Ingeniería inversa de un software para modelar procesos de negocio basado en una notación propietaria, utilizando notación UML

Harry Carpio S. [1] Jorge Rodríguez E. [2]
Facultad de Ingeniería en Electricidad y Computación (FIEC) [1][2]
Escuela Superior Politécnica del Litoral (ESPOL)
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil-Ecuador
hcarpio@espol.edu.ec [1] jirodrig@espol.edu.ec [2]

Resumen

El presente trabajo se centra en la realización del proceso de ingeniería inversa a un sistema para modelar procesos de negocios basado en una notación propietaria con el fin de obtener como resultados un manual para realizar su despliegue y la documentación técnica de sus aspectos más relevantes usando diagramas UML. El sistema a estudiar está implementando usando el framework de código abierto Ruby on Rails el cual está escrito en el lenguaje, también de código abierto, Ruby. El trabajo realizado se divide de manera general en cinco etapas, el estudio del estado del arte relacionado a ingeniería inversa y Ruby on Rails, el análisis de los requerimientos en donde se delimitan las características específicas del sistema a ser estudiadas, la recuperación del ambiente en donde se realiza el despliegue del sistema y la documentación paso a paso de cómo lograrlo, la recuperación del diseño utilizando diagramas UML y el análisis de la recuperación del diseño obtenido.

Palabras Claves: *Ingeniería inversa, top-down, vista arquitectónica, Ruby on rails, UML, BPMS, framework.*

Abstract

This work is focused on the realization of the reverse engineering process to a business process modeler system based on proprietary notation with the purpose of get as result an instructive deployment and technical documentation of the most important aspects of it using UML diagrams. The system under study is implementing using the open source framework Ruby on Rails which is written in the programing language, also open source, Ruby. The realized work is divided in a general way in five stages, the study of state of art about reverse engineering and Ruby on Rails, the requirement analysis where the specific characteristics of the system to be studied are delimited, the recovery of environment where the system deployment is done and it is documented step by step, the design recovery using UML diagrams and the analysis of the design recovery.

Keywords*: Reverse engineering, top-down, architectural view, Ruby on rails, UML, BPMS, framework.*

1. Introducción

En este documento se consideran los aspectos de mayor relevancia del proyecto de ingeniería inversa al sistema PYX4 [1].

La motivación de este trabajo se fundamenta bajo la premisa de que la fase de mantenimiento es la más extensa y costosa del ciclo de vida del software y que la complejidad de agregar una nueva característica o funcionalidad a un sistema es inversamente proporcional a la cantidad de información de alto nivel sobre su implementación que se encuentra disponible.

Este proyecto se enmarca como fase preliminar de un proyecto de reingeniería para el sistema pyx4 que tiene como objetivo agregar soporte para la notación BPMN que es la estándar para modelar procesos de negocio, sin embargo, el trabajo aquí presentado está limitado a describir al estado actual del sistema previo a agregar el soporte mencionado.

Como insumo para iniciar la ingeniería inversa se tiene el código fuente del sistema pyx4 que está implementado sobre el framework de código abierto Ruby on Rails escrito en lenguaje Ruby, Ruby es un lenguaje interpretado y de scripting con orientación a objetos pura, es decir, para Ruby todo es un objeto.

En las secciones posteriores se describe el proceso llevado a cabo para lograr la documentación, se presentan los diagramas más relevantes y se mencionan los aspectos más importantes a tener en cuenta respecto al estado actual de la implementación del sistema.

2. Objetivos

Los objetivos del trabajo desarrollado son los siguientes:

- Desplegar el sistema PYX4 proveyendo un instructivo para su realización.
- Realizar el proceso de ingeniería inversa a partir del código fuente del BPMS PYX4 documentando su arquitectura.
- Generar los diagramas de clases, componentes, despliegue y entidad-relación a partir de la ingeniería inversa para ser incorporados en la documentación del sistema.

3. Generalidades

En rasgos generales la reingeniería [2] es un mecanismo usado en diversas áreas cuando se requiere mejorar algún aspecto de un producto de ingeniería, por su parte y de manera específica, la reingeniería de software busca modernizar las capacidades de un sistema o mejorar su mantenibilidad.

El proceso de reingeniería de software está compuesto por dos etapas, ingeniería inversa para el análisis del sistema existente e ingeniería directa para la reconstrucción o implementación de nuevos componentes [3]. Ver Figura 1.

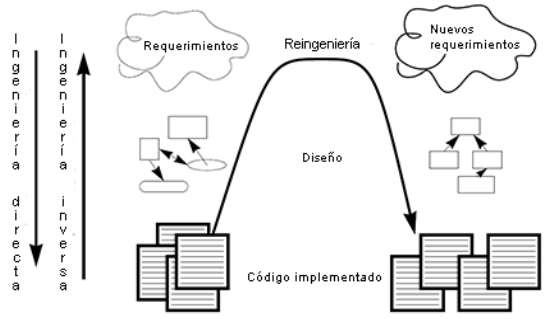

Figura 1. Proceso de reingeniería

4. Metodología

Las metodologías que fueron consideradas poseen diferentes enfoques tales como ingeniería inversa de lógica o proceso, ingeniería inversa de datos e ingeniería inversa de interfaces de usuario, estos enfoques permiten focalizar el esfuerzo dependiendo de las necesidad que se requieran atender.

Además de definir el enfoque se requiere establecer un mecanismo para realizar el análisis, dado el escenario descrito en secciones anteriores, para llevar a cabo el trabajo, consideramos como referencia la metodología top-down [4], que se basa en que a partir del conocimiento general llagar hacia al específico, de lo general hacia el detalle. Adicional se siguió una estrategia outside-in, la cual consiste en que a partir de estímulos externos al sistema, los cuales están relacionados a los requerimientos funcionales iniciales, se va a determinar los procesos lógicos que los manejan así como la integridad de los datos asociados.

En cuanto al uso de herramientas, fueron consideradas principalmente las correspondientes a depuradores, herramientas de análisis automático y herramientas CASE para recrear el diseño del sistema [5].

5. Lenguaje de modelado unificado (UML)

Para realizar las representaciones de la implementación del sistema se ha considerado el lenguaje UML [6], la notación dispone de una variedad de diagramas que obedecen a una semántica general basada en elementos y las relaciones entre ellos. Los diagramas UML en su versión 2.0 se separan en estructurales y comportamentales, de estos últimos hay un sub grupo correspondiente a los diagramas de interacción. Ver Figura 2.

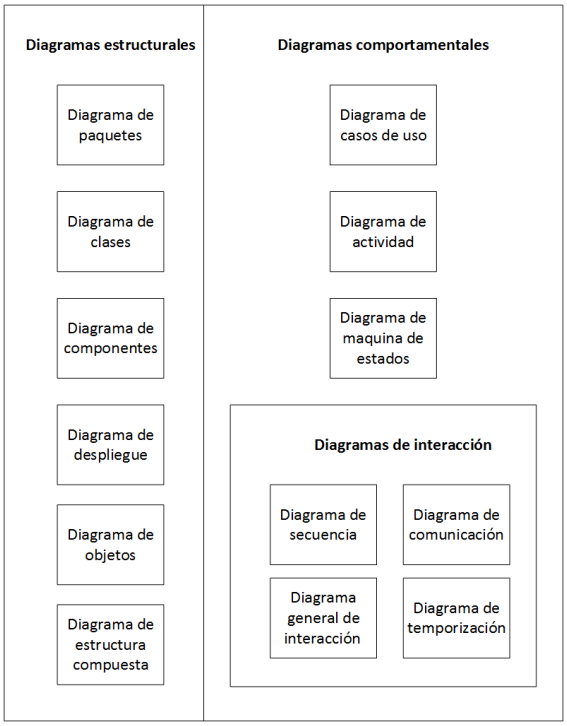

Figura 2. Diagramas UML 2.0

6. Ruby on Rails

Ruby on Rails es un framework para desarrollo web de código abierto construido usando Ruby como lenguaje [7].

Los pilares de RoR son DRY (*Don't repeat yourself*) y la conveción sobre la configuración, estos pilares buscan agilizar el desarrollo y minimizar la cantidad de líneas escritas para realizar una tarea [8].

RoR implementa el patrón MVC, y estructuralmente lo tiene bien definido en sus directorios con una carpeta para cada capa del modelo. En la Figura 3 se muestra el orden de llamadas que se lanzan al realizarse una petición HTTP desde el cliente web, en síntesis, todo empieza con una llamada a una URL, Rails enruta la llamada hacia el controlador correspondiente desde donde se hace uso del o los modelos definidos los cuales consumen datos de la base de datos, como respuesta el controlador actualiza la vista correspondiente por medio de las plantillas en formato html.erb y devuelve el código HTML al cliente que lanzó el requerimiento.

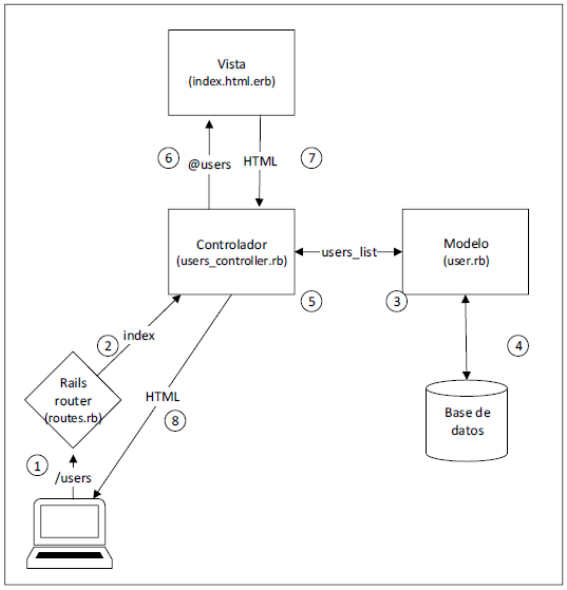

Figura 3. Procesamiento de una petición HTTP desde la perspectiva MVC de RoR

7. Desarrollo

El trabajo realizado se rige de acuerdo a las fases mostradas en la Figura 4, en el (1) análisis de requerimientos se establecen las tareas que lleva a cabo el sistema, (2) en la recuperación de la implementación se realiza el despliegue completo del sistema, (3) en la recuperación del diseño se reproducen los diagramas UML principales y (4) en el análisis de la recuperación se discute acerca de las implicaciones del trabajo realizado.

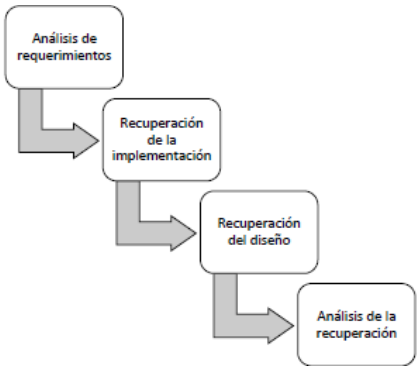

Figura 4. Fases del proyecto de ingeniería inversa

8. Resultados

Los resultados de este proyecto corresponden a documentación relevante acerca del estado del software PYX4, como anexo del trabajo es posible encontrar el manual de despliegue y los controladores Rails del sistema.

Como particularidad, la estructura de la base datos no presenta relaciones por lo cual un diagrama de entidad-relación no resulta gráficamente útil y en su lugar se obtuvo un diagrama de dependencia de modelos PYX4 basado en las relaciones declaradas en los modelos del patrón MVC de Rails.

En la Figura 5 se presenta el diagrama de despliegue del sistema PYX4 con el fin de que el lector pueda tener una idea general del sistema y pueda valorar el seguir profundizando en este trabajo.

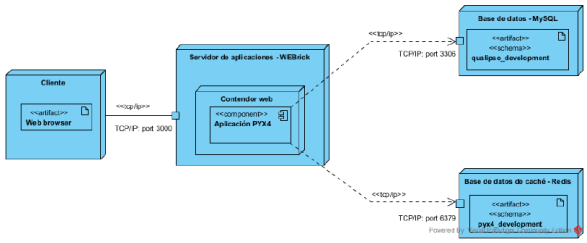

Figura 5. Diagrama de despliegue PYX4

9. Conclusiones

Todo proceso de ingeniería inversa debe de estar correctamente delimitado con el objetivo de que orientar el análisis en la dirección necesaria. En este trabajo la atención se centró en la diagramación de modelos y documentación.

El sistema se encuentra escrito usando versiones legadas de Ruby y de Ruby on Rails por lo cual, si existe la necesidad de agregar nuevas características, se debería empezar procurando actualizarlo a las versiones más recientes lo que si bien puede

desencadenar una serie de errores de compilación y ejecución; sin embargo, es un ejercicio necesario para procurar disminuir la dificultad de mantenimientos futuros.

10. Referencias

[1] Globalliance France, The PYX4 Solution, http://www.pyx4.co.uk/our-software-solution, consultado en Marzo del 2015.

[2] Demeyer S., Ducasse S., Nierstrasz O, Object-Oriented Reengineering Patterns, Square Bracket Associates, 2009.

[3] Chikofsky E., Cross J., Reverse Engineering and Design Recovery: A Taxonomy, IEEE Software, 1990.

[4] Dennett D. (Center for Cognitive Studies), Cognitive Science as Reverse Engineering: Several Meanings of "Top-Down" and "Buttom-Up", http://users.ecs.soton.ac.uk/harnad/Papers/Py104/dennett.eng.html, consultado en Marzo del 2015.

[5] Tutorials Point (I) Pvt. Ltd., Software Case Tools Overview, http://www.tutorialspoint.com/software_engineering/case_tools_overview.htm, consultado en Marzo del 2015.

[6] Riva C., Selonen P., Systä T., Xu J., UML-Based Reverse Engineering and Model Analysis Approaches for Software Achitecture Maintenance, IEEE International Conference on Software Maintenance (ICSM'04), 2004.

[7] Corbridge R., Introducing Ruby ON Rails Part One, http://www.softwaredeveloper.com/features/intro-to-ruby-on-rails-042507/ , consultado en Abril del 2015.

[8] Rubyonrails.org, Getting Started with Rails, http://guides.rubyonrails.org/v3.2.12/getting_started.html , consultado en Abril del 2015.