MEJORA DEL PROCESO DE DESARROLLO DE SOFTWARE DEL ÁREA DE TECNOLOGÍA DE INFORMACIÓN DE LA UNIVERSIDAD DE MANIZALES

Diana Carolina Gómez Benavides

UNIVERSIDAD AUTÓNOMA DE MANIZALES
FACULTAD DE INGENIERÍA
MAESTRÍA EN GESTIÓN Y DESARROLLO DE PROYECTOS DE SOFTWARE
MANIZALES, COLOMBIA
2021

MEJORA DEL PROCESO DE DESARROLLO DE SOFTWARE DEL ÁREA DE TECNOLOGÍA DE INFORMACIÓN DE LA UNIVERSIDAD DE MANIZALES

CALIDAD Y MÉTRICAS DE SOFTWARE

Autora
DIANA CAROLINA GÓMEZ BENAVIDES

Proyecto de trabajo de grado presentada(o) como requisito parcial para optar al título de:

Magister en Gestión y Desarrollo de Proyectos de Software

Director (a):
Phd. Faber Danilo Giraldo

UNIVERSIDAD AUTÓNOMA DE MANIZALES
FACULTAD DE INGENIERÍA
MAESTRÍA EN GESTIÓN Y DESARROLLO DE PROYECTOS DE SOFTWARE
MANIZALES, COLOMBIA
2021

RESUMEN

Evidenciando la dificultad en el proceso de calidad de software de la Universidad de Manizales, que se lleva a cabo en los desarrollos gestionados en el área de TI, y, ante la necesidad de mejorar los procesos productivos del área, generando más confiabilidad ante los usuarios y directivos, se busca la implementación e implantación de una, y/o la unión de diversas metodologías que suplan las necesidades específicas de la universidad la cual dará inicio bajo un proceso estandarizado y apoyado en el análisis de varios métodos que optimizan la falencia, forjando como resultado final un proceso de software que cumple con las características y necesidades de todos los usuarios, y que recíprocamente disminuirá en altos porcentajes los reprocesos en base al desarrollo.

El presente documento tiene como fin mostrar detalladamente cómo se realizó la construcción del proceso de desarrollo de software para la Universidad de Manizales, el cual se apoyó del Modelo IDEAL como modelo de mejora de proceso, OPEN UP y SCRUM como modelos de referencia y la norma ISO NTC-29110 cómo guía de referencia y evaluación de procesos, para la mejora de la relación de calidad entregada en cada proyecto del área de TI.

Palabras clave: Proceso de Desarrollo, ISO 29110, Modelo IDEAL, OPEN-UP, SCRUM, RUP.

ABSTRACT

Evidencing the difficulty in the software quality process of the University of Manizales, which is carried out in the developments managed in the IT area, and, given the need to improve the productive processes of the area, generating more reliability before users and managers, seeks the design and implementation of one, and / or the union of various methodologies that meet the specific needs of the university which will start under a standardized process and supported by the analysis of various methods that optimize the shortcoming, forging as a final result, a software process that meets the characteristics and needs of all users, and that reciprocally will reduce reprocessing based on development in high percentages.

The purpose of this document is to show in detail how the construction of the software development process for the University of Manizales was carried out, which was supported by the IDEAL Model as a process improvement model, OPEN UP and SCRUM as reference models and the standard ISO NTC-29110 as a reference guide and process evaluation, to improve the quality relationship delivered in each project in the IT area.

Keywords: development process, ISO 29110, IDEAL Model , OPEN-UP, SCRUM, RUP.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

LISTA DE SÍMBOLOS Y ABREVIATURAS

Abreviatura	Término
SEI	Software Engineering Institute
IDEAL	Modelo para la Mejora Continua de Procesos
SPI	Agil Software Process Improvement o Mejora de Procesos de Software
CMMI	Integración de sistemas modelos de madurez de capacidades o Capability Maturity Model Integration
OPEN-UP	Open Unified Process o Proceso Unificado Abierto
RUP	Rational Unified Process o Proceso Unificado de Racional

1 INTRODUCCIÓN

En la actualidad, existen empresas que desarrollan software y cuyo proceso se hace de forma manual o bajo metodologías poco aptas para dichas compañías, esta situación conlleva a que temas como la calidad del software, entregas en tiempos acordados y bajos costos se vuelvan todo un desafío o un gran reto.

En Colombia, gran parte de las empresas que se dedican al desarrollo de software no utilizan una metodología o proceso de desarrollo ya sea por razones de tiempo, costos, impacto o simplemente no se acomodan a los procesos de desarrollos conocidos, ya sea tradicional o ágil.

A pesar de todo lo anterior, la calidad del software es un tema que cada día coge más fuerza y relevancia entre la industria del software y más aún entre los usuarios, esto debido a que se está convirtiendo en un elemento estratégico gracias a su gran impacto en la competitividad y en la satisfacción de las necesidades del cliente.

Por todo esto, en el área de T.I de la Universidad de Manizales ha surgido la necesidad de adaptar, instaurar e institucionalizar un proceso de desarrollo que apoyen sus procesos institucionales y que a su vez mejoren la calidad de los productos, mejore la estabilidad del desarrollo, la predictibilidad de un proyecto en términos de tiempos y recursos, que brinde al equipo de trabajo una guía para construir aplicaciones eficientes y logren la estandarización de los producto de software que se generan dentro del área.
Para desarrollar esta tesis fue necesario evaluar los diferentes estándares internaciones y metodologías de desarrollo para no solo definir el "qué hacer" sino además el "cómo proceder" en la definición e instauración un proceso de desarrollo propio para el área de T.I.

Los estudios pertinentes que se llevaron a cabo y la aplicación de la metodología definida, dio como resultado la mejora de un proceso de desarrollo para el área de T.I de la Universidad de Manizales.

2 PLANTEAMIENTO DEL PROBLEMA DE INVESTIGACIÓN Y SU JUSTIFICACIÓN

2.1 DESCRIPCIÓN DEL ÁREA PROBLEMÁTICA

Las tecnologías de la información y las comunicaciones, con su impetuoso avance, que data desde mediados del siglo XX y lo que ha transcurrido del siglo XXI, han impactado positivamente la vida cotidiana del hombre. De la misma forma, todas las áreas de desempeño de la humanidad han recibido un impulso de dichas tecnologías, a tal punto que nadie concibe el funcionamiento de una organización sin un adecuado sistema de información, donde incluso para ciertas empresas la información misma se ha convertido en su activo más importante.

Esto es válido en el sector educativo y particularmente en las instituciones de educación superior, donde múltiples procesos asociados tanto a la información de los estudiantes (docencia), como a los demás procesos misionales (investigación y proyección social) se han acelerado gracias al desarrollo de robustas soluciones tecnológicas que apoyan la labor de docentes, estudiantes y administrativos.

Actualmente, la Universidad de Manizales cuenta con un área de Tecnologías de Información (T.I) que entre sus funciones se encuentran gestionar, desarrollar y administrar sistemas de información que apoyen los procesos misionales e institucionales, convirtiéndose así en una pieza fundamental y estratégica dentro de la Universidad, permitiendo aumentar la competitividad y darle valor agregado a dicha institución. Así lo asegura González (2015) “Con una gestión adecuada y alineada con la estrategia de la empresa, el área de Tecnologías de información ayuda a crear nuevas oportunidades de negocio y a incrementar la competitividad. Las organizaciones que no se enfoquen en este tema están perdiendo oportunidades de monetizar inversiones” (p. 13).

Una de las actividades que realiza el área de T.I que más genera valor a las institución es diseñar software con calidad, sin embargo, y a pesar de que él área cuenta con un proceso

de desarrollo no formal, la entrega a tiempo de los desarrollos, la falta de calidad de sus productos, la cantidad de errores y bugs encontrados luego de liberar un aplicativo, la dificultad a la hora de emplear y administrar las soluciones tanto tercerizadas como las desarrolladas al interior de la Universidad, la dificultad a la hora de integrar soluciones en las cuales participan más de un desarrollador, hace que se dificulte la labor y genere descontento entre los usuarios finales y peor aún entre las directivas.

Es por esto que el área de T.I, en su necesidad de obtener calidad en sus productos y superar todas las dificultades anteriormente mencionadas, ha decidido mejorar su proceso de desarrollo con la ayuda de una o varias metodologías de desarrollo que se ajusten a las necesidades de la organización, que permita generar productos con calidad y por ende generar un parte de tranquilidad entre los usuarios y directivas de dicha institución.
Para suplir las necesidades anteriormente mencionadas, se hace necesario que el área de T.I mejore sus procesos con la creación, implementación e incorporación de un proceso de desarrollo propio, el grande desafío es seleccionar la o las metodologías que suplan dichas necesidades y se adapten a los procesos organizacionales.

2.2 FORMULACIÓN DEL PROBLEMA

¿Cómo mejorar el proceso de Desarrollo de Software que se lleva a cabo dentro del área de T.I. de la Universidad de Manizales?

3 JUSTIFICACIÓN

Las funciones que lideran las áreas de Tecnologías de Información en las empresas, y específicamente en las entidades de educación superior, van más allá de la gestión y administración de los recursos informáticos en la institución, ya que juegan un papel muy importante dentro de las instituciones, esto debido a que aportan de manera significativa a la innovación, a la competitividad y al logro de metas institucionales: metas operativas, de rendimiento y de utilidad.

De esta forma, en la Universidad de Manizales, la gestión del área TI ha trascendido de la administración de la tecnología y recursos informáticos (servidores, redes, equipos de cómputo, ...) a la gestión de aplicaciones informáticas que apoyan y automatizan los procesos organizacionales e institucionales. Esto ha hecho que el tema de la calidad esté tomando cada vez más fuerza dentro del área y la misma institución, ya que, los compromisos institucionales así lo exigen, más aún cuando hablamos de una Universidad con acreditación Institucional y reconocida a nivel regional y nacional.

Por otro lado, debido al aumento de nuevos requerimientos y, por ende, el crecimiento de nuevos desarrollos, el tiempo de respuesta de dichos requerimientos es cada vez más exigente, la calidad en los productos se ha vuelto más que una necesidad un deber y la necesidad de estandarizar documentos y procesos que guíen al equipo de desarrollo se hace cada vez más evidente.

Es por esto que se hizo necesario que el área de T.I diseñara e implementara un proceso de desarrollo propio con ayuda de mejores prácticas, que le permitan superar las dificultades que se viven a la hora de gestionar, desarrollar o administrar aplicativos institucionales, como mejorar la calidad de los productos, mejorar la predictibilidad de los proyectos y que permita contar con un marco de trabajo que ayude hacer las cosas de la manera más eficiente. Todo esto con el fin de generar software de calidad, que satisfaga los requerimientos del cliente y de la Institución, que sirva como apoyo fundamental a sus procesos y permita aportar de forma significativa a la innovación, la competitividad, al

logro de metas institucionales y por ende sea un área generadora de valor dentro de la institución.

4 ANTECEDENTES

Se realizó una revisión preliminar de estudios e investigaciones publicadas relacionadas con los diferentes procesos de desarrollo y su aporte a la calidad del software. Entre estos estudios y de acuerdo con su ámbito regional se destacan:

A Nivel Regional:

Eduardo, L., & Valencia, P. (2012). Plantean el diagnóstico de la calidad del proceso de desarrollo de software a partir del levantamiento de información mediante un instrumento, en este caso una encuesta, con el fin de que dicho diagnóstico se convierta en un insumo relevante a la hora de proponer un modelo propio. Con dicho estudio se pudo concluir que el 85% de las empresas encuestadas utilizan un modelo o metodología de desarrollo y en su mayoría hacen una mezcla de varias creando una propia, casi todas apoyadas por RUP.

Britto, Jaime A. (2014). Presenta el diseño y adaptación de un proceso de desarrollo en el área de sistemas de Confamiliar Risaralda mediante la selección de mejores prácticas obtenidas de las metodologías de Scrum, XP e ICONIX. En este trabajo se propuso un nuevo proceso de desarrollo el cual fue aplicado con una prueba piloto y evaluado con respecto a CMMI, el resultado fue una mejora del 25% en el cumplimiento de prácticas frente a CMMI y se observa dificultades a la hora de aplicar prácticas relacionadas con documentación extensiva y toma de medidas.

A Nivel Nacional:

Blanco, Monica A. (2007). En conjunto con el grupo de investigación Qualdev Group de la Universidad de los Andes, plantean una estrategia completa para ayudar las empresas a implementar CMMI. Dicha estrategia incluyó un modelo de mejora, un a matriz de grados de logros de las prácticas y una herramienta de apoyo que facilite su puesta en marcha. El modelo ofreció a las empresas pequeñas de desarrollo de software un conjunto de artefactos que facilitan la implementación del modelo CMMI, haciendo su aporte significativo más aún para las empresas pequeñas de desarrollo de software que optan por no implementar un

modelo tan robusto como CMMI por la inversión en tiempo y recursos que se hace necesario hacer.

Ramírez, L., & Flórez, A. (2014). Plantean una serie de buenas prácticas basadas en metodologías ágiles para un mejor desarrollo de software, la cual abarca desde la obtención de requerimientos hasta la entrega al cliente y la seguridad de que éste está satisfecho, buscando así que tanto el equipo de desarrollo como el cliente estén en permanente contacto y trabajen en equipo. Finalmente, se llegó a la conclusión de que existen diferentes metodologías de desarrollo ágil que pueden ser adaptadas a un proyecto, pero existen proyectos en los cuales no se adapta ninguna metodología, por lo que se hace necesario investigar las necesidades del proyecto y tomar algunas prácticas estructuradas de dichas metodologías para ser ejecutadas en el proyecto.

Morales, J., & Pardo, C. (2016). Llevaron a cabo una revisión sistemática del desarrollo de software dirigido por modelos y las metodologías ágiles. Con el resultado de dicha revisión se logró identificar un alto interés tanto por el enfoque de desarrollo de software orientado por modelos como por la adaptación de éste al enfoque y/o procesos de desarrollo con enfoque ágil. Se concluyó que la industrialización de los procesos de desarrollo de software es necesaria y se reconoció los beneficios del desarrollo de software dirigido por modelos, así como el esfuerzo que se debe realizar en la definición, adaptación e implementación de soluciones y estrategias que permitan mejorar los procesos productivos en la industria de software.

A Nivel Internacional:

Pesado, P., Bertone, R. A., Esponda, S., Pasini, A. C., Boracchia, M., Martorelli, S., & Swaels, M. (2013). Realizaron una revisión de los modelos y normas orientados a la calidad del producto para las organizaciones prestadoras de servicios de software, como es la ISO/IEC 9126, ISO/IEC 15504, ISO/IEC 12207, ISO/IEC 2000, ISO/IEC 9001, ISO/IEC 90003, CMMI e ITIL. Con dicha revisión llegaron a la conclusión de que para que una empresa implemente alguno de estos modelos de calidad, es necesario realizar una

inversión financiera importante la cual se ve reflejada a largo plazo lo que hace que las direcciones lo vean como un gasto y no como inversión. En el caso de las Pymes, es aún más complejo acceder a este tipo de modelos, debido a ello han aparecido algunos como MoProSoft, Competisoft, ISO/IEC 29110 que se establece como estándar en las Mejoras de Procesos para PyMEs.

Elizabeth, M., & Ciudad, A. (2017). Con el apoyo del Instituto de Investigación Científica de la Universidad de Lima, llevó a cabo un estudio donde se caracterizaron las Pymes productoras de software de ciudad de Lima (Perú), se analizaron y clasificaron los modelos de calidad de mejora de procesos adecuados a categoría Pymes, se determinó que un modelo de mejora adecuado debe ser aplicable en tres aspectos: calidad, costo y tiempo. Finalmente, se demostró que las Pymes, aunque no tienen un uso adecuado de los modelos de calidad de *software*, están dispuestas a adoptar un modelo de referencia que permita mejorar sus procesos y generar productos con calidad.

Chavarría, A. E., Oré, S. B., & Pastor, C. (2016). Con el propósito de mejorar la calidad del proceso de desarrollo y del producto software, implementaron un modelo de aseguramiento de la calidad que integra CMMI (Capability Maturity Model Integration), TSP (Team Software Process y PSP (Personal Software Process) en una organización desarrolladora de software. Los resultados de dicha implementación demostraron mejoras significativas en la reducción de costes de calidad y la cantidad de defectos. Además, se evidenció aumento en la rentabilidad del proyecto y en la satisfacción del cliente.

David, J., González, Y., Jesús, C., & Calvache, P. (2015). Realizaron una revisión sistemática de la implementación de metodologías ágiles y otros modelos en micro, pequeñas y medianas empresas (MiPyMEs), el objetivo era conocer lo que se había realizado y logrado en este tipo de empresas respecto a las metodologías ágiles y otros modelos desde el punto de vista de: tendencias, propuestas, experiencias, factores de éxito, entre otros. Con dicha revisión se pudo observar un creciente interés en la aplicación e implementación de las metodologías ágiles, en empresas desarrolladoras de software y se

concluyó que es imposible que un solo modelo o estándar solucione todas las necesidades de una empresa, por ende, se hace necesario que las MiPyMEs implementen e institucionalicen múltiples prácticas a partir de múltiples enfoques, con el fin de solucionar múltiples necesidades y que además sea acorde a sus necesidades.

5 OBJETIVOS

5.1 OBJETIVO GENERAL

Implementar e implantar un proceso de desarrollo de software y adopción de buenas prácticas con el fin de mejorar la calidad del software del área de T.I de la Universidad de Manizales.

5.2 OBJETIVOS ESPECÍFICOS

- Diagnosticar el estado actual del proceso de desarrollo de software del área de T.I siguiendo los lineamientos de modelo IDEAL.
- Diseñar un proceso de desarrollo basado en una integración/adaptación metodológica de (de los) proceso (s) más idóneos según las necesidades de la organización.
- Implementar el proceso de desarrollo diseñado en un proyecto piloto.
- Identificar lecciones aprendidas y posibles mejoras enmarcadas con el estándar ISO / IEC 29110.

6 REFERENTE TEÓRICO

La implementación de una metodología de desarrollo de software y adopción de buenas prácticas en el área de T.I, integra diferentes conceptos descritos a continuación con el fin de proporcionar una visión general del presente trabajo, entre ellos se destacan los modelos de mejora de procesos, los modelos de procesos de software y los modelos de evaluación de procesos.

6.1 MODELOS DE MEJORA DE PROCESOS

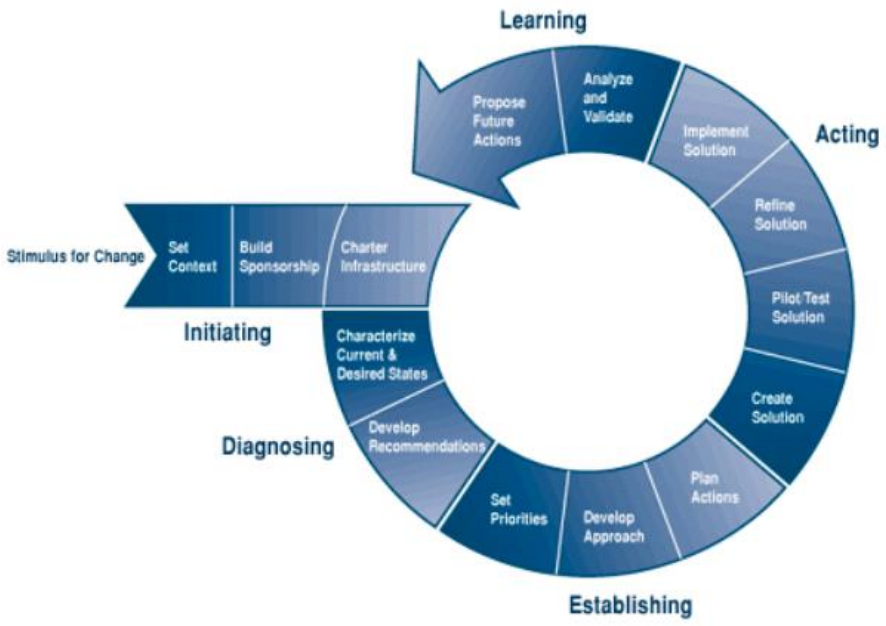
Un modelo de mejora de proceso es un modelo que describe las actividades necesarias para la evolución de los procesos de la organización y tiene como propósito proveer una guía para mejorar los procesos de desarrollo, la capacidad de gestionarlos y la adquisición y el mantenimiento de productos de software.

6.1.1 Modelo IDEAL

El modelo IDEAL (Ideal Model et al., 2001) es un modelo de mejora organizativa elaborado por el SEI que sirve como guía para la iniciación, planificación e implementación de acciones de mejora para el proceso de software en las organizaciones.
El modelo establece los fundamentos para una estrategia a largo plazo basado en siguientes cinco fases:

- Iniciar
- Diagnosticar
- Establecer
- Actuar
- Aprender

Figura 1: Modelo IDEAL



Fuente: (Ideal Model et al., 2001)

Fase 1 - Iniciar

Este es el paso inicial del modelo IDEAL, en esta fase es donde la dirección entiende la necesidad de mejora, se establece la infraestructura inicial de mejora, se definen las funciones y responsabilidades y se asignan los recursos iniciales. Se crea el plan SPI (Mejora de Procesos de Software) que guía la organización a través de la completitud de las fases de inicio, diagnóstico y establecimiento. Sobre este plan también se acuerdan y definen las interfaces y los requisitos de alto nivel.

El propósito de esta fase es establecer el contexto y apoyo para el programa SPI, conocer los costos y beneficios del programa y comprometer los recursos necesarios para implementar y gestionar el programa.

Actividades de Fase Iniciar:

1. Fijar Contexto
2. Asegurara el patrocinio o apoyo

3. Establecer Infraestructura.

Fase 2 – Diagnosticar

El objetivo de esta fase es obtener el entendimiento completo del trabajo que se debe realizar, para esto es necesario conocer y caracterizar el estado actual y futuro de la organización. Este diagnóstico generalmente se realiza con un modelo de referencia como CMMI o como en este caso la norma ISO 29110.

Como resultado del diagnóstico son los hallazgos finales y se proponen recomendaciones que sirven de base para definir las actividades siguientes y que influyen en la toma de decisiones de la alta gerencia.

Actividades de Fase Diagnosticar:

1. Describir estado actual y deseado
2. Elaborar recomendaciones

Fase 3 - Establecer

El objetivo de esta fase es la elaboración o actualización de un plan de acción estratégico para la mejora de procesos de software (SPI) por parte del equipo de administración, basado en la visión de la organización, el plan de negocios junto con los resultados de los hallazgos y el diagnóstico de la fase anterior.

Primero se debe definir las prioridades para el esfuerzo de mejora, luego se identifica el enfoque a seguir teniendo en cuenta las prioridades, los hallazgos y el resultado del diagnóstico, finalmente, se definen la métrica que van a medir el progreso alcanzado y se define y capacita el grupo de técnicos que desarrollaran el proceso.

Actividades de Fase Establecer:

1. Establecer Prioridades
2. Aproximar solución
3. Planear acción

Fase 4 - Actuar

En esta fase es donde las mejoras se desarrollan, se ponen en práctica y se implementan en toda la organización, por esto suele ser la fase que más tiempo y recurso consume. La fase empieza con la definición de la solución que cubren los objetivos de la organización, dicha solución comprende herramientas, habilidades, procesos, información y asesoría la cual es desarrollada por el grupo de técnico establecido en fases anteriores.

La solución propuesta primero se lleva a cabo en proyectos piloto los cuales dan como resultados experiencia, conocimiento y lecciones aprendidas que posteriormente permiten mejorar y refinar la solución. El proceso itera hasta obtener una solución satisfactoria, finalmente la solución obtenida se empieza a implantar en la organización.

Actividades de Fase Actuar:

1. Crear la solución
2. Piloto / Prueba de la solución
3. Refinar la solución
4. Implementar la solución

Fase 5 - Aprender

Entrar a esta fase indica que la organización ha completado el ciclo IDEAL, y es necesario revisar lo sucedido durante todo el ciclo y prepararse para el siguiente a través del modelo. El objetivo de esta fase es completar el plan y garantizar que el próximo ciclo sea más efectivo y eficiente. Se revisa toda la información recolectada en las fases anteriores y se evalúa los objetivos y logros alcanzados con el fin de implementar un cambio más efectivo y eficaz en el futuro. Finalmente, se re-evalúan las metas de negocio, se verifica el cumplimiento y se propone mejoras para el siguiente ciclo.

Actividades de Fase Aprender:

1. Análisis y validación
2. Propuestas y acciones futuras

6.1.2 Agile SPI Process

Agile SPI (Pardo et al., 2010), es un proceso ágil y liviano que está basado en el proceso IDEAL, es sencillo de leer y de interpretar, permite definir el qué el cómo hacerlo y es aplicable a micros, pequeñas y medianas empresas. Agile SPI está caracterizado por:

- Contiene mini-ciclos o iteraciones.
- Guía la mejora de los procesos de desarrollo de software, manteniendo el nivel de agilidad que la empresa desee.
- Adapta principios y características de modelos y metodologías de gestión para el desarrollo ágil y liviano.
- Se adapta a una industria en constante cambio, dinámica y creativa como es la industria del software.

El modelo establece los fundamentos para una mejora de procesos de software basado en siguientes cinco fases:

- Instalación
- Diagnóstico
- Formulación
- Mejora
- Revisión

Figura 2: Modelado bajo SPEM de las fase 1

Fuente: (Pardo et al., 2010)

Fase 1 - Instalación

En esta fase se definen los objetivos y se crea una propuesta basada en las necesidades del negocio, la cual servirá de guía para la definición de las actividades de las siguientes fases.

Fase 2 - Diagnóstico

En esta fase realizan actividades de valoración con el fin de conocer el estado general de los procesos de la organización. Adicionalmente, se realizar una priorización de actividades de los casos de mejorar con base en el análisis de los resultados realizado.

Fase 3 - Formulación

En esta fase se planifica una primera iteración de mejora de acuerdo a los casos de mejora que tienen mayor prioridad, esto con el fin de medir el esfuerzo que sirva de base para la estimación del esfuerzo de las siguientes fases del proyecto.

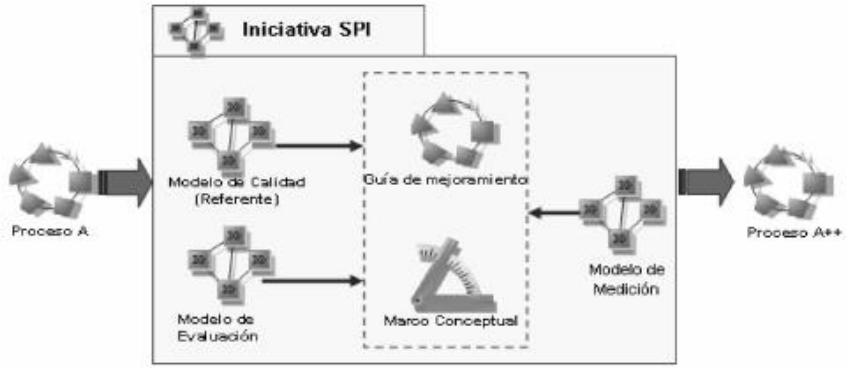
Fase 4 - Mejora

En esta actividad se gestiona y lleva a cabo todo el esfuerzo de los casos de mejora. Para esto, se retoman las planificaciones correspondientes a las diferentes iteraciones y se ejecutan. Se crea un documento donde se consigna la ejecución que se lleva a cabo, de los pilotos de prueba, la evaluación de lo nuevo o la nueva mejora realizada. En el caso de que el piloto se haya desarrollado con éxito, se deben crear planes de aceptación de los nuevos procesos en la empresa.

Fase 5 - Revisión

En esta fase se realiza una retroalimentación de todo el ciclo de mejora, esto con el fin de identificar y consignar lecciones aprendidas, métricas desarrolladas y así medir el cumplimento de los objetivos. Con toda la información recolectada se debe evaluar el trabajo realizado y ajustar los elementos relacionado con la ejecución del ciclo, como los métodos utilizados, los canales de comunicación, la infraestructura establecida y soluciones a los problemas identificados.

Figura 3: Arquitectura conceptual de Agile SPI Process



Fuente: (Pardo et al., 2010)

6.2 MODELOS DE REFERENCIA

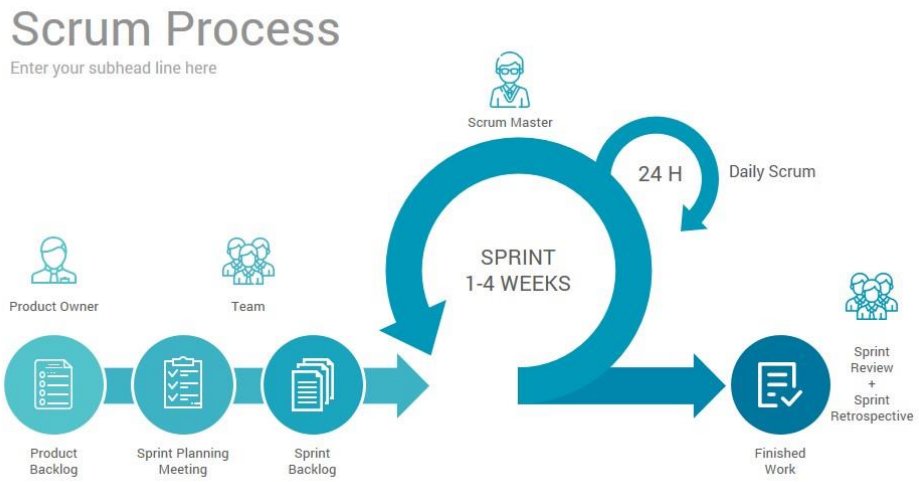
Un modelo de referencia es un modelo a seguir para el establecimiento de un proceso para el desarrollo de software. Dichos procesos incluyen actividades, roles, reglas, políticas, estructuras de organización, componentes de software y herramientas utilizadas para definir, crear y desarrollar productos de software. Entre los diferentes modelos de referencia de software se destacan:

6.2.1 SCRUM

"Scrum es un marco de trabajo para el desarrollo y el mantenimiento de productos complejos" (Schwaber, 2013). Scrum ha sido utilizado desde los años 90 para gestionar productos complejos, está enfocado en la satisfacción del cliente a través de la reducción de la complejidad en el desarrollo de productos de software y puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto.

Scrum "consiste en los Equipos Scrum y sus roles, eventos, artefactos y reglas asociadas. Cada componente dentro del marco de trabajo sirve a un propósito específico y es esencial para el éxito de Scrum y para su uso" (Schwaber, 2013). Está basado en a la auto-organización, esto con el fin de resolver problemas complejos y adaptarse a lo imprevisible.

Figura 4: Proceso Scrum



Fuente: (Frechina et al., 2018)

Equipo o Roles en Scrum

Un equipo Scrum está compuesto por un grupo de 3 a 9 integrantes. Los miembros del equipo son auto organizados y multifuncionales, esto permite que elijan la mejor forma de llevar a cabo su trabajo y sean tan competentes para su labor que no tengan que depender de otra persona. Está compuesto por:

El Dueño de Producto (Product Owner)

El dueño del producto es quien se encarga de que el equipo Scrum trabaje desde la perspectiva del negocio y es el responsable de maximizar el valor del producto. Es la única persona responsable de gestionar la Lista del Producto o Product Backlog (artefacto Scrum descrito más adelante). Las decisiones del Dueño de Producto se reflejan en el contenido y en la priorización de la Lista del Producto por ende y para que pueda realizar bien su trabajo, la organización debe respetar sus decisiones.

El Equipo de Desarrollo (Development Team)

El equipo de desarrollo tiene la responsabilidad de desarrollar y entregar el producto. La organización es la encargada de estructurar el equipo de desarrollo. Son autónomos y multidisciplinares, esto quiere decir que requieren muy poco de personas externas al equipo y disponen de las habilidades necesarias para poder identificar y ejecutar todas las tareas. Se recomienda que sea conformado por un pequeño equipo de 3 a 9 personas con habilidades transversales necesarias para realizar el trabajo (análisis, diseño, desarrollo, pruebas, documentación, etc.).

Scrum Master

El Scrum master es el facilitador del scrum, no es el líder pues el equipo se autorregula, pero es quien se encarga de actuar como protector entre el equipo y cualquier influencia que le distraiga, es quien se asegura de que el proceso y las reglas se cumplan.

Eventos Scrum

En Scrum existen una serie de eventos que se crearon con el fin de regularizar las reuniones, por lo general, las duraciones de los eventos dependen de si se alcanzaron los objetivos de dicha actividad, a excepción del Sprint que tiene duración fija y no puede acortarse o alargarse. "La falta de alguno de estos eventos da como resultado una reducción de la transparencia y constituye una oportunidad perdida de inspección y adaptación. "(Schwaber, 2013). Estos eventos se describen a continuación:

Scrum Diario (Daily Scrums)

Los Scrum Diarios tiene una duración de 15 minutos en los cuales el Equipo de Desarrollo sincroniza sus actividades y crea un plan para las siguientes 24 horas.

Por lo general, el Scrum diario se lleva a cabo a la misma hora y en el mismo lugar, en la cual cada miembro del equipo debe responder las siguientes preguntas:

- ¿Qué hice ayer para lograr el Objetivo del Sprint?
- ¿Qué haré hoy para lograr el Objetivo del Sprint?
- ¿Veo algún impedimento que evite que se logre el Objetivo del Sprint?

Las responsabilidades del Scrum Master en este evento es asegurar que el equipo de desarrollo tenga la reunión, mantener el Scrum diario en los límites del bloque de 15 minutos y mantener las reglas, el equipo de desarrollo son quienes dirigen el Scrum.

Planificación de la Iteración (Sprint Planning)

En el Sprint Planning se planifica el trabajo que se realizará en el Sprint con la colaboración de todo el equipo de Scrum.

La duración del Sprint planning es de máximo 8 horas para un Sprint de un mes, y si el Sprint es más corto, el evento suele ser más corto.

El Sprint Planning se realiza para responder a las siguientes preguntas:

- ¿Cuál va a ser el incremento de valor del próximo Sprint?
- ¿Cómo se logrará el trabajo necesario para lograrlo?

Revisión de la Iteración (Sprint Review)

El Sprint Review ocurre al final de Sprint, en este sprint el dueño del producto y el que presentan a los interesados o stakeholders el incremento terminado para su debida revisión. El software ya ha sido validado previamente por el dueño del producto antes de esta reunión. En esta reunión los interesados aprovechan para realizar las preguntas que crean necesarias sobre el producto. Por lo general, son los miembros del equipo quienes muestran el incremento.

Retrospectiva de la Iteración (Sprint Retrospective)

El Sprint Retrospective ocurre al final de Sprint Review, en él se hace un análisis del último Sprint y se identifican posibles mejoras para el próximo sprint.

Artefactos Scrum

Los artefactos Scrum son elementos que forman parte del marco de trabajo Scrum los cuales están compuestos por:

Lista de Producto (Product Backlog)

La lista de producto es una especie de inventario en donde se consignan los requisitos del sistema en forma de una lista ordenara por prioridad. La lista de productos evoluciona a medida que lo hace el producto, por esto se dice que es dinámica, "cambia constantemente para identificar lo que el producto necesita para ser adecuado, competitivo y útil. Mientras el producto exista, su Lista de Producto también existe". (Schwaber, 2013).
El Dueño de Producto (Product Owner) es el responsable de esta lista, de su contenido, disponibilidad y ordenación, los integrantes del equipo de desarrollo son los que realizan todas las estimaciones (tiempo, costo...). El Dueño de Producto podría influenciar al Equipo ayudándoles a entender y seleccionar las prioridades, pero las personas que harán el trabajo son las que hacen la estimación final.

Lista de Pendientes del Sprint (Spring Backlog)
La Lista de Pendientes del Sprint es una especie de predicción realizada por el Equipo de Desarrollo acerca de las funcionalidades que formará parte del próximo Incremento y del trabajo necesario para entregar esa funcionalidad en un Incremento "Terminado". La lista de pendientes solo se puede modificar durante el Sprint y a medida que va avanzando el proyecto va apareciendo nuevo trabajo el cual se va añadiendo a dichas lista por parte del Equipo de Desarrollo.

Incremento
Los incrementos representan los requisitos que ya fueron completados en una iteración y por lo tanto están en condiciones de ser utilizados. Al final de un sprint el nuevo incremento debe estar terminado.

6.2.2 RUP

El Rational Unified Process o Proceso Unificado de Racional es un proceso de desarrollo de software orientado a objetos el cual fue desarrollado y comercializado inicialmente por la Rational Software y ahora por IBM, junto con UML conforman la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos. Su objetivo es asegurar la producción de software de alta y de mayor

calidad para satisfacer las necesidades de los usuarios que tienen un cumplimiento al final dentro de un límite de tiempo y presupuesto previsible.

Características

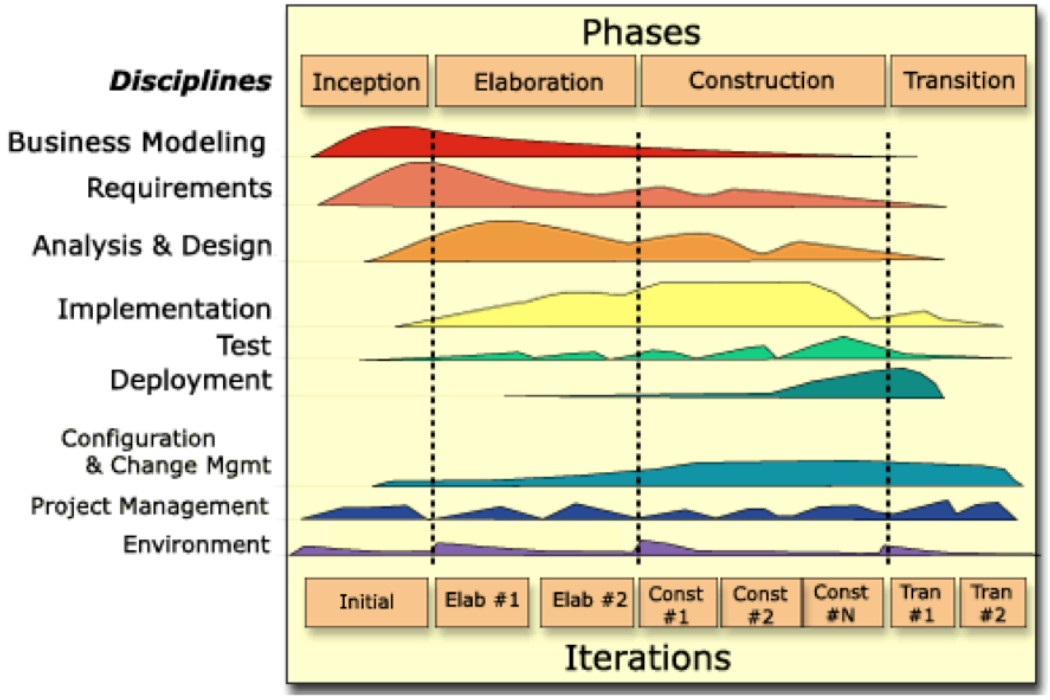
- Permite reducir la complejidad del mantenimiento por la facilidad de los cambios.
- Es indicado para desarrollar proyectos grandes y complejos.
- Es orientado a objetos y centrado en la arquitectura
- Facilidad para la construcción de prototipos.
- Se adapta a las necesidades del cliente
- Proceso iterativo e incremental.

El Proceso unificado tiene dos dimensiones:

Estructura dinámica: la dimensión horizontal representa la estructura dinámica o la dimensión de tiempo del proceso, muestra como el proceso, expresado en términos de ciclos, fases, iteraciones se despliega sobre el ciclo de vida de un proyecto.

Estructura estática: la dimensión vertical representa la estructura estática del proceso. Describe como los elementos del proceso, actividades, disciplinas, artefactos y roles son lógicamente asociados en el núcleo del proyecto o flujos de trabajos.

Figura 5: Ciclo Vida RUP



Fuente: The rational unified process made easy: a practitioner's guide to the RUP

La estructura dinámica va de la mano con el ciclo de vida o la dimensión del tiempo de un proyecto. RUP provee una estructura enfocada en desarrollo iterativo, dividiendo el proyecto en cuatro fases:

- Iniciación o Concepción: énfasis en el alcance del sistema.
- Preparación: énfasis en la arquitectura.
- Construcción: énfasis en el desarrollo.
- Transición: énfasis en la aplicación.

Fase de Iniciación o Concepción

En la fase de iniciación o concepción se establece un buen entendimiento de lo que va a hacer el sistema a construir para entender a un alto nivel los requerimientos y establecer el alcance. Se establece el acuerdo de las partes interesadas con los objetivos, la arquitectura y la planificación del proyecto. Si estos actores tienen un buen conocimiento, no será necesario analizar. De lo contrario, se requiere un análisis más elaborado.

En esta etapa, se transforman los requisitos esenciales del sistema en los casos de uso. Lo que se busca es definir los riesgos y los costos del negocio, para definir si es factible continuar con el proyecto. Lo ideal es realizar iteraciones, las cuales deben estar bien definidas en cuanto a su importancia y objetivos.

Los objetivos de la fase de iniciación o concepción son:

- Definir los límites del proyecto y su ámbito.
- Definir los casos de uso críticos del sistema.
- Tener un borrador de arquitecturas candidatas para los escenarios principales.
- Estimar los costos en recurso y tiempo del proyecto.
- Estimar los riesgos.

Fase de Elaboración

El objetivo principal de esta fase es elaborar la arquitectura del sistema. Se debe tener especial atención en las dificultades técnicas que se puedan presentar en el diseño, implementación, pruebas y la línea base como ejecutable de la arquitectura.

En esta etapa es donde el proyecto comienza a tomar forma. Al final de esta fase el equipo debe estar en la capacidad de decidir si se puede construir un sistema que funcione según los requerimientos.

Los objetivos de esta fase son:

- Definir, validar y establecer la arquitectura propuesta.
- Completar la visión.
- Crear un plan fiable para la fase de construcción el cual puede evolucionar en diferentes iteraciones.
- Comprobar que la arquitectura seleccionada es la adecuada y que soportará la visión con un costo y tiempo razonable.

La etapa de elaboración es considerada como un hito, si el proyecto no pasa este hito es posible que sea cancelado o rediseñado, por ende, su importancia, ya que se pueden tomar decisiones importantes.

Fase de Construcción

En la fase de construcción se inicia el desarrollo físico del sistema usando la arquitectura creada en la fase anterior. Es donde se hace la mayor parte de la codificación y las pruebas. Al final de esta etapa se deben tener una versión beta para pruebas. El producto debe ser estable para que pueda ser probado por los usuarios finales y así no se retrase la etapa de transición.

Durante esta fase todos las características, componentes y requerimientos deben ser implementados, integrados y probados, obteniendo una versión aceptable del producto.

Los objetivos de esta fase son:

- Optimizar recursos con el fin de minimizar costos.
- Construir versiones funcionales (alfa, beta, y otras versiones de prueba).

Fase de Transición

Esta es la fase de despliegue del software. Se enfoca en la transición del sistema desde la etapa de desarrollo hasta la puesta en producción, por esto, se debe asegurar que se cumple con todos los requerimientos y necesidades del cliente, esto incluye pruebas de producto, evaluación y retroalimentación de los mismos. Esto último solo como parte de refinamiento del sistema ya que en este punto se debe cumplir con todas las necesidades de los usuarios.

Los objetivos de esta fase son:

- Un producto final que cumpla los requisitos esperados, que funcione y satisfaga suficientemente al usuario.

Flujo de Trabajo RUP

RUP se agrupa en 9 flujos de trabajo principales, los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo, dichos flujos van transversales a las fases las cual van desarrollando cada uno de estos flujos si este le concierne.

Modelo del Negocio*:* el flujo de modelo de negocio describe como su nombre lo indica, los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.

Requerimiento*:* se define qué es lo que el sistema debe hacer, teniendo en cuenta los requerimientos funcionales y las restricciones establecidas.

Análisis y Diseño*: se* describe y define cómo se va a realizar el sistema a partir de los requerimientos y restricciones establecidas en el flujo anterior.

Implementación: se define cómo deben estar organizados las clases y objetos en componentes, la ubicación en ellos de los componentes y la estructura de capas de la aplicación.

Prueba: este flujo pretende encontrar defectos o errores durante todo el ciclo de vida.

Instalación o despliegue: en este flujo se realiza actividades de empaque, instalación, asistencia a usuarios, etc., para entregar el software a los usuarios finales.

Administración del proyecto: se realizan actividades de administración de proyectos con las que se busca producir un producto que satisfaga las necesidades de los clientes.

Administración de configuración y cambios: se define cómo se va a controlar los cambios y la forma de llevar el control de versiones, etc.

Ambiente: describe las actividades, los procesos y las herramientas que soportarán el equipo de trabajo del proyecto.

Estructura Estática de RUP

RUP define cuatro elementos que responden a las preguntas ¿Quién?, ¿Cómo?; ¿Qué? y ¿Cuándo? Estos cuatro elementos son:

- Roles - ¿Quién?
- Actividades – ¿Cómo?
- Artefactos – ¿Qué?

- Flujos de trabajo de las disciplinas - ¿Cuándo?

Roles

Un ***rol*** define el comportamiento y las responsabilidades de un individuo, o un grupo de individuos trabajando juntos como un equipo. Es como un sombrero que una persona lleva durante un proyecto, un individuo puede llevar diferentes sombreros. En RUP un rol define como un individuo va a realizar su trabajo en un proyecto determinado, y un individuo puede realizar más de un rol.

Un rol tiene la responsabilidad de llevar a cabo un conjunto de actividades y de ser dueño de un conjunto de artefactos.

Figura 6: Roles, Actividades y Artefactos

Fuente: https://www.cscjournals.org/manuscript/Journals/IJSE/Volume5/Issue2/IJSE-142.pdf (pg. 13)

Actividades

Una actividad de un rol específico es una unidad de trabajo del individuo. Es una tarea que tiene un propósito claro y es realizada por un trabajador.

Una actividad tiene un propósito claro usualmente expresado en términos de creación o actualización de un artefacto como un modelo, plan o componente. Cada actividad es asignada a un rol específico quien se encarga de ejecutarla y generalmente lleva pocas horas o días para ser terminada. Una actividad debe tener la duración adecuada ya que, si es

demasiado pequeña será abandonada y si es demasiado grande será difícil de desarrollar y terminar, en estos casos se recomienda dividirlas por partes.

Artefactos

Un artefacto es un producto tangible del proyecto. Es información que se produce, modifica y utiliza el proyecto. Son producidos por un individuo dentro de un rol y a su vez es la salida de las actividades asignadas a dicho individuo.
Los artefactos pueden ser de ingeniería o de gestión.

6.2.3 OPEN UP

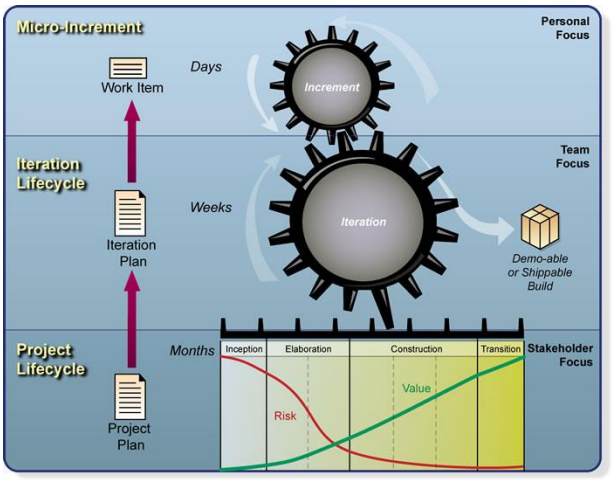
OpenUP es un proceso de desarrollo de software con un enfoque ágil, iterativo e incremental dentro del ciclo de vida estructurado, consta de cuatro fases: Inicio, elaboración, construcción y transición, se centra en la naturaleza colaborativa del desarrollo de software.

OpenUP divide los proyectos en iteraciones que constan de intervalos de tiempo planeado, generalmente medidos en semanas. Las iteraciones se enfocan en una entrega de valor por parte del equipo. Los equipos de OpenUP se auto organizan para lograr los objetivos de cada iteración y se comprometen a entregar los resultados.

El ciclo de vida del proyecto proporciona a los interesados y miembros del equipo puntos de visibilidad para toma de decisiones a lo largo del proyecto.

Las iteraciones en OpenUP mantienen al equipo enfocado en entregar un producto de valor en pocas semanas el cual va en incremento conforme pasan las iteraciones. “Esto crea un enfoque saludable para garantizar que todo lo que se trabaje sea valioso para las partes interesadas” (Eclipse Process Framework, 2004-2007).

Figura 7: Capas OpenUP: micro-incrementos, iteraciones de ciclo de vida y ciclo de vida del Proyecto



Fuente: (Eclipse Process Framework et al., 2004-2007)

El desarrollo iterativo se enfoca en reducir el riesgo, esto se hace dando una demostración del código de trabajo frecuentemente que permitan que las correcciones del curso se tomen según sea necesario.

"La planificación de la iteración, la estimación y el seguimiento del progreso se centran en los elementos de trabajo" (Eclipse Process Framework, 2004-2007). Se utiliza la estimación ágil para saber cuántos elementos de trabajo lleva una iteración y que permita al equipo cumplir con los objetivos acordados en la iteración. Una iteración comienza con una reunión de planificación de iteración que dura unas pocas horas.

OpenUP planifica iteraciones en un conjunto de fases. Cada fase termina con un hito el cual tiene el propósito ser supervisado por las partes interesadas.

Fase de Inicio

El propósito de la fase de inicio es encontrar equilibrio entre las partes interesadas en los objetivos del ciclo de vida del proyecto. El equipo comprende qué es lo que se desea construir, se define el objetivo general del proyecto, se identifica a los interesados, se identifica los requisitos a través del mecanismo determinado por el equipo y por el dueño

del producto y finalmente se define una posible arquitectura a usar. Generalmente, la fase de inicio tiene una iteración, aunque esto también depende del tamaño del proyecto

Los objetivos de la fase de inicio son:

- Determinar la visión, el alcance del sistema y sus restricciones.
- Identificar los requisitos críticos del sistema.
- Identificar una posible solución, una arquitectura candidata.
- Estimar costos, tiempos y riesgos del proyecto

Fase de Elaboración

El objetivo de esta fase es mitigar los riesgos técnicos y no técnicos significativos y definir la arquitectura del sistema. Una forma de mitigar los riegos es detallando los requisitos y diseñando, implementando y probando la arquitectura

El equipo de trabajo debe obtener un entendimiento más detallado de los requerimientos del sistema y una "definición clara y precisa de los casos de uso, los actores, la arquitectura del sistema y un prototipo ejecutable de la misma". (Ríos 2013). SE detallan los casos de uso más relevantes del sistema.,

Lo objetivos de la fase de elaboración son:

- Obtener un entendimiento más detallado de los requerimientos.
- Diseñar, implementar, validar y establecer la línea de base para la arquitectura.
- Mitigar los riesgos principales.

Fase de Construcción

El objetivo de la fase de construcción es desarrollar de manera rentable un producto con características completas, incluidos los aspectos de diseño, implementación y prueba. Todos los componentes y funcionalidades del sistema se deben de realizar, probar e integrar. Se realizan múltiples iteraciones con el fin de entregar un sistema parcial en cada de ellas, el equipo selecciona los requisitos para implementar según las prioridades dadas

por el dueño del producto. Finalmente, las iteraciones terminan cuando el sistema esté completo.

Lo objetivos de la fase de construcción son:

- Optimizar los recursos y minimizar los costos.
- Desarrollar de manera iterativa un producto completo que esté listo para su implementación.

Fase de Transición

Esta fase es la final del proyecto. El objetivo de la fase de transición es asegurar que el software esté listo para ser entregado a los usuarios y que cumpla con todas sus expectativas, es posible que continúe otro ciclo de vida del proyecto para comenzar otra versión del producto.

Un elemento importante de la fase de transición son las lecciones aprendidas para mejorar el proceso y el desempeño que tuvo el equipo durante todo el ciclo de vida del producto.
Lo objetivos de la fase de transición son:

- Realizar pruebas para validar que se cumplan las expectativas del cliente.
- Realizar pruebas de aceptación del producto con el fin de lograr concurrencia entre las partes interesadas.
- Documentar las lecciones aprendidas para mejorar el rendimiento del proyecto futuro.

Fase de Disciplina

Para OPENUP una disciplina es un conjunto de tareas que están relacionadas con un área importante dentro de un proyecto que pretenden lograr un objetivo o realizar tareas de trabajo.
Las disciplinas de OPENUP son:

- Arquitectura
- Configuración y administración de cambios

- Desarrollo
- Administración de proyectos
- Requerimientos
- Pruebas

Arquitectura

El objetivo de esta disciplina es desarrollar una arquitectura robusta para el sistema.

Tareas

- ***Delinear la arquitectura***

  El propósito de esta tarea es identificar los objetivos arquitectónicos para una iteración que guiará el desarrollo y las pruebas. Se centra en la recopilación de la experiencia adquirida en sistemas similares o en lecciones aprendidas de otros proyectos con el fin de direccionar al equipo comience y continúe evolucionando la arquitectura.

- ***Perfeccionar la arquitectura***

  El objetivo de esta tarea es tomar las decisiones arquitectónicas necesarias para respaldar los objetivos de la iteración actual del proyecto las cuales deben ser concretas e inequívocas.

Configuración y Gestión de Cambios

El objetivo de esta disciplina es mantener versiones de artefactos y configuraciones consistentes y realizar una adecuada gestión de cambios. La configuración y gestión de cambios es realizada por todos los miembros del equipo, todas las demás disciplinas se basan en la configuración y gestión de cambios para mantener un conjunto de productos coherentes y actualizado,

Tareas

- ***Solicitar Cambios***

  En esta tarea se recopilan la información de la solicitud de cambio y se actualizan las listas de elementos de trabajo.

- ***Integrar y Crear***

Los objetivos de esta tarea es integrar todos los cambios realizados por todos los desarrolladores e identificar los problemas de integración para que puedan ser corregidos por la persona y el momento adecuado.

Desarrollo

El objetivo de esta disciplina es construir el sistema incrementalmente, lo cual se consigue transformando los requerimientos en un diseño y ese diseño llevarlo a la implementación en un entorno de desarrollo. Cada iteración de esta disciplina traerá una compilación del producto más completa y estable.

Tareas

- ***Implementar pruebas de desarrollador***

  La prueba del desarrollador consiste en probar el comportamiento esperado de las unidades de código. A medida que el código crece también lo hacen las pruebas de desarrollador.

- ***Implementar la solución***

  El objetivo de esta tarea es crear una implementación que sea parte de la solución, o corregir uno o más defectos.

- ***Ejecutar pruebas***

  El objetivo de esta tarea es comprobar que la haga lo que se especificó en los requerimientos y el diseño.

- ***Diseñar la solución***

  El objetivo de esta tarea es identificar y describir los elementos del sistema para que admitan el comportamiento requerido, sean de alta calidad y se ajusten a la arquitectura.

Administración de Proyectos

Los objetivos de esta disciplina es estimular al equipo para que realice entregas continuas de software, crear planes a corto y largo plazo para el proyecto, gestionar el riesgo del proyecto, fomentar la priorización del trabajo y crear un entorno de trabajo efectivo para maximizar la productividad del equipo.

Tareas

- ***Evaluar los Resultados***

  La evaluación consiste en discutir con el equipo sobre los resultados de la iteración, el producto, sobre lo que salió bien y mal durante la iteración, intercambiar ideas nuevas y adaptar el conocimiento nuevo. Dicha evaluación debe ser coordinada del Gerente del proyecto y su finalidad es tener herramientas para tomar decisiones para la próxima iteración y determinar el mejor plan de acción para el proyecto

Requerimientos

Los objetivos de esta disciplina es comprender el problema a resolver, las necesidades de los interesados, definir los requisitos y el alcance de la solución, identificar restricciones y proporcionar las bases iniciales para estimar costos y cronograma.

Tareas

- ***Definir Visión***

  La visión consiste en describir el problema y sus características basadas en las solicitudes, necesidades o problemas que expresan las partes interesadas, con el fin de que el equipo del proyecto entiende mejor que es lo que debe hacer y lo que debe atender.

- ***Detallar los Requerimientos***

  El objetivo de esta tarea es detallar uno o más requerimientos con el nivel de detalle necesario para ser comprendidos por los miembros del equipo y los interesados, con esto llegar a un acuerdo y finalmente empezar con el desarrollo.

Pruebas

Los objetivos de esta disciplina es tener una retroalimentación temprana y con una frecuencia moderada del cumplimento de los requerimientos en el sistema, identificar problema de forma temprana con requisitos, diseños e implementaciones y garantizar que los cambios introducidos no generen nuevos errores.

Las pruebas se deben realizar en cada iteración del ciclo de vida del desarrollo, es normal que una iteración tenga muchos ciclos de prueba.

Tareas

- ***Crear Casos de Prueba***

  Consiste en desarrollar casos de prueba para probar los requerimientos del sistema.

Artefactos

Los artefactos o productos de trabajo en OPEN UP se organizan por dominios.

Arquitectura

La lista de artefactos relacionados con la arquitectura es:

Notas de Arquitectura

Este artefacto describe como se debe construir el sistema. En él se consignan las decisiones, fundamentos, supuestos, explicaciones e implicaciones de la formación de la arquitectura. Los arquitectos deben utilizar este artefacto para colaborar con otros miembros del equipo en el desarrollo de la arquitectura, y para ayudar a los miembros del equipo a la toma de decisiones. El arquitecto debe informar cómo está organizado el sistema al Project Manager y otros miembros del equipo, esto con el fin de que puedan adaptarse a las necesidades del sistema.

Desarrollo

La lista de artefactos relacionados con el desarrollo es:

Diseño

Este artefacto describe los elementos que conforman el sistema para que puedan ser revisados y entendidos de una manera sencilla.

Construcción

Este artefacto está compuesto por una cantidad de archivos que se generan después del proceso de compilación de un sistema implementado.

Pruebas de Desarrollo

Este artefacto está compuesto por la documentación de las pruebas de desarrollo que se llevan a cabo para evaluar o validar un componente del sistema. En dicho artefacto se especifica las entradas, condiciones de ejecución, resultados esperados y las salidas.

Implementación

Este artefacto es la reunión de uno o más de los siguientes elementos:

- Archivos de código fuente
- Archivos de información
- Scripts

Otros archivos que se transforman en el sistema ejecutable.

Gestión de Proyectos

La lista de artefactos relacionados con la gestión de proyectos es:

Plan de Iteración

Los objetivos del plan de iteración es brindar al equipo un lugar central para obtener información sobre los objetivos de la iteración, las tareas asignadas, los resultados de la evaluación y el progreso de la iteración.

Plan de proyecto

El objetivo de este artefacto es facilitar un documento el cual puede ser consultado por cualquier miembro del equipo del proyecto y en el cual se puede encontrar información sobre cómo se gestionará el proyecto.

El gerente de proyecto es el responsable de crear el plan de proyecto en conjunto con los miembros del equipo, este permite que las partes interesadas y miembros del equipo estén al tanto del proyecto y de consultar y registrar las lecciones aprendidas.

Lista de riesgos

En este artefacto se consignan y priorizan los riesgos asociados al proyecto

Lista de elementos de trabajo

Este artefacto está conformado por una lista la cual contiene:

- Todo el trabajo programado que se realizará dentro del proyecto y que debe priorizarse y posteriormente estimarse.
- Todas las solicitudes de mejoras o adiciones al sistema.

Requerimientos

La lista de artefactos relacionados con los requerimientos es:

Especificación de Requisitos

En este artefacto se capturan los requisitos de calidad, funcionales globales y todos aquellos que no son capturados en escenarios o casos de uso.

Visión

Este artefacto proporciona una visión completa del sistema. Contiene un resumen de los requisitos básicos previstos para el sistema y la definición de la vista de las partes interesadas del producto a desarrollar.

Casos de Usos

Este artefacto tiene como objetivo capturar el comportamiento requerido del sistema desde la perspectiva del usuario final con el fin de para producir un resultado de valor. Los clientes lo usan para aprobar el comportamiento del sistema, los usuarios para entenderlo, los arquitectos para identificar funcionalidades arquitectónicamente significativas y los administradores para planificar y evaluar el trabajo en cada iteración.

Glosario

Este artefacto define los términos importantes utilizados por el proyecto, los cuales, son la base para una colaboración efectiva entre las partes interesadas y los miembros del equipo. Su objetivo principal es proveer un vocabulario que sea entendido entre todos los interesados.

Modelo de Casos de Usos

Este artefacto muestra una visión general del comportamiento del sistema. Actúa como un contrato entre las partes interesadas y el equipo de proyecto, en el cual, se acuerdan las funcionalidades previstas para el sistema.

Pruebas

La lista de artefactos relacionados con las pruebas es:

Casos de Pruebas

En este artefacto se especifican un conjunto de entradas de prueba, condiciones de ejecución y resultados esperados con el fin de determinar si un requisito es completamente satisfactorio.

Log de Pruebas

En este artefacto se guardan un registro detallado generado a partir de la ejecución y realización de una prueba de software, el cual sirve para comprobar que se ejecutaron una serie de pruebas y cuál fue su resultado.

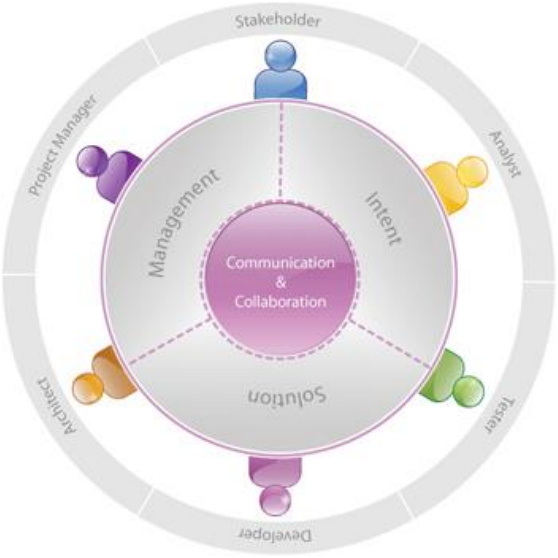
Script de Pruebas

En este artefacto contiene un paso a paso del cómo se debe realizar una prueba.

Roles

Los roles son la parte humana del proceso de desarrollo de software. Un rol no es la persona quien ejecuta una tarea, en lugar de eso, un rol describe la colaboración entre los miembros del equipo para llevar a cabo una tarea.

Figura 8 Principales roles en OpenUP y su interacción



Fuente: (Eclipse Process Framework et al., 2004-2007)

Los principales roles en OPENUP son:

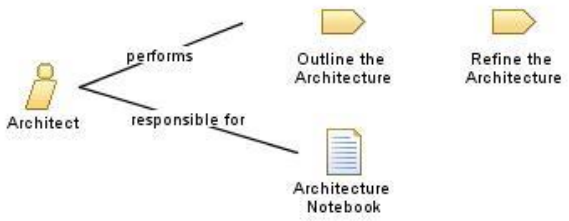
Arquitecto

El arquitecto es quien define la arquitectura de software, lo cual incluye la toma de decisiones claves que delimitan el diseño y la implementación del proyecto.

La persona en este rol es quien:

- Define la visión
- Crea los requisitos de búsqueda y esquema
- Define el detalle de los requisitos.
- Diseña la solución
- Participa en la creación del plan de proyecto
- Evalúa los resultados
- Maneja las iteraciones
- Participa en la creación del plan de proyecto

Figura 9: Relaciones Rol Arquitecto



Fuente: (Eclipse Process Framework et al., 2004-2007)

Gerente de Proyecto

El Gerente de Proyecto es quien dirige la planificación del proyecto, coordina las iteraciones y encamina al equipo del proyecto al cumplimiento de los objetivos del proyecto, finalmente, es el responsable de la gestión de los riesgos del proyecto y de la aceptación del producto por parte del cliente.

La persona en este rol es responsable de:

- Crear el plan de proyecto
- Crear la lista de elementos de trabajo
- Crear la lista de riesgos
- Crear el plan de iteración

Figura 10: Relaciones Rol Gerente de Proyecto

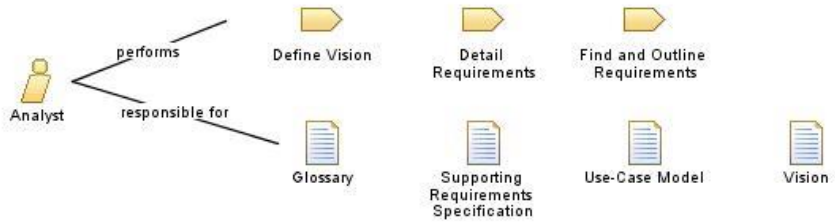
Fuente: (Eclipse Process Framework et al., 2004-2007)

Analista

El analista es quien tiene relación directa con el cliente y el usuario final y por ende es quien representa sus preocupaciones. Recopila la información de las partes interesadas, captura y establece prioridades para los requisitos y comprende el problema a resolver. La persona en este rol es responsable de:

- Crear casos de prueba
- Delinear la arquitectura
- Diseñar la solución
- Implementar scripts de prueba
- Crear el plan de proyecto.
- Evaluar los resultados
- Manejar las iteraciones
- Participar en la creación del plan de proyecto

Figura 11: Relaciones Rol Analista



Fuente: (Eclipse Process Framework et al., 2004-2007)

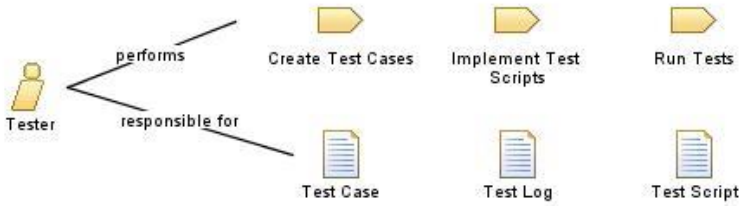
Tester

El tester es quien se encarga de las actividades relacionadas con las pruebas de software, tales como identificación, definición, implementación, realización y registro de las pruebas, así como el análisis de los resultados.

La persona en este rol es responsable de:

- Caso de prueba
- Script de prueba
- Registro de prueba

Figura 12: Relaciones Rol Tester



Fuente: (Eclipse Process Framework et al., 2004-2007)

Desarrollador

El desarrollador es quien se encarga de desarrollar una parte del sistema, incluido ajustes a la arquitectura, pruebas de unidad y la integración de componentes.

La persona en este rol es responsable de:

- Implementación
- Especificación de requisitos de apoyo
- Caso de uso
- Pruebas de desarrollador
- Registro de prueba
- Lista de elementos de trabajo
- Diseño

Figura 13: Relaciones Rol Desarrollador

Fuente: (Eclipse Process Framework et al., 2004-2007)

Stakeholder

Este rol representa a un grupo de interés cuyas necesidades deben ser satisfechas por el proyecto. Este papel puede ser desempeñado por cualquier persona que se vea afectada de forma positiva o negativa por el proyecto.

Ciclo de Vida

OpenUP es un proceso iterativo con iteraciones distribuidas a través de sus fases: iniciación, elaboración, construcción y transición. Cada fase puede tener tantas iteraciones requiera y su longitud varía dependiendo del proyecto. Generalmente las iteraciones duran alrededor de un mes.

Iteración de la fase de inicio

La iteración de la fase de inicio es la primera iteración importante del proyecto. En esa iteración se tiene que cumplir los siguientes criterios de evaluación:

- Concordancia con los interesados
- Definición del alcance
- Estimaciones iniciales del cronograma.
- Definiciones y prioridades para un conjunto inicial de requisitos.
- Riesgos identificados y estrategias de mitigación propuestas.

Iteración de la fase de elaboración

La mayoría de las actividades durante la iteración de diseño ocurren en paralelo. Los principales objetivos de esta iteración es tener una mejor comprensión de los requisitos,

creando y estableciendo una línea de base de la arquitectura para el sistema y mitigando los riesgos de máxima prioridad.

Iteración de la fase de construcción

El objetivo principal de la fase de construcción es entregar versiones preliminares del sistema hasta llegar una final, esto conlleva a realizar construcciones cada vez más completas y estables.

Iteración de la fase de transición

El objetivo principal de la fase de transición es ajustar la funcionalidad, el rendimiento y la calidad general del producto beta desde el final de la fase de construcción.

6.3 MODELOS DE EVALUACIÓN DE PROCESOS

Un modelo de evaluación de procesos es un modelo que permite evaluar los procesos usados en una organización con el fin de determinar la capacidad de dichos procesos. Su propósito es identificar debilidades, fortalezas y habilidades del proceso para controlar o evitar las causas de baja calidad, desviaciones en cronograma o en el presupuesto.

6.3.1 ISO/IEC 15504

El Estándar internacional ISO/IEC 15504, también conocido como Software Process Improvement Capability Determination (SPICE), es un modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento de los productos de software. Su objetivo principal es evaluar de forma sistemática la calidad de los procesos de desarrollo software en una empresa.

ISO/IEC 15504 proporciona un modelo para la evaluación y mejora de los procesos de una organización y a su vez ofrece un enfoque estructurado para la evaluación de los procesos, con el ánimo de conocer el estado de los mimos y así proponer una mejora.

El estándar ISO/IEC 15504 consta de 3 elementos importantes:

- 6 niveles de capacidad de proceso definido.
- Un conjunto de requisitos para definir procesos.
- Un conjunto de requisitos sobre cómo realizar evaluaciones consistentes.

Y comprende**:**

- Evaluación de procesos
- Mejora de procesos
- Evaluación de la capacidad y/o madurez de los procesos

Para determinar la madurez de los procesos ISO/IEC 15504 compara los procesos de la organización vs los objetivos organizacionales, esto por medio de la implementación de una serie de evaluaciones que determinan dicha madurez. Los resultados de estas evaluaciones son compartidos con la organización, con el fin de utilizar estos resultados como un programa de mejora continua.

Niveles de Madurez

El modelo en niveles de ISO 15504 permite evaluar la capacidad del proceso de desarrollo, para ello cuenta con 6 niveles de capacidad posibles para un proceso determinado. Para que un proceso alcance un nivel de capacidad deseado, debe cumplir con los requisitos determinados y las acciones concretas que requiere cada nivel hasta llegar al nivel máximo le cual indica que el proceso está integrado completamente con la organización.

- Nivel 0: Incompleto
- Nivel 1: Realizado
- Nivel 2: Gestionado
- Nivel 3: Establecido
- Nivel 4: Predecible
- Nivel 5: En optimización

Nivel 0

Este nivel se caracteriza porque no hay procesos en la organización, no se controlan los costos de producción ni los tiempos de respuesta, finalmente no existe planificación de actividades y tiempos.

Nivel 1

En este nivel existe evidencia de la realización de un proceso. La organización simplemente implementa y alcanza de manera básica los resultados del proceso.

Este nivel es capaz de:

- Entregar productos de acuerdo a los requerimientos del cliente y los de la organización.
- Definir los requisitos y restricciones del sistema que satisfaga las necesidades del cliente.
- Transformar los requisitos de los stakeholders en requisitos técnicos del sistema.

Nivel 2

En este nivel se evidencia la existencia y control no solo del proceso si no de sus productos de trabajo asociados, su implementación está planificada, monitoreada y ajustada.

Este nivel es capaz de:

- Establecer políticas y procedimientos para la gestión
- Elaborar y comunicar de forma efectiva y fiable los planes de proyecto
- Establecer el estado del proyecto, lo cual incluye cronograma, presupuesto y objetivos técnicos.
- Mantener la integridad de todos aquellos elementos que forman parte del producto software.
- Identificar necesidades y diseñar medidas a partir de estas necesidades.

Nivel 3

En este nivel el proceso está establecido y documentado con el fin de garantizar la capacidad para cumplir los objetivos. Para alcanzar este nivel es necesario implementar los procesos y requisitos de los dos niveles anteriores.

Nivel 4:

En este nivel el proceso es predecible. La organización mide y analiza la realización de los procesos y por ende se realizan dentro de unos tiempos establecidos. Para alcanzar este nivel es necesario implementar los procesos y requisitos de los anteriores niveles.

Nivel 5:

En este nivel el proceso está optimizado, es la adaptación de todos los procesos anteriores con el fin de alcanzar los objetivos de negocio de la empresa. Se monitorean los procesos de la empresa y se analizan los datos obtenidos. Los procesos de la organización pueden cambiar dinámicamente para adaptarse a los objetivos presentes y futuros de la empresa.

Atributos del proceso

Los atributos del proceso permiten evaluar el alcance de un nivel de capacidad determinado para un proceso. Los atributos del proceso están ligados a cada nivel de madurez.

Nivel 1:

- PA 1.1 Realización del proceso

Nivel 2:

- PA 2.1 Gestión de la realización
- PA 2.2 Gestión del producto de trabajo.

Nivel 3:

- PA 3.1 Definición del proceso
- PA 3.2 Despliegue del proceso.

Nivel 4:

- PA 4.1 Medición del proceso
- PA 2.2 Control del proceso

Nivel 5:

- PA 5.1 Innovación del proceso
- PA 5.2 Optimización continua.

El cumplimiento de los atributos del proceso determinará el nivel de capacidad del proceso. Cada atributo del proceso tiene unas prácticas de cumplimiento asociadas a dicho atributo.

Aunque los procesos tienen partes comunes como son los atributos y las prácticas, también cuentan con partes específicas como son los outcomes y las actividades. Un outcome describe las características únicas que debe implementarse para alcanzar cada proceso y los cuales son requeridos. Las actividades son únicas para el proceso.

Atributos de calificación

Finalmente, se define una escala de calificación cuyo valor se basa en el porcentaje de cumplimento de los atributos:

- N, no implementado (0-15%)
- P, Parcialmente implementado (> 15-50%)
- L, Ampliamente implementado (> 50-85%)
- F, completamente implementado (> 85%)

6.3.2 ISO/IEC 29110

El Estándar ISO/IEC 29110, "ha sido desarrollada para mejorar la calidad de los productos y servicios, y la ejecución y funcionamiento de los procesos" (ISO 29110-5-1, 2014).
El estándar proporciona una serie de guías para mejorar el proceso de desarrollo de software en las entidades pequeñas, las cuales pueden ser empresas, organizaciones, departamentos o proyectos de hasta 25 personas, ayudándolas en la implementación de buenas prácticas para incrementar la calidad de los productos y/ servicios, reducción de tiempos y costos de producción. Está compuesto por:

Perfiles:

- Perfil de entrada
- Perfil básico
- Perfil intermedio

- Perfil avanzado

Categorías de procesos

- Proceso de Gestión de Proyectos
- Procesos de Implementación de Software

Elementos del proceso

- Objetivos
- Tareas
- Roles
- Productos de trabajo

Perfiles del Ciclo de Vida

Los perfiles genéricos se aplican a una gran mayoría de microempresas que no desarrollan un software crítico. Existen cuatro perfiles (entrada, básico, intermedio y avanzado) los cuales ofrece un enfoque progresivo para servir a la mayoría de las microempresas.

Perfil de Entrada

El perfil de entrada es aplicable a empresas o microempresas que trabajan en proyectos pequeños, con alrededor de 6 personas. "Describe las prácticas de desarrollo de software de una sola aplicación por un solo equipo de trabajo, y sin riesgo especial o factores situacionales" (Laporte, 2013).

Perfil de Básico

El perfil de básico está dirigido a pequeñas empresas que desarrollan un proyecto a la vez.

Perfil Intermedio

El perfil intermedio está dirigido a pequeñas empresas que desarrollan más de un proyecto a la vez dentro del contexto de la organización.

Perfil Avanzado

El perfil avanzado está dirigido a pequeñas empresas que desean mantenerse y crecer como empresas de desarrollo de software independientes y competitivas.

Categorías de Procesos

Proceso de Gestión de Proyectos (GP)

El objetivo del proceso de Gestión de Proyectos es establecer y llevar a cabo las tareas del proyecto que permitan cumplir con los objetivos del proyecto en los tiempos, costos y con la calidad esperada.

El Proceso de GP utiliza la declaración del trabajo del cliente para elaborar el plan de proyecto. Las tareas de evaluación de proyecto comparan los progresos del proyecto con el plan de proyecto, esto con el fin de tomar decisiones respecto a la eliminación de desviaciones o incorporar cambios. La actividad de cierre recibe la aceptación del cliente para formalizar la finalización del proyecto.

Actividades de la Gestión de Proyectos

Las actividades de la gestión de proyectos incluyen:

- GP1 Planificación del Proyecto
- GP2 Ejecución del Plan de Proyecto
- GP3 Evaluación y Control del Proyecto
- GP4 Cierre del Proyecto

GP1 Planificación del Proyecto

La actividad de Planificación del Proyecto documenta los detalles de planificación para la gestión del proyecto.

Tareas de la Planificación del Proyecto

- GP1.1 Revisión de la declaración del trabajo

- GP1.2 Definir con el cliente las instrucciones de entrega de cada uno de los entregables especificados en la declaración del trabajo.
- GP1.3 Identificar las tareas específicas para producir los entregables y sus componentes de software
- GP1.4 Establecer la duración estimada para llevar a cabo cada tarea.
- GP1.5 Identificar y documentar los recursos necesarios para llevar a cabo cada tarea.
- GP1.6 Establecer la composición del equipo de trabajo con asignación de roles y recursos.
- GP1.7 Establecer fechas de inicio y finalización a cada una de las tareas del proyecto.
- GP1.8 Calcular y documentar los costos estimados del proyecto.
- GP1.9 Identificar los riesgos del proyecto
- GP1.10 Documentar la estrategia de control de versiones
- GP1.11 Generar el plan de proyecto integrado.
- GP1.12 Incluir descripción del producto, alcance, objetivos y entregables del proyecto.
- GP1.13 Obtener la aprobación del plan de proyecto.
- GP1.14 Revisar y aceptar el plan de proyectos.
- GP1.15 Establecer el repositorio del proyecto.

GP2 Ejecución del Plan de Proyecto

La actividad de Ejecución del Plan de Proyecto implementa el plan documentado del proyecto.

Tareas de la Ejecución del Plan de Proyecto

- GP2.2 Analizar y evaluar las solicitudes de cambio en cuanto a tiempo, costos e impacto.
- GP2.3 Llevar a cabo las reuniones de revisión con el equipo de trabajo.

- GP2.4 Llevar a cabo reuniones de revisión con el cliente, registrar los acuerdos y realizarles seguimiento hasta el cierre.
- GP2.5 Realizar respaldos
- GP2.6 Realizar la recuperación del repositorio del proyecto.

GP3 Evaluación y Control del Proyecto

La actividad de Evaluación y Control del Proyecto evalúa el desempeño del plan vs los compromisos documentados.

Tarea de Evaluación y Control del Proyecto

- GP2.1 Hacer seguimiento de la ejecución del Plan de Proyecto.
- GP3.1 Evaluar el progreso del proyecto contra el plan de proyecto.
- GP3.2 Establecer acciones para corregir desviaciones, problemas o riesgos identificados.
- GP3.3 Identificar cambios a los requisitos y/o al plan de proyecto.

GP3 Cierre del Proyecto

La actividad de Cierre del Proyecto proporciona la documentación y productos del proyecto según los requisitos del contrato.

Tareas de Cierre del Proyecto

- GP4.1 Formalizar la finalización del proyecto.
- GP4.2 Actualizar el repositorio del proyecto.

Proceso de Implementación de Software (IS)

El objetivo del proceso de Implementación de software es la realización de las actividades de análisis, diseño, construcción, integración y pruebas para productos de software nuevos o modificados, según los requisitos del cliente.

El proceso de IS da inicio al comenzar con las actividades de revisión del plan de proyecto, el cual guía la ejecución del análisis de requerimientos de software, el diseño arquitectónico, la construcción, integración y las pruebas de software.

Actividades de la Implementación de Software

Las actividades de la implementación de software incluyen:

- IS1 Iniciación de la Implementación de Software
- IS2 Análisis de Requisitos de Software
- IS3 Diseño Arquitectónico y Detallado de Software
- IS4 Construcción del Software
- IS5 Integración y Pruebas de Software
- IS6 Entrega del Producto

IS1 Iniciación de la Implementación de Software

La actividad de Iniciación de la Implementación de Software asegura que sea cumplido el plan de proyecto por el equipo de trabajo.

Tareas de Iniciación de la Implementación de Software

IS1.1 Revisión del plan de proyecto con todos los miembros del equipo.

IS1.2 Configurar o actualizar el ambiente de implementación.

IS2 Análisis de Requisitos de Software

La actividad de Análisis de Requisitos de Software analiza los requisitos que se acordaron con el cliente y establece los requisitos de validación del proyecto.

Tareas de Iniciación de la Implementación de Software

- IS2.1 Asignar tareas a los miembros del equipo basados en el plan de proyecto.
- IS2.2 Documentar o actualizar la especificación de requisitos.
- IS2.3 Verificar la exactitud de la especificación de requisitos.

- IS2.4 Obtener la aprobación de la especificación de requisitos.
- IS2.5 Documentar la versión preliminar de la documentación del software de usuario.
- IS2.6 Obtener la aprobación de la documentación del software de usuario.
- IS2.7 Incorporar a la configuración del software en las líneas base la especificación de requisitos y la documentación de software del usuario.

IS3 Diseño Arquitectónico y Detallado del Software

La actividad de Diseño Arquitectónico y Detallado del Software transforma los requisitos de software en la arquitectura y diseño detallado de software.

Tareas de Diseño Arquitectónico y Detallado del Software

- IS3.1 Asignar tareas a los miembros del equipo basados en el plan de proyecto.
- IS3.2 entender la especificación de requisitos
- IS3.3 Documentar o actualizar el diseño de software
- IS3.4 Obtener la aprobación del diseño de software
- IS3.5 Establecer o actualizar los casos de prueba y procedimientos de prueba para las pruebas de integración,
- IS3.6 Obtener la aprobación de los casos de prueba y procedimientos de prueba.
- IS3.7 Actualizar el registro de trazabilidad incorporando los casos de prueba y procedimientos de prueba.
- IS3.8 Incorporar como parte de las líneas base el diseño de software y el registro de trazabilidad.

IS4 Construcción del Software

La actividad de Construcción del Software consiste en desarrollar el código y los datos de software a partir del diseño del software.

Tareas de Construcción del Software

- IS4.1 Asignar tareas a los miembros del equipo basados en el plan de proyecto.

- IS4.2 Entender el diseño del software
- IS4.3 Construir o actualizar los componentes del software.
- IS4.4 Diseño y aplicación de pruebas unitarias.
- IS4.5 Corregir defectos.
- IS4.6 Actualizar el registro de trazabilidad.
- IS4.7 Incorporar los componentes de software y el registro de trazabilidad.

IS5 Integración y Pruebas de Software

La actividad de Integración y Pruebas del Software consiste asegurar que los componentes de software satisfagan lo requerimientos de software.

Tareas de Construcción del Software

- IS5.1 Asignar tareas a los miembros del equipo basados en el plan de proyecto.
- IS5.2 Entender los casos y procedimientos de pruebas.
- IS5.3 Integrar el software
- IS5.4 Realizar las pruebas de software
- IS5.5 Corregir los defectos encontrados y hacer pruebas de regresión.
- IS5.6 Actualizar el registro de trazabilidad.
- IS5.7 Documentar la guía de operación del producto
- IS5.8 Obtener aprobación para la guía de operación del producto.
- IS5.9 Realizar la documentación de usuario
- IS5.10 Obtener la aprobación de la documentación de software del usuario
- IS5.11 Incorporar a la configuración del software los casos de pruebas, procedimiento de pruebas, software, registro de trazabilidad, informe de pruebas, guía de operación del producto y documentación de software del usuario.

IS6 Entrega del Producto

La actividad de Entrega del Producto proporciona al cliente un producto de software integrado.

Tareas de Entregas del Producto

- IS6.1 Asignar tareas a los miembros del equipo basados en el plan de proyecto.
- IS6.2 Entender la configuración del software
- IS6.3 Realizar la documentación de mantenimiento.
- IS6.4 Obtener aprobación de la documentación de mantenimiento.
- IS6.5 Incorporar a la configuración del software la documentación de mantenimiento
- IS6.7 Llevar a cabo la entrega del producto.

Elementos del Proceso

Roles

Los roles descritos en el siguiente apartado corresponden al Grupo de perfil genérico: perfil básico

Analista

Es la persona que tiene conocimiento y experiencia en análisis de requisitos, en diseño de interfaces de usuario, en técnicas de revisión y edición y finalmente conocimientos en desarrollo y mantenimiento de software.

Cliente

Es la o las persona que tiene conocimiento y experiencia en los procesos del cliente, habilidad para explicar los requisitos del cliente y experiencia en dominio de la aplicación.

Diseñador

Es la persona que tiene conocimiento y experiencia en diseño de componentes de software. Conocimiento en técnicas de revisión, planificación y desempeño de las pruebas de integración, técnicas de edición y experiencia en mantenimiento de software.

Programador

Es la persona que tiene conocimiento y experiencia en programación, integración y pruebas unitarias. Conocimiento en técnicas de revisión, edición y experiencia en mantenimiento de software.

Gerente del Proyecto

Es la persona que tiene capacidad de liderazgo, con experiencia en toma de decisiones, planificación, gestión del personal, delegación, supervisión, finanzas y desarrollo de software.

Líder Técnico

Es la persona que tiene conocimiento y experiencia en dominio del proceso del software.

Equipo de Trabajo

Son un grupo de personas que tienen conocimiento y experiencia según sus roles en el proyecto.

7 DISEÑO METODOLÓGICO

La metodología propuesta para llevar a cabo el desarrollo del proyecto está basada en el modelo IDEAL descrito anteriormente.

A continuación, se describe las actividades que se llevaron a cabo en las fases propuestas llevando un orden cronológico, y mostrando además los recursos que se necesitaron para ejecutar dichas actividades.

Fase 1: Inicial

En la fase inicial es donde la dirección entiende la necesidad de mejora, se establece la infraestructura inicial de mejora, se definen las funciones y responsabilidades y se asignan los recursos iniciales

Para llevar a cabo la fase inicial, se deben llevar a cabo las siguientes actividades:

1. **Fijar Contexto**

   En esta actividad se realizó una descripción corta de la situación actual del área de TI, se mencionaron las razones para iniciar el cambio y finalmente se definió y especificó en donde se enfocaron los esfuerzos.

2. **Asegurar el patrocinio**

   En esta actividad se estableció el apoyo de las directivas de la institución para la consecución del proceso de desarrollo.

3. **Establecer Infraestructura**

   En esta actividad se estableció el mecanismo para la gestión de todos los detalles de la implementación. Se definió el recurso humano designado para construcción del proceso de desarrollo, la cantidad de horas designadas para dicha labor y se estableció el lugar donde se llevaron a cabo las actividades.

Fase 2: Diagnostico

En la fase de diagnóstico se buscó obtener el entendimiento completo del trabajo que se debía realizar, para esto fue necesario conocer y caracterizar el estado actual y futuro del área de T.I con relación a los procesos de desarrollo que se utilizan en esta área.

Para llevar a cabo la fase de diagnóstico, se realizaron las siguientes actividades:

1. **Identificar problemas y causales**

   En esta actividad se identificaron y definieron los problemas más relevantes del área de T.I relacionados con el desarrollo de software y se identificaron las causas asociadas a los problemas identificados.

2. **Fijar objetivos de mejora**

   Se llevaron a cabo una identificación de objetivos de mejora medibles, alcanzables y concretos

3. **Caracterizar el estado actual del área de T.I**

   Para la caracterización del estado actual del área de T.I se identificaron y documentaron los procesos que se llevan en el área de T.I, las personas que hacen parte de cada uno de los procesos identificados y como hacen cada uno de los procesos y que se requiere para hacerlo.

Fase 3: Análisis

En la fase de análisis se buscó fijar criterios de selección con base en las necesidades del área, analizar diferentes procesos de desarrollo, confrontarlos y seleccionar uno o diseñar uno basado en los criterios fijados.

Para llevar a cabo la fase de análisis, se realizaron las siguientes actividades:

Fijar Criterios de Selección

En esta actividad se identificaron y documentaron los problemas más relevantes y que requieren mayor atención y por ende solución con el proceso de desarrollo a implementar y con base en dicha identificación, se fijaron los criterios para seleccionar el proceso de desarrollo.

1. **Análisis de diferentes procesos de desarrollo**

   Para esta actividad se realizó una revisión bibliográfica de procesos de desarrollo como SCRUM, OPEN UP, RUP y XP. Posteriormente, se diseñó un cuadro comparativo entre los procesos de desarrollo mencionados basado en los criterios de selección fijados en el punto anterior.

2. **Selección de procesos y/o prácticas**

   En esta actividad se confrontaron los procesos de desarrollo analizados vs los criterios de selección y se seleccionaron los componentes que conformaron el proceso de desarrollo diseñado.

Fase 4: Diseño

En la fase de diseño se seleccionaron las mejores prácticas de las metodologías de desarrollo seleccionadas y de acuerdo con las necesidades de la organización se diseñó un proceso propio para el área de T.I de la Universidad de Manizales.

Para llevar a cabo la fase de diseño, se realizaron las siguientes actividades:

1. **Diseño del Proceso**

   En esta actividad se reunieron todos los elementos que permitieron encontrar la solución óptima, siempre apuntando a suplir las necesidades que se identificaron en el área de T.I de la Universidad de Manizales. Para esto se seleccionaron y documentaron elementos tales como fases con sus respectivas actividades, resultados, eventos, roles, disciplinas, artefactos y herramientas tecnológicas que hacen parte del proceso de desarrollo. La documentación del proceso se llevó a cabo tanto en un documento físico como en una herramienta navegable que es accesible para los miembros del equipo de desarrollo y la dirección del área.

Fase 5: Implementación

En la fase de implementación se buscó llevar a la práctica el proceso diseñado, la idea principal fue seleccionar un proyecto que cumpla con los criterios definidos e implementar el proceso de desarrollo previamente diseñado.

Para llevar a cabo la fase de implementación, se realizaron las siguientes actividades:

1. **Seleccionar un Proyecto Piloto**

   Una vez se diseñó la solución, se seleccionó un proyecto en el cual se logró probar el proceso de desarrollo, para dicha selección se tuvo en cuenta criterios de selección tales como tamaño del proyecto, tiempo de desarrollo, equipo desarrollador, funcionalidades, etc.

Adicionalmente, se definió y asignó el recurso humano que participó en todo el ciclo de vida de desarrollo del software y los cuales implementaron dicho proceso.

2. **Implementar el proceso de desarrollo en el proyecto seleccionado**

   En esta actividad se documentaron los resultados obtenidos al usar el proceso de desarrollo en el proyecto seleccionado en el punto anterior. Se describieron las actividades realizadas, datos del proyecto, artefactos, roles y finalmente se mostraron los resultados del desarrollo del proyecto piloto.

Fase 6: Validación

En la fase de validación se analizaron los resultados obtenidos en la implementación, como dificultados encontradas en el proceso, experiencias, cumplimiento de expectativas, etc.

Para llevar a cabo la fase de validación, se realizaron las siguientes actividades:

1. **Analizar resultados**

   La validación del proceso de desarrollo se llevó a cabo con el apoyo de la norma ISO/IEC 29110, lo que se buscó fue evaluar el cumplimiento de la norma específicamente el Proceso de Implementación de Software I.S, teniendo en cuenta actividades, tareas y cumplimiento de cada tarea en las diferentes fases del proceso de la norma.

2. **Comparar resultados**

   El comparativo de resultados se llevó a cabo entre dos aplicativos con naturaleza y funcionalidades similares, los cuales fueron desarrollados en épocas diferentes. Para esta comparación se identificaron y seleccionaron criterios que permitieron validar los proyectos desarrollados, tales como criterios de calidad, criterios propios del desarrollo del software y aspectos generales. A cada uno de los aspectos se les dío una calificación y una observación.

3. **Futuras mejoras y lecciones aprendidas**

   Esta fase buscó aprovechar la experiencia adquirida en la implementación del nuevo proceso de desarrollo, para validar que se hubiera cumplido los objetivos, y proponer estrategias que permitan realizar la implementación de nuevos cambios.

En las lecciones aprendidas como primera medida se realizó un análisis del diseño, implementación y puesta en marcha del proceso de desarrollo con el fin de listar aspectos aprendidos durante todo el proceso y a los cuales se les debía hacer un mayor énfasis. Con las futuras mejoras se buscó proponer acciones de mejoras relacionadas a las lecciones aprendidas.

8 RESULTADOS ESPERADOS

A continuación, se relaciona una tabla con los resultados esperados en el desarrollo de este proyecto Tabla 2.

Tabla 1: Resultados y productos esperados de la investigación

Objetivos	Resultados / productos esperados	Indicador	Descripción
Diagnosticar el estado actual del proceso de desarrollo de software del área de T.I siguiendo los lineamientos de modelo IDEAL	Diagnostico estado actual del proceso de desarrollo de software del área de T.I	Documento	Análisis y diagnóstico inicial del estado actual del proceso de desarrollo de software del área de T.I siguiendo los lineamientos de modelo IDEAL.
Diseñar un proceso de desarrollo basado en una integración/adaptación metodológica de (de los) proceso (s) más idóneos según las necesidades de la organización.	Selección y combinación del proceso de desarrollo	Documento	Documento donde se define y especifica el proceso de desarrollo para el área de T.I de la Universidad de Manizales
Diseñar un proceso de desarrollo basado en una integración/adaptación metodológica de (de los) proceso (s) más idóneos según las necesidades de la organización.	Definición de roles y artefactos	Documento	Documento donde se define los roles, las responsabilidades y los artefactos a realizar por el equipo de desarrollo del área de T.I
Diseñar un proceso de desarrollo basado en una integración/adaptación metodológica de (de los) proceso (s) más idóneos según las necesidades de la organización.	Proceso de Desarrollo de Software	Proceso en EPF Composer	Proceso de desarrollo para el área de T.I en la herramienta EPF Composer

Implementar el proceso de desarrollo diseñado en un proyecto piloto.	Implementación proceso de desarrollo en un proyecto piloto	Artefactos	Artefactos definidos en el proceso de desarrollo para el proyecto piloto
Identificar lecciones aprendidas y posibles mejoras enmarcadas con el estándar ISO / IEC 29110.	Comparativo proceso de desarrollo	Tabla comparativa	Tabla comparativa entre proyecto sin proceso de desarrollo implementado vs proyecto piloto
Identificar lecciones aprendidas y posibles mejoras enmarcadas con el estándar ISO / IEC 29110.	Lecciones Aprendidas	Documento	Documento donde identifican las lecciones aprendidas posibles mejoras enmarcadas con el estándar ISO / IEC 29110
Enviar artículo científico a revista indexada producto de trabajo final	Artículo Científico producto de trabajo de tesis	Documento	Envío de artículo científico a revista indexada

Fuente: Elaboración propia

9 DESARROLLO

En el área de tecnologías de Información (T.I) de la Universidad de Manizales se viene adelantando estudios e investigaciones para mejorar su proceso de desarrollo.

La mejora del proceso de software tiene como objetivo analizar y definir cómo mejorar las prácticas de desarrollo software de una organización, en este caso, dentro del área, partiendo de una evaluación del proceso actual y centrándose en mejorar el rendimiento, la utilidad y la efectividad de los procesos de una manera disciplinada.

Para llevar a cabo dicha mejora, se debe empezar por un diagnóstico del proceso de desarrollo que se lleva en el área. Dicho diagnóstico se realizó basado en el modelo IDEAL, el cual sirvió de guía para la iniciación, planificación e implementación de acciones de mejora para el proceso de software en las organizaciones.

En este documento se desarrolló un diagnóstico del proceso de desarrollo de software que se llevó a cabo en el área de T.I, con el cual se buscó conocer el estado actual del proceso de desarrollo y valorar como alinear la empresa con el modelo de calidad deseado, empezando por una etapa de inicio, diagnóstico y terminado con la etapa de establecer.

9.1 FASE 1: INICIAL

9.1.1 Fijar Contexto

El área de T.I de la Universidad de Manizales cuenta con un proceso de desarrollo muy limitado, que no se encuentra documentado y que no permite establecer tiempos o recursos para los proyectos de software que se llevan a cabo dentro del área. Al contar con un proceso de desarrollo tan limitado, se dificulta llevar un control sobre las actividades de desarrollo, las personas que intervienen en ellas, los tiempos de ejecución de dichas actividades, la adaptación al cambio, la integración de componentes y el control de versiones. Todo esto hace necesario mejorar dicho proceso de desarrollo, el cual se espera mejore los aspectos anteriormente mencionados, que permitan tener un mayor control de

los proyectos que pasan por el área, que sirva de guía a las personas involucradas en la ejecución de los proyectos en cuanto a actividades, responsabilidades y entregables en cada etapa del proyecto y finalmente que regrese así la credibilidad por partes de las directivas y de los demás miembros de la Universidad hacía el área de T.I.

9.1.2 Asegurar el Patrocinio o Apoyo

Al tratarse T.I de un área estratégica y de apoyo para los procesos institucionales de la Universidad de Manizales, el apoyo de las directivas en el mejoramiento del proceso de desarrollo es importante, ya que esto permitirá la ejecución de proyectos de una forma más eficiente y con mayor calidad, obteniendo así un ahorro en tiempo y finalmente en gastos. Para la construcción del proceso de desarrollo se cuenta con el apoyo tanto de la dirección del área de T.I como de las directivas de la institución, pues son conocedoras de las falencias presentadas en los productos de software, los retrasos y la baja calidad que ha llevado a un descontento general, pero a su vez, reconocen el esfuerzo del área para solucionar dichos inconvenientes.

9.1.3 Establecer Infraestructura

Para el desarrollo de las actividades del proceso de desarrollo, se destinó a la Líder de Desarrollo del área de T.I y la analista – desarrolladora del área. El tiempo asignado en cuanto a dedicación fueron 3 horas diarias por parte de la líder de desarrollo y 1 hora diaria por parte de la analista – desarrollador. Las actividades se llevaron a cabo en las instalaciones del área de T.I, cada una de ellas fueron acompañadas de evidencias, las cuales están definidas previamente en la metodología y cronograma del proyecto. Todas las actividades y las evidencias tuvieron aval de la dirección del área.

9.2 FASE 1: DIAGNÓSTICO

9.2.1 Identificar Problemas

Identificar y definir problemas

Entre los problemas que se identifican en el área de T.I están:

- Proyectos sin documentación
- Equipo de desarrollo sin una guía para entregables o paso a seguir en el desarrollo del proyecto, haciendo que todo quedé en cabeza del líder de desarrollo y generando pérdida de tiempo.
- Dificultades a la hora de estimar tiempos y recursos de las actividades de un proyecto.
- Equipo de desarrollo sin roles y responsabilidades definidas.
- Proyectos que salen a producción con muchos errores o bugs.
- Problemas a la hora de realizar integraciones.
- Dificultades a la hora de gestionar cambios.
- Cambios en las especificaciones de los proyectos
- El software no puede ser probado correctamente.
- Baja calidad en el desarrollo de software.

Identificar las causas asociadas a los problemas identificados

- Falta o mala elección y uso de una metodología de desarrollo
- Gestión deficiente de los proyectos de software
- Falta de comunicación entre los miembros del equipo.
- Falta de implicación por parte de las directivas.
- Mala elecciones o poco uso de herramientas de gestión y automatización de actividades.
- Presión en el desarrollo de software
- Cambios a último momento de actividades o prioridades a desarrollar.
- Documentación escasa.
- Estimación de proyectos con tiempos demasiados ajustados.

9.2.2 Fijar Objetivos de Mejora

- Identificar y listar objetivos de mejora medibles, alcanzables y concretos:

1. Reducir como mínimo un 20% la cantidad de bugs reportados después de puesta en marcha de un proyecto de software.
2. Mejorar la estimación de recursos dentro en un proyecto.
3. Mejorar la estimación de tiempos dentro del proyecto.
4. Mejorar las integraciones de componentes de software con el uso de herramientas de integración continua.
5. Mejorar la credibilidad del área de T.I y de sus proyectos por parte de los directivos y en sí de la comunidad académica.

9.2.3 Caracterizar el Estado Actual del Área de T.I

Identificar los procesos que se llevan en el área de T.I

- Identificar soluciones
- Gestión de proyectos
- Desarrollo y mantenimiento de software
- Soporte a usuarios
- Gestión de incidencias
- Capacitación a usuarios
- Administración de Base de Datos

Identificar las personas que hacen parte de cada uno de los procesos identificados

Tabla 2: Procesos T.I con responsables

#	Proceso	Personas
1	Identificar soluciones	Jefe del área
		Líder de Desarrollo
2	Gestión de proyectos	Líder de Desarrollo
3	Desarrollo de software	Equipo de Desarrollo
		Analista
		Desarrolladores
		Tester de calidad
		D.B.A
4	Mantenimiento de software	Equipo de Desarrollo
		Analista
		Desarrolladores
		Tester de calidad
		D.B.A
5	Soporte a usuarios	Desarrolladores
		Analista
6	Capacitación a usuarios	Analista
7	Administración de Base de Datos	D.B.A

Fuente: Elaboración propia

Identificar cómo hacen cada uno de los procesos y que se requiere para hacerlo

Identificar soluciones

Lo primero que se hace es recopilar los requerimientos que llegan al área, se organizan, priorizan y analizan. La recopilación se hace de dos formas, la primera es por medio de reuniones con las personas interesadas en donde exponen sus necesidades, la segunda es por medio de una plantilla de requerimientos, en la cual los interesados plasman su necesidad ya de forma escrita o por medio de dibujos o prototipos. Seguido de esta recopilación, se realiza un análisis de los requerimientos y una priorización de los mismo,

con base en la estrategia de la alta dirección. Con base en este estudio se buscan posibles soluciones y finalmente se propone la mejor solución. En este proceso generalmente interviene el jefe del área y el líder de desarrollo.

Gestión de Proyectos

La gestión de proyectos se lleva a cabo inmediatamente después de que se ha seleccionado un proyecto a desarrollar y se ha planteado una solución. Dicha gestión abarca desde la planificación, gestión de recursos humanos, técnicos, asignación de tiempo. La gestión de proyectos está a cargo del líder de desarrollo con apoyo del analista, empieza con la definición de actividades y la asignación de dichas actividades a los miembros del equipo. En esta labor no se realizar una adecuada asignación de tiempos y las actividades muchas veces no se detallan bien, por lo que se generan muchas veces retrasos y una dificultad a la hora de llevar un control del proyecto.

Desarrollo de Software

Este proceso es llevado a cabo por el equipo de desarrollo, empieza desde el análisis de los requerimientos del cliente hasta la codificación de la solución. Tiene como objetivo desarrollar software de calidad y que satisfaga las necesidades y requerimientos que llegan al área de T.I.

Mantenimiento de Software

En este proceso se realizan modificaciones al producto de software después de su entrega ya sea para corregir errores, mejorar el rendimiento, realizar un ajuste o alguna funcionalidad. Es llevado a cabo por el equipo de desarrollo. Generalmente no se lleva un control sobre las modificaciones, correcciones o mejoras que se solicitan y se realizan lo cual dificulta la estimación de tiempos y recursos de dichas modificaciones.

Soporte a usuarios

El soporte a usuarios se hace a nivel de aplicativos y de datos. Este proceso generalmente se lleva a cabo por los desarrolladores o el D.B.A, en caso de que tenga que ver con los

datos almacenados en la base datos. Dicho soporte se hace vía telefónica, correo electrónico o a través de un aplicativo diseñado para dicho fin.

Capacitación a usuarios

El proceso de capacitación a usuarios es realizado generalmente por el analista de software y consiste en la transferencia de conocimientos realizada durante el proceso de implementación de un sistema o una nueva funcionalidad. Con la entrega de un producto de software se programa las capacitaciones a las personas directamente relacionadas con el producto entregado. En el caso de una nueva funcionalidad, se diseña un instructivo corto indicando los pasos de la funcionalidad y se capacita a las personas que así lo requieren.

Administración de Base de Datos

Consiste en la gestión de todos los datos que se encuentran almacenados en la base de datos institucional (inserción, actualización, borrado), así como la creación de consultas, procesos y procedimientos que permitan presentar o gestionar la información que en ella se encuentra almacenada. Es llevada cabo por el D.B.A quien vela porque dichos datos estén correctamente estructurados y almacenados.

9.3 FASE 3: ANÁLISIS

9.3.1 Fijar Criterios De Selección

Analizar cuáles son los problemas más relevantes que se requieren abarcar y solucionar con el proceso de desarrollo a utilizar.

Los problemas más relevantes y significativos en el área de T.I y que se quieren solucionar o mitigar con el proceso de desarrollo se describen a continuación:

- Baja calidad en el desarrollo de software.
- Proyectos sin documentación

- Equipo de desarrollo sin una guía para entregables o paso a seguir en el desarrollo del proyecto, haciendo que todo quedé en cabeza del líder de desarrollo y generando pérdida de tiempo.
- Tiempos y recursos estimados de los proyectos generalmente equivocados.
- Equipo de desarrollo sin roles y responsabilidades definidas.
- Proyectos que salen a producción con muchos errores o bugs.
- Problemas a la hora de realizar integraciones.

De acuerdo a dicha identificación, fijar los criterios para seleccionar el proceso de desarrollo.

Los criterios de selección más relevantes elegidos para el área de T.I se describen en la tabla 3.

Tabla 3: Criterios de selección proceso de desarrollo

Criterios de selección	
Fases o ciclo de vida	Conjunto o series de fases por la que pasa el desarrollo de software desde sus etapa inicial hasta la final
Roles	Un rol describe la colaboración entre los miembros del equipo para llevar a cabo una tarea
Artefactos	Los artefactos son elementos tangibles de un proyecto, elementos que el proyecto produce o usa mientras se trabaja en busca del producto final.
Integraciones Permanentes	Es la práctica de unir todas las piezas del trabajo de los desarrolladores en una sola forma eficiente.
Plan de Entregas	Detalle de cómo serán las entregas del proyecto, especificando las actividades e hitos que la componen
Plan de Iteraciones	Detalle de la planificación detallada de un periodo corto de tiempo dentro del proyecto

Documentación Adecuada	Detalle de pequeñas iteraciones que aporten cierto valor y evolucionemos las funcionalidades en los siguientes ciclos
Desarrollo software sobre cualquier tecnología	El proceso de desarrollo debe ser independiente del lenguaje de programación o la tecnología a usar en el proyecto
Desarrollo incremental	Desarrollo evolutivo e incremental para obtener retroalimentación y mejoramiento continuo
Adaptable a cambios	Facilidad para gestionar cambios.
Tamaño equipo de desarrollo	Cantidad de personas necesarias para el desarrollo de uno o varios proyectos.

Fuente: Elaboración propia

9.3.2 Análisis de Diferentes Procesos de Desarrollo

El análisis de los diferentes procesos de desarrollo consistió en una selección de algunas metodologías de desarrollo que más se adaptaban a las necesidades del área, los criterios de selección se obtuvieron como unidad de desarrollo en base a dichas necesidades y a los criterios de selección recopilados en el punto anterior en aras del diseño del nuevo proceso. Finalmente, se plasmó el conocimiento adquirido en el cuadro comparativo" **Comparativo Aplicativos P.A y G.A parte I y II"** – Tabla 18. La información resultante fue utilizada como punto de partida para la incorporación de mejores prácticas a proponer para obtener un proceso de desarrollo de software con mayor calidad, menores tiempos, lo que posteriormente se traduce en mejora continua del proceso y calidad del producto de desarrollo. Ver **ANEXO C** para visualizar el cuadro comparativo.

9.3.3 Selección de Procesos y/o Prácticas

Fases

Las fases son las etapas en las que se dividen un proyecto. Dichas fases van alineadas con el ciclo de vida del desarrollo del software y deben tener un enfoque ágil, iterativo e incremental. OPEN UP y RUP cuenta con una división de proyectos por fases o etapas que se adaptan a las necesidades del área.

Eventos

Un evento es la definición de una reunión programada para validar el alcance de actividades, la resolución de dudas, de problemas o para mostrar avances del producto. Scrum aporta buenas prácticas en la definición de eventos, los eventos como el Sprint, Sprint Planning, el Sprint Review y Sprint Retrospective son prácticas que se ajustan a las necesidades del área.

Roles

Un rol define el comportamiento y las responsabilidades de un individuo, o un grupo de individuos trabajando juntos como un equipo. OPEN UP y RUP proveen buenas prácticas y adecuadas definiciones para los roles dentro de un equipo de desarrollo.

Disciplinas

Es una colección de áreas de intereses las cuales a su vez se agrupan en tareas a realizar dentro del proyecto. OPEN UP cuenta con unas disciplinas de interés que se acoplan adecuadamente a los requerimientos del proceso de desarrollo.

Herramientas

Herramientas tecnológicas que ayudan en la gestión de un proyecto de software.

9.4 FASE 4: DISEÑO

9.4.1 Diseñar El Proceso

Definición de Proceso de Desarrollo: Proceso de Desarrollo Mixto, combinando Scrum y OPEN-UP

Los objetivos más importantes de un proceso de desarrollo es mejorar la calidad de los productos de software, proporcionar una guía de ejecución entre las diferentes etapas del desarrollo del producto de software y mejorar la predictibilidad de los proyectos.

Con el fin de satisfacer los objetivos anteriormente mencionados, se ha definido un proceso de desarrollo para el área de T.I de la Universidad de Manizales, el cual está basado en los modelos SCRUM y OPEN UP.

La fusión de SCRUM y OPEN UP sugiere diversas ventajas, entre ellas, la gestión regular de las expectativas del cliente, flexibilidad en las necesidades de los mismos, progreso visible en etapas tempranas, retroalimentación temprana y la implementación de las mejores prácticas de la ingeniería de software.
A continuación, se describe en forma detallada los procesos, actividades y tareas, roles y herramientas aplicables al nuevo proceso de desarrollo que faciliten y contribuyan a la mejora de los procesos del área de T.I.

Figura 14: Proceso de Desarrollo U. Manizales

Fuente: Imagen propia

Como se puede ver en la Figura 14, el modelo define un conjunto de seis elementos esenciales:

Fases: etapas del ciclo de vida del proyecto.

Tareas: actividad o unidad de trabajo que debe ser realizada por un rol.

Disciplinas: es una colección de áreas de intereses las cuales a su vez se agrupan en tareas a realizar dentro del proyecto.

Roles: determina el papel o la responsabilidad que tiene un individuo o grupo dentro del proyecto.

Artefactos: son considerados todo aquello que una tarea necesita para su realización, sin elementos físicos o digitales que apoyan al proyecto y al equipo de trabajo.
Eventos: reuniones presenciales que buscan aumentar la comunicación y entendimiento del proyecto entre las partes interesadas.
Herramientas: Herramientas tecnológicas que ayudan en la gestión de un proyecto de software.

Fases del Proceso

Las fases definidas para la metodología de desarrollo están basadas en el modelo OPEN-UP las cuales de describen a continuación:

Fase de Inicio:

Es la primera fase del ciclo de vida del proyecto, en esta fase los interesados del proyecto y los integrantes del equipo de desarrollo exponen sus necesidades las cuales son plasmadas para definir el objetivo del proyecto colaboran para determinar el ámbito del proyecto, sus objetivos y determinar si el proyecto es viable.

Esta fase se enfoca en las siguientes actividades con sus respectivos resultados:

Tabla 4: Actividades y Entregables Fase Inici

Actividad	Entregables o resultados
Iniciar el Proyecto	Documento Plan de Proyecto
	Cronograma
	Documento Análisis de Riesgos
Identificar y refinar los requerimientos y requisitos	Documento Especificación Casos de Uso
	Documento Requerimientos

Fuente: Elaboración propia

Fase de Elaboración:

En la segunda fase del ciclo de vida del proyecto, se realizan tareas de análisis y definición de arquitectura del sistema. Se obtiene un entendimiento más detallado de los requerimientos del sistema, se mitigan los riesgos esenciales y se produce un cronograma. Adicionalmente, se define las herramientas, infraestructura y el entorno de desarrollo a utilizar.

Esta fase se enfoca en las siguientes actividades con sus respectivos resultados:

Tabla 5: Actividades y Entregables Fase Elaboración

Actividad	Entregables o resultados
Identificar y refinar los requerimientos	Actualizar documento Especificación de casos de uso Actualizar documento requerimientos
Desarrollar la arquitectura	Documento Arquitectura del Sistema
Diseñar prototipos	Prototipos
Crear diagrama de actividades	Diagrama de actividades

Fuente: Elaboración propia

Fase de Construcción:

En la tercera fase del ciclo de vida del proyecto, se implementan y prueban componentes y funcionalidades del sistema basados en la arquitectura, se termina de refinar los requerimientos y se completan en detalles los diseños.

Esta fase se enfoca en las siguientes actividades con sus respectivos resultados:

Tabla 6: Actividades y Entregables Fase de Construcción

Actividad	Entregables o resultados
Implementar componentes y funcionalidades	Componentes de software
Pruebas	Documento casos de pruebas

Fuente: Elaboración propia

Fase de Transición:

La última fase del ciclo de vida del proyecto se enfoca en el paso del producto de software a la plataforma tecnológica del cliente cuando el producto está lo suficientemente maduro. Los objetivos de esa fase es lograr pruebas que satisfaga los requerimientos del cliente, la aprobación de cliente en cuanto a satisfacción del producto y registrar lecciones aprendidas para futuros proyectos.

Disciplinas del Proceso

Gestión de Proyectos:

Esta disciplina explica como planificar y orientar los procesos del proyecto de principio a fin y apoyar al equipo, ayudándolo a lidiar con los riesgos y obstáculos encontrados al crear software.

El propósito de esta disciplina es:

- Estimular la colaboración del equipo en la creación de planes a largo y corto plazo para el proyecto.
- Centrar al equipo en la entrega continua de software probado para la evaluación de los interesados
- Mantener a las partes interesadas y al equipo informados sobre el progreso del proyecto.

Análisis

Esta disciplina explica cómo obtener, analizar, validar y documentar los requisitos para el sistema a desarrollar.

El propósito de esta disciplina es:

- Comprender el problema a resolver
- Recopilar y comprender las necesidades de los interesados
- Definir los requerimientos funcionales y no funcionales del sistema.
- Definir las restricciones y supuestos del proyecto.

Diseño

El objetivo de esta disciplina es desarrollar el diseño del sistema, basado en los requerimientos definidos y validados de la fase de análisis.

El propósito de esta disciplina es

- Desarrollar una arquitectura robusta para el sistema.
- Detallar los componentes que conformaran el sistema
- Diseñar una representación gráfica de los componentes del sistema con el fin de validarlos con el cliente.

Desarrollo

Esta disciplina explica cómo diseñar e implementar una solución técnica que respalde los requisitos y sea acorde al diseño planteado.

El propósito de esta disciplina es:

- Transformar los requerimientos del cliente en un diseño del futuro sistema
- Adaptar el diseño para que coincida con el entorno de implementación.
- Construir el sistema incrementalmente

Pruebas

Esta disciplina explica cómo validar y evaluar la implementación de los componentes del sistema.

El propósito de esta disciplina es:

- Probar si el software hace lo que debe.
- Descubrir un error que aún no ha sido descubierto.
- Encontrar el mayor número de errores con la menor cantidad de tiempo y esfuerzo posibles.
- Validar hasta qué punto las funcionalidades del sistema operan según las especificaciones y requisitos del cliente.

- La disciplina de prueba es iterativa e incremental. Aplica la estrategia de "probar temprano y probar a menudo" para eliminar los riesgos lo antes posible en el ciclo de vida del sistema.

Configuración y Administración de Cambios

Esta disciplina explica cómo controlar los cambios en los artefactos, asegurando la evolución sincronizada del conjunto de productos de trabajo que componen un sistema de software.

El propósito de esta disciplina es:

- Mantener un conjunto constante de productos de trabajo a medida que evolucionan
- Mantener construcciones consistentes del software
- Proporcionar un medio eficiente para adaptarse a cambios y problemas, y volver a planificar el trabajo en consecuencia
- Proporcionar datos para medir el progreso

Roles del Proceso

Es importante determinar cuáles son los actores y cuáles son sus responsabilidades dentro de un proyecto de software. A continuación, se describen los roles definidos para el proceso de desarrollo los cuales están basados en OPEN UP y SCRUM.

Líder de Proyecto:

El líder del proyecto es quien lidera la planeación del proyecto, coordina las actividades del equipo de proyecto y los mantiene enfocados en el alcance de los objetivos del proyecto.

Analista:

El analista es quien tiene relación directa con el cliente y el usuario final y por ende es quien representa sus preocupaciones. Realizar tareas de levantamiento, análisis y diseño de los requerimientos del proyecto.

Desarrollador:

El desarrollador es quien realiza la codificación de los componentes y funcionalidades del sistema basado en los documentos de diseño previamente diseñados.

Tester

El tester es el encargado de realizar las pruebas de calidad, compuestas por pruebas funcionales y no funcionales. Dichas pruebas son documentadas por el tester de calidad en el documento de casos de pruebas.

Dueño del producto

El dueño del producto es quien se encarga de transmitir al líder del proyecto y el equipo las necesidades del cliente. Es quien prioriza los productos o entregables y finalmente quien se encarga de aclarar dudas e inquietudes que tenga el equipo de desarrollo respecto al producto.

Artefactos del Proceso

Los artefactos son instrumentos ya sea físicos o digitales que guían al equipo en el desarrollo del proyecto y a mantenerlo organizado, a planificar cada una de las iteraciones, al trabajo colaborativo y a recopilar un conjunto de información vital para el desarrollo del mismo. Los artefactos del modelo se organizan por disciplinas los cuales se detallan a continuación.

Gestión de Proyectos

Tabla 7: Artefactos Gestión de Proyectos

Nombre Artefacto	Descripción
Documento Plan de Proyecto	Define el alcance, el objetivo, cronograma, riesgos, presupuesto, restricciones y supuestos de alto nivel del producto o proyecto.

Documento Análisis de Riesgos	Es donde se identifican, consignan, priorizan y valoran los riesgos asociados al proyecto,
Cronograma	Representación gráfica y ordenada de las tareas a ejecutarse dentro del proyecto, con tiempos estimados y generalmente recurso humano quien va a realizar la tarea.

Fuente: Elaboración propia

Análisis

Tabla 8: Artefactos Análisis

Nombre Artefacto	Descripción
Documento Especificación de Requerimientos	Capturan los requisitos de calidad, funcionales globales y todos aquellos que no son capturados en escenarios o casos de uso
Documento Especificación Casos de Uso	Captura el comportamiento del sistema desde la perspectiva del usuario final con el fin de para producir un resultado de valor.

Fuente: Elaboración propia

Diseño

Tabla 9: Artefactos Diseño

Nombre Artefacto	Descripción
Documento Arquitectura del Sistema	Se consignan las decisiones, fundamentos, supuestos, explicaciones e implicaciones de la formación de la arquitectura.
Prototipos	Representación gráfica de las funcionalidades o módulos del sistema.
Crear diagrama de actividades	Diagrama de actividades

Fuente: Elaboración propia

Desarrollo

Tabla 10: Artefactos Desarrollo

Nombre Artefacto	Descripción
Componentes del sistema	Componentes de software desarrollados e implementados.

Fuente: Elaboración propia

Pruebas

Tabla 11: Artefactos Pruebas

Nombre Artefacto	Descripción
Documento Casos de Pruebas	Recolecta los resultados de la ejecución de una o más pruebas en un ciclo completo de pruebas

Fuente: Elaboración propia

Eventos del Proceso

Los eventos son reuniones presenciales entre el equipo de trabajo, el líder de desarrollo y el dueño del producto, en donde se busca dar claridad de las necesidades del proyecto, construir en conjunto una propuesta de solución que pueda ser implementada y/o realizar retroalimentaciones. Las reuniones planteadas son una mezcla entre en las metodologías OPEN UP y SCRUM.,

Reunión de inicio

Este evento es una reunión que se lleva a cabo al iniciar un proyecto en la que participa el dueño del producto y el líder del proyecto y tiene como fin definir globalmente el alcance, lo objetivos a alcanzar, tiempos y recursos.

Sprint Planning

En la iteración inicial del proyecto, se reúnen el líder de desarrollo y el analista con el fin de desagregar el proyecto en módulos o funcionalidades más pequeñas, esto con el fin de

realizar un desarrollo iterativo e incremental. Se definen actividades para cada módulo, con tiempos estimados y recurso humano.

Reuniones de cambios urgentes

Son reuniones convocadas generalmente por el dueño del producto en el momento que él crea conveniente, debido a cambios que se presenten en el proyecto por fuerza mayor o para dar prioridad a alguna otra solución que requiera la institución de manera urgente.

Las siguientes iteraciones se realizan por módulos o funcionalidades definidas en la iteración inicial.

Iteración de la fase de inicio

En la iteración de fase de inicio se lleva a cabo entre el líder de desarrollo, el analista y en posible medida el dueño del producto, los cuales realizan una revisión de los artefactos de la fase de inicio para el módulo en el cual se esté trabajando y se realizan retroalimentaciones si es el caso. Esta fase da como oportunidad corregir a una etapa temprana posible errores de análisis del proyecto que se puedan presentar.

Iteración de la fase de diseño

En la iteración de la fase de diseño se lleva a cabo una retroalimentación de los artefactos realizados en la fase de diseño. El líder de desarrollo en conjunto con el analista y el diseñador verifican el resultado de los entregables con el fin de que sea acorde a lo planteado en la fase de análisis.

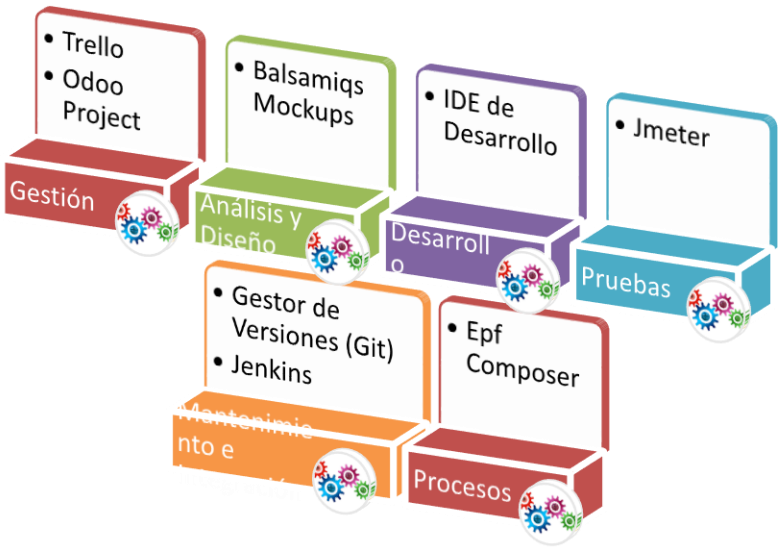
Iteración de la fase de construcción

En la iteración de fase de construcción se lleva a cabo entre el líder de desarrollo, el desarrollador y en posible medida el dueño del producto. Se realizan verificaciones del módulo o funcionalidad desarrollada y se realizan retroalimentaciones. El objetivo principal de la fase de construcción es entregar versiones preliminares del sistema hasta llegar una final

Herramientas del Proceso

Las herramientas del modelo están clasificadas en siete grupos que se detallaran a continuación, su finalidad es facilitar el proceso de implementación del proceso de desarrollo entre los miembros del equipo y estandarizar los procesos de la organización.

Figura 15: Herramientas del Procesos de Desarrollo



Fuente: Imagen propia

Herramientas de Gestión

Entre las herramientas utilizadas para la gestión de los proyectos están Trello y eventualmente Odoo Project. Trello es una herramienta de gestión de proyectos basado en el modelo kanban que permite gestionar tareas permitiendo organizar el trabajo de grupo de forma colaborativa por medio de tableros virtuales y que permite además compartirla con diferentes personas que formen el proyecto (https://trello.com). Odoo projects es un módulo de gestión de proyectos de la herramienta Odoo, que permite desglosar un proyecto en tareas y ser asignadas a un equipo de trabajo mejorando así la colaboración (https://www.odoo.com/).

Herramientas de Análisis y Diseño

En análisis y diseño se cuenta con una herramienta paga llamada Balsamiq Mockups. Con dicha herramienta es posible realizar diagrama de análisis como casos de uso y de diseño como diagramas de actividades, de clases, de componentes, etc (https://balsamiq.com/).

Herramientas de Desarrollo

El entorno de desarrollo principal es Netbeans versión 7.4 y 8 para los proyectos en Java y Visual Studio Code versión 1.3 o superior para los proyectos en con Node.js y Angular.

Herramientas de Pruebas

Para la automatización de pruebas, pruebas funcionales y de carga se utiliza la herramienta JMeter. JMeter es una herramienta que permite desde un ambiente local realizar pruebas de carga, estrés y que permiten diagnosticar el comportamiento de una aplicación en condiciones de producción (https://jmeter.apache.org/).

Herramientas de Mantenimiento e Integración

Una de las mayores preocupaciones entre los miembros de un equipo de desarrollo es la integración de los componentes y el mantenimiento del software, para esto, se usan herramientas que permitan llevar control de versionado, control de cambios de los diferentes artefactos, gestión de trabajo colaborativo y mantenimiento de una versión estable para ser distribuida y versiones de desarrollo y pruebas. Entre las herramientas que ofrece el mercado las seleccionadas para dichos fines son git y Jenkins.

Herramientas de Proceso

Para el modelamiento de procesos se utiliza la herramienta Epf Composer, la cual permite modelar el proceso de desarrollo de software y definir tareas, artefactos para cada tarea y el rol que debe realizar dicha tarea. Dicha herramienta permite que el equipo de desarrollo cuente con una base de conocimiento y una guía para el desarrollo de software desde las etapas iniciales hasta entrega del producto.

9.5 FASE 5: IMPLEMENTACIÓN

9.5.1 Implementar el Proceso de Desarrollo en el Proyecto Seleccionado

En esta sección se describen las actividades que se llevaron a acabo para la implantación del nuevo proceso de desarrollo de la Universidad de Manizales.

Montaje y presentación del proceso de desarrollo en herramienta navegable

El proceso de desarrollo diseñado en la herramienta EPF Composer se publicó en uno de los servidores privados de la Universidad de Manizales, esto con el fin de que sea accesible solo al equipo de desarrollo de la Universidad.

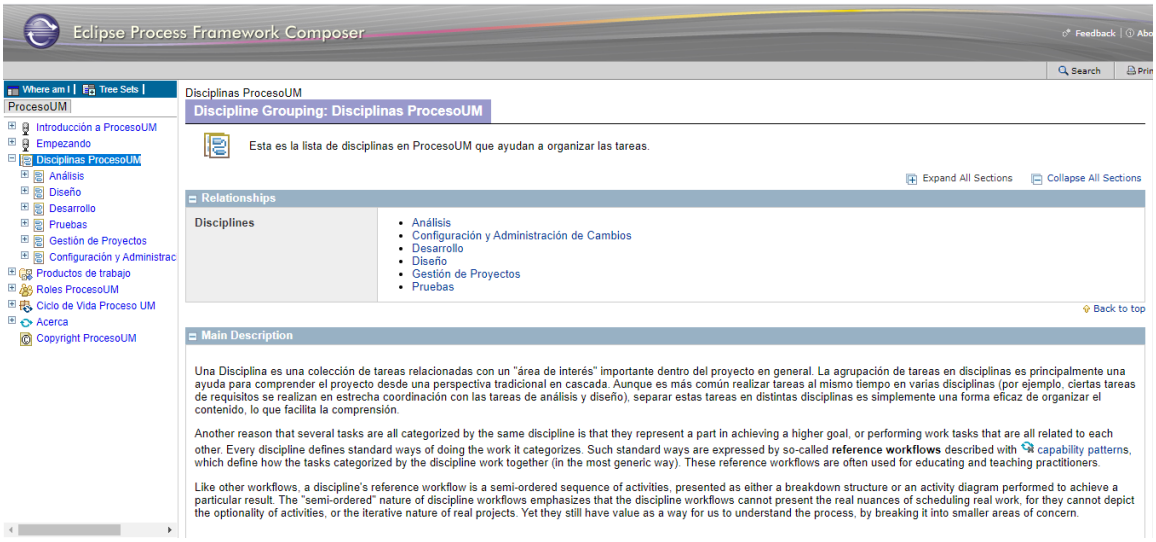
En la herramienta se definieron todos los elementos que componen el proceso, comenzando por las disciplinas, los productos de trabajo, los roles, los eventos y terminando con las iteraciones.

EPF Composer genera una vista HTML que permite la navegabilidad entre cada uno de los elementos anteriormente mencionados y su consulta entre los miembros de equipo. Las siguientes figuras muestran una parte del proceso en la herramienta.

Figura 16: Vista Epf composer Introducción a ProcesoUM

Fuente: Imagen propia

Figura 17: Vista Epf composer Disciplinas ProcesoUM

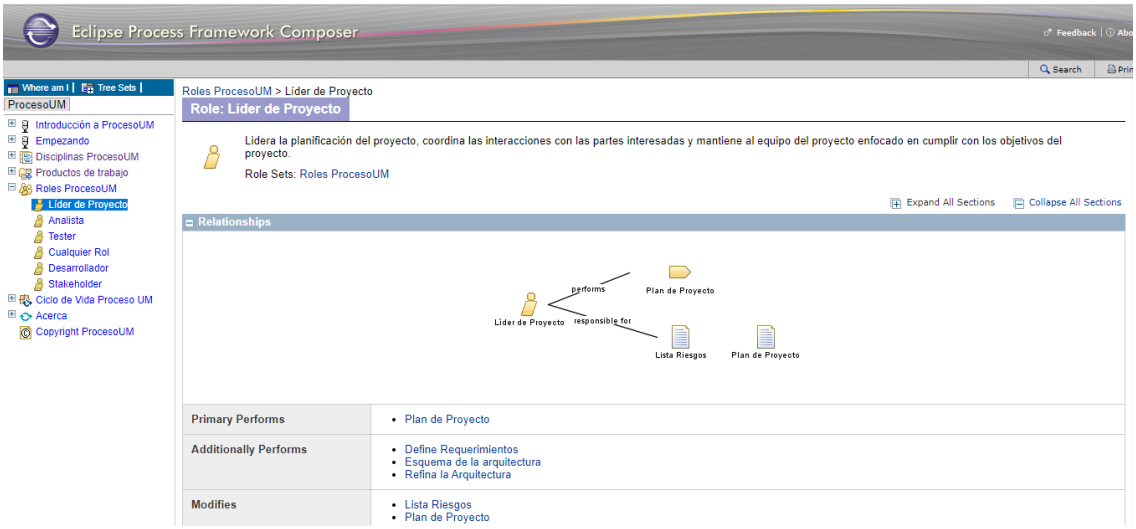


Fuente: Imagen propia

Figura 18: Vista Epf composer Plan de Proyecto ProcesoUM

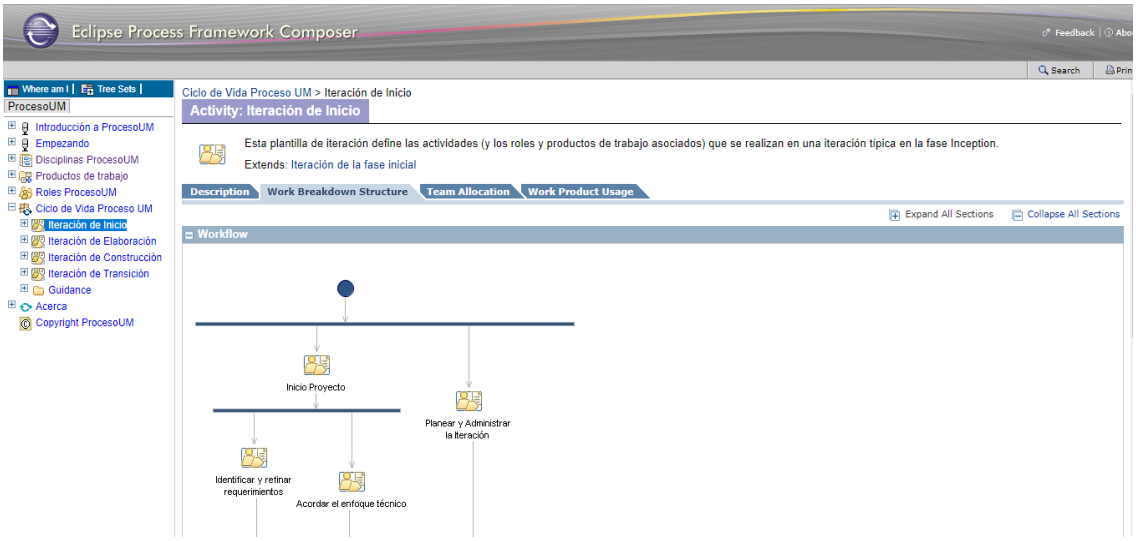
Fuente: Imagen propia

Figura 19: Vista Epf composer Roles del ProcesoUM



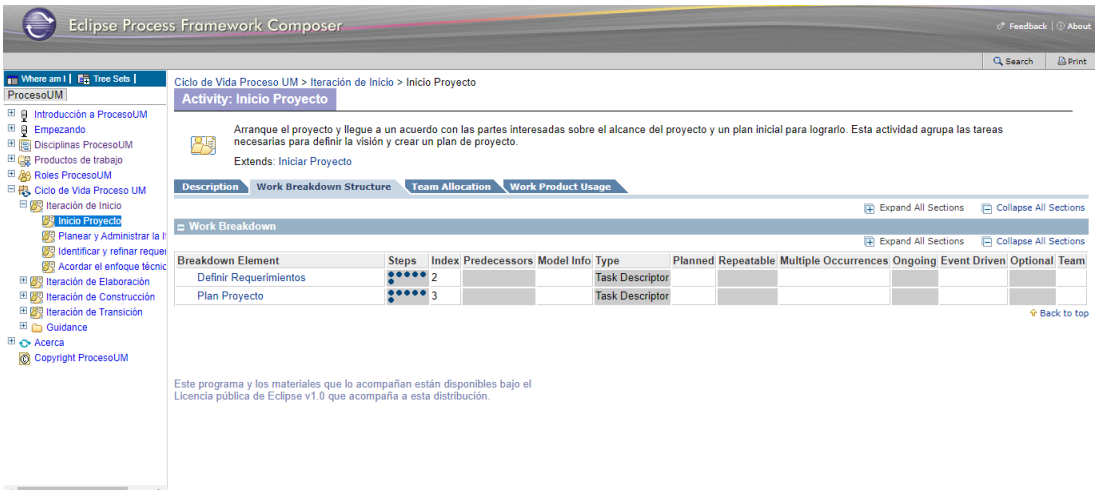
Fuente: Imagen propia

Figura 20: Vista Epf composer Ciclo de Vida ProcesoUM



Fuente: Imagen propia

Figura 21: Vista Epf composer Inicio del Proyecto ProcesoUM



Fuente: Imagen propia

Figura 22: Vista Epf composer Eventos ProcesoUM

Fuente: Imagen propia

Acceso a la herramienta navegable

El proceso de desarrollo de la Universidad de Manizales, denominado **procesoUM** fue publicado en uno de los servidores de aplicaciones con que cuenta la Universidad, este sitio navegable solo es visible y puede ser navegable para los funcionarios del área de T.I y sus directivas a través de la siguiente dirección:

https://172.28.20.93/procesoUM/

Socialización del proceso de desarrollo

La socialización del proceso de desarrollo se llevó a cabo los días 27 y 28 de junio del 2019 y contó con la participación de todos los miembros del área de T.I y sus directivas. En dicha socialización se trataron temas como la importancia de proceso de desarrollo tanto para el área como para la Universidad, ventajas, impactos esperados y posibles mejoras. Adicionalmente, se mostró la nueva forma de trabajo del área y se explicó cada uno de sus componentes.

Implementación y primer uso del nuevo proceso de desarrollo en proyecto piloto

A continuación, se describen de los resultados obtenidos al usar el nuevo proceso en un proyecto de desarrollo del área de T.I

Descripción del proyecto

Gestión Académica (G.A) es un sistema desarrollado para la generación de los compromisos académicos de los docentes de la Universidad de Manizales. G.A permite la gestión de horarios regulares e irregulares y procesos misionales como investigación, proyección social y gestión. Permite la consulta, edición y aprobación de los compromisos académicos y la generación de reportes de dichos compromisos en un periodo académico.

Descripción de las actividades realizadas en el proyecto

El desarrollo de Gestión Académica se hizo bajo los lineamientos del proceso de desarrollo, por ende, se realizaron todas las actividades referentes a dicho proceso, las cuales se describen a continuación:

Fases

Se llevaron a cabo las fases definidas en el proceso: inicio, elaboración, construcción y transición.

En la **Fase de inicio** se definió el alcance, los objetivos, se identificaron y definieron requerimientos funcionales y no funcionales y riegos iniciales del proyecto. Los resultados de esta fase fueron entregables como el documento de plan de proyecto, análisis de riegos preliminar, requerimientos, especificación de casos de uso y cronograma inicial. En esta fase la participación del equipo de desarrollo y directivas fue vital ya que para cada entregable fue necesario el conocimiento y la experticia de cada uno de ellos.

En la **Fase de elaboración** se llevaron a cabo la tarea de análisis y arquitectura y se refinaron aspectos como requerimientos funcionales y no funcionales, cronograma, y riesgos. Los resultados de esta fase fueron la actualización de artefactos como documento de requerimientos, riesgos y especificación de casos de uso y artefactos como mockups y arquitectura. Adicionalmente, se definieron aspectos como infraestructura, entorno de desarrollo y herramientas a utilizar para el desarrollo del producto de software.

En la **Fase de construcción** se llevaron a cabo actividades de implementación y pruebas de los componentes y funcionalidades del sistema. Esta fase generó entregables tales como componentes del sistema y documento de pruebas.
En la **Fase de transición** se realizaron actividades de despliegue en puesta en producción de Gestión Académica.

Eventos

Se llevó a cabo la **reunión de inicio** entre el dueño del producto y el líder de desarrollo y en donde se definieron de forma global el alcance, lo objetivos a alcanzar, tiempos y recursos.
Se realizó un **Sprint Planning** entre el líder del desarrollo y el analista con el fin de desagregar el proyecto en módulos o funcionalidades más pequeñas.
Se realizaron iteraciones para las fases del proceso tales como iteración de inicio, diseño, construcción. En cada iteración se revisaron actividades y entregables con el fin de ser validados y así poder continuar con las demás actividades.

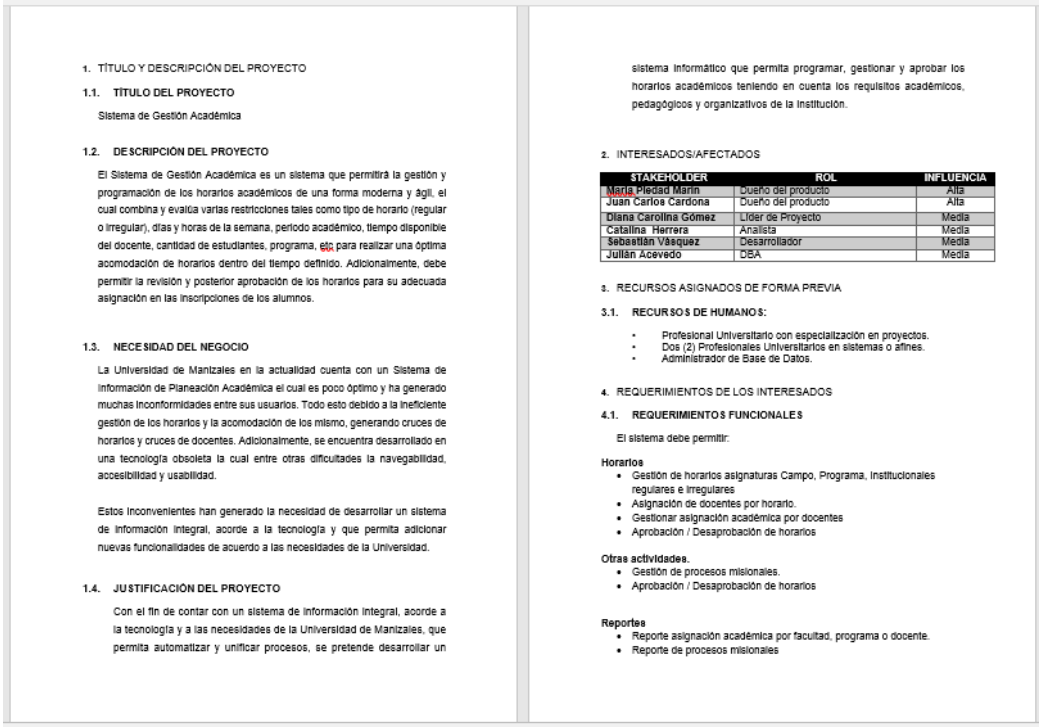
Artefactos

Los artefactos generados de acuerdo con el proceso de desarrollo implementados se describen a continuación:

Plan de Proyecto

En el artefacto Plan de Proyecto se definió y consignó el alcance, el objetivo, cronograma, riesgos, presupuesto, restricciones y supuestos de alto nivel del sistema **Gestión Académica**.

Figura 23: Plan de Proyecto Gestión Académica



1. TÍTULO Y DESCRIPCIÓN DEL PROYECTO

1.1. TÍTULO DEL PROYECTO

Sistema de Gestión Académica

1.2. DESCRIPCIÓN DEL PROYECTO

El Sistema de Gestión Académica es un sistema que permitirá la gestión y programación de los horarios académicos de una forma moderna y ágil, el cual combina y evalúa varias restricciones tales como tipo de horario (regular o irregular), días y horas de la semana, periodo académico, tiempo disponible del docente, cantidad de estudiantes, programa, etc para realizar una óptima acomodación de horarios dentro del tiempo definido. Adicionalmente, debe permitir la revisión y posterior aprobación de los horarios para su adecuada asignación en las inscripciones de los alumnos.

1.3. NECESIDAD DEL NEGOCIO

La Universidad de Manizales en la actualidad cuenta con un Sistema de Información de Planeación Académica el cual es poco óptimo y ha generado muchas inconformidades entre sus usuarios. Todo esto debido a la ineficiente gestión de los horarios y la acomodación de los mismo, generando cruces de horarios y cruces de docentes. Adicionalmente, se encuentra desarrollado en una tecnología obsoleta la cual entre otras dificultades la navegabilidad, accesibilidad y usabilidad.

Estos inconvenientes han generado la necesidad de desarrollar un sistema de información integral, acorde a la tecnología y que permita adicionar nuevas funcionalidades de acuerdo a las necesidades de la Universidad.

1.4. JUSTIFICACIÓN DEL PROYECTO

Con el fin de contar con un sistema de información integral, acorde a la tecnología y a las necesidades de la Universidad de Manizales, que permita automatizar y unificar procesos, se pretende desarrollar un sistema informático que permita programar, gestionar y aprobar los horarios académicos teniendo en cuenta los requisitos académicos, pedagógicos y organizativos de la institución.

2. INTERESADOS/AFECTADOS

STAKEHOLDER	ROL	INFLUENCIA
María Piedad Marín	Dueño del producto	Alta
Juan Carlos Cardona	Dueño del producto	Alta
Diana Carolina Gómez	Líder de Proyecto	Media
Catalina Herrera	Analista	Media
Sebastián Vásquez	Desarrollador	Media
Julián Acevedo	DBA	Media

3. RECURSOS ASIGNADOS DE FORMA PREVIA

3.1. RECURSOS DE HUMANOS:

- Profesional Universitario con especialización en proyectos.
- Dos (2) Profesionales Universitarios en sistemas o afines.
- Administrador de Base de Datos.

4. REQUERIMIENTOS DE LOS INTERESADOS

4.1. REQUERIMIENTOS FUNCIONALES

El sistema debe permitir:

Horarios

- Gestión de horarios asignaturas Campo, Programa, Institucionales regulares e irregulares
- Asignación de docentes por horario.
- Gestionar asignación académica por docentes
- Aprobación / Desaprobación de horarios

Otras actividades.

- Gestión de procesos misionales.
- Aprobación / Desaprobación de horarios

Reportes

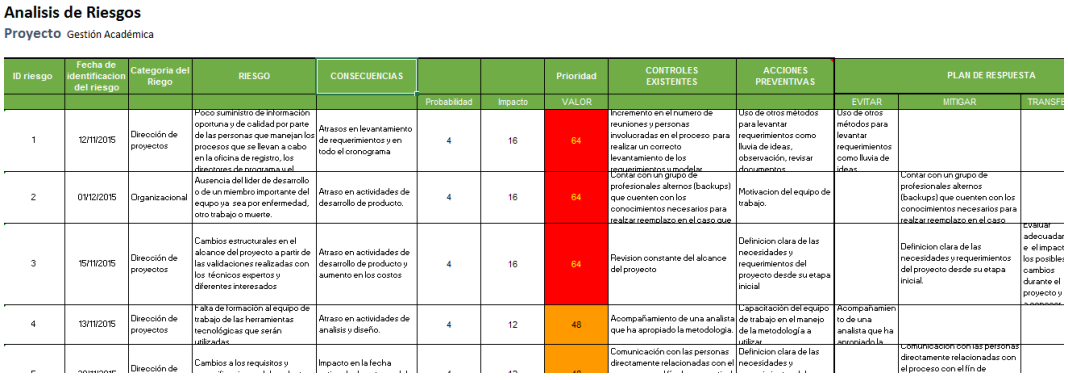
- Reporte asignación académica por facultad, programa o docente.
- Reporte de procesos misionales

Fuente: Imagen propia

Análisis de Riesgos

En el documento análisis de riesgos se identificaron, clasificaron y consignan los riesgos asociados al proyecto Gestión Académica.

Figura 24: Análisis de Riegos Gestión Académica



Analisis de Riesgos

Proyecto Gestión Académica

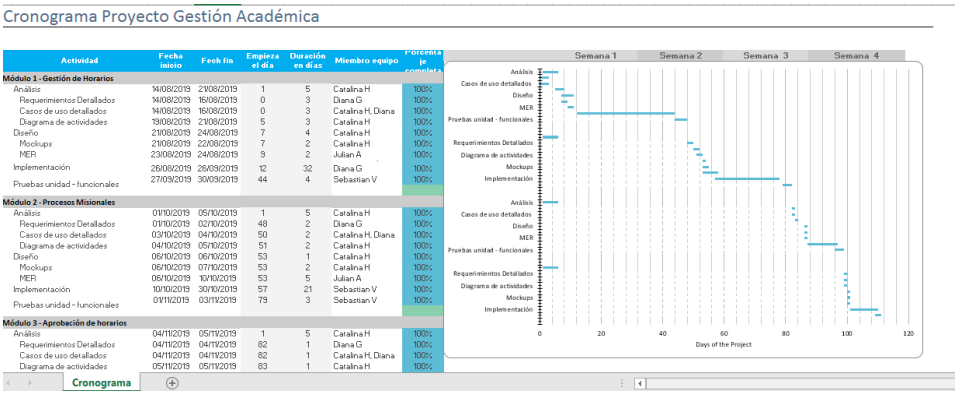
ID riesgo	Fecha de identificacion del riesgo	Categoria del Riego	RIESGO	CONSECUENCIAS			Prioridad	CONTROLES EXISTENTES	ACCIONES PREVENTIVAS	PLAN DE RESPUESTA		
					Probabilidad	Impacto	VALOR			EVITAR	MITIGAR	TRANSFE
1	12/11/2015	Dirección de proyectos	Poco suministro de información oportuna y de calidad por parte de las personas que manejan los procesos que se llevan a cabo en la oficina de registro, los directores de programa y el	Atrasos en levantamiento de requerimientos y en todo el cronograma	4	16	64	Incremento en el numero de reuniones y personas involucradas en el proceso para realizar un correcto levantamiento de los requerimientos y modelar	Uso de otros métodos para levantar requerimientos como lluvia de ideas, observación, revisar documentos	Uso de otros métodos para levantar requerimientos como lluvia de ideas		
2	01/12/2015	Organizacional	Ausencia del líder de desarrollo o de un miembro importante del equipo ya sea por enfermedad, otro trabajo o muerte.	Atraso en actividades de desarrollo de producto.	4	16	64	Contar con un grupo de profesionales alternos (backups) que cuenten con los conocimientos necesarios para realizar reemplazo en el caso que	Motivacion del equipo de trabajo.		Contar con un grupo de profesionales alternos (backups) que cuenten con los conocimientos necesarios para realizar reemplazo en el caso	
3	15/11/2015	Dirección de proyectos	Cambios estructurales en el alcance del proyecto a parte de las validaciones realizadas con los técnicos expertos y diferentes interesados	Atraso en actividades de desarrollo de producto y aumento en los costos	4	16	64	Revision constante del alcance del proyecto	Definicion clara de las necesidades y requerimientos del proyecto desde su etapa inicial		Definicion clara de las necesidades y requerimientos del proyecto desde su etapa inicial.	Evaluar adecuadar e el impact los posible cambios durante el proyecto y
4	13/11/2015	Dirección de proyectos	Falta de formación al equipo de trabajo de las herramientas tecnológicas que serán utilizadas	Atraso en actividades de analisis y diseño.	4	12	48	Acompañamiento de una analista que ha apropiado la metodología.	Capacitación del equipo de trabajo en el manejo de la metodología a utilizar	Acompañamien to de una analista que ha apropiado la		
5	20/11/2015	Dirección de	Cambios a los requisitos y	Impacto en la fecha	4	12	48	Comunicación con las personas directamente relacionadas con el	Definicion clara de las necesidades y		Comunicación con las personas directamente relacionadas con el proceso con el fin de	

Fuente: Imagen propia

Cronograma

En el artefacto cronograma se consignaron las actividades relacionadas al ciclo de vida del desarrollo del software Gestión Académica, con tiempos y recursos humano designado a realizar dichas actividades.

Figura 25: Cronograma Gestión de Proyectos

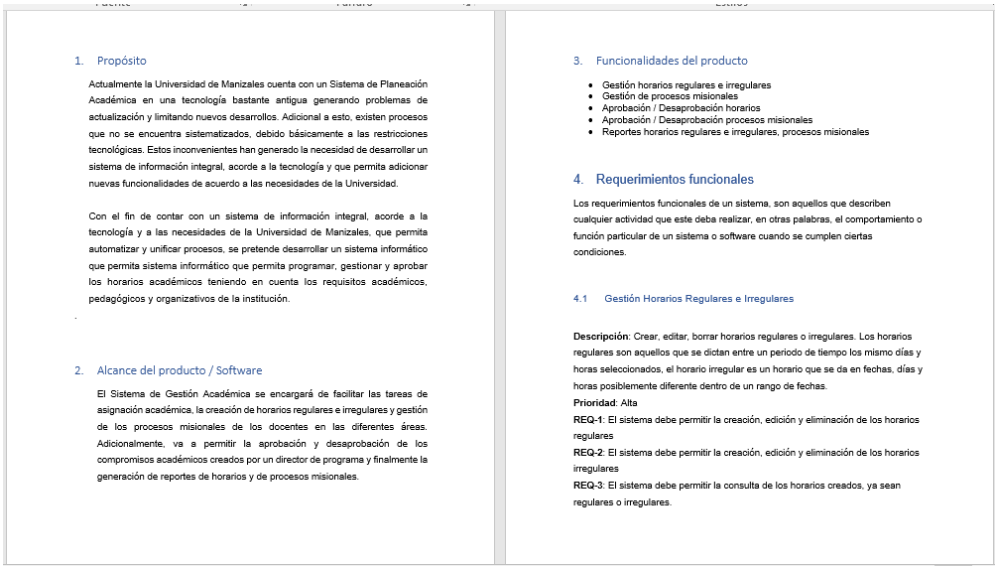


Fuente: Imagen propia

Especificación de Requerimientos

En el documento especificación se consignaron todos los requerimientos funcionales, no funcionales, restricciones y supuestos del proyecto Gestión Académica.

Figura 26: Especificación de Requerimientos Gestión Académica



1. Propósito

Actualmente la Universidad de Manizales cuenta con un Sistema de Planeación Académica en una tecnología bastante antigua generando problemas de actualización y limitando nuevos desarrollos. Adicional a esto, existen procesos que no se encuentra sistematizados, debido básicamente a las restricciones tecnológicas. Estos inconvenientes han generado la necesidad de desarrollar un sistema de información integral, acorde a la tecnología y que permita adicionar nuevas funcionalidades de acuerdo a las necesidades de la Universidad.

Con el fin de contar con un sistema de información integral, acorde a la tecnología y a las necesidades de la Universidad de Manizales, que permita automatizar y unificar procesos, se pretende desarrollar un sistema informático que permita sistema informático que permita programar, gestionar y aprobar los horarios académicos teniendo en cuenta los requisitos académicos, pedagógicos y organizativos de la institución.

2. Alcance del producto / Software

El Sistema de Gestión Académica se encargará de facilitar las tareas de asignación académica, la creación de horarios regulares e irregulares y gestión de los procesos misionales de los docentes en las diferentes áreas. Adicionalmente, va a permitir la aprobación y desaprobación de los compromisos académicos creados por un director de programa y finalmente la generación de reportes de horarios y de procesos misionales.

3. Funcionalidades del producto

- Gestión horarios regulares e irregulares
- Gestión de procesos misionales
- Aprobación / Desaprobación horarios
- Aprobación / Desaprobación procesos misionales
- Reportes horarios regulares e irregulares, procesos misionales

4. Requerimientos funcionales

Los requerimientos funcionales de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

4.1 Gestión Horarios Regulares e Irregulares

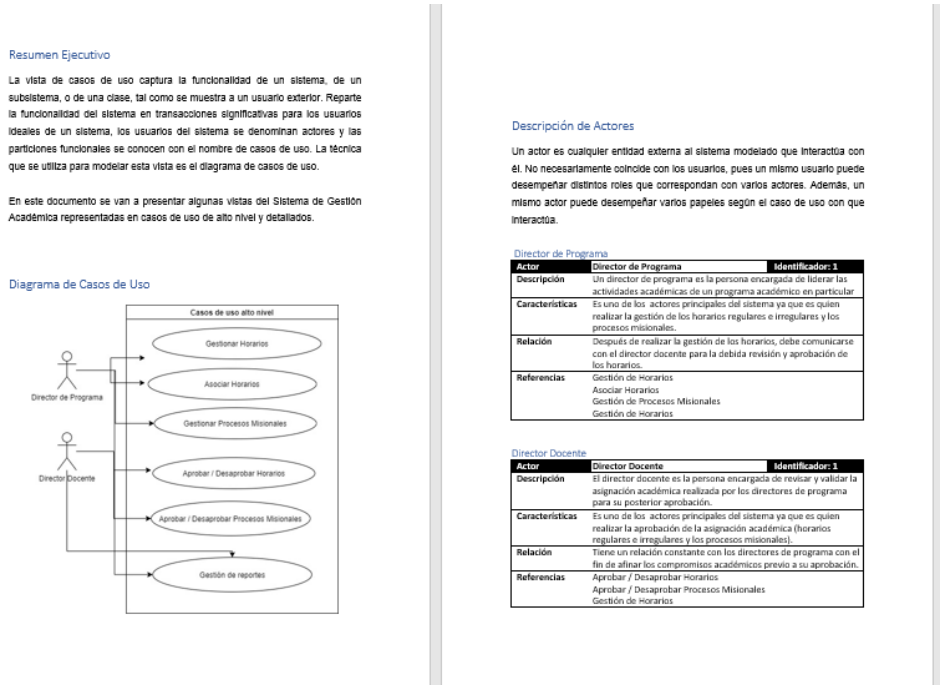
Descripción: Crear, editar, borrar horarios regulares o irregulares. Los horarios regulares son aquellos que se dictan entre un periodo de tiempo los mismo días y horas seleccionados, el horario irregular es un horario que se da en fechas, días y horas posiblemente diferente dentro de un rango de fechas.
Prioridad: Alta
REQ-1: El sistema debe permitir la creación, edición y eliminación de los horarios regulares
REQ-2: El sistema debe permitir la creación, edición y eliminación de los horarios irregulares
REQ-3: El sistema debe permitir la consulta de los horarios creados, ya sean regulares o irregulares.

Fuente: Imagen propia

Especificación Casos de Uso

En el documento de especificación de casos de uso se consignó el diagrama de casos de uso de alto nivel y se detallaron los casos de uso del sistema Gestión Académica.

Figura 27: Especificación Casos de Uso Gestión Académica



Resumen Ejecutivo

La vista de casos de uso captura la funcionalidad de un sistema, de un subsistema, o de una clase, tal como se muestra a un usuario exterior. Reparte la funcionalidad del sistema en transacciones significativas para los usuarios ideales de un sistema, los usuarios del sistema se denominan actores y las particiones funcionales se conocen con el nombre de casos de uso. La técnica que se utiliza para modelar esta vista es el diagrama de casos de uso.

En este documento se van a presentar algunas vistas del Sistema de Gestión Académica representadas en casos de uso de alto nivel y detallados.

Diagrama de Casos de Uso

Descripción de Actores

Un actor es cualquier entidad externa al sistema modelado que interactúa con él. No necesariamente coincide con los usuarios, pues un mismo usuario puede desempeñar distintos roles que correspondan con varios actores. Además, un mismo actor puede desempeñar varios papeles según el caso de uso con que interactúa.

Director de Programa

Actor	Director de Programa	Identificador: 1
Descripción	Un director de programa es la persona encargada de liderar las actividades académicas de un programa académico en particular	
Características	Es uno de los actores principales del sistema ya que es quien realizar la gestión de los horarios regulares e irregulares y los procesos misionales.	
Relación	Después de realizar la gestión de los horarios, debe comunicarse con el director docente para la debida revisión y aprobación de los horarios.	
Referencias	Gestión de Horarios Asociar Horarios Gestión de Procesos Misionales Gestión de Horarios	

Director Docente

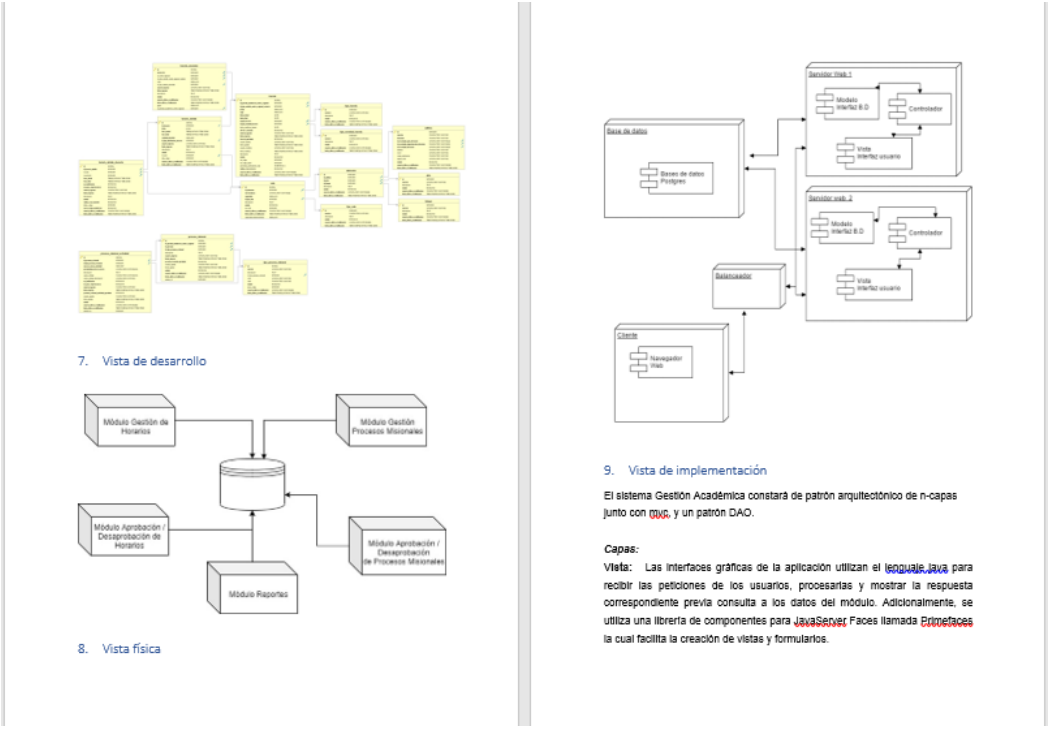
Actor	Director Docente	Identificador: 1
Descripción	El director docente es la persona encargada de revisar y validar la asignación académica realizada por los directores de programa para su posterior aprobación.	
Características	Es uno de los actores principales del sistema ya que es quien realizar la aprobación de la asignación académica (horarios regulares e irregulares y los procesos misionales).	
Relación	Tiene un relación constante con los directores de programa con el fin de afinar los compromisos académicos previo a su aprobación.	
Referencias	Aprobar / Desaprobar Horarios Aprobar / Desaprobar Procesos Misionales Gestión de Horarios	

Fuente: Imagen propia

Arquitectura

El documento de arquitectura contiene la vista lógica, vista física, vista de implementación del sistema Gestión Académica.

Figura 28: Documento Arquitectura Gestión Académica



7. Vista de desarrollo

8. Vista física

9. Vista de implementación

El sistema Gestión Académica constará de patrón arquitectónico de n-capas junto con mvc, y un patrón DAO.

Capas:

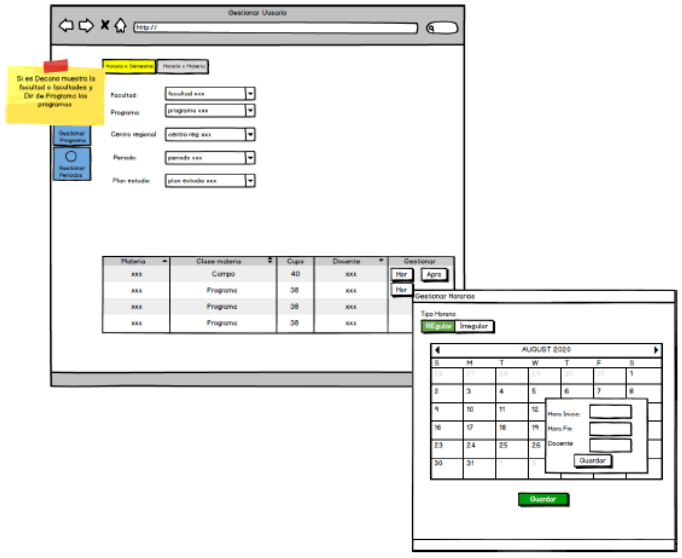
Vista: Las interfaces gráficas de la aplicación utilizan el lenguaje java para recibir las peticiones de los usuarios, procesarlas y mostrar la respuesta correspondiente previa consulta a los datos del módulo. Adicionalmente, se utiliza una librería de componentes para JavaServer Faces llamada Primefaces la cual facilita la creación de vistas y formularios.

Fuente: Imagen propia

Prototipos

El artefacto prototipos contiene la implementación parcial de todos los módulos del sistema y sus funcionalidades.

Figura 29: Prototipos

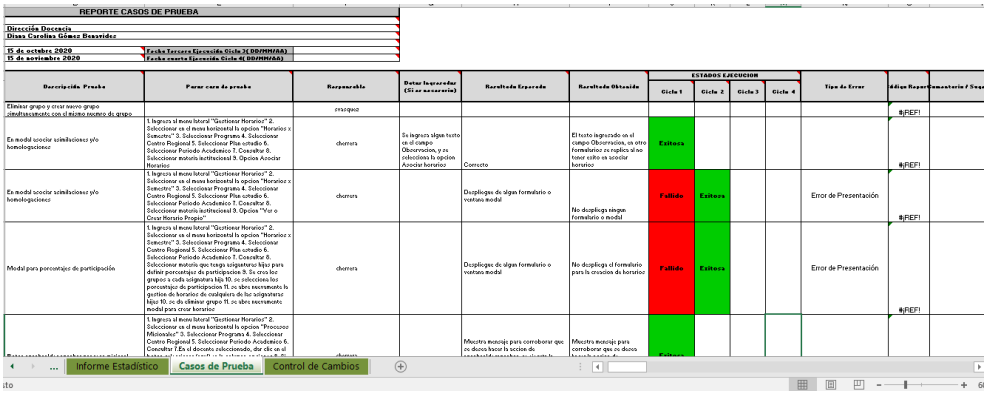


Fuente: Imagen propia

Registro de Pruebas

En el registro de pruebas se consignaron todas las pruebas funcionales que se le realizaron a cada módulo del sistema en sus diferentes ciclos de pruebas. Cada registro contó con un seguimiento específico más aún cuando un registro se reportaba como fallido.

Figura 30: Registro de Pruebas



Fuente: Imagen propia

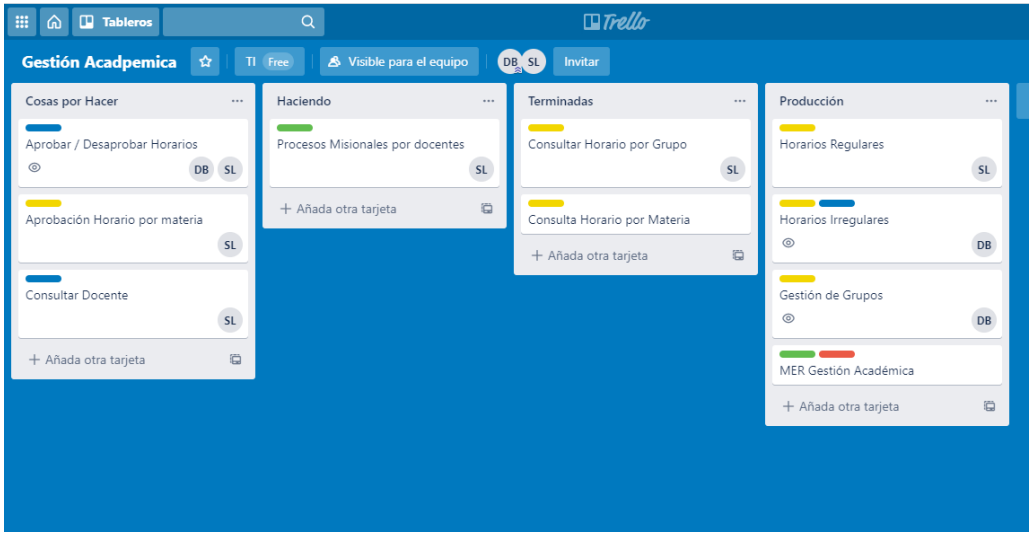
Herramientas tecnológicas

Las herramientas tecnológicas usadas para la implementación del sistema Gestión Académica se describen a continuación:

Trello

La gestión del proyecto Gestión Académica se llevó a cabo en la herramienta Trello. En ella se consignaron cada una de las actividades provenientes del cronograma y a las cuales se les asignó el recurso humano quien realizaría dicha tarea. Finalmente, cada tarea navegó por cada una de las columnas definidas en la herramienta permitiendo llevar debido seguimiento y control (Cosas por Hacer, Haciendo, Terminadas, Producción).

Figura 31: Trello Gestión Académica

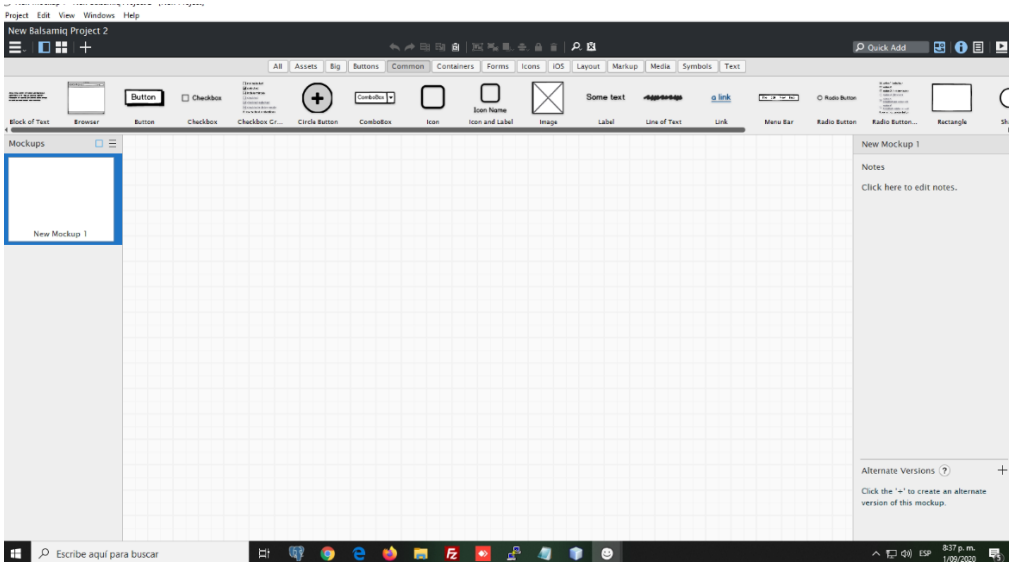


Fuente: Imagen propia

Balsamiq

Esta herramienta permitió realizar todos los diseños de los prototipos del sistema y por a su facilidad de manejo y flexibilidad permitió que la validación de los prototipos fuera de muy eficiente y eficaz.

Figura 32: Balsamiq Gestión Académica

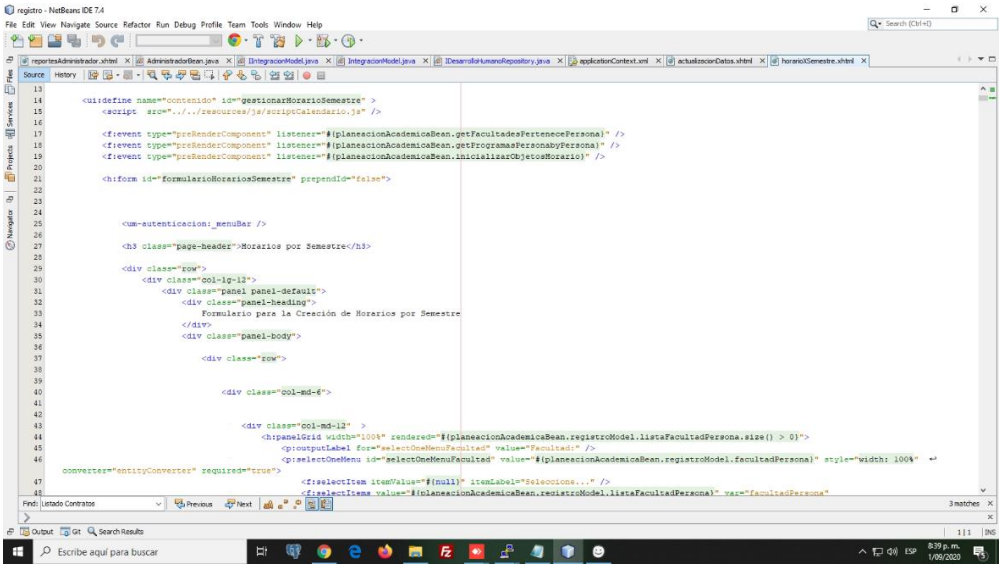


Fuente: Imagen propia

Netbeans

Se eligió y utilizó Netbeans como plataforma o IDE de desarrollo, esto debido a su facilidad para el desarrollo de aplicaciones en Java.

Figura 33: Netbeans Gestión Académica

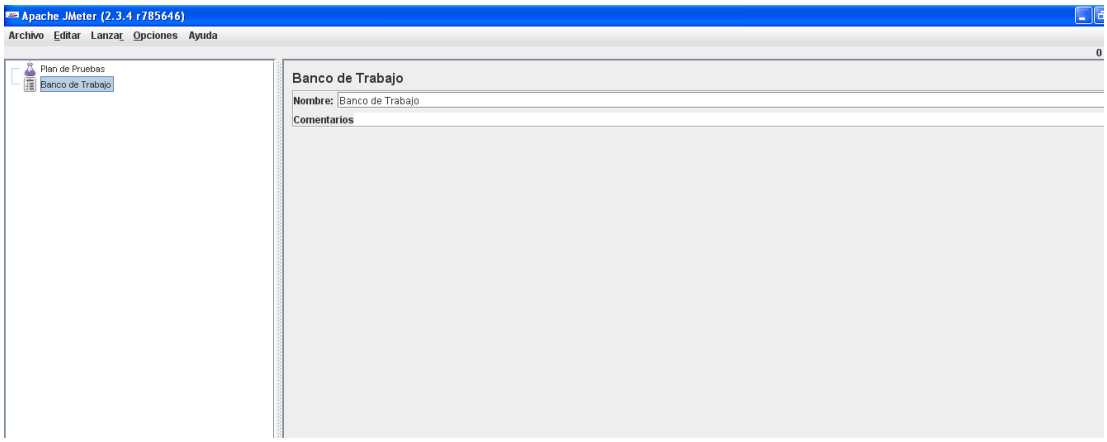


Fuente: Imagen propia

Jmeter

Para la realización de pruebas de carga y estrés se utilizó la herramienta de pruebas Jmeter.

Figura 34: Jmeter Gestión Académica

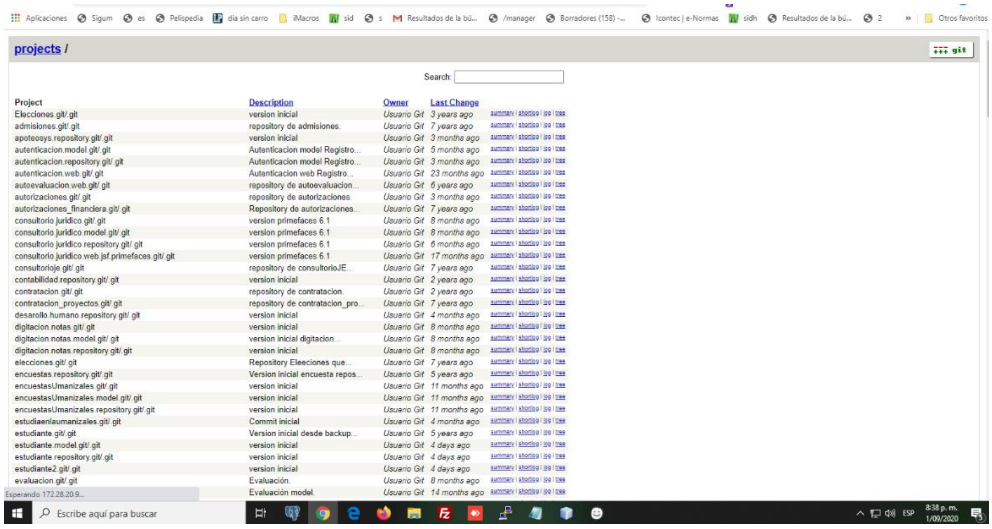


Fuente: Imagen propia

Git

Para el mantenimiento y gestión de versiones del aplicativo, se utilizó la herramienta git la cual fue instalada y configurar en uno de los servidores de la Universidad de Manizales**.**

Figura 35: Git Gestión Académica



Fuente: Imagen propia

Epf Composer

Finalmente, para el montaje y consulta del proceso de desarrollo se utilizó la herramienta Epf Composer.

Figura 36: Epf Composer Gestión Académica

Fuente: Imagen propia

Resultado final proyecto de software

La implementación de proceso de desarrollo en el Sistema de Gestión Académica generó entre algunas evidencias los artefactos y el producto final. El producto final fue debidamente publicado en los servidores de la Universidad de Manizales para su debido uso y la gestión de los compromisos académicos del 2020-1 y posteriores, haciendo parte integral del Sistema de Información Gerencial de la Universidad de Manizales (SIGUM). A continuación, se ilustra el Sistema de Gestión Académica:

Login

El acceso al Sistema de Gestión Académica se hace a través de SIGUM quien es el encargado de verificar usuarios, perfiles y de redirecciones a un usuario a los aplicativos a los cuales tienes permiso de acceso.

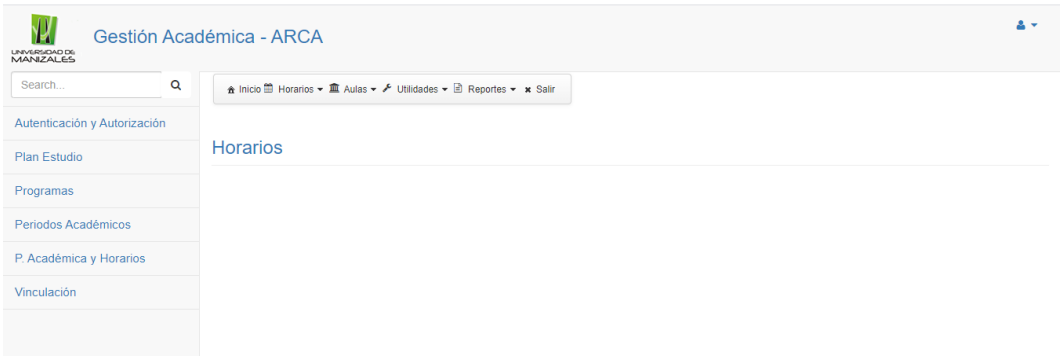
Figura 37: Login Sigum

Fuente: Imagen propia

Gestión de Horarios

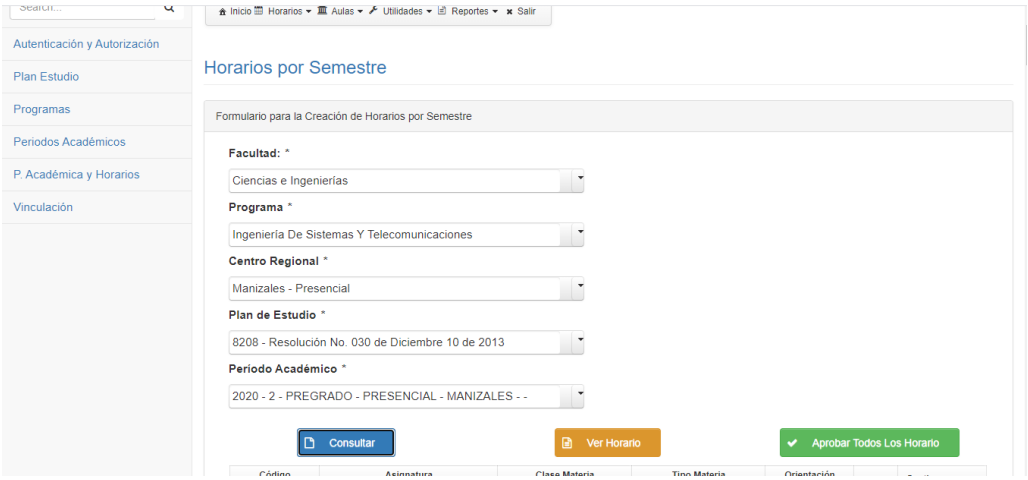
En esta sesión se gestionan (crear, actualizar, eliminar) los horarios tanto regulares como irregulares de un programa dentro de un plan de estudio y un periodo académico. Adicionalmente, se generan reportes de horarios, de docentes y se asocian horarios.

Figura 38: Gestión Académica



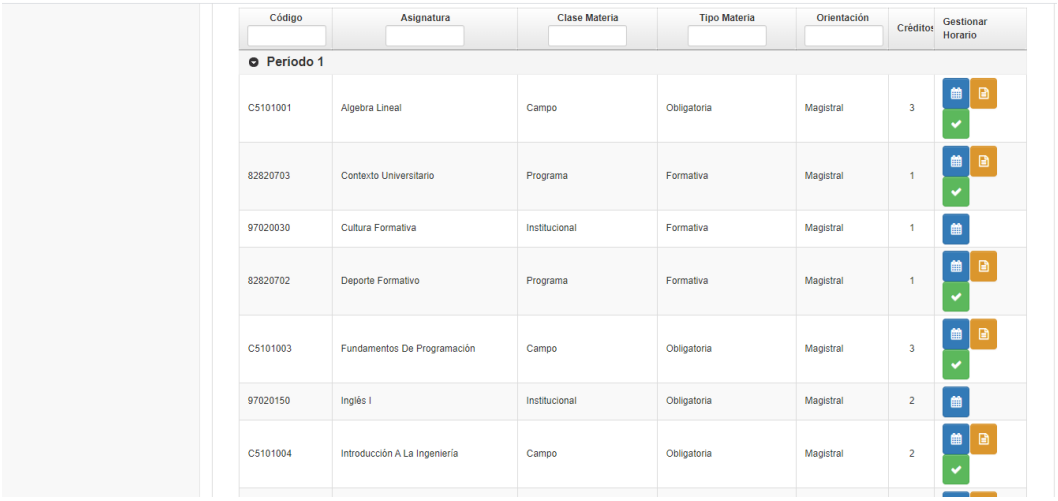
Fuente: Imagen propia

Figura 39: Gestión Académica Gestión Horarios



Fuente: Imagen propia

Figura 40: Gestión Académica Gestión Horarios II

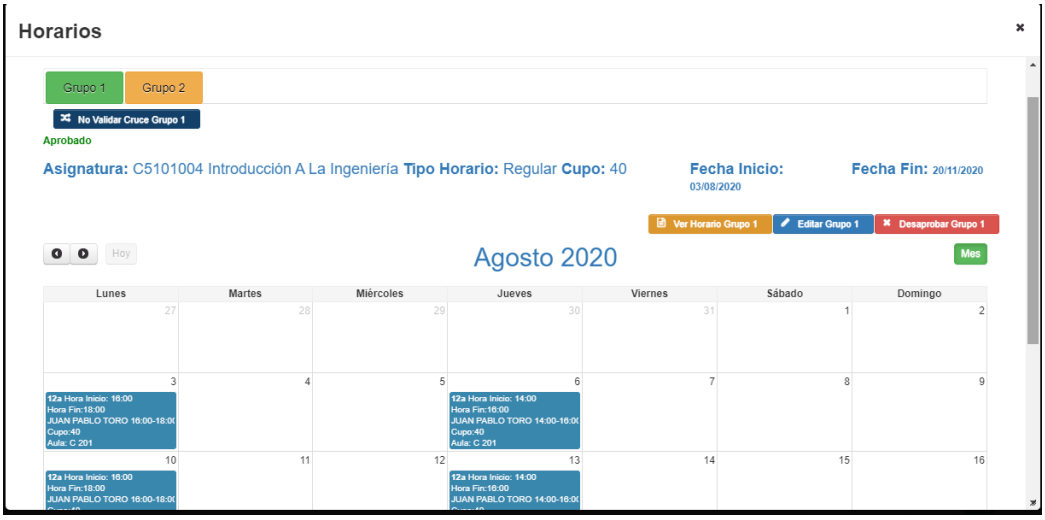


Fuente: Imagen propia

Figura 41: Gestión Académica Gestión Horarios III

Fuente: Imagen propia

Figura 42: Gestión Académica Gestión Horarios IV

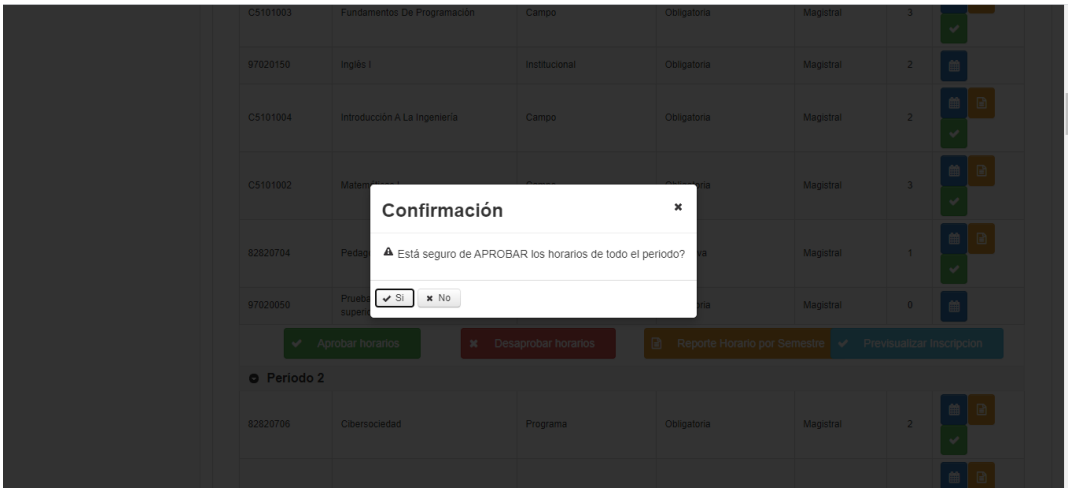


Fuente: Imagen propia

Aprobar / Desaprobar Horarios

En esta sesión la directora docente aprueba o desaprueba los compromisos académicos de un docente y a su vez, aprueba y/o desaprueba horarios regulares e irregulares.

Figura 43: Aprobar / Desaprobar Horarios

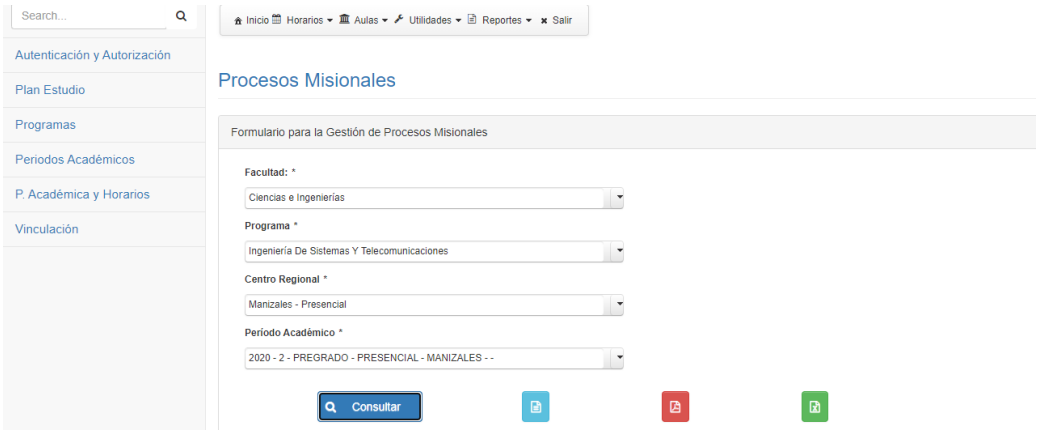


Fuente: Imagen propia

Procesos Misionales

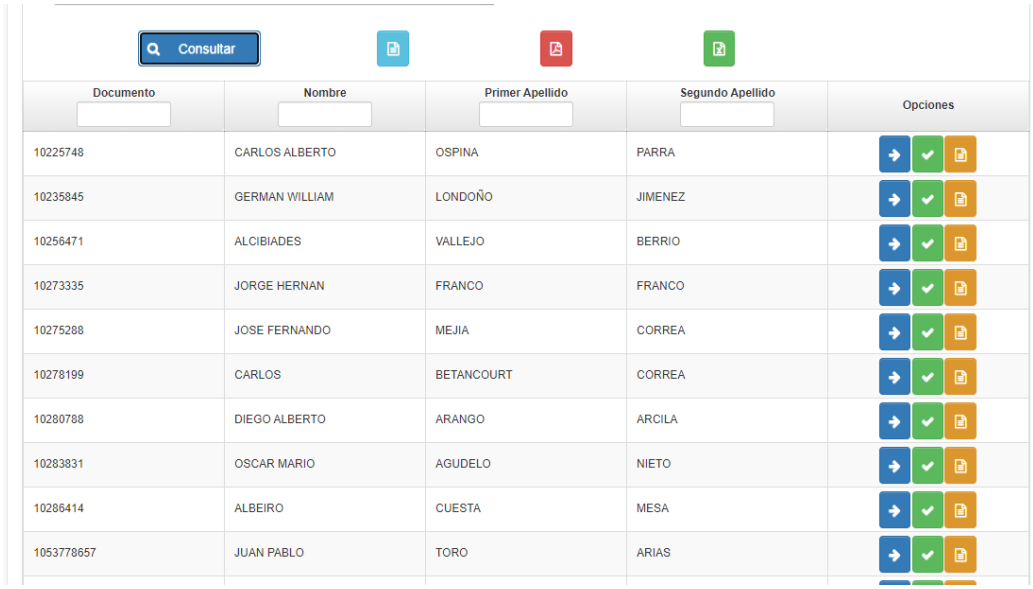
En esta sesión los directores de programa gestionan los compromisos académicos de uno o varios docentes que pertenecen a un programa y dentro de un periodo académico.

Figura 44: Procesos Misionales



Fuente: Imagen propia

Figura 45: Procesos Misionales II



Fuente: Imagen propia

Figura 46: Procesos Misionales III

Fuente: Imagen propia

Figura 47: Procesos Misionales IV

Fuente: Imagen propia

Aprobar / Desaprobar Procesos Misionales

En esta sesión la directora docente aprueba y/o desaprueba los procesos misionales de uno o varios docentes.

Figura 48: Aprobar / Desaprobar Procesos Misionales

Fuente: Imagen propia

Reporte de Horarios

En esta sesión se generan los reportes de horarios ya sea por horario, asignatura, semestre o periodo académico.

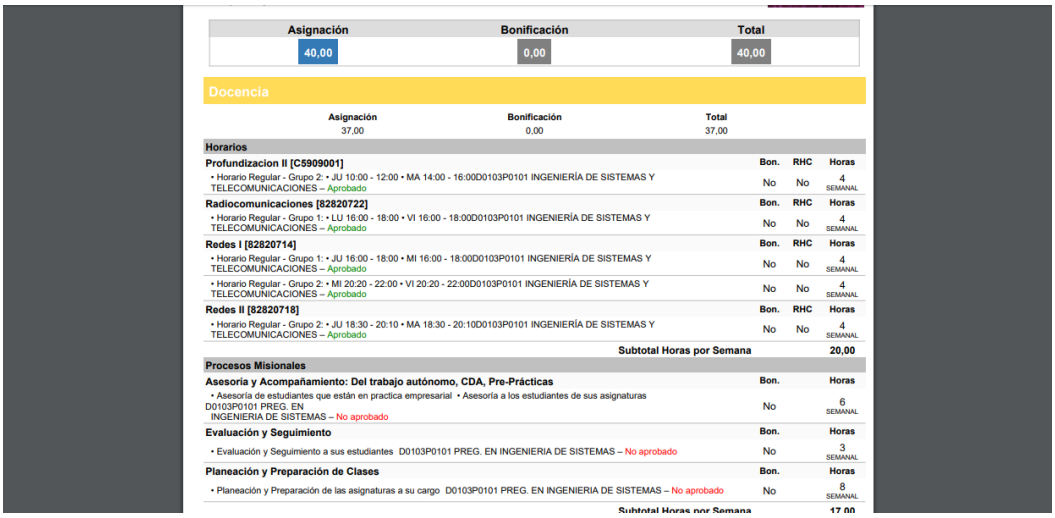
Figura 49: Reporte Horarios

Fuente: Imagen propia

Reporte de Procesos Misionales

En esta sesión se genera el reporte de procesos misionales de un docente ya sea por persona, programa o facultad.

Figura 50: Reporte Procesos Misionales



Fuente: Imagen propia

9.6 FASE 6: VALIDACIÓN

Luego de la definición e implementación del proceso de desarrollo para la Universidad de Manizales, se llevó a cabo su validación, para ello fue necesario el apoyo en la norma

ISO/IEC 29110, específicamente en el documento "Paquete de Despliegue Implementación del Software Perfil de Entrada" en su Actividades IS. Finalmente, se realizó un análisis comparativo de las experiencias vividas en el desarrollo del producto en sus dos versiones (Planeación Académica vs Gestión Académica).

9.6.1 Analizar Resultados

La validación del proceso de desarrollo de software se llevó a cabo teniendo en cuenta las siguientes actividades del **Proceso de Implementación de Software de** la Norma ISO 29110:

- Actividad: IS.1 Inicio de la Implementación del Software
- Actividad: IS2. Análisis de requisitos del software
- Actividad: IS.3 Identificación de los Componentes del Software
- Actividad: IS.4 Construcción del Software
- Actividad: IS.5 Pruebas e Integración del Software
- Actividad: IS.6 Entrega del Producto

Tabla 12: Evaluación Actividad IS.1

Actividad: IS.1 Inicio de la Implementación del Software		
Lista de Tareas	**Porcentaje Cumplimiento**	**Observación**
IS.1.1 Revisar el Plan del Proyecto Actual con los miembros del equipo para lograr un entendimiento común y obtener su participación con el proyecto.	*100%*	*El plan de Proyecto se construyó con el líder de proyecto, el dueño del producto y se socializó con el equipo de desarrollo con el fin de dar a conocer el alcance y demás ítems del plan.*
IS.1.2 Establecer o actualizar el entorno de implementación.	*100%*	*Se actualizó el entorno de implementación a Netbeans versión 8*

Fuente: Elaboración propia

Tabla 13: Evaluación Actividad IS.2

Actividad: IS2. Análisis de requisitos del software		
Lista de Tareas	**Porcentaje Cumplimiento**	**Observación**
IS.2.1 Asignar tareas a los miembros del equipo de trabajo de acuerdo con su rol, según el plan de proyecto actual.	*100%*	*Se creó un cronograma con asignación de actividades a los miembros del equipo y con tiempos estimados. Adicionalmente, se creó un tablero en trello con dichas actividades con el fin de hacerles seguimiento*
IS.2.2 Documentar o actualizar la *Especificación de Requisitos.*	*100%*	*Se creó el artefacto especificación de requerimientos*
IS.2.3 Validar y obtener la aprobación de la Especificación de Requisitos.	*63%*	*Se validaron los requerimientos con Registro, Dirección Docencia y algunos directores de programa, pero en vista de que es una cantidad elevada no se logró abarcarlos a todos. De aproximadamente 80 requerimientos se validaron y aprobaron 50.*

Fuente: Elaboración propia

Tabla 14: Evaluación Actividad IS.3

Actividad: IS.3 Identificación de los Componentes del Software		
Lista de Tareas	**Porcentaje Cumplimiento**	**Observación**
IS.3.1 Asignar tareas a los miembros del equipo de trabajo relacionadas con sus roles según al Plan de Proyecto actual.	*100%*	*Se creó un cronograma con asignación de actividades a los miembros del equipo y con tiempos estimados. Adicionalmente, se creó un tablero en trello con dichas actividades con el fin de hacerles seguimiento*
IS.3.2 Entender la especificación de Requisitos.	*100%*	*Se realizaron reuniones con diferentes interesados: Directores de Programa, Director Docente, Director de Registro, de estas reuniones se especificaron y entendieron los requerimientos del sistema.*
IS.3.3 Documentar o actualizar la identificación de componentes de Software.	100%	*Se creó el artefacto de arquitectura donde se describe entre otras los componentes del software. Adicionalmente, se crearon diagramas prototipos del sistema.*

Fuente: Elaboración propia

Tabla 15: Evaluación Actividad IS. 4

Actividad: IS.4 Construcción del Software		
Lista de Tareas	**Porcentaje Cumplimiento**	**Observación**
IS.4.1 Asignar las tareas a los miembros del equipo de trabajo relacionado con su rol, de acuerdo al *Plan de Proyecto* actual.	*100%*	*Se creó un cronograma con asignación de actividades a los miembros del equipo y con tiempos estimados. Adicionalmente, se creó un tablero en trello con dichas actividades con el fin de hacerles seguimiento*
IS.4.2 Comprender la Identificación de los *Componentes de Software*.	*100%*	*Se creó el artefacto de arquitectura donde se describe entre otras los componentes del software. Adicionalmente, se crearon prototipos del sistema.*
IS.4.3 Construir o actualizar los componentes de Software.	100%	*Se diseñó, codificó y verificó componentes de Software como fueron identificados*
IS.4.4 Establecer o actualizar los casos de prueba y los procedimientos de prueba para pruebas unitarias y de integración basado en la *Especificación de Requisitos* y la Identificación de los *Componentes de Software*	70%	*Las pruebas unitarias se realizaron a nivel de funcionalidades dentro del entorno de desarrollo pero estas no se documentaron, las pruebas de integración, validación ,verificación se especificaron y documentaron en el documento de casos de pruebas.*
IS.4.5 Probar los componentes de software. Corregir los defectos	100%	*Se probaron todos los componentes del sistema, se realizaron dos (2) ciclos de*

encontrados hasta que se logre una prueba unitaria exitosa.	*pruebas ya que en el primer ciclo salieron defectos que fueron corregidos y probados en el segundo ciclo.*

Fuente: Elaboración propia

Tabla 16: Evaluación Actividad IS.5

Actividad: IS.5 Pruebas e Integración del Software		
Lista de Tareas	**Porcentaje Cumplimiento**	**Observación**
IS.5.1 Asignar tareas a los miembros del equipo de trabajo relacionadas con sus roles de acuerdo al *Plan del Proyecto* actual.	*100%*	*Se creó un cronograma con asignación de actividades a los miembros del equipo y con tiempos estimados. Adicionalmente, se creó un tablero en trello con dichas actividades con el fin de hacerles seguimiento*
IS.5.2 Comprender los *Casos y Procedimientos de Prueba.*	*100%*	*Se crearon los casos de prueba de acuerdo con la especificación de requerimientos y casos de uso.*
Fijar o actualizar el entorno de prueba.	100%	*Se montó un ambiente de pruebas muy similar al de producción con el fin de realizar las pruebas*
SI.5.3 Integrar el Software utilizando los Componentes de Software y actualizando los Casos de Prueba y los Procedimientos de Prueba para integrar pruebas, como sea necesario.	60%	*Se integraron los componentes de software manualmente basados en casos de pruebas. Para que este proceso sea óptimo es necesario el uso de un herramienta de integración como Jenkis o gitlab.*

SI.5.4 Ejecutar las *Pruebas de Software* utilizando los *Casos de Prueba* y los *Procedimientos de Prueba* para su integración y documentación de resultados en el informe de pruebas.	100%	*Se ejecutaron las pruebas de software basados en los casos de pruebas previamente diseñados, dichas pruebas fueron debidamente documentadas.*
SI.5.5 Corregir los defectos encontrados hasta que se logren pruebas exitosas.	100%	*Se realizaron 2 ciclos de pruebas, esto con el fin de corregir los errores, probar nuevamente y lograr una prueba exitosa*
SI.5.6 Incorporar la Especificación de Requisitos y el Software con la Gestión de Configuración.	100%	*Se incorporó y actualizó el software para la gestión de la configuración: Git*

Fuente: Elaboración propia

Tabla 17: Evaluación Actividad IS.6

Actividad: IS.6 Entrega del Producto		
Lista de Tareas	**Porcentaje Cumplimiento**	**Observación**
IS.6.1 Asignar tareas a los miembros del equipo de trabajo relacionado a su rol, y de acuerdo con el *Plan del Proyecto* actual.	*100%*	*Se creó un cronograma con asignación de actividades a los miembros del equipo y con tiempos estimados. Adicionalmente, se creó un tablero en trello con dichas actividades con el fin de hacerles seguimiento*
IS.6.2 Revisar la claridad de la *Configuración del Software*.	0%	*No se definieron líneas base para la entrega del producto de software*

IS.6.3 Realizar la entrega al Gestor del Proyecto y apoyar la entrega de acuerdo con el *Plan del Proyecto*.	90%	*Se verificó que cada componente cumpliera con los criterios de aceptación, se realizaron manuales de usuario, capacitaciones y se realizó entrega formal del software, pero no se realizó un acta de entrega formal.*

Fuente: Elaboración propia

9.6.2 Comparar Resultados

Se realizó un comparativo entre el proyecto 1: **Planeación Académica** versus el proyecto 2: **Gestión Académica**.

Planeación Académica (P.A) es un proyecto desarrollado en el año 2011 por un equipo de desarrollo muy limitado y bajo ningún proceso de desarrollo. Las dificultades presentadas en el desarrollo, integración, puesta en producción y mantenimiento evidenciadas en casos de pruebas y experiencias de usuario.

Gestión Académica (G.A) fue desarrollado bajo los lineamientos del proceso de desarrollo propuesto y con un equipo de desarrollo que abarca los roles del proceso, adicionalmente, contó con la participación del desarrollador y único miembro del equipo de desarrollo del proyecto P.A. Gestión Académica se denomina la versión mejorada de Planeación Académica, el cual cuenta con las funcionalidades del proyecto 1 y algunas adicionales.

Los criterios seleccionados para comparar y validar los proyectos desarrollados son:

- Aspectos Generales
- Criterios de Calidad
- Aspectos propios del producto de software

Tabla 18: Comparativo Aplicativos P.A y G.A parte I

Comparativo Aplicativos Planeación Académica vs Gestión Académica			
Aspectos	**Planeación Académica P.A (2011)**	**Gestión Académica G.A (2020)**	**Observación**
Aspectos Generales			
Cantidad de Artefactos	0	8	
Cantidad de Roles	1	5	
Eventos	0	8	
Herramientas	1	6	
Cantidad de Iteraciones	0	3 por cada módulo	
Criterios De Calidad			
Fiabilidad	Baja	Alta	Este criterio de calidad fue medido basado el modelo de medición de la ISO 9241-1. Ver Anexo B
Eficiencia	Baja	Media - Alta	Este criterio de calidad fue medido basado el modelo de medición de la ISO 9241-1. Ver Anexo B
Mantenimiento	Media	Alta	Este criterio de calidad fue medido basado el modelo de medición de la ISO 9241-1. Ver Anexo B

Usabilidad	Baja	Media - Alta	Para este criterio de calidad se diseñó el instrumento de usabilidad basado en la ISO 9241-1. Ver Anexo C
Seguridad	Baja	Alta	Para este criterio de calidad se diseñó el instrumento de seguridad basado en la OWASP top 10. Ver Anexo D
Aspectos propios del desarrollo			
Cantidad de errores reportados en pruebas	Indeterminado	8	En P.A no hay un registro de errores reportados en pruebas, sin embargo, las historias de usuarios entrevistados y del desarrollador reportan alrededor de 30 fallas o errores reportados
Cantidad de errores reportados en producción	Indeterminado	3	En P.A no hay un registro de errores reportados en producción, sin embargo, las historias de usuarios entrevistados y del desarrollador reportan alrededor de 20 errores reportados
Cantidad de mejoras realizadas a partir de la puesta en producción	0	6	Entre ellas demoras en gestión de horarios, ineficiencia, múltiples errores, caídas, etc
Eliminación de fallos	70%	100%	
Tiempo de Desarrollo	5 meses	2 meses	

Experiencia de usuario	Baja	Alta	Se realizó una encuesta tanto a los desarrolladores del P.A y del G.A como a los usuarios finales que participaron en el uso y transición de ambos desarrollos (directores de programa)

Tabla 19: Comparativo Aplicativos P.A y G.A parte II

9.6.3 Lecciones Aprendidas Y Futuras Mejoras

Lecciones Aprendidas

Entre el equipo de desarrollo se realizó un análisis de todo el diseño, implementación y puesta en marcha del proceso de desarrollo, dicho análisis arrojó las siguientes lecciones aprendidas:

1. **Involucrar a los usuarios finales**

Involucrar a los usuarios finales es un punto clave ya que son ellos quien conocen la lógica del negocio y como debería operar el sistema, adicionalmente, se sienten involucrados y con un grado de responsabilidad importante ayudando a que el proyecto salga adelante. Por otro lado, en proyectos anteriores los desarrolladores debían tomar decisiones que no les correspondían, esto generaba descontento, reprocesos y más tiempo de desarrollo, con el proceso de desarrollo propuesto, se involucró más al usuario final lo cual generó muy buenos resultados.

2. **Herramienta Epf Composer**

La herramienta utilizada para el diseño del proceso de desarrollo permitió no solo diseñar el proceso en sí, sino que, además, estandarizar roles, procesos, artefactos y, en síntesis, fue la guía perfecta para el equipo en el desarrollo de cada una de las fases del ciclo de vida, permitiéndoles saber de antemano que pasos debían seguir, en qué momento, que documentos construir y en cuales apoyarse.

3. **Herramienta y capacitación al equipo de desarrollo**

Otro aspecto importante es asegurarse que todos los miembros del equipo y los interesados tengan acceso a las herramientas establecidas y correspondientes y cuenten con una adecuada capacitación para que todos puedan colaborar a lo largo del proyecto.

4. **Iteraciones**

Las iteraciones permitieron no solo comunicar a los usuarios finales el avance del proyecto, sino, además, nos permitió corregir aspectos a una etapa temprana, ahorrando tiempo y costos. La retroalimentación fue un aspecto muy importante y de gran valor en este proceso de desarrollo y a la hora de desarrollar el producto.

5. Gestión de Proyecto

Para la gestión de proyecto, el plan de proyecto fue de gran ayuda ya que permitió desde unas etapas iniciales definir y tener claro los objetivos del proyecto, permitió la definición de módulos y las fechas de entrega. Este objetivo de mejora tuvo un avance significativo ya que se logró la segregación de los módulos o funcionalidades en actividades más pequeñas, la estimación de tiempos de cada actividad basada en la experiencia por lo general del líder de proyecto y el desarrollador encargado de la actividad y la asignación de dicha tarea a uno o más miembros del equipo. En el cronograma se plasmó toda la estimación y aunque algunas actividades se terminaron después de lo pactado, permitió tanto al líder de proyecto y el dueño del producto tener un mayor control sobre el proyecto y estar informados de sus actividades.

6. Objetivos de mejora

Con relación a los objetivos de mejora planteados, se pudo evidenciar:

- Se redujo alrededor de un 60% los errores reportados después de la puesta en producción de, esto estimando que la cantidad de errores aproximados de P.A fueron 20 (pueden ser más), cumpliendo así el objetivo de mejora.
- En cuanto la estimación de recursos dentro del proyecto, se evidenció un gran avance ya que anteriormente no se realizaban estimaciones formales y se pasó de tener algunas aproximaciones a tener plan de proyecto, cronograma con asignación de recursos, tiempos, actividades y contar con herramientas para el seguimiento y control de dichas actividades.
- En relación a la estimación de tiempos, se logró proyectar la fecha inicio y fin de cada módulo, aunque no siempre fue acertada, lo que evidenció la necesidad de utilizar alguna técnica de estimación de esfuerzo y tiempo que permita estimar no solo basada en la experiencia si no tener en cuenta otros aspectos.
- La credibilidad del área de T.I en la Universidad de Manizales viene en aumento, no solo por el buen manejo de este proyecto y su gestión, si no, además, en la incorporación e involucramiento de los interesados, teniendo en cuenta sus puntos de vistas, opiniones y recomendaciones. Este aspecto se debe seguir trabajando pues son

muchas las necesidades de la comunidad académica, estudiantil y administrativa que se pueden atender y sistematizar

- La integración de los componentes presenta varias falencias en el proceso de desarrollo, puesto que las integraciones realizadas por el equipo de trabajo se realizan en su mayoría de forma manual, dado que la herramienta utilizada para esta no tiene como función principal la de integrar, si no que cumple la labor de gestión de repositorio, lo que entorpeció el proceso de integración.

Futuras Mejoras

A continuación, se listan algunas futras mejoras que dejó toda la construcción e implementación del proceso de desarrollo propuesto:

1. Capacitaciones al equipo

Con la implementación del proceso de desarrollo, se comenzaron a utilizar herramientas que ayudaron durante el ciclo de vida de un proyecto, esto a pesar de que fue una buena práctica, generó un poco de descontento entre los miembros del equipo ya que no conocían dichas herramientas, por ellos, se recomienda capacitación a los miembros del equipo en estos aspectos y en otros que se consideren relevantes para el buen desenvolvimiento del proyecto.

2. Gestión de Proyectos

A pesar de que la gestión de proyectos se abarcó dentro del proceso de desarrollo, es necesario enfatizar más en este aspecto, que permita no solo administrar, planificar, coordinar, hacer seguimiento y control de todas las actividades, sino que también, permita gestionar recursos, establecer tiempos más precisos y presupuestar costos, para esto es conveniente utilizar alguna técnica de estimación más acertada ya sea juicio de expertos, estimación análoga, métodos PERT, etc.

3. Integración

El manejo dado a la integración debe pasar de ser un proceso dispendioso o casi manual para el equipo de desarrollo del área de tecnologías, y debe ser abarcado e implementado en base a herramientas que estén enfocadas en esta labor (integrar), dado que el uso de tecnologías no adecuadas en el proceso de integración siempre será un traspié para el

debido proceso de los desarrollos; es por esto que se recomienda la implementación de herramientas tales como Jenkins o GitLab que sean y sirvan apoyo a todos los procesos del área y en específico a los orientados al desarrollo.

9.6.4 Seleccionar Un Proyecto Piloto

Seleccionar un proyecto en el cual se va a implementar el proceso de desarrollo.
Para la implementación del proceso de desarrollo propuesto se seleccionó el proyecto denominado “Gestión Académica” en los siguientes módulos:

- Gestión de Horarios (regulares e irregulares)
- Aprobar / Desaprobar Horarios
- Gestión de Procesos Misionales
- Reportes

El proyecto fue elegido debido a que ya se tenía una versión inicial la cual se había desarrollado hace algunos años atrás y de la cual se tienen alguna documentación de casos de pruebas y de problemas reportados por los usuarios. Adicionalmente, este proyecto cuenta con un equipo de desarrollo muy similar al actual. Finalmente, el tamaño del proyecto y de sus módulos permiten una adecuada asignación de roles, artefactos, y una gestión de iteraciones y eventos.

Seleccionar el equipo de desarrollo

Para iniciar, se reunió al equipo de desarrollo, se socializó el alcance y los artefactos que hasta el momento se tenía del proyecto tales como el Documento Plan de Proyecto, Documento Análisis de Riesgos y el Documento Especificación de Requerimientos, finalmente, se socializó el proceso de desarrollo.

El equipo de desarrollo seleccionado es similar al que participó en el desarrollo de la primera versión de **Gestión Académica:**

- Dueño del producto: Juan Carlos Cardona
- Líder de Proyecto: Diana Carolina Gómez
- Analista: Catalina Herrera - Diana Carolina Gómez
- Desarrolladores – Sebastián Vásquez – Yeison Osorio
- DBA: Julián Acevedo
- Tester – Equipo de Desarrollo (Fredy Román, Catalina Herrera, Diana Carolina Gómez)

Definir roles dentro del equipo

Inicialmente el proyecto comenzó con los roles: Dueño del producto, líder de proyecto, un analista, un desarrollador y el DBA. Cuando el proyecto ya cumplía alrededor del 40% de avance, se adicionó una segunda analista, un segundo desarrollador y un tester de calidad.

10 CONCLUSIONES

El diseño e implementación de un proceso de desarrollo de software y adopción de buenas prácticas requiere de un compromiso tanto de la alta dirección como del personal del área de T.I, puesto que sin éstos no existe el sistema ni la mejora a los procesos.

La implementación del proceso de desarrollo propuesto en el proyecto de software Gestión Académica redujo significativamente la cantidad de defectos encontrados antes y después de la puesta en producción y con esto la reducción de inconformidades percibidas por el cliente.

El uso de una herramienta para el modelamiento del proceso de desarrollo, en este caso, EPF Composer, fue de gran ayuda ya que permitió formalizar el proceso diseñado, y así mismo, se convirtió en una guía para el equipo en el desarrollo del proyecto de software y su debida documentación.

Las iteraciones permitieron no solo comunicar a los usuarios finales el avance del proyecto, sino, además, permitió corregir aspectos a una etapa temprana, ahorrando tiempo y costos.

Las retroalimentaciones en cada una de las iteraciones en acompañamiento con el cliente final redujeron los tiempos que se destinan a realizar ajustes y fueron de gran valor en este proceso de desarrollo.

Sensibilizar, motivar y capacitar al personal del área en cuanto al proceso de desarrollo de software se hizo necesario para cumplir con los objetivos propuestos, adicionalmente, involucrarlos en las propuestas de desarrollo y etapas iniciales de los proyectos permitió que el equipo permaneciera motivado, realizando aportes significativos no solo a la hora de estimar tiempos sino en todas las etapas del proceso.

Involucrar a los interesados no solo en las primeras etapas del desarrollo de software sino a lo largo de todo proceso permitió aumentar la comunicación del proyecto, reduciendo así reprocesos y errores generados por realizar funcionalidades basadas en los criterios del equipo y no del usuario final.

11 REFERENCIAS BIBLIOGRÁFICAS

González, F. (2015). El área de TI como generador de valor en el negocio. 2018, de KPMG International Cooperative Sitio web: https://assets.kpmg/content/dam/kpmg/pa/pdf/delineandoestrategias/DE-area-TI-como-generador-de-valor-negocio.pdf

Britto, J (2014). Adaptación de un proceso de desarrollo de software basado en buenas prácticas (Tesis de Grado Maestría Gestión y desarrollo de proyectos de software). Universidad Autónoma de Manizales. Manizales.

Pélaez, L., Toro, A., López, J. & Ramirez, A. (2012). Caracterización del proceso de desarrollo de software en Colombia, una mirada desde las PYMES productoras. Revista académica e institucional de la UCPR. *Páginas (92)*, p. 89-98.

Blanco, M. (2007). Propuesta estratégica para el mejoramiento de procesos en organizaciones pequeñas de desarrollo de software: caso qualdev group (Tesis de Grado Maestría en Ingeniería Industrial). Universidad de los Andes.

Ramirez, L. & Florez, A. (2014). Buenas prácticas, una solución para un mejor desarrollo de software. Revista Mundo Fesc. *Vol 4 (8)*, p. 37-45.

(2001). Ideal model. Recuperado de https://www.sei.cmu.edu. Carnegie Mellon University.

Morales, J. & Pardo, C. (2016). Revisión sistemática de la integración de modelos de desarrollo de software dirigido por modelos y metodologías ágiles. *Vol (80)*, p. 87–99.

Pesado, P., Bertone, R. A., Esponda, S., Pasini, A. C., Boracchia, M., Martorelli, S., & Swaels, M. (2013). Mejora de procesos en el desarrollo de sistemas de software y en procesos de gestión. Experiencias En PyMEs, *Vol (1)*, p. 581–585.

Elizabeth, M. (2017). Propuesta de modelo de mejora para mypes productoras de software. Revista Interfaces. Vol (12), p. 57–74.

Chavarría, A. E., Oré, S. B., & Pastor, C. (2016). Aseguramiento de la Calidad en el Proceso de Desarrollo de Software utilizando CMMI, TSP y PSP, 62–77.

Yepes, J., Pardo, C. & Gómez, O. (2015). Revisión sistemática acerca de la implementación de metodologías ágiles y otros modelos en micro, pequeñas y medianas empresas de software. Revista Tecnológica ESPOL – RTE. Vol (*28)*, p. 464–479.

Piattini, M. Jadwiga, H., Orozco, M., & Alquicira, C. (2008). *Competisoft. Mejora de Procesos Software para Pequeñas y Medianas Empresas y Proyectos*. España. RA-MA S.A. Editorial y Publicaciones.

Pardo, C., Hurtado, J. A., & Collazos, C. A. (2010). Mejora de Procesos de software ágil con Agile-SPI Process. *Revista DYNA*, *77*, 251–263. Retrieved from http://www.researchgate.net/publication/265088413_Mejora_de_Procesos_de_software_gil_con_Agile-SPI_Process.

Ken Schwaber, J. S. (2013). La Guía de Scrum. Retrieved from https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-es.pdf

Frechina, A. (2018). Metodología Scrum ¿Qué es? Retrieved from https://winred.es/management/metodologia-scrum-que-es/gmx-niv116-con24594.htm

Philippe Kruchten, Per Kroll. (2003). The rational unified process made easy: a practitioner's guide to the RUP. Boston, MA, USA: Addison-Wesley Longman Publishing Co.

Anwar, A. (2014). A Review of RUP (Rational Unified Process). Ashraf Anwar International Journal of Software Engineering (IJSE), (5), 8. Retrieved from https://www.cscjournals.org/manuscript/Journals/IJSE/Volume5/Issue2/IJSE-142.pdf

Eclipse Process Framework. (2004 - 2007). Introduction to OpenUP. Recuperado de http://www.utm.mx/~caff/doc/OpenUPWeb/index.htm

Ríos, S., Hinojosa, C., & Delgado, R. (2013). Aplicación De La Metodología Openup En El Desarrollo Del Sistema De Difusión De Gestión Del Conocimiento De La Espe. Escuela Politécnica Del Ejército, Ecuador., 10. Retrieved from http://repositorio.espe.edu.ec/bitstream/21000/6316/1/AC-SISTEMAS-ESPE-047042.pdf

ISO 29110-5-1. (2014). ISO/IEC TR 29110-5-1-2:2014 Software Engineering-Lifecycle Profiles for Ver Small Entities (VSEs)-Part 5-1-2: Management and Engineering Guide: Generic Profile Group: Basic Profile. Int'l Org for Standardization.

ISO 29110-5-2. (2014). ISO/IEC TR 29110-5-1-2:2014 Software Engineering-Lifecycle Profiles for Ver Small Entities (VSEs)-Part 5-1-2: Management and Engineering Guide: Generic Profile Group: Basic Profile. Int'l Org for Standardization.

Hernández, G., González. (2017). Paquete de Despliegue Implementación del Software Perfil de Entrada. http://www.etsmtl.ca/Professeurs/claporte/Publications.

12 ANEXOS

Anexo 1: Comparativo Procesos

Procesos de Desarrollo	**SCRUM**	**OPEN UP**	**RUP**	**XP**
Características	Enfocado en la satisfacción del cliente a través de la reducción de la complejidad en el desarrollo de productos de software. Está basado en a la auto-organización, esto con el fin de resolver problemas complejos y adaptarse a lo imprevisible.	Enfoque ágil, iterativo e incremental. Las iteraciones se enfocan en una entrega de valor por parte del equipo. Los equipos de OpenUP se auto organizan para lograr los objetivos de cada iteración y se comprometen a entregar los resultados	Es una metodología de desarrollo de software. - Proceso centrado en la arquitectura. - Iterativo e incremental - Dirigido por casos de uso	Basado en: - Trabajo en equipo - Satisfacción del cliente - Actuar sobre variables como: tiempo, alcance, costo y calidad - Entregas pequeñas - Diseño simple - Refactorización - Programación en parejas - Integración continua
Fases	**Eventos:** - Sprint - Scrumm Diario (Daily Scrums)	**4 Fases:** - Inicio - Elaboración - Construcción - Transición	**4 Fases:** - Inicio - Elaboración - Construcción - Transición	**6 Fases:** - Exploración - Planificación de la entrega (Release) - Iteraciones - Producción - Mantenimiento - Muerte del Proyecto

Roles	**Roles:** - El Dueño de Producto (Product Owner) - El Equipo de Desarrollo (Development Team) - Scrum Master	**Roles:** - Arquitecto - Gerente de Proyecto - Analista - Tester - Desarrollador	**Roles:** - Analistas - Desarrolladores - Probadores - Gerentes - Otros	**Roles:** - Programador - Cliente - Encargado de pruebas (Tester) - Encargado de seguimiento (Tracker) - Entrenador (Coach) - Consultor - Gestor (Big boss)

Artefactos	**Artefactos:** - Lista de Producto (Product Backlog) - Lista de Pendientes del Sprint (Spring Backlog) - Historias de Usuarios - Plan de Proyecto	**Artefactos:** - Notas de Arquitecturas - Pruebas de Desarrollo - Plan de Iteración - Plan de Proyecto - Lista de Riesgos - Lista de elementos de trabajo - Visión - Casos de Uso - Glosario - Modelo de Casos de Uso - Casos de Pruebas - Log de Pruebas - Script de Pruebas	**Artefactos principales:** - Plan de desarrollo - Caso de Negocio - Lista de Riesgos - Modelo de Casos de Uso - Visión - Especificación Adicional - Glosario - Modelo de diseño - Documentación de la arquitectura de sw - Modelo de la implementación - Plan de test - Plan de despliegue	**Artefactos:** - Historias de usuarios - Tareas de ingeniería - Tarjetas CRC - Plan de entregas - Plan de iteraciones - Metáfora - Codificación estándar - Unidad de prueba - Código de producción - Visión .

Ventajas	**Ventajas** - Es un proces liviano - Flexibilidad y adaptación a cambios - Gestión de las expectativas del usuario - Alineamiento entre el cliente y el equipo de desarrollo. - Los problemas se identifican con suficiente antelación a través de las reuniones diarias y por lo tanto se pueden resolver con rapidez	**Ventajas** - Promueve la retroalimentación temprana y continua de los stakeholders. - Permite detectar errores tempranos a través de un ciclo iterativo. - Evita la elaboración de documentación, diagramas e iteraciones innecesarios.	**Ventajas** - Existe un involucramiento continuo de las partes interesadas. - Verificar la calidad del software en todo el tiempo de vida de desarrollo. - Fácil mantenimiento y modificaciones locales. - Reutilización de roles y otros aspectos. - Mitigación temprana de posibles riesgos	**Ventajas** - Es un proceso ligero - Estimula la comunicación entre el cliente y los programadores. - El cliente tiene el control sobre las prioridades - La mayoría de los errores se descubren en el momento en que se codifican, ya que el código es permanentemente revisado por dos personas. - El equipo resuelve problemas en forma más rápida

Desventajas	**Desventajas** - Dificultad de aplicación en grandes proyectos - Funciona bien en equipos pequeños. - Requiere una exhaustiva y detallada definición de las tareas y sus plazos - Exige que quienes lo utilicen tengan una alta formación. - Demasiadas reuniones pueden resultar cansador y estresante, algunos van perdiendo el interés en el proyecto	**Desventajas** - El proyecto pueda perder su rumbo debido a la desorganización y que es una metodología de bajo formalismo.	**Desventajas** - Es un proceso pesado - Es más apropiada para proyectos grandes - Requiere conocimiento previo sobre UML.	**Desventajas** - Dificultades a la hora de costear un proyecto. - Poca o mínima documentación - No es recomendable para proyectos a largo plazo. - En equipos pequeños, la programación en parejas puede suponer un problema.
Integraciones Permanentes	-	Integración continua	Utiliza técnicas de integración continua	Integración continua, establece que cada tarea que se completa se integra al sistema, un proceso que puede darse, incluso, varias veces al día
Plan de Entregas	Entregas incrementales	Diseño evolutivo	Diseño evolutivo	-

Plan de Iteraciones	Sprint Scrum diario	Desarrollo iterativo Ciclo de vida orientado por riesgo-valor Arquitectura evolutiva D	Los proyectos se entregan, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto	Iteraciones pequeñas y tempranas
Documentación	Mínima	Moderada	Máxima	Mínima
Desarrollo Sw en cualquier tecnología	Si	Si	Si	Si
Desarrollo incremental	Desarrollo de incremental en iteraciones que van de 2 a 4 semanas máximo	Desarrollo evolutivo para obtener retroalimentación y mejoramiento continuo	Caracterizado por ser iterativo e incremental	Entregas Pequeñas: ciclos cortos de desarrollo (iteraciones) que le muestran software terminado al cliente y obtienen retroalimentación de él
Adaptable cambios	Si	SI	Si, pero al ser un proceso pesado puede costar más el cambio	No

Tamaño del equipo o proyecto	Recomienda un tamaño de entre 3 y 9	Puede emplearse con equipos de proyectos de cualquier tamaño.	Puede emplearse con equipos de proyectos de cualquier tamaño.	Equipos de desarrollo de software pequeños o medianos, entre 2 y 10 desarrolladores

Anexo 2: Métricas de Calidad

1. Fiabilidad

Planeación Académica (P.A)	
Nombre	**Detección de Fallos**
Propósito	¿Cuántas fallas se detectaron en el producto revisado?
Fórmula	X = A / B A = Número absoluto de fallas detectadas en revisión B = Número de fallas estimadas que se detectarían en la revisión
Aplicación Formula	A: 30 B: 10 X = 30/10, X = -3
Interpretación	0 <= X .Un valor alto para X implica una buena calidad del producto

Gestión Académica (G.A)	
Nombre	**Detección de Fallos**
Propósito	¿Cuántas fallas se detectaron en el producto revisado?
Fórmula	X = A / B A = Número absoluto de fallas detectadas en revisión B = Número de fallas estimadas que se detectarían en la revisión
Aplicación Formula	A: 8 B: 15 X = 8/15, X = 0,5
Interpretación	0 <= X .Un valor alto para X implica una buena calidad del producto

Resultado	Poca calidad del producto

Resultado	Buena calidad del producto

Nombre	**Remoción de Fallos**
Propósito	¿Cuántas fallas se han corregido?
Fórmula	Y = A / B A = Número de fallas corregidas en diseño / codificación. B = Número de fallas detectadas en revisión.
Aplicación Formula	A: 30 B: 30 Y = 27/30, X = 0,9
Interpretación	0 <= Y <= 1 Cuanto más cerca de 1, mejor (más fallas eliminadas)
Resultado	Faltaron fallas por corregir

1

Nombre	**Remoción de Fallos**
Propósito	¿Cuántas fallas se han corregido?
Fórmula	Y = A / B A = Número de fallas corregidas en diseño / codificación. B = Número de fallas detectadas en revisión.
Aplicación Formula	A: 8 B: 8 Y = 8/8, X = 1
Interpretación	0 <= Y <= 1 Cuanto más cerca de 1, mejor (más fallas eliminadas)
Resultado	Todas las fallas corregidas

2. Eficiencia

Planeación Académica (P.A)	
Nombre	**Tiempo de Respuesta**

Gestión Académica (G.A)	
Nombre	**Tiempo de Respuesta**

Propósito	¿Cuál es el tiempo estimado para completar una tarea específica?
Fórmula o medición	X = tiempo (calculado o simulado)
Aplicación	270 segundos
Interpretación	Entre mas corto mejor
Resultado	Baja

Propósito	¿Cuál es el tiempo estimado para completar una tarea específica?
Fórmula o medición	X = tiempo (calculado o simulado)
Aplicación	161
Interpretación	Entre mas corto mejor
Resultado	Media - Alta

3. Mantenibilidad

Planeación Académica (P.A)	
Nombre	**Cumplimiento de la capacidad de mantenimiento**
Propósito	Cuán compatible es la capacidad de mantenimiento del producto con las regulaciones, estándares y convenciones aplicables.
Fórmula	X = A / B A = Número de elementos implementados correctamente relacionados con el cumplimiento de la capacidad de

Gestión Académica (G.A)	
Nombre	**Cumplimiento de la capacidad de mantenimiento**
Propósito	Cuán compatible es la capacidad de mantenimiento del producto con las regulaciones, estándares y convenciones aplicables.
Fórmula	X = A / B A = Número de elementos implementados correctamente relacionados con el cumplimiento

	mantenimiento confirmados en la evaluación B = Número total de elementos de cumplimiento
Aplicación Formula	A: 12 B: 10 X = 9/12, X =0.75
Interpretación	0 <= X <= 1 Cuanto más cerca de 1, más compatible
Resultado	Compatible Medio

	de la capacidad de mantenimiento confirmados en la evaluación B = Número total de elementos de cumplimiento
Aplicación Formula	A: 12 B: 12 X = 12/12, X =1
Interpretación	0 <= X <= 1 Cuanto más cerca de 1, más compatible
Resultado	Compatible Alto

3. Usabilidad

Planeación Académica (P.A)	
Nombre	**Usabilidad**
Propósito	Cuán compatible es la capacidad de mantenimiento del producto con las regulaciones, estándares y convenciones aplicables.

Gestión Académica (G.A)	
Nombre	**Usabilidad**
Propósito	Cuán compatible es la capacidad de mantenimiento del producto con las regulaciones, estándares y convenciones aplicables.

Fórmula	X = A / B A = Número de elementos implementados correctamente relacionados con el cumplimiento de la capacidad de mantenimiento confirmados en la evaluación B = Número total de elementos de cumplimiento
Aplicación Formula	A: 12 B: 10 X = 9/12, X =0.75

Fórmula	X = A / B A = Número de elementos implementados correctamente relacionados con el cumplimiento de la capacidad de mantenimiento confirmados en la evaluación B = Número total de elementos de cumplimiento
Aplicación Formula	A: 12 B: 12 X = 12/12, X =1

MEDIDA FIABILIDAD

Fiabilidad	**Deteccion Fallos**	**Remoción fallos**
Alta	0 <= X	0 <= Y <= 1
Baja	0 > X	0 > Y > 1

Planeación Académica (P.A)

Fiabilidad	**Detección Fallos**	**Remoción fallos**
Alta		
Baja	-3	0,9

Gestión Académica (G.A)

Fiabilidad	Detección Fallos	Remoción fallos
Alta	0,5	1
Baja		

MEDIDA DE EFICIENCIA

Tiempo de respuesta

Funcionalidad	Tiempo de respuesta en min

* Los tiempos fueron tomados con ejercicios reales en ambas aplicaciones y en tiempo real

Planeación Académica (P.A)

Funcionalidad	Tiempo de respuesta en seg
1. Crear grupo	15
2. Crear horario regular	35
3. Asignar docente a horario	15
4. Crear horario irregular	40
5. Aprobar horario grupo	20
6. Desaprobar horario grupo	20
7. Editar horario	35

Gestión Académica (G.A)

Funcionalidad	Tiempo de respuesta en seg
1. Crear grupo	5
2. Crear horario regular	20
3. Asignar docente a horario	20
4. Crear horario irregular	10
5. Aprobar horario grupo	8
6. Desaprobar horario grupo	8
7. Editar horario	20

8. Eliminar horario	20
9. Crear proceso misional	25
10. Editar proceso misional	15
11. Eliminar proceso misional	10
12. Generar reporte de horarios	20
	270

8. Eliminar horario	15
9. Crear proceso misional	10
10. Editar proceso misional	10
11. Eliminar proceso misional	15
12. Generar reporte de horarios	20
	161

MEDIDA MANTENIMIENTO

Mantenimiento	**X**
Alta	1 <= X > 0.8
Medio	0.8 <= X > 0.4
Bajo	0.4 <= X > 0

Planeación Académica (P.A)

Fiabilidad	**Detección Fallos**
Alta	
Medio	0.75
Baja	

Gestión Académica (G.A)

Fiabilidad	**Detección Fallos**

Alta	1
Medio	
Baja	

Anexo 3: Instrumento de Usabilidad

Instrumento Usabilidad

		Opciones de Respuestas		
#	Preguntas	En desacuerdo	Ni de acuerdo ni en desacuerdo	En acuerdo
	Aprendibilidad			
1	El sistema es fácil de usar			
2	El sistema es simple de usar			
3	El sistema es amigable con el usuario.			
4	El sistema requiere el menor número de pasos para lograr lo que quiero			
5	El sistema puede ser usado o sin instrucciones escritas			
6	En el sistema puede corregir los errores rápida y fácilmente			
	Comprensibilidad			
7	He aprendido a utilizar rápidamente el sistema de			
8	Recuerdo fácilmente cómo usar el sistema de			
9	Es fácil aprender a usar el sistema de			
	Satisfacción del Sistema			
10	Estoy satisfecho con el sistema con el sistema de			
11	Es agradable de usar			
	Conformidad de Uso			
12	El sistema me ayuda a ser más eficaz			
13	El sistema me ayuda a ser más productivo			
14	El sistema me da un mayor control sobre las actividades que realizo			
15	El sistema me ahorra tiempo cuando lo uso			

Personas encuestadas	Cantidad
Directores de programa pregrado	6
Directores de programa posgrado	3
Director docente	1
Asistente dirección docencia	2

	Σ
Formula	ValorOr(n)*Cantidad

RESPUESTAS

PLANEACIÓN ACADÉMICA		Opciones de Respuestas		
#	Preguntas	Totalmente en desacuerdo	Ni de acuerdo ni en desacuerdo	Totalmente de acuerdo
	Aprendibilidad	12	24	36
1	El sistema es fácil de usar	7	5	
2	El sistema es simple de usar	10	2	
3	El sistema es amigable con el usuario.	10	2	
4	El sistema requiere el menor número de pasos para lograr lo que quiero	6	6	
5	El sistema puede ser usado sin instrucciones escritas	3	9	
6	En el sistema puede corregir los errores rápida y fácilmente	12		
	Comprensibilidad			
7	He aprendido a utilizar rápidamente el sistema	12		
8	Recuerdo fácilmente cómo usar el sistema	10	2	
9	Es fácil aprender a usar el sistema	10	2	
	Satisfacción del Sistema			
10	Estoy satisfecho con el sistema con el sistema	12		
11	Es agradable de usar	12		
	Conformidad de Uso			
12	El sistema me ayuda a ser más eficaz	8	4	
13	El sistema me ayuda a ser más productivo	7	5	
14	El sistema me da un mayor control sobre las actividades que realizo	4	8	
15	El sistema me ahorra tiempo cuando lo uso	7	5	

GESTIÓN ACADÉMICA	Opciones de Respuestas

#	Preguntas	Totalmente en desacuerdo	Ni de acuerdo ni en desacuerdo	Totalmente de acuerdo
	Aprendibilidad			
1	El sistema es fácil de usar		2	10
2	El sistema es simple de usar	1	3	8
3	El sistema es amigable con el usuario.		2	10
4	El sistema requiere el menor número de pasos para lograr lo que quiero	2	6	4
5	El sistema puede ser usado sin instrucciones escritas	6	6	
6	En el sistema puede corregir los errores rápida y fácilmente		8	4
	Comprensibilidad			
7	He aprendido a utilizar rápidamente el sistema		4	8
8	Recuerdo fácilmente cómo usar el sistema		2	10
9	Es fácil aprender a usar el sistema		4	8
	Satisfacción del Sistema			
10	Estoy satisfecho con el sistema con el sistema	1	7	4
11	Es agradable de usar		6	6
	Conformidad de Uso			
12	El sistema me ayuda a ser más eficaz	1	5	6
13	El sistema me ayuda a ser más productivo	2	7	3
14	El sistema me da un mayor control sobre las actividades que realizo	4	8	
15	El sistema me ahorra tiempo cuando lo uso	4	6	2

Anexo 4: Lista de Chequeo Seguridad

	PLANEACIÓN ACADÉMICA				
#	**Descripción**	**Si**	**No**	**NA**	**Observaciones**
	Validaciones de Entrada				
1	¿La aplicación valida las entradas tanto del lado del cliente como del lado del servidor?	X			
2	¿La aplicación valida SQL Injection y otros tipos de ataques de entradas?	X			
3	¿Todas las salidas visibles están basadas en los roles de usuarios, según se definen en la aplicación?		X		
	Autenticación y autorización				
4	¿Los roles y cuentas de usuario utilizadas en la aplicación han sido configuradas considerando el principio de mínimo privilegio?		X		
5	¿La aplicación restringe el número de intentos fallidos de acceso?		X		
6	¿Existe una política de contraseñas estricta o un mecanismo de autenticación construido dentro de la aplicación?		X		
7	¿Los tokens relacionados con autenticación, tales como cookies, se transmiten a través de conexiones seguras?		X		
8	¿La información de autenticación está cifrada?		X		
9	¿Existe información de autenticación codificada (hard-coded) dentro de la aplicación?		X		
	Almacenamiento seguro				
10	¿Cuáles son los algoritmos de criptografía utilizados por la aplicación para requerimientos de cifrado y hashing?		X		Se utiliza cifrado Sha512

11	¿Los datos sensibles se cifran o enmascaran para protegerlos?	X			
12	¿Se protegen las copias temporales de datos sensibles contra el acceso no autorizado, y se programa su remoción cuando ya no van a ser utilizados?	X			
	Comunicación segura				
13	¿La aplicación utiliza módulos de criptografía, los cuales refuerzan el uso de algoritmos de seguridad?		X		Se utiliza cifrado Sha512
14	¿La aplicación identifica mecanismos de protección para datos sensibles que son enviados sobre la red (interna y externa)?	X			
	Manejo de errores				
15	¿Existe un enfoque estándar apropiado para estructurar el manejo de excepciones en la aplicación?		X		El aplicativo maneja excepciones
16	¿La aplicación identifica el nivel de auditoría y almacenamiento (logging) necesario y los parámetros clave para ser almacenados y auditados?		X		Se lleva a cabo una auditoría a nivel de B.D
17	¿Se asegura que los recursos sean liberados si ocurre un error?			X	
	Administración de sesiones				
18	¿La aplicación genera un nuevo identificador de sesión (session ID) cuando un usuario requiere ser reautenticado en pantallas que involucran acciones sensibles?	X			
19	¿Los valores de sesión expiran automáticamente después de un lapso predefinido?		X		Las sesiones no caducan

20	Utiliza funcionalidades de log-out y de finalización de las sesiones de usuarios.	X			

	GESTIÓN ACADÉMICA				
#	**Descripción**	**Si**	**No**	**NA**	**Observaciones**
	Validaciones de Entrada				
1	¿La aplicación valida las entradas tanto del lado del cliente como del lado del servidor?	X			
2	¿La aplicación valida SQL Injection y otros tipos de ataques de entradas?	X			
3	¿Todas las salidas visibles están basadas en los roles de usuarios, según se definen en la aplicación?	X			
	Autenticación y autorización				
4	¿Los roles y cuentas de usuario utilizadas en la aplicación han sido configuradas considerando el principio de mínimo privilegio?	X			
5	¿La aplicación restringe el número de intentos fallidos de acceso?	X			Esto se hace desde sigum
6	¿Existe una política de contraseñas estricta o un mecanismo de autenticación construido dentro de la aplicación?	X			Esto se hace desde sigum
7	¿Los tokens relacionados con autenticación, tales como cookies, se transmiten a través de conexiones seguras?		X		
8	¿La información de autenticación está cifrada?	X			Esto se hace desde sigum
9	¿Existe información de autenticación codificada (hard-coded) dentro de la aplicación?		X		
	Almacenamiento seguro				

10	¿Cuáles son los algoritmos de criptografía utilizados por la aplicación para requerimientos de cifrado y hashing?	X			Se utiliza cifrado Sha512
11	¿Los datos sensibles se cifran o enmascaran para protegerlos?	X			
12	¿Se protegen las copias temporales de datos sensibles contra el acceso no autorizado, y se programa su remoción cuando ya no van a ser utilizados?	X			
	Comunicación segura				
13	¿La aplicación utiliza módulos de criptografía, los cuales refuerzan el uso de algoritmos de seguridad?	X			Se utiliza cifrado Sha512
14	¿La aplicación identifica mecanismos de protección para datos sensibles que son enviados sobre la red (interna y externa)?	X			
	Manejo de errores				
15	¿Existe un enfoque estándar apropiado para estructurar el manejo de excepciones en la aplicación?	X			El aplicativo maneja excepciones
16	¿La aplicación identifica el nivel de auditoría y almacenamiento (logging) necesario y los parámetros clave para ser almacenados y auditados?	X			Se lleva a cabo una auditoría a nivel de B.D
17	¿Se asegura que los recursos sean liberados si ocurre un error?			X	
	Administración de sesiones				
18	¿La aplicación genera un nuevo identificador de sesión (session ID) cuando un usuario requiere ser	X			

	reautenticado en pantallas que involucran acciones sensibles?				
19	¿Los valores de sesión expiran automáticamente después de un lapso predefinido?	X			Esto ocurre 20 min después de inactividad del usuario
20	Utiliza funcionalidades de log-out y de finalización de las sesiones de usuarios.	X			