MODELO PARA GESTIONAR PROCESOS DE TESTING EN DESARROLLOS SOFTWARE, REALIZADOS EN GRUPOS DE INVESTIGACIÓN DE LA UIS: CASO DE ESTUDIO GRUPO STI

DIEGO ARMANDO VILLARREAL DÍAZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍA FÍSICO MECÁNICA
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
MAESTRÍA EN INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2016

MODELO PARA GESTIONAR PROCESOS DE TESTING EN DESARROLLOS SOFTWARE, REALIZADOS EN GRUPOS DE INVESTIGACIÓN DE LA UIS: CASO DE ESTUDIO GRUPO STI

DIEGO ARMANDO VILLARREAL DÍAZ

Trabajo de investigación realizado para optar por el título de Magíster en Ingeniería de Sistemas e Informática

Directora:
SONIA CRISTINA GAMBOA SARMIENTO
Ingeniera de Sistemas, Ph.D

Codirector:
LUIS CARLOS GÓMEZ FLÓREZ.
Ingeniero de Sistemas, M.Sc

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍA FÍSICO MECÁNICA
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
MAESTRÍA EN INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2016

DEDICATORIA

Este trabajo lo quiero dedicar a mi familia quien me apoyo y acompaño durante todo este proceso de formación.

AGRADECIMIENTOS

Al profesor Luis Carlos por permitirme formar parte del grupo STI y brindarme su apoyo y orientación en el desarrollo de este proyecto.

A la profesora Sonia por toda su comprensión y apoyo en el desarrollo de este proyecto. Sin el cual este proyecto no hubiera sido posible.

A todos los miembros del grupo STI, quienes me han enseñado muchas cosas que atesoro y me han permitido crecer como persona.

A toda mi familia por su paciencia y apoyo continuo.

A mi hermano por darme ánimo y no permitirme desfallecer.

A Sandra Liliana Uribe Hernández por su apoyo y compañía.

A la Universidad Industrial de Santander por permitirme continuar con mi proceso de formación.

A Colciencias por permitirme ser Joven Investigador en dos ocasiones.

CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ANEXOS

Pág.

RESUMEN

TÍTULO: MODELO PARA GESTIONAR PROCESOS DE TESTING EN DESARROLLOS SOFTWARE, REALIZADOS EN GRUPOS DE INVESTIGACIÓN DE LA UIS: CASO DE ESTUDIO GRUPO STI[*].

AUTOR: Diego Armando Villarreal Díaz[**]

PALABRAS CLAVES: Grupos de investigación, Testing y PyME

DESCRIPCIÓN

Los grupos de investigación pueden verse en la necesidad de desarrollar diferentes tipos de software para apoyar sus actividades de investigación o como resultado de las mismas. En el caso de los grupos de investigación de las universidades, el desarrollo software generalmente es realizado por estudiantes como parte de sus trabajos de final de carrera. El grupo de investigación en Sistemas y Tecnologías de la Información (STI), avalado por la Universidad Industrial de Santander (UIS), ha tenido que extender las funcionalidades del software desarrollado por los estudiantes. Sin embargo, al realizar esta labor se ha encontrado que lo nuevos proyectos invierten una cantidad de tiempo considerable en la corrección de errores. En la ingeniería del software las pruebas son una herramienta efectiva para la detección de errores. En este punto es donde surge este trabajo de investigación, que busca mejorar la calidad de los desarrollos software en grupos de investigación a través de la incorporación de las pruebas de software. Para lograr esto se plantearon dos actividades de revisión; en la primera se hizo uso de SCOPUS para estudiar cómo se desarrolló software en pequeñas y medianas empresas desarrolladoras de software. En la segunda se estudiaron los trabajos de final de carrera de los estudiantes de ingeniería de sistemas con el propósito de identificar características en este tipo de desarrollo software. Con la información recolectada se formuló un modelo el cual contribuye a gestionar las pruebas de software en grupos de investigación. Finalmente, como trabajo futuro se recomienda estudiar la optimización de la cobertura de los casos de pruebas propuestos por los integrantes de los grupos de investigación.

[*] Trabajo de Investigación de Maestría

[**] Facultad de Ingenierías Físico-mecánicas. Escuela de Ingeniera de Sistemas e informática. Directora Ing. Sonia Cristina Gamboa Sarmiento, Ph.D. Codirector Ing. Luis Carlos Gómez Flórez, M.Sc.

ABSTRACT

TITLE: MODEL TO MANAGE THE TEST PROCESS IN SOFTWARE DEVELOPMENTS, MADE IN RESEARCH GROUPS AT UIS: GROUP STI CASE STUDY[*].

AUTOR: Diego Armando Villarreal Díaz[**]

KEYWORDS: Research groups, Testing and SMEs

DESCRIPTION

The research groups may need to develop different types of software to support their research activities or as a result. In the case of the research groups of universities, software development is usually performed by students as part of their final projects. The research group of investigación en Sistemas y Tecnologías de la Información (STI), research group at the Universidad Industrial de Santander (UIS), has had to extend the functionalities of the software developed by students. However, doing this work it has been found that new projects invest a considerable amount of time correcting errors. In software engineering testing is an effective tool for detecting errors. This research seeks to improve the quality of software developments in research groups through the incorporation of software testing. To achieve this, two review activities were done; the first one was made in Scopus to study how software is developed in small and medium-sized software development companies. The second one studied the final projects of systems engineering students with the purpose of identifying the main features in this type of software development. With the information gathered a model which helps to manage software testing at research groups was formulated. Finally, as further work is recommended to study the optimization of coverage of cases proposed by members of the research groups testing.

[*] Research work

[**] School of Physics and Mechanical Engineering. School of Engineering and Information Systems. Director Ing. Sonia Cristina Gamboa Sarmiento, Ph.D. Co-director Ing. Luis Carlos Gómez Flórez, M.Sc.

INTRODUCCIÓN

Los grupos de investigación con frecuencia se ven en la necesidad de realizar diferentes tipos de desarrollo software como resultado directo y para apoyar sus actividades de investigación. Hay que tener en cuenta que los grupos de investigación trabajan sobre temas novedosos en los que pueden no existir herramientas software que satisfagan sus necesidades de forma adecuada. El desarrollo software en grupos de investigación generalmente es realizado por estudiantes de pregrado en sus últimos niveles como parte de sus trabajos de fin de carrera. Un grupo de investigación que conoce esta situación de primera mano es el grupo de investigación en Sistemas y Tecnologías de la Información (STI), grupo avalado por la Universidad Industrial de Santander (UIS), cuenta con tres desarrollos reconocidos a nivel institucional: QUIS[1] usado para medir la calidad del producto, proceso y proyecto de un desarrollo software; HSLAB[2] usado para gestionar la información de los servicios de ensayo de laboratorios de análisis de muestras según la norma ISO 17025; y CYSACJ[3] usado para apoyar el control de las actuaciones del consultorio jurídico de la UIS. Cada una de estas aplicaciones inició y ha sido extendida a través de proyectos de final de carrera. Durante este proceso, el grupo STI, ha encontrado que al iniciar un proyecto para extender el software existente es necesario invertir una cantidad de tiempo considerable en la corrección de errores antes de implementar las nuevas funcionalidades.

[1] León Martínez, N. E., Mendoza Castellanos, A., & Gómez Flórez, L. C. Propuesta de un sistema para la evaluación de calidad de software derivado de actividades de investigación. Bucaramanga. 2011

[2] Ramírez Gómez, H., De La Hoz Freyle, J. E., & Gómez Flórez, L. C. Herramienta software basada en la arquitectura soa para el control de procesos del servicio de ensayo y mantenimiento de equipos del laboratorio de cromatografía de la Universidad Industrial de Santander. 2010

[3] Osorio Sanabria, M. A., Gonzáles Zabala, M. P., & Gómez Flórez, L. C. Sistema de informacion para apoyar el control de las actuaciones de los estudiantes del consultorio juridico de la UIS CYSACJ-UIS. 2006

Por lo anterior, el grupo STI buscó mejorar la calidad de estos desarrollos software y es ahí donde surge este proyecto que tiene por objetivo mejorar la calidad de los desarrollos software en grupos de investigación a través de la propuesta un modelo que estructura y guía el proceso de pruebas de software. Para lograr esto se realizó una revisión bibliográfica en Scopus que buscó identificar lineamientos para realizar las pruebas de software en pequeñas y medias empresas (PyME), para lo cual se realizó una revisión sistemática que se basó en los lineamientos de Kitchenham[4] donde se identificó el uso de técnicas estáticas de pruebas de software en PyME. Adicionalmente, se estudió los proyectos de final de carrera de estudiantes de pregrado y posgrado pertenecientes la Escuela de Ingeniería de Sistemas e Informática con el propósito de identificar la forma como se desarrolla el software en grupos de investigación. Como resultado de esta actividad se encontró que el software realizado en los trabajos de final de carrera, de forma general, cuenta con un bajo nivel de calidad. La información recolectada permitió formular un modelo para gestionar las pruebas de software en grupos de investigación. Para formular este modelo se partió del modelo ISO/IEC/IEEE 2119. El modelo propuesto se compone de 3 niveles que van desde la formulación de lineamientos, generales, de pruebas de software por parte del grupo de investigación hasta la realización de pruebas por parte del estudiante para su posterior seguimiento. El modelo fue aplicado a cuatro desarrollos software del grupo de investigación STI produciendo, un software cuyo proceso de pruebas formuló y ejecutó casos de pruebas que permitieron la detección y corrección oportuna de problemas. Mejorando de esta forma la calidad del software generado.

[4] Kitchenham, B. Guidelines for performing Systematic Literature Reviews in Software Engineering, Version 2.3. 2007

1. PRESENTACIÓN DEL PROYECTO

1.1 SITUACIÓN ACTUAL

El grupo de investigación en Sistemas y Tecnologías de la Información (STI), avalado por a la Universidad Industrial de Santander (UIS), cuenta con tres desarrollos de software reconocidos a nivel institucional: QUIS[5] usado para medir la calidad del producto, proceso y proyecto de un desarrollo software; HSLAB[6] usado para gestionar la información de los servicios de ensayo de laboratorios de análisis de muestras según la norma ISO 17025; y CYSACJ[7] usado para apoyar el control de las actuaciones del consultorio jurídico de la UIS. Estos desarrollos de software surgieron como resultado de investigaciones llevadas a cabo dentro del grupo. Conforme se ha ido avanzado en las líneas de investigación ha sido necesario adicionar nuevas funcionalidades y realizar labores de mantenimiento a estos desarrollos. Al realizar estos trabajos, el grupo STI ha encontrado que al iniciar un proyecto para extender el software existente, es necesario invertir una cantidad de tiempo considerable en la corrección de errores antes de implementar las nuevas funcionalidades.

Hay que tener en cuenta que el desarrollo software en grupos de investigación generalmente es realizado por estudiantes de pregrado en sus últimos niveles como parte de sus trabajos de fin de carrera. Los problemas encontrados al extender las funcionalidades de los mismos permiten afirmar que la calidad de estos desarrollos software no es la mejor. Al revisar los trabajos de final de carrera

[5] León Martínez, N. E., Pimentel Ravelo, J. I., & Gómez Flórez, L. C. Herramienta computacional para la gestión y evaluación de procesos software enmarcados en actividades de investigación. Scientia et Technica, 3(49). 2011

[6] Ramírez Gómez, De La Hoz Freyle, & Gómez Flórez, Op. Cit.

[7] Osorio Sanabria, Gonzáles Zabala, & Gómez Flórez, Op. Cit.

de miembros de otros grupos de investigación, ver anexo A, es claro que los problemas en la calidad no son un tema exclusivo del grupo STI. En general, el proceso de desarrollo software tiende a ser poco estructurado e informal. Lo cual puede ser el resultado de la falta de experiencia de los estudiantes al realizar estos desarrollos software. Sin embargo, los problemas en la calidad de software no es un tema nuevo, como lo describe Sommerville en su libro, Software Engineering (2011), desde que se inició con el desarrollo de sistemas de gran tamaño y complejidad, en 1960 y durante todo el siglo 20. El software desarrollado era poco confiable y difícil de mantener[8]. Por lo anterior, el aseguramiento de la calidad ha surgido como un tema de vital importancia en todo desarrollo software. El aseguramiento de la calidad define un conjunto de procesos y actividades necesarios para proporcionar confianza en que el proceso o producto software cumplirá con los requerimientos técnicos y de calidad establecidos. El proceso para el aseguramiento de la calidad usa los resultados de las pruebas junto otra información para clasificar y reportar cualquier problema en el diseño, planeación y ejecución del proceso de la ingeniera del software[9]. En este punto es donde se introduce el concepto de pruebas el cual se define como un "Proceso que consiste en todas las actividades del ciclo de vida software, tanto estáticas como dinámicas, concernientes con la planificación, preparación y evaluación de productos software y los productos de trabajo relacionados para determinar que éstos satisfacen los requisitos especificados, para demostrar que se ajustan al propósito y para detectar defectos"[10]. Habitualmente, las pruebas de software suelen ser simplificadas a una actividad al final de los proyectos donde se hace uso del software para encontrar errores. Sin embargo, como lo sugiere la definición anterior las pruebas de software son un proceso que necesita de planeación y preparación para determinar que un producto o proceso satisface los

[8] Sommerville, I. Software Engineering (9th ed.). 2011
[9] ISO/IEC/IEEE 29119-1. Software and systems engineering — Software testing — Part 1: Concepts and definitions. 2013
[10] ISTQB. Foundation Level Syllabus. 2011

requisitos establecidos, se ajuste al propósito para el cual fue creado y finalmente detectar defectos.

En los últimos años, El grupo STI ha trabajado en el mejoramiento de la calidad de los desarrollos de software en grupos de investigación y para ello ha buscado acercar normas y estándares internacionales, tales como: ISO/IEC 14598 estándar para evaluar la calidad del producto y ISO/IEC 9126 estándar para evaluar la calidad del software. Para lo cual se formuló un sistema para la evaluación de calidad de software derivado de actividades de investigación[11]. En el año 2012 el grupo STI incursionó en el tema de las pruebas de software al trabajar con el estándar para la documentación de pruebas software IEEE 829[12]. Continuando con esta línea de investigación y recordando que las pruebas software son una herramienta de gran valor en el aseguramiento de la calidad[13] motivó el surgimiento de este trabajo que busca contribuir en el mejoramiento de la calidad del software desarrollado en grupos de investigación a través de las pruebas de software.

1.2 OBJETIVOS

1.2.1 Objetivo general. Proponer un modelo para gestionar los procesos Testing de los desarrollos software realizados en grupos de investigación de la UIS: Caso de Estudio grupo STI.

[11] Leon Martinez, N. E., Aguilar Garcia, J. L., Vega Morales, E. F., & Gomez Florez, L. C. Herramienta computacional para la documentación de pruebas de software enmarcado en actividades de investigación Computer tool for software testing documentation under research activities. Scientia et Technica, 18(4), 682–689. 2013
[12] Ibid
[13] ISO/IEC/IEEE 29119-1. 2013, Op. Cit.

1.2.2 Objetivos específicos

1. Revisar en la literatura científica trabajos relacionados con la gestión del Testing y los procesos de desarrollo software en organizaciones pequeñas y medianas. Esto con el propósito de identificar lineamientos y marcos de referencia sobre los que se fundamentará el modelo a plantear.
2. Describir los procesos de desarrollo software empleados por los grupos de investigación de la UIS. Esto con el fin de determinar sus características más representativas.
3. Plantear el modelo bajo el cual se puede gestionar los procesos Testing de los desarrollos software efectuados en grupos de investigación de la UIS. Para ello se hará uso de lo encontrado en la literatura y de las características propias de los desarrollos realizados en estos grupos.
4. Aplicar el modelo propuesto a los desarrollos software realizados en el grupo de investigación STI, para ilustrar la implantación del modelo propuesto.
5. Desarrollar un prototipo software que implemente el modelo propuesto, para facilitar, agilizar y promover su socialización e implantación.

1.3 PLANTEAMIENTO DE LA METODOLOGÍA

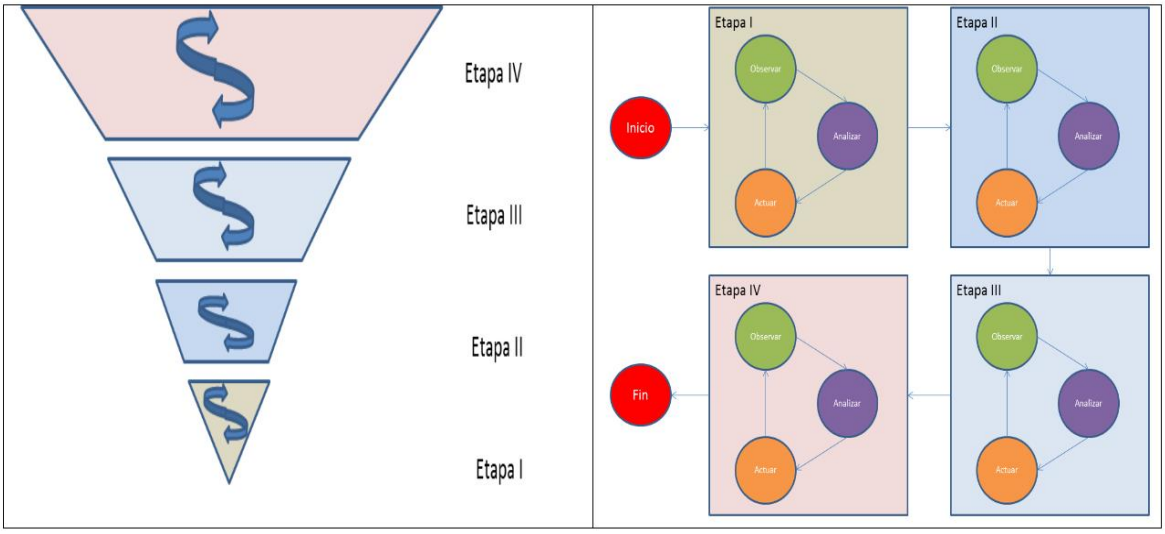
La metodología usada en el desarrollo de este proyecto se fundamentó en la naturaleza en espiral y las fases esenciales (observar, pensar y actuar) de la metodología de Investigación-acción[14]. La selección de la metodología base surge del propósito de formular un modelo que tenga en cuenta a los integrantes de los grupos de investigación para lo cual era importante unir la experiencia del

[14] Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucia, M. del P. Metodología de la investigación (5th ed.). 2010

investigador, que realizó este trabajo, con el conocimiento y vivencias de los integrantes de los grupos de investigación.

La metodología propuesta para esta investigación se conformó de cuatro (4) etapas; cada una de estas etapas constaba de tres fases: observar, analizar y actuar, en la figura 1 se encuentra una representación gráfica de la metodología propuesta. En la etapa I se estudió y contribuyó al mejoramiento del proceso de desarrollo del grupo STI. En la etapa II se revisó el proceso de desarrollo software en grupos de investigación de la UIS. En la etapa III se formuló una propuesta de cómo las pruebas de software podrían contribuir a mejorar la calidad de los desarrollos software en grupos de investigación, y se aplicó en el grupo de investigación STI. Finalmente, en la etapa IV se creó un prototipo en el cual se implementó el modelo propuesto y se aplicó en el grupo de investigación STI.

Figura 1 Etapas que compone el proyecto



En la tabla 1 se detallan las actividades realizadas en cada etapa, así como su contribución a los objetivos del proyecto.

Tabla 1. Resumen de las actividades desarrolladas en cada etapa propuesta.

Etapas	Fase	Actividades	Objetivo al que contribuye
I	Observar	- Se interactuó con los miembros del grupo STI y se revisaron los libros resultantes de los trabajos de final de carrera de los estudiantes de pregrado y posgrado del grupo de investigación STI donde se realizó un desarrollo software.	2
	Analizar	- Se analizó la información recolectada y se complementó con una revisión de la literatura científica relacionada con el aseguramiento de la calidad en el proceso de desarrollo software en las PyME.	1
	Actuar	- Se realizaron algunas recomendaciones de cómo mejorar el proceso de desarrollo software lo cual permitió definir los actores del modelo.	3 y 4
II	Observar	- Se revisaron los libros resultantes de los trabajos de final de carrera de los estudiantes de pregrado y posgrado de los grupos de investigación de "Filosofía y Enseñanza de la Filosofía" y "Grupo SIMON de Investigaciones en Modelamiento y Simulación" donde se realizó un desarrollo software.	2
	Analizar	- Se analizó la información recolectada y se complementó con una revisión de la literatura científica relacionada con el aseguramiento de la calidad en el proceso de desarrollo software en las PyME.	1
	Actuar	- Se planteó una serie de recomendaciones para mejorar la calidad de los desarrollos software que permitió definir los procesos que debe tener en cuenta el modelo.	3 y 4
III	Observar	- Se aplicó el modelo propuesto en los desarrollos software del grupo STI	4
	Analizar	- Se analizó el efecto del modelo en los desarrollo software.	3
	Actuar	- Se realizaron ajustes al modelo.	3 y 4
IV	Observar	- Se formuló el prototipo software y se implementó en el grupo STI	5
	Analizar	- Se analizó el efecto de usar de la herramienta software a la hora de incorporar el modelo.	4 y 5
	Actuar	- Se ajustó el prototipo.	4 y 5

2. MARCO TEÓRICO

El marco teórico de este proyecto surgió de la realización de dos revisiones, la primera de la literatura (Ver Anexo A) y la segunda en los trabajos de final de carrera (Ver Anexo B). La primera revisión se llevó a cabo en SCOPUS. Esta revisión fue guiada por los trabajos hechos por la profesora Kitchenham en el area de revisiones sistematicas. Según Kitchenham[15] una revisión sistemática es una forma de identificar, evaluar e interpretar todas las investigaciones relevantes y disponibles para una pregunta de investigación, un área temática o un fenómeno de interés. El proceso de revisión inició con la definición de un protocolo de revisión que se basó en la plantilla propuesto por (Biolchini, Mian, Candida, & Natali[16]. A continuación, se describen los pasos generales:

1. Formulación de la(s) pregunta(s) que busca responder la revisión.
2. Selección de las fuentes de consulta.
3. Selección de los estudios de interés.
4. Extracción de la información.
5. Resumen y resultados.

Para la segunda revisión se partió de lo aprendido en la revisión literaria y se planteó un proceso de revisión que se conforma de las siguientes actividades:

1- Determinar la fuente de los documentos.
2- Definir criterios de búsqueda.
3- Seleccionar documentos de interés (criterios de inclusión y exclusión).

[15] Kitchenham, B. a, Mendes, E., & Travassos, G. H. Cross- vs. Within-Company Cost Estimation Studies: A Systematic Review. IEEE Transactions on Software Engineering, 33(5), 316–329. 2007. Disponible en: http://doi.org/http://dx.doi.org/10.1109/TSE.2007.1001
[16] Biolchini, J., Mian, P. G., Candida, A., & Natali, C. Systematic Review in Software Engineering. 2005

4- Extraer información de interés de los libros.

5- Sintetizar y resumir la información encontrada.

6- Interpretar los resultados.

2.1 DESARROLLO SOFTWARE EN PYME

Las pequeñas y medianas empresas (PyME) desarrolladoras de software tienen un gran impacto en el crecimiento de países tales como; Estados Unidos, Brasil, Canadá, China, India, Finlandia, Irlanda y Hungría, donde representan hasta un 85% de la industria del software[17]. Sin embargo, las PyME poseen recursos limitados y bajos niveles de madurez en sus procesos de desarrollo software[18]. Las PyME no son sólo versiones reducidas de las grandes compañías[19] ya que las grandes compañías y las PyME tienen culturas muy diferentes. Las PyME son más informales, no tienden a seguir estándares y se autoorganizan[20].

De acuerdo con Basri & O'Connor[21] en su conferencia "Understanding the Perception of Very Small Software Companies towards the Adoption of Process Standards" el ciclo de vida de los desarrollos software en PyME generalmente es muy simplificado y algunas de las fases de desarrollo no se encuentran formalizadas. La mayoría de los proyectos son guiados por estrategias a corto plazo, por lo cual no se da mucha importancia a la gestión del riesgo. Las PyME tienen unos recursos económicos bastante limitados y un número limitado de

[17] Richardson, I., & Von Wangenheim, C. G. Why Are Small Software Organizations Different? IEEE Software, 24(1). 2007. Disponible en: http://doi.org/10.1109/MS.2007.12

[18] Batista, & Diaz de Figueiredo, Op. Cit.

[19] Richardson, & Von Wangenheim, Op. Cit.

[20] Pressman, R. S. Software Engineering A pracitioner's Approach (7th ed.). McGraw-Hill Science/Engineering/Math. 2009

[21] Basri, S., & O'Connor, R. V. Understanding the Perception of Very Small Software Companies towards the Adoption of Process Standards. 2010

personal calificado[22], razón por la cual no pueden adquirir la experiencia requerida[23]. De acuerdo a las profesoras Richardson y Von Wangenheim (2007) en "Why are Small Software Organizations Different? a pesar de que hay muchos modelos y estándares para la evaluación y mejora del proceso de software, generalmente no son conocidos por las PyME. Las personas en estas organizaciones creen frecuentemente que las buenas prácticas son costosas, consumen mucho tiempo y se enfocan solo en compañías grandes, y por lo tanto son difíciles de aplicar en organizaciones pequeñas. A diferencia de las grandes compañías, las PyME no tienen suficiente personal para desarrollar funcionalidades específicas que les permitan realizar tareas secundarias a su producto[24]. Según Batista y Diaz de Figueiredo en su trabajo "SPI in a Very Small Team: a Case with CMM" aun cuando una PyME pueda tener varios años en el negocio, suelen no contar con una coordinación interna o externa; además sus habilidades técnicas y prácticas no son muy sofisticadas[25].

Finalmente, se podría decir que las PyME se enfrentan a problemas similares con relación a la mejora de la calidad del software y el proceso de evaluación que las grandes compañías, sin embargo, la principal diferencia radica en que las PyME rara vez tienen recursos especializados o competentes para resolver estas dificultades[26].

[22] Yeşildoruk, F. Ç., Bozlu, B., & Demirörs, O. The Tool Coverage of Software Process Improvement Frameworks for Small and Medium Sized Enterprises. 2009
[23] Richardson, & Von Wangenheim, Op. Cit.
[24] Ibid.
[25] Batista, & Diaz de Figueiredo, Op. Cit.
[26] Wangenheim, C. G. Von, & Varkoi, T. Standard Based Software Process Assesments in Small Companies. Software Process: Improvement and Practice, 11(3). 2006. Disopnible en: http://doi.org/10.1002/spip

2.2 DESARROLLO SOFTWARE EN GRUPOS DE INVESTIGACIÓN

De acuerdo con Colciencias, un grupo de investigación se define como "el conjunto de personas que se reúnen para realizar investigación en una temática dada, formulan uno o varios problemas de su interés, trazan un plan estratégico de largo o mediano plazo para trabajar en él y producir unos resultados de conocimiento sobre el tema en cuestión"[27]. A su vez, Colciencias establece que los grupos de investigación están conformados por cuatro tipos de integrantes: investigador, investigador en formación, estudiantes de pregrado e integrante vinculado[28]. Generalmente, dentro de un grupo de investigación se reconocen tres roles básicos: el director, el líder de línea de investigación y los auxiliares de investigación[29]. El director es un docente con experiencia en investigación, que orienta a los demás miembros en el desarrollo de sus actividades. Los líderes de investigación suelen ser profesionales que en algunas ocasiones se encuentran realizando sus posgrados. Los auxiliares normalmente son estudiantes de pregrado. Durante el desarrollo de sus actividades o como resultados de las mismas un grupo de investigación puede verse en la necesidad de realizar algún tipo de desarrollo software[30]. Sin embargo, si el desarrollo software no es la actividad central de un grupo de investigación, los recursos para tal fin son limitados. Es de notar que el desarrollo software en grupos de investigación es, generalmente, realizado por estudiantes de últimos niveles como parte de sus trabajos de fin de carrera. Un ejemplo de esto es el grupo STI cuyas líneas de

[27] Colciencias. Disponible en: http://www.colciencias.gov.co/faq última. Fecha de consulta 10/03/2016

[28] Colciencias. Modelo De Medición De Grupos De Investigación, Desarrollo Tecnológico O De Innovación Y De Reconocimiento De Investigadores Del Sistema Nacional De Ciencia, Tecnología E Innovación. 2015. Disopnible en: http://www.colciencias.gov.co/sites/default/files/ckeditor_files/files/DOCUMENTO MEDICIÓN GRUPOS - INVESTIGADORES VERSIÓN FINAL 15 10 2014 (1).pdf

[29] León Martínez, N. E., Pinto Chacon, N., & Gómez Flórez, L. C. Herramienta computacional para la evaluación de calidad de productos software enmarcados en actividades de investigación. Scientia et Technica, XVI(48). 2011

[30] León, N. E., & Pimentel, J. I. Herramienta computacional para la gestión y evaluación de proyectos software enmarcados en actividades de investigación. Scientia et Technica, (47). 2011

investigación[31] son: Administración de la información y Gestión del Conocimiento en las organizaciones; Auditoría y control en Sistemas y Tecnologías de Información; Gestión de procesos y calidad en Ingeniería del software; Pensamiento y Metodología de los de Sistemas Blandos -MSB-; Planificación y Gerencia de Sistemas y Tecnologías de Información; Sistemas y Tecnologías de la Información en las organizaciones. Ninguna de sus líneas de investigación exige de forma explícita la realización de un desarrollo software para su ejecución. Sin embargo, el grupo STI ha realizado desarrollos software como resultado del trabajo en sus líneas de investigación.

El equipo de trabajo de un proyecto de fin de carrera de pregrado se compone generalmente de uno o dos docentes en el rol de director(es) y de uno o dos estudiantes en el caso de pregrado o un estudiante en el caso de posgrado. Al no contar con mucho personal para el desarrollo de un proyecto cada uno de los miembros de estos equipos puede encargarse de varios roles conforme el proyecto avanza.

En el caso de las Universidad Industrial de Santander (UIS), la Escuela de Ingeniería de Sistemas e Informática (EISI), debido a su área de estudio, es la llamada a realizar los desarrollos de software en la institución. La EISI cuenta con 6[32] grupos de investigación avalados por la UIS, listados a continuación:

- Filosofía y enseñanza de la Filosofía (FILOEN).
- Grupo de Investigación en Sistemas y Tecnología de la Información STI (STI).
- Grupo SIMON de Investigaciones en Modelamiento y Simulación (SIMON).

[31] Grupo de Investigación en Sistemas y Tecnología de la Información STI. Disponible en: http://scienti.colciencias.gov.co:8080/gruplac/jsp/visualiza/visualizagr.jsp?nro=00000000001261. Última fecha de consulta 10/03/2016

[32] Listado de Grupos Facultad de Ingenierías Fisicomecánicas. Disponible en: https://www.uis.edu.co/webUIS/es/investigacionExtension/gruposInvestigacion/gruposFisicoMecanicas.html última fecha de consulta 10/08/2016

- Grupo de Investigación en Ingeniería Biomédica.
- Grupo de Investigación en Diseño de Algoritmos y Procesamiento de Datos Multidimensionales (HSDP).
- Supercomputación y Cálculo Científico (SC3)

Cada grupo tiene unas líneas o temáticas de investigación. El trabajo continuo sobre ellas genera nuevos proyectos los cuales pueden necesitar el desarrollo de un software o extender las funcionalidades de un software existente.

En esta investigación fueron revisados los trabajos de final de carrera, donde se realizó algún tipo de desarrollo software, de los grupos STI, FILOEN y SIMON. Ver anexo A. Los desarrollos software de estos grupos son guiados por el director del grupo, que es la persona de mayor experiencia y supervisa todos los trabajos realizados en el grupo. Es de notar que el director suele recibir ayuda, de los investigadores con más experiencia del grupo, en esta supervisión. En la revisión se encontró que un 91% de los trabajos del STI, un 90% de los trabajos del FILOEN y 73% de los trabajos de SIMON definieron de forma explícita algún tipo de metodología de software. Sin embargo, al revisar, más a fondo los trabajos con una metodología definida, se encontró que solo un 56% de estos trabajos en STI y FILON y un 75% en SIMON contaban con evidencia del seguimiento de la metodología. De los grupos revisados SIMON fue el que obtuvo un mejor comportamiento, sin embargo, si se tiene en cuenta que el trabajo de final de carrera es donde el estudiante aplica sus conocimientos y se encuentra que 75% de un 73% de los trabajos, cuentan con evidencia de haber seguido la metodología propuesta, no es un resultado muy positivo. Lo cual conduce a pensar que por su inexperiencia los estudiantes suelen no realizar el proceso de desarrollo software de una forma estructurada como lo proponen las metodologías de desarrollo software. Un dato que llamó la atención de la revisión fue cuando se buscó evidencia explicita del uso del desarrollo software por el usuario final donde se encontró que solo un 43% de los trabajos del STI, un 20% de los trabajos de

FILOEN y 73% de los trabajos contaban con dicha evidencia. Sin embargo, al buscar capturas de pantalla del desarrollo software se encontró que un 86% de los trabajos del STI, un 90% de los trabajos del FILOEN y un 73% de los trabajos SIMON contaban con capturas de pantalla del desarrollo software. Es decir, en una buena parte de los trabajos de final de carrera revisados, no se tiene certeza de que los desarrollos software hayan sido realizados de forma completa, pues no se tiene evidencia de haber sido usados por el usuario final, y generalmente solo se tienen capturas de pantalla del mismo.

2.3 RELACIÓN ENTRE PYME DESARROLLADORAS DE SOFTWARE Y LOS GRUPOS DE INVESTIGACIÓN

Al comparar los problemas de las PyME con lo encontrado en grupos de investigación surgen varias semejanzas. En ambos casos se cuenta con una cantidad limitada de recursos, no se tienden a seguir estándares ni metodologías y el proceso de desarrollo software es bastante informal, lo cual se evidencia en lo encontrado por Basri y O'Conner (2010) en PyME y el bajo porcentaje de evidencia en el seguimiento de las metodologías de software por parte de los estudiantes de pregrado de los grupos de investigación como STI y FILOEN. Es de notar que aun cuando se observa un comportamiento similar, las razones detrás de estos comportamientos no son necesariamente las mismas. En el caso de los desarrollos software en grupos de investigación su desarrollo no suele estar ligado a un beneficio económico, el cual en algunas situaciones puede existir, pues los estudiantes buscan realizar sus trabajos de final de carrera con el objetivo de obtener sus títulos como profesionales. Sin embargo, en este punto es donde se unen estos dos escenarios pues son estos egresados la principal fuente de personal de las PyME. Una de las razones de esto son los bajos costos, a corto plazo, en los que incurre una PyME al contratar un recién egresado. Sin embargo,

como las PyME, generalmente, no tienen mecanismos claros para el aseguramiento de la calidad los recién egresado pueden contribuir de forma negativa al proceso de desarrollo software de la organización.

2.4 PRUEBAS DE SOFTWARE

Una de las primeras definiciones de pruebas de software y posiblemente la más conocida fue la propuesta por Myers (1979), en su libro The Art Of Software Testing, quien la definió como "el proceso de ejecutar un programa con el propósito de encontrar errores"[33]. En esta definición, podría decirse, se capta la esencia de lo que son las pruebas de software. Sin embargo, este concepto ha ido evolucionando con el tiempo y los estándares lo han definido de una forma más precisa. La IEEE la ha definido como: "el proceso de hacer funcionar un sistema o componente bajo condiciones especificadas, observando o registrando los resultados, y haciendo una evaluación de algunos aspectos del sistema o componente"[34] y "Actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, los resultados son observados o registrados, y una evaluación es hecha de algún aspecto del sistema o componente"[35].

La International Software Testing Qualifications Board (ISTQB), una organización sin ánimo de lucro que certifica pruebas de software, define las pruebas de software como un "Proceso que consiste en todas las actividades del ciclo de vida software, tanto estáticas como dinámicas, concernientes con la planificación, preparación y evaluación de productos software y los productos de trabajo

[33] Myers, G. J., Badgett, T., Thomas, T. M., & Sandler, C. The Art of Software Testing (2nd ed.). 2004
[34] IEEE Std 610.12-1990. IEEE Standard Glossary of Software Engineering Terminology (Vol. 121990). 1990
[35] IEEE Std 829. Standard for Software and System Test Documentation (Vol. 2008). 2008

relacionados para determinar que éstos satisfacen los requisitos especificados, para demostrar que se ajustan al propósito y para detectar defectos"[36]. Una definición más reciente es la propuesta por Bourque & Fairley (2014), según la cual "las pruebas de software consiste en la verificación dinámica que un programa se comporta de forma esperada ante un conjunto finito de casos de prueba, adecuadamente seleccionados de, normalmente, un dominio infinito de ejecuciones"[37]. Al observar las definiciones anteriores de pruebas de software, es claro que con el pasar del tiempo esta actividad ha ido evolucionando y ya no solo se trata de encontrar defectos[38].

Según el NIST (2002) los errores en el software cuestan alrededor de 59.5 mil millones de dólares anualmente y cerca de un tercio de los mismos, 22.2 mil millones de dólares, podrían ser eliminados a través de las pruebas de software[39]. Las pruebas de software aportan información valiosa para el aseguramiento de la calidad[40]. Sin embargo, las pruebas de software suelen considerarse una actividad laboriosa. Esto como resultado de que los casos de pruebas son normalmente generados, ejecutados y analizados de forma manual[41]. Según Sommerville[42] las pruebas de software representan cerca de un 40% del costo del software[43]. En las grandes compañías muchas personas participan en las pruebas de software. Sin embargo, en las PyME estos roles pueden solaparse[44]. De la revisión realizada a los desarrollos software en grupos STI, FILOEN y SIMON se encontró que el nivel de pruebas que se realizó con mayor frecuencia fue el de sistema y la técnica más

[36] ISTQB. Foundation Level Syllabus. 2011
[37] Bourque, P., & Fairley, R. E. Guide to the Software Engineering - Body of Knowledge. IEEE Computer Society. 2014. Disponible en: http://doi.org/10.1234/12345678
[38] Hapter, C. Software Testing. In SWEBOK. 2004
[39] NIST. Software Errors Cost U.S. Economy $59.5 Billion Annually. 2002. DIsponible en: http://www.abeacha.com/NIST_press_release_bugs_cost.htm
[40] ISO/IEC/IEEE 29119-1. 2013. Op. Cit.
[41] Naik, K., & Tripathy, P. Software Testing and Quality Assurance. Hoboken, NJ, USA: John Wiley & Sons, Inc. 2008. Disponible en: http://doi.org/10.1002/9780470382844
[42] Sommerville, I. Software Engineering (9th ed.). 2011
[43] Ibid
[44] Vinay, P. F. Manage Software Testing. 2008

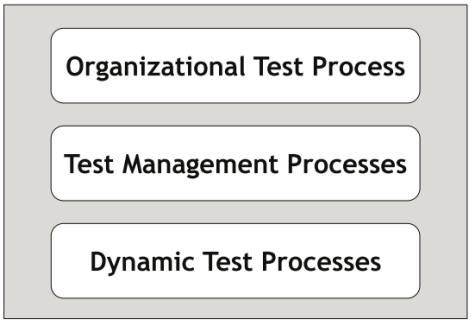
usada fue la de caja blanca. Ver anexo B. Es decir, las pruebas de software suelen ser realizadas en etapas avanzadas de los desarrollos software. Lo cual puede ser el resultado de la inexperiencia de los estudiantes en el tema de las pruebas de software. Hecho que es consistente con el poco tiempo que dedica el plan de estudio de Ingeniería de Sistemas al tema de las pruebas de software, ver anexo C. Un hallazgo que llamo la atención de la revisión de los trabajos de los grupos STI, FILOEN y SIMON fue el bajo nivel de pruebas de aceptación encontrado. Lo cual indica que no se tienden a realizar pruebas orientadas al cumplimiento de los requisitos y restricciones definidos por el cliente.

2.5 ESTÁNDAR DE PRUEBAS DE SOFTWARE ISO/IEC/IEEE 29119

Como resultado de la revisión bibliográfica realizada en SCOPUS se identificaron los siguientes estándares: el IEEE 829 que es un estándar para documentar pruebas de software; el IEEE 1008 que es un estándar para la realización de pruebas unitarias; el BS 7925-1 que es un estándar que define un vocabulario de pruebas; y el BS 7925-2 que es un estándar para la realización de pruebas de componentes. Sin embargo, en el año 2013 surge el estándar ISO/IEC/IEEE 29119 el cual unifica los estándares antes mencionados. El estándar internacional de pruebas ISO/IEC/IEEE 29119 define un modelo de tres capas; el modelo inicia con una capa organizacional que gestiona a un alto nivel la especificación de las pruebas, tales como la política organizacional de prueba y la estrategia organizacional de prueba. La capa media se encarga de la gestión de las pruebas, mientras que la capa inferior define una serie de procesos empleados en las pruebas dinámicas[45]. En la figura 2 se encuentra un diagrama con los niveles mencionados.

[45] ISO/IEC/IEEE 29119-1. 2013. Op. Cit.

Figura 2 Relación entre las múltiples capas y el proceso de pruebas



Fuente: ISO/IEC/IEEE 29119-1. Software and systems engineering — Software testing — Part 1: Concepts and definitions. 2013

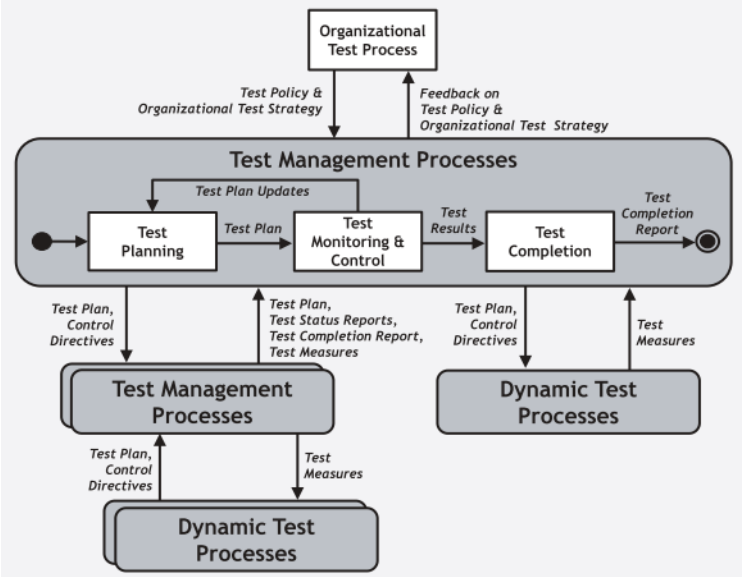
Este trabajo se enfoca en la gestión de las pruebas de software, es por esto que se procede a detallar la capa intermedia del modelo del estándar ISO/IEEE 29119. En ella se define el proceso de gestión de pruebas, el cual se compone de tres subprocesos[46], listados a continuación:

- Planeación de pruebas
- Monitoreo y control de pruebas
- Finalización de pruebas

El proceso de gestión de pruebas puede ser aplicado en un proyecto, a un nivel de pruebas específico y para gestionar diferentes tipos de pruebas[47]. A continuación, se encuentra una figura donde se ilustra este proceso.

[46] ISO/IEC/IEEE 29119-2. Software and systems engineering — Software testing — Part 2: Test processes. 2013
[47] Ibid.

Figura 3. Ejemplo de las relaciones del proceso de gestión de pruebas.



Fuente: ISO/IEC/IEEE 29119-2. Software and systems engineering — Software testing — Part 2: Test processes. 2013. Pág. 15

A continuación, se procede a realizar una breve descripción de los tres procesos que componen la capa de gestión de pruebas.

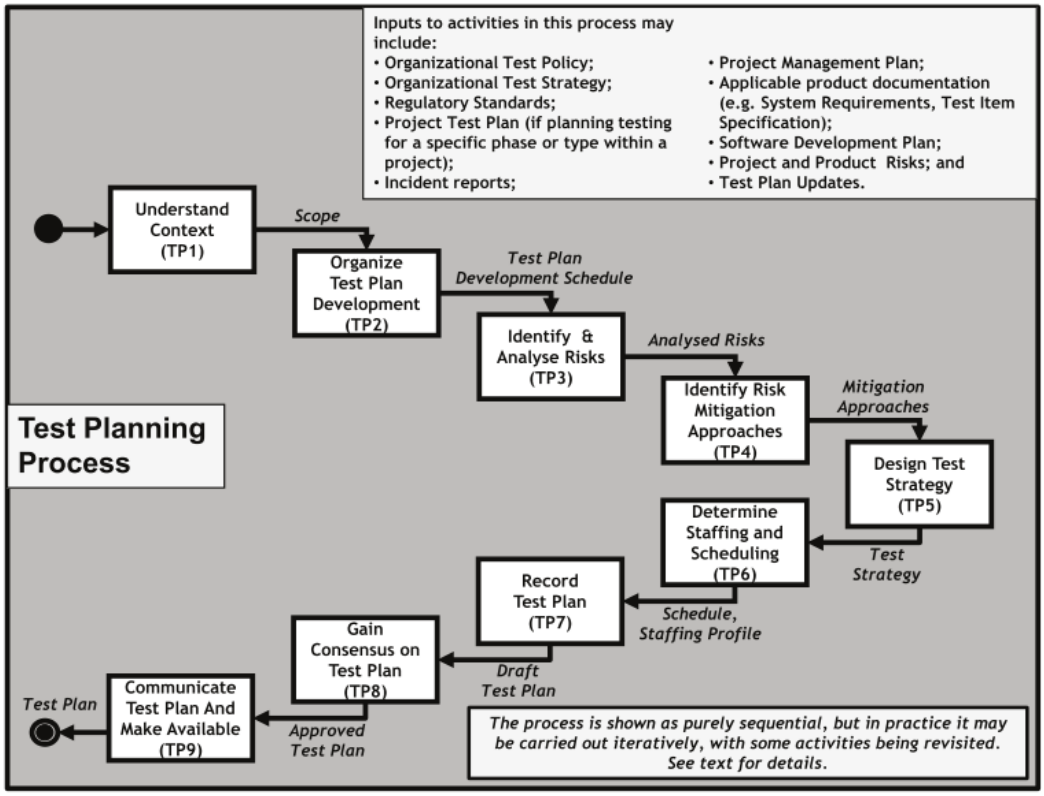
2.5.1 Proceso de planeación de pruebas. El proceso de planeación de pruebas es usado para desarrollar el plan de pruebas, para lo cual se definen nueve actividades, las cuales se encuentran listadas en la tabla 2. La interacción entre cada una de estas actividades se muestra la figura 4.

Tabla 2. Actividades del proceso de planeación de pruebas

Sigla	Actividad
TP1	Entender el contexto
TP2	Ordenar el desarrollo del plan de pruebas(TP2):
TP3	Identificar y analizar los riesgos
TP4	Identificar lineamientos para mitigar el riesgo
TP5	Diseñar la estrategia de pruebas
TP6	Definir el personal y cronograma

Sigla	Actividad
TP7	Registrar del plan de pruebas
TP8	Obtener consenso del plan de pruebas
TP9	Comunicar el plan de pruebas y hacerlo disponible

Figura 4. Proceso de planeación de pruebas.



Fuente: ISO/IEC/IEEE 29119-2. Software and systems engineering — Software testing — Part 2: Test processes. 2013

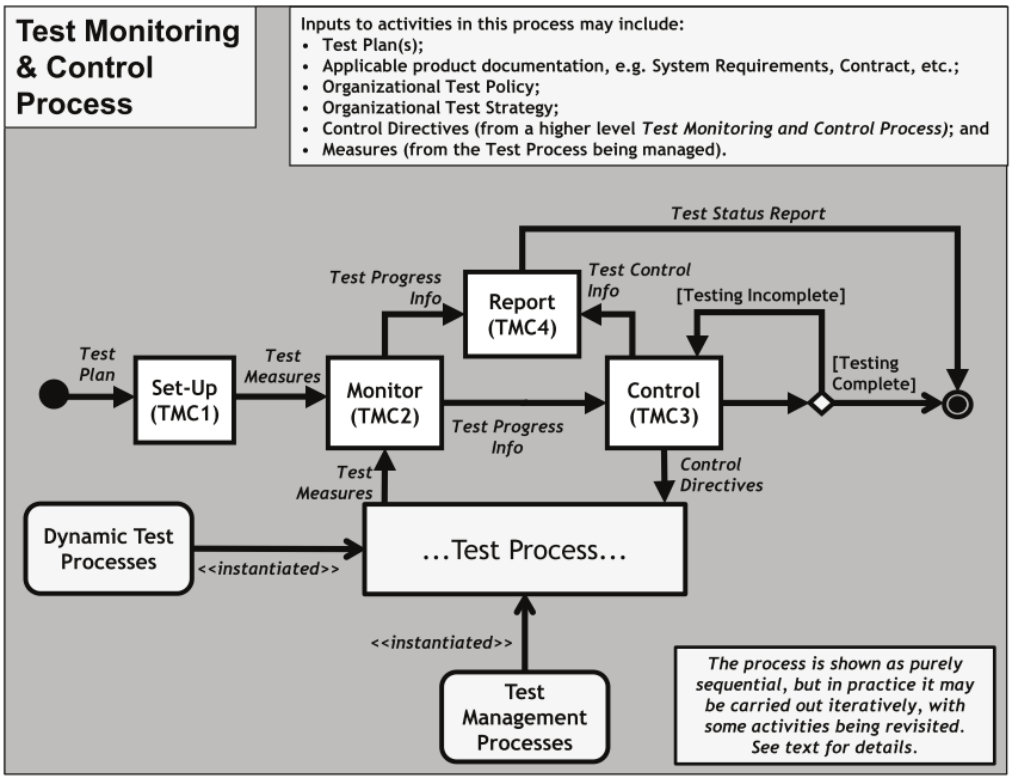
2.5.2 Proceso de monitoreo y control de pruebas. El proceso de monitoreo y control de pruebas determina si el proceso de pruebas se está llevando a cabo de acuerdo al plan de pruebas y las especificaciones organizacionales de las pruebas (por ejemplo, la política y estrategia organizacional de pruebas).

Además, este proceso se encarga de establecer acciones de control e identifica posibles actualizaciones necesarias al plan de pruebas, para lo cual se definen cuatro actividades, las cuales se encuentran listadas en la tabla 3. La interacción entre cada una de estas actividades se muestra la figura 5.

Tabla 3. Actividades del Proceso de Monitoreo y Control de Pruebas

Sigla	Actividad
TMC1	Preparación
TMC2	Monitoreo
TMC3	Control
TMC4	Reporte

Figura 5. Proceso de monitoreo y control de pruebas.



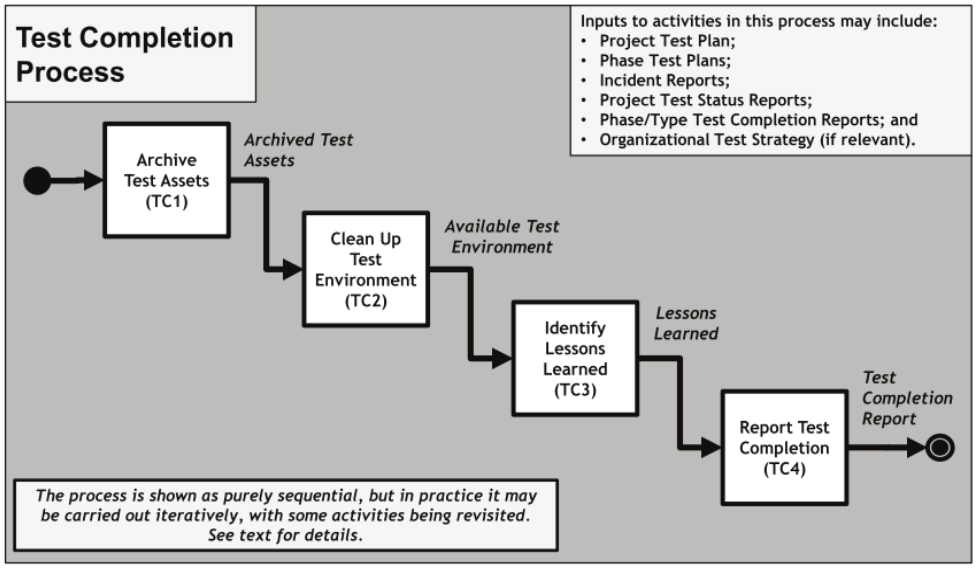
Fuente: ISO/IEC/IEEE 29119-2. Software and systems engineering — Software testing — Part 2: Test processes. 2013

2.5.3 Proceso de finalización de pruebas. Este proceso tiene el objetivo de hacer disponible los activos de pruebas para su posterior uso, dejar el entorno de pruebas en una condición satisfactoria, guardar y comunicar los resultados relevantes a los interesados. Los activos de las pruebas incluyen el plan de pruebas, especificación de casos de pruebas, scripts de pruebas, herramientas de pruebas, datos de prueba y la infraestructura del entorno de pruebas, para lo cual se define cuatro actividades, las cuales se encuentran listadas en la tabla 4. La interacción entre cada una de estas actividades se muestra la figura 5.

Tabla 4. Actividades del Proceso de Finalización de Pruebas

Sigla	Actividad
TC1	Archivar activos de pruebas
TC2	Limpiar el entorno de pruebas
TC3	Identificación lecciones aprendidas
TC4	Reporte de finalización de pruebas

Figura 6. Proceso de finalización de pruebas.



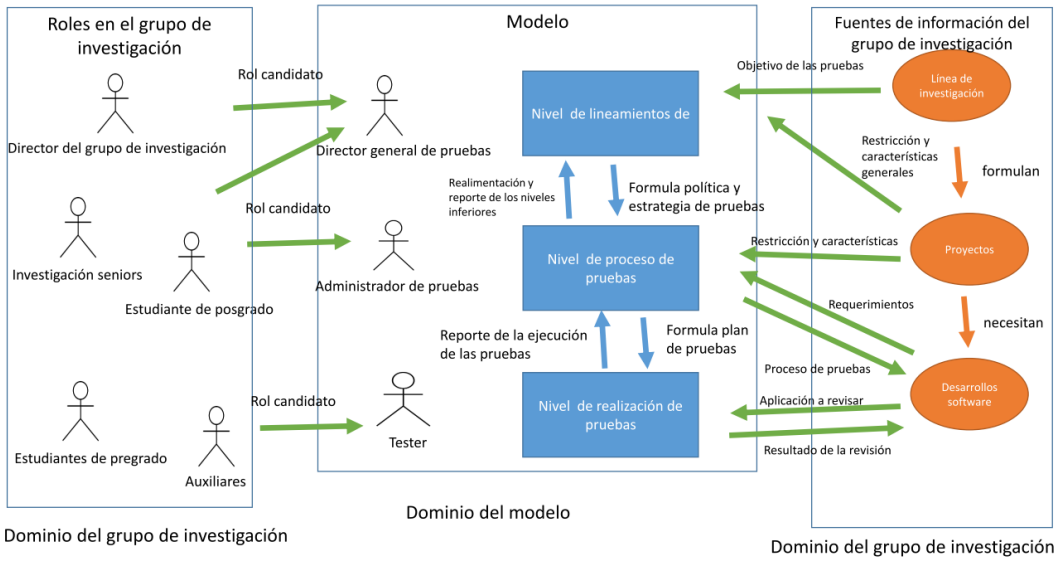
Fuente: ISO/IEC/IEEE 29119-2. Software and systems engineering — Software testing — Part 2: Test processes. 2013

3. MODELO PROPUESTO

3.1 PLANTEAMIENTO DEL MODELO

El objetivo de este trabajo es construir un modelo que proporcione una serie de lineamientos que ayuden a los integrantes del grupo en la planificación, construcción y ejecución de las pruebas de los desarrollos software realizados mejorando la calidad de los mismos. El modelo desarrollado en este trabajo se fundamentó en la información recolectada en la revisión de los trabajos de final de carrera, las características de los grupos de investigación (estructura del grupo y recurso humano disponible), la información recolectada en la revisión bibliográfica y el estándar ISO/IEC/IEEE 29119. En la figura 7 se muestra un diagrama resumen del modelo.

Figura 7. Diagrama resumen del modelo.

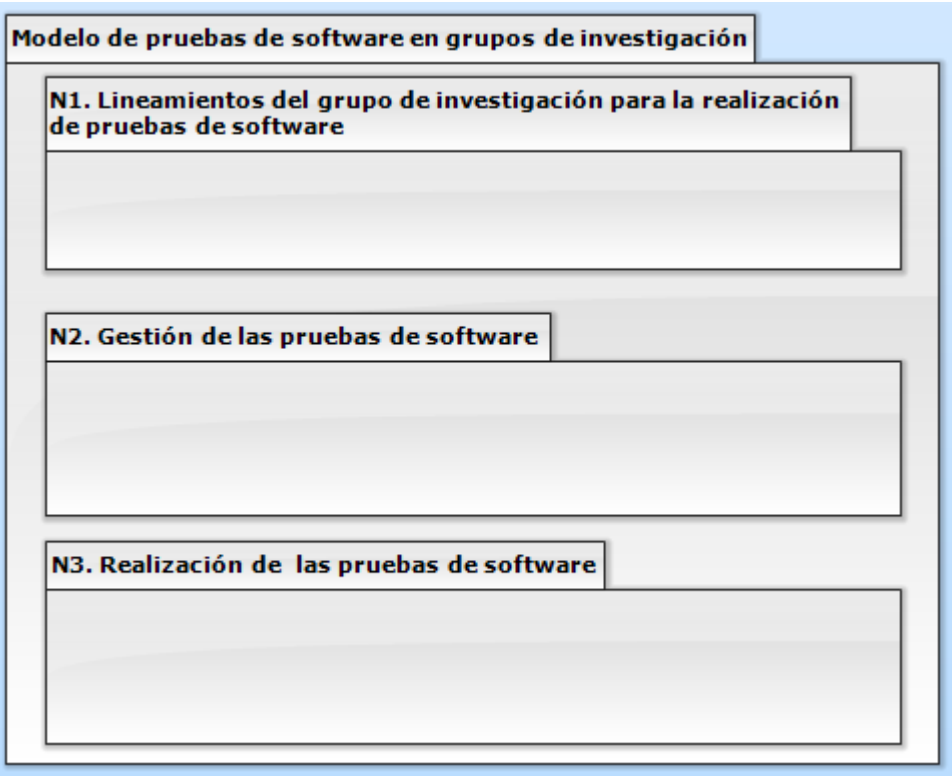


3.2 ESTRUCTURA GENERAL DE MODELO

El modelo se compone de tres niveles. En el primer nivel se encuentran los lineamientos definidos por el grupo de investigación para la realización de pruebas de software, en el segundo nivel se encuentra la gestión de las pruebas para un desarrollo software específico y finalmente, en el tercero nivel se encuentra la construcción y ejecución de las pruebas, en la figura 8 se muestran los niveles propuestos en el modelo. A continuación, se describe cada nivel:

1. Lineamientos del grupo de investigación para la realización de las pruebas de software: El grupo de investigación debe plantear la política y estrategias que guiará las pruebas de software en los desarrollos software dentro del grupo de investigación.
2. Gestión de las pruebas de software: Antes de realizar cualquier acción de pruebas es necesario revisar la política y estrategia planteada por el grupo. Una vez, se conoce esta información se procede a definir el proceso de prueba que el desarrollo software debe seguir. En este nivel se debe realizar seguimiento de las actividades propuestas.
3. Realización de las pruebas de software: Definido el proceso de pruebas sigue su puesta en marcha. En este nivel; se construyen y ejecutan las pruebas de software; y se reportan errores. Es un nivel operativo.

Figura 8. Niveles del modelo de pruebas de software para grupos de investigación.



3.3 ACTORES DEL MODELO

El modelo define tres roles fundamentales: director general de pruebas, administrador de pruebas y tester. Cado uno se esto roles juega un papel específico en el modelo. En la tabla 5 se describen las funciones principales asociados a cada rol.

Tabla 5. Actores del modelo

Rol del modelo	Funciones
Director general de las pruebas	Se encarga de administrar todo el proceso de pruebas de software en el grupo de investigación. Es quien define el proceso general de pruebas de software. Adicionalmente, realiza seguimiento periódico al proceso de pruebas dentro de los desarrollos software del grupo.
Administrador de pruebas	Es el encargado de definir el plan de pruebas para un desarrollo software en específico. Adicionalmente, realiza seguimiento a las actividades de pruebas de dicho desarrollo.
Tester	Es el encargado de seguir el proceso definido, reportar los resultados de la ejecución de las pruebas y los incidentes generados.

A continuación, se realiza la asociación de los roles en el grupo de investigación con los roles establecidos en el modelo:

Tabla 6. Relación entre los roles del modelo y los roles en el grupo de investigación

Rol del modelo	Rol candidato del grupo de investigación
Director general de las pruebas	Director del grupo de investigación o una persona designada por éste
Administrador de pruebas	Director o codirector del proyecto
Tester	Estudiante desarrollando el software

3.4 ACTIVIDADES DEL MODELO

El modelo define para cada uno de los niveles una seria de actividades y los roles relacionados a las mismas. Es de notar que algunas actividades pueden requerir del desarrollo de sub-actividades.

3.4.1 Actividades del nivel de lineamientos del grupo de investigación para las pruebas de software. Las actividades de este nivel son realizadas por el director general de las pruebas. Este nivel es el encargado de definir los lineamientos generales que guiarán el proceso de pruebas de software en el grupo de investigación y comprende las siguientes actividades:

Tabla 7. Actividades del nivel de lineamientos del grupo de investigación para las pruebas de software.

Actividad	Descripción
N1.1 Gestionar lineamientos	Se define la política y estrategia de pruebas que serán tenidas en cuenta durante el proceso de pruebas del grupo de investigación.
N1.2 Realizar seguimiento de cumplimiento de los lineamientos generales	Se revisa que las pruebas de software se estén realizando de acuerdo a lo planteado en los lineamientos

La actividad “N1.1 Gestionar lineamientos” se subdivide en tres sub-actividades. Ver tabla 10.

Tabla 8. Sub-actividades para gestionar lineamientos.

N1.1 Gestionar lineamientos	
Actividad	**Descripción**
N1.1.1 Formular política de pruebas	Esta actividad se encarga de definir la política de pruebas del grupo de investigación.
N1.1.2 Formular estrategia de pruebas	Esta actividad se encarga de definir la estrategia de pruebas del grupo de investigación.
N1.1.3 Ajustar lineamientos	Esta actividad se encarga de ajustar la política y estrategia de pruebas del grupo de investigación.

3.4.1.1 Formular política de prueba: La política de pruebas define los objetivos y principios de las pruebas de software que serán aplicados dentro del grupo de investigación. La política define que se debe lograr con las pruebas de software, sin embargo, no brinda detalles de cómo se deben llevar a cabo.

Figura 9. Actividades para formular la política de pruebas.

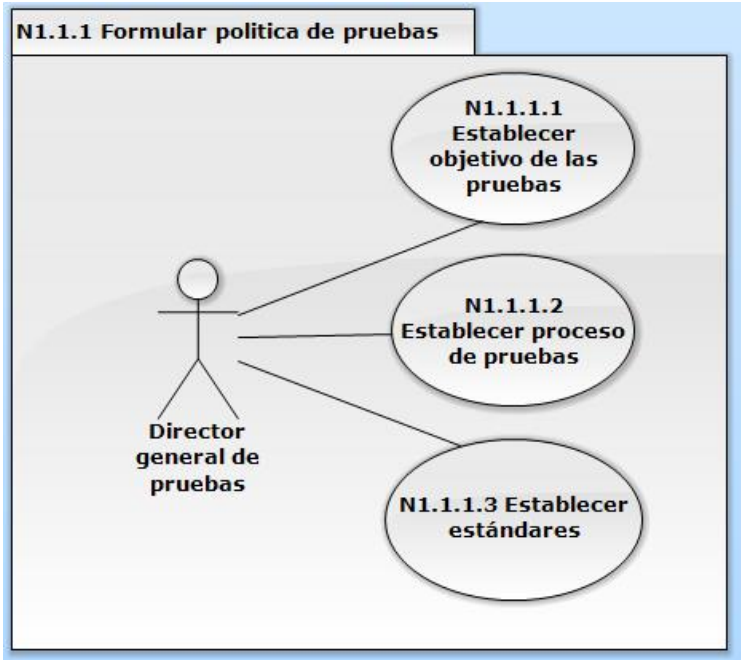


Tabla 9. Actividades para formular política de pruebas

Formular políticas de pruebas	
Actividad	**Descripción**
A1.1.1.1 Establecer objetivo de las pruebas	Definir el propósito y el alcance de las pruebas en el grupo de investigación.
A1.1.1.2 Establecer proceso de pruebas	Identificar el proceso de pruebas que el grupo seguirá, lo cual puede implicar referencias a documentos con información más detallada.
A1.1.1.3 Establecer estándares	Define los estándares de calidad que deben ser tenidos en cuenta por los desarrollos del grupo de investigación.

3.4.1.2 Formular estrategias de pruebas: La estrategia de pruebas proporciona los lineamientos de cómo llevar a cabo las pruebas dentro del grupo de investigación y es independiente del desarrollo software en específico. En la figura 10 se muestran las actividades necesarias para su formulación.

Figura 10. Formular estrategia de pruebas.

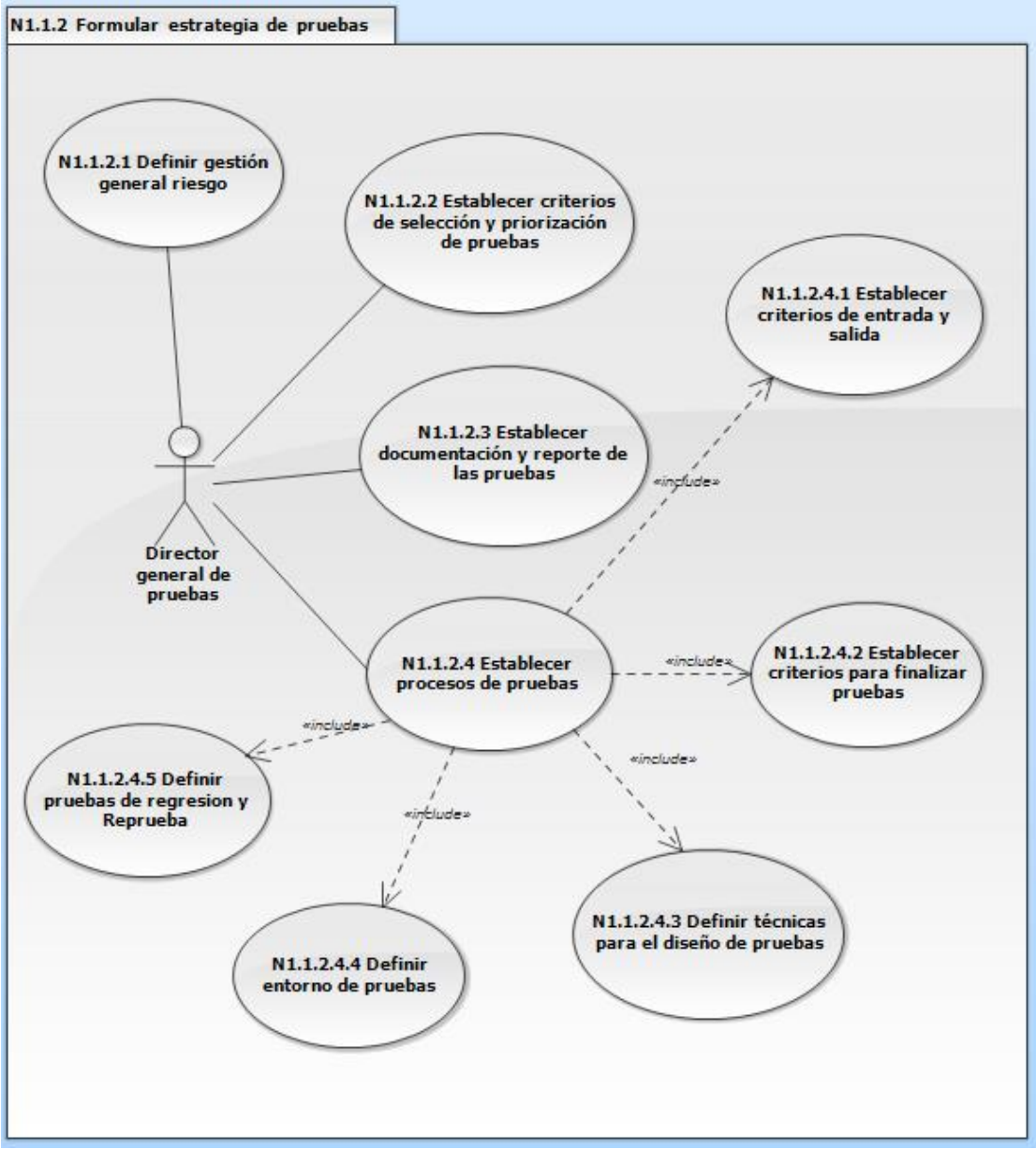


Tabla 10. Actividades para formular estrategia de pruebas

N1.1.2 Formular estrategias de pruebas	
Sub-actividades	**Descripción**
N1.1.2.1 Definir gestión general riesgo	Establece el enfoque general para el manejo del riesgo que se espera sea usado para orientar las actividades de pruebas
N1.1.2.2 Establecer criterios de selección y priorización de pruebas	Definir la forma como el grupo de investigación seleccionará y priorizará la ejecución de pruebas.
N1.1.2.3 Establecer documentación y reporte de las pruebas	Definir cuáles documentos deben ser generados por el proceso de pruebas
N1.1.2.4 Establecer proceso de pruebas	Definir cómo se llevará a cabo el proceso de pruebas.

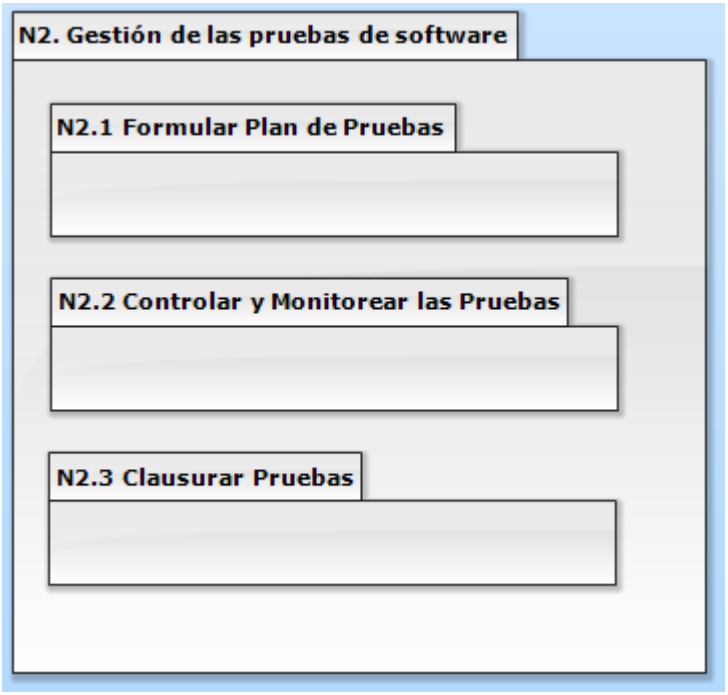
La actividad "N1.1.2.4 Establecer procesos de pruebas" por su complejidad e importancia es descrita con mayor detalle a continuación:

Tabla 11. Sub-actividades para establecer proceso de pruebas.

N1.1.2.4 Establecer procesos de pruebas	
Sub-actividades	**Descripción**
N1.1.2.4.1 Establecer criterios de entrada y salida	El proceso de pruebas se conforma de las actividades: diseño e implementación de pruebas, puesta en marcha y mantenimiento del entorno de pruebas, ejecución de pruebas, y reporte de errores. Diferentes criterios de entrada y salida pueden ser definidos para cada uno de forma individual, un grupo de ellos o para todo el proceso.
N1.1.2.4.2 Establecer criterios para finalizar pruebas	Define cuándo el grupo de investigación considera que las actividades de pruebas han sido completadas.
N1.1.2.4.3 Definir técnicas para el diseño de pruebas	Especifica las técnicas de diseño de pruebas que serán usadas en el diseño de pruebas.
N1.1.2.4.4 Definir entorno de pruebas	Especifica el entorno para el proceso de pruebas.
N1.1.2.4.5 Definir pruebas de regresión y reprueba	Especifica la estrategia, condiciones y actividades para realzar pruebas de regresión o probar nuevamente algo.

3.4.2 Actividades del nivel de gestión de las pruebas de software. Este nivel es el encargado de guiar las pruebas de software dentro de un desarrollo específico. Este nivel recibe como insumos los lineamientos generales definidos por el grupo, los ajusta al desarrollo en cuestión y realiza el seguimiento de las actividades de pruebas en el desarrollo particular. Las acciones de este nivel son realizadas por el administrador de pruebas y el tester. En la figura 11 se muestran las actividades necesarias para este nivel.

Figura 11. Actividades para la gestión de pruebas.



3.4.2.1 Formular plan de pruebas: Cada proyecto de software debe contar por lo menos con un plan de pruebas. El plan de pruebas es realizado por el administrador de pruebas y el Tester teniendo en cuenta los lineamientos definidos en la política y estrategia de pruebas de software. Además, debe contar con la

aprobación del director general de pruebas. Para la formulación del plan se plantean las siguientes actividades.

Figura 12. Actividades para formular el plan de pruebas.

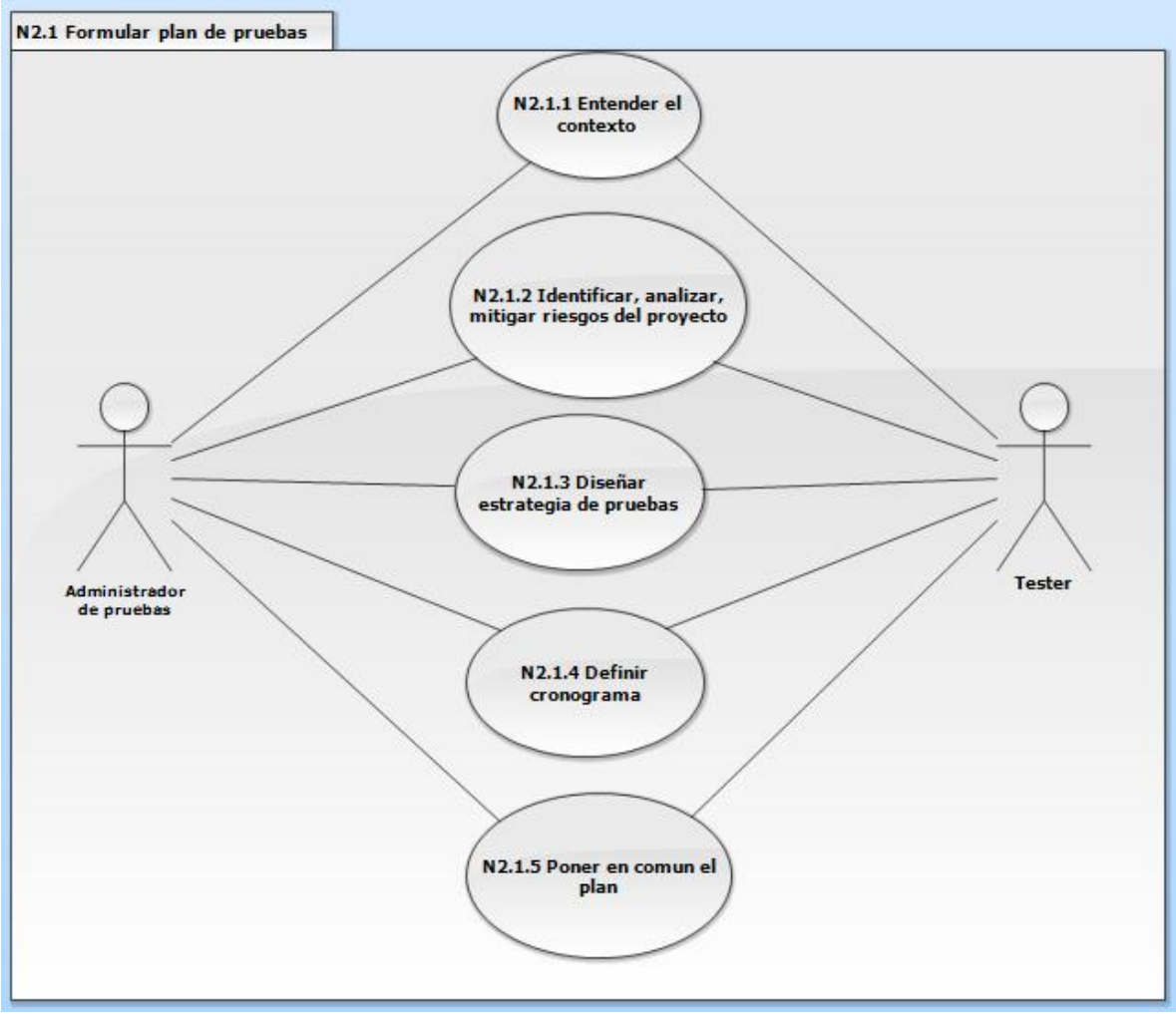


Tabla 12. Actividades para formular plan de pruebas

N2.1Formular plan de pruebas	
Actividad	**Descripción**
N2.1.1 Entender el contexto de las pruebas	Se revisan los lineamientos establecidos a nivel de grupo tales como: la política y estrategia de pruebas; y las características propias del desarrollo software en cuestión.
N2.1.2 Identificar, analizar y mitigar riesgos del proyecto	Se analizan los riesgos, ya detectados, en el desarrollo software y se estudia como las pruebas pueden tratarlos. A cada riesgo se le debe asignar un impacto y una probabilidad. Así como algún medio para mitigarlo.
N2.1.3 Diseñar estrategia de pruebas	Se parte de la información definida en la estrategia de pruebas y se ajusta de acuerdo a los niveles de pruebas, tipos de pruebas, características a ser probadas, técnicas de diseño de pruebas, fundamentos de las pruebas, y restricciones organizacionales, propias del desarrollo software.
N2.1.4 Definir cronograma	A cada actividad de pruebas definida en la estrategia se le debe asignar una fecha
N2.1.5 Poner en común el plan	Dar a conocer el plan de pruebas a todos los interesados.

3.4.2.2 Controlar y monitorear las pruebas: Con el propósito de velar por el correcto seguimiento de lo establecido en el plan de pruebas y las especificaciones de pruebas del grupo de investigación, el administrador de pruebas del desarrollo debe realizar las siguientes actividades:

Figura 13. Controlar y Monitorear las pruebas.

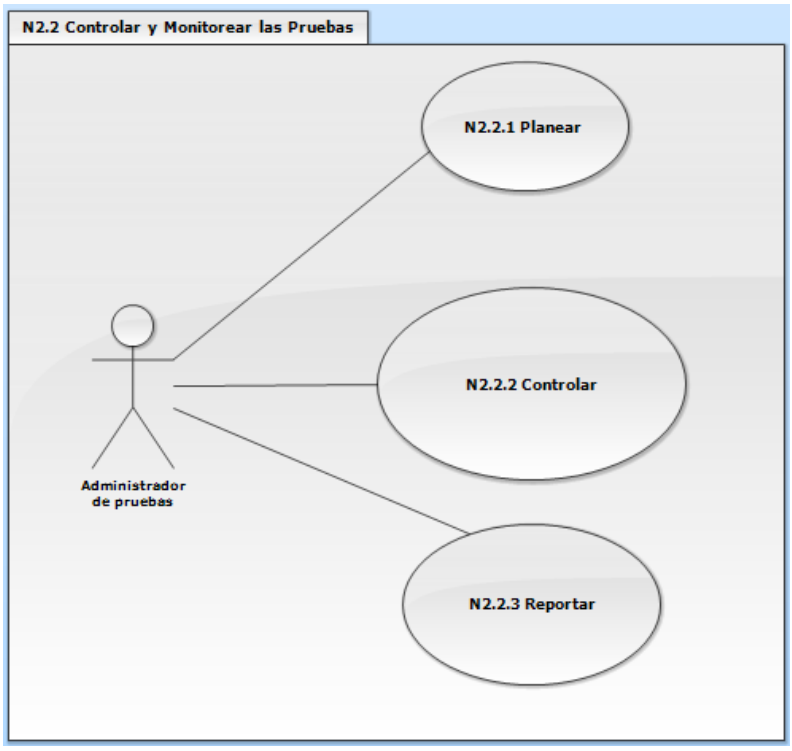


Tabla 13. Actividades para controlar y monitorear las pruebas

N2.2 Controlar y monitorear las pruebas	
Actividad	**Descripción**
N2.2.1 Planear	Se deben seleccionar las métricas que serán usadas para contrastar el proceso de pruebas y el plan de pruebas. Además, se deben establecer puntos de chequeo dentro del cronograma de actividades para el reporte de métricas y el estado del proceso de pruebas
N2.2.2 Controlar	En los puntos de chequeo se debe contrastar el avance del proceso de pruebas, a través de las métricas recolectadas, con lo establecido en el plan de pruebas (por ejemplo, cronograma). En caso de discrepancia se procede a tomar acciones correctivas tales como modificar el plan de pruebas a través de un análisis de riesgos y prioridades de los mimos.
N2.2.3 Reportar	Una vez culminada la actividad de control es necesario documentar los resultados obtenidos.

- **Clausurar pruebas**

Se busca documentar las lecciones aprendidas durante el proceso de pruebas y restaurar el entorno de pruebas en caso de haber modificado alguna configuración. Esta actividad debe ser realizada entre el administrador de pruebas y el Tester. En la figura 14 se muestran las actividades necesarias.

Figura 14. Actividades para clausurar las pruebas

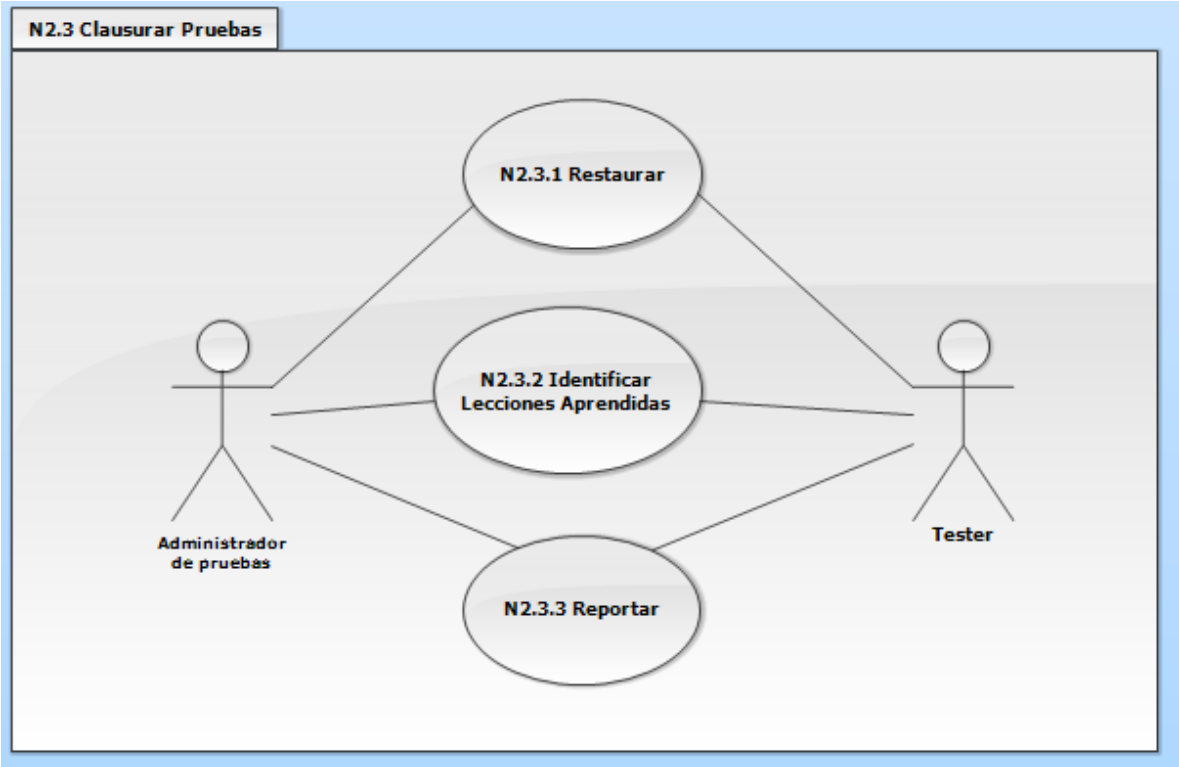
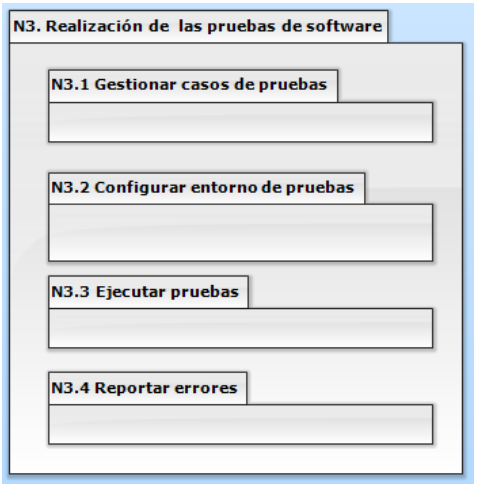


Tabla 14. Actividades para controlar y monitorear las pruebas

N2.3 Controlar y monitorear las pruebas	
Actividad	**Descripción**
N2.3.1 Restaurar	Cualquier cambio de software o hardware resultante de las actividades de pruebas debe ser revertido (por ejemplo, el entorno de pruebas).
N2.3.2 Identificar lecciones aprendidas	Una vez finalizadas las actividades de pruebas, es necesario analizar el camino recorrido, problemas encontrados y resultados obtenidos. Con el propósito de contribuir a un mejor desarrollo de los trabajos futuros. Esta actividad es particularmente útil si se tiene en cuenta que al iniciar un desarrollo software las personas que van a probar pueden no conocer cómo se lleva a cabo el proceso de las pruebas.
N2.3.3 Reportar	Al final se debe generar un documento donde se condense la información hallada.

3.4.3 Actividades del nivel de realización de pruebas. Una vez definido el plan de pruebas se procede a su ejecución. La ejecución está a cargo del Tester, quien debe realizar las actividades mostradas en la figura 15.

Figura 15. Actividades del nivel de realización de pruebas.



3.4.3.1 Gestionar casos de pruebas: Una vez definida las características del sistema que serán tenidas en cuenta en el proceso de pruebas, el Tester debe proceder a crear una serie de casos de pruebas para cada una de estas características. En la figura 16 se muestran las actividades necesarias.

Figura 16. Gestionar casos de pruebas.

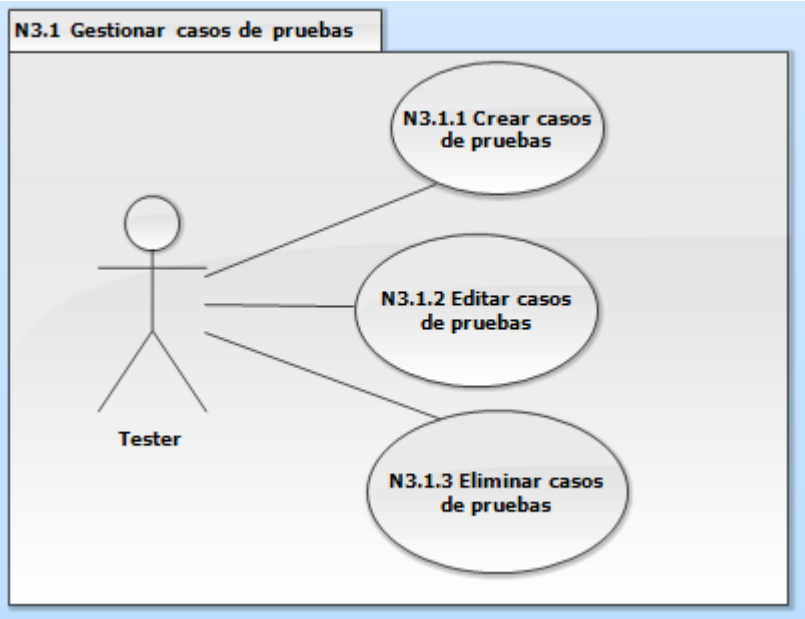


Tabla 15. Gestionar casos de pruebas.

N3.1 Gestionar casos de pruebas	
Actividad	**Descripción**
N3.1.1 Crear casos de pruebas	Implementar el caso de pruebas que revisara la característica asociado al mismo.
N3.1.2 Editar casos de pruebas	Modificar el caso de pruebas en caso de cambios en la característica asociado al mismo.
N3.1.3 Borrar casos de pruebas	Remover test case en caso de que ya no sea necesario.

3.4.3.2 Configurar entorno de pruebas: En el plan de pruebas se brindan lineamientos generales del entorno de pruebas. Esta actividad es responsabilidad del Tester, el cual debe realizar el montaje de dicho entorno de pruebas y tendrá que velar por su correcto funcionamiento. En la figura 17 se muestran las actividades necesarias.

Figura 17. Actividades para configurar entorno de pruebas

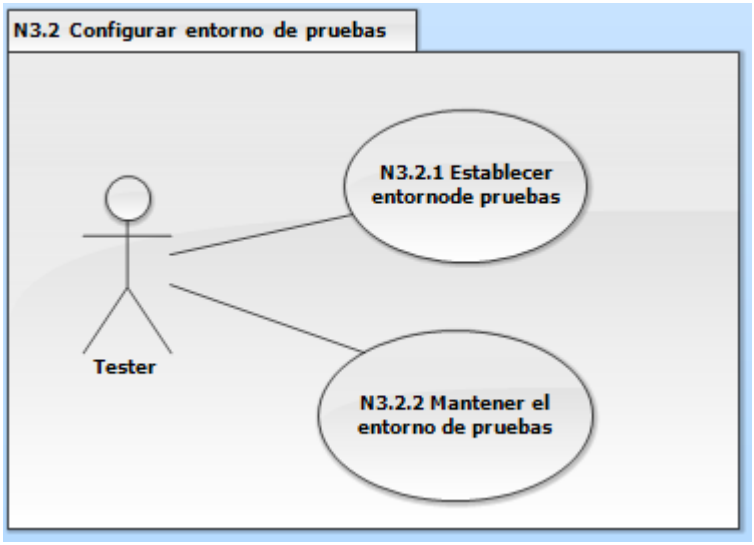


Tabla 16. Actividades para configurar entorno de pruebas

N3.2 Configurar entorno de pruebas	
Actividad	**Descripción**
N3.2.1 Establecer entorno de pruebas	Realizar el montaje del entorno de pruebas.
N3.2.2 Mantener el entorno de pruebas	Realizar los ajustes necesarios para que el entorno de pruebas esté disponible y funcionando.

3.4.3.3 Ejecutar pruebas: Una vez creados los casos de pruebas se procede a su ejecución. La ejecución de los casos de pruebas es realizada principalmente por el Tester. Sin embargo, con el propósito de incrementar las posibilidades de encontrar errores es recomendable incluir Testers independientes o externos al proyecto de software. En la figura 18 se muestran las actividades necesarias.

Figura 18. Actividades para ejecutar pruebas.

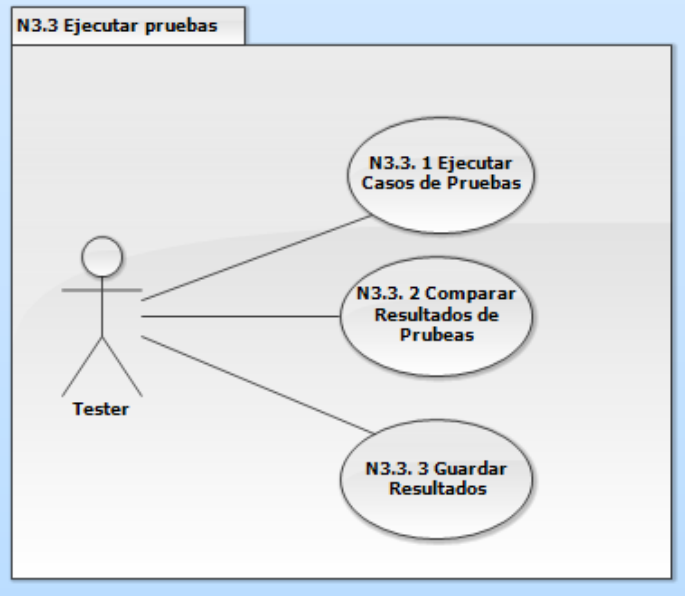


Tabla 17. Actividades para ejecutar pruebas.

N3.3 Ejecutar Pruebas	
Actividad	**Descripción**
N3.3.1 Ejecutar casos de pruebas	Los casos de pruebas creados deben ser ejecutados en el entorno de pruebas.
N3.3.2 Comparar resultados de pruebas	Una vez ejecutados los casos de pruebas se debe comparar los resultados obtenidos con los esperados.
N3.3.3 Guardar resultados	Los resultados obtenidos durante la ejecución deben ser guardados para posteriores análisis.

- **REPORTE DE INCIDENTES**

El propósito de esta actividad es generar una serie de informes que permitan conocer el estado del software. Reportes que están a cargo del Tester. En la figura 19 se muestran las actividades necesarias.

Figura 19. Reporte de errores

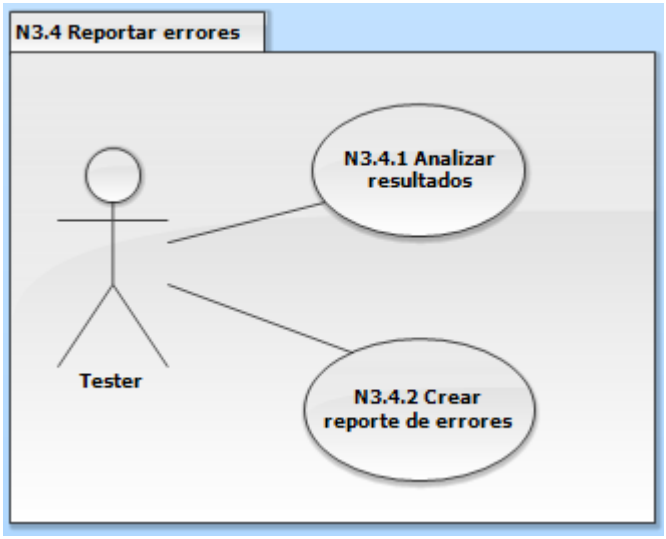


Tabla 18. Reporte de errores

N3.4 Reportar errores	
Actividad	**Descripción**
N3.4.1 Analizar resultados	Al comparar los resultados obtenidos con los esperados se procede a analizar si las discrepancias corresponden a errores.
N3.4.2 Crear reporte de errores	Una vez se ha establecido que las discrepancias entre lo encontrado y lo esperado corresponden a errores. Se procede a generar los reportes correspondientes.

4. INTERVENCIÓN EN PROYECTOS DE GRADO

El modelo propuesto en este trabajo se fue construyendo de forma iterativa e incremental. Por lo anterior, conforme se fue avanzando en el modelo, se dio la intervención en el grupo STI. La intervención en un proyecto iniciaba con una reunión donde los estudiantes explicaban en qué consistían sus proyectos y en qué estado se encontraba. A continuación, se hacía una revisión sobre lo que se tenía. Producto de esta revisión se realizaba una serie de observaciones y se orientaba frente al tema de calidad desde las pruebas de software. Finalmente, se acordaban una serie de acciones encaminadas a guiar el proceso de pruebas y una próxima reunión era programada. Como resultado de la intervención se formulaban una serie de casos de pruebas, que tenían en cuenta los lineamientos del grupo, y eran ejecutados por el estudiante. Con los que fue posible identificar y corregir errores de forma oportuna. Mejorando de esta forma la calidad del software generado. Para facilitar la lectura de cada intervención se proporcionará la siguiente información:

1- Nombre del proyecto
2- Descripción del proyecto intervenido
3- Gestión de la calidad en el desarrollo software
4- Recomendaciones para mejorar la calidad del desarrollo software
5- Resultados Obtenidos.

4.1 INTERVENCIÓN 1

4.1.1 Nombre del proyecto. Adaptación y desarrollo de módulos de la herramienta software HSLAB en Cloud Computing, para el mejoramiento al control de procesos de análisis de aguas residenciales, naturales y subterráneas del laboratorio CEIAM

4.1.2 Descripción del proyecto intervenido. Este proyecto extendió las funcionalidades de la aplicación Web HSLAB para reemplazar todos los formatos de papel en el proceso del servicio de ensayo del laboratorio CEIAM y de esta forma ayudar a contribuir al proceso de toma de decisiones de investigadores y directivos[48]. Al momento de realizar la intervención el proyecto se encontraba a punto de ser terminado.

4.1.3 Gestión de la calidad en el desarrollo software. Al empezar a interactuar con el autor de este proyecto fue claro que no se había preocupado por la calidad de su desarrollo. Al usar la aplicación rápidamente se encontraron dificultades en la navegación en el sitio pues al ir de página en página no se validaba de forma adecuada el volver a la página anterior. En reuniones siguientes, el autor de este trabajo fue realizando los ajustes con lo que su desarrollo software fue mejorando de forma notoria.

[48] Reyes Giraldo, L. F., & Gomez Florez, L. C. Adaptación y desarrollo de modulos de la herramienta software hslab en cloud computing, para el mejoramiento al control de procesos de análisis de aguas residuales, naturales y subterraneas del laboratorio CEIAM. 2014. Disponible en: http://doi.org/10.1017/CBO9781107415324.004

4.1.4 Recomendaciones para mejorar la calidad del desarrollo software. Al encontrar los problemas de navegación, se realizaron recomendaciones al autor de este trabajo y se formularon casos de pruebas donde se verificará la correcta navegación para diferentes usuarios del sistema.

4.1.5 Resultados obtenidos. Cuando se inició la intervención, este proyecto no contaba con la definición de casos de pruebas. Lo cual se evidenció en los problemas encontrados en las primeras reuniones. El autor de este proyecto corrigió los errores encontrados y por iniciativa propia formuló varios casos de pruebas y empezó a usar su desarrollo software tal como un usuario real lo haría. Con lo cual encontró varios problemas que se pudieron corregir antes de mostrar su trabajo al usuario final. Al finalizar el proyecto se formularon un total de 29 casos de pruebas los cuales fueron ejecutados de forma exitosa sin encontrar errores con lo que se mostró un buen nivel de calidad.

4.2 INTERVENCIÓN 2

4.2.1 Nombre del proyecto. Creación de un portal web para facilitar el acceso a los recursos y a la información del consultorio jurídico de la Universidad Industrial de Santander.

4.2.2 Descripción del proyecto intervenido. El consultorio jurídico no contaba con un medio que permitiera al beneficiario conocer el estado de su proceso.

Este trabajo desarrolló un portal Web que hizo uso de la información registrada a través del sistema de información del consultorio jurídico (CYSACJ 2.0) desde la Web[49]. Al momento de realizar la intervención el proyecto se encontraba en pleno desarrollo.

4.2.3 Gestión de la calidad en el desarrollo software. Los autores de este proyecto no poseían un conocimiento formal en el tema de la calidad. Sin embargo, desde que iniciaron su proyecto se enfocaron en que todo funcionara de la mejor forma posible.

En la primera revisión se encontró que las funcionalidades de algunos roles del sistema no se encontraban disponibles pues al acceder a las paginas correspondientes se generaban errores. En las posteriores revisiones se encontraron problemas en la funcionalidad de búsqueda.

4.2.4 Recomendaciones para mejorar la calidad del desarrollo software. Ante los problemas encontrados en la primera sesión se recomendó formular casos de pruebas que garantizaran navegar por todas las vistas del sistema. Sacando provecho de los casos de pruebas se recomendó realizar pruebas que contemplaran todos los roles del sistema, así como las funcionalidades más relevantes del mismo.

[49] Moreno tarazona, m. A., vergel yaruro, p. L., & gómez flórez, l. C. Creación de un portal web para facilitar el acceso a los recursos y a la información del consultorio jurídico de la Universidad Industrial de Santander. 2014. Disponible en: http://doi.org/10.1017/CBO9781107415324.004

4.2.5 Resultados obtenidos. Al empezar a formular casos de pruebas y documentar la ejecución de los mismo se encontraron varios errores. En la primera ejecución de los casos de pruebas se encontró que 4 de 10 casos de prueba fallaron. Sin embargo, conforme se avanzaba la cantidad de problemas se fueron reduciendo. Al tener casos de pruebas que hicieran uso de todas las vistas permitió que los autores de este trabajo hicieran pruebas para casi todo el sistema. Al finalizar el proyecto se ejecutaron 31 casos de pruebas de forma exitosa. Lo cual generó un producto final más confiable y de mejor calidad.

4.3 INTERVENCIÓN 3

4.3.1 Nombre del proyecto. Desarrollo del sistema de información para el programa de selecciones deportivas adscrito a la dirección de bienestar Universitario de la Universidad Industrial de Santander

4.3.2 Descripción del proyecto intervenido. En este trabajo se efectuó un estudio de necesidades para la realización de actividades biomédicas, dirección y consulta por parte de los deportistas. Para lograr esto se desarrolló una herramienta web para apoyar el desarrollo deportivo[50]. Al momento de realizar la intervención, este proyecto se encontraba en pleno desarrollo.

[50] Plata Robles, C. C., Sepùlveda Sarmiento, a. A., & Gómez Flórez, l. C. Desarrollo del sistema de información para el programa de selecciones deportivas adscrito a la dirección de bienestar universitario de la Universidad Industrial de Santander. 2015. Disopnible en: http://doi.org/10.1017/CBO9781107415324.004

4.3.3 Gestión de la calidad en el desarrollo software. En el primer encuentro con los autores de este proyecto se identificaron muchas dificultades al usar el sistema como administrador. Es de notar que en revisiones posteriores la cantidad de errores se redujo de forma notoria.

4.3.4 Recomendaciones para mejorar la calidad del desarrollo software. Teniendo en cuenta que este proyecto efectuó un estudio de necesidades, los requerimientos se encontraban documentados de forma clara. Por lo cual se propuso crear la cantidad de casos de pruebas necesarios para revisar los requerimientos en su totalidad.

4.3.5 Resultados obtenidos. Después de introducir los casos de pruebas a los estudiantes. En la primera ejecución de los mismos se encontró que 5 de 11 casos de pruebas fallaron. En posteriores ejecuciones la cantidad de errores se fue reduciendo. Al final, se ejecutaron en total 23 casos de pruebas de forma exitosa. Por lo cual el prototipo final mostro un buen nivel de calidad.

4.4 INTERVENCIÓN 4

4.4.1 Nombre del proyecto. Sistema de información para el apoyo de las actividades del centro de conciliación adscrito al consultorio jurídico de la UIS: SICJ IES

4.4.2 Descripción del proyecto intervenido. Este trabajo extendió las funcionalidades del desarrollo software CYSACJ-UIS a los procesos de casos de conciliación. Es de notar que CYSACJ-UIS es un software insignia del grupo STI[51]. Al momento de realizar la intervención el proyecto se encontraba iniciando.

4.4.3 Gestión de la calidad en el desarrollo software. Teniendo en cuenta que este proyecto se acompañó desde el principio, se planteó una revisión CYSACJ-UIS antes de realizar las nuevas funcionalidades. Revisión que encontró varias dificultades que fueron atendidas con el autor de la versión anterior de CYSACJ-UIS. Una vez hecho esto, se propuso formular un caso de pruebas por cada necesidad.

4.4.4 Recomendaciones para mejorar la calidad del desarrollo software. Al extender las funcionalidades de un software existente del que no se conoce su calidad resulta necesario realizar una revisión del mismo. Esta revisión, en lo posible debe ser guiada por la documentación del desarrollo software. Hay que tener en cuenta que no solo se debe revisar cada funcionalidad de forma individual también es necesario revisar la forma como interactúan cada uno de los requerimientos por parte del usuario final.

4.4.5 Resultados obtenidos. En este proyecto se introdujo el tema de las pruebas de forma temprana al desarrollo software. Por lo cual fue posible estructurar de una mejor forma el proceso de las pruebas.

[51] Ardila Corredor, H. L., Flórez Gómez, L. C., & Villarreal Díaz, D. A. Sistema de información para el apoyo de las actividades del centro de conciliación adscrito al consultorio jurídico de la UIS: SICJ IES. 2015. Disponible en: http://doi.org/10.1017/CBO9781107415324.004

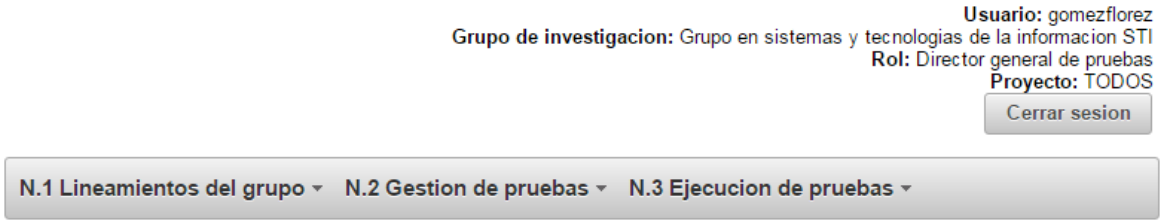
El cual inicio con una inducción al objetivo del grupo STI frente al tema de las pruebas. Se procedió a formular algunos de casos de pruebas para cubrir todos los requerimientos de este desarrollo software, en total se generaron 15 casos de pruebas, los cuales fueron ejecutados, por el autor de ese trabajo, con forme se fue avanzando en el desarrollo del proyecto. Razón por la cual, la cantidad de errores encontrados fue baja. Una vez, fueron implementados todos los requisitos y los casos de pruebas fueron ejecutados de forma exitosa. Se procedió a formular una serie de escenarios que implicaban la agrupación y ejecución de varios casos de pruebas, esto con el propósito de simular la interacción real del usuario final con el sistema. Al final, se formularon 5 escenarios que fueron ejecutado de forma exitosa. El formular caso de pruebas ayudó al desarrollador a analizar situaciones adversas que se pueden presentar cuando el usuario hace uso de su desarrollo y de esta forma realizar acciones correctivas de forma oportuna. Desde el punto de vista del director del proyecto, el revisar los escenarios contemplados y las correcciones realizadas por el desarrollador incrementa la confianza en el desarrollo realizado y le permite concluir que se encuentra en un mejor nivel de calidad.

5. PROTOTIPO SOFTWARE

Con el propósito de facilitar la puesta en común del modelo para gestionar procesos de pruebas en grupos de investigación. Se plantea el diseño de un prototipo software, el cual tuvo el nombre de TestRG (Testing in Research Group). En el Anexo D se encuentra detalles técnicos del prototipo.

El prototipo se compone de tres módulos, cada módulo se relaciona, de forma directa, con un nivel del modelo propuesto. El menú se compone de las opciones: N.1 Lineamientos del grupo, N.2 Gestión de pruebas, N.3 Ejecución de pruebas. Ver tabla 20.

Figura 20. Menú principal de TestRG



En el menú *N.1 Lineamientos del grupo*, apoya el desarrollo de las actividades del nivel general de pruebas. Este menú cuenta con las sub-opciones de *policía* y *estrategia*., de pruebas. Las cuales como lo sugiere el modelo son gestionadas por usuarios del rol *Director general de pruebas* y puede ser consultado por los usuarios de otros roles del sistema. En la figura 21 y figura 22 se encuentra la política y estrategia de pruebas definida para el grupo STI.

En el menú *N.2 Gestión de pruebas* se encuentra la opción *Plan de pruebas*, apoya el desarrollo de las actividades del nivel de gestión de pruebas. Este menú cuenta con las sub-opción *plan de pruebas*. Como lo sugiere el modelo esta opción es gestionada por usuarios del rol *Director general de pruebas* y *Administrador de pruebas* y puede ser consultado por los usuarios de otros roles del sistema. En la figura 23 se muestra el plan de pruebas de un proyecto del grupo STI

Figura 21. Política de pruebas del grupo STI

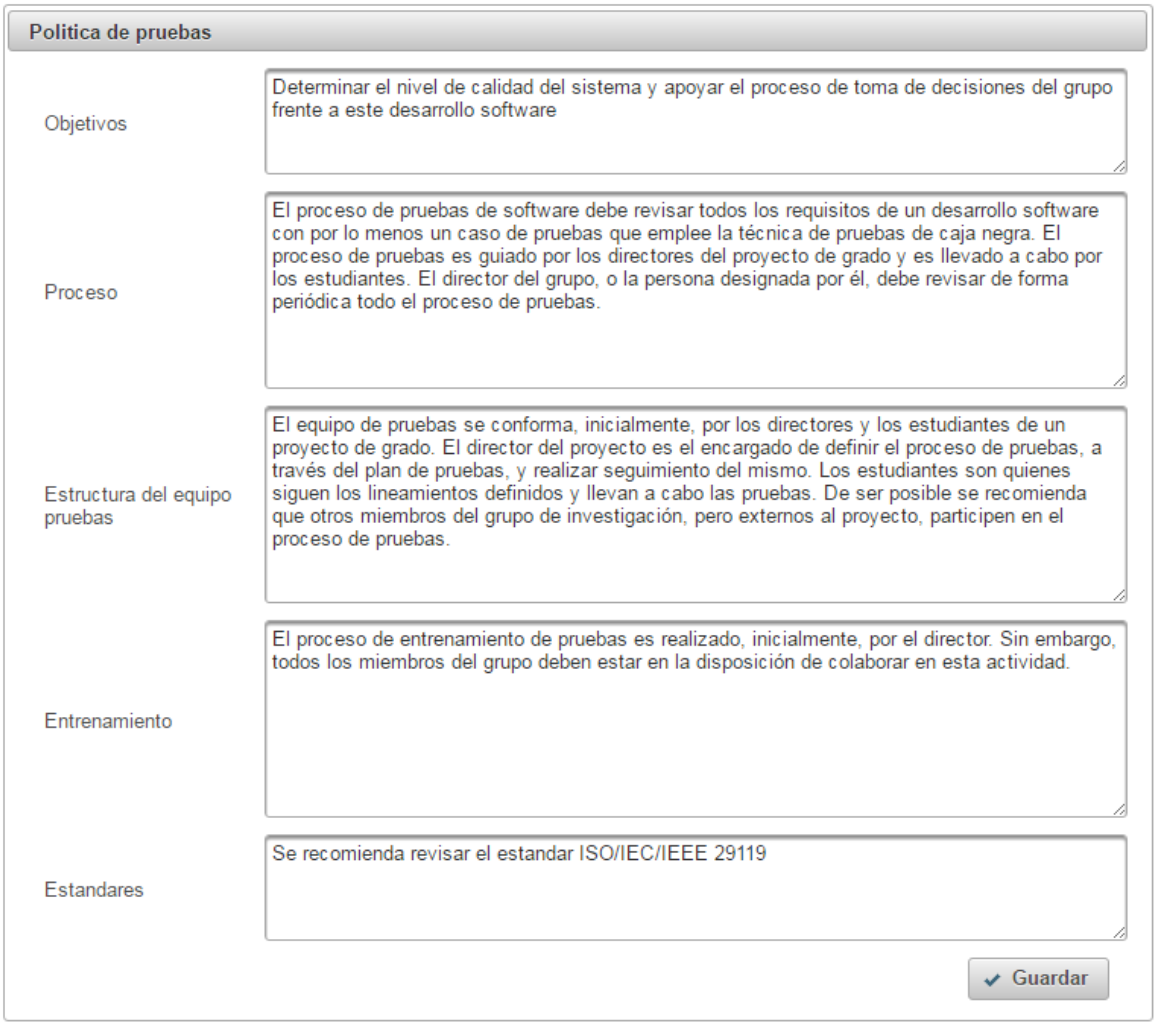


Figura 22. Estrategia de pruebas del grupo STI

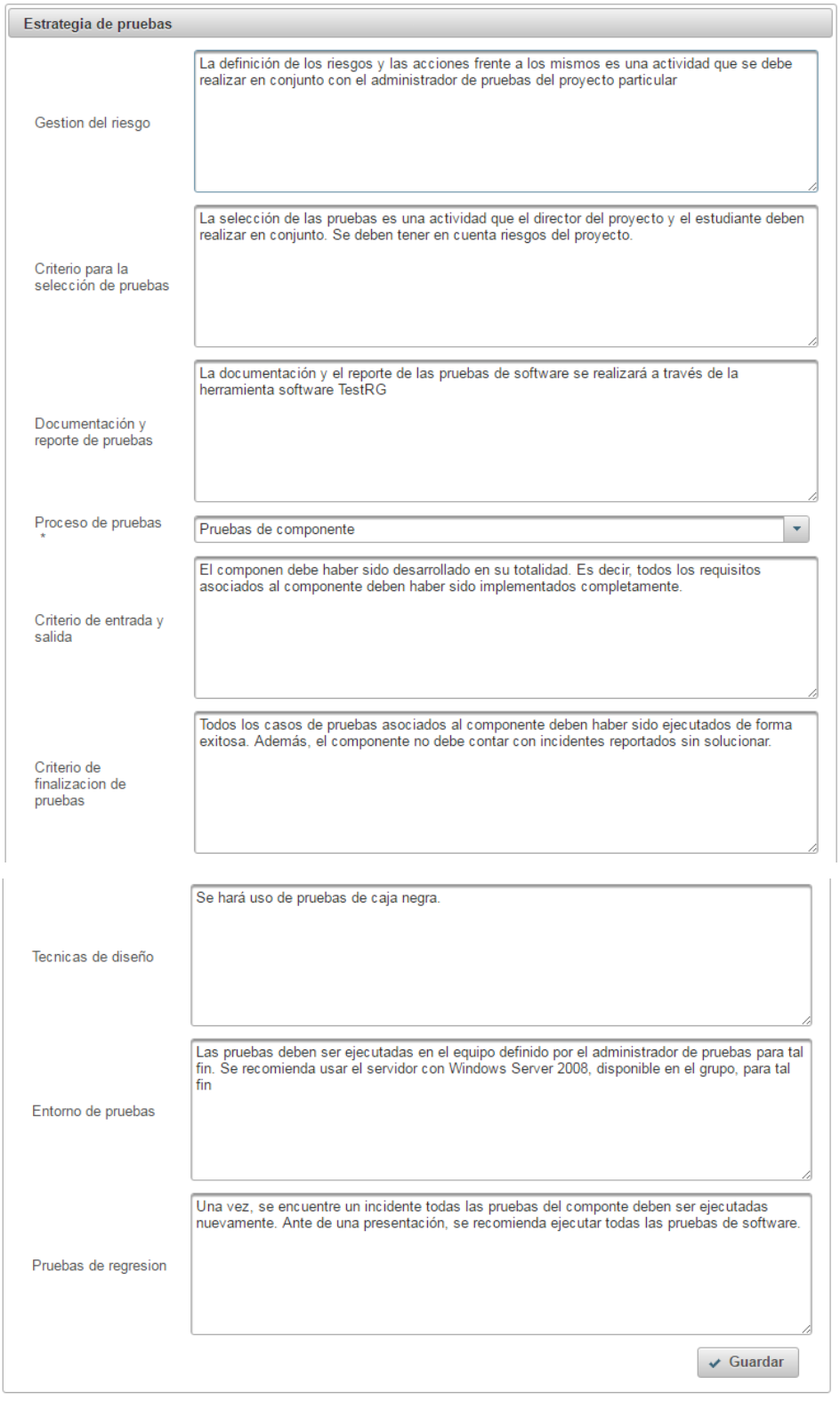
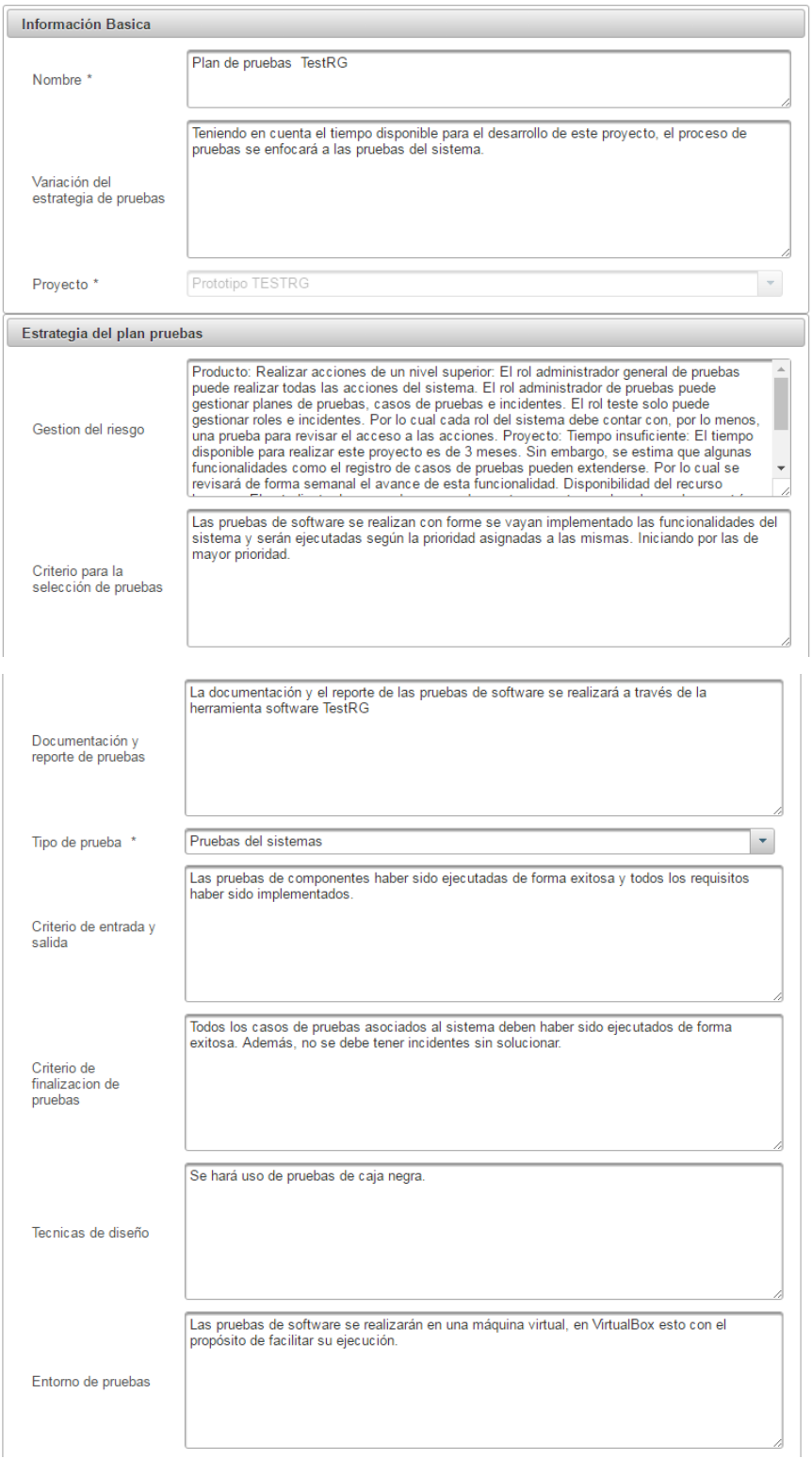


Figura 23. Plan de pruebas del grupo STI



En el menú *N.3 Ejecución de pruebas*, apoya el desarrollo de las actividades del nivel realización de pruebas. Este menú cuenta con las sub-opciones de *caso de pruebas* e *incidentes*. Las cuales como lo sugiere el modelo son gestionadas de forma directa por los usuarios del rol *Tester*. En la figura 24 y figura 25 se muestra el listado de casos de pruebas e incidentes definida para un proyecto del grupo STI.

Figura 24. Captura de pantalla de la ventana donde se listan los casos de pruebas

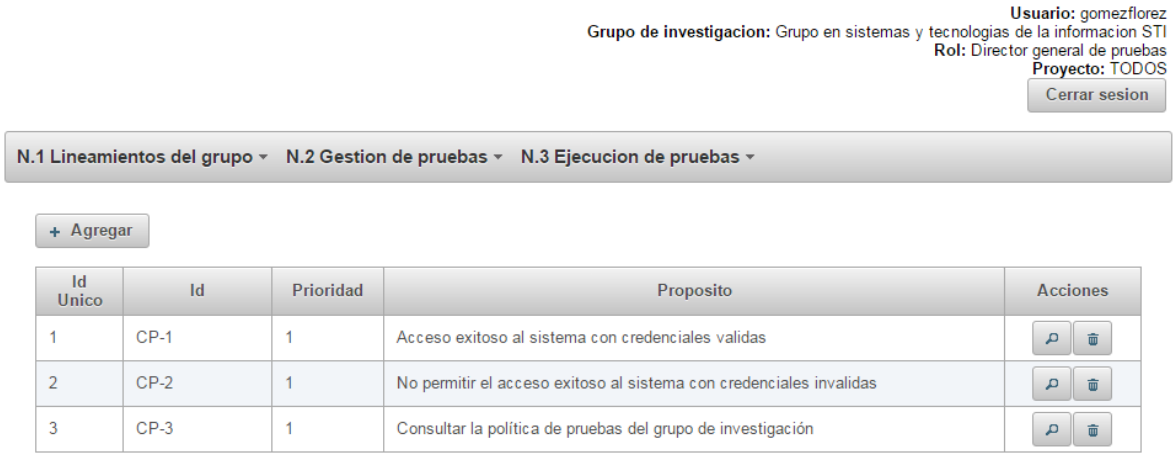
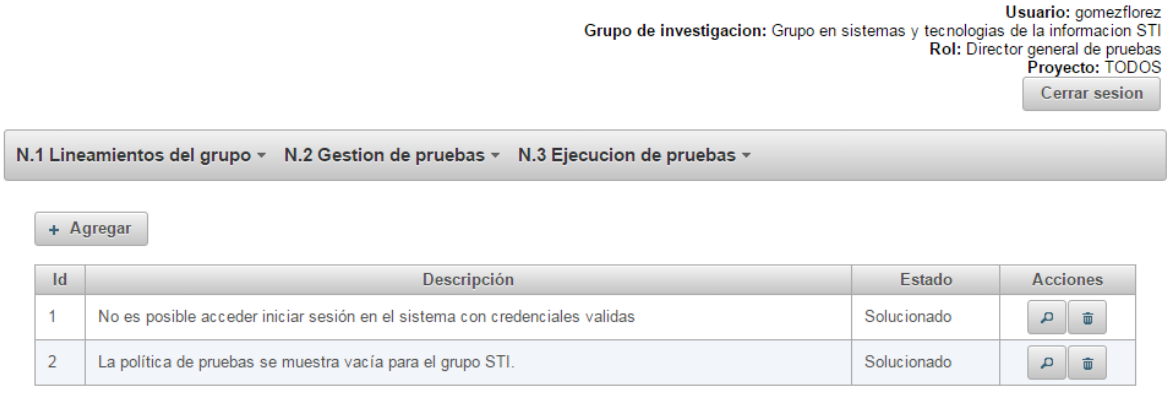


Figura 25. Captura de pantalla de la ventana donde se listan los incidentes



6. RESULTADOS

Los resultados y logros obtenidos en el desarrollo de esta investigación fueron:

- Una revisión de los proyectos de grado de los grupos STI, SIMON y FILOEN. Revisión que fue presentada en el tercer congreso Ecuatoriano de Tecnologías de la información y la Comunicación.
- El proyecto se presentó y fue aceptado en la convocatoria de Colciencias de jóvenes investigadores de 2013.
- Un prototipo software llamada TestRG (Testing in Research Group).
- Surgió el proyecto “Integrar las normas ISO/IEC 25010 y 25040 al sistema para la evaluación de calidad de software derivado de actividades de investigación” que se presentó y fue aceptado en la convocatoria de Colciencias de jóvenes investigadores de 2015.

7. CONCLUSIONES Y TRABAJOS FUTUROS

La información recolectada de los proyectos de final de carrera permite afirmar que la documentación del desarrollo software no se realiza de forma adecuada, lo cual no es algo nuevo en el ámbito del software. Sin embargo, si se tiene en cuenta que los trabajos de final de carrera son la oportunidad de los estudiantes para aplicar lo aprendido durante su proceso de formación resulta claro que por su inexperiencia los estudiantes de Ingeniería de Sistemas no prestan la atención adecuada a la documentación del software. Por esta razón, se debe incrementar el acompañamiento de los directores en los trabajos de grado en esta temática.

De la intervención realizada en el grupo STI, se encontró que la mayoría de los problemas detectados pudieron haber sido evitados si los estudiantes hubieran recibido algún tipo de orientación practica en temas de calidad, en particular en el tema de las pruebas de software. Introducir conceptos como casos de pruebas, pruebas unitarias, pruebas de aceptación, etc., hace que los estudiantes vean sus desarrollos de una forma diferente y contemplen escenarios en los que su trabajo podría no cumplir con lo especificado.

En la revisión literaria se encontró que la automatización de las pruebas de software puede ser una herramienta valiosa para que las PyME mejoren la calidad de sus desarrollos. Esto como resultado de los bajos costo, a largo plazo, que implica su adopción y posterior mantenimiento. Sin embargo, este tipo de pruebas debe ser usado con cautela en los grupos de investigación pues, debido a su naturaleza automática, pueden no permitir el desarrollo completo de competencias en el tema de calidad por parte de sus integrantes.

El modelo propuesto en este trabajo se fundamentó en el estándar ISO/IEC/IEEE 29119. Sin embargo, por su inexperiencia, el estándar ISO/IEC/IEEE 29119 puede ser difícil de asimilar por algunos integrantes de los grupos de investigación en especial de los estudiantes. Por esta razón, el modelo propuesto buscó simplificar las actividades y documentación necesaria para su incorporación. Resaltando, los procesos y elementos claves del estándar ISO/IEC/IEEE 29119. Entre las ventajas del modelo propuesto se encuentra que facilita la comunicación entre los integrantes del grupo de investigación de las pruebas de software pues define de forma clara el papel de cada uno en este proceso.

Resulta conveniente contar con un prototipo software que implemente el modelo pues la herramienta puede orientar a los integrantes del grupo de investigación en las actividades que debe llevar a cabo el proceso de las pruebas de software. Además, puede ser usado como una fuente de consulta de procesos de pruebas previos por parte de todos los integrantes del grupo de investigación.

Finalmente, como trabajo futuro se recomienda estudiar la optimización de la cobertura de los casos de pruebas propuestos por los integrantes de los grupos de investigación.

BIBLIOGRAFÍA

Ardila Corredor, H. L., Flórez Gómez, L. C., & Villarreal Díaz, D. A. Sistema de información para el apoyo de las actividades del centro de conciliación adscrito al consultorio jurídico de la UIS: SICJ IES. 2015. Disponible en: http://doi.org/10.1017/CBO9781107415324.004

Basri, S., & O'Connor, R. V. Understanding the Perception of Very Small Software Companies towards the Adoption of Process Standards. 2010

Batista, J., & Diaz de Figueiredo, A. SPI in a Very Small Team: a Case with CMM. Science And Technology, 5(4), 243–250. 2000. Disponible en: http://doi.org/10.1002/1099-1670(200012)5:4<243::AID-SPIP126>3.0.CO;2-0

Biolchini, J., Mian, P. G., Candida, A., & Natali, C. Systematic Review in Software Engineering. 2005

Bourque, P., & Fairley, R. E. Guide to the Software Engineering - Body of Knowledge. IEEE Computer Society. 2014. Disponible en: http://doi.org/10.1234/12345678

Colciencias. Modelo De Medición De Grupos De Investigación, Desarrollo Tecnológico O De Innovación Y De Reconocimiento De Investigadores Del Sistema Nacional De Ciencia, Tecnología E Innovación. 2015. Disopnible en: http://www.colciencias.gov.co/sites/default/files/ckeditor_files/files/DOCUMENTO MEDICIÓN GRUPOS - INVESTIGADORES VERSIÓN FINAL 15 10 2014 (1).pdf

Gleirscher, M., Golubitskiy, D., Irlbeck, M., & Wagner, S. Introduction of static quality analysis in small- and medium-sized software enterprises: experiences from technology transfer. Software Quality Journal, 1–44. 2013. Disponible en: http://doi.org/10.1007/s11219-013-9217-z

Hapter, C. Software Testing. In SWEBOK. 2004

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucia, M. del P. Metodología de la investigación (5th ed.). 2010

IEEE Std 610.12-1990. IEEE Standard Glossary of Software Engineering Terminology (Vol. 121990). 1990

IEEE Std 829. Standard for Software and System Test Documentation (Vol. 2008). 2008

IEEE, & ACM. Software Engineering 2004 Degree Programs in Software Engineering. 2004

ISO/IEC/IEEE 24765. Systems and software engineering — Vocabulary (Vol. 2010). 2010

ISO/IEC/IEEE 29119-1. Software and systems engineering — Software testing — Part 1: Concepts and definitions. 2013

ISO/IEC/IEEE 29119-2. Software and systems engineering — Software testing — Part 2: Test processes. 2013

ISTQB. Foundation Level Syllabus. 2011

Kitchenham, B. Guidelines for performing Systematic Literature Reviews in Software Engineering, Version 2.3. 2007

Kitchenham, B. a, Mendes, E., & Travassos, G. H. Cross- vs. Within-Company Cost Estimation Studies: A Systematic Review. IEEE Transactions on Software Engineering, 33(5), 316–329. 2007. Disponible en: http://doi.org/http://dx.doi.org/10.1109/TSE.2007.1001

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. Systematic literature reviews in software engineering – A systematic literature review. Information and Software Technology, 51(1), 7–15. 2009. Disponible en: http://doi.org/10.1016/j.infsof.2008.09.009

Kitchenham, B., Pretorius, R., Budgen, D., Pearl Brereton, O., Turner, M., Niazi, M., & Linkman, S. Systematic literature reviews in software engineering – A tertiary study. Information and Software Technology, 52(8), 792–805. 2010. Disopnible en: http://doi.org/10.1016/j.infsof.2010.03.006

Leon Martinez, N. E., Aguilar Garcia, J. L., Vega Morales, E. F., & Gomez Florez, L. C. Herramienta computacional para la documentación de pruebas de software enmarcado en actividades de investigación Computer tool for software testing documentation under research activities. Scientia et Technica, 18(4), 682–689. 2013

León Martínez, N. E., Mendoza Castellanos, A., & Gómez Flórez, L. C. Propuesta de un sistema para la evaluación de calidad de software derivado de actividades de investigación. Bucaramanga. 2011

León Martínez, N. E., Pimentel Ravelo, J. I., & Gómez Flórez, L. C. Herramienta computacional para la gestión y evaluación de procesos software enmarcados en actividades de investigación. Scientia et Technica, 3(49). 2011

León Martínez, N. E., Pinto Chacon, N., & Gómez Flórez, L. C. Herramienta computacional para la evaluación de calidad de productos software enmarcados en actividades de investigación. Scientia et Technica, XVI(48). 2011

León, N. E., & Pimentel, J. I. Herramienta computacional para la gestión y evaluación de proyectos software enmarcados en actividades de investigación. Scientia et Technica, (47). 2011

Márquez Sosa, G. Glosario Estándar de Términos utilizados en pruebas Software. 2008

Moreno tarazona, m. A., vergel yaruro, p. L., & gómez flórez, l. C. Creación de un portal web para facilitar el acceso a los recursos y a la información del consultorio jurídico de la Universidad Industrial de Santander. 2014. Disponible en: http://doi.org/10.1017/CBO9781107415324.004

Myers, G. J., Badgett, T., Thomas, T. M., & Sandler, C. The Art of Software Testing (2nd ed.). 2004

Naik, K., & Tripathy, P. Software Testing and Quality Assurance. Hoboken, NJ, USA: John Wiley & Sons, Inc. 2008. Disponible en: http://doi.org/10.1002/9780470382844

NIST. Software Errors Cost U.S. Economy $59.5 Billion Annually. 2002. DIsponible en: http://www.abeacha.com/NIST_press_release_bugs_cost.htm

Oberhauser, R. Towards automated test practice detection and governance. In 1st International Conference on Advances in System Testing and Validation Lifecycle, VALID 2009. 2009. Disponible en: http://doi.org/10.1109/VALID.2009.25

Osorio Sanabria, M. A., Gonzáles Zabala, M. P., & Gómez Flórez, L. C. Sistema de informacion para apoyar el control de las actuaciones de los estudiantes del consultorio juridico de la UIS CYSACJ-UIS. 2006

Petticrew, M., & Roberts, H. Systematic Reviews in the Social Sciences: A Practical Guide. Cebma.Org. 2006. Disponible en: http://doi.org/10.1027/1016-9040.11.3.244

Pino, F. J., García, F., & Piattini, M. Software process improvement in small and medium software enterprises: a systematic review. Software Quality Journal, 16(2), 237–261. 2007. Disopnible en: http://doi.org/10.1007/s11219-007-9038-z

Plata robles, c. C., sepùlveda sarmiento, a. A., & gómez flórez, l. C. Desarrollo del sistema de información para el programa de selecciones deportivas adscrito a la dirección de bienestar universitario de la Universidad Industrial de Santander. 2015. Disopnible en: http://doi.org/10.1017/CBO9781107415324.004

Pressman, R. S. Software Engineering A pracitioner's Approach (7th ed.). McGraw-Hill Science/Engineering/Math. 2009

Ramírez Gómez, H., De La Hoz Freyle, J. E., & Gómez Flórez, L. C. Herramienta software basada en la arquitectura soa para el control de procesos del servicio de ensayo y mantenimiento de equipos del laboratorio de cromatografía de la Universidad Industrial de Santander. 2010

Reyes Giraldo, L. F., & Gomez Florez, L. C. Adaptación y desarrollo de modulos de la herramienta software hslab en cloud computing, para el mejoramiento al control de procesos de análisis de aguas residuales, naturales y subterraneas del laboratorio CEIAM. 2014. Disponible en: http://doi.org/10.1017/CBO9781107415324.004

Richardson, I., & Von Wangenheim, C. G. Why Are Small Software Organizations Different? IEEE Software, 24(1). 2007. Disponible en: http://doi.org/10.1109/MS.2007.12

Sommerville, I. Software Engineering (9th ed.). 2011

Sulayman, M., & Mendes, E. A Systematic Literature Review of Software Process Improvement in Small and Medium Web Companies. 2009

UIS. Contenido Ingenieria del Software I. 2013a Disopnible en: http://www.uis.edu.co/webUIS/es/asignaturas/contenidoAsig.jsp?codigo=22969

UIS. Contenido Ingenieria del Software II. 2013b. Disponible en: http://www.uis.edu.co/webUIS/es/asignaturas/contenidoAsig.jsp?codigo=22973

UIS. Plan de Estudios del Programa Académico de Ingeniería de Sistemas. 2013c. Disopnible en: http://www.uis.edu.co/webUIS/es/academia/facultades/fisicoMecanicas/escuelas/ingenieriaSistemas/programasAcademicos/ingenieriaSistemas/planEstudios.html

Vinay, P. F. Manage Software Testing. 2008

Wangenheim, C. G. Von, & Varkoi, T. Standard Based Software Process Assesments in Small Companies. Software Process: Improvement and Practice, 11(3). 2006. Disopnible en: http://doi.org/10.1002/spip

Yeşildoruk, F. Ç., Bozlu, B., & Demirörs, O. The Tool Coverage of Software Process Improvement Frameworks for Small and Medium Sized Enterprises. 2009

ANEXOS

ANEXO A. Proceso de Revisión bibliográfica

Dado que las pruebas de software juegan un papel importante en el aseguramiento de la calidad y las PyME desarrolladoras de software tienen características particulares, se planteó una revisión de la literatura en busca de trabajos en donde se hayan adelantado esfuerzos en la incorporación de las pruebas de software en este tipo de organizaciones.

Con el propósito de seguir un proceso estructurado y ordenado en la revisión bibliográfica. Se decidió hacer uso de la revisión literaria sistemática. Según Kitchenham una revisión sistemática es una forma de identificar evaluar e interpretar todas las investigaciones relevantes y disponibles para una pregunta de investigación, un área temática o un fenómeno de interés (B Kitchenham, 2007, p. 3). El proceso de revisión fue guiado de forma inicial por los siguientes documentos de interés:

- Systematic Review in Software Engineering(Biolchini et al., 2005): define un protocol de revision sistematicas en el area de ingeneira del software
- Guidelines for performing Systematic Literature Reviews in Software Engineering, Version 2.3 (B Kitchenham, 2007): es un conjunto de recomendaciones para realizer reviiones sistematicas en el area de la ingeniera del software
- Systematic literature reviews in software engineering – A systematic literature review(Barbara Kitchenham et al., 2009): Es un estudio donde se hace uso del proceso de revision definido por (B Kitchenham, 2007)

El procedimiento de revisión se fundamentó en los anteriores documentos y otras revisiones realizadas en el área de ingeniería del software (B a Kitchenham, Mendes, & Travassos, 2007; Pino, García, & Piattini, 2007; Sulayman & Mendes, 2009). El protocolo de revisión empleado se basó en el témplate propuesto en (Biolchini et al., 2005, p. 25). A continuación se describen los pasos generales:

6. Formulación de la(s) pregunta(s) que busca responde la revisión
7. Selección de las fuentes de consulta
8. Selección de los estudios de interés
9. Extracción de la información
10. Resumen y resultados

1 Formulación de la(s) pregunta(s) que busca responde la revisión

La formulación de la pregunta de investigación se hizo a través del criterio conocido como PICOC[52] (Petticrew & Roberts, 2006). Donde se estable que una pregunta bien formulada debe constar de 5 partes: población, intervención, contexto, resultados, y comparación.

Para el propósito de esta investigación: la población está conformada por las pequeñas y medianas empresas, se intervendrá en el tema de las pruebas de software, buscando resultados relaciones con recomendaciones, lineamientos y guías en la implantación y aplicación de las pruebas en el contexto del software. Para el propósito de este trabajo no será tenido en cuenta la parte de comparación, pues no hace parte del propósito del presente trabajo. Ver tabla 1.

[52] Siglas en ingles de population, , intervention, context, outcome and comparison

Tabla 1: Resumen de términos base del planteamiento PICOC

N°	Criterio	Términos en Español	Términos en Ingles
1	Population y Context	Pequeña y mediana empresa	Small and medium enterprise
2	Intervention	Prueba	Testing
3	Comparison	N/A	N/A
4	Outcomes	Buenas Practicas	Good Practices

Teniendo en cuenta lo anterior se plantearon la siguiente pregunta principal de investigación:

PI1. ¿Qué buenas prácticas son usadas por las PyME desarrolladora de software para llevar a cabo las pruebas de software?

Con el propósito de ayudar las PyME desarrolladoras de software en la implantación de los hallazgos, se planta la siguiente pregunta secundaria:
PI2-¿Que problemas se encuentran al tratar de usar las pruebas de software en PyME?

2 Selección de las fuentes de consulta

Partiendo de la pregunta principal de investigación planteada, se procede a formular la ecuación de búsqueda. Para lo cual es necesario seleccionar los términos a usar. Se parte de los términos empleados en el criterio PICOC, ver tabla 1. Los cuales son luego extendidos a través del uso de sinónimos y temimos relacionados encontrados durante el proceso de revisión. En la tabla 2 se encuentra el listado de términos usados.

Tabla 2: Todos los términos usados en la búsqueda

N°	Términos Base	Términos relacionados
1	Small and medium enterprise	SMEs , SME , small enterprise , Small and medium business , Small and medium-sized business , small business , SMBs , SMB , Small and medium , organization , Small and medium-sized , organization , small , organization , Very small entity , Very small entities , VSE , small company , small companies , small setting , small firm
2	Testing	software testing , software test , test process , test processes , verification , validation
3	Good practice	model, framework, methodology, approach, technique , standard , best practice , best practices, good practice, good practices, review

Para completar la formulación de la ecuación se búsqueda, se procede a usar el operador lógico AND para unir cada una de las partes definidas en el PICOC y el operador lógico OR para unir términos relacionados (B a Kitchenham et al., 2007, p. 318). Obteniendo la ecuación de búsqueda:

("Small and medium enterprise" or "small and medium-sized enterprise" or "small and medium-sized enterprises" or SMEs or SME or "small enterprise" or "Small and medium business" or "Small and medium-sized business" or "small business" or SMBs or SMB or "Small and medium organization" or "Small and medium-sized organization" or "small organization" or "Very small entity" or "Very small entities" or VSE or "small company" or "small companies" or "small setting" or "small firm") and
("software testing" or "software test" or "test process" or "test processes" or testing or verification or validation) and
(model or framework or methodology or approach or technique or standard or "best practice" or "best practices" or "good practice" or "good practices" or review) and
(software)

La búsqueda se llevó acabo en el motor de base datos Scopus. Esto como resultado del gran número de revistas indexadas con las que cuenta, y, las herramientas de búsqueda avanzada que posee.

3 Selección de los estudios de interés

Para la selección de los estudios de interés se hizo uso del título, abstract y palabras claves, como se menciona en (Barbara Kitchenham et al., 2010, p. 795), de cada uno de los trabajo encontrado al usar la ecuación de búsqueda previamente definida, en el motor de bases de datos Scopus. Es de notar que solo fueron tenidos en cuenta los trabajos que podían ser accedidos desde los recursos electrónicos de la UIS y que se encontraran en los idiomas inglés, español y portugués. En total, fueron encontraron 28 trabajos los cuales fueron clasificados en tres categorías:

- C1: Estudios centrados en las pruebas de software en PyME desarrolladora de software.
- C2: Estudios que no se centran en las pruebas de software en PyME desarrolladora de software, sin embargo realizan algún tipo de aporte, que permite conocer cómo son llevadas a cabo las pruebas de software en PyME desarrolladora de software.
- C3: En el estudio no se analizan las pruebas de software, ni se aporta información valiosa de su realización en PyME desarrolladora de software.

En la tabla 3, se muestra la cantidad de artículos encontrados en cada categoría.

Tabla 3: Clasificación de los artículos encontrados.

Categoría	# de Artículos
C1	2
C2	5
C3	21

Al finalizar la revisión, se obtuvieron dos artículos en la categoría 1. Es decir, estos dos artículos satisfacen los criterios de inclusión.

- **A1**: Oberhauser R., Towards automated test practice detection and governance, 1st International Conference on Advances in System Testing and Validation Lifecycle, VALID 2009, 2009, (Oberhauser, 2009)
- **A2**: Gleirscher M., Golubitskiy D., Irlbeck M., Wagner S., Introduction of static quality analysis in small- and medium-sized software enterprises: experiences from technology transfer, Software Quality Journal, 2013, (Gleirscher, Golubitskiy, Irlbeck, & Wagner, 2013)

4 Extracción de la información

El propósito esta actividad es obtener la información necesaria para responder la pregunta de investigación (B Kitchenham, 2007, p. 9). La información que se extraída de cada artículo fue tabulada de la siguiente manera:

- Titulo
- Fuente (Conferencia, revista, etc)
- Año de publicación
- Autores
- Objetivo del estudio
- País donde se realizó el estudio
- Número se PyME involucradas
- Información relacionada con las pruebas de software
- Buenas prácticas de prueba de software recomendada
- Problemas encontrados al incorporar y hacer uso de dichas buenas practicas
- Resultados del estudio

5 Resumen de resultados:

PI1. ¿Qué buenas prácticas son usadas por las PyMEDS para llevar a cabo las pruebas de software?

En (Gleirscher et al., 2013) se plantea el uso del análisis estático automatizado (ASA). Estudio que se llevó a cabo en 5 proyectos de 5 PyMEDS ubicadas en

Munich. Siendo usadas 3 técnicas ASA: detección de código duplicado, detección de patrones de bug y análisis de cumplimiento de la arquitectura. Este estudio encontró que el esfuerzo necesario para introducir las técnicas ASA en PyMEDS fue en general inferior a una hora-persona para cada técnica.

Hallazgos que son consistentes con la teoría donde se establece que las técnicas estáticas son una parte importante de las pruebas de software al permitir una temprana detección defecto, una reducción del costo total del proyecto y contribuyen al cumplimiento de los cronogramas (ISO/IEC/IEEE 29119-1, 2013, p. 14).

PI2-¿Que problemas se encuentran al usar estas buenas prácticas de pruebas de software en PyME?

Recomendación	Dificultad Encontrada
Uso de técnicas ASA	A nivel general las dificultades encontradas son de carácter técnico, relacionadas con la configuración y puesta a punto de las herramientas usadas para ASA. Respecto a cada técnica empleada se tiene: - Detección de código duplicado: Se encontraron algunos problemas menores, relacionando con la configuración de las herramientas a usar. - Detección de patrones de bug: Puede ser difícil relacionar de forma directa los hallazgos de las herramientas empleadas en esta técnica con atributos cualidades específicos. Presencia de un gran número de falsos positivos.(Gleirscher et al., 2013, p. 18) - Análisis de cumplimiento de la arquitectura: no siempre se cuenta con la arquitectura del sistema y cuando se tiene, no siempre se cuenta correctamente documentada(Gleirscher et al., 2013, p. 20).

ANEXO B. Revisión de los trabajos de grado

El propósito de esta revisión es conocer la forma como se desarrolla software[53] dentro de los grupos de investigación y en especial la forma como se gestiona las pruebas de software. La escuela de sistemas, por su área de estudio es la llamada a desarrollar el software dentro de la universidad. Siendo los estudiantes de pregrado los encargados de llevar a cabo esta tarea. El software es realizado en el marco de los proyectos de grado de final de carrera. La universidad exige la entrega de un libro como evidencia del proyecto de grado. Por lo anterior, se revisaron los documentos de final de carrera de los estudiantes vinculados a los grupos de investigación “Grupo de Investigación en Sistemas y Tecnología de la Información STI”, “Filosofía y Enseñanza de la Filosofía” y “Grupo SIMON de Investigaciones en Modelamiento y Simulación”. El proceso de revisión busco identificar la metodología de desarrollo software empleada, analizar la forma como se documentan los requisitos, el diseño y finalmente se buscó evidenciar el uso de la aplicación por parte del cliente. Las pruebas se analizaron de forma independiente y con un grado mayor de detalle, esto a cause de ser el tema central de la investigación en la que se enmarca esta revisión. El proceso de revisión se llevó a cabo a través del siguiente procedimiento:

1 Determinar la fuente de los documentos.
2 Definir criterios de búsqueda.
3 Seleccionar documentos de interés (criterios de inclusión y exclusión).
4 Extraer información de interés de los libros.
5 Sintetizar y resumir la información encontrada.
6 Interpretar los resultados.

[53] El proceso de desarrollo software es el proceso mediante el cual las necesidades del usuario se traducen en un producto software (ISO/IEC/IEEE 24765, 2010, p. 331)

1 Determinar la fuente de los documentos

Al finalizar sus carreras los estudiantes de la UIS, deben hacer entrega de un libro donde se consigna el desarrollo y los resultados obtenidos en sus proyectos de final de carrera. Este documento es un requisito para graduarse y puede ser consultado directamente en la biblioteca de la institución o a través del portal web dispuesto para tal fin (http://tangara.uis.edu.co/). El sitio web permite realizar búsquedas por autor, fecha de entrega, titulo, etc. Por esta razón, se decidió hacer uso de este portal web como la fuente de los documentos a revisar.

2 Definir criterios de búsqueda

El portal web de la biblioteca permite consultar los proyectos de fin de carrera de todos los egresados de esta institución de los últimos años. Por lo anterior, se hace necesario definir una serie de criterios que faciliten la búsqueda de los documentos de interés. A continuación, se listan los criterios tenidos en cuenta:

N°	Criterio	Justificación	Valor
1	Ventana de tiempo	Se definió una ventana de 2 años (2014-2015) para los grupos de investigación SIMON y FILOEN esto con el propósito de tener una visión reciente del proceso de desarrollo software. En el caso del grupo STI, como el caso de estudio del proyecto, se tuvo en cuenta una venta de tiempo mayor (2001-2015). Esto con el propósito de conocer más detalles de los trabajos allí realizados.	2 años (Grupo SIMON y FILOEN) 5 años (Grupo STI)
2	Autor	Se tuvieron en cuenta los proyectos donde el director del grupo fue uno de los autores del proyecto. Generalmente, el director del grupo participa de forma directa o indirecta en los proyectos del grupo y figura como autor o coautor del proyecto.	Director del grupo de investigación

N°	Criterio	Justificación	Valor
3	Tipo de documento	En el libro del proyecto de grado se encuentra consignada la información resultante de las actividades realizada durante el proyecto de grado en el marco de un proyecto de investigación. Por esto último, no fueron tenidas en cuenta las practicas.	Proyecto de grado

3 Seleccionar documentos de interés (criterios de inclusión y exclusión)

En este estudio solo fueron tenidos en cuenta los proyectos que contaban con un proceso de desarrollo de software. Para agilizar el proceso de revisión solo se tuvieron en cuenta los documentos con versión digital.

4 Extraer información de interés de los libros

El proceso de desarrollo software implica traducir las necesidades del usuario a requisitos software, convertir los requisitos a un diseño, implementar el diseño en código, probar el código, instalar y revisar el software para uso operacional (ISO/IEC/IEEE 24765, 2010, p. 331). El proceso de revisión se enfocó en estudiar las cuatro áreas fundamentales del proceso de desarrollo software los requerimientos, el diseño, el proceso de construcción y las pruebas. Se definió el siguiente formato para tabular la información de interés de cada trabajo revisado:

Tabla 2: Formato para extraer la información

Categoría	Variable-Criterio	Descripción de Valor
Información General	Título del Proyecto	Texto separado por espacios
	Número de Estudiantes	Numero entero (1 o 2)
	Nombre de Estudiantes	Texto separado por espacios y ;
	Numero de Directores	Numero entero positivo
	Nombre de Directores del Proyecto	Texto separado por espacios y ;
	Tipo de proyecto	Investigación, Practica
	Nivel de escolaridad	Pregrado/Posgrado

Categoría	Variable-Criterio	Descripción de Valor
	Año de Evaluación	Numero Entero
	Calificación Obtenida	Numero Decimal entre 3.5 y 5
	Palabras Claves	Texto separado por espacios y ,
	Se desarrolló una aplicación software	Si/No
	Tipo de desarrollo software	Web, Escritorio, móvil, librería, Web Services
Modelo de desarrollo software	Se usó algún tipo de modelo o metodología de ciclo de vida	Si/No
	Nombre del modelo o Metodología de Desarrollo Software empleada	Texto separado por espacios(en caso de no ser el resultado de varias se usara la palabra "Personalizada")
	Coherencia entre la metodología seleccionada y el desarrollo del proyecto plasmado en el documento	Si/No
Requisitos	Existencia de la definición de requisitos	Si/No
	Declaración explicita requisitos Funcionales	Si/No
	Declaración explicita requisitos no Funcionales	Si/No
Diseño	Existencia de un diseño (Result)	Si/No, no solo se debe mostrar el esquema general, la idea es que se ajuste a las tecnologías usadas en el proyecto
	Existencia Architectural design	Si/No
	Architectual Style	Multilayer, Multitier, client/server, etc o "No determinada"
	Existencia Detailed design	Si/No
	Declaración explicita del uso de patrones	Si/No
	Patrones usados	Testo separado por comas
	Evidencia del diseño de Interfaces de usuario	Si/No
	Presencia de UML	Si/No. En caso de haber usado por lo menos un diagrama UML se tiene Si
	Existencia de diagrama de casos de uso	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de diagramas de clases	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de diagramas de actividades	Si/No Presencia de al menos un diagrama de este tipo

Categoría	Variable-Criterio	Descripción de Valor
	Existencia de diagrama de paquetes	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de diagrama de estados	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de diagrama de despliegue	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de diagrama de colaboración	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de Diagrama de secuencia	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de Diagrama de componentes	Si/No Presencia de al menos un diagrama de este tipo
	Existencia de una base de datos	Si/No
	Diseño de la base de datos	Si/No
	Existencia de documentación de la base de datos	Si/No, Diagrama de Clases, modelo relacional
Codificación	Existencia de algún Estándar o Convención para codificar	Si/No
Pruebas	Existencia de un marco Teórico de pruebas	Si/No
	Existencia de Pruebas Caja Blanca (structural test desing technique)	Si/No
	Existencia de Pruebas Caja Negra (beavioral test desing technique)	Si/No
	Existencia de Pruebas Unitarias	Si/No
	Existencia de Pruebas de Componentes o subsistemas	Si/No
	Existencia de Pruebas de integración	Si/No
	Existencia de Pruebas de sistema	Si/No
	Existencia de Pruebas de Aceptación o aceptación de usuario	Si/No
	Existencia de Pruebas Beta (Pruebas de campo)	Si/No
	Existencia de Pruebas positivas y negativas	Si/No
	Existencia de Evidencia de pruebas (test case)	Si/No
	Existencia de Reporte de Errores	Si/No
	Existencia de Pruebas de usabilidad	Si/No

Categoría	Variable-Criterio	Descripción de Valor
Evidencia uso de la aplicación	Existencia de Pantallazos de la Aplicación	Si/No
	Evidencia del uso de la aplicación (experiencia de uso), opinión de usuario	Si/No
Información Adicional	Anotaciones	Texto
	Es claramente un proyecto del grupo	Si, No

5 Sintetizar y RESUMIR LA información encontrada

En la tabla 7 se encuentran los documentos encontrados al aplicar los criterios de búsqueda en la fuente de consulta establecida.

Tabla 19 Documentos encontrados por grupo

Grupo de Investigación	Documentos encontrados	Documentos de interés
STI	51	35
SIMON	12	10
FILOEN	12	11

Al revisar los documentos de interés de cada grupo se encontraron los siguientes hechos de interés con relación al proceso de desarrollo software en general:

1. En el caso del grupo STI y FILOEN usualmente se define un modelo de desarrollo software. En el caso del grupo SIMON a menudo se define un modelo de desarrollo software. A continuación, se muestra el porcentaje de trabajos en los que se definió un modelo de desarrollo software:

Grupo	%
STI	91
FILOEN	90
SIMON	73

2. En caso de haber definido un modelo de desarrollo en el grupo SIMON a menudo se documenta su seguimiento mientras que en los grupos STI y FILOEN es algo que se realiza con menos frecuencia. A continuación, se muestra el porcentaje de trabajos que habiendo definido un modelo de desarrollo software contaban con evidencia de su seguimiento:

Grupo	%
STI	56
FILOEN	56
SIMON	75

3. En los trabajos revisados a menudo tienen capturas de pantalla de su implementación. Sin embargo, en el caso del grupo STI y FILOEN menos del 45% de los trabajos contaban con evidencia de uso. A continuación, se muestra el porcentaje de trabajos con capturas de pantalla y evidencia de uso:

Grupo	% Capturas de Pantalla	% Evidencia de uso
STI	86%	43%
FILOEN	90%	20%
SIMON	73%	73%

4. Normalmente, los trabajos del grupo STI y FILOEN hacen uso de los UML para documentar sus desarrollos software mientras que grupo SIMON son a menudo. A continuación, se muestra el porcentaje de trabajos que hicieron uso de por lo menos un diagrama UML:

Grupo	%
STI	94
FILOEN	100
SIMON	73

Con relación al proceso de pruebas de software se encontraron los siguientes hechos:

1. El grupo SIMON a menudo realiza pruebas del sistema mientras que los grupos STI y FILOEN las realizan algunas veces. Las pruebas de componentes, integración y aceptación no son realizadas muy a menudo. A continuación, se muestra el porcentaje de trabajos por cada nivel de pruebas:

Tabla 20 Uso de los niveles de pruebas software

	% Proyectos con pruebas de componentes	**% Proyectos con pruebas de integración**	**% Proyectos con pruebas del sistema**	**% Proyectos con pruebas de aceptación**
STI	9	11	31	6
FILOEN	0	10	70	0
SIMON	0	9	45	18

2. Técnicas usadas (ver tabla 9).

Tabla 21 Uso de las técnicas de pruebas software

	% Proyectos con pruebas de Caja Negra	**% Proyectos con pruebas de Caja Blanca**
STI	54	14
FILOEN	80	10
SIMON	64	27

6 Interpretar resultados

De acuerdo a la información encontrada se puede concluir:

Tabla 22 Conclusiones

Situación	Evidencia
No se tiene certeza que los desarrollo software hayan sido realizado, pues no se tiene evidencia de que el usuario final lo haya usado, solo se tienen capturas de pantalla.	Hecho general 3
No se documenta el sigue la metodología de desarrollo software propuesta.	Hecho general 2
Se tiende a usar UML para documentar los desarrollos software. Siendo lo diagramas de casos de uso, clases, estados y secuencia.	Hecho general 4
Se tiende a hacer pruebas en etapas avanzadas de los desarrollos software.	Hecho de pruebas 1 y 2
Los interesados en estos desarrollos software no tienden a exigir la realización de pruebas de aceptación.	Hecho de pruebas 1

7 Contribución de las pruebas de software en los DESARROLLO DE SOFTWARE de GI

Entre los aportes que las pruebas software pueden brindar a los desarrollos software en GI se encuentran:

1- Complementar la documentación de los desarrollos software.
2- Brindar evidencia del uso de la aplicación.
3- Ver los casos de prueba definidos por el estudiante permitiendo al director del proyecto tener una mejor idea del grado de compresión del estudiante frente al software a realizar.
4- Incrementar la confianza en la calidad del producto final software.
5- Le permite al estudiante contemplar escenarios que por su falta de experiencia podría haber pasado por alto.

8 Análisis de lo Encontrado

Un factor que afecta en gran medida los desarrollo software en GI es la inexperiencia de los estudiantes en el desarrollo software, lo que se evidencia en los bajos índices de seguimientos de metodologías de software y realización de pruebas de software. Si bien es cierto y se detectaron algunas falencias en los planes de estudio, dada la naturaleza teórico-práctica del desarrollo software podrían superarse con el acompañamiento adecuado. Por lo anterior resulta de gran importancia invitar a la realización de pruebas de software en este tipo de desarrollo software. La realización de pruebas unitaria y de aceptación podría influir de forma positivo a la calidad del desarrollo software realizado.

ANEXO C. Plan de estudio de la escuela de ingeniería de sistemas e informática

El bajo grado de formalidad de las de pruebas de software encontrada conduce, ver anexo B, a pensar que tal vez es un problema de formación. Por lo anterior se procede a revisar el plan de estudios la carrera de ingeniería de sistemas e informática.

El programa académico para Ingeniería de Sistemas ofrecido por la (UIS) (UIS, 2013c) cuenta con dos asignaturas orientadas hacia el diseño, operación y mantenimiento del software. Ingeniera del software I y II, cada asignatura cuenta con una intensidad horaria de 64 horas. A continuación se lista el contenido de estas asignaturas(UIS, 2013a, 2013b):

Ingeniería del Software I	Ingeniería del Software II
1. Ciclo de vida. 2. Requisitos. 3. La arquitectura Software (Diseño Conceptual Software). 4. Diseño Detallado. 5. Pruebas software.	1. Administración de equipos de desarrollo software 2. Control de Calidad Software 3. Proyecto Software 4. Arquitecturas, funcionalidades y aplicaciones 5. Ingeniería del software soportado en componentes

Las pruebas de software es un tema que se trabaja en el curso de ingeniera del software I. Sin embargo, surge la inquietud de si este curso es suficiente para cubrir los conceptos fundamentales de las pruebas. Para responder a esta pregunta se acude al Software Engineering 2004 Volume (SE2004), el cual define el Software Engineering Education Knowledge (SEEK). El SEEK no representa un currículo, sin embargo proporciona el fundamento para el diseño, implementación

y entrega de las unidades educativas que constituyen un currículo en ingeniería del software(IEEE & ACM, 2004).

El SEEK está organizado jerárquicamente en tres niveles(IEEE & ACM, 2004):

- Área de conocimiento: son elementos estructurales de alto nivel usados para la organización, clasificación, y describen el conocimiento en la ingeniería del software.
- Unidades: las cuales representan módulos temáticos individuales en un área.
- Tópicos: las unidades son divididas en tópicos.

Áreas y unidades de conocimiento del SEEK

KA/KU	Title	hrs	KA/KU	Title	hrs
CMP	**Computing Essentials**	**172**	**VAV**	**Software V & V**	**42**
CMP.cf	Computer Science foundations	140	VAV.fnd	V&V terminology and foundations	5
CMP.ct	Construction technologies	20	VAV.rev	Reviews	6
CMP.tl	Construction tools	4	VAV.tst	Testing	21
CMP.fm	Formal construction methods	8	VAV.hct	Human computer UI testing and evaluation	6
			VAV.par	Problem analysis and reporting	4
FND	**Mathematical & Engineering Fundamentals**	**89**	**EVL**	**Software Evolution**	**10**
FND.mf	Mathematical foundations	56	EVO.pro	Evolution processes	6
FND.ef	Engineering foundations for software	23	EVO.ac	Evolution activities	4
FND.ec	Engineering economics for software	10			
PRF	**Professional Practice**	**35**	**PRO**	**Software Process**	**13**
PRF.psy	Group dynamics / psychology	5	PRO.con	Process concepts	3
PRF.com	Communications skills (specific to SE)	10	PRO.imp	Process implementation	10
PRF.pr	Professionalism	20			
MAA	**Software Modeling & Analysis**	**53**	**QUA**	**Software Quality**	**16**
MAA.md	Modeling foundations	19	QUA.cc	Software quality concepts and culture	2
MAA.tm	Types of models	12	QUA.std	Software quality standards	2
MAA.af	Analysis fundamentals	6	QUA.pro	Software quality processes	4
MAA.rfd	Requirements fundamentals	3	QUA.pca	Process assurance	4
MAA.er	Eliciting requirements	4	QUA.pda	Product assurance	4
MAA.rsd	Requirements specification & documentation	6			
MAA.rv	Requirements validation	3			
DES	**Software Design**	**45**	**MGT**	**Software Management**	**19**
DES.con	Design concepts	3	MGT.con	Management concepts	2
DES.str	Design strategies	6	MGT.pp	Project planning	6
DES.ar	Architectural design	9	MGT.per	Project personnel and organization	2
DES.hci	Human computer interface design	12	MGT.ctl	Project control	4
DES.dd	Detailed design	12	MGT.cm	Software configuration management	5
DES.ste	Design support tools and evaluation	3			

Fuente: (IEEE & ACM, 2004, p. 27)

Al comparar el contenido de la asignatura de ingeniera del software I con el SEEK, en las aéreas de conocimiento MAA, DES y el VAV. Permiten ver falencias en el contenido de la asignatura. En el caso de las pruebas, el SEEK propone 21 horas para el desarrollo de este tema, lo cual sería casi un tercio de la duración de la asignatura de ingeniera del software I, de lo que se concluye que difícilmente en el curso de ingeniería del software I, se adquieran todos los conocimientos básicos relacionados con las pruebas de software. Esto explicaría por qué los desarrollos software de la escuela tienden a carecer de planes de pruebas e incluso de casos de pruebas.

ANEXO D. Detalles del prototipo software

1 Tecnologías empleadas

El sitio web se desarrolló en JSF y Primefaces. La base de datos se realizó en MySQL.

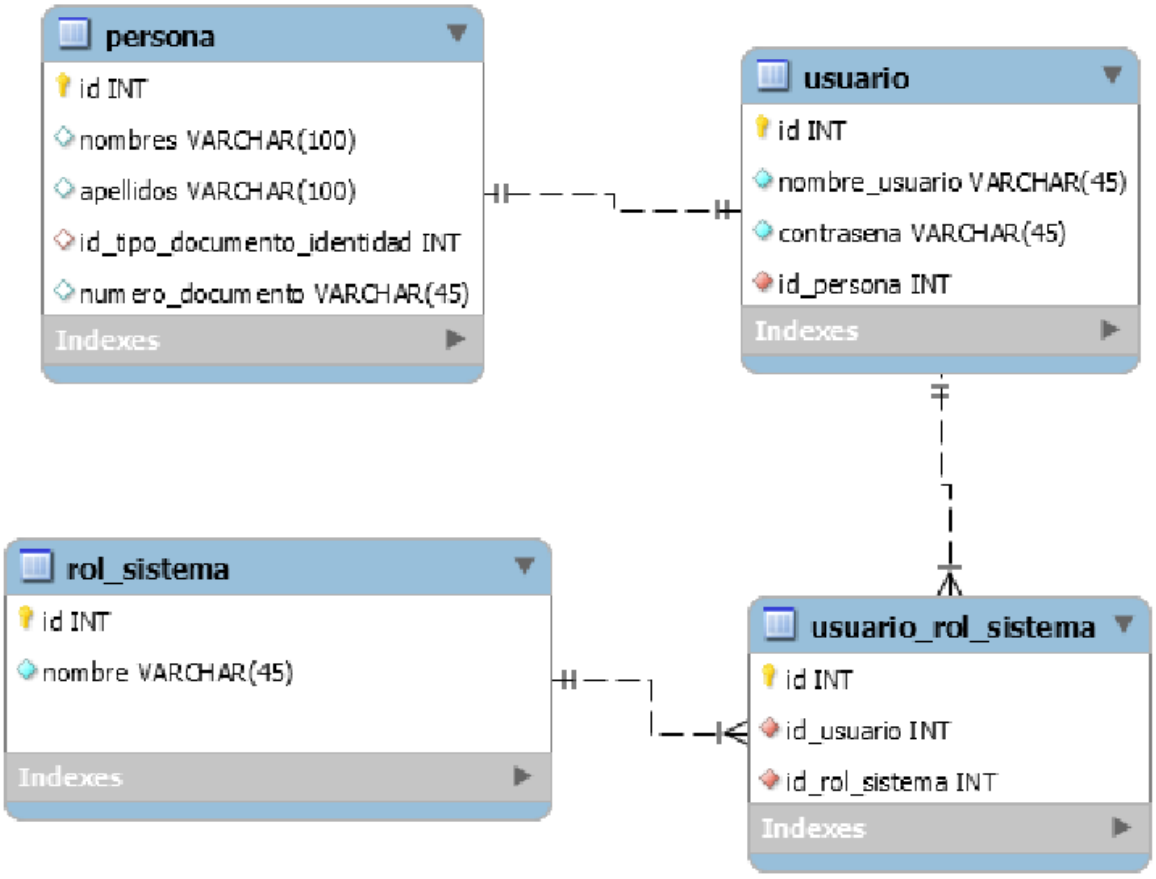
2. Definición de Requisitos

ID	Actores	Requisito
RQ-1	Director general de pruebas	Asignar director general
RQ-2	Director general de pruebas	Reasignar director general
RQ-3	Director general de pruebas	Formular política de pruebas
RQ-4	Director general de pruebas	Editar política de pruebas
RQ-5	Director general de pruebas	Formular estrategia de pruebas
RQ-6	Director general de pruebas	Editar estrategia de pruebas
RQ-7	Administrador de pruebas	Crear plan de pruebas
RQ-8	Administrador de pruebas	Modificar plan de pruebas
RQ-9	Administrador de pruebas	Eliminar plan de pruebas
RQ-10	Todos los usuarios	Consultar política de pruebas
RQ-11	Todos los usuarios	Consultar estrategia de pruebas
RQ-12	Todos los usuarios integrantes de un proyecto	Consultar plan de pruebas
RQ-13	Tester	Crear caso de pruebas
RQ-14	Tester	Editar caso de pruebas
RQ-15	Tester	Eliminar caso de pruebas
RQ-16	Todos los usuarios integrantes de un proyecto	Crear un incidente
RQ-17	Todos los usuarios integrantes de un proyecto	Actualizar el incidente

2 Modelo Entidad Relación

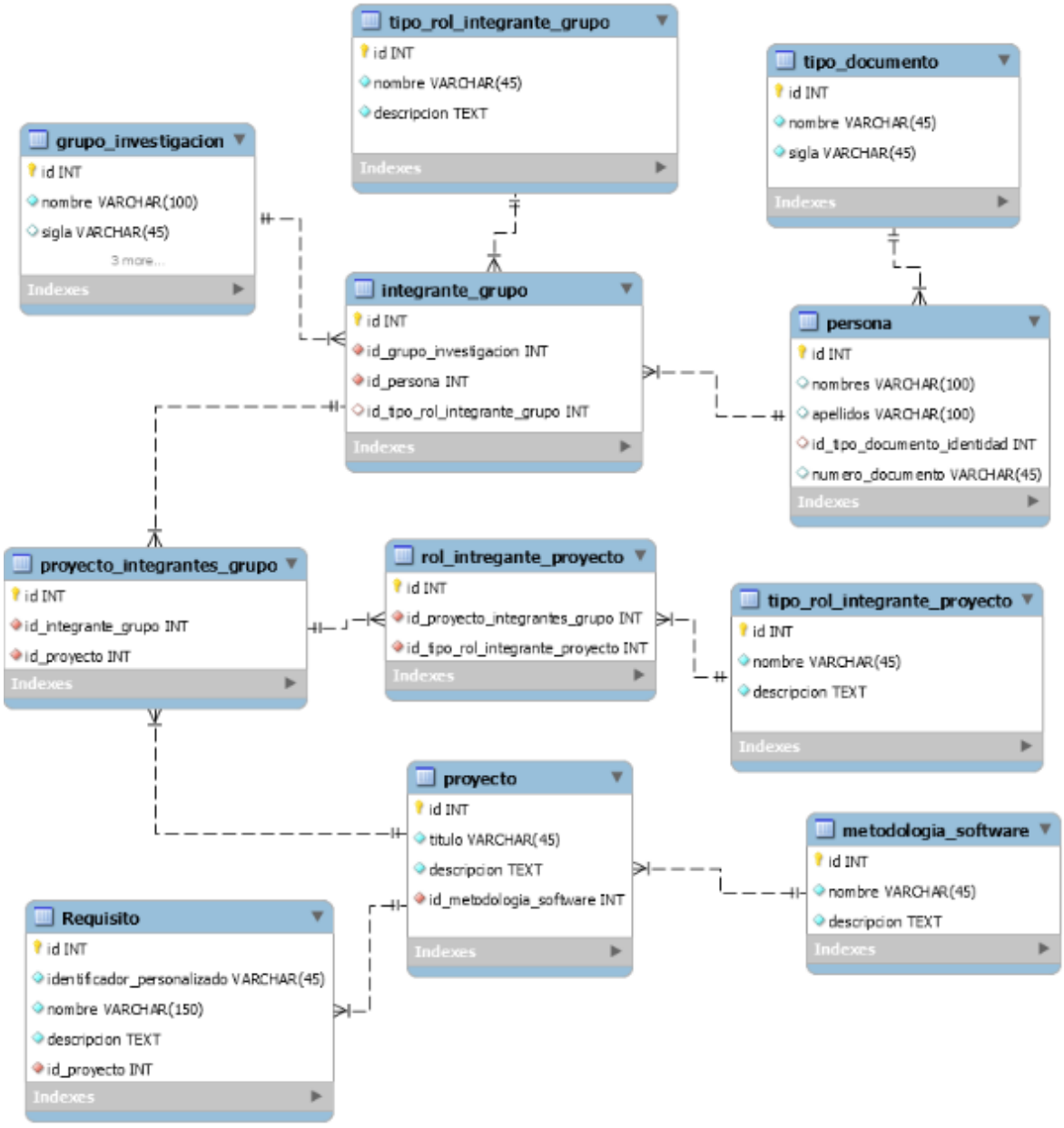
Usuarios

Figura 26 Diagrama entidad relacion usuarios



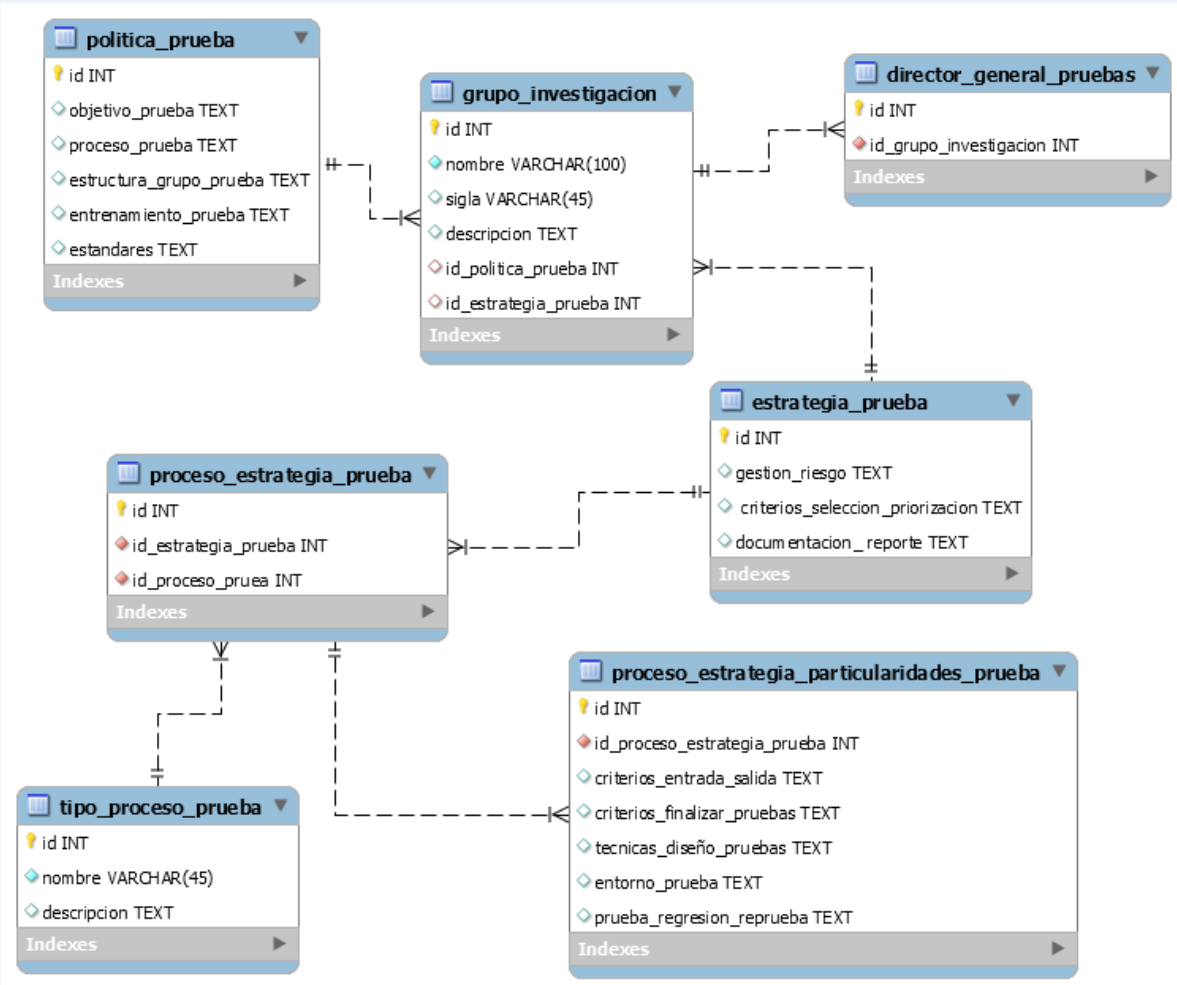
Grupo de investigación

Figura 27 Diagrama entidad relación información del grupo de investigación



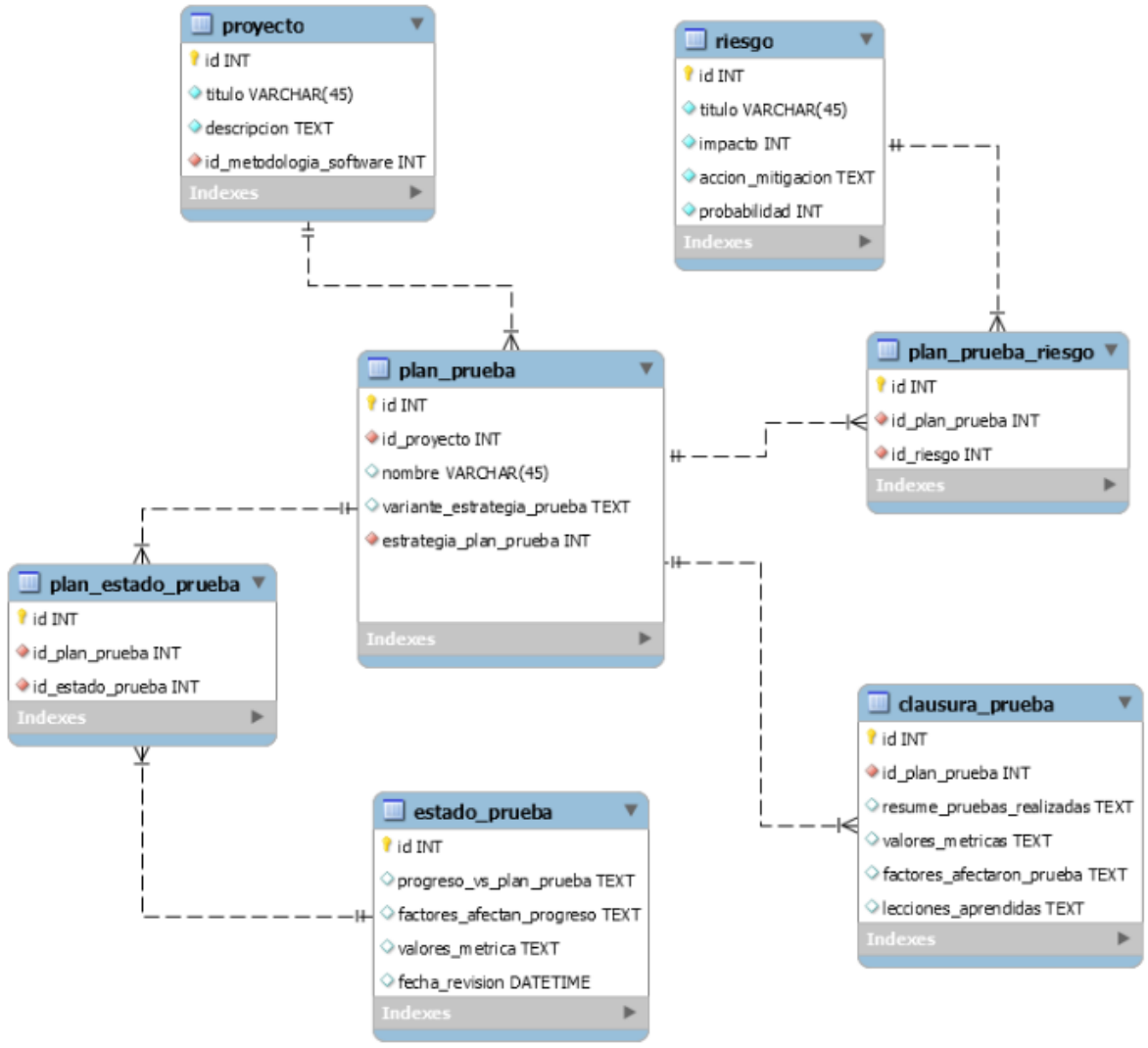
Nivel general de preubas

Figura 28 Diagrama entidad relación nivel general de pruebas



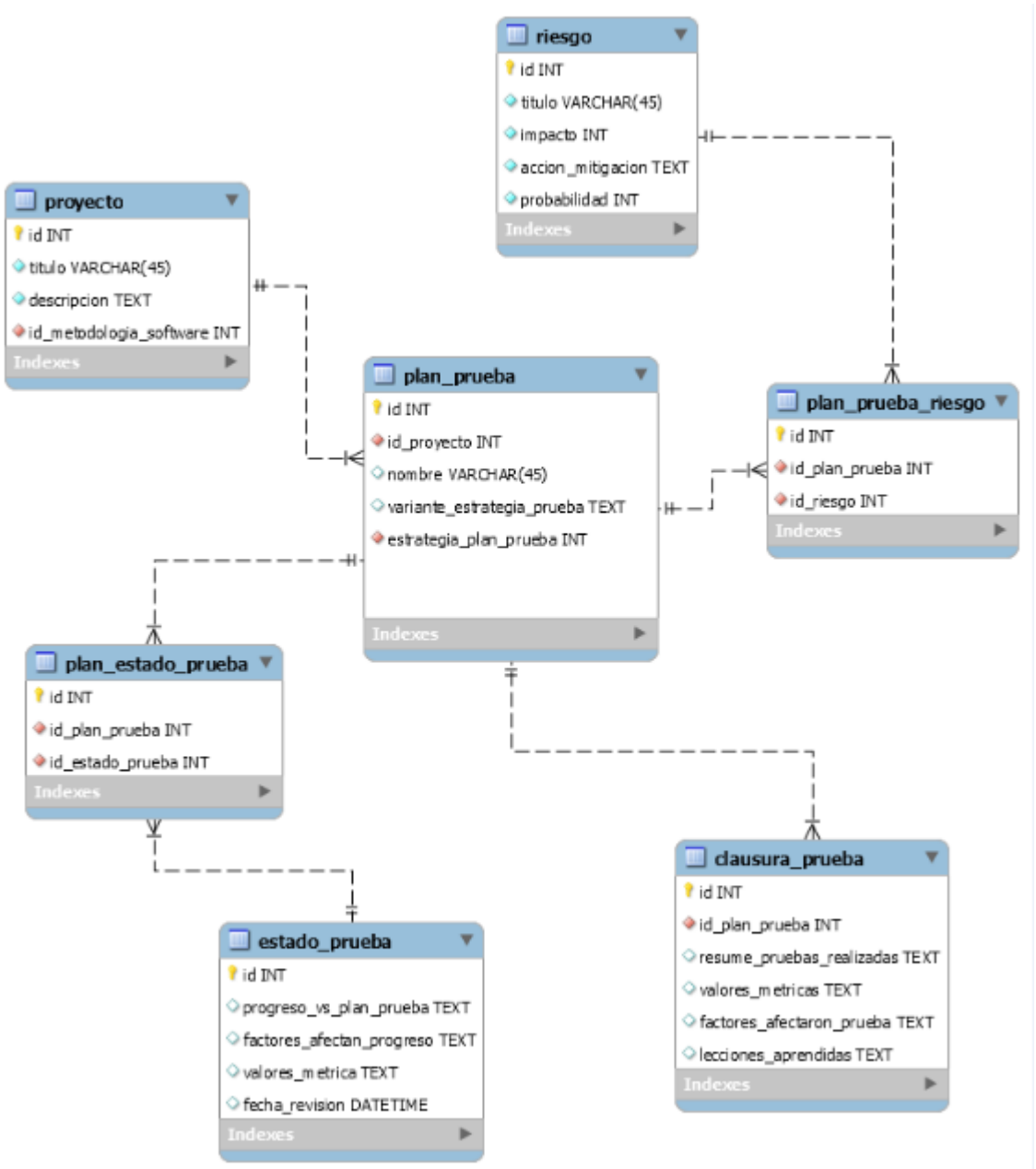
Nivel de gestion de pruebas

Figura 29 Diagrama entidad relación nivel de gestión de pruebas



Nivel de realización de pruebas

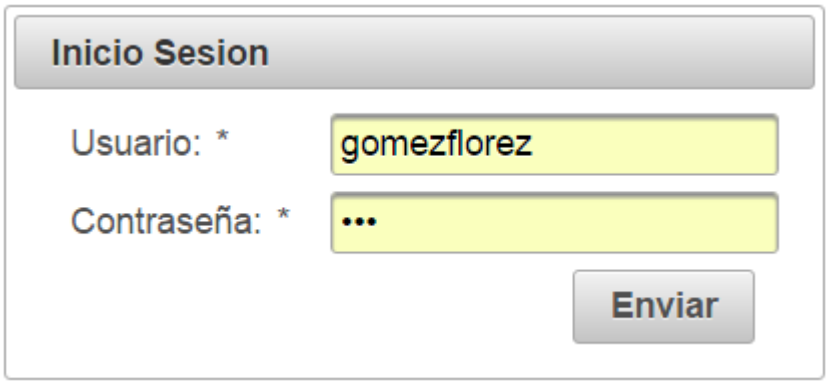
Figura 30 Diagrama entidad relación nivel de realización de pruebas



3 Capturas de pantalla de la aplicación

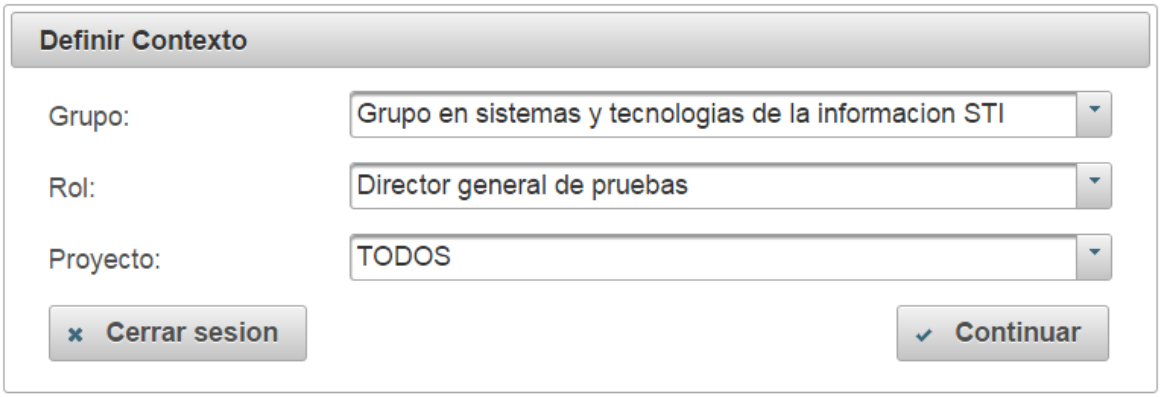
Inicio de sección

Figura 31 Captura de pantalla del inicio de sesión



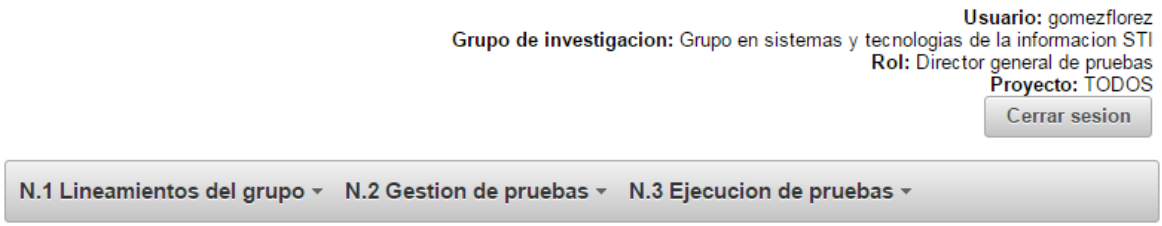
Selección del contexto

Figura 32 Captura de pantalla de la ventana de contexto



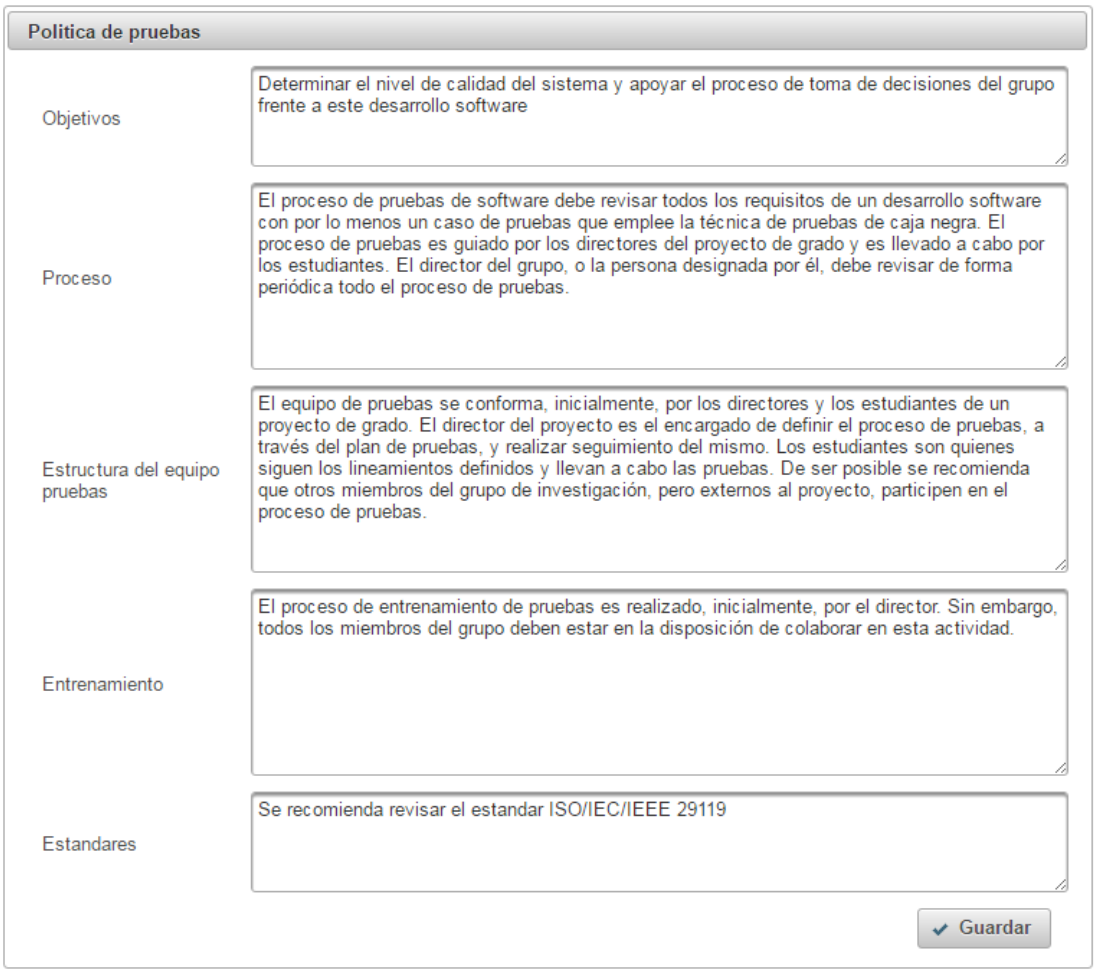
Menú Principal

Figura 33 Captura de pantalla del menú principal



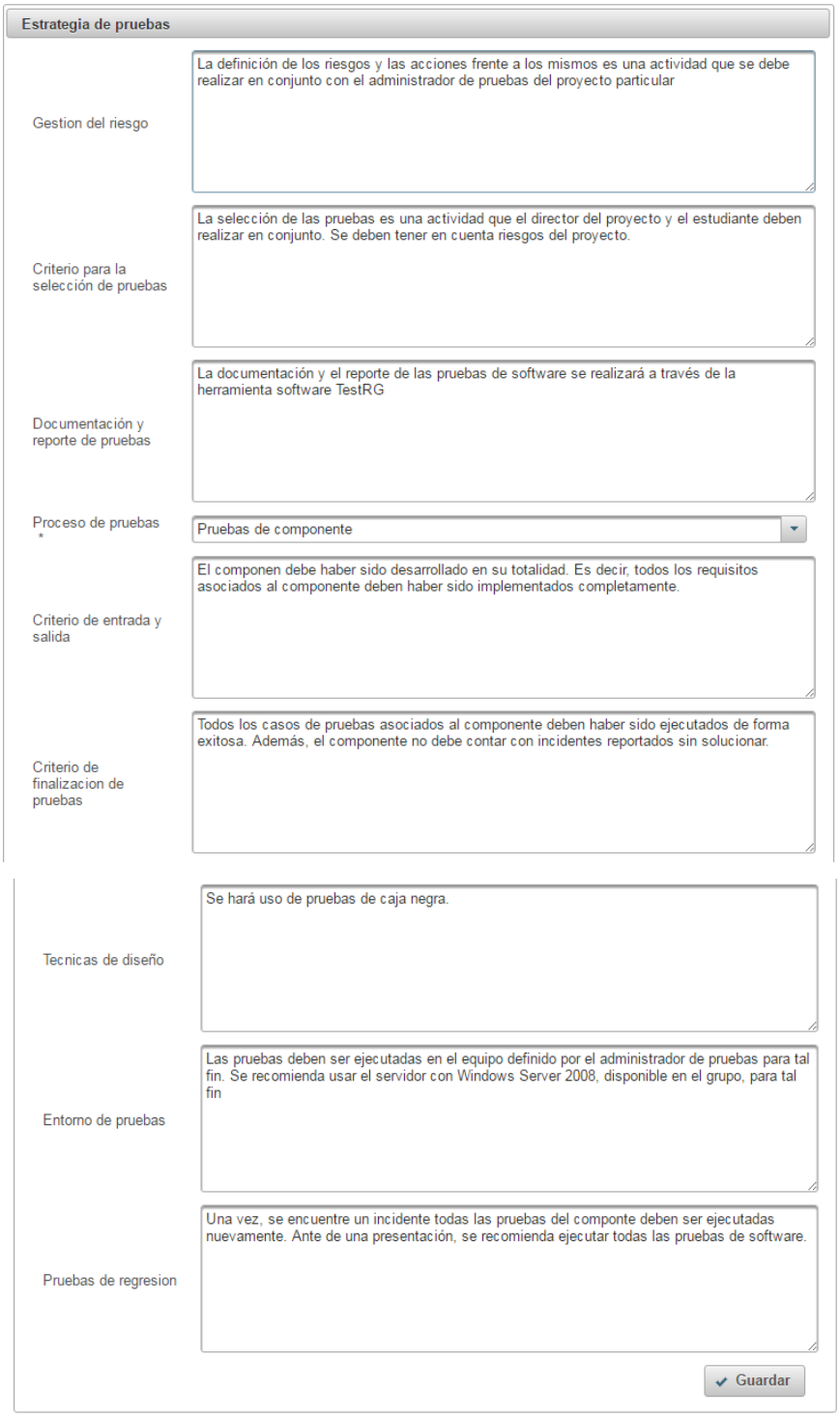
Política de pruebas

Figura 34 Captura de pantalla de la política de pruebas



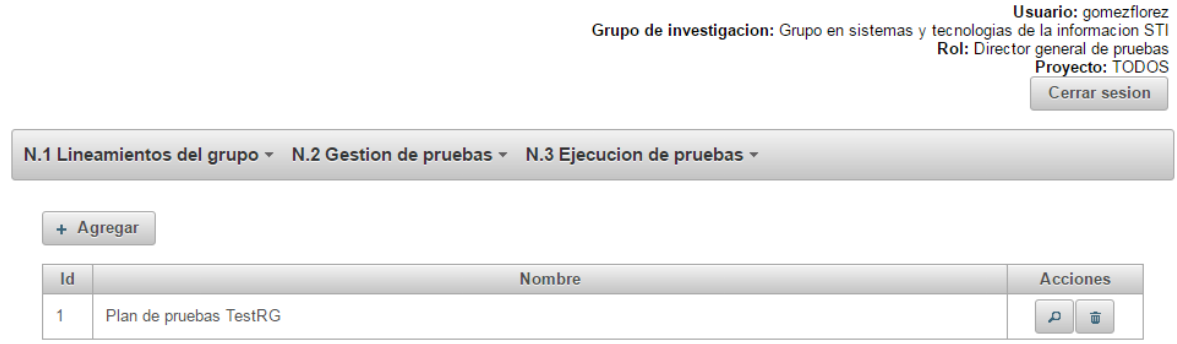
Estrategia

Figura 35 Captura de pantalla de la ventana de estrategia de pruebas



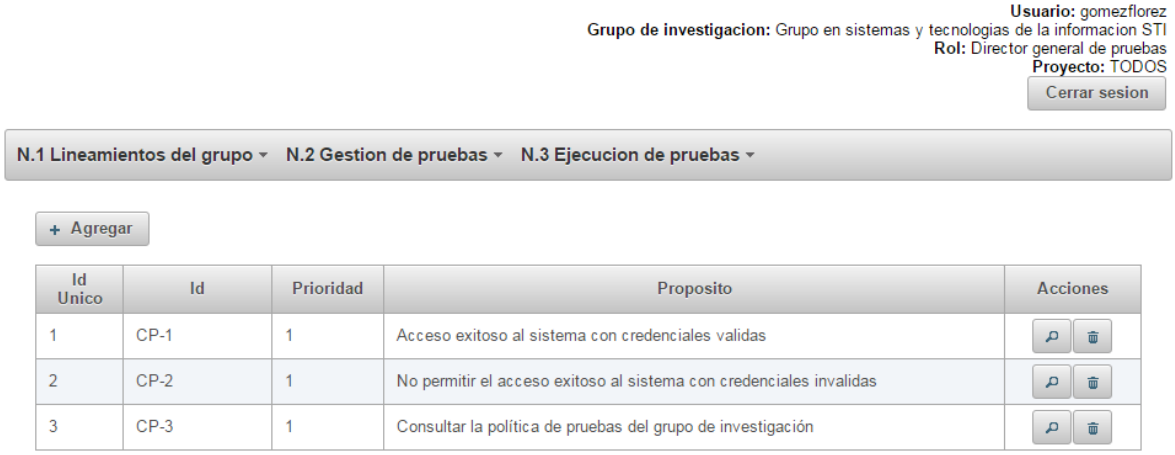
Listado de plan de pruebas

Figura 36 Captura de pantalla de la ventana donde se listan los planes de pruebas



Listado de casos de pruebas

Figura 37 Captura de pantalla de la ventana donde se listan los casos de pruebas



Listado de incidentes

Figura 38 Captura de pantalla de la ventana donde se listan los incidentes