). Sobre este fue realizado una optimización en [8] que permite obtener resultados más precisos, dicha optimización recibe el nombre de *K-Nearest Neighbors Field* (*K-NNF*). En cualquier caso, Barnes et al. [19] emplean en su algoritmo de *PatchMatch* un paso de inicialización, donde realiza pocas iteraciones iniciales del algoritmo *NNF* o *K-NNF* con una inicialización aleatoria.

Basados en la este concepto, sobre $\Omega^c \in I'$ se realiza un muestreo uniforme, a partir del cual se aplicará el algoritmo *NNF*, pero con algunas variantes de las funciones *Propagation* y *Random Search*.

La función de muestreo que aplica el Algoritmo de Las Vegas recibe dos parámetros importantes: el porcentaje de píxeles que serán tomados por fila o columna, y un valor δ que indica la cantidad de filas o columnas a muestrear. Si el ancho de la imagen es mayor que el alto de la misma, el porcentaje va dirigido a las filas y el valor δ indica cada cuantas filas se toma el muestreo de las mismas, en caso contrario aplica igual pero para las columnas.

El porcentaje de píxeles tomados por cada fila o columna es del 25%, mientras que el δ toma el valor constante 2. Estos valores fueron estimados de forma empírica, realizando diferentes pruebas. El algoritmo probabilístico aplicado devuelve una lista Υ con píxeles seleccionados. En dicha lista todos los píxeles son diferentes entre sí en cuanto a ubicación espacial en la imagen.

El muestreo es realizado cada vez que se detecta un borde $\delta\Omega$, por lo que los píxeles en Υ_j son los candidatos a sustituir a los píxeles $p_i \in \delta\Omega_j$ por cada $\delta\Omega_j \subset \Omega$, la Figura 8a presenta el muestreo realizado para cada capa sobre la máscara de la imagen a procesar y la Figura 8b muestra los valores que toma Υ_j para $\delta\Omega_j$.

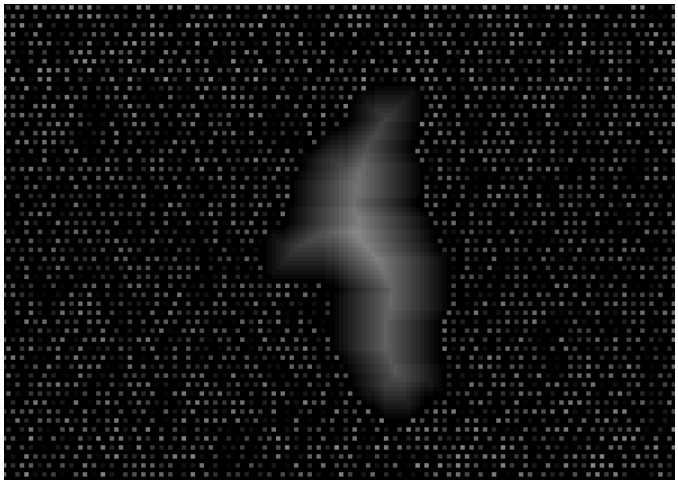
Los N mejores candidatos para cada píxel $p \in \Omega$ son almacenados en un *max-heap* , por ello se construye una matriz de tipo *max-heap* de las dimensiones de la zona a reconstruir. Como la forma de la zona, puede ser irregular, dentro de la matriz pueden existir estructuras vacías para los píxeles que no pertenecen a Ω.

Con los mejores N candidatos para cada píxel desconocido del agujero, se asigna el mejor candidato al píxel p_i de la zona a tratar en la imagen. Por ser el paso de inicialización, se está reconstruyendo el área Ω de I' y se indica en una máscara auxiliar cuales píxeles fueron sustituidos. Dado que este proceso es iterativo, siempre se usa una máscara auxiliar para no perder la localidad espacial del área de la imagen que se está procesando (ver el Algoritmo 2).

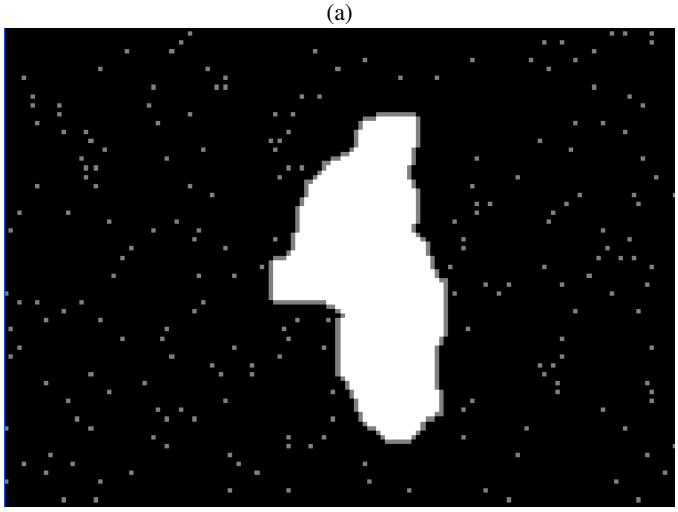
La *K-Propagación* y *Búsqueda Aleatoria* son aplicados sobre todas las muestras con respecto a cada píxel p_i y encuentran los mejores N píxeles a sustituir el valor desconocido de cada p. Así,ya se ha calculado $\varphi(p)$ $\forall p \in \Omega \subset I'$. Para calcular $\varphi(p)$, se utiliza una variación sobre la suma de diferencias cuadradas (i.e. ssd), de manera similar a como lo plantea Komodakis [12], donde la diferencia de los vecinos en posiciones desconocidas, no son calculadas (ver Ecuación 11).

$$ssd(N_p, N_q) = \sum_{\tau \in N_0} M(p+\tau) * \|\iota(p+\tau) - \iota(q+\tau)\|^2 \quad (11)$$

donde $p \in \Omega$, $q \in \Omega^c$ e $\iota = \Omega \cup \Omega^c$.



(a)



(b)

Figura 8: Representación del (a) Muestreo Realizado por Cada Capa, y (b) Un Ejemplo del Muestreo Sobre una Sola Capa (*la más externa*)

Algorithm 2 Algoritmo para inicializar Ω

```
Array Υ
MaxHeap Φ[widthHole][heightHole]
Image M = mask ▷ Se asigna la máscara (mask) de I' a una máscara auxiliar
while Ω ≠ ∅ do
    Seleccionar δΩ_j
    RealizarMuestreo(Υ_j, Mask, ...)
    for cada píxel p ∈ δΩ_j do
        Hallar los n mejores candidatos a partir de Υ_j y almacenarlos en Φ[p.X][p.Y]
        I(p) = φ(p)
        M(p) = 0
    end for
end while
```

C. Procesamiento de Textura

Este paso tiene como entrada el conjunto Φ, compuesto por los candidatos electos para cada píxel p. Estos píxeles han sido escogidos mediante la Ecuación 11, siendo estos los primeros valores que toma Ω en el paso de inicialización. Así, se llama $\varphi^0(p)$ al mejor candidato q que sustituyen a cada píxel p. Nótese que que el algoritmo *k-Inpainting* es iterativo y a partir de $\varphi^0(p)$ es cuando los términos de energía son estimados.

Para procesar una textura hay que sintetizarla. Para esto, se utiliza una variación de la versión original de K-NNF (Obtener candidatos) con r iteraciones, donde se utiliza la Ecuación 2 en lugar de la Ecuación 11. Esto es para refinar el procesamiento de la textura y obtener un mejor acabado.

Los candidatos a ser evaluados provienen del conjunto $\Phi(p)$ para cada p, como se mencionó anteriormente y estos son actualizados en cada iteración (ver Algoritmo 3).

Algorithm 3 Algoritmo para procesar textura

```
1: r = 0                                  ▷ Sea valor el número de iteraciones
2: while r < valor do ▷ Aplicar el algoritmo para Obtener candidatos usando como candidatos a los q ∈ Φ(p)
3:     while Ω ≠ ∅ do
4:         Seleccionarδ Ω_j
5:         for cada píxel p ∈ δΩ_j do
6:             λ = Kpropagacion(Φ(p))
7:             BusquedaAleatoria(λ, Φ(p))        ▷ Φ(p) se actualiza en la Búsqueda Aleatoria
8:             ι = φ(p)
9:         end for
10:    end while
11:    r = r + 1
12: end while
```

Basados en una experimentación realizada por los autores, cuando r toma como valor máximo 5, se obtienen resultados satisfactorios. Sin embargo este valor puede variar de acuerdo a característica particulares de la imagen. Los puntos p pertenecientes a Ω siempre se toman de afuera hacia dentro.

D. Aplicar Función de Energía

Para aplicar la función de energía, se generan las imágenes de difusión v y coherencia w. Para este momento debe existir una matriz o conjunto de *max-heap* Φ como se expresa en la Ecuación 12 a continuación.

$$\Phi|\Phi[p.X][p.Y] \vee \Phi(p) \tag{12}$$

Dicha ecuación es equivalente a los mejores N candidatos para un $p \in \Omega$. Estos candidatos han sido escogidos empleando pasos que emplean solo el primer término de energía para mejorar los candidatos mediante un refinamiento de la textura, tal como el paso 3 (i.e. Procesar textura) y paso 5 (i.e. Proyección) de la Figura 4.

Luego, se intenta mejorar estos candidatos calculando las magnitudes de las ecuaciones 3, 5 y 6 para todos los píxeles $q \in \Phi(p)$, y las constantes α, β y γ. Posteriormente, se calcula la Ecuación 8 actualizando la lista de $\Phi(p)$, tomándose como valor de distancia el dato calculado por la función de energía.

Seguidamente, se toma el mejor valor de cada $\Phi(p)$ para cada píxel $p \in \Omega$, es decir $\varphi(p)$, y se actualiza la imagen a tratar. Una vez actualizada la imagen, se vuelve a calcular las imágenes de difusión (v) y coherencia (w) y se repite el proceso antes descrito, y así sucesivamente hasta que se hayan cumplido un número de n iteraciones. Para diferenciar las iteraciones del procesamiento de textura de éstas iteraciones, aquí se habla de n iteraciones en vez de r iteraciones.

La manera como los píxeles p son seleccionados, es de afuera hacia adentro, siguiendo un esquema de capas y contando con una máscara auxiliar que permita distinguir los píxeles que faltan por procesar de los que no. Una vez procesados todos los píxeles, a la máscara auxiliar se le asigna la máscara original para repetir el proceso de aplicar la función de energía en la siguiente iteración. En este punto, es donde el proceso iterativo es fundamental en nuestra propuesta.

El número de iteraciones n para esta parte del algoritmo es un parámetro fijo que puede ser manipulado por el usuario, sin embargo para $n = 10$ se obtienen buenos resultados [4].

En *k-Inpainting* esta variable puede tomar otros valores que serán mostrados en la sección VI. Es importante destacar, que para la primera iteración del algoritmo la Ecuación 3 no es calculada, si no que se toma el valor que se obtuvo del paso anterior, el cual ha buscado los mejores candidatos mediante este primer término de energía.

E. Proyección

En una proyección es natural proyectar cuatro puntos a una imagen grande a partir de un punto de la imagen de menor tamaño. Sin embargo, ese tipo de proyección puede provocar que se procesen puntos de Ω en un orden inadecuado, esto causa que se presenten problemas en el algoritmo *k-Inpainting*.

Entonces, la proyección se hace a partir de la imagen de mayor tamaño a la de menor tamaño, es decir, se toma un punto $p \in \Omega \subset I$ y se busca el punto correspondiente en I'. Una vez encontrado ese punto en I', se proyectan ahora de la forma $I' \Longrightarrow I$ los candidatos $q' \in \Phi(p'), p' \in \Omega \subset I'$, donde cada q' proyecta cuatro puntos $q \in \Omega^c \subset I$.

Como no es deseable aumentar la cantidad de candidatos mediante la Ecuación 11, se calcula cuál de los cuatro puntos proyectados es el nuevo candidato para p. El mejor q para cada p, viene a ser $\varphi(p)^0$ para esta imagen y reemplazan en una iteración inicial a los valores incógnitas $p \in \Omega \subset I$, es decir a $I(p) = \varphi(p)^0$.

Posteriormente, como indica la Figura 9, se aplica una síntesis de textura con el algoritmo para *Obtener candidatos* con el primer término de energía. Sin embargo, como ya se tiene una buena estimación espacial, el algoritmo es realizado una única vez, es decir, no se itera r veces.

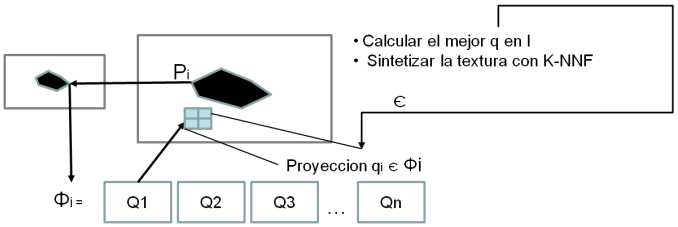


Figura 9: Esquema de Procesamiento de la Etapa de Proyección

Tabla I: Descripción de los Sistemas Utilizados

Sistema	Procesador	RAM
1	Intel Core i3 3.07Ghz	4 GB
2	Intel Core 2 Quad 2.40 GHz	4 GB
3	Intel Pentium 4 3.20 GHz	1 GB

Tabla II: Descripción Tarjetas Gráficas Utilizadas

Sistema	Tarjeta	Memoria	Núcleos
1	GeForce GTX 470	1280 MB	448
2	GeForce GT 8800	640 MB	112
3	GeForce GT 9600	512 MB	64

Anteriormente se menciona que para cada imagen son almacenados los puntos máximos y mínimos con respecto a la imagen anterior, así como sus módulos. Esto es para hacer una buena correspondencia de puntos en el momento de proyectar. Sea p un punto dentro de $\Omega \subset I$ y sea $pmin$, $pmax$, $pminmod$ y $pmaxmod$ los puntos máximos y mínimos de Ω con sus correspondientes módulos, para buscar el punto p' correspondiente dentro de Ω de I', se calcula como $p' = (p - m - pminmod)/2$, donde m es un punto dentro de la ventana $[0, 0] \times [1, 1]$.

Luego de aplicar la proyección, Φ ha sido redimensionado y actualizado con los mejores candidatos para $\Omega \subset I$, que fueron escogidos primero con la Ecuación 11 y luego mejorados con la Ecuación 2. Buscando mejorar los candidatos, se aplica nuevamente la función de energía, tal como se ilustra en el paso 6 de la Figura 4, y finalmente se obtienen los resultados de la imagen.

F. k-Inpainting en la GPU

El algoritmo de *k-Inpainting* obtiene buenos resultados en una versión secuencial, sin embargo su tiempo de ejecución es extenso. Por ello, se planteó realizar una versión paralela en la GPU empleando la arquitectura CUDA provista por las tarjetas gráficas de Nvidia. Entonces, cada etapa del algoritmo se ejecuta en paralelo en sus sub-etapas, a excepción de *Obtener datos de entrada*.

Una diferencia importante con respecto al diagrama secuencial, es la encapsulación de las sub-etapas *K-Propagación* y *Búsqueda Aleatoria* en una etapa llamada *K-NNF*. En la etapa de *Procesamiento de textura*, aplicando un ciclo de r iteraciones para que sea ejecutado una vez si se procesa con todas las muestras o r veces si se paraleliza.

VI. Pruebas y Resultados

En esta sección se presentan las diferentes pruebas realizadas para reconstruir el área desconocida de cada imagen, así como el ámbito de ejecución de las mismas, para finalmente realizar un análisis de los resultados obtenidos.

El algoritmo fue probado en tres equipos con especificaciones de hardware distintas (ver Tabla I). Igualmente, el hardware gráfico constó de tarjetas Nvidia con soporte CUDA (ver Tabla II).

El lenguaje de programación C# fue empleado para la GUI y el algoritmo *k-Inpainting* sobre C++. El ambiente de trabajo fue Visual Studio sobre el sistema operativo Windows. Para la manipulación inicial de las imágenes se empleó la biblioteca EmguCv. Igualmente, se utiliza el API de CUDA para C++.

A. Escenarios

Se efectuaron experimentos en diferentes niveles. Un primer nivel hace un estudio a fondo de los parámetros, realizando tantas ejecuciones como resulten de la combinación de todos éstos. Luego, en un segundo nivel se elaboran pruebas con las mejores combinaciones obtenidas de los experimentos de primer nivel en 4 imágenes distintas.

Existen diferentes parámetros que afectan el resultado luego de aplicar el algoritmo en cualquiera de sus versiones. Algunas de éstas variables tienen influencia significativa en el producto final. Dichos valores son: el lado de la ventana (L); el parámetro de propagación (K); la cantidad de candidatos (N); la cantidad de iteraciones para el procesamiento de textura (r); y la cantidad de iteraciones para aplicar la función de energía (n).

Por otro lado, se implementa el algoritmo secuencial (totalmente en la CPU); otro algoritmo en la GPU donde se ejecuta cada etapa a la vez, pero teniendo como entrada las muestras que se emplearon en la etapa de inicialización (GPU T1); e iterando r veces en las sub-etapas que la componen, teniendo como entrada a dicha etapa los mejores candidatos seleccionados en la etapa anterior (GPU T2).

B. Experimentos (Nivel 1)

Los experimentos de nivel 1 prueban el algoritmo considerando todas las combinaciones posibles de los diferentes parámetros descritos anteriormente. Para este estudio se realizaron 7680 ejecuciones, producto de las 384 combinaciones que surgen a partir de la composición de dichos parámetros, multiplicado por 20 para obtener un promedio.

El tiempo de ejecución del algoritmo depende de tres factores: la complejidad del algoritmo, el tamaño de la imagen y el tamaño del agujero o zona a reconstruir. Debido a la cantidad de pruebas a realizar y a la complejidad del algoritmo, se decide emplear una imagen relativamente pequeña de dimensiones 228×250 píxeles con un orificio pequeño. El objetivo de este experimento es elegir el sub-conjunto de las mejores combinaciones de los diferentes valores, para luego aplicar estas combinaciones a otros experimentos.

Entonces, se hace un estudio métrico donde se necesita una imagen de referencia que sea correcta, tomando una imagen

sin ningún tipo de pérdida de información, como la Figura 10a y se edita para causar pérdida de información como se ve en la Figura 10b. Luego, se hace una selección del área dañada como se observa en la Figura 10c y esa es la región a reconstruir. Con la Figura 10a, es posible tener un punto de referencia adecuado y realizar una diferencia píxel a píxel con cada imagen resultado.

(a) Original (b) Editada (c) Selección

Figura 10: Procedimiento Realizado Para los Experimentos en la Imagen Tomada del Nivel 1

Para evaluar los resultados de la imagen se calcula la diferencia promedio entre la imagen 10a y la imagen reconstruida a partir de 10c en la región tratada con la técnica. Dada la imagen original O y la imagen resultante R, se calcula en el espacio L*a*b la diferencia euclidiana promedio de los η píxeles sobre Ω como:

$$Dif(\varphi) = \frac{\sum_{i \in \Omega} \|\iota(O_i) - \iota(Ri)\|^2}{\eta} \quad (13)$$

En la Tabla III, se muestra el tiempo que tardó cada versión en alcanzar el resultado visual presentado en la Figura 11.

Tabla III: Tiempos de Ejecución para cada Versión

Versión	Tiempo	Parámetros
Secuencial	3 m y 19 s	$L = 9$, $K = 8$, $r = 3$, $n = 10$
GPU T1	1 m y 18 s	$L = 9$, $K = 32$, $n = 5$
GPU T2	33 s	$L = 5$, $K = 64$, $r = 3$, $n = 10$

(a) (b) (c)

Figura 11: Resultados Visuales de los Diferentes Algoritmos *k-Inpainting*

En la Figura 12 se puede visualizar el tiempo que tardó la versión secuencial en ejecutar las 384 combinaciones, el tiempo en ejecutarse las 192 combinaciones de la implementación *GPU T1* de la versión paralela y el tiempo en ejecutarse las 384 combinaciones en la implementación *GPU T2* de la versión paralela.

C. Experimentos (Nivel 2)

En los experimentos nivel 2, se tratarán las imágenes agrupadas en la Figura 13. Dichas imágenes serán procesadas

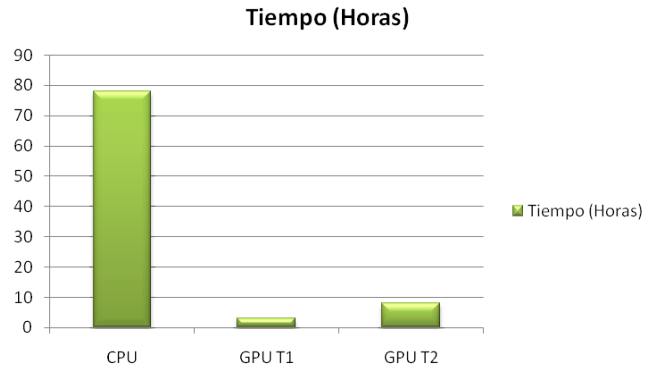


Figura 12: Tiempos de Ejecución de las Versiones de *k-Inpainting* en Reconstruir el Área Desconocida

con los parámetros escogidos en los experimentos de nivel 1. Sin embargo, solo se presentará la implementación GPU T1 dada la extensión de los resultados obtenidos por las otras implementaciones. Sin embargo, en [21] se puede explorar todas las pruebas realizadas. Como las imágenes son distintas, pueden existir casos especiales, en los cuales se deba variar el valor del tamaño de la ventana L o el valor de propagación K. Por otro lado, la Tabla IV resume las características de las imágenes a procesar.

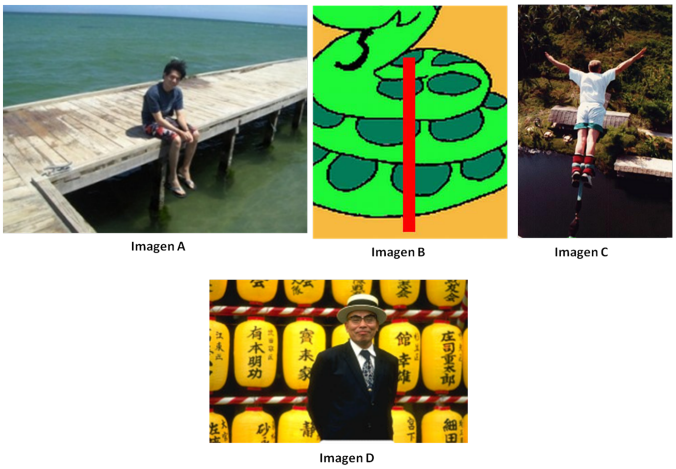


Figura 13: Imágenes de Prueba Para los Experimentos Nivel 2

Tabla IV: Características de las Imágenes a Procesar en los Experimentos Nivel 2

Imagen	Dimensiones en píxeles
A	301×226
B	292×291
C	206×308
D	256×162

Las imágenes serán tratadas por el orden indicado en la Figura 13, es decir, siguiendo la secuencia alfabética A,B,C,D. La primera imagen a procesar es la imagen A, la segunda imagen con la cual experimentar es la B y así sucesivamente.

1) Imagen A: En la imagen A, se observa una persona sentada en un puente. La idea es remover a dicha persona para

ver el paisaje de la fotografía. Luego que el usuario selecciona el área a restaurar rodeando al individuo en la foto, en la imagen queda un agujero. Este agujero está constituido por 6.398 píxeles, que forman el 9,4% de la imagen.

En la Tabla V se despliega el subconjunto de parámetros para procesar la región desconocida de la imagen con su respectivo tiempo de ejecución. La Figura 14 muestra los resultados obtenidos para cada combinación, donde se observa que los resultados son muy parecidos entre las imágenes 2, 3, 4, 5 y 6, por consiguiente el mejor resultado es aquel que tomó menos tiempo de ejecución, es decir, para este caso, el mejor resultado corresponde a la imagen 6.

Tabla V: Diversos Parámetros y Tiempo de Ejecución de la Imagen A

Resultado	Porcentaje	L	K	n	Tiempo
1	0,1%	9	16	10	6 minutos
2	0,05%	9	32	5	10 minutos
3	0,05%	5	64	5	8 minutos y 40 segundos
4	0,05%	9	32	10	10 minutos y 10 segundos
5	0,05%	9	64	5	27 minutos
6	0,1%	5	16	10	1 minuto y 40 segundos

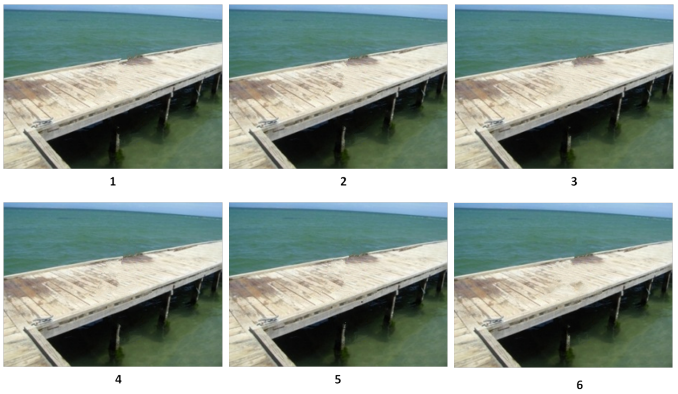


Figura 14: Resultados Luego de Procesar la Imagen A con los Parámetros de la Tabla V

Nótese la percepción visual de cada una de las imágenes con respecto a la restauración del área que se desea eliminar. La capacidad de reconstrucción visual es notable, mostrando un buen resultado el algoritmo.

2) Imagen B: La imagen B tiene un área dañada (rectángulo rojo) en la parte central de la imagen, afectando de forma transversal la estructura de la serpiente en dicha imagen. En este caso se busca restaurar la región. Luego que el usuario selecciona el área a tratar rodeando el área dañada, en la misma queda un orificio. Este orificio está constituido por 5.617 píxeles, que forman el 7,9% de la imagen B.

El subconjunto de parámetros se presentan en la Tabla VI para esta imagen. En la Figura 15 se observa el resultado para cada conjunto de parámetros en la tabla. Los resultados son muy parecidos entre las primeras cuatro imágenes, por lo que se selecciona como mejor resultado la que se ejecuta en menor tiempo entre ellas, que corresponde a la imagen 2.

Tabla VI: Diversos Parámetros y Tiempo de Ejecución de la Imagen B

Resultado	Porcentaje	L	K	n	Tiempo
1	0,1%	11	8	10	5 minutos
2	0,05%	11	8	10	3 minutos 45 segundos
3	0,05%	9	32	5	9 minutos
4	0,05%	9	32	10	9 minutos y 20 segundos
5	0,05%	5	64	10	8 minutos y 17 segundos
6	0,1%	9	16	10	5 minutos y 25 segundos

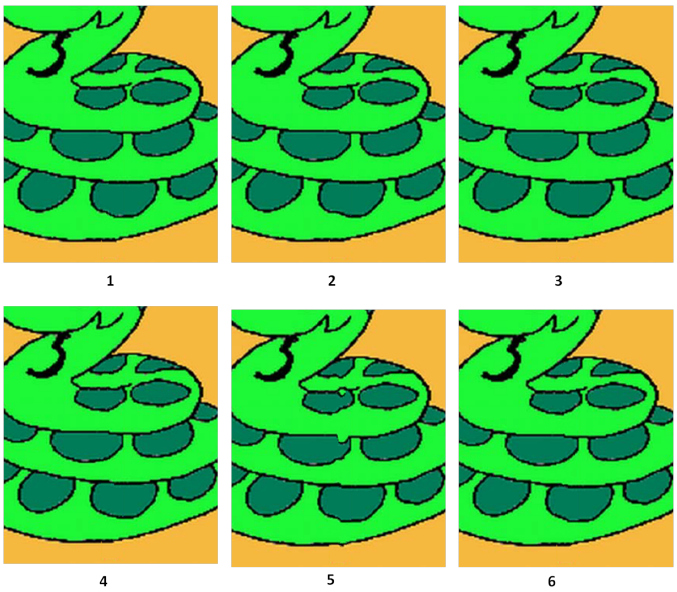


Figura 15: Resultados Después de Aplicar la Primera Versión del Algoritmo Paralelo con los Parámetros de la Tabla VI

3) Imagen C: En la imagen C se busca remover la persona que bloquea el paisaje, la cual forma el agujero que tiene 12.117 píxeles y constituye el 19,09% de la imagen.

La recopilación de los parámetros de esta ejecución se muestra en la Tabla VII, mostrando sus resultados visuales en la Figura 16.

Tabla VII: Diversos Parámetros y Tiempo de Ejecución de la Imagen C

Resultado	Porcentaje	L	K	n	Tiempo
1	0,1%	5	16	10	2 minutos y 41 segundos
2	0,1%	11	8	10	7 minutos
3	0,1%	9	32	10	16 minutos

4) Imagen D: Se busca remover el individuo presente en la foto y reconstruir el patrón detrás de él. Esta imagen es compleja, ya que el tamaño del agujero representa un porcentaje importante y tiene un fondo complicado. El agujero que queda en la imagen luego de seleccionar el área a tratar ocupa el 26,67% de la imagen con 11.061 píxeles. A continuación se despliega en la Tabla VIII los parámetros utilizados para procesar esta imagen con cada variante del algoritmo.

A diferencia de las imágenes anteriores, solo la primera combinación es mostrada, debido a que la imagen arroja mejores resultados con una ventana $L = 11$ para cualquier versión. Los parámetros de la segunda y tercera combinación de la tabla se escogieron a partir del resultado obtenido para la

1 2 3

Figura 16: Resultados Luego de Procesar la Imagen C con los Parámetros Indicados en la Tabla VII

Tabla VIII: Diversos Parámetros y Tiempo de Ejecución de la Imagen D

Versión	Porcentaje	L	K	r	n	Tiempo
CPU	0,05%	11	16	5	10	2 horas
GPU T1	0,05%	11	32	-	10	44 minutos
GPU T2	0,05%	11	32	5	5	50 minutos

primera. En la Figura 17 se observan las imágenes obtenidas a partir de estos parámetros en su proceso iterativo.

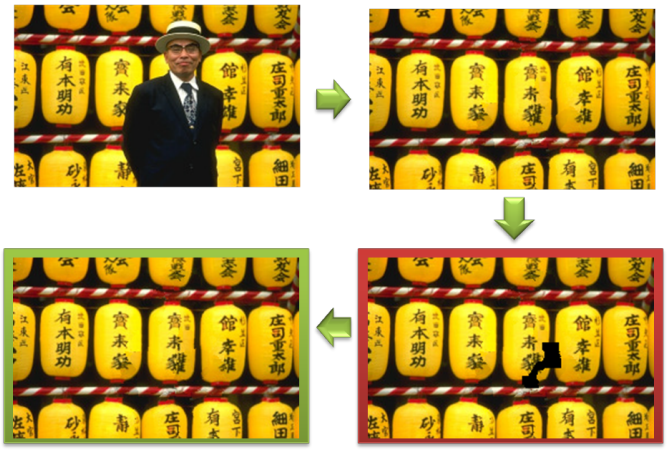


Figura 17: Diversas Etapas Iterativas Aplicadas en la Imagen D

Los resultados aquí presentados, fueron corroborados a su vez por una serie de encuestas para determinar la percepción visual de usuarios regulares antes las imágenes presentadas. Dicho estudio ayudo a distinguir qué imágenes (y bajo qué parámetros) se puede considerar a una imagen "visualmente agradable" con discontinuidades no notorias a simple vista.

VII. Consideraciones Finales

En este trabajo se presentó un nuevo algoritmo híbrido denominado *k-Inpainting* que se basa en dos trabajos concebidos de manera distinta, incluyendo modificaciones a dichos trabajos para acoplarse en un mismo algoritmo. Los trabajos base, manejan la síntesis de textura basada en parches [19], [8], o basada en píxeles [4], y adoptándo para el proceso de inicialización un campo aleatorio de Markov, la cual es una característica común en los algoritmos de síntesis de textura basada en píxeles. Básicamente, se sustituye la inicialización mediante un campo aleatorio de Markov, por un algoritmo de síntesis de textura para procesar la misma mediante parches, pero adaptándo este último a su vez a píxeles.

k-Inpainting provee buenos resultados y logra el objetivo de la técnica. La implementación funciona eficientemente cuando no hay partes importantes del agujero en los bordes de la imagen, completa bien los contornos, realiza una buena síntesis de textura, y no es estrictamente necesario aplicar una técnica de difusión al área que ha sido tratada o un post-procesamiento adicional. Adicionalmente, su rapidez depende más de las dimensiones de la región Ω a ser reconstruida y de los parámetros que del tamaño mismo de la imagen. Los parámetros que más influyen en el tiempo son las dimensiones del sistema de vecindad, el parámetro de propagación y el parámetros N. Hasta ahora no existe para *k-Inpainting* una combinación que funcione para todas las imágenes, por lo que sería necesario estimar estos valores también de forma automática. Un aspecto importante es que la selección a mano alzada no es la ideal, pues se pierden píxeles que podrían ayudar a reconstruir mejor el área a tratar.

En esta propuesta solo se contemplan las imágenes, sin embargo, pensando en una extensión para video, cuando se desea retirar un cuerpo en movimiento, se proveen más datos a las imágenes del video a reconstruir, ya que a través de las imágenes en otros momentos del mismo se puede extraer información. Por ello se considera que la reconstrucción de imágenes estáticas tiene una mayor complejidad al reconstruir un área donde había un cuerpo en movimiento.

El cálculo de la diferencia de vecindades requiere tiempo computacional cuando se procesa una cantidad considerable de píxeles. Este fenómeno ocurre incluso en las variantes paralelas, ya que esta función es secuencial en sí misma (i.e. un hilo a la vez).

Trabajos Futuros

Diversos cambios se proponen con el objetivo de mejorar nuestra propuesta. Hasta ahora, se ha construido el algoritmo de forma secuencial y dos variantes paralelas. También se ha construido una interfaz sencilla para que el usuario seleccione el área a tratar. Entre dichos cambios se encuentran:

- Elaborar otras herramientas para el usuario, como ofrecer una herramienta precisa de selección de la región a tratar.
- Hallar un método automático para detectar los parámetros más adecuados de una imagen de entrada.
- Incluir el uso de OpenMP o Numa en el algoritmo secuencial para acelerar los tiempos de respuesta.
- Emplear estructuras de datos que exploten más la capacidad de la tarjeta gráfica.

Para acelerar el tiempo de respuesta, se propone elaborar una versión basada en parches, en vez de basada en píxeles. En cuanto a la implementación, se propone migrar el código actual desarrollado en CUDA a un ambiente independiente de la arquitectura como Compute Shaders.

REFERENCIAS

[1] J. F. Aujol, S. Ladjal, and S. Masnou, *Exemplar-Based Inpainting From a Variational Point of View*, SIAM Journal on Mathematical Analysis, vol. 43, no. 3, pp. 1246-1285, 2010.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, *Image Inpainting*, in Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00), pp. 417-424, 2000.

[3] S. Li, *Markov Random Field Modelling in Image Analysis*, pp. 353-354, Springer, Londres, 1999.

[4] A. Bugeau, M. Bertalamio, V. Caselles, and G. Shapiro, *A Comprehensive Framework For Image Inpainting*, IEEE Transactions on Image Processing, vol. 19, no. 1, pp. 2634-2645, 2010.

[5] M. Taschler, *A Comparative Analysis of Image Inpainting Techniques*, Tesis de PhD, The University of York, 2006.

[6] R. Suthar and K. Patel, *A Survey on Various Image Inpainting Techniques to Restore Image*, International Journal of Engineering Research and Applications, vol. 4, no. 2, pp. 85-85, 2014.

[7] J. Joshua and G. Darsan, *Digital Inpainting Techniques - A Survey*, International Journal of Latest Research in Engineering and Technology (IJLRET), vol. 2, no. 1, pp. 34-36, 2016.

[8] C. Barnes, E. Shechtman, D. Goldman, and A. Finkelstein, *The Generalized Patchmatch Correspondence Algorithm*, in Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III, pp. 29-43, 2010.

[9] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, *PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing*, ACM Transactions on Graphics, vol. 28, no. 3, pp. 24:1, 24:11, 2009.

[10] K. Plataniotis and A. Venetsanopoulos, *Color Image Processing and Applications*, pp.32-39, Springer-Verla, Berlín, 2000.

[11] L-Y. Wei and M. Levoy, *Fast Texture Synthesis Using Tree-Structured Vector Quantization*, in Proceedings of the 27th annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00), pp. 479-488, 2000.

[12] N. Komodakis, *Image Completion Using Global Optimization*, in Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 442-452, 2006.

[13] K. Cao, K. Ding, G. Christensen, M. Raghavan, R. Amelon, and J. Reinhardt, *Unifying Vascular Information in Intensity - Based Non-Rigid Lung CT Registration*, Biomedical Image Registration, pp. 5-6, Springer, Alemania, 2010.

[14] P. Burt and E. Adelson, *The Laplacian Pyramid as a Compact Image Code*, IEEE Transactions On Communications, vol. 31, pp. 532-540, USA, 1983.

[15] R. Sedgewick, *Algoritmos en C++*, pp. 555-561, Addison - Wesley / Diaz de Santos, USA, 1995.

[16] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*, 2da edición, Cambridge, USA, 2002.

[17] L. Demanet, B. Song, and T. Chan, *Image Inpainting by Correspondence Maps: a Deterministic Approach*, reporte técnico, Instituto Tecnológico de California y Universidad de California, Los Angeles, USA, 2000.

[18] A. Efros and T. Leung, *Texture Synthesis by Non-parametric Sampling*, in Proceedings of the Seventh IEEE International Conference on Computer Vision, IEEE Computer Society, 1999.

[19] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, *PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing*, ACM Transactions on Graphics, vol. 28, no. 3, 2009.

[20] S. Shin, T. Nishita, and S. Shin, *On Pixel-Based Texture Synthesis by Non-Parametric Sampling*, Computer & Graphics, vol. 30, no. 5, 2006.

[21] K. Pedrique, *Un Enfoque Híbrido del Algoritmo Inpainting bajo CPU y GPU*, Tesis de Pregrado, Universidad Central de Venezuela, 2011.

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