



## Sistema para asistir en la generación de pruebas automáticas de software

Trabajo Terminal No. 2018-B036

Alumnos: Dominguez de la Rosa Bryan\*, Gómez Reus Jorge, Pacheco Díaz Alejandro

Directores: López Rabadán José Jaime

e-mail: bryan.dominguez.delarosa@gmail.com

PRIMA: Unico HORA: 12:00

**Resumen** – Dentro del desarrollo de un software, el proceso de pruebas requiere de una inversión grande de recursos del proyecto. Se propone un sistema que simplifique la tarea de generar pruebas funcionales con base en la documentación de la etapa de análisis usando una base de datos que guarda los elementos que componen los casos de uso, para reutilizar el trabajo de la etapa antes mencionada.

**Palabras clave** –Ingeniería de Software, Pruebas funcionales, Java, Selenium.

### 1. Introducción

Uno de los procesos dentro del ciclo de vida del desarrollo de software [1] es el de pruebas, el cual tiene 2 finalidades: Demostrar, tanto al desarrollador como al cliente, que el software satisface los requerimientos y también descubrir defectos, como comportamientos incorrectos, no deseables o que no cumplen alguna especificación [2]. El principal inconveniente de las pruebas de software es la cantidad de recursos necesarios para poder realizarlas, los métodos actuales para realizar pruebas de software están sujetos a contratar y capacitar capital humano que se dedique a preparar y realizar dichas pruebas, además de los detalles del negocio y los requerimientos, por lo que se estima que el costo de realizar pruebas de software puede llegar a representar entre el 30% y el 50% del costo total del software desarrollado [3]. Tomando como entrada una base de datos con la información asociada a los casos de uso elaborados en la etapa de análisis y utilizando una herramienta como Selenium, que permita controlar las acciones realizadas en un navegador web, se puede asistir en la construcción de pruebas funcionales automáticas para así, reutilizar la documentación realizada en la etapa de análisis.

En la etapa de análisis, el modelado de los requerimientos comienza con la creación de escenarios en forma de casos de uso. Los modelos basados en el escenario son los que buscan satisfacer los requerimientos desde el punto de vista de actores y sus interacciones con el sistema. Los casos de uso describen, en un lenguaje claro, un escenario específico desde el punto de vista de un actor definido y muestra la forma en la que este actor interactúa con un sistema para alcanzar un objetivo. Los elementos principales que conforman un caso de uso son: identificador, mensajes, reglas de negocio, interfaces y trayectorias compuestas por pasos. Cada paso está constituido por un actor, una acción y uno o más sustantivos.

En un modelo basado en el escenario, se generen casos de prueba a partir de los elementos de los casos de uso. Los casos de prueba especifican lo que se va a probar de un sistema y están formados por un conjunto de entradas, las condiciones bajo las que se deben ejecutar esas pruebas además los resultados esperados [5] y se construyen tomando del caso de uso, los pasos de la trayectoria. En cada paso se identifica el actor, la acción que realiza (a través del verbo usado) y las entidades afectadas (por medio de los sustantivos y descritas en el modelo de datos).

Por lo tanto, el proceso de construcción de los casos de prueba hace uso, de la documentación de los casos de uso, pero no la reutiliza de una manera óptima, pues es necesario de la especialización de personal, o como tal, de un departamento exclusivo para realizar pruebas, ya que hay que tomar cada caso de uso e identificar cada elemento para realizar el proceso antes señalado y posteriormente ejecutar dichas pruebas y analizar sus resultados.

Utilizando la herramienta Selenium y una base de datos, previamente poblada con los casos de uso de la etapa de análisis, este proyecto pretende diseñar y desarrollar un sistema que tome los datos de la base de datos y sea capaz de interpretar casos de uso para poder generar pruebas automáticas y ejecutarlas sobre el sistema en desarrollo bajo los parámetros que un usuario pueda dar (Un conjunto de datos entrada), así, al no ser necesaria una capacitación técnica en Selenium ni generar cada prueba individualmente, se optimizaría la reutilización de la documentación por casos de uso de la etapa de análisis. La manera en que la base de datos de casos de uso es poblada está contemplada en otro proyecto, este tomará la información de los casos de uso de una ya existente.

Recibi protocolo  
30/Nov/2018

30/11/18 Recibi Protocolo Reestructurado

Recibi protocolo 30/Nov/18

Actualmente no existe una herramienta tan especializada con esta, aunque existen proyectos similares que hacen uso de tecnologías como inteligencia artificial [9] o casos más específicos con redes neuronales [6], no encontramos algún sistema, propuesta o TT que aproveche, del mismo modo que pretende este proyecto, la documentación realizada en la etapa de análisis para la generación de las pruebas a nivel funcional, al menos en modelos basados en escenarios, ya que requiere de la reutilización de la información de los casos de uso para poder generar dichas pruebas, aún así, en la siguiente tabla describimos las características de herramientas que son utilizadas en el mercado de pruebas de software.

<b>Herramienta</b>	<b>Interfaz gráfica</b>	<b>Configuración utilizando programación o API</b>	<b>Pruebas a software web</b>	<b>Reutilización de análisis u otra etapa</b>
Katalon Studio [11]	Si	No	Si	No
Testim[12]	Si	No	Si	No
SmartBear - TestComplete [9]	Si	Sin información	Si	Si, reutiliza las pruebas antes realizadas
Watir[13]	Si, con programas de terceros	Si	Si	No
Microfocus[14]	Si	Si	Si	No
Ranorex[15]	Si	Si	Si	No
Squish[16]	Si	Si	Si	No
TT 2013-B061[18]	No	No	Si	Si
Tesis de modelo de pruebas[19]	Si	No	Únicamente al de Seguros Atlas	No

*Tabla 1: Comparación de software con fines similares a la propuesta.*

Ninguna de los softwares encontrados en uso en el mercado contiene un módulo similar al que nuestra propuesta pretende hacer, que es la reutilización de la documentación previa, además de que no se encontró algún Trabajo Terminal, Tesis o proyecto de esta índole en la base de datos del Instituto Politécnico Nacional [17]. Sin embargo, algunas herramientas de la tabla 1 contienen ciertos elementos que se pueden detallar, como por ejemplo *Watir* que utiliza como motor base Selenium combinado con su API de Ruby o *Microfocus* que dentro de su API incluye conexión con Selenium Web Driver para ejecutar dichas pruebas. A pesar de compartir la misma herramienta, la manera en la que son generadas las pruebas de software sigue siendo tradicional y solo buscan cumplir el objetivo de automatizar la ejecución de pruebas más no su generación.

## 2. Objetivo

### 2.1 Objetivo general

Diseñar y desarrollar un sistema que asista en la generación de pruebas funcionales automáticas de software con base en la documentación por casos de uso y su ejecución haciendo uso de la herramienta Selenium.

### 2.2 Objetivos específicos

- 1.- Utilizar la documentación por casos de uso previamente vaciada en una base de datos para generar casos de prueba.
- 2.- Ejecutar los casos de prueba y guardar los resultados obtenidos.
- 3.- Permitir a un usuario:
  - a. Definir los conjuntos de entrada para los casos de prueba generados.
  - b. Gestionar los resultados obtenidos.

## 3. Justificación

Las pruebas de software son una función del control de calidad que tiene como objetivo principal detectar errores. Las pruebas funcionales son las que tienen como propósito analizar el software en desarrollo y buscar aquello que no va de acuerdo con las especificaciones externas [4] y son este tipo de pruebas las que aseguran al cliente que el sistema desarrollado cumple sus requerimientos con base a la documentación realizada en la etapa de análisis. Pero como se mencionó anteriormente, no es óptima la manera en que se reutiliza dicha documentación, por lo tanto, el desarrollo de esta herramienta permitiría aprovechar de mejor manera los recursos ya invertidos en la etapa de análisis y por lo tanto optimizar el desarrollo del software en general.

En este proyecto desarrollará una herramienta de tal manera que, mediante una sola aplicación, un *tester* (Personal encargado de realizar pruebas a un sistema) pueda establecer parámetros específicos (Como un conjunto de datos de entrada) y, tomando la información de los casos de uso de la base de datos, la aplicación generará pruebas automáticas y realizará dichas pruebas al sistema en desarrollo haciendo uso del Selenium Web Driver.

Es posible encontrar casos de uso que representen una dificultad mayor a otros, debido a los elementos con los que el usuario tiene interacción durante su ejecución, así que la clasificación que usaremos para nuestro sistema se describe en la tabla 2. Teniendo en cuenta dicha propuesta, este proyecto cubrirá los casos de uso básicos y un prototipo para casos de uso intermedios utilizando el framework web estándar de Oracle, Java Server Faces.

<b>Casos de uso básicos</b>	<b>Nivel 1</b>	No necesitan precarga de información y solo utiliza los tipos básicos (campo de texto) del elemento <i>input</i> del estándar HTML [10].
	<b>Nivel 2</b>	Necesitan precarga de información y utilizan cualquier tipo de entrada del elemento <i>input</i> del estándar HTML (radio, checkbox, etc).
<b>Casos de uso intermedios</b>	<b>Nivel 3</b>	Pueden o no necesitar una carga de información y utilizan elementos específicos de algún framework de desarrollo web, pero los datos introducidos pueden ser identificados en el árbol HTML (calendar, picklist, etc).
<b>Casos de uso avanzados</b>	<b>Nivel 4</b>	Elementos específicos de algún framework de desarrollo web que necesitan de información externa para poder ser interpretados (APIs)

*Tabla 2: Propuesta de categorización por niveles de casos de uso.*

El desarrollo de este proyecto recopila fundamentos aprendidos a lo largo de la carrera de Ingeniería en Sistemas Computacionales, tales como: Entender y ejecutar el proceso de desarrollo de un software y la importancia de demostrar la calidad de este, además de hacer uso de la habilidad de analizar y usar sistemas y/o tecnologías desconocidas y también la gestión de proyectos y la utilización de metodologías de ingeniería de software.

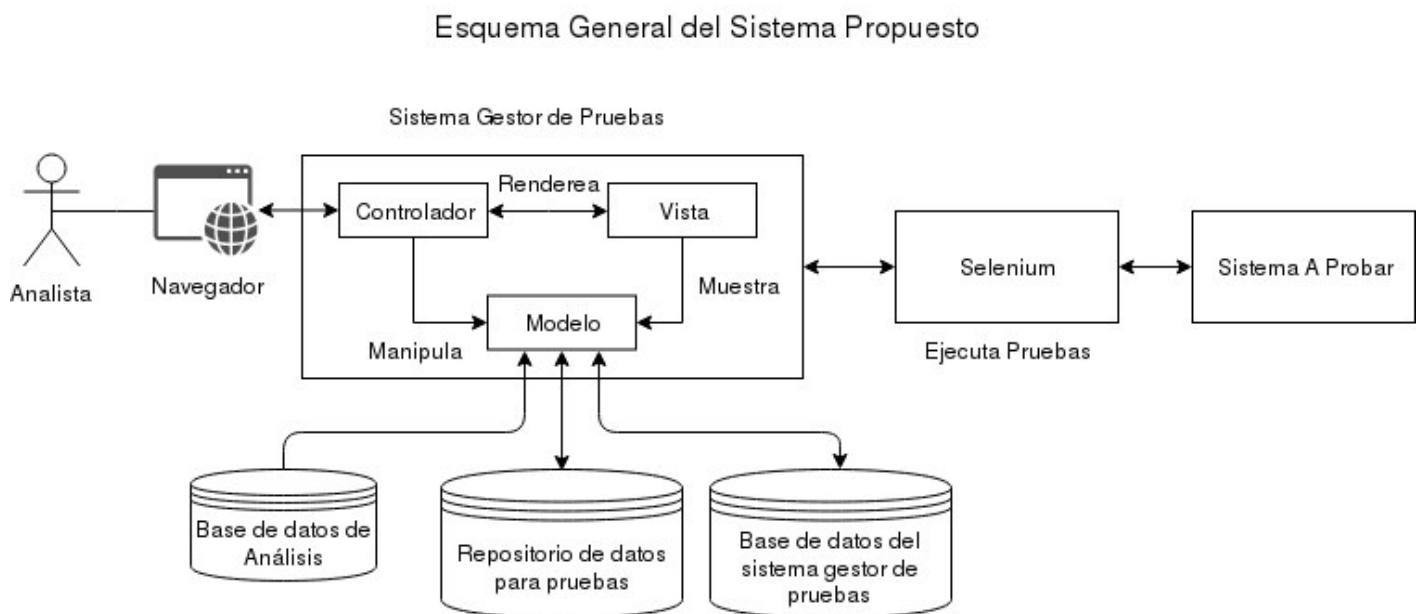
#### 4. Productos o Resultados esperados

El esquema propuesto para el sistema se muestra en la *Fig1*. Se proporcionará una interfaz gráfica al analista para asistirlo en la generación de las pruebas, posteriormente el sistema toma la información necesaria de una base de datos de análisis, proporcionada por otro proyecto, y de un repositorio de datos para pruebas, que contendrá los datos de entrada. El sistema tendrá su propia base de datos en la que se podrán guardar las pruebas que se generen. Al estar definidas las pruebas y sus datos el sistema las puede ejecutar usando Selenium, el cual interactuará con el sistema a probar para realizarlas.

Para la elaboración del proyecto se tienen contemplados, con base en las tecnologías a utilizar (Java), los siguientes patrones de diseño:

- Modelo – Vista – Controlador (MVC)
- Strategy
- Singleton
- Factory
- Data Access Object (DAO)
- Data Transfer Object (DTO)

Los cuales podrán cambiar en caso de que el análisis del sistema lo requiera.



*Fig. 1 Esquema general del sistema propuesto*

Documentos esperados a entregar al finalizar el TT:

1. Código fuente.
2. Documentación técnica.
  - a. Documentación por prototipo (Análisis, maqueta y pruebas).
  - b. Documento de casos de uso
3. Base de datos.
  - a. Scripts.
  - b. Modelo de datos.
4. Manual de usuario.
5. Guiones de prueba.



## 5. Metodología

Utilizaremos una propuesta de metodología híbrida que toma como base el modelo en espiral [7], el cual consiste en una serie de entregas evolutivas donde en cada iteración se generará un prototipo del sistema, teniendo en cuenta también los principios del desarrollo ágil [8].

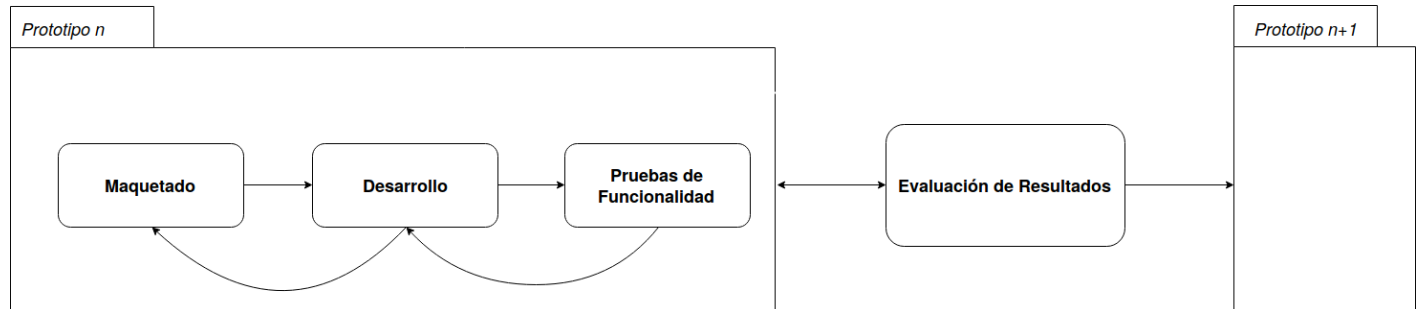


Fig. 2 Esquema general de una iteración de la metodología propuesta

Cada prototipo tendrá un objetivo particular definido, de tal manera que nos acerquemos al objetivo general al cumplirse, por completo o parcialmente, un objetivo particular. Como se aprecia en la figura 2, se realizará una maqueta (si se especificó qué habría), un desarrollo y pruebas de funcionalidad. Al terminar la iteración, se evaluarán los resultados obtenidos con el fin de diseñar el prototipo para la siguiente iteración.

## 6. Cronograma

Los cronogramas por integrante se anexan al final del documento.

## 7. Referencias

- [1] K. Lunn, "Software Development Cycle" in *Software Development with UML*, Ed. Londres: Palgrave, 2003.
- [2] I. Sommerville, "Pruebas de Software" in *Ingeniería del Software*, Ed. Madrid: Pearson Education, 2005, pp 492.
- [3] G. Myers, "Higher-Order Testing" in *The Art of Software Testing*, Ed. New Jersey: John Wiley & Sons, 2004, pp 90 – 95.
- [4] R. Pressman, "Prueba de aplicaciones convencionales" in *Ingeniería de Software, un enfoque práctico*, Ed. Nueva York: McGraw-Hill, 2010.
- [5] O. Jordán, O. Vázquez, "Generación de casos de prueba a partir de casos de uso en las pruebas de software", *Ingeniería Industrial*, 2010. [Online]. Available: <http://rii.cujae.edu.cu/index.php/revistaind/article/view/101/80>
- [6] Lilan Wu, Bo Liu, Yi Jin and Xiaoyao Xie, "Using back-propagation neural networks for functional software testing," 2008 2nd International Conference on Anti-counterfeiting, Security and Identification, Guiyang, 2008, pp. 272-275.
- [7] B. Boehm, "A spiral model of software development and enhancement", in *Computer*, vol. 21, no. 5, May 1998, pp. 61-72.
- [8] R. Pressman, "Desarrollo Ágil" in *Ingeniería de Software, un enfoque práctico*, Ed. Nueva York: McGraw-Hill, 2010, pp 55-80
- [9] Smart Bear Software, 2018. [Online]. Available: <https://smartbear.com/product/testcomplete>
- [10] S. Faulkner, A. Eicholz, T. Leithead, A. Danilo, S. Moon, "HTML5.2 Specification", W3C Recommendation, 14 December 2017. [Online]. Available: <https://www.w3.org/TR/html5/>
- [11] Katalon Studio, 2018. [Online]. Available: <https://www.katalon.com/>
- [12] Testim, 2018. [Online]. Available: <https://www.testim.io/>
- [13] Watir, 2018. [Online] Available: <http://watir.com/guides/>
- [14] Mirofocus Software, 2018. [2018] Available: <https://software.microfocus.com/en-us/products/unified-functional-automated-testing/features>
- [15] Ranorex, 2018. [Online]. Available: <https://www.ranorex.com/features/>
- [16] Squish, 2018. [Online]. Available: <https://www.froglogic.com/squish/>
- [17] Tesis institucionales, Instituto Politécnico Nacional, [Online]. Available: <https://tesis.ipn.mx/>
- [18] J. E. Álvarez y F.J. Ponce, "Metodología para el desarrollo de pruebas de software basadas en métricas de funcionalidad y rendimiento", Trabajo terminal, Escuela Superior de Cómputo del Instituto Politécnico Nacional, México, 2015.
- [19] G. Espinosa, "Desarrollo de un modelo de pruebas y calidad de software para la empresa de seguros Altas. S.A.", Tesis, Unidad Profesional Interdisciplinaria de Ingeniería Y Ciencias Sociales y Administrativas del Instituto Politécnico Nacional, México, 2016

## 8. Alumnos y directores

*Bryan Dominguez de la Rosa.* - Alumno de la carrera de Ing. en Sistemas Computacionales en ESCOM, Especialidad Sistemas, Boleta: 2015630117, Tel. 55-3280-7176, Correo: bryan.dominguez.delarosa@gmail.com

Firma: \_\_\_\_\_



*Jorge Gómez Reus.* - Alumno de la carrera de Ing. en Sistemas Computacionales en ESCOM, Especialidad Sistemas, Boleta: 2015630176, Tel. 55-5032-0041, Correo: j-g1996@live.com

Firma: \_\_\_\_\_



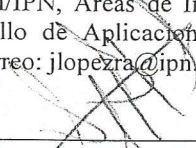
*Fernando Jair Pacheco Díaz.* - Alumno de la carrera de Ing. en Sistemas Computacionales en ESCOM, Especialidad Sistemas, Boleta: 2015630360, Tel. 5529131013, Correo: jair\_pacheco@outlook.com

Firma: \_\_\_\_\_



*José Jaime López Rabadán.* - M. en C. Ingeniería Eléctrica con especialidad en Computación por el CINVESTAV - IPN, Lic. Computación por la UAM, Profesor de ESCOM/IPN, Áreas de Interés: Ingeniería de Software y Desarrollo de Aplicaciones Web, Tel. 5729 6000 ext. 52004, Correo: jlopezra@ipn.mx

Firma: \_\_\_\_\_



*Hermes Francisco Montes Casiano.* - M. en C. Computación por el CINVESTAV - IPN, Ing. Sistema Computacionales por ESCOM - IPN, Profesor de ESCOM/IPN, Áreas de Interés: Sistemas Distribuidos, Desarrollo de Aplicaciones Web y Sistemas de Información Geográfica, Tel. 5729 6000 ext. 52004, Correo: hermes.escom@gmail.com

Firma: \_\_\_\_\_



CARÁCTER: Confidencial  
FUNDAMENTO LEGAL: Art. 3, fracc. II, Art. 18, fracc. II y  
Art. 21, lineamiento 32, fracc. XVII de la L.F.T.A.I.P.G.  
PARTES CONFIDENCIALES: No. de boleta y Teléfono.

Título del TT: Sistema para asistir en la generación de pruebas automáticas de software

Título del TT: Sistema para asistir en la generación de pruebas automáticas de software

[illegible]

Nombre del alumno(a): Jorge Gómez Reus TT No.: 2018-B036

Nombre del alumno(a): Jorge Gómez Reus TT No.: 2018-B036

Título del TT: Sistema para asistir en la generación de pruebas automáticas de software

[illegible]



Título del TT: Sistema para asistir en la generación de pruebas automáticas de software

Título del TT: Sistema para asistir en la generación de pruebas automáticas de software

[illegible]