DOI: http://dx.doi.org/10.21501/21454086.1967



*Lámpsakos* | No.17 | pp. 69-76 | enero-junio | 2017 | ISSN: 2145-4086 | Medellín–Colombia

# Construcción de un repositorio de activos de software para el desarrollo agil de aplicaciones aplicando un método para el reuso

## *Construction of a repository of assets agile development software for applying a method of applications for reuse*

**Yeimar Alonso Castro**
*Grupo SODA, Universidad Cooperativa de Colombia*
*Medellín, Colombia*
yeimar.castro@campusucc.edu.co

**Javier Darío Fernández Ledesma, MSc.**
*Grupo SODA, Universidad Cooperativa de Colombia*
*Medellín, Colombia*
javier.fernandez@ucc.edu.co

**Julián Alberto Rivera**
*Grupo SODA, Universidad Cooperativa de Colombia*
*Medellín, Colombia*
julian.riverao@campusucc.edu.co

**Eder Acevedo Marin, MSc.**
*Grupo SODA, Universidad Cooperativa de Colombia*
*Medellín, Colombia*
ederaam@gmail.com

(Recibido el 24-05-2016, Aprobado el 20-09-2016, Publicado el 17-01-2017)

Estilo de Citación de Artículo:
Y. Castro, J. Rivera, J. Fernández, E. Acevedo, "Construcción de un repositorio de activos de software para el desarrollo ágil de aplicaciones aplicando un método para el reúso", Lámpsakos, no. 17, pp 69-76, 2017
DOI: http://dx.doi.org/10.21501/21454086.1967

**Resumen** – En este artículo de investigación se expone el estudio y la construcción de un repositorio de activos de software para el desarrollo ágil de aplicaciones aplicando un método para el reúso que permite manejar, manipular, crear, almacenar, recuperar y reutilizar diversas fuentes de códigos y activos de software para la agilización de procesos sistemáticos con el fin de crear cimientos en procesos industriales que requieran la intervención directa de un software.

Este software ha sido desarrollado luego de una amplia investigación y recopilación de antecedentes, una amplia arquitectura de reglas de manejo, enfoque en minería de datos y propósitos de promover el reúso de activos de software como una importante metodología dentro de la ingeniería, así con dicha implementación de esta herramienta se busca explorar, ayudar, fundamentar y respaldar en etapas tempranas de formación de un software, donde el campo de interés radica en pequeñas y medianas empresas de software que necesitan una metodología y una herramienta elástica para la mejora de procesos en sus instalaciones sin escatimar la calidad que del producto final que definitiva es el intangible "software".

**Palabras clave**: Activos, metaproceso, repositorio, reúso, software

**Abstract** - In this research paper the study and construction of a repository of software assets for agile development of applying a method to reuse that allows you to manage, manipulate, create, store, retrieve and reuse various sources of codes and active applications exposed software for streamlining systematic processes in order to create foundations in industrial processes requiring direct intervention of a software.

This software has been developed after extensive research and gathering background, a wide architecture management rules, focus on data mining and purposes of promoting the reuse of software assets as an important methodology in engineering, and with that implementation of this tool seeks to explore, help, inform and support in the early stages of forming a software, where the field of interest lies in small and medium software companies that need a methodology and an elastic tool for process improvement in their without skimping quality facilities that the final product is the ultimate intangible "software".

**Keywords**: Assets, metaprocess, repository, reuse, software

Yeimar Alonso Castro; MSc. Javier Darío Fernández Ledesma
Julián Alberto Rivera; MSc. Eder Acevedo Marin
DOI: http://dx.doi.org/10.21501/21454086.1967

## 1. INTRODUCCIÓN

Dentro del ámbito de la industria del desarrollo de software han existido relevantes inconvenientes a la hora de hablar de la reutilización de software. Para esto, con el transcurrir de los tiempos, se han venido investigando y adecuando cierta serie de metodologías para la planeación, construcción y retroalimentación de los proyectos de software, llegando a obtener resultados dentro del desarrollo basado en componentes.

En este artículo se relacionan antecedentes referentes a los repositorios de software propios del reúso y el manejo de activos, dentro de la composición de proyectos de software y la utilización de un método de reúso de activos que facilita y clarifica el proceso de desarrollo de un nuevo proyecto, siendo construido con artefactos de fases de proyectos disponibles para el reúso.

En este artículo se presenta desde los antecedentes la problemática central del reúso de activos de software, que se centra en la forma inadecuada para la recuperación de los artefactos de software que se requieren. Seguidamente en el marco conceptual se expone el análisis, diseño, metodología usada y el modo en el que se desarrolló un repositorio de activos de software, desde su investigación hasta los resultados de su desarrollo para dar solución a la problemática analizada.

## 2. ANTECEDENTES

La creación de un repositorio de activos de software para el desarrollo ágil de aplicaciones, aplicando un método para el Reuso a priori, suena como un concepto de nueva era o distante a esta época, pero realmente no lo es. Existen varios antecedentes en la historia que muestran la evolución, reinvención y adecuación del concepto actual, el cual se redefine para dar solución a un medio creciente como lo es el de la industria del software y a sus procesos. Todo data desde la década de los setenta cuando se hablaba de la reutilización de componentes de software en los procesos.

Inspirado en la forma en que se desarrollan y construyen los módulos en la ingeniería de sistemas, muchos autores de hoy en día, consideran que el software actualmente permea en gran medida las actividades del hombre, llegando a niveles de complejidad que han originado la llamada “crisis del software” [1].

Gracias a dicha “crisis” nacida en los 70’s, se crearon diversas metodologías para el desarrollo de software tales como: Programación estructurada SOL (años 70s), que básicamente era “la idea básica de que todo comportamiento secuencial puede modelarse por medio de un autómata finito”; Ingeniería de la información (años 80s); Programación orientada a objetos (años 90s) el cual se basa en las interacciones de objetos representado de la vida real; y Proceso Unificado Ágil (RUP), del nuevo milenio, que consiste en describir de una forma simple y sencilla de entender el proceso de desarrollo de aplicaciones de software de negocio usando técnicas ágiles.

La reutilización de activos y el uso de componentes en el entorno de la ingeniería de sistemas, consiste en la capacidad de una herramienta o producto, de ser reciclada en lo posible, para el desarrollo de sistemas y aplicaciones. Algunos ejemplos de reutilización de activos se ven reflejados en algoritmos, patrones de diseño, artefactos, esquemas de base de datos, arquitecturas de software, especificaciones de requerimientos, de diseño y de prueba, entre otros. En este contexto surgen propuestas como UML, BPMN y tecnologías como XML, que soportan enfoques de desarrollo orientados a la transformación de modelos como es el caso de Model Driven Arquitecture (MDA) [3], hacia la producción en masa de software o el enfoque de líneas de productos de software.

En el pasado siglo, difícilmente se podía representar en un repositorio, artefactos de software por su contenido, lo cual no permitía que cualquiera pudiese acceder a ellos sin conocimiento previo de su existencia. Por este motivo, para el reúso se realizó el proceso inverso: primero detener la organización para ver qué podía ser reutilizable y después hacerlo reutilizable. Lo siguiente era obligar a que todos en la organización conocieran lo que existe para reutilizar (puesto que si no se hacía esto, no sabrían que cosa existe), y luego intentar obligar a que reutilizaran lo existente, bien sea en forma de patrones, líneas de producto o Frameworks.

Para el reúso, en la actualidad se procede de manera mucho más flexible, evitando restricciones en lo concerniente (requisitos, diagramas UML, pruebas, manuales, código fuente, riesgos, planificaciones de proyectos, experiencias post-mortem, reglas de negocio, personas, etc..), hasta el punto de usar artefactos reutilizables solamente cuando se sabe que se van a reutilizar al menos una vez. Así que cada vez que se habla de reúso de software y de diseños de software enseguida se nos vienen a la cabeza los patrones de diseño.

DOI: http://dx.doi.org/10.21501/21454086.1967

"La idea de los patrones fue introducida por Christopher Alexander [ALEX] en el dominio de la arquitectura, aunque hace ya años que la idea ha sido acogida en el diseño de software. "Cada patrón describe un problema que ocurre una y otra vez en un entorno dado, describiendo el núcleo de la solución al problema de tal forma que pueda ser empleada un millón de veces sin hacerlo dos veces de la misma forma" [4], en donde hacen una clasificación en patrones creacionales, estructurales y de comportamiento.

En la actualidad se han dado cambios influyentes en la industria del software a la hora de hablar de costo y tiempo de desarrollo de sistemas y aplicaciones. Gracias a esto ha surgido a vista de la ingeniería un nuevo activo reutilizable denominado componente de Software.

Esta breve reseña refleja importancia la reutilización de activos de software en nuestro entorno cotidiano.

## 3. METODOLOGIA

La investigación previa a la elaboración y presentación del resultado del producto bautizado ***"Activos"***, Fig. 1, contó con una secuencia de recopilación de información y documentación de fuentes valiosas y confiables con renombre y amplio campo de acción en el modelado y desarrollo de este tipo de software especializado. En particular, se estudiaron las metodologías y métodos de reúso y los componentes tecnológicos de los repositorios de activos de software. Esto dio paso para obtener las bases conceptuales del reúso y el uso de los repositorios de activos de software para el desarrollo ágil de aplicaciones desde sus etapas más tempranas.


**Fig. 1.** Pantalla de bienvenida al aplicativo

Como siguiente medida se optó por observar una problemática específica de un sector en particular como lo es la industria. Seguidamente se halló una solución adecuada a partir de la ingeniería de sistemas, independiente de ser interdisciplinaria. El foco de esta investigación y su posterior resultado y/o aplicativo está enfocado netamente al desarrollo de un software que atendiera las necesidades para lo que fue creado.

Dentro del proceso de construcción de ***"Activos"***, luego de haber obtenido un conocimiento más amplio acerca del reúso de artefactos y las técnicas utilizadas para el desarrollo de métodos de gestión de activos de software, se procedió a definir el diseño de la herramienta haciendo uso del lenguaje notacional UML y BPMN.

Teniendo definido el diseño de la herramienta se hizo la elección del lenguaje de programación que se utilizó en la codificación de las herramientas de apoyo para las pruebas del método y la construcción del repositorio teniendo en cuenta factores de compatibilidad y rendimiento.

Seguidamente, se procedió a la construcción del prototipo el cual consistió en la integración de herramientas CASE (Computer-Aided Software Engineering) para el diseño de sistemas con el módulo de reúso diseñado en el proyecto y el método propuesto. Como paso siguiente se hizo una búsqueda de los mecanismos de prueba más utilizados para aplicárselos al método propuesto y el repositorio desarrollado, y se realizó un esquema de resultados y se compararon con sistemas similares.

***"Activos"*** cumple básicamente tres necesidades: La recopilación de artefactos para solucionar un problema; la manipulación y reciclaje o reúso en diversos contextos; y ser duraderos en el tiempo, transformables y personalizable en ejecución. Estas tres necesidades son fundamentales para entender el proceso de negocio que en otras palabras son las reglas que establecen el control y la realización de los requisitos los cuales está dirigido el aplicativo.

Para entender el modelado de procesos de negocio en este contexto se necesita ser realmente muy gráfico, didáctico y pragmático. Normalmente se recurre a los diagramas de flujo, representación de nodos o actividades a ejecutar, lo que en ***"Activos"*** es recurrente y habitual. Se somete a una serie de reglas que posteriormente serán explicadas y desglosadas debidamente. Dichas reglas permiten que los elementos, dominio, fases, y patrones, interactúen, donde el dominio es una materia especifica o contexto especifico (por ejemplo medicina, biología, fisiología, química, informática, robótica, o automatización), siendo este contexto

Yeimar Alonso Castro; MSc. Javier Darío Fernández Ledesma
Julián Alberto Rivera; MSc. Eder Acevedo Marin
DOI: http://dx.doi.org/10.21501/21454086.1967

especifico el eje y el engranaje para el reutilización de múltiples y variables dominios para la construcción de un procesos adyacente, proponen Pérez et al. [2].

Este paradigma combina los siguientes conceptos:

a) Lenguajes de dominio específico (DSL) usados para formalizar la estructura de la aplicación, el comportamiento y los requisitos dentro de un dominio particular. Los DSL son descritos usando metamodelos, que definen relaciones entre elementos dentro de un dominio.

b) los Motores de transformación y generadores, que analizan ciertos aspectos de los modelos, que después crean varios tipos de artefactos, tales como código fuente, entradas de simulación, descripciones de uso o documentos XML, o representaciones alternativas de dicho modelo [2].

Por su parte las fases hacen parte integral del proceso de construcción de los metaprocesos. Estas fases, están compuestas de patrones conectados con un orden lógico y semántico, supeditados a unas respectivas reglas definidas para estos. Para la organización de cada uno de los modelos, cada una de las fases está referenciada por eventos de iniciación, eventos intermedios o eventos de finalización, y conectadas por medio de una relación de asociación, los cuales son los que darán pie para la correcta conexión entre patrones según las reglas definidas para el funcionamiento del proceso.

Cada fase debe de estar asociada a un metaproceso y poseer una descripción sobre la función que cumple dentro de un procedimiento, asimismo lleva un indicativo "TYPE" el cual es verificado y cotejado con el archivo adjuntado a un repositorio. Las fases que actúan dentro del repositorio, ya han sido analizadas y procesadas desde su estructuración por medio de la modelación BPMN. Cada uno de los procesos representados en las fases, fueron previamente evaluados y con estados previos de éxito dentro de los demás proyectos.

Los patrones por su parte, conceptualmente permiten que en un modelo no exista ambigüedad porque estos son específicos, dado a que pueden ser representados en un diagrama de clases o de objetos donde se captura semánticamente exactamente tal y como lo es en la vida real con sus atributos y operaciones que este posee.
En el prospecto de la construcción del software (Aplicación) tal como lo es *"**Activos"*** se han tenido en cuenta variables como: modelos, reuso y activos de software, las cuales han planteado la necesidad de generar un aplicativo que permita el análisis, la medición, la evaluación de calidad del software, aspectos claves para la producción de alta factura y de impacto industrial.

Para todo lo anterior ***"Activos"*** ha sido desarrollado con base en metaprocesos que parten de la experiencia del Modelo de Referencia Promoter (PRM) que está precedido de cuatro reglas básicas tales como son: el análisis de tareas, la provisión de tecnologías, la realización y la ejecución de los procesos de software.

Greenwood et al. reconocen la necesidad de partir del concepto fundamental de los PSEEs [9] [citado en 8], quienes se basan en cuatro servicios fundamentales: administración del diálogo interusuarios, abstracción de información, repositorio de procesos y productos y herramientas de comunicación. Todo está previsto para que **"*Activos"*** no solo sea una aplicación usada por un usuario específico sino por un usuario final asociado más a los patrones de interfaces graficas entendibles, interactivas y amigables con cualquier usuario.

La creación y el uso del software ***"Activos"*** está basado en los estándares de especificación de la OMG que definen las pautas para la elaboración de activos reusables tal como se pueden encontrar en la reseña "Unified Modeling Language: Infrastructure version 2.0 formal/05-07-05", conocido como UML 2.0, el cual destaca la arquitectura del lenguaje, formalismos del lenguaje, *Cores* y entre otras características que debe poseer un software en sus etapas de elaboración más primitivas [6]. Para ello la OMG plantea los diferentes tipos de activos de software que son reusables a partir de tres dimensiones fundamentales: la granularidad, la variabilidad y la articulación. En esta última dimensión "***Activos"*** obtuvo sus cimientos.

Para que la estructura de ***"Activos"*** sea legible en cuanto a su estructura semántica cuenta un manifiesto de archivo (documento XML). Así se garantiza la veracidad del activo de software. Dicho manifiesto permite atravesar e inspeccionar restricciones semánticas para lo cual los *asset* son parte fundamental, tal como Avila et al. aplica su método a un entorno de aplicación utilizando la especificación de activos reutilizables RAS (Reusable Asset Specification) [5], [6], que es un estándar para empaquetar artefactos digitales con una detallada especificación de un conjunto de

DOI: http://dx.doi.org/10.21501/21454086.1967

directrices y recomendaciones sobre la estructura, contenido y descripción de los activos de software reutilizables y el metamodelo de procesos de software SPEM (Software Process Engineering Metamodel), en este se realiza el modelado de tareas ejecutadas por roles que consumen artefactos de entrada y producen artefactos de salida".

Desentrañando las bases del software **"Activos"**, Fig. 3, en este artículo de investigación se sustenta y solventa la creación de esta aplicación, adoptando bibliografía y metodologías amparadas en organismos internacionales expertos es la tecnologías del reúso de activos haciendo uso del "Esquemas de XML sobre el Manifiesto de Activos de Software Reusables" [8].

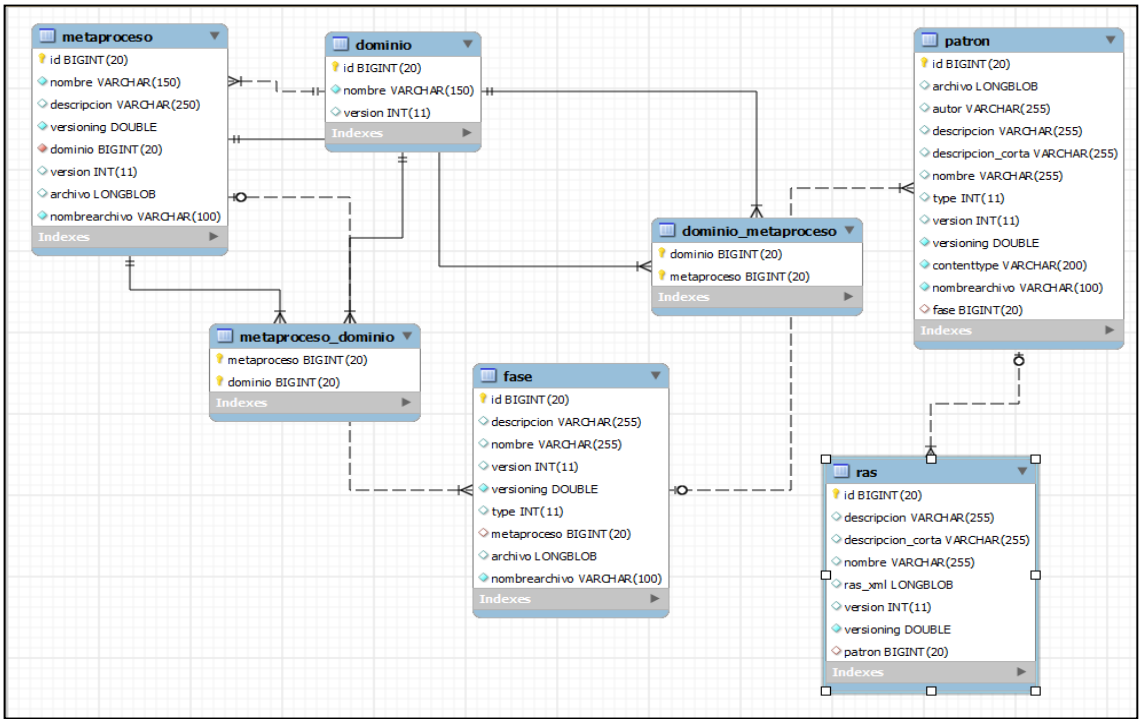


**Fig. 3.** Esquema de base de datos del aplicativo

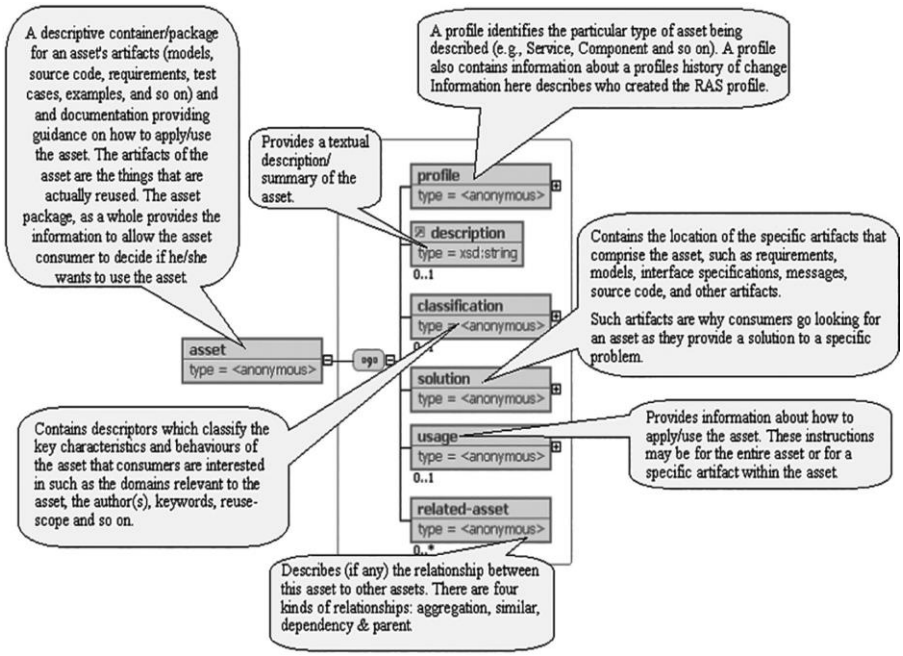


**Fig. 4.** Definición de atributos de un activo de software

Yeimar Alonso Castro; MSc. Javier Darío Fernández Ledesma
Julián Alberto Rivera; MSc. Eder Acevedo Marin
DOI: http://dx.doi.org/10.21501/21454086.1967

## 4. RESULTADOS

Para el desarrollo de este repositorio, desde un comienzo se empezaron a definir reglas para cada uno de los componentes que estarían integrados dentro del proceso de reúso de activos de software. Dentro de estas reglas se acogieron los tipos de archivos que serían adjuntos al repositorio y su composición, debido a que se investigó y escrudiñó dentro del archivo con extensión .XPDL, para analizar las propiedades que contiene, para poder verificar que el modelo que se encuentra representado dentro de este archivo, así contenga los elementos que se están describiendo en el repositorio. Con este análisis se especifica si el archivo realmente contiene elementos únicos de una fase de iniciación, intermedia o de finalización.

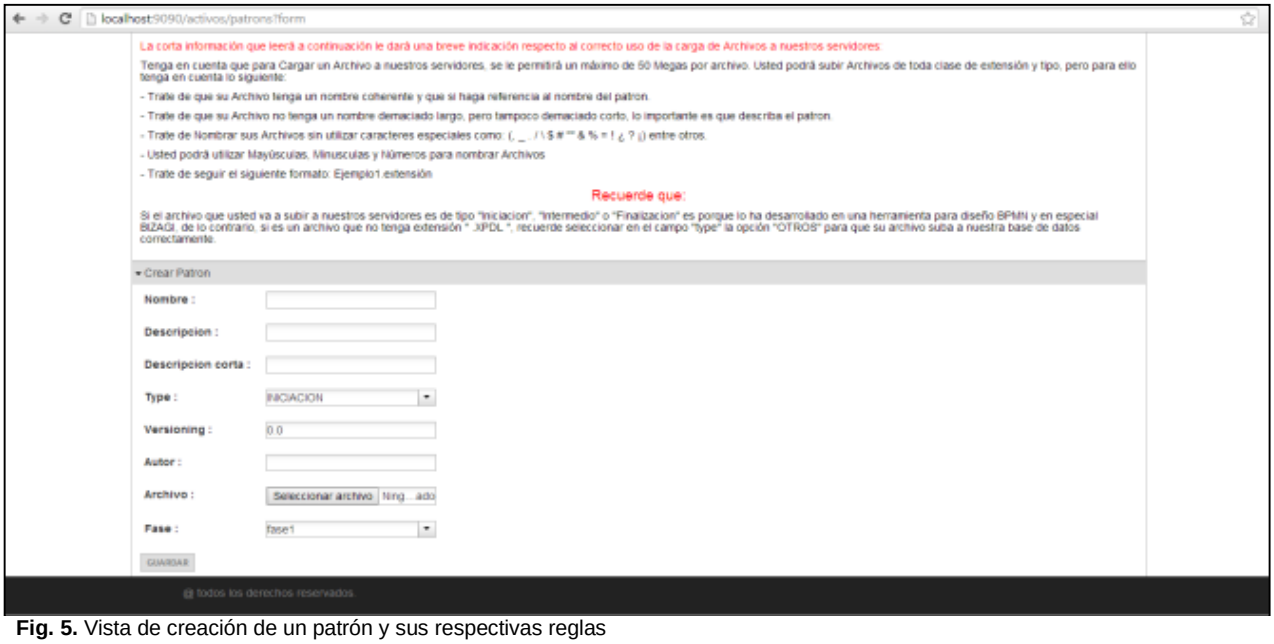


**Fig. 5.** Vista de creación de un patrón y sus respectivas reglas

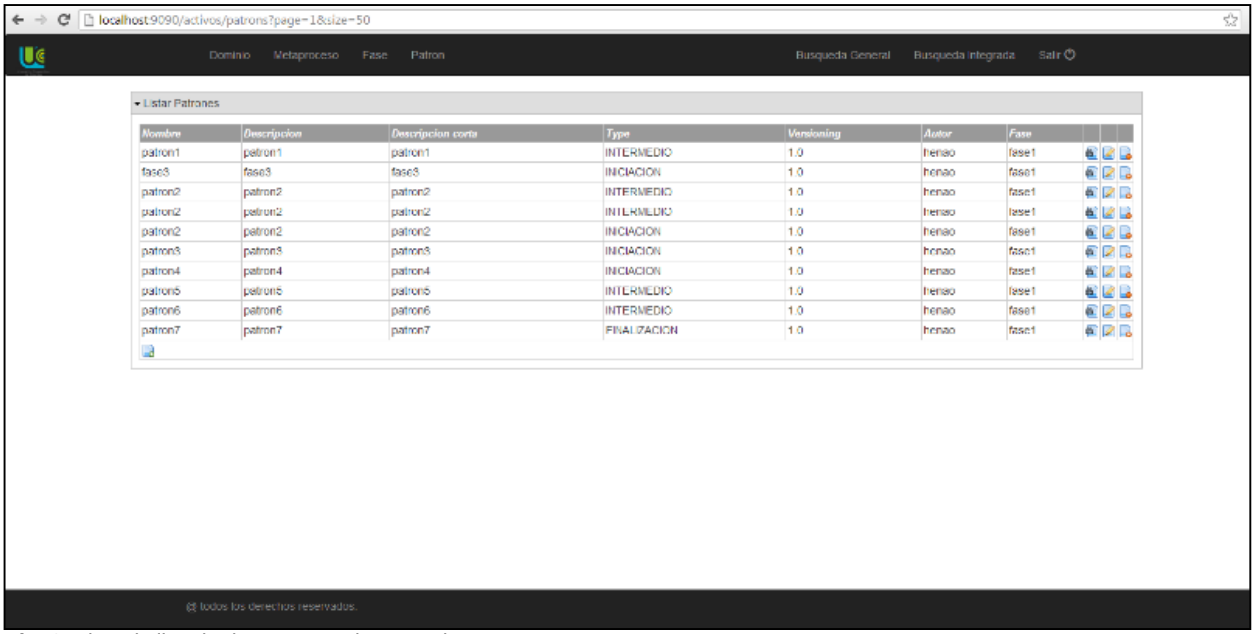


**Fig. 2.** Vista de listado de patrones almacenados

DOI: http://dx.doi.org/10.21501/21454086.1967


**Fig. 6.** Datos de un Metaproceso con las fases y patrones asociados

La generación de estas reglas define la estructuración de un proceso sin inconvenientes, que contiene un proceso secuencial, en donde se dispone de textos de ayuda para la guía y correcta utilización del repositorio de activos de software. Dentro de estos parámetros establecidos se define un orden consecutivo el cual debe de ser realizado de manera correcta para que el activo pueda mantenerse durante el tiempo requerido dentro del repositorio y cumplir su función de servir de base en modelos complementarios o pertenecer como pieza fundamental y decisiva en un nuevo desarrollo.

El desarrollo del repositorio de activos de software por medio de artefactos reusables se creó específicamente para dar soluciones de reúso y agilidad al momento de desarrollar nuevo software. Para esto se adjuntaron funciones como la descarga de archivos subidos en las fases y patrones, y por último la descarga del archivo, RAS, el cual será usado para realizar las búsquedas necesarias dentro del repositorio, ya que este archivo es el que contiene toda la información de un patrón en su totalidad, tanto desde a que dominio está asociado como que tipo de modelo es y en qué tipos de temas dentro del repositorio este se puede acoplar perfectamente.

Estas son las recomendaciones que el repositorio como función principal cumple. Para esto, dentro existen variedad de búsquedas las cuales ayudan a consultar los activos que están almacenados dentro del repositorio.

## 5. CONCLUSIONES

Se obtienen las siguientes conclusiones:

- Se construyó un aplicativo que suple con las necesidades del entorno y la problemática planteada en los antecedentes.
- Se logró materializar y crear arquetipos de los diferentes tipos de modelos UML, VISAGI, para la elaboración del repositorio de activos de software con la finalidad de prestar un servicio a especie de hemeroteca para las grandes industrias del software.
- Se unificaron de diferentes momentos del desarrollo de un proyecto para la construcción de nuevo software seguro, ágil y usando métodos de reúso.
- Se disminuyó los tiempos de desarrollo y modelado en las arquitecturas de proyectos de software.
- Se mejoró en la calidad y productividad en los procesos de desarrollo de software.
- Se Dotó de un repositorio de activos de software flexible y capas de recomendar elementos pertinentes según atributos relevantes del proyecto.

Yeimar Alonso Castro; MSc. Javier Darío Fernández Ledesma
Julián Alberto Rivera; MSc. Eder Acevedo Marin
DOI: http://dx.doi.org/10.21501/21454086.1967

## REFERENCIAS

[1] R. S. Pressman, Ingenieria del software, 5 ed., R.S. Mc Graw Hill, 2001.

[2] J. M. Pérez, F. Ruiz y M. Piattini, “Model Driven Engineering Aplicado a Business Process Management”, 2007. Disponible en https://previa.uclm.es/dep/tsi/pdf/UCLM-TSI-002.pdf

[3] A. G. Kleppe, J. B. Warmer, and W. Bast, MDA explained: the model driven architecture: practice and promise. Addison-Wesley Professional, 2003.

[4] E. Gamma, *Patrones de diseño: elementos de software orientado a objetos reutilizable*. Pearson Educación, 2002.

[5] O. Ávila-García, A. Estévez-García, V. Sánchez-Rebull, y J. L. Roda-García. “Using software product lines to manage model families in model-driven engineering.” In SAC 2007: Proceedings of the 2007 ACM Symposium on Applied Computing, track on Model Transformation, pages 1006_1011. ACM Press, Mar 2007.

[6] O. Avila-García, A. Esteves-García, E. V. Sánchez-Rebull, J. L. Roda-García, “Combinando Modelos de Procesos y Activos Reutilizables en una Transición poco Invasiva hacia las Líneas de Producto de Software”. *Proceedings 12th Conference on Software Engineering and Databases*, pp. 1-6. 2007.

[7] G. Om, “Unified Modeling Language: Infrastructure, version 2.0, formal/05-07-05,” 2006.

[8] J. D. Fernandez, *Metaprocesos: modelos, reuso y activos de software*, U. Pontificia Bolivariana, 2014, ISBN: 9789587641998

[9] Greenwood et al., 1999

[10] S. Acuña y X. Ferré, "Software Process Modelling". In: Proceedings of the 5th. World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001). Orlando Florida, USA. , pp.1-6, 2001.

[11] J. J. Alfaro. Sistemas para la medición del rendimiento en la empresa, México, ed. Limusa, 2008.

[12] F. Allilaire, J. Bezivin, F. Jouault, y I. Kurtev, ATL – Eclipse support for Model Transformation, Procedings of the Eclipse Technology Exchange Workshop at the ECOOP 2006 Conference, Nantes, Francia. 2006.

[13] E. Almeida, et all. Component reuse in software engineering. p. 219. 2008.

[14] V. Ambriola, et all. “Assessing Process-Centered Software Engineering Environments”. *ACM Transactions on Software Engineering and Methodology*, vol. 6, no. 3, pp. 283–328. (1997, jul.). Disponible en https://dl.acm.org/citation.cfm?doid=258077.258080

[15] V. Anaya, et al. The Unified Enterprise Modelling Lenguage – Overview and Further Work. In IFAC World Congress, PapersOnLine, 2008.

[16] J. Andreoli, J. L. Meunier, D. Pagani. “Process Enactment and Coordination”. *Process Technology*, pp. 9-11, 1996. Disponible en https://link.springer.com/chapter/10.1007/BFb0017745

[17] S. Arbaoui, F. Oquendo, PEACE: goal-oriented logic-based formalism for process modelling. In A. Finkelstein, J. Kramer, and B. Nuseibeh, editors, Software Process Modelling and Technology. John Wiley & Sons Inc., 1994.

[18] T. Asikainen y T. Männistö, Nivel: a metamodelling language with a formal semantics. *Software & Systems Modeling*. vol. 8, N. 4, 2009, pp 521-549

[19] M. Baldi, Et al. E3: object-oriented software process model design. In A. Finkelstein, J. Kramer, and B. Nuseibeh, editors, Software Process Modelling and Technology. John Wiley & Sons Inc., 1994.

[20] S. Bandinelli, et al. SPADE: An Environment for Software Process Analysis, Design, and Enactment. In A. Finkelstein, J. Kramer, and B. Nuseibeh, editors, Software Process Modelling and Technology. John Wiley & Sons Inc., 1994.

[21] N. Barglloutí. Supporting Cooperation in the MARVEL Process-Centered SDE. In ACM-SDE-1 2/92/VA, pp. 21-31. 1992.