ISBN: 978-958-15-0337-7

METODOLOGÍA PARA LA IMPLEMENTACIÓN DEL ASEGURAMIENTO DE CALIDAD EN LOS PRODUCTOS SOFTWARE, DESARROLLADOS POR APRENDICES DEL SENA

Sandra L. Buitrón
Luis E. Garzón
Julio C. Palechor

SERVICIO NACIONAL DE APRENDIZAJE

SENNOVA
Sistema de Investigación, Desarrollo Tecnológico e Innovación

SINERGIA
GRUPO DE INVESTIGACIÓN

Sinergia
Semillero de Investigación

Las empresas software de Colombia necesitan llevar a cabo estrategias de calidad de procesos y productos de manera planeada y sistemática, con el fin de promover el desarrollo de software de calidad. Estas estrategias deben ser sistemáticas y controladas para garantizar la fiabilidad de los productos generados durante el proceso de desarrollo y deben ser apoyadas por diferentes actores (entre los que cabe destacar el gobierno, la industria y la academia) buscando incrementar la calidad de productos software.

En este sentido, la academia en su contexto debe abordar el tema de calidad de software y por tanto brindar la importancia requerida durante la formación de estudiantes relacionados con la industria del software, lo que conduce a que muchos de los profesionales que trabajan en esta industria cuenten con fundamentos sólidos para apoyar la construcción de las estrategias de calidad, apropiadas para las organizaciones, así como con mecanismos para la implementación del aseguramiento de la calidad en estos procesos de desarrollo.

Este libro presenta una alternativa de implementación del aseguramiento de calidad y evaluación de la calidad de productos software en un contexto educativo, a través de la Metodología MECAPS - Metodología para la evaluación de la calidad de productos software.

SENA
SERVICIO NACIONAL DE APRENDIZAJE
SENNOVA
Sistema de Investigación, Desarrollo Tecnológico e Innovación
SINERGIA
GRUPO DE INVESTIGACIÓN
Sinergia
Semillero de Investigación

Metodología para la implementación del aseguramiento de calidad en los productos software, desarrollados por aprendices del SENA

Sandra L. Buitrón
Luis E. Garzón
Julio C. Palechor

Servicio Nacional de Aprendizaje
SENA
Centro de Comercio y Servicios
Regional Cauca

Catalogación en la publicación. SENA Sistema de Bibliotecas

Buitrón, Sandra L.
Metodología para la implementación del aseguramiento de calidad en los productos software, desarrollados por aprendices del SENA / Sandra L. Buitón, Luis E. Garzón, Julio C. Palechor. – Popayán: Servicio Nacional de Aprendizaje (SENA). Centro de Comercio y Servicios, 2018.

1 recurso en línea (80 páginas : PDF).

Referencias bibliográficas: páginas 79-80.
Contenido: Metodología MECAPS como estrategia de investigación -- Actividades de construcción -- Metodología para la evaluación de la calidad de los productos software desarrollados por los aprendices ADSI -- Modelado del proceso de desarrollo de ADSI -- Evaluación del proceso de desarrollo de ADSI -- Identificación de oportunidades de mejora -- Estructuración de la metodología -- Descripción de los mecanismos de SQA -- Conclusiones -- Aplicación de la metodología.
ISBN: 978-958-15-0337-7.

1. Desarrollo de programas para computador--Control de calidad 2. Aseguramiento de la calidad I. Garzón, Luis E. II. Palechor, Julio C. III. Servicio Nacional de Aprendizaje (SENA). Centro de Comercio y Servicios.

CDD: 005.14

ISBN: 978-958-15-0337-7
Primera edición, marzo de 2018

Director General
José Antonio Lizarazo Sarmiento, MSc.
Servicio Nacional de Aprendizaje - SENA

Director Regional Cauca
Hernando Ramírez Dulcey, Ing.
Servicio Nacional de Aprendizaje - SENA

Subdirector de Centro Comercio y Servicios
Edward Enrique Vargas Vivas, MSc.
Servicio Nacional de Aprendizaje - SENA

Líder Sennova Centro Comercio y Servicios
Alfaro Tandioy Fernández
Servicio Nacional de Aprendizaje - SENA

Asesor de Investigación Proyecto MECAPS
Francisco José Pino Correa
Servicio Nacional de Aprendizaje - SENA
Universidad del Cauca

Grupo y Semillero de Investigación
Sinergia

Corrección de estilo
Jaime Dávila

Diseño y Preprensa
Ingeniería Gráfica S.A.

SENA - REGIONAL CAUCA
Call 4 No. 2 – 80,
PBX: (028) 824 4372
www.sena.edu.co
Popayán, Cauca – Colombia

Dedicatoria

Sandra L. Buitrón
"A mis hijas, Belen y Celeste, quienes son el motor de mi vida"

Luis E. Garzón
"A mis hijos; Miguel Ángel y Simón, a Jimena mi esposa. A ellos por compartirme parte de su tiempo y compartirlo en este proyecto".

Julio C. Palechor
"A mi familia, en el día a día de mi proceso educativo. A mi Padre, con su apoyo continuo. A mi hija, compartiendo nuestro presente".

AGRADECIMIENTOS

Queremos expresar nuestro agradecimiento, a los aprendices del programa de formación Análisis y Desarrollo de Sistemas de Información del Centro de Comercio y Servicios de Popayán, en especial a los aprendices Cristian Duban Chilito, Lina Magali Dagua Taquinas, Flor Deli Julicue Escue, a los aprendices de la Especialización Tecnológica en Pruebas de Software y a las Ingenieras Raquel Sofía Florez y Vanessa Agredo quienes participaron en la ejecución de este proyecto de investigación. A Francisco J. Pino, por todos los aportes brindados y su asesoría para el desarrollo de la metodología. A los estudiantes de la asignatura de Calidad de Software del Programa de Ingeniería de Sistemas de la Universidad del Cauca, por su valiosa contribución en la evaluación de algunos artefactos software generados durante el proceso de desarrollo de ADSI, para este proyecto.

A Alfaro Tandioy, por su ánimo y apoyo constante, haciendo posible la adecuada ejecución del proyecto y para que esta publicación concluyera con éxito. Y a la Subdirección del Centro de Comercio y Servicios de Popayán, por toda su colaboración administrativa para que el proyecto se desarrollara de acuerdo con lo planeado.

Sandra L. Buitrón
Luis E. Garzón
Julio C. Palechor

CONTENIDO

TABLAS

FIGURAS

PLANTILLAS

AUTORES

Sandra L. Buitrón

Magíster en Computación, Universidad del Cauca, Colombia. Ingeniera de Sistemas de la Universidad Cooperativa de Colombia, Especialista en Redes de Comunicación de la Universidad del Valle (Colombia), Especialista en Sistemas Gerenciales de Ingeniería de la Pontificia Universidad Javeriana (Colombia). Estudiante de Doctorado en Ciencias de la Electrónica de la Universidad del Cauca. Profesora Ocasional, adscrita a la Facultad de Electrónica y Telecomunicaciones de la Universidad del Cauca. Miembro del grupo de Investigación IDIS (Investigación y Desarrollo en Ingeniería del Software) de la Universidad del Cauca. Basic Agile Testing Certification (CP-BAT) por BDGuidance. Ha trabajado en el área de gestión de proyectos y calidad de software en organizaciones software por más de diez años.

Luis Enrique Garzón

Periodista. Tecnólogo en Análisis y Desarrollo de Sistemas de Información. Estudiante de la Especialización en Metodologías de Calidad para el Desarrollo de Software. Ponente en el XI Encuentro Departamental del Semilleros de Investigación de la RedCOLSI Nodo Cauca "Metodología para Evaluación de la Calidad de Productos Software desarrollado por los aprendices de ADSI en el Centro de Comercio y Servicios de Popayán". Miembro del Grupo y Semillero de Investigación SINERGIA del SENA Centro de Comercio y Servicios de la Regional Cauca. Ha trabajado como analista en el proyecto "Construcción de un prototipo funcional que apoye la metodología de evaluación de la calidad de productos software".

Julio C. Palechor

Ingeniero de Sistemas, Universidad del Cauca. Estudiante de Máster en Tecnología Informática Avanzada, Universidad Castilla La Mancha – UCLM, España. Fue miembro del grupo IDIS (Investigación y Desarrollo en Ingeniería de Software) de la Universidad del Cauca. Miembro del grupo SINERGIA del SENA en el Centro de Comercio y Servicios de la Regional Cauca, apoyo a inicio de semillero de investigación, participación en fortalecimiento sobre formulación de proyectos, investigador en proyecto "Diseño de una metodología para la evaluación de la calidad de los productos software desarrollado por los aprendices de ADSI en el Centro de Comercio y Servicios de Popayán". Ponencia en el Quinto Congreso Colombiano de Computación con "pseudo patrones de procesos ágiles". Ha trabajado como instructor SENA, especialmente en formación profesional integral con tecnólogos ADSI por cuatro años.

CAPÍTULO 1

INTRODUCCIÓN

La industria del software en Colombia está conformada en su mayoría por pequeñas empresas [1], con el potencial para convertirse en un sector económico importante para el país que supla la demanda interna, conquiste mercados en el exterior y contribuya a consolidar el crecimiento del producto interno bruto; además esta industria ayuda a otros sectores de la economía a ser más eficientes y organizados, brindándoles soporte, agilizando sus procesos, facilitando sus comunicaciones y reduciendo sus costos de operación [2]. Esto evidencia la importancia de realizar esfuerzos para fortalecer el quehacer de este tipo de empresas, con el fin de que puedan tener mayor competitividad [3, 4]. Los productos que no hacen exactamente lo que se espera, que no se utilizan por la dificultad de su manejo, que son imposibles de mantener cuando desaparecen las personas que los desarrollaron, que no se terminan nunca, que son poco seguros, junto con las pérdidas monetarias que estos han ocasionado son problemas que aparecen como consecuencia de la falta de calidad en sistemas ya implantados. Estos hechos hacen necesaria la actualización de la percepción que se tiene acerca de la calidad del software [5].

La industria del software reconoce la importancia de establecer una estrategia de calidad como instrumento para que los productos de software desarrollados tengan la calidad esperada por el usuario final, y de esta manera se protejan de los riesgos de litigios, sobrecostos y pérdida de reputación ante los clientes [6] [7]. La premisa de que "la calidad del producto depende en gran parte de la calidad del proceso" [8] sugiere la necesidad de definir estas estrategias de calidad durante el proceso de desarrollo del software. Del mismo modo, el mercado valora cada día más la calidad, por lo tanto, la industria del software exige la disminución de errores, entregas oportunas y ejecución adecuada de los proyectos [9]. Como un ejemplo de esta relevancia, el proceso de pruebas, el cual hace parte de la estrategia de calidad y por tanto del aseguramiento de la calidad, es generalmente la fase más costosa que puede ser responsable de más del 50 % de los costos de desarrollo [10]; esto parece ser una gran inversión en una estrategia de calidad, sin

embargo, un proceso de pruebas puede ser eficiente con tan sólo el 20 % de los recursos [11], lo cual supone costos más bajos y productividad más elevada [12].

Las estrategias de calidad implementadas en las empresas dedicadas al desarrollo de software permiten: (i) generar información acerca de la calidad del producto y sacar a la luz la falta de calidad, que se revela a través de defectos/incidencias/no conformidades encontradas, (ii) disminuir problemas inherentes a sus productos, (iii) cumplimiento de los plazos y presupuestos, (iv) satisfacción del usuario y mejora en la productividad, y (v) aumento de la calidad en producto [9]. Definir una estrategia de calidad dentro de una organización, involucrando procesos, personas y productos de trabajo, es valioso en la medida en que se incremente la calidad del producto, se facilite la comprensión y comunicación entre los miembros del equipo, se dé el soporte necesario para mejorar continuamente el proceso y se proporcione soporte a la ejecución de los procedimientos [13] [14].

En este sentido, las empresas software de Colombia necesitan llevar a cabo estrategias de calidad de procesos y productos de manera planeada y sistemática, con el fin de promover el desarrollo de software de calidad [15]. Estas estrategias deben ser sistemáticas y controladas para garantizar la fiabilidad de los productos generados durante el proceso de desarrollo [16] y deben ser apoyadas por diferentes actores (entre los que cabe destacar el gobierno, la industria y la academia) buscando incrementar la calidad de productos software. Sin embargo, el tema de calidad de software es poco abordado por la academia, y por tanto no se brinda la importancia requerida durante la formación de estudiantes relacionados con la industria del software, lo que conduce a que muchos de los profesionales que trabajan en esta industria no cuenten con fundamentos sólidos para apoyar la construcción de las estrategias de calidad, apropiadas para las organizaciones, así como con mecanismos para la implementación del aseguramiento de la calidad en estos procesos de desarrollo.

Entre los elementos que puede incluir una estrategia de calidad están: pruebas de software, inspecciones de software, verificación y validación de software, revisiones de software y auditorías de software, entre otras. Las pruebas de software son comúnmente realizadas para el control de calidad en los proyectos de desarrollo [13], la inspección de software [17, 18], es una técnica de garantía de calidad del desarrollo del software; la verificación de software busca confirmar que cada producto de trabajo, o servicio de un proceso o proyecto, refleje adecuadamente los requisitos especificados, en tanto que la validación del software pretende confirmar que los requisitos para un uso intencionado específico del producto software están satisfechos [19]. Por su parte, las revisiones de software buscan mantener un entendimiento común con los stakeholders sobre el progreso de los objetivos del proyecto y sobre qué puede hacerse para ayudar a asegurar el desarrollo de un producto que satisfaga sus necesidades, mientras que las auditorías de software intentan determinar el cumplimiento de los requisitos, planes y acuerdos, por parte de los productos o procesos seleccionados independientemente [19].

Bajo este escenario, este libro presenta una alternativa de implementación del aseguramiento de calidad y evaluación de la calidad de productos software en un contexto educativo, a través de la Metodología MECAPS Metodología para la evaluación de la calidad de productos software (Figura 1). MECAPS es una herramienta de apoyo, generada al interior del proyecto de investigación denominado "Diseño de una metodología para la evaluación de la calidad de los productos software desarrollados por los aprendices de ADSI en el Centro de Comercio y Servicios de Popayán", el cual ha sido propuesto y ejecutado por el Semillero de Investigación SINERGIA y financiado por el Sistema de Investigación Desarrollo Tecnológico e Innovación - SENNOVA del SENA. Dentro de este contexto, MECAPS permite: (i) contribuir a la calidad de los productos software desarrollados por los aprendices ADSI del SENA, (ii) habilita a los aprendices acerca de los fundamentos básicos sobre calidad de productos software, y (iii) mejora la experiencia de aprendizaje sobre el proceso de desarrollo software, mediante la utilización práctica de elementos referentes al aseguramiento de la calidad. Estas destrezas y beneficios obtenidos durante la formación de los aprendices ADSI permitirán que ellos sirvan de agentes replicadores de la calidad de software cuando hagan parte de la Industria del software del país.

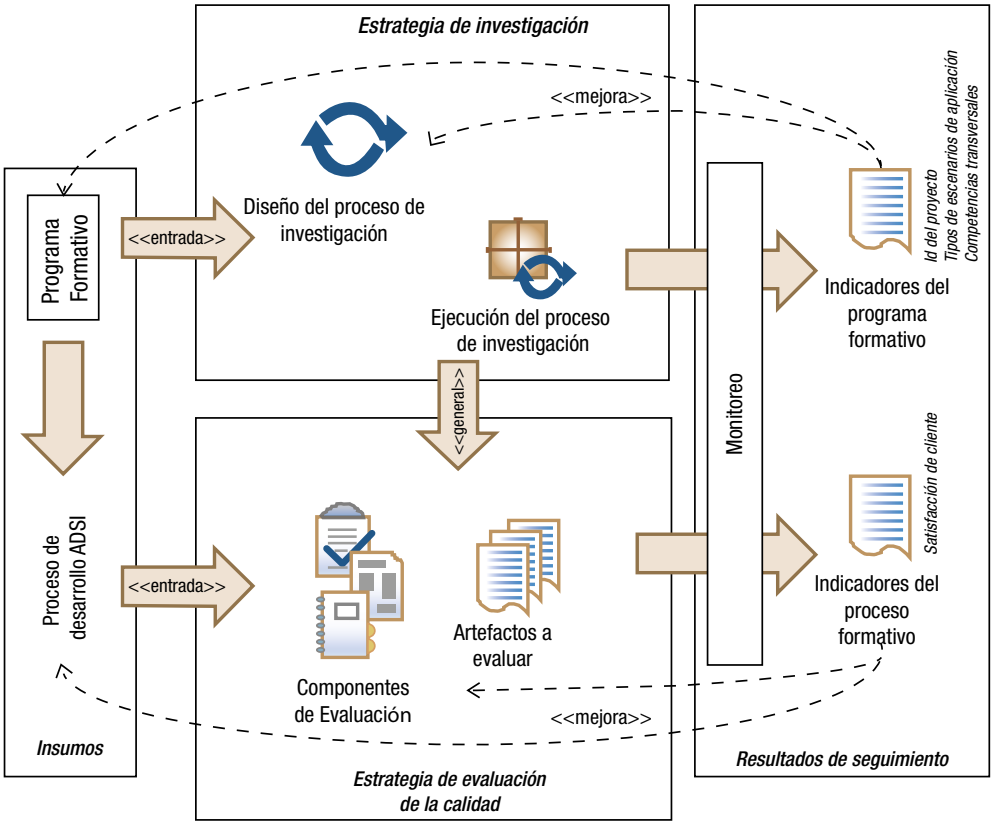


Figura 1. Estructura general de MECAPS

Como estrategia de evaluación de la calidad, MECAPS define: a) al proceso de desarrollo ADSI y los productos software generados durante este proceso como los objetos de evaluación (lo que se va a evaluar), b) a los estándares ISO de calidad de productos software, CMMI-DEV y modelos de referencia para gestión de proyectos como los referentes para la evaluación (contra lo que se va a comparar el objeto), y c) a los mecanismos de aseguramiento de calidad de software (tales como: RTF, pruebas de caja blanca, pruebas de caja negra, lista de chequeo sobre requisitos no funcionales y lista de chequeo de diagramas de clases, entre otros) como métodos y técnicas de evaluación (cómo se va a evaluar).

Así mismo, MECAPS como estrategia de investigación cuenta con elementos que permiten: (i) su replicación en proyectos de formación ADSI, (ii) extensión hacia otros programas formativos, (iii) conexión con competencias transversales de otros programas formativos, y (iv) el monitoreo de su efectividad en la formación. De esta manera, MECAPS actúa como un mediador del conocimiento, adquirido a través del proceso Enseñanza-Aprendizaje liderado por los instructores, junto con el material de apoyo, y que puede ser ampliado con base en las propias experiencias de los aprendices ADSI, en escenarios reales de aplicación. Esta capacidad constructivista del aprendizaje inculcada por MECAPS es visible a través de atributos como: Identificación única de los proyectos de formación ADSI, tipos de escenarios de aplicación (de acuerdo con el tipo de programa formativo), grupo de competencias transversales involucradas en el programa formativo, calificación del nivel de satisfacción de los clientes reales (empresas del sector relacionadas con el programa formativo), entre otros.

CAPÍTULO 2

METODOLOGÍA MECAPS COMO ESTRATEGIA DE INVESTIGACIÓN

A continuación, se describe MECAPS como estrategia de investigación, para lo cual se estima el conocimiento previamente alcanzado por los aprendices e instructores ADSI durante el proceso de desarrollo de productos software, y a la vez los involucra en una dinámica de reconocimiento de los fundamentos de calidad de software. A partir de esta participación en la construcción de la estrategia para la evaluación de la calidad se obtiene: (i) mayor capacidad técnica del proceso de desarrollo ADSI, capacidad que puede ser replicada en proyectos realizados por los aprendices SENA, (ii) mayor habilidad técnica en los aprendices e instructores ADSI, de manera que pueda ser utilizada en escenarios reales de organizaciones, de las cuales hagan parte en su futuro laboral, actuando como agentes replicadores de la metodología en el contexto de la industria software.

Como estrategia de investigación, MECAPS cuenta con dos tipos de actividades: Unas actividades de construcción, que conllevan la estructuración de la metodología MECAPS y otras actividades de apoyo que robustecen las habilidades investigativas y técnicas del equipo del proyecto. A continuación, se listan las actividades que hacen parte de la estrategia de investigación de MECAPS (Tabla 1).

Tabla 1. Actividades de la estrategia de investigación

Actividades de construcción	Actividades de apoyo
A1:Modelado del proceso de desarrollo ADSI	Ap1:Investigación acerca de estándares de calidad del producto software
A2:Evaluación del proceso de desarrollo ADSI	Ap2:Investigación acerca de estándares de calidad del proceso de desarrollo software

Continúa pág. 20

Viene pág. 19

A3:Identificación de oportunidades de mejora	Ap3:Investigación acerca de mejores prácticas de aseguramiento de calidad
A4:Estructuración de la metodología	Ap4:Investigación acerca de tendencias en el aprendizaje
A5:Evaluación de la metodología	Ap5:Ejecución de mecanismos de aseguramiento de calidad sobre artefactos software generados por los aprendices ADSI

Del mismo modo, la estrategia de investigación requiere la definición de tipos de investigadores de acuerdo con su participación, los cuales se muestran en la Tabla 2.

Tabla 2. Tipos de investigadores

Tipos de investigadores	Responsabilidades
Investigadores asesores	Asesorar en la creación de estrategias de investigación, planes y acciones para la generación de habilidades I+D en instructores y aprendices durante el desarrollo del proyecto de investigación para la generación del producto y/o productos esperado(s) del semillero.
Investigadores instructores	Gestionar las actividades de investigación requeridas por el proyecto, guiados por los asesores, validar los entregables de investigación que elaboran los aprendices y apoyar en el cumplimiento de los planes establecidos.
Investigador líder SENNOVA	Realizar todas las gestiones de investigación de índole institucional para el proyecto, presentando informes de ejecución a la dirección, realizar asesoría y seguimiento a los procesos contractuales y presupuestales definidos para el proyecto durante su ejecución.
Investigadores aprendices	Operar las actividades de investigación requeridas para el proyecto, orientadas por los asesores y con el acompañamiento de los investigadores instructores. Generar los productos y entregables de investigación asociados al proyecto. Cumplir con los planes establecidos para el proyecto de investigación.

2.1 Actividades de construcción

Partiendo de la contextualización del proyecto, donde se revisa y se comparte con el equipo de trabajo los objetivos a cumplir, los entregables, así como los roles y responsabilidades dentro del proyecto, se inicia la construcción de la metodología MECAPS a través de las siguientes actividades (ver Figura 2).

2.1.1 Modelado del proceso de desarrollo ADSI

El modelado de procesos software consiste en definir una representación abstracta de la arquitectura, diseño y composición del proceso software ejecutado por una entidad [20]. Un proceso software o proceso de ingeniería de software se define como el conjunto de actividades necesarias para administrar, desarrollar y mantener aplicaciones software [21], donde dichas actividades son ejecutadas por un grupo de personas bajo una estructura organizacional y apoyadas por herramientas tecnológicas. Sin embargo, levantar o elicitar un proceso software no es una tarea trivial y requiere una estrategia adecuada para obtener el modelo de proceso de manera exitosa [21].

Durante esta fase, los investigadores de ADSI analizan, en conjunto con los investigadores asesores, la información referente al proceso actual que realiza ADSI en el desarrollo de los productos software con el propósito de definir y modelar el proceso actual de manera que sirva como referente para el proceso de mejora que se busca con la metodología MECAPS.

Esta actividad de construcción cuenta con la actividad de apoyo Ap1 (Tabla 1) a través de la cual los investigadores ADSI identifican aspectos de calidad de los productos software a partir del estudio y síntesis de estándares de calidad ISO tales como: Estándares de Verificación, Validación y revisiones IEEE1012_1028, Estándar Calidad, específicamente métricas de calidad ISO9126-2, ISO9126-3, ISO9126-4, IEEE 1061, Estándar de Pruebas de Software ISO 29119).

2.1.2 Evaluación del proceso de desarrollo ADSI

La evaluación de los procesos software tiene como objetivo detectar aspectos de un proceso software que se pueden mejorar [22], así como (i) establecer los elementos necesarios para evaluar el cumplimiento de los procesos de una entidad con respecto a un modelo de procesos de referencia, y (ii) conocer los puntos fuertes y débiles de un proceso de desarrollo para que sirvan de guía en la mejora de dicho proceso [23].

En esta fase, el equipo del proyecto de investigación participa de manera colaborativa en la comparación del proceso de desarrollo ADSI frente al área de proceso de asegura-

miento de la calidad de proceso y producto de CMMI-DEV, a través de la evaluación de las prácticas que propone este estándar de calidad de procesos software.

Esta actividad de construcción cuenta con la actividad de apoyo Ap2 (Tabla 1) a través de la cual los investigadores ADSI identifican aspectos de aseguramiento de calidad de proceso y de producto, a partir de la revisión de las prácticas de CMMI-DEV.

2.1.3 Identificación de oportunidades de mejora del proceso de desarrollo ADSI

La mejora de procesos de desarrollo de software consiste en aplicar buenas prácticas que proporcionen mejores resultados al momento de producir software a la medida, estas prácticas consiguen que los involucrados en el proceso se sientan comprometidos con el proyecto, con el fin de establecer parámetros de seguimiento y autocontrol que permitan cambiar o eliminar las prácticas que generan problemas en el proceso. En [1] se presenta una caracterización de las necesidades observadas en las pymes para implementar algún tipo de mejora de procesos que sea exitosa; en el trabajo descrito en [2] se expone que en la comunidad de Ingeniería de Software hay un interés creciente en mejora de procesos, por ejemplo, la mayoría de las investigaciones realizadas surgen en el sector educativo. El modelo de procesos y la metodología más utilizada es CMMI y Xtreme Programing, respectivamente. El estándar más empleado es el ISO/IEC 15504 y el proceso de soporte para el software en su ciclo de vida mayormente abordado es SQA. Lo anterior sugiere que el aplicar mejoras de procesos presenta una reducción en los costos de producción, mejora la calidad del producto y se ajusta a las necesidades de los clientes, con ello se obtiene una reducción de la tasa de rotación del personal, una mejora en las relaciones de trabajo con los clientes y asertividad en la credibilidad de la empresa.

Esta fase tiene como propósito revelar las oportunidades de mejora que se pueden aplicar al proceso de desarrollo de software de ADSI, a partir de los resultados obtenidos previamente en la fase de evaluación.

Esta actividad de construcción cuenta con la actividad de apoyo Ap3 (Tabla 1) a través de la cual los investigadores ADSI estudian las mejores prácticas relacionadas con el aseguramiento de calidad, de manera que se identifican posibles prácticas de pruebas que pudieran ser incluidas en el proceso ADSI.

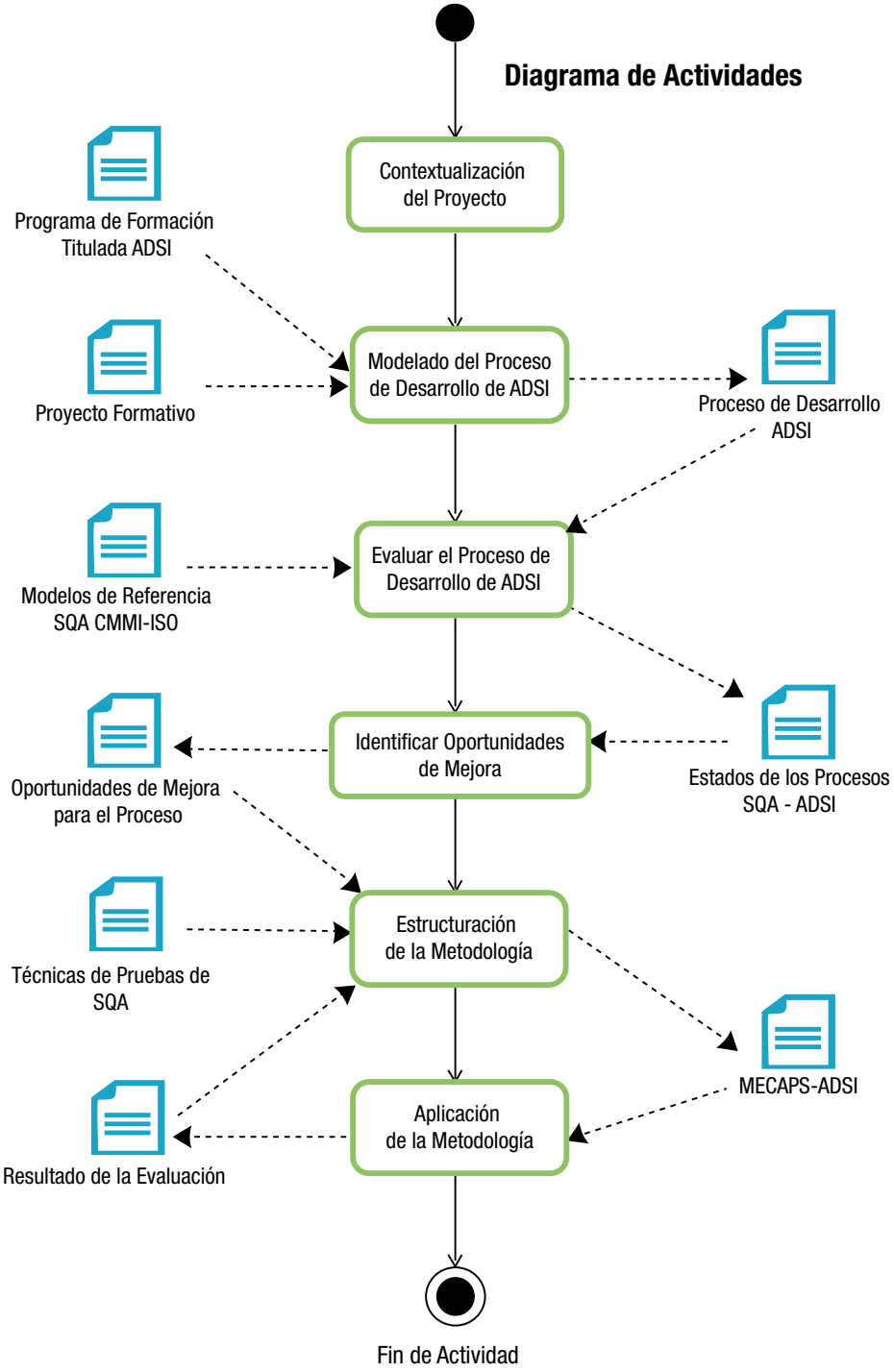


Figura 2. Estrategia de investigación

CAPÍTULO 3

METODOLOGÍA PARA LA EVALUACIÓN DE LA CALIDAD DE LOS PRODUCTOS SOFTWARE DESARROLLADOS POR LOS APRENDICES ADSI

En este capítulo se describe la metodología MECAPS, uno de los productos generados durante la ejecución del proyecto de investigación en el marco SENNOVA. En primer lugar, se encuentra la descripción del proceso formativo del tecnólogo en Análisis y Desarrollo de Sistemas de Información ADSI a través del modelamiento de las fases de ingeniería de software, las actividades que se cumplen y los productos que se obtienen en cada fase. Posteriormente, se describe de forma general la evaluación del proceso de desarrollo de los productos software generados por los aprendices ADSI en el contexto de su formación como tecnólogos y las oportunidades de mejora encontradas. En este capítulo y en un sentido fundamental, se muestra la estructura de la Metodología para la evaluación de la calidad de los productos software desarrolladoras por los aprendices ADSI.

3.1 Modelado del proceso de desarrollo de ADSI

Para poder determinar qué aspectos de calidad de proceso y producto software son considerados por los aprendices ADSI durante la ejecución del proyecto formativo, fue necesario modelar el proceso de desarrollo que los aprendices siguen durante la realización de proyectos. Este proceso emerge a partir de un análisis de la relación entre el programa formativo de ASDI y las actividades de desarrollo software que lo componen, como se describe a continuación en la Figura 3 y la Tabla 3.

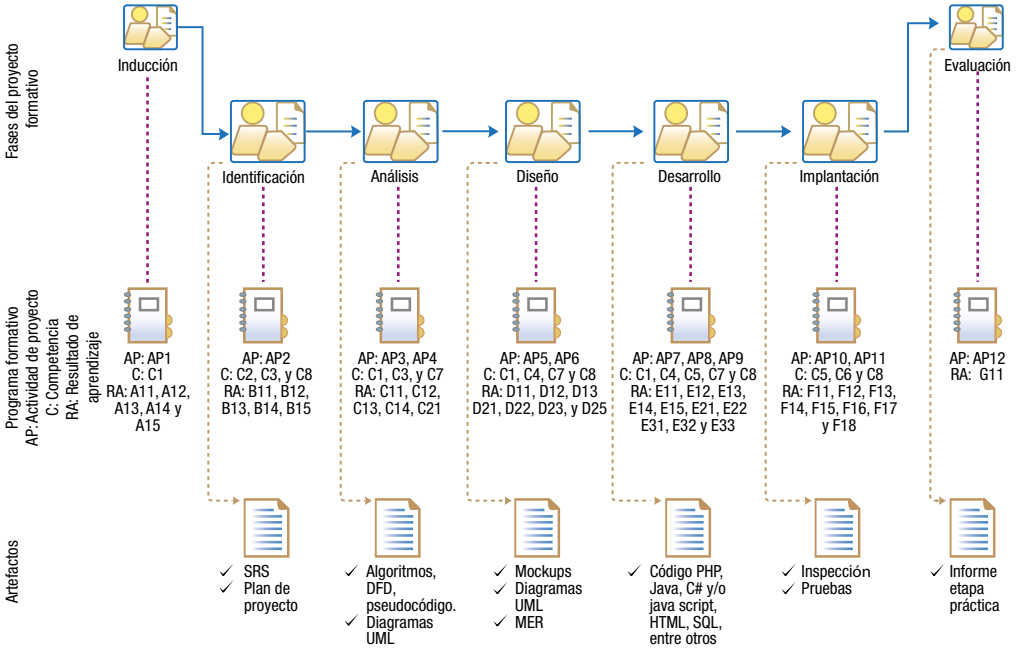


Figura 3. Modelo del proceso de desarrollo de ADSI

Tabla 3. Competencias del proyecto formativo.

COMPETENCIAS	
Código	**Descripción**
C1	Promover la interacción idónea consigo mismo, con los demás y con la naturaleza en los contextos laboral y social.
C2	Especificar los requisitos necesarios para desarrollar el sistema de información de acuerdo con las necesidades del cliente.
C3	Analizar los requisitos del cliente para construir el sistema de información.
C4	Diseñar el sistema según los requisitos del cliente.
C5	Construir el sistema que cumpla con los requisitos de la solución informática.
C6	Implantar la solución que cumpla con los requisitos para su operación.
C7	Participar en el proceso de negociación de tecnología informática para permitir la implementación del sistema de información.
C8	Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, de acuerdo con el referente adoptado en la empresa.

Continúa pág. 27

Viene pág. 26

Actividad de proyecto	
Código	**Descripción**
AP1	Identificar los lineamientos institucionales relacionados con la formación profesional integral y presentación del Proyecto Formativo
Competencias asociadas a todos los RA de AP1: C1-Promover la interacción idónea consigo mismo, con los demás y con la naturaleza en los contextos laboral y social.	
Resultados de aprendizajes asociados a AP1	
Código	**Descripción**
A11	Reconocer el rol de los participantes en el proceso formativo, el papel de los ambientes de aprendizaje y la metodología de formación, de acuerdo con la dinámica organizacional del SENA.
A12	Asumir los deberes y derechos con base en las leyes y la normativa institucional en el marco de su proyecto de vida.
A13	Gestionar la información de acuerdo con los procedimientos establecidos y con las tecnologías de la información y la comunicación disponibles.
A14	Identificar las oportunidades que ofrece el Sena en el marco de la formación profesional según el contexto nacional e internacional.
A15	Concertar alternativas y acciones de formación para el desarrollo de las competencias del programa de formación, con base en la política institucional.
Actividad de proyecto	
Código	**Descripción**
AP2	Determinar las especificaciones funcionales del sistema de información.
Competencias asociadas a AP2: C2-Especificar los requisitos necesarios para desarrollar el sistema de información, de acuerdo con las necesidades del cliente. C3- Analizar los requisitos del cliente para construir el sistema de Información. C8- Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, según el referente adoptado en la empresa.	

Resultados de aprendizajes asociados a AP2		
Código	**Descripción**	**Competencia asociada**
B11	Aplicar las técnicas de recolección de datos, diseñando los instrumentos necesarios para el procesamiento de información, de acuerdo con la situación planteada por la empresa.	C2

Continúa pág. 28

Viene pág. 27

B12	Elaborar mapas de procesos que permitan identificar las áreas involucradas en un sistema de información, utilizando herramientas informáticas y las TIC, para generar informes según las necesidades de la empresa.	C2
B14	Plantear diferentes alternativas de modelos tecnológicos de información empresarial, teniendo en cuenta la plataforma tecnológica de la empresa y las tendencias del mercado, para dar solución a las situaciones relacionadas con el manejo de la información de la organización.	
B13	Interpretar el informe de requerimientos, para determinar las necesidades tecnológicas en el manejo de la información, según las normas y protocolos establecidos en la empresa.	C3
B15	Identificar las características de los procesos de desarrollo de software, frente al referente de calidad adoptado por la empresa, ajustándolos a los resultados de las mediciones, evaluaciones y recomendaciones realizadas.	C8
Actividad de proyecto		
Código	**Descripción**	
AP3	Analizar los procesos y datos del Sistema de Información.	
Competencias asociadas a AP3: C3- Analizar los requisitos del cliente para construir el sistema de Información. C7-Participar en el proceso de negociación de tecnología informática para permitir la implementación del sistema de información.		
Resultados de aprendizajes asociados a AP3		
Código	**Descripción**	**Competencia asociada**
C11	Representa el bosquejo de la solución al problema presentado por el cliente, mediante la elaboración de diagramas de casos de uso, apoyado en el análisis del informe de requerimientos, al confrontar la situación problemática con el usuario según normas y protocolos de la organización.	C3
C12	Valorar la incidencia de los datos en los procesos del macrosistema, tomando como referente el diccionario de datos y las mini especificaciones, para la consolidación de los datos que intervienen, de acuerdo con los parámetros establecidos.	
C13	Elaborar el informe de los resultados del análisis del sistema de información, según los requerimientos del cliente sobre normas y protocolos establecidos.	C3

Continúa pág. 29

Viene pág. 28

C14	Interpretar el diagnóstico de necesidades informáticas, para determinar las tecnológicas requeridas en el manejo de la información, de acuerdo con las normas y protocolos establecidos por la empresa.	C7
Actividad de proyecto		
Código	**Descripción**	
AP4	Especificar el modelo conceptual del Sistema de Información.	
Competencias asociadas a AP4: C3- Analizar los requisitos del cliente para construir el sistema de Información.		
Resultados de aprendizajes asociados a AP4		
Código	**Descripción**	**Competencia asociada**
C21	Construir el modelo conceptual del macrosistema frente a los requerimientos del cliente, mediante el uso e interpretación de la información levantada, representado en diagramas de clase, de interacción, colaboración y contratos de operación, de acuerdo con las diferentes secuencias, fases y procedimientos del sistema.	C3
Actividad de proyecto		
Código	**Descripción**	
AP5	Diseñar la Estructura Tecnológica del Sistema de Información	
Competencias asociadas a AP5: C4- Diseñar el sistema de acuerdo con los requisitos del cliente. C7-Participar en el proceso de negociación de tecnología informática para permitir la implementación del sistema de información. C8- Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, según el referente adoptado en la empresa.		
Resultados de aprendizajes asociados a AP5		
Código	**Descripción**	**Competencia asociada**
D11	Diseñar la estructura de datos, a partir del modelo conceptual determinado en el análisis del sistema, utilizando herramientas tecnológicas de bases de datos, según las normas y estándares establecidos.	C4

Continúa pág. 30

Viene pág. 29

D12	Construir el prototipo del sistema de información, a partir del análisis de las características funcionales del sistema en relación con facilidad de manejo, funcionalidad y experiencia del usuario, apoyado en software aplicado según protocolos de diseño.	C4
D13	Definir estrategias para la elaboración de términos de referencia y procesos de evaluación de proveedores, en la adquisición de tecnología, según protocolos establecidos.	C7

Actividad de proyecto	
Código	**Descripción**
AP6	Diseñar la estructura tecnológica del Sistema de Información.

Competencias asociadas a AP6:
C4- Diseñar el sistema de acuerdo con los requisitos del cliente.
C7- Participar en el proceso de negociación de tecnología informática para permitir la implementación del sistema de información.
C8- Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, según el referente adoptado en la empresa.

Resultados de aprendizajes asociados a AP6		
Código	**Descripción**	**Competencia asociada**
D21	Elaborar el informe de diseño del sistema de información, de acuerdo con la selección de las herramientas, tanto de software como de hardware, requeridas para la solución informática.	C4
D22	Diseñar la arquitectura del software, mediante la interpretación de las clases, objetos y mecanismos de colaboración, utilizando herramientas tecnológicas de diseño, de acuerdo con las tendencias de las tecnologías de la información y la comunicación.	
D23	Diseñar la arquitectura tecnológica del sistema de información, mediante el reconocimiento de hardware y software, según la tecnología disponible en el mercado, el informe de análisis levantado y el diagrama de distribución.	
D24	Participar en los perfeccionamientos de contratos informáticos, estableciendo cláusulas técnicas que respondan a las necesidades de los actores de la negociación, de acuerdo con la ley de contratación.	C7
D25	Identificar los puntos críticos de control en los procesos de desarrollo de software, para establecer las acciones a seguir, garantizando el cumplimiento de los estándares de calidad, siguiendo los lineamientos establecidos por la organización.	C8

Continúa pág. 31

Viene pág. 30

Actividad de proyecto	
Código	**Descripción**
AP7	Desarrollar la estructura de datos y la interfaz de usuario del Sistema de Información.
Competencias asociadas a AP7: C4- Diseñar el sistema de acuerdo con los requisitos del cliente. C5- Construir el sistema que cumpla con los requisitos de la solución informática. C8- Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, según el referente adoptado en la empresa.	

Resultados de aprendizajes asociados a AP7		
Código	**Descripción**	**Competencia asociada**
E11	Interpretar el informe técnico de diseño, para determinar el plan de trabajo durante la fase de construcción del software, de acuerdo con las normas y protocolos establecidos en la empresa.	C5
E12	Construir la base de datos, a partir del modelo de datos determinado en el diseño del sistema, utilizando sistemas de gestión de base de datos, según los protocolos establecidos en la organización.	
E13	Construir la interfaz de usuario, apoyado en la evaluación del prototipo, determinando las entradas y salidas requeridas en el diseño y definiendo los lineamientos para la navegación, de acuerdo con las necesidades del usuario.	
E14	Aplicar políticas y mecanismos de control en el diseño del sistema de información, mediante el análisis de la vulnerabilidad de la información, siguiendo los parámetros establecidos por la organización.	C4
E15	Aplicar los estándares de calidad involucrados en los procesos de desarrollo de software, siguiendo el plan establecido para mantener la integridad de los productos de trabajo definidos, según las prácticas de configuración establecidas por la empresa.	C8

Actividad de proyecto	
Código	**Descripción**
AP8	Codificar los módulos del Sistema de Información.
Competencias asociadas a AP8: C5- Construir el sistema que cumpla con los requisitos de la solución informática.	

Continúa pág. 32

Viene pág. 31

Resultados de aprendizajes asociados a AP8		
Código	**Descripción**	**Competencia asociada**
E21	Realizar la codificación de los módulos del sistema y el programa principal, a partir de la utilización del lenguaje de programación seleccionado, de acuerdo con las especificaciones del diseño.	C5
E22	Construir el programa de instalación del aplicativo, utilizando las herramientas de desarrollo disponibles en el mercado, según las características de la arquitectura de la solución.	
Actividad de proyecto		
Código	**Descripción**	
AP9	Determinar el cumplimiento de las buenas prácticas de calidad en el desarrollo de Software.	
Competencias asociadas a AP9: C7- Participar en el proceso de negociación de tecnología informática para permitir la implementación del sistema de información. C8- Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, según el referente adoptado en la empresa.		
Resultados de aprendizajes asociados a AP9		
E31	Elaborar el informe sobre el cumplimiento de los términos de referencia previstos en la negociación, de acuerdo con la participación de cada uno de los actores en relación con la satisfacción de los bienes informáticos contratados y recibidos, según normas y protocolos de la organización.	C7
E32	Elaborar instrumentos e instructivos, requeridos por el aseguramiento de la calidad, para documentar y evaluar los procesos de desarrollo de software, de acuerdo con las normas y procedimientos establecidos por la empresa.	C8
E33	Evaluar procesos y productos de desarrollo de software, documentar y concertar acciones a seguir, para garantizar el cumplimiento de las normas establecidas, de acuerdo con el plan definido y con los criterios de medición, métricas y políticas determinados por la empresa.	C8

Continúa pág. 33

Viene pág. 32

Actividad de proyecto	
Código	**Descripción**
AP10	Desarrollar las tareas de configuración y puesta en marcha del Sistema de Información.

Competencias asociadas a AP6:
C5- Construir el sistema que cumpla con los requisitos de la solución informática.
C6- Implantar la solución que cumpla con los requisitos para su operación.
C8- Aplicar buenas prácticas de calidad en el proceso de desarrollo de software, de acuerdo con el referente adoptado en la empresa.

Resultados de aprendizajes asociados a AP10		
Código	**Descripción**	**Competencia asociada**
F11	Elaborar el informe final del proceso de gestión de calidad en el desarrollo de software, que consolide la información de las evidencias, hallazgos y novedades frente al seguimiento y control de los productos, según normas internacionales y protocolos de la organización.	C8
F12	Ejecutar y documentar las pruebas del software, aplicando técnicas de ensayo-error, de acuerdo con el plan diseñado y los procedimientos establecidos por la empresa.	C5
F13	Elaborar el manual técnico de la aplicación, de acuerdo con la complejidad del aplicativo y según normas y procedimientos establecidos por la empresa.	C5
F14	Configurar el software de la aplicación para cliente y servidor, mediante la utilización del hardware disponible, ejecutándola en la plataforma tecnológica, según normas y protocolos establecidos por la empresa.	C6
F15	Capacitar a los usuarios del sistema, sobre la estructuración y el manejo del aplicativo, de acuerdo con el plan establecido, y el perfil de los usuarios, según políticas de la organización.	C6
F16	Definir estrategias para la validación de manuales de usuario y de operación, respondiendo a las necesidades y satisfacción del cliente, frente a la solución informática propuesta, según políticas de la organización.	C6
F17	Elaborar informes técnicos relacionados con la solución informática implantada, de acuerdo con las propuestas de alternativas aplicadas, teniendo en cuenta las técnicas de comunicación y según normas y protocolos establecidos.	C6

Continúa pág. 34

Viene pág. 33

F18	Elaborar el informe administrativo, siguiendo los protocolos de la organización, basado en los planes de instalación, respaldo y migración del sistema de información, facilitando la operatividad y mantenimiento de la solución informática.	C6

3.2 Evaluación del proceso de desarrollo de ADSI

Para determinar la idoneidad de los aspectos de calidad de proceso y producto considerados por el proceso de desarrollo utilizado por los aprendices ADSI, fue necesario evaluar dicho proceso con respecto a un referente internacional. En este sentido, se realizó una evaluación del modelo de proceso de desarrollo de software definido previamente con el área de proceso de "Aseguramiento de calidad de proceso y producto" del modelo internacional CMMI-DEV versión 1.3, el cual es el modelo de calidad más usado y conocido por la industria del software mundial. El objetivo de esta área de proceso es proporcionar al personal y a la gerencia una visión objetiva de los procesos y de los productos de trabajo asociados.

3.3 Identificación de oportunidades de mejora

La información brindada por el grupo ADSI a partir de los resultados de la enseñanza del proceso de desarrollo de software, permitió identificar y analizar los siguientes rasgos de calidad:

1. Calidad en los productos software
2. Calidad de los artefactos del proceso de desarrollo
3. Calidad en la gestión del proceso de desarrollo como proyecto
4. Características que permitan evaluar los resultados del aprendizaje
5. Características que permitan evaluar la satisfacción de los clientes de ADSI

La metodología a construir deberá proponer mecanismos que permitan cubrir los puntos de mejora para cada uno de estos rasgos.

3.4 Estructuración de la Metodología

Dentro del ciclo de desarrollo de software existen tres fases primordiales para obtener un producto software: Análisis, Diseño y Construcción, a través de las cuales se identifica la necesidad del cliente, se analiza y diseña la solución más viable y adecuada que posteriormente se implementa. En este contexto, la metodología MECAPS define cuatro componentes de evaluación de la calidad de los productos y proyectos software desarrollados por ADSI, los cuales incluyen mecanismos de aseguramiento de calidad de manera que cada una de las iniciativas de desarrollo que surjan en el SENA utilicen estos mecanismos durante su proceso de formación, es decir en cada una de estas etapas

del ciclo de desarrollo (Figura 4). El uso adecuado de estos mecanismos, aseguran que el proceso de ADSI logre productos de software de calidad.

Desde el punto de vista de la gestión del proyecto de desarrollo de software MECAPS interviene el proceso ADSI re-estructurando el contenido de las especificaciones SRS, de manera que se logre planear previamente la gestión del proyecto en términos de tiempo, riesgos y restricciones, haciendo uso de un plan básico de gestión para que pueda ser monitoreada y controlada su posterior ejecución. El objetivo del plan básico de gestión del proyecto permitirá a los aprendices ADSI fortalecer las habilidades de gestión requeridas en un contexto laboral en el cual podrán participar de manera efectiva en los procesos de planeación de las organizaciones. Este plan de gestión del proyecto asegura la calidad del proyecto en cuanto al cumplimiento de los objetivos estratégicos del mismo, para lo cual el desarrollo del producto software será su principal evidencia de cumplimiento y apoyo a los objetivos de la organización.

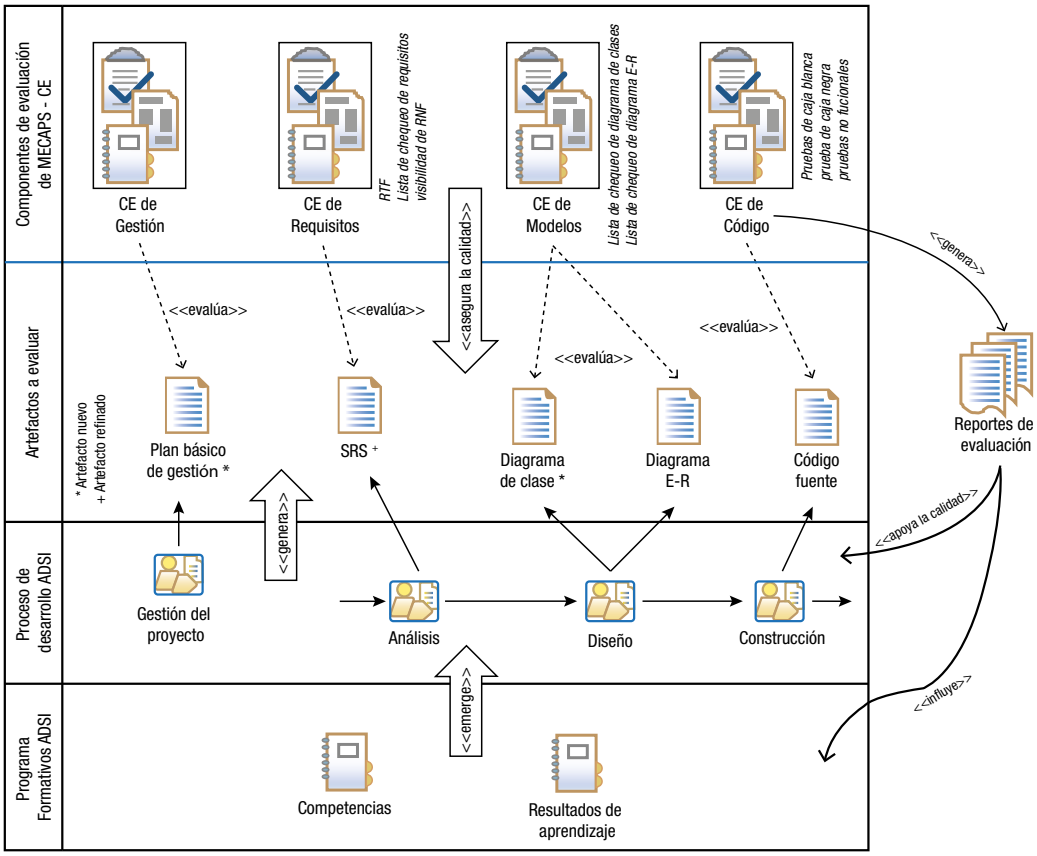


Figura 4. Estructura de MECAPS

3.4.1 Objetivo de la metodología

Guiar el proceso de desarrollo de software del grupo ADSI del SENA Popayán frente a aspectos relacionados con el aseguramiento de la calidad de los productos software y de la calidad de su gestión.

En la Figura 4 podemos observar los elementos de aseguramiento de calidad (SQA) que MECAPS propone para cada una de las etapas de ciclo de desarrollo. En la etapa de Análisis, donde el objetivo es identificar y plasmar la necesidad del cliente en términos funcionales y no funcionales del futuro producto software, MECAPS incluye los siguientes elementos de SQA en el componente de evaluación de requisitos (CE de Requisitos):

- Revisión Técnica Formal
- Lista de chequeo de requisitos
- Visibilidad de los requisitos no funcionales
- Validación de los requisitos no funcionales

Aplicados sobre la especificación del sistema (Formato SRS de ADSI).

Continuando con el ciclo de desarrollo, en su etapa de diseño de la solución, donde el objetivo del proceso es plasmar la solución a implementar con el producto software, MECAPS incluye los siguientes elementos de SQA en el componente de evaluación de modelos (CE de modelos):

- Guía para el proceso de diseño de base de datos
- Lista de chequeo del diagrama de clases
- Lista de chequeo del modelo E-R
- Lista de chequeo del modelo relacional

Aplicados sobre el proceso de diseño de base de datos, sobre los modelos de datos y el diagrama de clases.

En la etapa de construcción del producto software, MECAPS aporta con la inclusión de los siguientes elementos de SQA en el componente de evaluación de código (CE de código):

- Pruebas de Caja Blanca
- Pruebas de Caja Negra
- Pruebas unitarias
- Pruebas integrales
- Pruebas de sistema

Aplicados sobre el código del sistema, sobre la funcionalidad y la no funcionalidad implementadas, software, interfaces y su entorno.

Un proceso de desarrollo de software, debe ser visto además como un proyecto, puesto que implica un esfuerzo temporal para lograr un objetivo específico con restricciones de tiempo, costo y otros recursos. Es importante realizar de alguna manera mínima una gestión del proyecto para poder trabajar en asegurar la calidad de esta gestión. La gestión del proyecto debe incluir aspectos como: (i) el tiempo del proyecto, (ii) el alcance, (iii) la calidad, (iv) los recursos humanos, (v) los recursos financieros, (vi) la comunicación, (vii) los riesgos del proyecto, y (viii) las compras/contrataciones (ver ítem 3.5.6). Para esto, MECAPS propone un plan mínimo de gestión de proyectos, para asegurar que no sólo se logre un producto de software de calidad sino también una gestión del proceso de desarrollo adecuado de manera que se logre una satisfacción del cliente desde esta perspectiva.

3.5 Descripción de los mecanismos de SQA

3.5.1 Mecanismos de SQA en etapa de especificación

Frente a la especificación, artefacto principal que se obtiene en la etapa de elicitación de requisitos, el aseguramiento de calidad puede ser realizado a través de mecanismos de control tales como:

3.5.1.1 Revisión Técnica Formal (RTF) [24]

Es una técnica que dirige la atención a un producto software específico, con alcance limitado, por ejemplo: una parte del modelo de requerimientos, el diseño detallado de un componente del software, o un fragmento de código. Tiene como objetivos: (1) Descubrir errores en la función, la lógica o la implementación de cualquier representación del software, (2) Verificar que el software bajo revisión alcanza sus requisitos; (3) Garantizar que el software ha sido representado de acuerdo con ciertos estándares predefinidos; (4) Conseguir un software desarrollado de forma uniforme (5) Hacer que los proyectos sean más manejables.

La revisión técnica formal se realiza con una reunión, la cual debe planearse, controlarse y ejecutarse. Esta reunión debe ser convocada entre tres y cinco personas, su duración no requerirá más de dos horas de trabajo de cada persona. Al finalizar la revisión, todos los participantes deciden si se acepta el producto evaluado sin posteriores modificaciones, si lo aceptan provisionalmente, es decir con problemas menores que de igual manera deben ser corregidos, pero sin necesidad de una nueva revisión, o si lo rechazan, debido a los serios errores encontrados, los cuales una vez corregidos generan una nueva revisión. Luego de la decisión grupal, se culmina el proceso con las firmas de los participantes en representación de acuerdo con lo plasmado en los resultados.

Este mecanismo RTF requiere básicamente dos documentos que permiten resumir las no conformidades encontradas durante la evaluación del producto/artefacto software.

El primer documento es el informe que consolida los resultados o informe sumario de la revisión técnica formal, y responde a tres preguntas: 1. ¿Qué fue revisado? 2. ¿Quién lo revisó? 3. ¿Qué se descubrió y cuáles son las conclusiones? El segundo documento, es la lista de sucesos de revisión, el cual registra información acerca de las áreas problemáticas dentro del producto y las respectivas recomendaciones de mejora, para que el responsable del producto/artefacto las tenga presentes al momento de realizar las correcciones.

Las plantillas que se han definido en MECAPS para el aseguramiento de calidad a través de la técnica RTF se presentan a continuación (Plantilla 1):

Plantilla 1. Revisión técnica formal (RTF)

Nombre del proyecto: Informe de Revisión Técnica Formal (RTF)
Versión [1.0]

[Este documento es la plantilla base para elaborar el documento Informe de Revisión técnica formal (RTF). Los textos que aparecen entre paréntesis rectos son explicaciones de que debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda]

Historia de revisiones

Fecha	Versión	Descripción	Autor
[dd/mm/aaaa]	[x.x]	[detalles]	[nombre]

1. Producto revisado

1.1. Nombre y versión del producto revisado
[Se debe especificar el nombre, la versión y el área a la que corresponde el producto revisado]

1.2. Participantes de la revisión
[Nombre y rol de los participantes de la RTF]

1.3. Técnica utilizada
[Forma en que se hizo la revisión, por ej. Lista de chequeo definida en el plan de SQA para la revisión de productos, entradas, salidas y descripción de actividades]

Continúa pág. 39

Viene pág. 38

2. Objetivos de la RTF

[Se plantean brevemente los aspectos del producto que serán revisados, qué propiedades de la calidad se buscará que cumpla y en qué grado, qué principios y estándares de calidad aplican al producto, si ya hubo revisión de otra versión del producto qué correcciones quedaron pendientes de realizar. De la misma forma se establecen los aspectos del proceso -en lo que tiene que ver con la elaboración del producto- que serán revisados con los participantes de la RTF. Esta información es la misma que se incluyó en la convocatoria para la realización de la RTF enviada a los participantes durante la etapa de control de la sesión de revisión]

3. Problemas detectados

[Se describen en forma detallada los problemas detectados, tanto en el producto como en el proceso de elaboración del mismo, y se plantean las sugerencias de corrección para la próxima versión del producto]

3.1. [Problema detectado 1]

[Se describe el problema detectado, estableciendo sus características, gravedad y especificando su ubicación]

3.1.1. Sugerencia de corrección

[Se sugieren las correcciones a realizar para que el producto cumpla con los principios y estándares de calidad establecidos y los procedimientos definidos]

3.2. [Problema detectado 2]

4. Evaluación

[Se realiza una evaluación global del producto revisado, de acuerdo con los objetivos planteados y los problemas detectados]

4.1. Estado actual del Producto

[Se describe el estado actual del producto, por ej. se debe rehacer o corregir, se puede entregar al cliente]

4.2. Acciones a tomar

[Se describen las recomendaciones y/o acciones a seguir, generales y concluyentes, de la evaluación del producto]

4.3. Próxima revisión del producto

[Se establece la próxima revisión de acuerdo con los puntos detallados anteriormente, en términos de fechas o fase e iteración]

Continúa pág. 40

Viene pág. 39

Revisores

Revisor	Firma	Fecha
[nombre]	[Firma]	[dd/mm/aaaa]

3.5.1.2 Lista de chequeo de requisitos [25]

Esta técnica es un tipo de lista de comprobación, como mecanismo primario de control de la calidad de las especificaciones del software. Su objetivo es encontrar defectos en la definición de las necesidades del usuario para el sistema o componente, una mala descripción de los requisitos de usuario, funcionalidades incorrectas o incompatibles, interfaces de usuario no claras, elementos de software y/o hardware incorrectos en los puntos de interacción del sistema con el entorno. Basado en el modelo de calidad internacional ISO 25010 donde se establecen las características de calidad que se evalúan en un producto software, se construye la plantilla para el aseguramiento de calidad de los requisitos del sistema *(Plantilla 2)*.

Plantilla 2. Lista de chequeo de requisitos de sistemas

Evaluación de requisitos del sistema frente a las características de calidad del producto		
Funcionalidad		Respuesta
1	¿Se han especificado todas las tareas que debe realizar el sistema/software?	
2	Para cada tarea especificada, ¿se ha detallado el contenido de datos/información utilizado por la tarea	
3	¿Se ha especificado el contenido de datos/información que se obtendrá como resultado de la misma?	
4	¿El sistema de información cuenta con componentes que actúen como "compresores", es decir, proyectados para recibir más información de la que transmiten? (Compresión)	
5	¿Se ofrecen las herramientas necesarias para añadir, borrar, mantener, exhibir, imprimir, buscar y actualizar datos?	
6	¿Se cumple con la función de procesamiento, que implica la modificación de la base de datos para mantenerla actualizada? (Procesamiento)	

Continúa pág. 41

Viene pág. 40

7	¿Permite la recuperación de datos en tiempo real? (Recuperación)	
8	¿Se han especificado las pre-condiciones para cada una de las funcionalidades del sistema/software?	
Eficiencia/Desempeño		Respuesta
9	¿El sistema tiene configurado un tiempo máximo de sesión de usuario?	
10	¿Se ha especificado el tiempo de respuesta esperado de todas las operaciones especificadas?	
11	¿Se han especificado otras consideraciones temporales, tales como el tiempo de procesamiento, el de transferencia de datos o la tasa de transferencia?	
12	¿Se ha establecido el tiempo promedio de movimiento entre pantallas?	
13	¿Se ha establecido el tiempo máximo de movimiento entre pantallas?	
14	¿Se ha establecido, para el uso de la memoria del servidor, un porcentaje que no exceda el uso de la memoria disponible?	
15	¿Se ha especificado la fiabilidad del sistema/software, incluyendo las consecuencias en el caso de que falle?	
Portabilidad		Respuesta
16	¿Se ha definido la adaptabilidad del sistema ya sea hardware, software, operaciones o de uso?	
17	¿Se ha especificado qué tipo de servidor se va a utilizar?	
18	¿Se ha especificado la capacidad del servidor?	
19	¿Se ha definido el motor de base de datos?	
20	¿Se ha definido la forma para ser instalado o desinstalado de manera exitosa en determinado entorno?	
21	¿Se ha definido cómo ser reemplazado por otro producto software teniendo en cuenta el mismo propósito y entorno del software?	
22	¿Se ha definido el sistema operativo que se va a utilizar? *móvil	
23	¿Se ha definido la versión del sistema operativo? *móvil	

Continúa pág. 42

Viene pág. 41

Fiabilidad		Respuesta
24	¿Se especificaron las consecuencias por causa de fallas en la implementación del requerimiento?	
25	¿Se definió el plan de contingencia en caso de fallas?	
26	¿Se definió la estrategia de detección y corrección de errores a causa de las fallas?	
27	¿Se identifican en los requisitos atributos relacionados con la capacidad del sistema para mantener su nivel de prestación de servicio en las condiciones establecidas durante un período?	
28	¿La capacidad del sistema para tolerar errores está especificada en los requisitos?	
29	¿La capacidad del sistema para tolerar sobrecargas en el volumen de información, de usuarios o de procesos está especificada en los requisitos?	
Seguridad		Respuesta
30	¿Se tiene prevista la capacidad de protección contra el acceso de datos e información no autorizados, ya sea accidental o deliberadamente?	
31	¿Se considera el control de accesos o modificaciones no autorizados a datos o programas de ordenador?	
32	¿Se tiene previsto demostrar la identidad de un sujeto o un recurso?	
Compatibilidad		Respuesta
33	¿Se incluye la respuesta del sistema a los cambios en el entorno operativo, las interfaces, la precisión, el rendimiento, y otras capacidades adicionales predecibles?	
34	¿Se incluye la manera en que el sistema interactuará con otros software?	
35	¿Se especifica la comunicación del sistema con otros sistemas software?	
36	¿Se determina cuáles son los recursos que el sistema compartirá con otros productos software y cómo se hará dicho proceso?	
37	¿Se especifica el ambiente sobre el cual el sistema funcionará y con cuáles ambientes no lo hará?	
38	¿Se especifican los requerimientos software y hardware sobre los cuales el sistema funcionará correctamente?	

Continúa pág. 43

Viene pág. 42

39	¿Cuántas conexiones soporta el sistema en un día sin reducir su rendimiento?	
40	¿Qué requisitos mínimos son necesarios para que la aplicación funcione correctamente?	
41	¿Qué tipo de servicio en la nube utilizarán (Paas, Iaas, Saas) y será el más adecuado?	
42	¿Se especifica cómo interactuará el sistema con el servicio en la nube que escojan?	
Usabilidad		Respuesta
43	¿Se ha especificado el concepto de usabilidad para el sistema/software?	
44	¿Se ha especificado, bajo el concepto de usabilidad, cuál será el objetivo de este dentro del sistema/software?	
45	¿Se ha especificado si existen necesidades gráficas o de interfaz particulares, en torno a usabilidad teniendo en cuenta los usuarios finales del sistema/software?	
46	¿Se ha especificado un panorama general respecto a la interfaz del sistema/software, como: colores, fuentes, logos?	
47	¿Se ha especificado si el sistema/software debe cumplir con requerimientos de accesibilidad y/o para personas con alguna discapacidad?	
48	¿Se ha especificado según el concepto anterior dichos requerimientos de accesibilidad?	
49	¿Se ha especificado claramente de qué forma se accede a las funcionalidades del sistema/software	
50	¿Se han especificado los requerimientos para la comunicación entre los componentes del sistema/software?	
51	¿Se han definido las interfaces externas, como por ejemplo usuarios o hardware?	
52	¿Se han definido las interfaces internas, como por ejemplo el software o el hardware?	
Mantenibilidad		Respuesta
53	¿El sistema tiene la capacidad, cuando hay cambios en componentes, de que exista un impacto mínimo en los demás?	

Continúa pág. 44

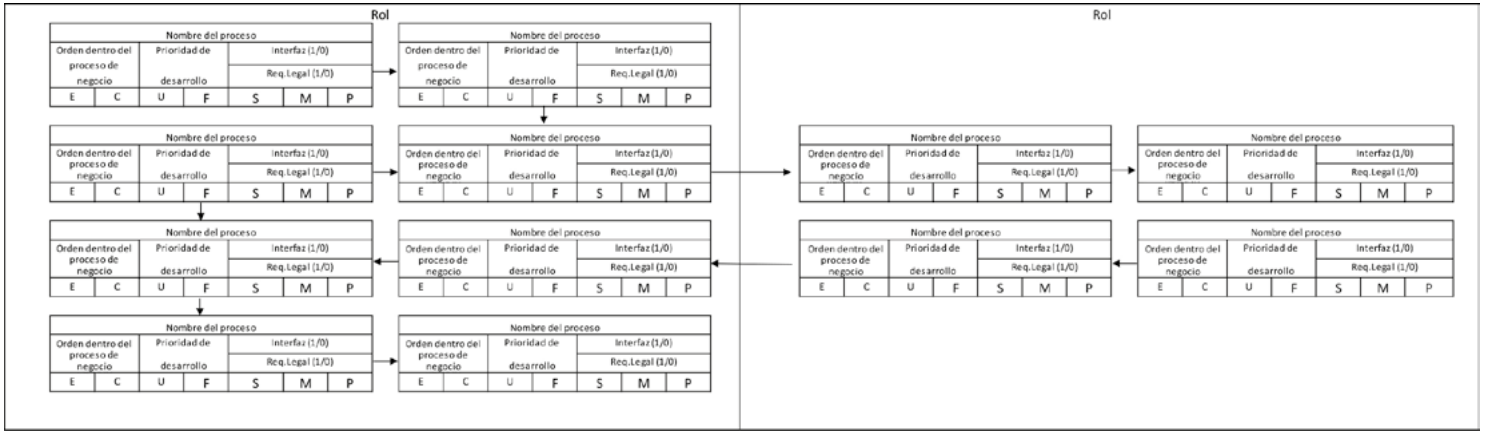
Viene pág. 43

54	¿El sistema permite que sus componentes sean utilizados en los demás elementos del mismo, o en la construcción de nuevos componentes? (Reusabilidad)	
55	¿El sistema cuenta con facilidad en el momento de evaluar el impacto con un determinado cambio sobre el software?	
56	¿El sistema cuenta con la capacidad de ser modificado sin introducir defectos o degradar el desempeño?	
57	¿El sistema cuenta con facilidad para establecer criterios de prueba?	

3.5.1.3 Visibilidad de requisitos no funcionales [25]

Los requisitos no funcionales son las restricciones de diseño y de comportamiento que debe cumplir un producto software, entre ellos están: eficiencia de desempeño, seguridad, portabilidad, mantenibilidad, compatibilidad, usabilidad y fiabilidad [1]. MECAPS sugiere el uso de la siguiente plantilla de especificación de requisitos no funcionales, para realizar el proceso de elicitación de estos requisitos, de manera que sean visibles ante el usuario final, el negocio y puedan ser tenidos en cuenta en las etapas siguientes del ciclo de desarrollo de software. Esta plantilla complementa el documento SRS que actualmente es elaborado por los aprendices ADSI (Plantilla 3) durante la etapa de análisis del producto software. Esta plantilla podrá estar acompañada de la lista de chequeo de requisitos del sistema descrita en el ítem 3.5.1.2 de este documento.

Plantilla 3. Visibilidad de requisitos no funcionales



Plantilla 4. Especificación de requisitos no funcionales

Número del proceso			n	C1 (p.e: E)	C2	C3	C4	C5	C6	C7
Interfaces			Requisitos legales	atributo 1	atributo 1	atributo 1	atributo 1	atributo 1	atributo 1	atributo 1
Int 1	*Entradas (se anexan formatos o prototipos en caso de ser necesario)*	*Salidas (se anexan formatos o prototipos en caso de ser necesario)*	Legal 1	atributo 2	atributo 2	atributo 2	atributo 2	atributo 2	atributo 2	atributo 2
M			Listar los puntos especificos de la norma o reglamentación externa que se deba aplicar (por ejemplo: punto 3.2.5 de la resolución XXXX)	atributo x	atributo x	atributo x	atributo x	atributo x	atributo x	atributo x
S										
A										
I2	*Entradas (se anexan formatos o prototipos en caso de ser necesario)*	*Salidas (se anexan formatos o prototipos en caso de ser necesario)*	Legal 2							
M			Listar los puntos especificos de la norma o reglamentación externa que se deba aplicar (por ejemplo: punto 3.2.5 de la resolución XXXX)							
S										
A										
In	*Entradas (se anexan formatos o prototipos en caso de ser necesario)*	*Salidas (se anexan formatos o prototipos en caso de ser necesario)*	Legal 3							
M			Listar los puntos especificos de la norma o reglamentación externa que se deba aplicar (por ejemplo: punto 3.2.5 de la resolución XXXX)							
S										
A										

3.5.1.4 Validación de requisitos no funcionales [25]

Luego de que se ha logrado la especificación de los aspectos no funcionales del producto software, es importante que el usuario final o cliente los valide de manera que se asegure su conformidad y entendimiento. Para este propósito, MECAPS sugiere la ejecución de un proceso de validación de requisitos no funcionales (Figura 5).

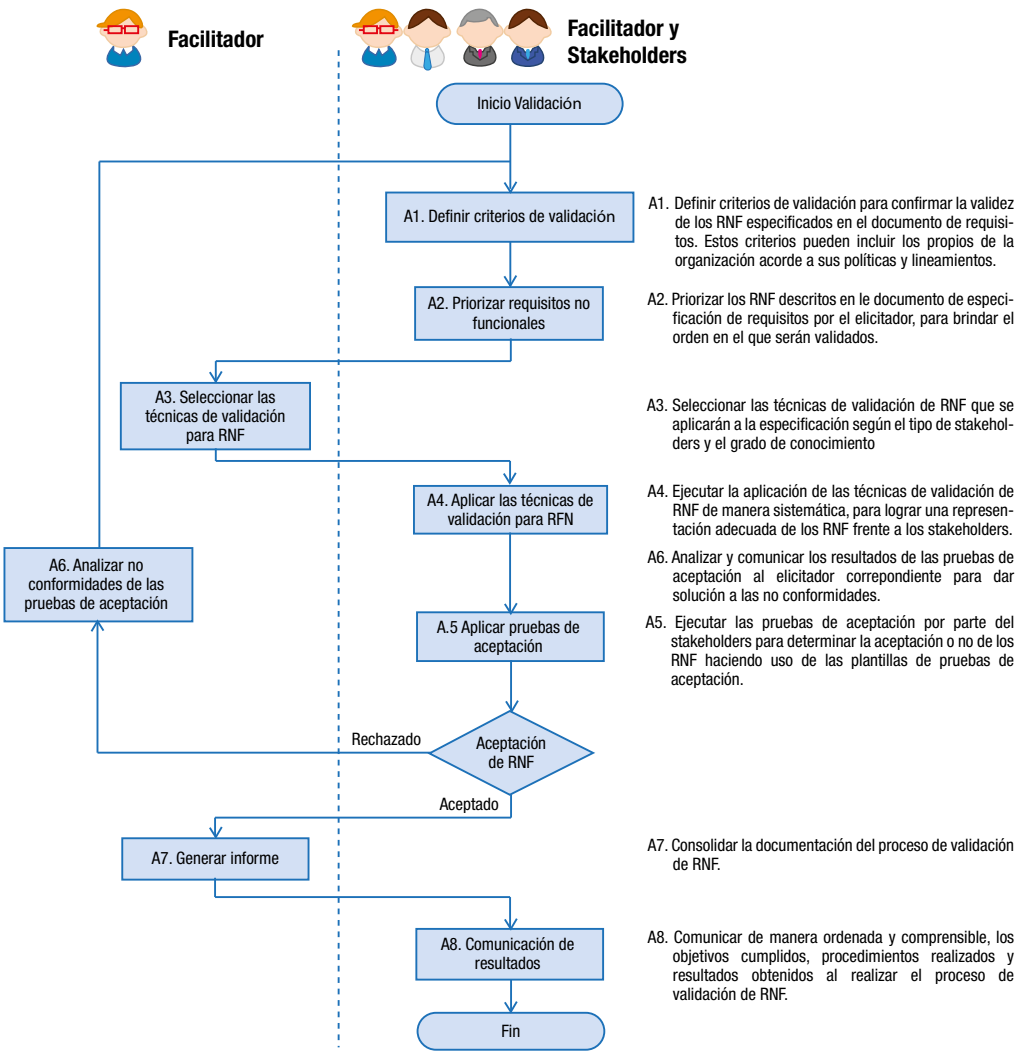


Figura 5. Proceso de validación de requisitos no funcionales

En la actividad A4: *Ejecutar la aplicación de las técnicas de validación de requisitos no funcionales* que indica el proceso para la validación de este tipo de requisitos, se sugiere el

uso de las siguientes plantillas (Plantilla 5, Plantilla 6) que permiten un entendimiento de los requisitos no funcionales por parte del usuario final del producto software. La información de escenarios que proponen las plantillas de validación puede ser construida a partir de las especificaciones de los requisitos funcionales (casos de uso, SRS, listado de funcionalidades, entre otras) permitiendo relacionar los procesos de negocio (un lenguaje entendido por los usuarios finales) y las características de calidad del producto software a construir (requisitos no funcionales).

Plantilla 5. Especificación de escenarios de negocio relacionados con RNF

CONOCIMIENTO TÓPICO		
	Escenario de negocio	
	Identificador del Escenario: Identificador único del Escenario de calidad: Ejemplo: ESC-01	
	Descripción del Escenario: Descripción del escenario de calidad.	
	RNF Relacionados: Identificador(es) de o los RNF que pertenecen al escenario de calidad.	
	Facilitador: Persona que se encarga de realizar y dirigir el proceso de validación de RNF.	**Histórico de cambios:** Fechas en las que se realizan ajustes al Escenario de calidad (dd/mm/aaaa)
	Respuesta al escenario	
	Simulación (ver sección 3.2.1.1)	**Respuesta (ver sección 3.2.1.1)**
	Origen del estímulo: Es la entidad (un usuario, un sistema, etc.) que produce el estímulo.	**Respuesta:** Actividad que se lleva a cabo luego de que el estímulo llegue al sistema.
	Estímulo: Es el activador (una solicitud, una operación, un ataque) del escenario.	
	Entorno: Condición en la que se encuentra el sistema al presentarse el estímulo.	**Medida:** Métrica definida para el tipo de estímulo y poder determinar un valor cuantificable.
	Artefacto: Compononte o destionario que recibe el estimulo (puede ser todo sistema, una o más partes)	

Plantilla 6. Pruebas de validación del requisito no funcional

CONOCIMIENTO TÓPICO			
	Requisito no funcional		
	Identificador del RNF: Identificador único del RNF. Ejemplo: RNF-001	**Tipo de RNF:** Característica de calidad a la que pertenece el RNF según ISO 25010 (Eficiencia de desempeño, Compatibilidad, Usabilidad, Fiabilidad, Seguridad, Mantenibilidad o Portabilidad).	**Prioridad:** Prioridad de validación del RNF (Alta, Media o Baja).
	Descripción del RNF: Descripción del RNF.		
	Respuesta al RNF (ver seccion 3.2.1.1)		
	Identificador del escenario: Identificador del escenario de calidad al cual pertenece el RNF.		
	Respuesta: Actividad que se lleva a cabo luego de que el estímulo llegue al sistema.	**Medida:** Métrica definida para el tipo de estímulo y poder determinar un valor cuantificable.	
	Facilitador: Persona que se encarga de realizar y dirigir el proceso de validación de RNF.	**Histórico de cambios:** Fechas en las que se realizan ajustes al RNF (dd/mm/aaaa)	

CONOCIMIENTO EPISÓDICO				
	Prueba de aceptación:			
	Stakeholder: Persona que realiza la prueba de aceptación del RNF.			
	Criterios de validación	**Aprobado**		**Observaciones:** Descripción detallada del porque no se aprueba el criterio de validación respecto al RNF.
	Nombre	**Si**	**No**	
	No ambigüedad:			
	Concisión:			
	Completitud:			
	Consistencia:			
	Verificabilidad:			
	Trazabilidad:			
	Factibilidad:			
	Criterios propios de la organización.			
	Estado del RNF: Estado final del RNF (Aprobado/No aprobado)			

3.5.2 Mecanismos de SQA en etapa de diseño

Entre los diagramas que soportan de manera importante la etapa de diseño del software se encuentran: (i) el diagrama de clases de diseño (DCD) el cual apoya la codificación

del producto software y el modelado de los datos, y (ii) los modelos entidad relación (MER), modelo relacional y modelo físico que pertenecen al proceso de diseño de la base de datos del sistema. El diagrama de clases de diseño es una estructura estática del sistema, que muestra las relaciones que existen entre las distintas clases, interfaces y objetos del sistema, así como la estructura interna de estos elementos (atributos, métodos, navegabilidad) centrándose en cada uno de ellos de forma independiente del tiempo, es decir de forma no dinámica [2][3]. Este diagrama de clases empezará a ser insumo para el proceso de diseño de la base de datos respondiendo a la pregunta: ¿Qué información debe ser almacenada en el sistema?

Para el proceso de diseño de la base de datos, el modelo entidad-relación es un tipo de modelo de datos basado en objetos que representa de manera conceptual los datos basándose en la información proporcionada en una vista de la organización, independiente de los detalles de implementación o físicos. El modelo relacional, en cambio, es un modelo de datos basado en registros, y representa de manera lógica los datos y sus relaciones, a partir del modelo conceptual de los datos, con el propósito de que el usuario perciba a la base de datos como una serie de tablas. Finalmente, el modelo físico representa cómo se debe implementar la base de datos adaptándose a un sistema de gestión de base de datos (SGBD) específico [4]

Teniendo en cuenta la relevancia de estos elementos para la calidad de la etapa de diseño del software, se construye una lista de chequeo que permite validar el cumplimiento o no de la construcción de los modelos conceptual y lógico de datos del producto (Plantilla 7).

Plantilla 7. Lista de chequeo para el proceso de diseño de base de datos

Guía para el proceso de diseño de una base de datos (Adaptado de [4])	
Diseño del modelo conceptual de los datos	**Chequear**
Identificar y describir datos conceptuales (entidad, relación, atributos)	
Asociar los datos conceptuales entre ellos	
Determinar los dominios de los atributos	
Determinar los atributos de claves (candidata, principal y alternativa)	
Ejecutar análisis de redundancia en relaciones	
Validar cubrimiento de transacciones requeridas por los usuarios	
Punto de Control del Modelo Conceptual de los Datos: (Si los ítems de verificación anteriores están cubiertos, continuar con el diseño lógico de los datos).	

Continúa pág. 50

Viene pág. 49

Diseño del modelo lógico de los datos	
Crear las tablas relacionales a partir de las entidades, relaciones y atributos del modelo conceptual	
Aplicar, si es requerida, la normalización de los datos	
Corroborar manualmente las transacciones especificadas en el modelo conceptual	
Documentar y verificar restricciones de integridad (valores permitidos, tipos de datos, multiplicidad, referencial)	
(Punto de control) Revisión del modelo lógico con el usuario	
Definir número y modo de gestión de las vistas de usuario requeridas	
Analizar aspectos de crecimiento de los datos a futuro para el diseño	
Punto de Control del Modelo Lógico de los Datos: (Si los ítems de verificación anteriores están cubiertos, continuar con el diseño físico de los datos)	

Frente a la evaluación de aspectos de calidad de cada uno de los artefactos de diseño descritos anteriormente, MECAPS dispone de las siguientes listas de chequeo (Plantilla 8, Plantilla 9, Plantilla 10):

Plantilla 8. Evaluación de diagramas de clases

Evaluación del diagrama de clases frente a aspectos de calidad		
Comprensibilidad		Respuesta
1	¿Se explica brevemente cada término de los diagramas?	
2	¿Se entiende el proceso de los diagramas?	
3	¿Los atributos de las clases están bien definidos?	
4	¿Los métodos están especificados correctamente?	
5	¿Los métodos y atributos tienen visible la simbología para su reconocimiento?	
6	¿El diagrama es fácil de leer?	
7	¿Los símbolos usados para el diagrama corresponden al definido en UML para este tipo?	
8	¿Según el problema a modelar, el diagrama puede mostrar dicho problema?	
9	¿El diagrama es fácil de entender?	

Continúa pág. 51

Viene pág. 50

Analizabilidad		Respuesta
10	¿En el diagrama es fácil encontrar errores?	
11	¿En el diagrama es fácil encontrar deficiencias?	
12	¿Es posible crear instancias de las clases?	
Modificabilidad		Respuesta
13	¿El diagrama de clases se puede modificar sin alterar la estructura de las demás clases?	
14	¿En el diagrama de clases se facilita la modificación?	
Nomenclatura		Respuesta
15	¿Todas las clases tienen nombre?	
16	¿Se determina si los métodos son privados, públicos o protegidos?	
17	¿La cardinalidad entre las clases está bien relacionada?	
18	¿Todas las asociaciones tienen verbo?	
19	¿Todas las asociaciones tienen cardinalidad?	
20	¿Todos los atributos tienen definidos los tipos de datos?	
21	¿Los métodos de la clase tienen definido el tipo de retorno o se especifica que son tipo void?	
22	¿La relación de herencia entre las clases está bien definida?	
23	¿La agregación por valor entre las clases es la correcta?	
24	¿La agregación por referencia entre las clases es la correcta?	
25	¿Las asociaciones entre las clases están correctas?	
26	¿Cada clase tiene su constructor y destructor?	

Plantilla 9. Evaluación del MER

Evaluación del diagrama entidad relación frente a aspectos de calidad		**Respuesta**
Comprensibilidad		
1	¿Existe un texto que explique, en lenguaje de alto nivel, el diagrama entidad relación?	
2	¿El texto explicativo es coherente con el diagrama propuesto? Objetivo: El MER diseñado debe ser coherente con el modelo relacional.	

Continúa pág. 52

Viene pág. 51

Analizabilidad		
3	¿Los atributos declarados identifican, califican, cuantifican, clasifican o expresan el estado de las entidades asociadas a ellos?	
4	¿Los atributos declarados tienen dominio?	
5	¿El texto que describe el diagrama describe y es consecuente con la cardinalidad?	
Nomenclatura		
6	¿Las relaciones entre entidades se conectan mediante un verbo (conjugado o infinitivo)? Ejemplo: Profesor -con/a-alumno en vez de profesor - evalúa - alumno.	

Plantilla 10. Evaluación del modelo relacional construido

Evaluación del modelo relacional frente a aspectos de calidad		
Legibilidad		Respuesta
1	¿El modelo visualmente es comprensible, de manera que no hay cruces entre las relaciones, superposiciones, o errores de tipografía, entre otros?	
Completitud		Respuesta
2	¿El modelo incluye totalmente lo que se quiere diseñar, que es aquello que se encuentra plasmado en los requerimientos del sistema por desarrollar?	
3	¿El modelo representa todo lo mostrado en el modelo Entidad-Relación?	
Minimalidad		Respuesta
4	¿El modelo tiene información redundante o duplicada?	
5	¿El modelo es fácil de comprender a la lectura?	
6	¿Los nombres de las relaciones y atributos son explícitos para la información que almacenan?	
Expresividad		Respuesta
7	¿El modelo representa la realidad?	
Extensibilidad		Respuesta
8	¿El modelo se descompone fácilmente en partes? (módulos, vistas)	
Integridad		Respuesta
9	Regla de llave primaria: ¿Todos los tipos de entidades tienen una llave primaria (propia, compuesta o heredada)?	

Continúa pág. 52

Viene pág. 52

10	Regla de denominación: ¿Todos las relaciones y atributos tienen nombre?	
11	Regla de tipo de dato: ¿Todos los atributos definidos tienen tipo de dato?	
12	Regla de la participación de la entidad: ¿Todos los tipos de entidad participan en al menos una relación, excepto los de una jerarquía de generalización?	
13	Regla de llave foránea: ¿Las relaciones que tienen llave foránea, dichas llaves son explicitas en su relación de procedencia?	
Consistencia		Respuesta
14	Regla de nombres de la relación: ¿Los nombres de una relación son únicos?	
15	Regla de nombre de atributo: ¿Los nombres de atributos son únicos dentro de las relaciones?	
16	Regla de nombre de atributos heredados: ¿Los nombres de los atributos de un subtipo no coinciden con los nombres de los atributos heredados?	
17	Regla de tipo de conexión relación/relación: ¿Todos los enlaces conectan dos tipos de relación? (no necesariamente distintos)	
18	Regla de conexión relación/relación: ¿Las relaciones no se conectan con otras relaciones?	
19	Regla de la llave foránea redundante: ¿Existen llaves foráneas redundantes?	

3.5.3 Mecanismos de SQA en etapa de desarrollo

El código generado durante la etapa de desarrollo del producto software debe ser validado por los mismos desarrolladores, o por pares, con el objetivo de identificar fallas en la estructura del código. Para esto, se realizan pruebas a cada unidad desarrollada (pruebas unitarias) validando las instrucciones lógicas y bloques que se ejecutarán a través de unos casos de prueba específicos, asegurando su ejecución como mínimo una vez. Las pruebas unitarias permiten conocer cuánto código se ha recorrido a partir de un caso de prueba [5] y su robustez, es decir soportando el ingreso de datos erróneos o inesperados y demostrando así la capacidad para manejar errores de manera controlada. Las pruebas unitarias centran su aplicación en lo que se denomina la "unidad de prueba", que puede ser una clase, un método, o un subsistema.

Para el diseño de estas pruebas unitarias se utilizan las técnicas de caja blanca, entre las cuales están: (i) técnica del camino básico, y (ii) técnica de bucles. MECAPS propone el uso de la técnica de camino básico como mínimo para asegurar la calidad de los códigos construidos por los aprendices ADSI (Tabla 4). El camino básico permite obtener una medida de la complejidad lógica del diseño procedimental y usarla para definir un conjunto básico de caminos de ejecución que garanticen que cada sentencia del programa

se recorre al menos una vez y sirve para representar el flujo de control lógico. Se basa en la representación del código a probar a través de círculos y arcos que indican el flujo de control [REF]. Cada círculo representa una o más sentencias. Las estructuras en forma de grafo de flujo son:

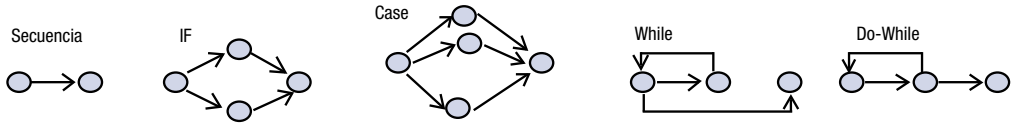
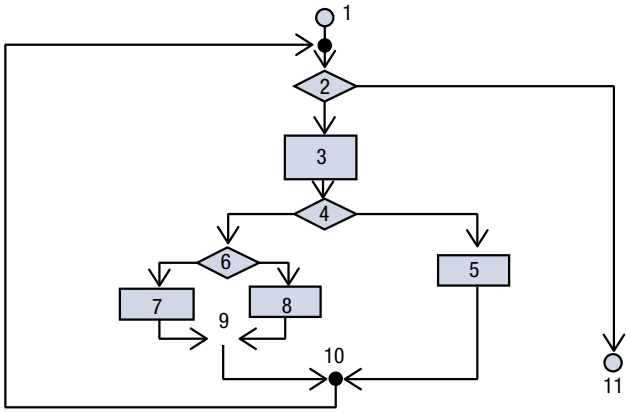
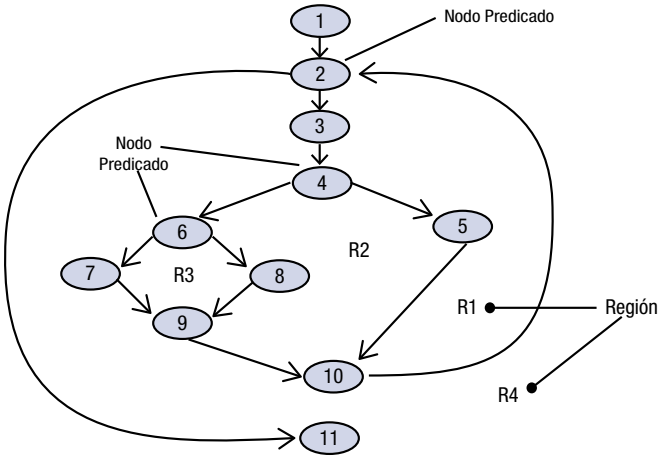


Tabla 4. Tareas para diseñar pruebas unitarias

Tareas para pruebas de caja blanca – Técnica del camino básico
1. Diseñar el diagrama de flujo: (Ejemplo) 
2. Diseñar el grafo de flujo: (Ejemplo) 

Continúa pág. 55

Viene pág. 54

3. Calcular la complejidad ciclomática del grafo (corresponde al número de caminos que componen un conjunto básico de caminos) (Ejemplo) C(G) = Número de regiones del grafo C(G) = (Número de aristas - Número de nodos) + 2 = (A - N) + 2 C(G) = Número de nodos predicado + 1 = P + 1 Complejidad ciclomática del grafo de ejemplo: C(G) = Existen cuatro regiones = 4 C(G) = (13 aristas - 11 nodos) + 2 = 4 C(G) = 3 nodos predicado + 1 = 4 (Los nodos predicado son el nodo 2, el nodo 4 y el nodo 6)
4. Determinar un conjunto básico de caminos independientes Camino 1: 1 - 2 - 11 Camino 2: 1 - 2 - 3 - 4 - 5 - 10 - 2 - 11 Camino 3: 1 - 2 - 3 - 4 - 6 - 7 - 9 -10 - 2 - 11 Camino 4: 1 - 2 - 3 - 4 - 6 -8 - 9 - 10 - 2 - 11
5. Diseñar los casos de pruebas que permitan la ejecución de cada camino básico (ejemplos en Tabla 5)
6. Ejecutar cada caso de prueba y comparar los resultados con lo esperado

A continuación, se incluyen algunos ejemplos de cómo se deben reportar estos casos de prueba diseñados a partir de los caminos básicos generados para las pruebas unitarias:

Tabla 5. Ejemplos de diseño de pruebas unitarias

Camino	Valores de entrada	Valor esperado	Valor obtenido
1	Hora inicial: 10:35 Hora final: 14:50	Diferencia: 4 horas 15 minutos	
2	Hora inicial: 12:55 Hora final: 13:40	Diferencia: 0 horas 45 minutos	
3	Hora inicial: 22:10 Hora final: 00:50	Diferencia: 2 horas 40 minutos	
4	Hora inicial: 23:50 Hora final: 00:21	Diferencia: 0 horas 31 minutos	

Continúa pág. 56

Viene pág. 55

Camino	Valores de entrada	Valor esperado	Valor obtenido
1	Vector de entrada vacío	Vector de salida vacío	
2	Vector con NO múltiplos únicamente Valor = 5 {1, 2, 3, 28, 4, 6, 7, 8, 9, 11}	Vector de salida IGUAL al vector de entrada {1, 2, 3, 28, 4, 6, 7, 8, 9, 11}	
3	Vector con múltiplos únicamente Valor = 5 {15, 25, 30, 40, 55, 60, 75, 85, 95, 105}	Vector de salida vacío	

Camino	Valores de entrada	Valor esperado	Valor obtenido
1	Respuesta = 'N'	Programa finalizado sin efectuar proceso	
2	Respuesta = 'S' y número en rango	Número invertido y pregunta al usuario si desea otro número	
3	Respuesta = 'S' y número 0	Número invertido = 0 y pregunta al usuario si desea otro número	
4	Respuesta = 'S' y número mayor al límite superior del rango	"Valor no válido. Fuera de rango" y pregunta al usuario si desea otro número	
5	Respuesta = 'S' y número menor al límite inferior del rango	"Valor no válido. Fuera de rango" y pregunta al usuario si desea otro número	

3.5.4 Mecanismo de SQA en etapa de pruebas

Durante esta etapa es importante realizar pruebas del sistema, pruebas integrales y pruebas de aceptación. Las pruebas de sistema tienen como objetivo encontrar errores en el sistema como un todo, cuando este se relaciona con su entorno (otras máquinas, otro hardware, redes, fuentes reales de información, etc) [6]. Las pruebas de sistema incluyen pruebas funcionales y pruebas no funcionales. Las funcionales se centran en los

requisitos funcionales del sistema mientras que las no funcionales se centran en probar aspectos restrictivos del diseño tales como: usabilidad, recuperación, volumen, seguridad, estrés y rendimiento). Las pruebas funcionales y las no funcionales pueden estar basadas directamente en los casos de uso, las reglas del negocio y la especificación del sistema. Por su parte, las pruebas integrales tienen como propósito encontrar errores en las unidades de prueba (que han superado las pruebas unitarias) cuando se integran, de manera que el centro de atención está en la arquitectura software, es decir, en los flujos de comunicación entre las unidades y las entidades externas (interfaces del sistema).

Frente a las pruebas de sistema, MECAPS recomienda el uso de la técnica de caja negra y para las pruebas integrales las técnicas top-down y down-top. En la técnica de caja negra sólo se conocen las entradas y salidas esperadas de cada funcionalidad del sistema y permite diseñar casos de prueba ingresando datos válidos y no válidos para verificar los resultados esperados para los datos válidos y mensajes apropiados de error y precaución para los datos no válidos (Plantilla 11). La técnica top-down consiste en iniciar la prueba con los módulos de nivel superior, de manera que se confirme que estos módulos llaman a los módulos de nivel inferior haciendo uso de los parámetros correctos, mientras que en la técnica down-top, se trata de realizar la prueba de llamado desde los módulos de nivel inferior hacia los módulos de nivel superior haciendo uso de los parámetros correctos. Para la técnica de caja negra, MECAPS elaboró una plantilla que permite a los aprendices ADSI diseñar estos casos de prueba basados en la técnica (Plantilla 11).

Plantilla 11. Diseño y ejecución de casos de prueba funcionales

<table>
<tr><th colspan="4">Información del caso de prueba</th></tr>
<tr><td rowspan="2">Caso de prueba No.</td><td rowspan="2"><Número del caso de prueba constituido por SP-[número del caso de uso]-[Número del caso de prueba]></td><td>Versión de ejecución</td><td><Versión diligenciada por el analista de pruebas en el momento de ejecutarla. Este número se incrementa de 1 en 1></td></tr>
<tr><td>Fecha de ejecución</td><td><Fecha de ejecución diligenciado por el analista de pruebas></td></tr>
<tr><td>Caso de uso:</td><td><Identificación del caso de uso objeto de la prueba></td><td>Módulo del sistema</td><td><Nombre del módulo al que corresponde el caso de uso objeto de la prueba></td></tr>
<tr><td>Descripción del caso de prueba:</td><td colspan="3"><Descripción de lo que se pretende probar en el caso de prueba></td></tr>
</table>

Continúa pág. 58

Viene pág. 57

1. Caso de prueba						
a. Precondiciones						
<Lista de precondiciones que deben cumplirse para realizar la prueba>						
b. Pasos de la prueba						
<Pasos secuenciales que deben ser ejecutados por el analista de pruebas o usuario, ante el sistema para ejecutar la prueba>						
Datos de entrada			**Respuesta esperada de la aplicación**	**Coincide**		**Respuesta del sistema**
Campo	**Valor**	**Tipo de escenario**		**Sí**	**No**	
<Descripción del dato de entrada>	<Valor que debe ser suministrado en la prueba para el dato de entrada>	<Tipo de escenario que pretende probarse: Correcto/ Incorrecto>	<Respuesta que se espera de la aplicación>			<Respuesta que se obtuvo de la aplicación en el momento de la ejecución de la prueba>
c. Poscondiciones						
<Lista de poscondiciones que deben cumplirse después de realizar la prueba>						

2. Resultados de la Prueba	
Defectos y desviaciones	**Veredicto**
<Lista de defectos o desviaciones encontrados por el analista o usuario al ejecutar la prueba>	☐ **Paso** ☑ **Falló**
Observaciones	**Probador**
<Observaciones generales del analista o usuario sobre la ejecución de la prueba>	Firma: Nombre: Fecha:

Las pruebas de aceptación, por su parte, permiten que sea el usuario final quien confirme, en su propia infraestructura (pruebas Beta) los resultados de las pruebas del sistema de información, de manera que a través de la replicación de los casos de prueba (del sistema e integrales) que previamente hayan sido ejecutados de manera satisfactoria por los testers y desarrolladores en el ambiente de calidad.

Dentro de esta etapa es importante organizar y planear la forma en que se van a desarrollar las pruebas mencionadas anteriormente (pruebas unitarias, pruebas de sistema y pruebas integrales), de manera que se preparen todos los elementos, recursos y disponibilidad de los ambientes (ambiente de desarrollo, ambiente de calidad, ambiente productivo) para poder ejecutarlas de manera controlada. Para esta planeación, MECAPS propone la elaboración de un plan de pruebas que se construye con el objetivo de responder a preguntas como: ¿Qué hay que probar?, ¿Qué se va a probar?, ¿Cómo se va a probar?, ¿Cuándo se va a probar?, ¿Cuáles son los criterios de comienzo de las pruebas?, ¿Cuáles son los criterios de finalización de las pruebas?, ¿Cuál es el entorno de pruebas y sus riesgos? y ¿Cuáles son los casos de prueba a ejecutar? Para apoyar este proceso de planeación de las pruebas, MECAPS recomienda el uso de la Plantilla 12.

Plantilla 12. Plan de pruebas

PLAN DE PRUEBAS
Sistema XXXXXXXXXXX

Año, Mes, Día

Propósito

Este documento tiene como propósito definir la estrategia y la táctica de un proceso de pruebas que se realizará sobre algunos artefactos de ingeniería del proyecto software denominado "xxxxxxxxxxxxxxxxxx" que se encuentra en etapa de implementación. En la parte estratégica se definen los elementos a ser probados, el alcance, los criterios de inicio y fin de las pruebas, así como los criterios de aceptación de las mismas. En el sentido táctico se definen las tareas por cada uno de los elementos a ser probados según el criterio de fin de las pruebas.

Continúa pág. 60

Viene pág. 59

Estrategia de pruebas

(Qué hay que probar y qué se va a probar)
El proyecto "xxxxxxxx" tiene como objetivo estratégico xxxxxxxx y contienen funcionalidades las cuales fueron requeridas por xxxxxxxx. De este total de funcionalidades se ejecutarán pruebas de caja blanca a los siguientes componentes de código:
1
2
3
4

Así mismo, serán probadas bajo la técnica de caja blanca las siguientes funcionalidades a partir de sus especificaciones (Anexar las especificaciones de entrada).

Recursos del Plan de Pruebas: *(Qué recursos se utilizarán)*
En la realización de estas pruebas participarán los siguientes recursos humanos:
1
2

Y se utilizarán los siguientes recursos de hardware, software, herramientas de soporte:
1
2

El ambiente de pruebas configurado consta de: *Explicar el número y las características de los equipos utilizados para la prueba y cómo están conectados, accesibilidad a la aplicación para pruebas*

Procesador: Quad Core Xeon E5405 Porcessor2x6MB Cache, 2,0GHz, 1333MHz FSB, PE2950
Memoria: 4GB 667MHz (4x1GB) Daul Ranked Fully Buffered DIMMs
Tarjeta de video: LOM NICs are TOE Ready
Disco duro: 2 dd c/u 300GB, 10K RPM Serial-Attach SCSI 3Gbps 3,5 in HotPlug HardDrive
Red: 2 Integradas NetXtreme II 5708 Gigabit NICs TOE Capable

Criterios de aceptación de las pruebas (Evaluación de los resultados): *(Responde a la pregunta: ¿Cuándo los resultados de las pruebas cumplen con los criterios de calidad de los productos?)*

Pruebas unitarias:
- Detectar errores en la ejecución de las pruebas.
- El 90% de las pruebas realizadas deben ser exitosas.

Continúa pág. 61

Viene pág. 60

Pruebas funcionales:

- El resultado de cada caso de prueba debe ser igual al resultado de salida esperado.
- Encontrar fallas al ejecutar los diferentes casos de pruebas.
- La aplicación cumple con los requerimientos funcionales especificados en la fase de análisis
- La aplicación cumple con los requerimientos mínimos para el funcionamiento

Criterios para el comienzo de las pruebas:

- Se poseen los sets de pruebas con escenarios claros.
- El entorno de pruebas es el adecuado para el tipo de pruebas a iniciar.
- Todos los artefactos requeridos se encuentran disponibles.
- Todos los recursos humanos y técnicos necesarios se hallan disponibles.

Criterios para la finalización de las pruebas:

- Se ejecutaron todas las pruebas del sistema.
- Todas las pruebas se ejecutaron de acuerdo con los criterios de evaluación.
- Los incidentes encontrados en las pruebas fueron reportados.

(Cómo se va a probar)

Tipos y técnicas de pruebas

Tipo de prueba	Definiciones	Técnica
Unitarias	Permite verificar la funcionalidad y estructura de cada componente individualmente del sistema, una vez que ha sido codificado.	Caja Blanca
Funcionales	Asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.	Caja Negra

Secuencia de actividades y horarios *(Cuándo se va a probar y horarios de pruebas)*
Para la estimación de las duraciones de las pruebas se tiene en cuenta la restricción de entrega (Elaborar un cronograma de actividades incluyendo las fases de: Recopilación de la información, Elaboración del plan de pruebas, Construcción del set de pruebas de caja blanca y caja negra, Ejecución de pruebas unitarias, Ejecución de pruebas funcionales)

Diseño de las pruebas

Para el diseño de las pruebas se han tenido en cuenta los siguientes aspectos:

Continúa pág. 62

Viene pág. 61

1. Los componentes de código a ser probados deben contar con estructuras de control y condicionales lógicos anidados.
2. Los componentes funcionales deberían contar con una especificación suficiente (Descripción, entradas, salidas de acuerdo con los formatos establecidos por el cliente)
3. Los componentes a probar en este documento cuentan con el mismo nivel de prioridad 1, por tanto, el set de pruebas deberá ejecutarse en un mismo momento
4. Las pruebas serán realizadas una única vez para la versión de componentes del sistema "xxxxxx".
5. Las técnicas definidas serán utilizadas por los probadores bajo un proceso manual de prueba

Criterios de terminación

A continuación, se señalan los criterios de terminación de las pruebas a ejecutar.

- Se ejecutaron todas las pruebas del sistema.
- Todas las pruebas se ejecutaron de acuerdo con los criterios de evaluación.
- Los incidentes encontrados en las pruebas fueron reportados.

Formato casos de prueba unitarias (Anexos)

Formato casos de prueba funcionales (Anexos)

A continuación se presenta un resumen de los tipos de prueba y las técnicas recomendadas por MECAPS para el aseguramiento de la calidad de los desarrollos ADSI (Tabla 6).

Tabla 6. Resumen de tipos de pruebas y técnicas

Tipo de prueba	Objetivo	Técnica	Participantes (Roles)
Unitaria	Detectar errores en los datos, la lógica y algoritmos	Caja Blanca	Desarrolladores
Funcional o de comportamiento	Detectar errores en la implementación de requerimientos	Caja Negra (probar cada caso de uso o función (RF) y reglas de negocio)	Testers, Analistas
Sistema (Tercerización) (Funcionales y no funcionales)	Detectar fallas en el cubrimiento de los requerimientos	Smoke (Ad Hoc, no son exhaustivas) End to end Alfa	Testers, Analistas (Sobre un ambiente replica)

Continúa pág. 63

Viene pág. 62

Integral	Detectar errores de interfaces y relaciones entre componentes	Caja Blanca Top Down Bottom Up	Desarrolladores
Aceptación	Detectar fallas en la implementación del sistema	Funcional Beta	Testers, Analistas, Cliente
Relacionadas con cambios	Detectar errores luego de los cambios. Puede ser usada especialmente en cambios de emergencia cuando se encuentra el sistema ya en producción.	-Verificar que el defecto identificado se corrigió adecuadamente -Verificar que el componente ya probado no tiene nuevos errores luego de las repruebas	Desarrolladores Testers, Analistas
De mantenimiento	Detectar errores luego de que el sistema se encuentra en producción (migraciones, desincorporación de software, planes de mejora, emergencias, parches)	Caja Negra Top Down Bottom Up	Testers, Analistas

Fuente: Extraído de Basic Agile Testing Certified Professional 2016– BDGuidance

Frente a los ambientes mínimos que se necesitan para poder gestionar de manera adecuada y suficiente las pruebas de un producto software, MECAPS recomienda contar con un ambiente de desarrollo para las pruebas unitarias, un ambiente de calidad para realizar las pruebas de sistema y las pruebas integrales y un ambiente de producción. El ambiente de desarrollo puede ser el equipo del desarrollador, de manera que no hay muchas restricciones de memoria, performance y almacenamiento. El ambiente de calidad si requiere de manera importante que tenga unas características muy similares al ambiente de producción, en su configuración (sistema operativo, licencias, interfaces, base de datos, otras aplicaciones requeridas, permisos) y en su capacidad (memoria, almacenamiento, performance). Esta similitud permite asegurar que los resultados de las pruebas de sistema e integrales serán los obtenidos cuando el sistema se encuentre en marcha en el ambiente productivo del cliente.

3.5.5 Mecanismo de SQA en etapa de implementación

La etapa de implementación, no es menos importante dentro del ciclo de desarrollo de software, es por esto que MECAPS recomienda que antes de pasar el sistema desarrollado al ambiente productivo del cliente se hayan realizado las pruebas de aceptación conforme al plan de pruebas. La metodología recomienda en esta parte realizar una revisión básica para poder controlar que todos los componentes técnicos y logísticos estén a punto para el momento de salir a producción (Plantilla 13).

Plantilla 13. Lista de chequeo para salida a producción

Lista de chequeo para salida/montaje en producción		
Ítem	**Descripción**	**Chequear**
1	El producto software fue recibido a conformidad por el cliente luego de las pruebas de aceptación	
2	Se ha cumplido el mecanismo de capacitación de los usuarios finales que harán uso del producto software	
3	Se ha cumplido el mecanismo de capacitación a los recursos técnicos que deberán iniciar con el soporte en producción del producto software	
4	Se ha diseñado y aplicado el esquema de permisos de los usuarios acorde con las políticas del cliente sobre el aplicativo	
5	Se encuentran a punto los elementos de arquitectura que participan en el montaje de producción del producto software (configuraciones de bases de datos, servidores, enlaces, redes, equipos cliente, conexiones remotas, carpetas compartidas, etc)	
6	Se encuentra confirmado y entrenado el grupo de soporte a la aplicación en un periodo de tiempo suficiente luego del montaje a producción, bajo un esquema de garantía específico	
7	Se ha entregado, a los respectivos implicados, las últimas versiones de los documentos del sistema: manuales de usuario, manuales técnicos o de configuración, manuales de instalación/desinstalación	
8	Se han establecido y generado copias para respaldo de los datos de producción en caso de requerir un punto de retorno del ambiente productivo (en etapa de pruebas deben haberse probado estas restauraciones para validar el mecanismo completo)	

Continúa pág. 65

Viene pág. 64

9	Se han definido estrategias (manuales o secundarias) para los procesos críticos del negocio, en caso de que una falla inesperada en el producto instalado no logre atender estos procesos y por lo tanto posterior a su captura manual la información pueda ser ingresada al sistema, de manera que se disminuya el impacto de afectación a la operación del negocio mientras se normaliza el proceso	

Los elementos que no se hayan verificado deberán ser en lo posible cubiertos para que pueda asegurarse de alguna manera la calidad del montaje a producción y disminuir los impactos sobre la operación del negocio.

3.5.6 Mecanismo de SQA para la gestión del proyecto

Dentro de la gestión de proyecto se debe considerar la gestión del producto y la gestión propia del proyecto, en esta última es relevante monitorear el desempeño del proyecto, el cual se determina a partir de la gestión del alcance, tiempo y costo. Debido a que MECAPS se enfoca en la evaluación de la calidad del producto, el aseguramiento de la gestión del proyecto se restringe a la gestión del producto software que se está desarrollando.

Todos los mecanismos, estrategias y/o procedimientos que permitan garantizar una gestión adecuada del proyecto deberán ser plasmados en un plan de gestión del proyecto. MECAPS recomienda las siguientes actividades mínimas para esta gestión (Figura 6).

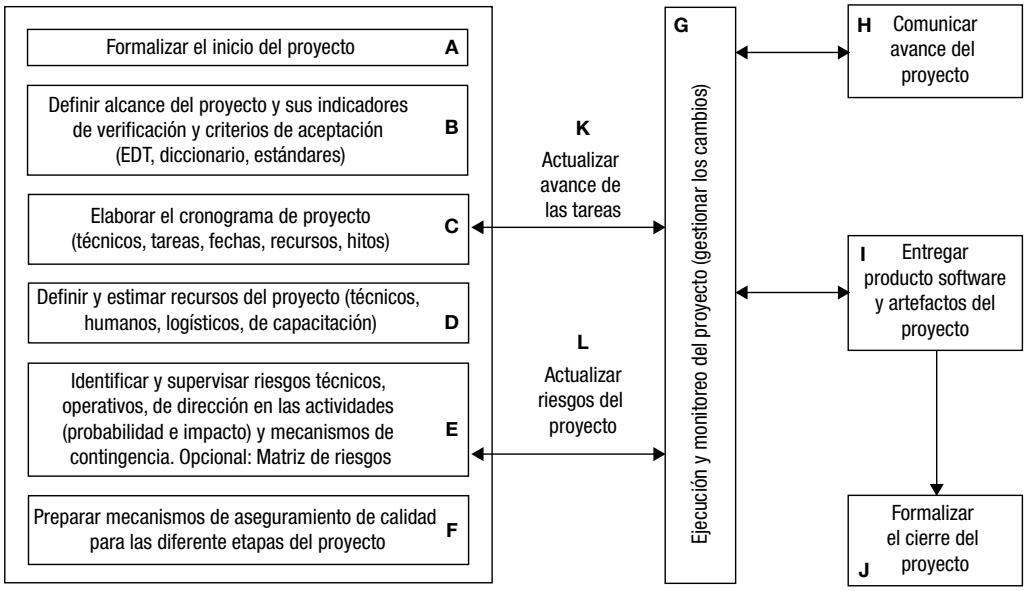


Figura 6. Plan mínimo de gestión de proyecto de desarrollo de software

A continuación, MECAPS presenta una lista de chequeo que permite asegurar este plan mínimo de gestión del proyecto, haciendo un mapeo entre las actividades presentadas en la Figura 6 y unas preguntas que apoyan esta validación.

Plantilla 14. Lista de chequeo para el aseguramiento de la gestión de proyecto

Actividad	Pregunta	Chequear
A	¿Está confirmada la viabilidad del proyecto por parte del cliente y los objetivos son claros?	
B	¿Está definido el alcance del trabajo para el proyecto?	
B	¿Están definidos en el alcance los entregables del proyecto?	
C	¿Están definidas las tareas técnicas y las de control asociadas al alcance del proyecto?	
C	¿Están definidos los recursos humanos requeridos para lograr el alcance del proyecto?	
D	¿Está estimado el cronograma del proyecto?	
D	¿Está estimada la duración (fechas) de las actividades del proyecto?	
D	¿Está estimado el esfuerzo requerido para las actividades del proyecto?	
D	¿Están estimados los recursos técnicos para el proyecto?	
D	¿Está estimado el costo para las actividades del proyecto?	
E	¿Están identificados los riesgos del proyecto?	
E,G	¿Están monitoreados los riesgos durante la ejecución del proyecto?	
D	¿Está desarrollada una estrategia de control de versiones software?	
C	¿Está implementada una estrategia de versión de controles software?	
C	¿Está definido el sistema de Backup?	
C	¿Está definido el mecanismo de recuperación?	
F	¿Están preparados los mecanismos de aseguramiento de calidad del producto?	
C,G	¿Está monitoreado el progreso del proyecto contra la planeación?	
H	¿Se ha comunicado el progreso del proyecto a los interesados?	
K	¿Están tomadas en cuenta las acciones de ajuste en el plan de ejecución del proyecto?	

Continúa pág. 67

Viene pág. 66

K	¿Están tomadas en cuenta las acciones correctivas en el plan de ejecución del proyecto?	
C,G	¿Están definidas las actividades de revisión del plan entre el equipo de trabajo y el cliente, que garanticen que el trabajo realizado está conforme con los requisitos software y el plan de ejecución?	
C,G	¿Están registrados los acuerdos resultantes de las actividades de revisión?	
J	¿Está realizado el cierre del proyecto, previa aceptación del cliente?	

CAPÍTULO 4

CONCLUSIONES

4.1 Aplicación de la metodología

De acuerdo con [26] la comprensión de una disciplina implica aprendizaje, es decir: (i) la observación, la reflexión y la encapsulación del conocimiento; (ii) la construcción de propuestas/modelos en el área de conocimiento; (iii) la experimentación con las mismas; y (iv) la evolución de la propuesta/modelo. La Ingeniería de Software, al igual que cualquier otra disciplina, tiene dos objetivos fundamentales [27]: (i) aumentar el conocimiento teórico, con el fin de entender por qué las cosas suceden en un área de interés particular, y (ii) mejorar la práctica del área de interés para que los resultados de la investigación puedan ser útiles. Es importante entonces que los nuevos conocimientos teóricos propuestos (modelos, procesos, técnicas, etc.) cuenten con una adecuada aplicación en la práctica. En este sentido, según [28] un método empírico utilizado actualmente en Ingeniería de Software es el de Reporte de Experiencias. Los Reportes de Experiencia son llamados por diferentes nombres como "Informes de lecciones aprendidas" o "Informes de experiencia de la industria", los cuales describen la experiencia de aplicar en la práctica alguna propuesta teórica por un profesional o *stakeholder* [29]. Es importante que el informe sea sólido y mantenga la objetividad como requisitos importantes para fortalecer el campo de investigación [29].

En este sentido, la aplicación de algunos de los mecanismos de aseguramiento de la calidad propuestos en los diferentes componentes de evaluación de MECAPS han sido aplicados en la práctica a productos de trabajo generados por los aprendices durante la ejecución de sus proyectos de formación. Esta aplicación de MECAPS se presenta a continuación siguiendo las indicaciones para los informes de experiencias recomendadas por [29]; por tanto en las siguientes secciones se describen estas experiencias en términos de: la descripción del contexto donde fueron aplicados los mecanismos (Revisión técnica formal a la especificación de requisitos, Lista de chequeo de requisitos y Mecanismos de SQA en la etapa de desarrollo) y el informe de su aplicación.

4.1.1 Descripción del contexto de aplicación

Una vista general de los diferentes proyectos en los que se aplicaron algunos mecanismos de aseguramiento de la calidad descritos por MECAPS se muestran a continuación:

- Proyecto 1: Semillas de vida Jambaló.
- Proyecto 2: Jambaló Cuenta
- Proyecto 3: Repórtate (Cultura y Deporte)
- Proyecto 4: Justicia propia
- Proyecto 5: Lenguas
- Proyecto 6: Gestión de matrículas
- Proyecto 7: GDProyect

Proyecto	Hoja de vida de los proyectos				
	Problema	**Justificación**	**Alcance**	**Número de aprendices**	**Cliente**
Semillas de vida	Actualmente CDI semillas de vida gestiona las encuestas establecidas por el ICBF de forma manual. Se opta por un sistema de información que agilice esta labor.	Actualmente, la recolección de datos de los beneficiarios del programa CDI se lleva a cabo de forma manual mediante una Encuesta establecida por el ICBF.	Desarrollo del módulo dos de la ficha de caracterización que corresponde a niños y niñas menores de 5 años, esta será realizada tanto en web como en aplicativo móvil.	3	Cabildo Indígena de Jambaló

Continúa pág. 71

Viene pág. 70

Jambaló Cuenta	El Cabildo indígena de Jambaló no tiene definido un instrumento de recolección de datos que permita gestionar el censo de la población indígena.	Actualmente el proceso tradicional no es adecuado porque se lleva a cabo manualmente. Es necesario definir un instrumento de recolección de datos con variables acordes con las necesidades de las autoridades tradicionales y de la comunidad de Jambaló.	Construcción del instrumento para que permita gestionar información acerca de: datos personales, núcleo familiar, vivienda, economía, a nivel veredal y municipal.	4	Cabildo indígena de Jambaló
Repórtate (Cultura y Deporte)	No hay un control sistematizado de registros de evidencias, además no se cuenta con un formato o un medio para realizar la gestión de los eventos y no existe registro de los juegos tradicionales.	Permite abordar los procesos culturales expresados en diferentes eventos educativos de recreación, danza, deporte y biblioteca.	Facilitar la gestión de la programación de eventos culturales a través de peticiones, donde cada evento será publicado en el cronograma de actividades, contar con un módulo de PQRS.	3	Unidad de deporte y cultura del municipio de Jambaló

Continúa pág. 72

Viene pág. 71

Justicia propia	Pérdida de documentos, pérdida de tiempo para funcionarios de dicha entidad y difícil gestión de la información.	Tener acceso oportuno, seguro, confidencial y controlado sobre los casos de Justicia Propia que se presentan dentro del territorio indígena.	Desarrollar un sistema de información que optimice el proceso de recepción, control, trámite y seguimiento de los casos, permitiendo la recolección de datos, su registro, modificación y generación de reportes, de forma automatizada.	3	Comisión jurídica del Cabildo Indígena Jambaló
Lenguas	Control de formación de los cursos trimestrales, difícil manejo de la información consignada en papel y dispositivos USB, gestiones insuficientes para el registro y atención de las P.Q.R.S.	Controlar la formación y por qué se realiza, teniendo la opción de registrar la formación adecuadamente. Incluyendo la gamificación al sistema para convertirlo en un apoyo de enseñanza al programa, dándole mayor usabilidad y una forma divertida de aprender las lenguas tradicionales jugando.	El sistema permitirá llevar un control organizado de la información tanto del personal administrativo (contrataciones, terminación de contratos o renuncias) como de estudiantes (inscripciones, deserciones, terminación de curso). Lo anterior, basado en un control de acceso.	3	Programa de lenguas del Cabildo Indígena de Jambaló

Continúa pág. 73

Viene pág. 72

Gestión de matrículas	Las matrículas y el registro de los datos del recurso humano se realizan de forma manual. El control de los registros se maneja en archivos físicos y medios magnéticos.	La coordinación del núcleo de educación del resguardo de Jambaló no cuenta con un sistema de información propio. para la gestión de las matrículas estudiantiles.	Gestión de las matrículas y el registro de datos de las personas que laboran en las instituciones educativas, este será realizado tanto en web como en aplicativo móvil.	3	Núcleo de educación del Cabildo Indígena Jambaló
GDProyect	No se lleva una organización adecuada de los proyectos que reposan en cada carpeta donde se encuentra el archivo., no se cuenta con una herramienta que les permita llevar un control de cada documento que se genera durante la ejecución de algún proyecto. Todos los registros se hacen manualmente en hojas de papel.	Mejorar el almacenamiento y gestión de documentos de los proyectos, de manera que se gane eficiencia en los tiempos de consulta tanto del documento como de su información.	El módulo a desarrollar gestionará los documentos digitalizados y apoyará el trabajo del líder de SENNOVA. Entre las funciones están: almacenamiento de proyectos, su consulta, generar alertas y preservar el documento	2	Semillero de investigación de Sinergia- SENA

A todos los proyectos se les aplicó el mecanismo de aseguramiento de calidad RTF y lista de chequeo para la etapa de especificación; sin embargo, sólo al proyecto GDProyect, se le aplicaron tècnicas de aseguramiento de la etapa de desarrollo que propone MECAPS, dado que era el único proyecto que contaba con un producto software terminado.

4.1.2 Revisión técnica formal a la especificación de requisitos

Para la aplicación de esta técnica en los proyectos realizados por los aprendices ADSI, el grupo de investigadores definió cuatro actividades: Planeación del mecanismo, Socialización de los proyectos a los involucrados, Ejecución del mecanismo y Realimentación de resultados. A continuación se describen brevemente estas actividades:

- *Planeación del mecanismo:* Inicialmente los investigadores asesores e investigadores instructores intercambiaron información sobre los diferentes proyectos que estaban llevando diferentes grupos de aprendices ADSI para analizarlos y determinar cuáles serían seleccionados para la aplicación de la técnica. De esta manera se seleccionaron los proyectos descritos en la sección anterior. También se definió el grupo de evaluadores (quienes fueron un grupo de estudiantes con conocimientos en el tema de calidad de software de la Universidad del Cauca) como el grupo de calidad que ejecutaría la técnica de RTF a las especificaciones de los proyectos seleccionados. Se planearon ocho grupos de calidad con un promedio de cuatro miembros para aplicar este mecanismo de aseguramiento a cada uno de los proyectos.
- *Socialización de proyectos:* Los investigadores aprendices (quienes eran los encargados del desarrollo de los proyectos de formación) socializaron cada uno de los proyectos a evaluar a los grupos de calidad. Esta socialización consistió de una presentación por treinta minutos donde se mostró la descripción del problema que se pretende atacar con el proyecto, además de su justificación, alcance y objetivos. Se aclararon aspectos relacionados con el proyecto con el fin de tener un acercamiento inicial a cada proyecto por parte del grupo de calidad correspondiente. En esta sesión los investigadores aprendices entregaron el documento de especificación de requisitos al grupo de calidad y se aclararon aspectos relacionados con este documento.
- *Ejecución del mecanismo:* Cada grupo de calidad utilizó el mecanismo de RTF, incluido entre los mecanismos de calidad sugeridos por MECAPS, para evaluar la SRS asignada y generar al finalizar el proceso de evaluación, un reporte de hallazgos de su aplicación. Cada grupo de calidad identificó un conjunto de estándares aplicables como referente de evaluación, los cuales fueron analizados por los evaluadores para determinar los criterios de revisión del SRS frente al estándar. Estos criterios fueron aplicados para detectar los problemas o insuficiencias del documento SRS, con el fin de sugerir recomendaciones y puntos de mejora. Finalmente, el grupo de calidad emitió un concepto (rechazado/aprobado) sobre el estado actual del artefacto SRS relacionado con que si cumple las condiciones mínimas de calidad para servir como base para la siguiente etapa.
- *Realimentación de resultados:* Finalmente, cada uno de los grupos de calidad socializó a los investigadores aprendices y a los investigadores instructores los resultados de haber aplicado la técnica de RTF a cada uno de los SRS bajo análisis. Esta realimentación consistió de una presentación por 30 minutos donde se revelarón los puntos fuertes, problemas e insuficiencias detectados del documento SRS, sugiriendo recomendaciones y puntos de mejora.

A continuación, se presentan a manera de ejemplo los resultados de la revisión técnica formal realizada a uno de los proyectos formativos que se eligió como caso de estudio para la aplicación de MECAPS

4.1.2.1 Ejemplo de aplicación del RTF

GDProyect es un proyecto formativo y uno de sus objetivos es el desarrollo de un producto software para la gestión de historias laborales del SENA Regional Cauca.

La historia laboral es un conjunto de unidades documentales de estructura y contenido homogéneo, generadas desde un mismo órgano o sujeto productor, como consecuencia del ejercicio de sus funciones específicas [7].

Para este proceso de RTF, se toma como referente los estándares ISO 830 (Especificación de requisitos software) , ISO 25000 (Requisitos y Evaluación de Calidad de Productos de Software) e ISO 27000 (Mejores prácticas para la seguridad de la información), a partir de los cuales se realiza el proceso de evaluación técnica del proyecto GDProyect (Tabla 7).

Tabla 7. Resultados de la aplicación de RTF en el proyecto formativo GDProyect

3.2. Problema detectado 1 Faltan algunas funcionalidades por incluir en el sistema para que se cumpla con el alcance de los objetivos general y específicos de la sección 2.1 y 2.2 respectivamente. **3.2.1. Sugerencia de corrección** Para el correcto funcionamiento del sistema, de gestión documental se propone incluir las siguientes funcionalidades en relación con los documentos digitales: cambiar estado del documento digital (vigente, archivado,etc), gestión de metadatos del documento, incluir la opción de borrar una alerta. Esto en cumplimiento de la norma ISO 25010 que establece que la adecuación funcional es la capacidad del producto software para proporcionar funciones que satisfacen las necesidades declaradas e implícitas, cuando el producto se usa en las condiciones especificadas [1] y que además se subdivide en: **Completitud funcional.** Grado en el cual el conjunto de funcionalidades cubre todas las tareas y los objetivos del usuario especificados. **Pertinencia funcional.** Capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos del usuario especificados.
3.3. Problema detectado 2 Al revisar los objetivos específicos sección 2.2 se encuentra que no se hace referencia a las tareas de protección de la información, requisito fundamental para un sistema de Gestión Documental, aunque se puede ver que en la sección de requisitos no funcionales se menciona.

Continúa pág. 76

Viene pág. 75

3.3.1. Sugerencia de corrección

Se debería incluir como un objetivo específico la seguridad del sistema debido al grado crítico que este tiene, tomando como base para este objetivo la importancia de la seguridad de la información descrita en la norma ISO 27001:2013 que define la información como el activo más valioso de una organización, como una necesidad, y con la cual se busca:

- Garantizar la confidencialidad, la integridad y disponibilidad de la información sensible.
- Reducir la incertidumbre por el conocimiento de los riesgos e impactos asociados (Cláusula 6.1.2 Evaluación de los riesgos de la seguridad de la información).
- Permitir la mejora continua en la gestión de seguridad de la información. Entre otras.

4.1.2.2 Resultados generales de la aplicación del mecanismo

La Tabla 8 resume los resultados encontrados después de aplicar RTF sobre el SRS de cada uno de los siete proyectos formativos, incluye los referentes tenidos en cuenta durante la revisión cumplida por los evaluadores; así mismo el concepto final sobre la calidad de cada documento:

Tabla 8. Resultados generales después de aplicar RTF en siete proyectos formativos

Proyecto	Concepto de Evaluación Final	Referentes
Semillas de vida	Rechazado	ISO 29148; IEEE 830; ISO 15408; ISO 9001; ISO 19148
Jambaló Cuenta	Rechazado	ISO 29148; ISO 1233; ISO 25010
Repórtate (Cultura y Deporte)	Rechazado	IEEE 830
Justicia propia	Rechazado	ISO 25030
Lenguas	Rechazado	ISO 29148
Gestión de matrículas	Aceptado con correcciones	ISO 29148; ISO 25010
GDProyect	Aceptado con modificaciones menores	IEEE 830; ISO 25010; ISO 27000

A partir de los resultados de evaluación de la calidad de la especificación de requisitos del producto software (SRS), presentados en las tablas 7 y 8, se puede observar que:

- Frente a qué tan completo es el documento de especificación de requisitos (SRS): en un 71,4% de los casos, la especificación esta incompleta, sólo 2 proyectos muestran la aceptación del documento con modificaciones menores. Este resultado permite establecer que hay una oportunidad de mejora para el proceso de elaboración del SRS de los productos software.

- Frente a la inclusión de los requisitos no funcionales del producto software en la SRS: el 100% de los casos, cuenta con baja consideración y detalle de estos requisitos en el documento. Este resultado permite concluir que el alcance de los documentos de especificación en los proyectos puede ser ampliado con la especificación de los requisitos no funcionales del producto software, de manera que incluyan aspectos de calidad (propuestos por la ISO 25010) relevantes y específicos a cada producto.

4.2 Reflexiones

La aplicación de técnicas explícitas de calidad sobre artefactos de ingeniería en proyectos de desarrollo software permite un aprendizaje adecuado sobre cómo realizar la evaluación de los productos software en su ciclo de desarrollo, así como fortalecer la habilidad de identificar puntos de mejora para el proceso. La evaluación de la calidad en etapas tempranas del desarrollo, a través de mecanismos como la revisión técnica formal sobre la especificación de requisitos, como un ejemplo de ésta, permite minimizar errores en el planteamiento de la solución y en los artefactos siguientes del ciclo de desarrollo.

Es de vital importancia para el equipo de instructores y el aprendiz determinar las normas de calidad que se deben tener en cuenta durante el desarrollo del proyecto formativo, esencialmente a partir de la revisión del alcance del proyecto y la definición de los productos que se van a evaluar durante el proceso de ingeniería. Lo anterior se puede lograr por medio de dos momentos esenciales de aprendizaje sobre aseguramiento de la calidad del software: (i) un primer momento, es la identificación de las normas de calidad esenciales que sean acordes a la calidad esperada del producto dentro del alcance del proyecto, esto es, teniendo en cuenta las restricciones bajo las cuales se puede desarrollar el producto en el contexto de formación profesional, y (ii) el momento de la revisión del cumplimiento de las normas de calidad en cada artefacto generado y que a la vez facilite la mejora continua en todas las fases del proceso de ingeniería.

La aplicación de MECAPS en un proyecto formativo facilita: (i) el aprendizaje práctico sobre aseguramiento de calidad de productos software y, (ii) la identificación y definición de los diferentes roles participantes en el proceso de desarrollo de software (como son líder del proyecto de desarrollo, analista, diseñador, desarrollador, tester,

entre otros); Estos aspectos de formación se han identificado en la industria software como una necesidad vigente de los sectores productivos regional y nacional para esta industria.

BIBLIOGRAFÍA

[1] C. M. Zapata and G. González Calderón, "Revisión de la literatura en interoperabilidad entre sistemas heterogéneos de software", Ingeniería e investigación, vol. 29, pp. 42-47, 2009.

[2] J. Hurtado, F. Pino, J. Vidal, C. Pardo, and L. Fernández, "Agile SPI: Software Process Agile Improvement, A Colombia Approach to Software Process Improvement in Small Software Organizations". vol. 3, L. U. Pierre F. Tiako Usa, Ed., ed USA: Idea Group Inc., 2009, pp. 3994-3994.

[3] J. Capote, C. J. Llantén, C. Pardo, A. J. González, and C. A. Collazos, "Gestión del conocimiento como apoyo para la mejora de procesos software en las micro, pequeñas y medianas empresas", Ingenieria e investigacion, vol. 28, pp. 137-145, 2008.

[4] K. Heshusius, "Colombia: Desafíos de una industria en formación", in Desafíos y oportunidades de la industria del software en América Latina, P. Bastos and F. Silveira, Eds., ed: Cepal en coedición con Mayol Ediciones S.A., 2009, pp. 139-170.

[5] D. M. Fernández, M. Rodríguez, J. Verdugo, M. Piattini, and E. Fernández-Medina. (2012, Diciembre). Evaluacion de la Calidad y Seguridad en Productos Software. Available: https://administracionelectronica.gob.es/pae_Home/dms/pae_Home/documentos/Estrategias/pae_Tecnimap/pae_TECNIMAP-2010/pae_TECNIMAP_2010_Zaragoza/Tecnimap-2010-Zaragoza/pae_COM_2010-Iniciativas_legales_y_tecnologicas/28inciativas-legales.pdf

[6] J. Kasurinen, "Elaborating software test processes and strategies", in 2010 Third International Conference on Software Testing, Verification and Validation, 2010, pp. 355-358.

[7] T. Koomen, B. Broekman, L. van der Aalst, and M. Vroon, TMap next: for result-driven testing: Uitgeverij kleine Uil, 2013.

[8] R. Conradi and A. Fuggetta, "Improving Software Process Improvement", IEEE Software, vol. 19, pp. 92-99, 2002.

[9] M. Piattini, F. García, I. García Rodríguez, and F. Pino, Calidad de Sistemas de Información. Tercera edición, Segunda Edición ed. Madrid: Rama, 2014.

[10] E. Kit and S. Finzi, Software testing in the real world: improving the process: Addison-wesley, 1995.

[11] L. Huang and B. Boehm, "How much software quality investment is enough: A value-based approach", IEEE Software, vol. 23, pp. 88-95, 2006.

[12] H. James, El coste de la mala calidad. Madrid: Ediciones Díaz de Santos, 1990.
[13] I. Burnstein, T. Suwanassart, and R. Carlson, "Developing a testing maturity model for software test process evaluation and improvement," in Test Conference, 1996. Proceedings., International, 1996, pp. 581-589.
[14] F. Ruiz and M. G. P. Velthuis, "Entorno global para la gestión del proceso de mantenimiento del software", in JISBD'2001: Jornadas de Ingeniería del Software y bases de datos: 21 y 23 de noviembre de 2001, Almagro (Ciudad Real), 2001, pp. 157-170.
[15] Fedesoft, "Estudio de la caracterización de productos y servicios de la industria del software y servicios asociados 2012", Fedesoft 2012.
[16] A. de Rojas, T. E. Vos, and B. Marín, "Experiencias de una PYME en la mejora de procesos de pruebas", Innovación, Calidad e Ingeniería del Software, vol. 5, p. 63, 2009.
[17] M. E. Fagan, "Design and code inspections to reduce errors in program development," in Pioneers and Their Contributions to Software Engineering, ed: Springer, 2001, pp. 301-334.
[18] M. E. Fagan, "Advances in software inspections", in Pioneers and Their Contributions to Software Engineering, ed: Springer, 2001, pp. 335-360.
[19] ISO, "ISO/IEC FDIS 12207:2007(E). Systems and software engineering - Software life cycle processes.", International Organization for Standardization, Geneva2007.
[20] S. Acuña and N. Juristo, Software Process Modeling (The Kluwer International Series in Software Engineering): Springer-Verlag New York, Inc., 2005.
[21] S. Acuna and M. Sanchez Segura, New Trends in Software Process Modelling (Software Engineering and Knowledge Engineering) (Series on Software Engineering and Knowledge Engineering): World Scientific Publishing Co., Inc., 2006.
[22] F. García, "FMESP: Marco de Trabajo Integrado para el Modelado y la Medición de los Procesos Software," Doctorado, Escuela Superior de Informática, Universidad Castilla-La Mancha, Ciudad Real, 2004.
[23] F. Pino, C. Pardo, F. García, and M. Piattini, "Assessment methodology for software process improvement in small organizations", Information and Software Technology, vol. 52, pp. 1044-1061, 2010.
[24] R. S. Pressman, Ingeniería del software, Un enfoque práctico vol. Séptima Edición: Mc GrawHill.
[25] R. F. Z. Sotés, "Las inspecciones de Software y las Listas de Comprobación", Magister, Informática, Universidad del Valle.
[26] V. Basili, "Keynote on "Experimental Software Engineering", in Proceedings 7th European Workshop on Software Process Technology (EWSPT 2000), Kaprun (Austria), 2000, p. 150.
[27] P. A. Philips, "Disseminating and Applying the Best Evidence", Medical Journal of Australia (MJA), vol. 168, pp. 260-261, 1998.
[28] M. Zelkowitz, "An update to experimental models for validating computer technology", Journal of Systems and Software, vol. 82, pp. 373-376, March 2009.
[29] M. Montesi and P. Lago, "Software engineering article types: An analysis of the literature", Journal of Systems and Software, vol. 81, pp. 1694-1714, 2008.