PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR-MATRIZ

FACULTAD DE CIENCIAS ADMINISTRATIVAS Y CONTABLES

TRABAJO DE TITULACIÓN DE MAGÍSTER EN ADMINISTRACIÓN DE EMPRESAS CON MENCIÓN EN GERENCIA DE LA CALIDAD Y PRODUCTIVIDAD

DISEÑO DE UN SISTEMA DE GESTIÓN DE LA CALIDAD PARA EL DEPARTAMENTO DE DESARROLLO DE SOFTWARE DE LA PUCE

ING. ALEJANDRO PAÚL ALDÁS ALARCÓN

DIRECTOR: LCDO. FREDDY ARÉVALO CHÁVEZ, MBA.

QUITO, 2016

DIRECTOR:

Lcdo. Freddy Arévalo Chávez, MBA.

INFORMANTES:

Ing. Jaime Cadena Echeverría, Mgtr.

Ing. Juan Carlos Piñuela, MBA.

DEDICATORIA

A mi esposa Mayu y a mis hijos Daniel, Andrés y David por darme el apoyo para realizar este estudio y la paciencia para comprender las muchas noches de desvelo. Les amo con todo mi corazón.

AGRADECIMIENTO

A mi Dios por acompañarme en las largas noches de lectura y ayudarme a no desfallecer en el desarrollo de esta investigación. A mis padres y a mis hermanas por su ayuda desinteresada cuando más lo necesité. A mi director y lectores de tesis por sus sabios consejos y guía para la culminación exitosa de este estudio.

RESUMEN EJECUTIVO

En el Departamento de Desarrollo de Software de la PUCE no se tiene un estándar de desarrollo general, ni procesos documentados ni aprobados, lo que da como resultado que cada analista lleve a cabo, como pueda, su propio ciclo de desarrollo de software con el peligro de obviar actividades importantes en el proceso, lo que desemboca en un círculo vicioso de desperdicio de tiempo y recursos.

Como fortaleza de los procesos actuales del Departamento de Desarrollo de Software de la PUCE podemos mencionar los procesos de Ingeniería que frecuentemente se llevan a cabo para producir software funcional.

Como debilidades podemos mencionar los procesos para la Gestión de los proyectos y los procesos de Soporte, que rara vez se gestionan adecuadamente, evitando así que los procesos de Ingeniería tengan una base estable que les permita ejecutarse eficientemente durante periodos de crisis.

Como resultado del análisis de la situación actual se propone el diseño e implantación de un sistema de gestión de la calidad para el Departamento de Desarrollo de Software de la PUCE, basado el modelo CMMI-DEV V1.3 que provee disciplina y buenas prácticas para desarrollar productos software de calidad, combinado con metodologías ágiles de software como Scrum y XP, que

suministran recomendaciones para gestionar los procesos de manera incremental y evolutiva haciendo más eficiente el aprovechamiento de recursos.

Se identifica que un buen inicio es la descripción detallada de los siete procesos del nivel 2 de madurez de CMMI-DEV que proveerán una base sólida para el desarrollo posterior de los procesos de los siguientes niveles de madurez de CMMI.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 CAPÍTULO I: INTRODUCCIÓN.

1.1 ANTECEDENTES.

El tema central del presente estudio es la calidad y su relación con el ciclo de vida del desarrollo de software.

La investigación se realizará en el Departamento de Desarrollo de Software de la Pontificia Universidad Católica del Ecuador en Quito. Este departamento pertenece a la Dirección de Informática la cual a su vez depende de la Dirección General Administrativa.

El Departamento de Desarrollo de Software se encarga del análisis de requerimientos, el diseño de especificaciones, el desarrollo de código fuente, las pruebas y el control de calidad, la implementación y el mantenimiento de software interno creado para satisfacer las necesidades de la comunidad universitaria de la PUCE y también de los clientes externos.

Además, el Departamento de Desarrollo de Software administra y da mantenimiento al software de terceros adquirido por la PUCE para el funcionamiento de aplicaciones específicas de cada unidad académica o administrativa.

Para la presente investigación se considerará el período de septiembre 2014 a diciembre 2015 y se estima que los resultados serán válidos por un período de al menos dos años si la estructura organizacional del departamento no cambia.

1.2 PROBLEMA.

1.2.1 Planteamiento del problema

La Dirección de Informática de la PUCE sede Quito tiene como objetivos implementar soluciones informáticas de acuerdo a las necesidades de las unidades académicas y administrativas, brindar servicios de capacitación en informática, integrar las sedes a la estructura informática de la sede matriz, diseñar y administrar la red de telecomunicaciones, recaudar fondos por medio de la autogestión y mantener el portal universitario.

La Dirección de Informática se compone de los siguientes departamentos: Operaciones (hardware), Redes (netware), Desarrollo (software) y Base de datos.

El Departamento de Desarrollo de Software se encarga del desarrollo de aplicaciones informáticas, gestionar los proyectos de automatización de la información en áreas estratégicas, brindar el mantenimiento de las aplicaciones informáticas implementadas, analizar los requerimientos de información de los usuarios y proponer soluciones tecnológicas especializadas.

En los últimos años, dado el crecimiento del número de estudiantes, profesores y administrativos en la PUCE cada vez se requieren de mejores aplicaciones informáticas que brinden el acceso a la información académica, administrativa y financiera de la manera más confiable, ágil y segura posible.

Las diversas unidades de la PUCE solicitan constantemente a la Dirección de Informática el desarrollo de nuevas y diversas aplicaciones para manejar su información lo que ha llevado a que aumente y se diversifique la carga de trabajo de los analistas del Departamento de Desarrollo de Software, esto sumado al mantenimiento de las aplicaciones antiguas hacen que cada analista tenga una carga de trabajo importante y se ocupe en su trabajo de las tareas más urgentes como: cumplir plazos de entrega, hacer correcciones a la información solicitada, y se dejen de lado las tareas más importantes como: el análisis y diseño de requerimientos y el control y pruebas de la calidad del software.

Esto ha llevado a que en el departamento cada analista realice los procesos del ciclo de vida del desarrollo de software a su conveniencia y de acuerdo a la urgencia o prioridad que tenga la información necesitada por el usuario que solicita el software.

Además, en el departamento no se tiene un estándar de desarrollo general ni procesos claramente definidos para el desarrollo de una aplicación informática lo que da como resultado que cada analista lleve a cabo, como pueda, su propio

ciclo de desarrollo de software con el peligro de obviar actividades importantes en el proceso, lo que da como resultado software con errores, falta de documentación y datos históricos incompletos, las consiguientes quejas de los usuarios y el consecuente reproceso del software, lo que desemboca en un círculo vicioso de desperdicio de tiempo y recursos.

Dado este problema la presente investigación pretende analizar los procesos clave del desarrollo de aplicaciones informáticas en el departamento de desarrollo de software de la PUCE con el fin de encontrar y estandarizar la manera de optimizar y registrar dichos procesos para reducir el número de errores en el código de programación, reducir el tiempo de desarrollo hasta la puesta en producción, y reducir la cantidad de reprocesos o correcciones hechas al software liberado.

Con esto se logrará la meta de complacer a los usuarios y optimizar los recursos y presupuesto con los que cuenta la Dirección de Informática de la PUCE.

1.2.2 Formulación del problema

¿Cuál es un sistema de gestión de la calidad que apoya en la mejora continua de los procesos clave que intervienen en el desarrollo de aplicaciones informáticas del departamento de Desarrollo de Software de la PUCE?

1.2.3 Sistematización del problema

- ¿Cuáles son los modelos del proceso software con mejor correspondencia y aplicación a la estructura organizacional del departamento de Desarrollo de Software de la PUCE?

- ¿Cuáles son los errores más comunes que se cometen en el ciclo de desarrollo de software? ¿Cómo afectan los tiempos muertos en el cumplimiento de los plazos de la puesta en producción del software desarrollado? ¿Cómo afectan los reprocesos en la optimización del recurso humano dedicado al desarrollo de sistemas?

- ¿Cuál es la situación actual de los procesos clave que intervienen en el ciclo de desarrollo de aplicaciones informáticas del departamento de Desarrollo de Software de la PUCE? ¿Cuál es la estructura interna organizacional del departamento de Desarrollo de Software de la PUCE?

- ¿Cómo es la metodología para el mejoramiento continuo de los procesos clave del desarrollo de software? ¿Cuáles son los procedimientos para evaluar la satisfacción de los clientes con el software desarrollado?

1.3 JUSTIFICACIÓN.

1.3.1 Relevancia social

La presente investigación responde a la necesidad que tiene la Dirección de Informática de la PUCE de realizar el levantamiento de los procesos clave que intervienen en el ciclo de vida de construcción de aplicaciones informáticas con el objetivo de eliminar tiempos muertos, mejorar dichos procesos y la productividad, reducir el número de errores y evitar reprocesos, con la finalidad de aumentar la satisfacción de todos los miembros de la comunidad universitaria con el software desarrollado.

1.3.2 Relevancia académica

El diseño de un sistema de gestión de la calidad para un departamento de desarrollo de software contribuirá a ampliar la visión del tema en la implementación de procesos optimizados y documentados para la construcción de software de calidad libre de errores, pretende ser la base para futuras investigaciones y podría ser presentado en seminarios, cursos o congresos de ingeniería de software y calidad.

1.3.3 Relevancia personal

El presente estudio me permitirá especializarme en esta área de conocimiento para emprender a futuro nuevas investigaciones.

1.4 OBJETIVOS.

1.4.1 General

Proponer un sistema de gestión de la calidad para el Departamento de Desarrollo de Software de la PUCE.

1.4.2 Específicos

- Describir los modelos del proceso software con mejor correspondencia y aplicación a la estructura organizacional del departamento de Desarrollo de Software de la PUCE.
- Describir los modelos de administración de la calidad y mejoramiento de procesos para el ciclo de desarrollo de software.
- Determinar la situación actual de los procesos clave que intervienen en el desarrollo de aplicaciones informáticas del departamento de Desarrollo de Software de la PUCE.
- Determinar la metodología para el mejoramiento continuo de los procesos clave del Departamento de Desarrollo de Software de la PUCE.

1.5 PROCESO DE INVESTIGACIÓN.

1.5.1 Marco metodológico

La presente investigación será de naturaleza documental y práctica dado que se recurrirá a la recopilación de información de documentos y se estudiarán las características prácticas de los procesos clave del desarrollo de aplicaciones informáticas en el Departamento de Desarrollo de Software de la PUCE.

El tipo de la investigación será documental y de campo ya que se utilizará el estudio de documentos y estándares internacionales de calidad y el análisis de datos reales tomados directamente del ambiente de desarrollo de software.

El enfoque será cuantitativo debido a que estará basado en datos exactos y hechos concretos del manejo de la calidad en el desarrollo de software, y tendrá un alcance descriptivo ya que estudiará las propiedades de los componentes del desarrollo de software y determinará la metodología para el mejoramiento continuo de los procesos.

1.5.2 Técnicas y herramientas

Se utilizarán técnicas de recolección de información de campo como encuestas y entrevistas a expertos en el tema, la observación sistemática de los procesos del desarrollo de software y grupos de discusión.

Como herramientas de análisis de los datos se utilizará la distribución de frecuencias, diagramas de red, gráficos de barras y pastel.

También, se utilizarán herramientas informáticas para la recolección de información bibliográfica y el análisis de documentos, con rigor y cuidado para proporcionar confiabilidad y validez al estudio.

1.5.3 Fuentes.

Se utilizarán como fuentes de información primaria: los datos disponibles del Departamento de Desarrollo de Software de la PUCE, entrevistas y encuestas a los ingenieros de software del departamento en estudio, entrevistas a expertos en temas de calidad y desarrollo de software.

Como fuentes de información secundaria se considerarán: estándares internacionales de calidad, libros sobre calidad, productividad y desarrollo de software, revistas especializadas, documentos técnicos del tema, internet y bases de datos online.

1.5.4 Universo y determinación de la muestra

Los procesos que serán objeto de este estudio son los procesos clave en el desarrollo de aplicaciones informáticas.

1.5.5 Recursos

- Institucionales: Departamento de Desarrollo de Software de la PUCE.
- Infraestructura: computador, red de comunicaciones, bases de datos, servidores.
- Documentales: estructura organizacional, plan estratégico, documentos de diseño y análisis, registros de proyectos de software, estructura funcional, código fuente, cronogramas de proyectos, bases de datos de aplicaciones, manuales técnicos y de usuario.

2 CAPÍTULO II: FUNDAMENTACIÓN TEÓRICA.

2.1 PRINCIPIOS GENERALES.

El software son instrucciones de computadora que cuando se ejecutan proporcionan las características, funciones y desempeño deseado. (Roger Pressman, 2010, p. 3)

El software puede ser considerado como un producto, un vehículo para entregar información o un transformador de información.

El software se desarrolla con intelecto y se consigue su calidad a través de un buen diseño. El software reutiliza componentes para aumentar su agilidad y adaptabilidad. El software no se desgasta pero se deteriora debido a modificaciones que introducen errores y fallas en el diseño y en el código.

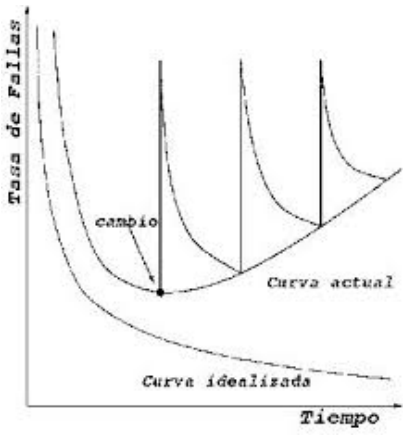


Figura 1. Curva de falla del software

Fuente: http://georgetee10.wordpress.com/author/solorzanogeor/page/2/

El software heredado o antiguo debe evolucionar constantemente debido a que el cambio es natural. Los dominios de aplicación del software pueden ser: de sistemas, de aplicación, de ingeniería y ciencias, incrustado, de línea de productos, webapps y de inteligencia artificial.

La ingeniería de software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software. (IEEE citado en Roger Pressman, 2010, p. 11)

En la ingeniería de software primero se debe entender el problema antes de dar una solución, por lo que el diseño es crucial para obtener software de alta calidad, fácil de mantener y mejorar.

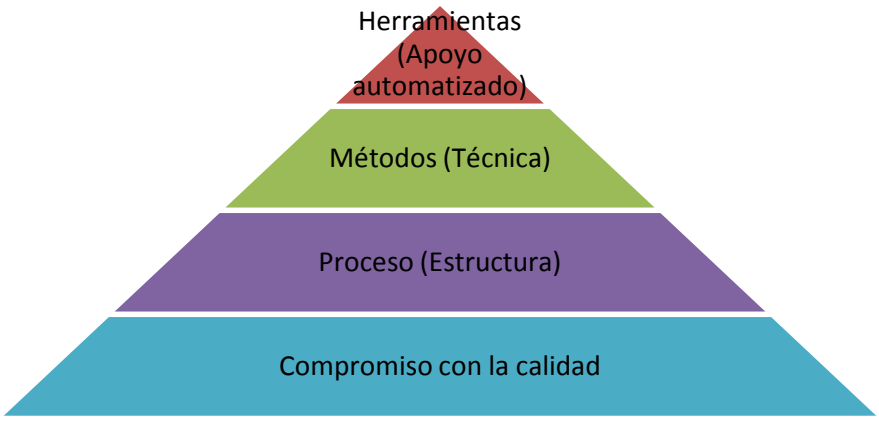


Figura 2. Capas de la ingeniería de software

Fuente: (Pressman, 2010, p. 12)

El Proceso del software es un conjunto de actividades, acciones y tareas con una estructura determinada que debe ser ágil y adaptable para producir un producto de calidad en un plazo definido.

El modelo general del proceso de software tiene las siguientes actividades estructurales:

- **Comunicación**: se obtienen los requerimientos de los participantes.
- **Planeación**: es el mapa del proyecto de software.
- **Modelado**: se realiza el diseño de los detalles del software.
- **Construcción**: se genera código fuente y se realizan pruebas
- **Despliegue**: el cliente evalúa el software y nos da retroalimentación para corregir errores y mejorar.

Existen actividades sombrilla como apoyo a las actividades generales estructurales del proceso software las cuales son: seguimiento y control del proyecto, administración del riesgo, aseguramiento de la calidad, revisiones técnicas, medición, administración de la configuración, administración de la reutilización, preparación del producto de trabajo.

“La mejor calidad conduce a menos repeticiones, lo que da como resultado tiempos de entrega más cortos.” (Roger Pressman, 2010, p. 20)

En la práctica de la ingeniería del software primero se debe entender el problema mediante la comunicación y el análisis, luego se debe planear la solución a través del

modelado y el diseño, enseguida se ejecuta el plan y se genera el código, por último se examina el resultado y se prueba y asegura la calidad.

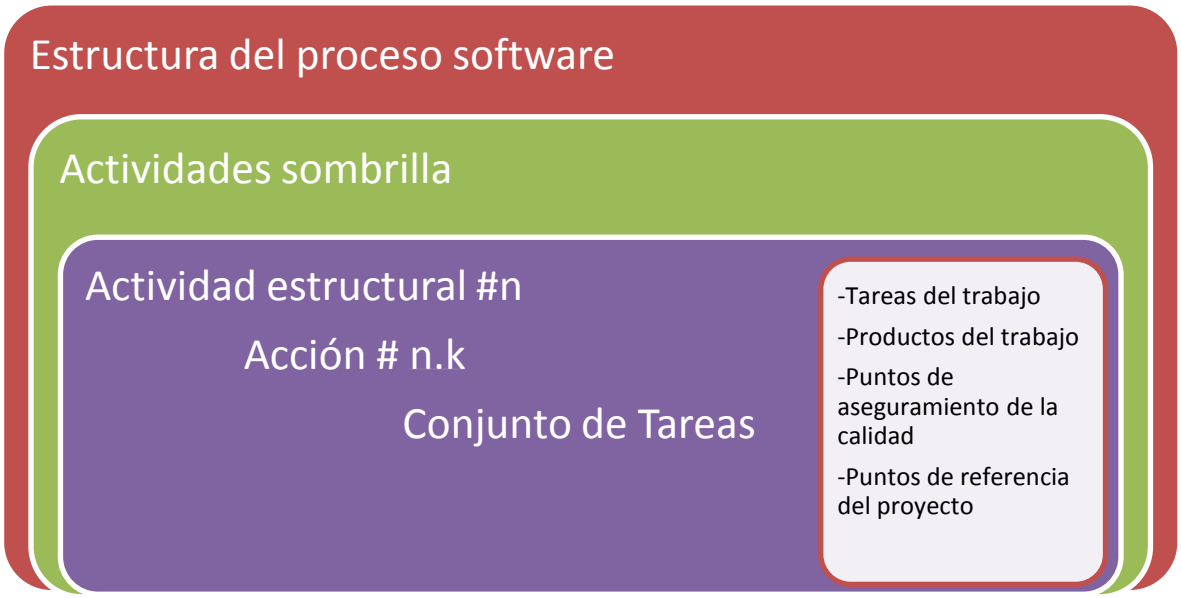


Figura 3 Modelo general del proceso software

Fuente: (Pressman, 2010, p. 27)

Las actividades estructurales se componen de acciones que son conjuntos de tareas que producen productos y se adaptan a cada equipo de proyecto software.

La evaluación y mejora del proceso busca entender el estado actual del proceso y mejorarlo. Algunos enfoques para la evaluación y mejora del proceso software son: SCAMPI (método de evaluación del estándar CMMI para el proceso de mejora), CBA IPI (evaluación basada en CMM para la mejora del proceso interno), SPICE (ISO/IEC 15504), ISO/IEC 90003:2004 (guía para la aplicación de ISO 9001:2000 a software)

Las actividades generales estructurales pueden ser organizadas mediante los siguientes flujos de proceso:

2.1.1 Lineal

En este flujo las actividades que intervienen en el desarrollo de software se organizan de manera secuencial una a continuación de la otra.

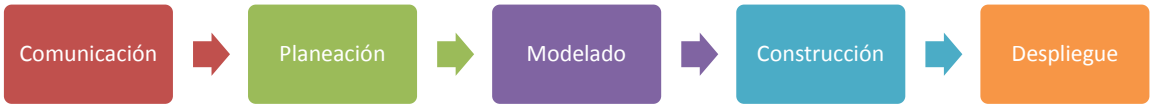


Figura 4. Flujo de proceso lineal

Fuente: (Pressman, 2010, p. 28)

2.1.2 Iterativo (bucles)

En este flujo las actividades que intervienen en el desarrollo de software se organizan de manera secuencial pero en cualquier punto pueden regresar hacia actividades anteriores.

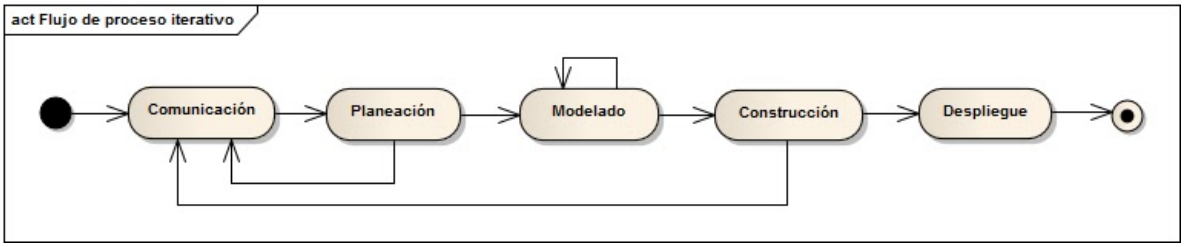


Figura 5. Flujo de proceso iterativo

Fuente: http://elbdeaina.blogspot.com/

2.1.3 Evolutivo (circular)

En este flujo las actividades que intervienen en el desarrollo de software se organizan de manera circular y en cada vuelta se obtiene un incremento de software que puede ser entregado a cliente.

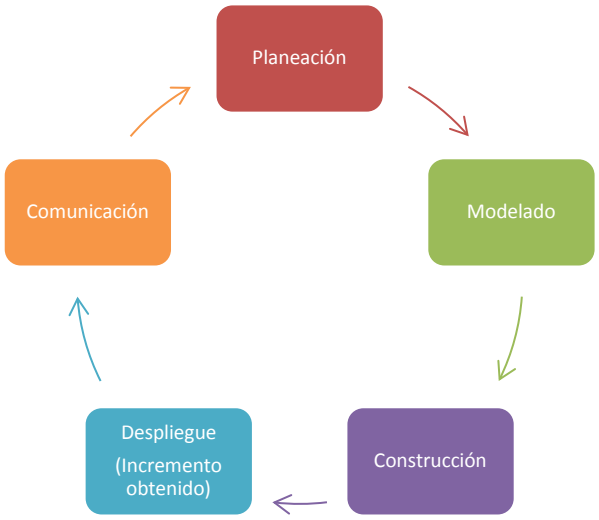


Figura 6. Flujo de proceso evolutivo

Fuente: (Pressman, 2010, p. 28)

2.1.4 En paralelo

En este flujo las actividades que intervienen en el desarrollo de software se desarrollan al mismo tiempo, es decir que pueden ser ejecutadas simultáneamente.

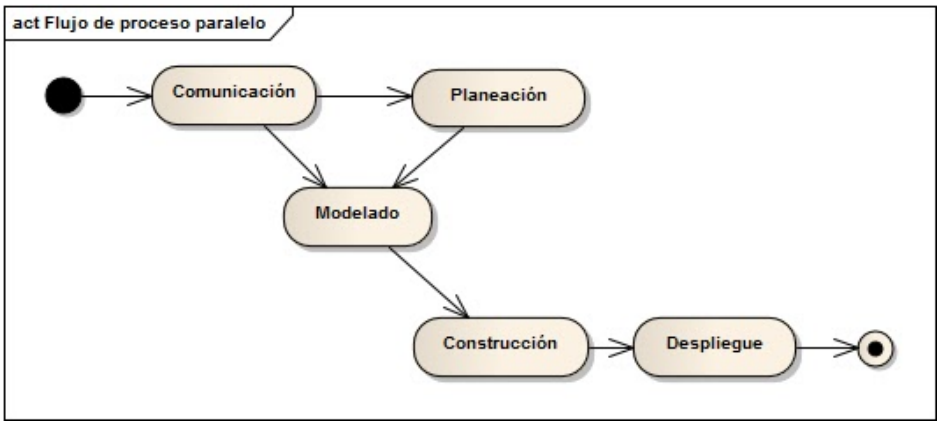


Figura 7. Flujo de proceso paralelo

Fuente: http://elbdeaina.blogspot.com/

2.2 MODELOS DEL PROCESO.

Los modelos del proceso software sirven para poner orden y estructura en el desarrollo de software para facilitar el cambio y la evolución.

2.2.1 Modelo de la cascada

El modelo de la cascada es el ciclo de vida clásico que tiene un enfoque secuencial y sistemático con un flujo lineal, en donde cada una de las actividades software se desarrolla solamente después de que la etapa anterior haya sido completada, hasta obtener el producto software terminado.

En este modelo primero se recaban los requerimientos, luego se realiza la estimación del calendario y el presupuesto, después se hace el análisis de los requisitos del cliente, luego de terminada esta etapa se procede al diseño de las especificaciones del software, a continuación se escribe el código de programación para los componentes software, y una vez terminada esta etapa se realiza la integración y pruebas de todos los componentes programados, para finalmente ejecutar la liberación y entrega del software completo al usuario final y darle el mantenimiento y soporte respectivo.

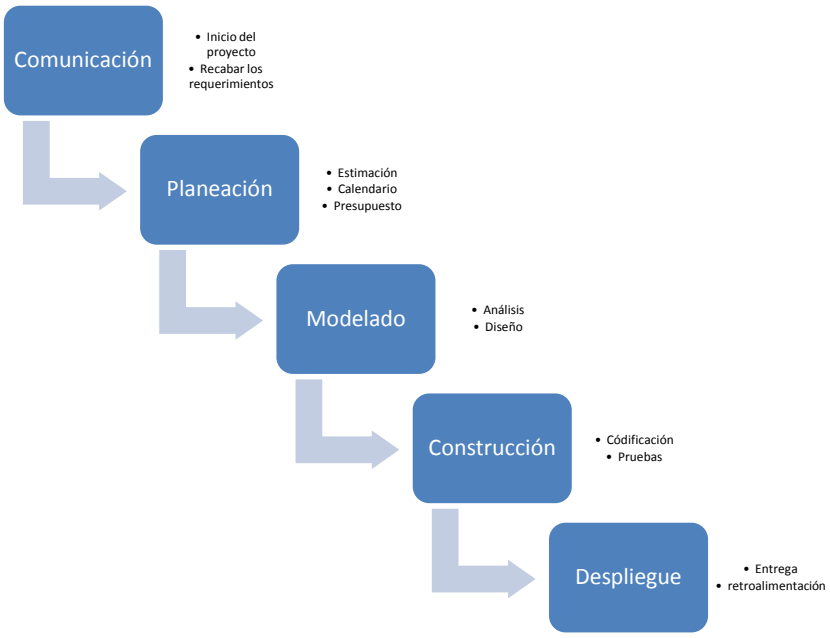


Figura 8. Modelo de la cascada

Fuente: (Roger Pressman, 2010)

Algunos problemas de este modelo son que los proyectos reales pocas veces siguen un flujo secuencial, los requerimientos deben estar completamente definidos al inicio, la primera versión funcional tarda mucho en estar operativa y al cliente no se le permite intervenir directamente en el proceso software.

2.2.2 Modelo en V

El modelo en V es similar al modelo de cascada con un flujo lineal, pero se incluyen actividades secuenciales de aseguramiento de la calidad asociadas a cada una de las actividades software principales, como por ejemplo: la validación de los requisitos del sistema, la verificación de los diseños del sistema, la verificación del código programado, la verificación de la integración de los componentes software y la validación del producto final con el cliente.

En este modelo los requerimientos del cliente no pueden cambiar una vez que ha iniciado el proyecto.

Figura 9. Modelo en V

Fuente: http://wiki.monagas.udo.edu.ve

2.2.3 Modelo de proceso incremental

Ejecuta avances que progresivamente con cada incremento dan más funcionalidad al software del cliente con un flujo lineal y en paralelo. El primer incremento da la funcionalidad básica y los demás incrementos proveen cada vez más funciones avanzadas.

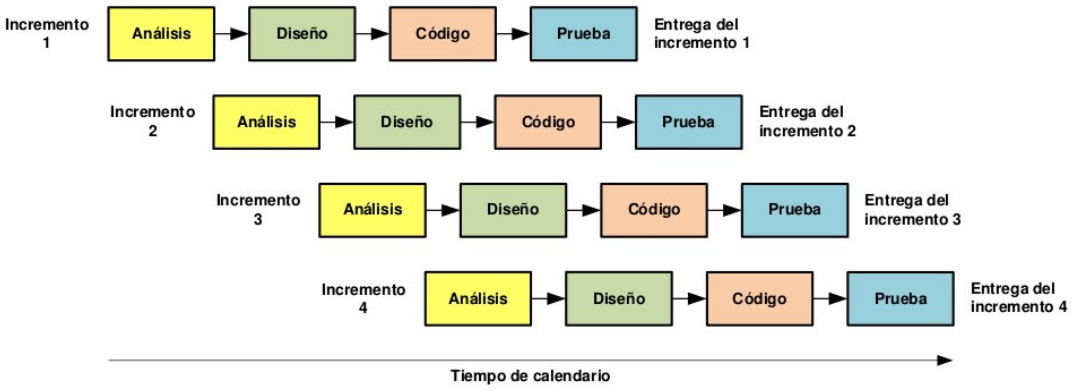


Figura 10. Modelo de proceso incremental

Fuente: http://modelosevolutivosprocesosoftware.blogspot.com

2.2.4 Modelos de proceso evolutivo

2.2.4.1 Prototipos

Este modelo se lo utiliza cuando el cliente define los objetivos generales del software pero no identifica los requerimientos detallados. Mejora la comprensión del software a elaborar cuando los requerimientos no están claros.

Utiliza un diseño rápido para identificar los detalles de los requerimientos y su construcción tiene una funcionalidad limitada. Sirve como mecanismo para entender de mejor manera el detalle de los requisitos deseados por los clientes.

Figura 11. Prototipos

Fuente: (Roger Pressman, 2010)

2.2.4.2 Modelo en espiral

Es un desarrollo rápido con entregas evolutivas (versiones) cada vez más complejas, usa prototipos en cualquier etapa de evolución del producto para reducir los riesgos técnicos antes de que se conviertan en un problema. El enfoque es cíclico con un crecimiento incremental, puntos de referencia y evaluación del riesgo técnico en cada iteración dependiendo del grado de definición del sistema.

El gerente del proyecto ajusta el número de iteraciones para determinar cuando el software ha sido terminado. El costo, la planificación y los riesgos técnicos se revisan en cada vuelta. Consta de un conjunto de puntos de referencia puntuales en cada paso evolutivo para asegurar el compromiso del participante con las soluciones y productos del trabajo.

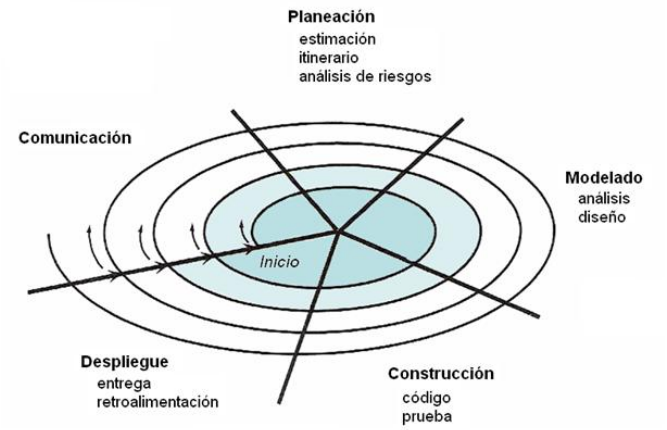


Figura 12. Modelo de espiral

Fuente: http://www.codejobs.biz/es/blog/2013/06/01/modelo-de-proceso-evolutivo#sthash.erdhTj6J.dpbs

2.2.5 Modelos concurrentes

Define eventos en la red del proceso que desencadenan transiciones de un estado a otro. Su flujo es en paralelo y proporciona un panorama del estado actual del proyecto.

Los posibles estados de las actividades pueden ser: inactivo, en desarrollo, cambios en espera, en evaluación, en revisión, terminado. Este modelo se usa cuando se tienen proyectos de ingeniería de productos y se tienen varios equipos de trabajo.

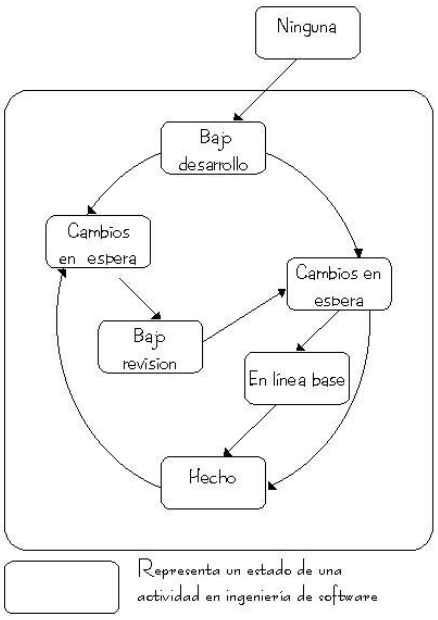


Figura 13. Elemento del proceso concurrente

Fuente: http://ingenieriadesoftwareijeanneth.blogspot.com/2010/09/modelo-de-desarrollo-rapido-de.html

2.2.6 Desarrollo basado en componentes

Su naturaleza es evolutiva con un enfoque iterativo, es parecido al modelo en espiral. Construye aplicaciones con fragmentos de software prefabricado (componentes), reutiliza software para reducir costos y tiempo de desarrollo y favorece la métrica del software.

En las actividades de modelado y construcción se realiza la identificación de los componentes candidatos desde fábricas de software, la integración de los componentes, el diseño de la arquitectura de software y las pruebas exhaustivas.

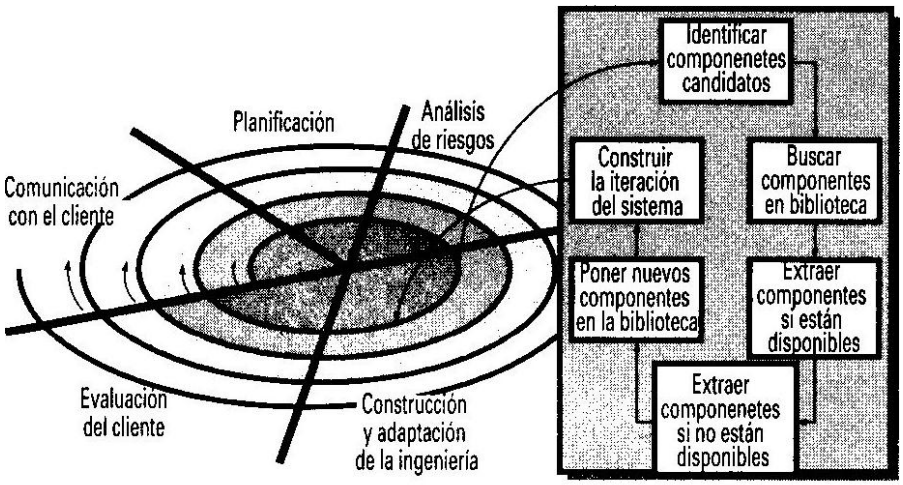


Figura 14. Desarrollo basado en componentes

Fuente: http://matriarm.wordpress.com/desarrollo-basado-en-componentes/

2.2.7 Modelo de métodos formales

Específica, desarrolla y verifica software mediante el uso de notación matemática formal rigurosa. Usa análisis matemático formal para detectar y eliminar errores y problemas de código (lo ambiguo, incompleto, inconsistente). (Roger Pressman, 2010)

Es útil para construir software de primera calidad libre de defectos como: seguridad, control aéreo, equipos médicos y bancos. Las dificultades son que consume mucho tiempo, es caro, requiere de mucha capacitación, es difícil de comprender y comunicar.

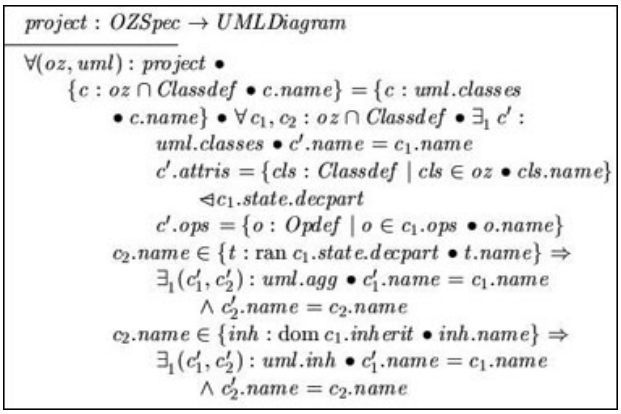


Figura 15. Modelo de métodos formales

Fuente: http://www.rodolfoquispe.org/blog/que-son-los-metodos-formales.php

2.2.8 Desarrollo de software orientado a aspectos

Es un modelo de proceso evolutivo y concurrente para definir, especificar, diseñar y construir aspectos y sus requerimientos como mecanismo para localizar el ámbito de una preocupación global (propiedades que requiere el cliente) que afectan a funciones, características e información del sistema y se extienden transversamente por toda la arquitectura de software.

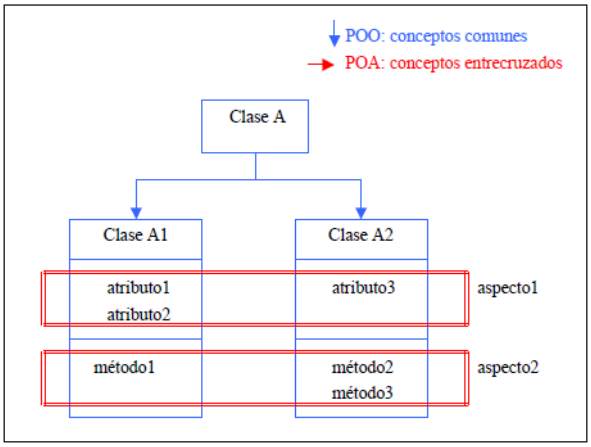


Figura 16. Desarrollo de SW orientado a aspectos

Fuente: http://www.elclubdelprogramador.com/2012/01/09/programacion-introduccion-a-la-programacion-orientada-a-aspectos/

2.2.9 Proceso Unificado

Es un proceso de flujo iterativo e incremental, impulsado por los casos de uso y centrado en la arquitectura de software, de modo que el sistema sea comprensible, permita cambios y la reutilización. Utiliza UML (lenguaje de modelado unificado) como notación robusta para el modelado de sistemas orientados a objetos.

Incluye los mejores rasgos de los modelos tradicionales junto con los mejores principios del desarrollo ágil de software. Utiliza métodos directos de comunicación con el cliente para obtener su punto de vista del sistema (casos de uso).

El proceso unificado tiene cinco fases principales:

- **Concepción**: incluye la planeación iterativa e incremental y la comunicación con el cliente para obtener los requisitos del negocio y una arquitectura aproximada del sistema.
- **Elaboración**: contiene la comunicación y el modelado que amplia y mejora los casos de uso.
- **Construcción**: desarrolla o adquiere los componentes de software necesarios para hacer operativos los casos de uso para los usuarios finales.

- **Transición**: incluye la construcción y el despliegue de la versión beta a los clientes para recibir su retroalimentación de defectos encontrados y cambios necesarios y se escriben manuales e información de apoyo.
- **Producción**: realiza el despliegue para el uso de los clientes y se presta atención a los cambios y cómo evoluciona el software.

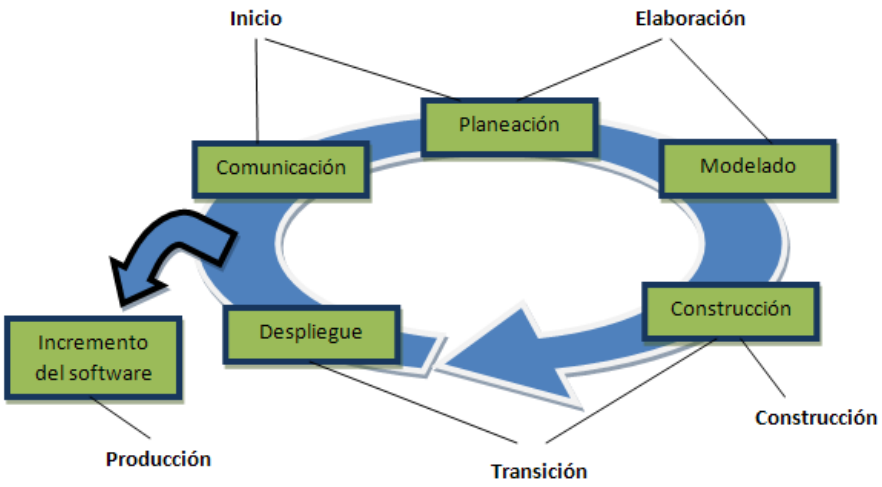


Figura 17. Proceso Unificado

Fuente: http://www.codejobs.biz/es/blog/2013/06/03/el-proceso-unificado-en-el-software

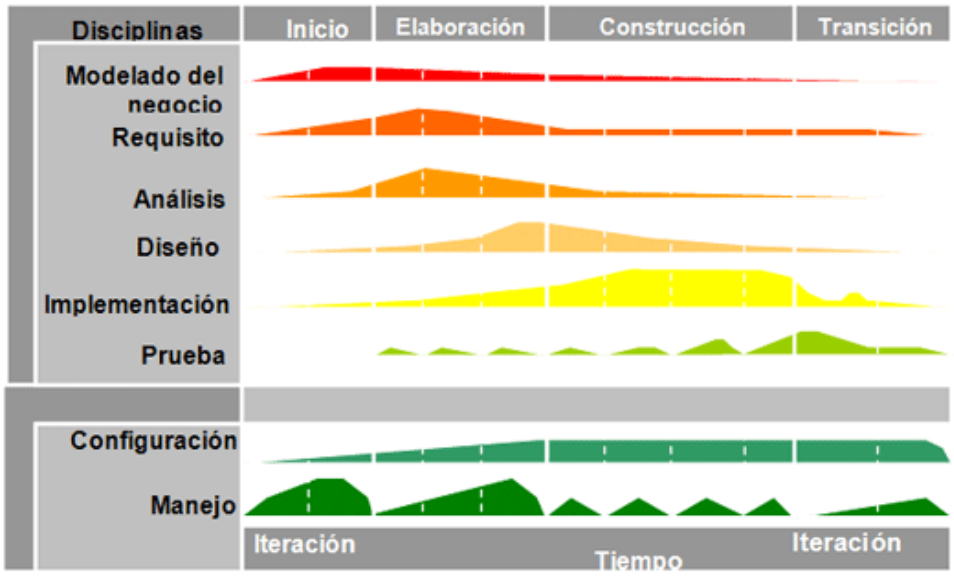


Figura 18. Flujo iterativo e incremental en el Proceso Unificado

Fuente: http://mdjesus.wordpress.com/2010/05/19/84/

2.2.10 Proceso personal de software (PPS)

Es un proceso con un enfoque disciplinado para la medición personal de la mejora en la calidad del producto software que detecta errores de manera temprana. Tiene cinco actividades estructurales:

- **Planeación**: obtiene los requerimientos del sistema y hace las estimaciones necesarias de tamaño y recursos.
- **Diseño de alto nivel**: construye prototipos, diseña y especifica componentes.
- **Revisión del diseño**: utiliza métodos de verificación para descubrir errores.
- **Desarrollo**: genera, revisa, compila y prueba código.
- **Post mórtem**: mejora la eficacia del proceso mediante mediciones.

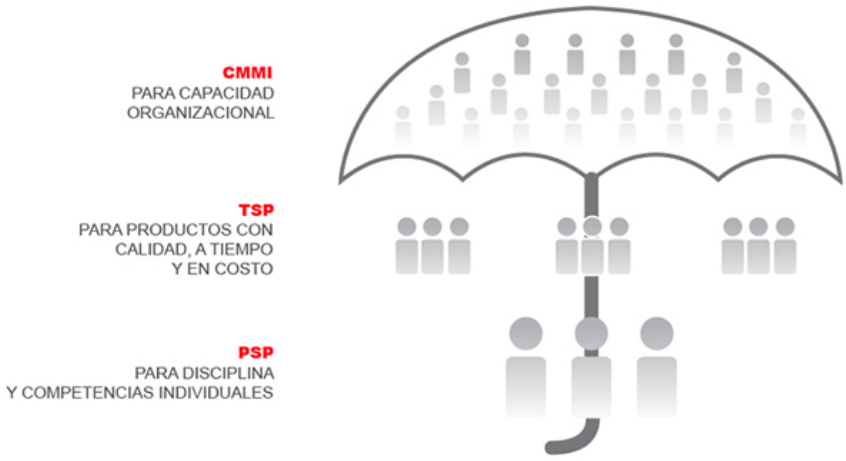


Figura 19. Proceso personal del software

Fuente: http://www.intergraphicdesigns.com/blog/2011/03/24/personal-software-process-y-team-software-process/

2.2.11 Proceso del equipo de software (PES)

En este proceso el equipo de software es autodirigido y organiza el trabajo para construir software de calidad en base al análisis de las mediciones. Define la responsabilidad de cada miembro del equipo, comprende sus metas y objetivos.

Evalúa como se reacciona al riesgo y administra el estado del proyecto. Da seguimiento cuantitativo a los datos del proyecto para la mejora continua. El gerente de software debe dirigir y motivar a su equipo.

Se compone de cinco actividades principales: inicio del proyecto, diseño de alto nivel, implementación, integración y pruebas, post mortem.

2.3 DESARROLLO ÁGIL.

> Manifiesto por el desarrollo ágil de software: Estamos descubriendo formas mejores de desarrollar software, por medio de hacerlo y de dar ayuda a otros para que lo hagan. Ese trabajo nos ha hecho valorar:
> Los individuos y sus interacciones, sobre los procesos y las herramientas.
> El software que funciona, más que la documentación exhaustiva.
> La colaboración con el cliente, y no tanto la negociación del contrato.
> Responder al cambio, mejor que apegarse a un plan.
> Es decir, si bien son valiosos los conceptos que aparecen en segundo lugar, valoramos más los que aparecen en primer sitio. ("Manifiesto por el Desarrollo Ágil de Software," n.d.)

En el desarrollo ágil es importante la satisfacción del cliente mediante la entrega rápida de software incremental. Este modelo es ideal para equipos pequeños motivados en el desarrollo sencillo de software que concluyen con rapidez sistemas exitosos. Un equipo ágil facilita la comunicación y la colaboración de los miembros del equipo.

La ingeniería de software ligero tiene las cinco actividades estructurales generales pero con un conjunto mínimo de tareas, el único producto que se entrega al cliente es un incremento de software operativo que satisfaga al cliente en la fecha acordada.

Un desarrollo ágil debe tener un plan de proyecto flexible, el cliente debe ser miembro del equipo de desarrollo, se hacen entregas rápidas incrementales y funcionales del software, la disciplina en el equipo es esencial, reduce los costos del cambio que se produce en fases tardías del proyecto, evoluciona continuamente, el tiempo de adaptación a los cambios inesperados es corto.

Este proceso se adapta a las necesidades del equipo y de las personas que deben compartir las siguientes características: competencia, enfoque común, colaboración, habilidad para tomar decisiones, capacidad para resolver problemas, confianza y respeto mutuo, organización propia.

El desarrollo ágil se adapta con rapidez a los cambios en el proyecto y en las condiciones técnicas, en cada incremento se realizan las adaptaciones apropiadas según la retroalimentación del cliente. Para mantenerse ágil el ingeniero de software debe definir procesos adaptativos y esbeltos que cubran las necesidades del negocio.

La Alianza ágil ("Agile Alliance :: Home," n.d.) define 12 principios de agilidad:

1. La prioridad más alta es satisfacer al cliente a través de la entrega pronta y continua de software valioso.
2. Son bienvenidos los requerimientos cambiantes.
3. Entregar con frecuencia software que funcione.
4. Las personas de negocios y los desarrolladores deben trabajar juntos.
5. Hay que desarrollar los proyectos con individuos motivados.
6. El método más eficiente para transmitir información es la conversación cara a cara.
7. La medida principal de avance es el software que funciona.
8. Los procesos ágiles promueven el desarrollo sostenible.
9. La atención continua a la excelencia técnica y el buen diseño mejora la agilidad.
10. Es esencial la simplicidad.

11. Las mejores arquitecturas, requerimientos y diseños surgen de los equipos con organización propia.
12. El equipo reflexiona a intervalos regulares sobre cómo ser más eficaz.

Figura 20. Desarrollo ágil

Fuente: http://solucionesagiles.net/ModeloAgil

Según McMahon (2011) las organizaciones que entienden e implementan prácticas de desarrollo ágil apropiadamente tienden a ser más disciplinadas en sus prácticas de gestión y desarrollo que muchas organizaciones tradicionales de desarrollo.

Tabla 1. Perspectivas tradicional y ágil en desarrollo de software

Tema	Enfoque tradicional	Enfoque ágil
Ciclo de vida del desarrollo	Lineal, cascada, en espiral	Iterativo, modelo de entrega evolutivo
Estilo de desarrollo	Anticipatorio	Adaptativo
Requerimientos	Conocido tempranamente, estable, claramente definido y documentado	Emergente, cambio rápido, desconocido – descubierto durante el proyecto
Arquitectura	Arquitectura pesada para requerimientos actuales y futuros	Precepto: No lo vas a necesitar.
Gestión	Centrado en el proceso, ordenar y controlar	Centrado en las personas, liderazgo y colaboración
Documentación	Pesada, detallada, conocimiento explícito	Ligera, reemplazada por comunicación cara a cara, conocimiento tácito
Meta	Predicción y optimización	Exploración y adaptación
Cambios	Aversión al cambio	Acepta el cambio
Miembros del equipo	Equipos distribuidos de especialistas, orientados a un plan, habilidades adecuadas	Ágil, conocedor, colaborativo y en el mismo lugar
Organización del equipo	Equipos pre-estructurados	Equipos auto organizados
Involucración del cliente	Pasivo, baja involucración	Activo, proactivo, cliente en el sitio y considerado un miembro del equipo
Cultura organizacional	Cultura de ordenar y controlar	Cultura de colaboración y liderazgo
Proceso de desarrollo de software	Enfoque universal y solución para proveer predicción y alto aseguramiento	Enfoque flexible adaptado a las necesidades del contexto para proveer un desarrollo más rápido
Medida de éxito	Conformidad con el plan	Valor comercial entregado

Fuente: (Moniruzzaman & Hossain, 2012)

En la siguiente figura de una encuesta realizada por Scott Ambler se muestra consistentemente que los proyectos ágiles son más exitosos comparados con los proyectos tradicionales.

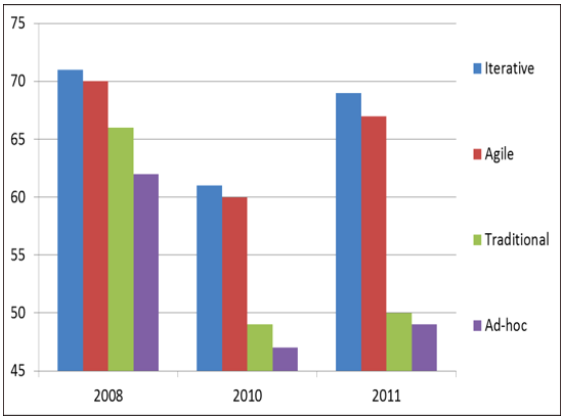


Figura 21. Tasa de éxito en proyectos de software

Fuente: ("Surveys Exploring The Current State of Information Technology Practices," n.d.)

2.3.1 Programación extrema (XP)

Existe un conjunto de cinco valores fundamentales en la programación extrema:

- Comunicación eficaz
- Simplicidad en el diseño de las necesidades inmediatas
- Retroalimentación del cliente y pruebas unitarias.
- Valentía y disciplina en diseñar solo para hoy
- Respeto entre los miembros. (Beck, 2004)

La programación extrema tiene su enfoque en la orientación a objetos y está formado por un conjunto de reglas agrupadas en cuatro actividades estructurales:

1. **Planeación**: se recogen requerimientos en forma de historias de los usuarios que describen las salidas y funcionalidades del producto software, el equipo asigna las historias a incluir en el próximo incremento de acuerdo a las prioridades.
2. **Diseño**: la idea es mantenerlo sencillo, el diseño guía la implementación de la historia tal cual se escribe sin más ni menos, el único producto de esta actividad son las tarjetas CRC (clase, responsabilidad, colaborador) que organizan las clases orientadas a objetos en el incremento actual.
3. **Codificación**: se desarrollan pruebas unitarias de cada historia en el incremento, se centra en implementar el código para pasar la prueba unitaria de cada requerimiento, el trabajo en parejas de programadores en la misma estación de trabajo permite descubrir errores tempranamente.

4. **Pruebas**: se realizan las pruebas unitarias que deben implementarse en una estructura automatizada que permita ejecutar las pruebas repetidas veces con facilidad. Las pruebas de integración validan el sistema a diario y nos dan indicaciones del avance o señales de alerta del proyecto software. Las pruebas de aceptación son características generales del sistema especificadas por el cliente. (Roger Pressman, 2010, p. 67)

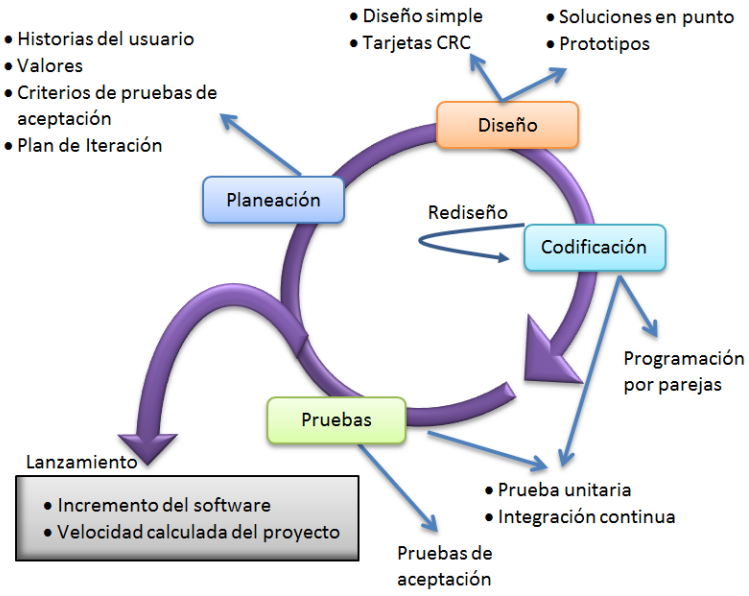


Figura 22. Proceso de la programación extrema

Fuente: http://www.codejobs.biz/es/blog/2013/06/05/programacion-extrema-xp#sthash.f0XWUOqG.dpbs

2.3.2 Desarrollo adaptativo de software (DAS)

Tiene un flujo iterativo con una planeación adaptativa que utiliza métodos rigurosos para levantar los requerimientos y para las revisiones técnicas usa métodos formales que proveen retroalimentación en tiempo real. La colaboración y el aprendizaje dentro del equipo son factores clave para elaborar sistemas complejos con esta técnica.

El DAS tiene tres actividades estructurales: especulación, colaboración y aprendizaje.

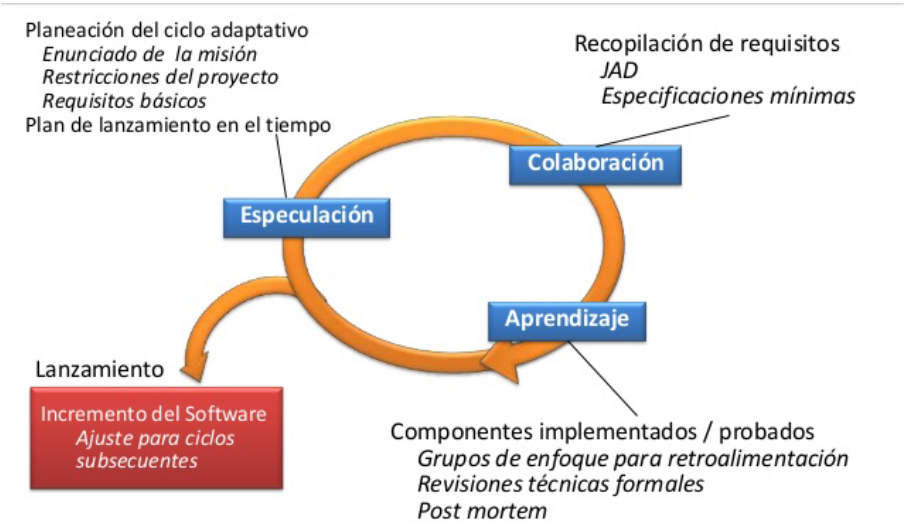


Figura 23. Desarrollo adaptativo de software

Fuente: http://es.slideshare.net/coesiconsultoria/4-desarrollo-gil-del-software

2.3.3 SCRUM

Scrum es un modelo para gestionar el desarrollo de productos de software, emplea un enfoque iterativo e incremental para mejorar el control del riesgo y la predicción. Scrum se basa en tres pilares: transparencia, inspección y adaptación. (Schwaber & Sutherland, 2013)

El Equipo Scrum está formado por:

- el Dueño del Producto: está a cargo de maximizar el trabajo del Equipo de Desarrollo y el valor real del producto final, y es el único responsable de gestionar el Backlog del Producto.
- el Scrum Master: es el encargado de asegurar que el Equipo Scrum siga las reglas y prácticas de Scrum, ayuda al Dueño del Producto a gestionar de manera más eficiente el Backlog de Producto, ayuda al Equipo de Desarrollo a eliminar impedimentos,
- el Equipo de Desarrollo: realiza el trabajo necesario para entregar un incremento de producto terminado al finalizar cada sprint. Son autos organizados, multifuncionales y su tamaño debe ser de entre 3 a 9 personas.

Los eventos más importantes de Scrum son:

- Sprint: es un bloque fijo de tiempo de menos de un mes de duración en el cual se desarrolla un incremento de producto listo para ser entregado, durante el sprint no se realizan cambios a los objetivos del sprint.

- Reunión de planificación del Sprint: el Equipo Scrum completo trabaja un día entero para planificar el siguiente sprint, aquí se responden las siguientes preguntas: ¿Qué historias de usuario pueden entregarse al final del sprint que está por comenzar? ¿Cómo el Equipo Scrum realizará el trabajo para entregar el incremento funcional de software?
- Objetivo de Sprint: es una meta que el Equipo Scrum se propone lograr a través de la implementación del Backlog del Sprint.
- Scrum Diario: es una reunión de 15 minutos en el mismo lugar y a la misma hora para que el Equipo de Desarrollo sincronice sus tareas, mejore la comunicación y responda las siguientes preguntas: ¿Qué se ha hecho desde ayer? ¿Se ha tenido algún problema? ¿Qué se va a hacer hasta mañana?
- Revisión del Sprint: es una reunión al finalizar el sprint en donde el Equipo Scrum presenta el incremento terminado a las partes interesadas relevantes para retroalimentar información y fomentar la colaboración.
- Retrospectiva de Sprint: su propósito es revisar cómo le fue al Equipo Scrum en el último Sprint e identificar las posibles mejoras para ser implementadas en el próximo sprint, tiene lugar después de la reunión de Revisión del Sprint.

Los artefactos de Scrum son los siguientes:

- Backlog de Producto: es una lista ordenada de los requerimientos de los clientes en forma de historias de usuario que pueden ser

necesarios para completar el producto, es la única fuente de requisitos para el producto. El Backlog de producto es dinámico, los elementos de orden más alto que han sido refinados por el Equipo Scrum y tienen mayor detalle son considerados como listos para ser incluidos en el siguiente Sprint.

- Gráficos de trabajo consumido (Burndown Chart): son gráficos que muestran la estimación del trabajo restante para completar todas las tareas del sprint.
- Backlog del Sprint: son las tareas del Backlog del Producto seleccionados por el Equipo Scrum como predicción para ser desarrolladas en el próximo sprint, incluida una descripción del trabajo necesario para entregar el incremento de software terminado. Esta lista debe tener el suficiente detalle para que los cambios en el progreso se hagan visibles en el Scrum diario.
- Incremento de software: al finalizar un sprint el nuevo incremento debe cumplir con la definición de Terminado y debe estar en condiciones de ser utilizado en el próximo despliegue y que funcione en conjunto y sin problemas con los incrementos anteriores.
- Definición de Terminado: se usa para determinar cuándo un elemento del Backlog de Producto está completado, todo el Equipo Scrum debe compartir esta definición.

Figura 24. Flujo del proceso Scrum

Fuente: http://metagdes.blogspot.com/2012_05_01_archive.html

2.3.4 Método de desarrollo de sistemas dinámicos (MDSD)

La filosofía de MDSD dice que el 80% de la aplicación puede entregarse en 20% del tiempo que tomaría entregarla completa. En cada incremento solo se realiza el trabajo suficiente para pasar al siguiente incremento. En cada iteración se aplica la regla del 80%. (Roger Pressman, 2010, p. 71)

El ciclo de vida MDSD tiene cinco actividades: estudio de factibilidad, estudio del negocio (requerimientos), iteración del modelo funcional (prototipos), diseño e iteración de la construcción, e implementación.

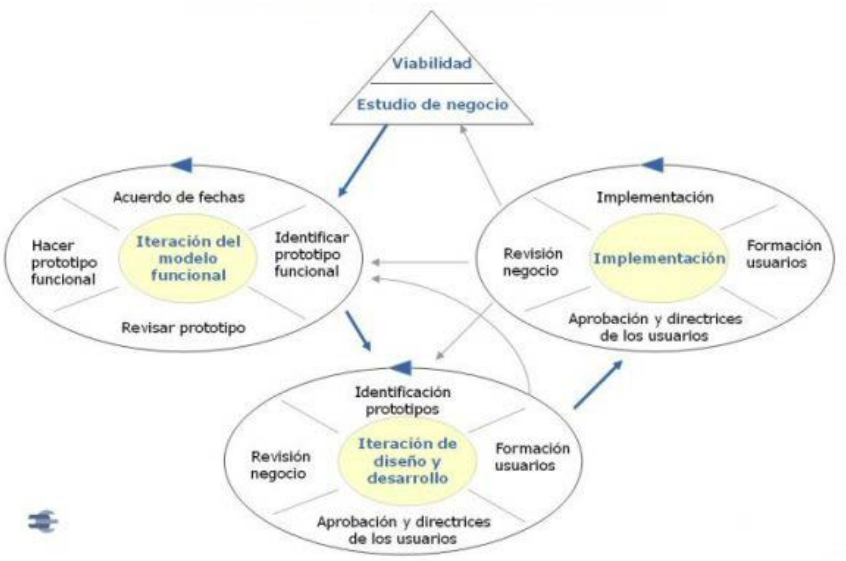


Figura 25. Método de desarrollo de sistemas dinámicos

Fuente: http://metodologiadsdm.wordpress.com/author/eslunap/

2.4 MODELO DE REFERENCIA ISO/IEC 12207 – 2008.

Es el modelo de referencia de procesos del ciclo de vida del software. Enumera y categoriza las actividades estructurales para implantar un proceso que considere todos los aspectos.

Los procesos se dividen en dos categorías:

- **Procesos de contexto del sistema**: son actividades que no son específicas de la ingeniería de sistemas y que permiten la sistematización de las actividades previas al desarrollo.

Tabla 2. Procesos de contexto del sistema

Procesos de Acuerdo	**Procesos Organizativos habilitadores del proyecto**	**Procesos de Proyecto**	**Procesos Técnicos**
Proceso de adquisición	Gestión de modelos de ciclo de vida	Proceso de planificación del proyecto	Proceso de definición de requisitos por parte de todas las partes interesadas
Proceso de provisión	Proceso de gestión de infraestructura	Proceso de evaluación y control de proyectos	Proceso de análisis de requisitos del sistema
	Proceso de gestión del portafolio de proyectos	Proceso de gestión de decisiones	Proceso de diseño de la arquitectura del sistema
	Proceso de gestión de recursos humanos	Proceso de gestión de riesgo	Proceso de implementación
	Proceso de gestión de la calidad	Proceso de gestión de la configuración	Proceso de pruebas
		Proceso de gestión de la información	Proceso de instalación
		Proceso de medición	Proceso de soporte de aceptación de software
			Proceso de operación del software
			Proceso de mantenimiento del software
			Proceso de baja del software

Fuente: (Sánchez, Sicilia, & Rodríguez, 2012)

- **Procesos específicos del software**: son actividades que son específicas de la ingeniería de sistemas y el desarrollo de software.

Tabla 3. Procesos específicos del software

Proceso de Implementación del software	**Procesos de Soporte**	**Procesos de Reutilización**
Proceso de análisis de requisitos del software	Proceso de gestión de la documentación del software	Proceso de ingeniería del dominio
Proceso de diseño de la arquitectura del software	Proceso de gestión de la configuración del software	Proceso de gestión de recursos reutilizables
Proceso de diseño detallado del software	Proceso de aseguramiento de la calidad	Proceso de gestión de programas de reutilización
Proceso de construcción del software	Proceso de la verificación del software	
Proceso de integración del software	Proceso de validación del software	
Proceso de Pruebas del software	Proceso de revisión del software	
	Proceso de auditoría del software	
	Proceso de resolución de problemas software	

Fuente: (Sánchez et al., 2012)

3 CAPÍTULO III: ADMINISTRACIÓN DE LA CALIDAD.

3.1 CONCEPTOS DE CALIDAD.

3.1.1 Calidad del Software

La calidad del software consiste en un proceso eficaz mediante la ingeniería de software, que crea un producto útil y confiable, que tiene un valor medible y rentable para quienes lo producen y para quienes lo utilizan, por ejemplo, agiliza un proceso del negocio. (Bessin, 2004)

Este producto útil proporciona el contenido, funciones y las características que desea el usuario final. El proceso utiliza infraestructura tecnológica como apoyo para la elaboración de software de alta calidad, implementa actividades sombrilla para la administración del cambio y para las revisiones técnicas.

3.1.1.1 Dimensiones de la calidad de Garvin

Se enfocan en un punto de vista multidimensional para la evaluación de la conformidad y lista un conjunto de factores suaves (subjetivos) para medir la calidad:

- *Calidad del desempeño*: contenido, funciones, características especificadas.

- *Calidad de las características*: agradables.
- *Confiabilidad*: libre de errores.
- *Conformidad*: concuerde con el diseño.
- *Durabilidad*: cambiar y depurarse.
- *Servicio*: corregir defectos.
- *Estética*: un flujo único.
- *Percepción*: buena reputación. (Garvin citado en Roger Pressman, 2010, p. 341)

3.1.1.2 Factores de la calidad de McCall

Están agrupados en tres categorías: operación del producto, revisión del producto y transición del producto. (McCall, Richards, & Walters, 1977)

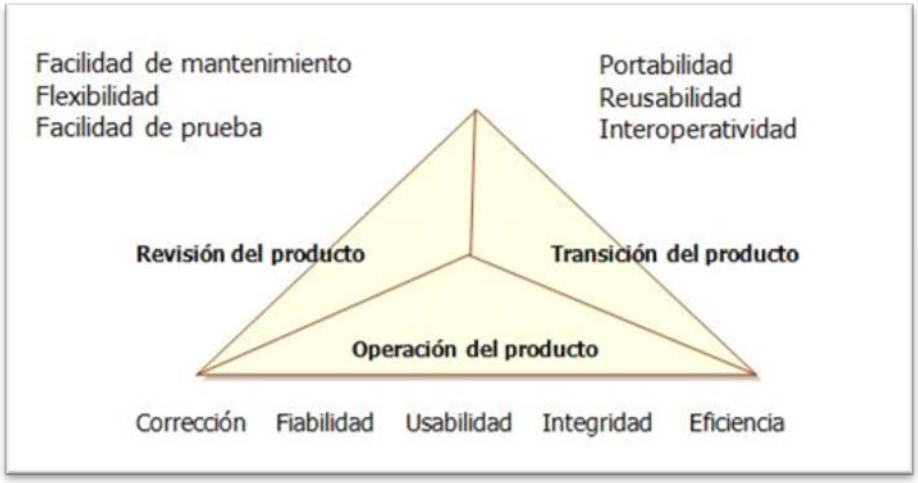


Figura 26. Factores de la calidad de McCall

Fuente: http://datateca.unad.edu.co/contenidos/301404/301404_ContenidoEnLinea/leccin_44__mtricas_tcnicas_del_software.html

- *Operación del producto*
    - *Corrección*: satisface especificaciones
    - *Confiabilidad*: cumple con su función.
    - *Eficiencia*: recursos requeridos.
    - *Integridad*: control de acceso.
    - *Usabilidad*: fácil de aprender.
- *Revisión del producto*
    - *Facilidad de recibir mantenimiento:* detectar y corregir un error.
    - *Flexibilidad:* modificar el programa.
    - *Susceptibilidad de someterse a pruebas:* probar el programa.
- *Transición del producto*
    - *Portabilidad:* transferir de un ambiente a otro.
    - *Reusabilidad:* puede volverse a utilizar.
    - *Interoperabilidad:* acoplar un sistema con otro.

3.1.1.3 Factores de la calidad ISO/IEC 25000 SQuaRE

Proporcionan una lista de comprobación para evaluar la calidad del sistema y para realizar mediciones indirectas cuantificables de los atributos clave de la calidad del software:

- *Funcionalidad*: adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad.
- *Confiabilidad*: madurez, tolerancia a fallas y recuperación.
- *Usabilidad*: entendible y operable.
- *Eficiencia*: comportamiento del tiempo y de los recursos.
- *Facilidad de recibir mantenimiento*: analizable, cambiable, estable, susceptible de someterse a pruebas.
- *Portabilidad*: adaptable, instalable, conformidad y sustituible. ("ISO/IEC 25010:2011(en), Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models," n.d.)

3.1.2 Costos de la calidad del software

Un software es suficientemente bueno si alcanza un balance entre el esfuerzo y la calidad aceptable sin enterrar al proyecto. Controlar la calidad en el desarrollo de software cuesta tiempo y dinero, pero el costo de la mala calidad normalmente tiene costos superiores. Un software de mala calidad produce fallas en la seguridad y riesgos tanto para el desarrollador como para el usuario final. (Roger Pressman, 2010)

Existen dominios de aplicación para los cuales un software suficientemente bueno o con algunos errores no es aceptable y se considera negligencia o delito, por ejemplo: software incrustado en tiempo real, software integrado con hardware, software de aviones.

El costo de la calidad puede dividirse en:

- *Costos de prevención*: incluyen actividades de administración.
- *Costos de evaluación*: incluyen actividades de control.
- *Costos de falla*
    - *Costos de falla internos*: detecta errores antes del envío a la siguiente actividad.
    - *Costos de falla externos*: encuentra defectos después del envío al usuario final.

Los costos se incrementan aceleradamente cuando los errores no se detectan a tiempo.

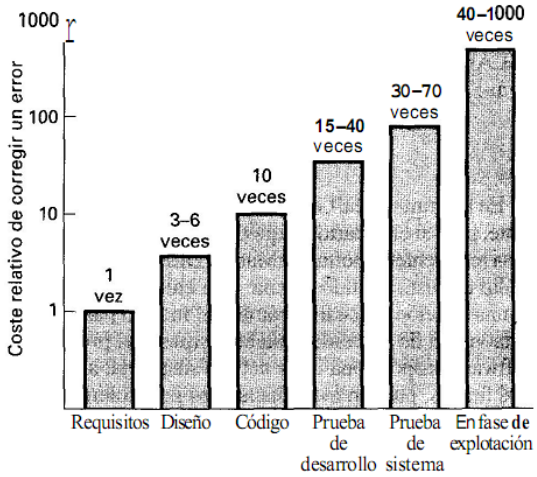


Figura 27. Costo relativo de corregir errores y defectos

Fuente: http://guiasadsikarlos.blogspot.com/

3.1.3 Lograr alta calidad

Una alta calidad del software es el resultado de una buena administración del proyecto en conjunto con una correcta práctica de la ingeniería del software.

Existen cuatro actividades principales para lograr una alta calidad:

- *Métodos de ingeniería de software*: se entiende el problema a resolver y se usan métodos de análisis y diseño.
- *Técnicas de administración de proyectos*: se realizan estimaciones de fechas, se tienen claras las dependencias entre las actividades, y se planea el riesgo.

- *Control de calidad*: se inspecciona el código para descubrir y corregir errores lo antes posible.
- *Aseguramiento de la calidad*: se toman acciones necesarias para lograr una alta calidad del software. (Roger Pressman, 2010)

3.2 TÉCNICAS DE REVISIÓN

3.2.1 Defectos del Software

Un *error* es un problema de calidad que se detecta antes de que se entregue el software a otra actividad estructural del proceso. Un *defecto* es un problema de calidad que se encuentra después de entregar al software a los usuarios finales.

Las revisiones técnicas se utilizan para encontrar errores en el proceso de software para que no se conviertan en defectos al liberar el software, entonces tienen como objetivo descubrir errores tempranamente. El costo total cuando se efectúan revisiones es casi tres veces más bajo que cuando no se hacen revisiones. (Roger Pressman, 2010)

3.2.2 Métricas de revisión

Se debe definir un conjunto de métricas que se utilicen para evaluar la eficacia de cada actividad.

Ejemplos de métricas para la revisión:

- *Esfuerzo de preparación (E_p)*: horas-hombre para revisar un producto antes de una reunión.
- *Esfuerzo de evaluación (E_a):* hora-hombre para la revisión real.
- *Esfuerzo de la repetición (E_r):* horas-hombre para la corrección de errores descubiertos durante la revisión.
- *Tamaño del producto del trabajo (TPT):* medición del tamaño del producto que se ha revisado.
- *Errores menores detectados ($Err_{menores}$):* requieren menos de un esfuerzo determinado para corregirse.
- *Errores mayores detectados ($Err_{mayores}$):* requieren más de un esfuerzo determinado para corregirse.
- *Esfuerzo total de revisión ($E_{revisión}$) = $E_p + E_a + E_r$*
- *Número total de errores descubiertos (Err_{tot}) = $Err_{menores} + Err_{mayores}$*
- *Densidad del error = Err_{tot} / TPT*. (Roger Pressman, 2010)

El ahorro en el esfuerzo en horas-hombre debidas a las revisiones técnicas se ve en la reducción del esfuerzo en la etapa de pruebas. Los beneficios son ciclos de

entrega más cortos, menor tiempo para liberar el software y una buena relación costo-beneficio.

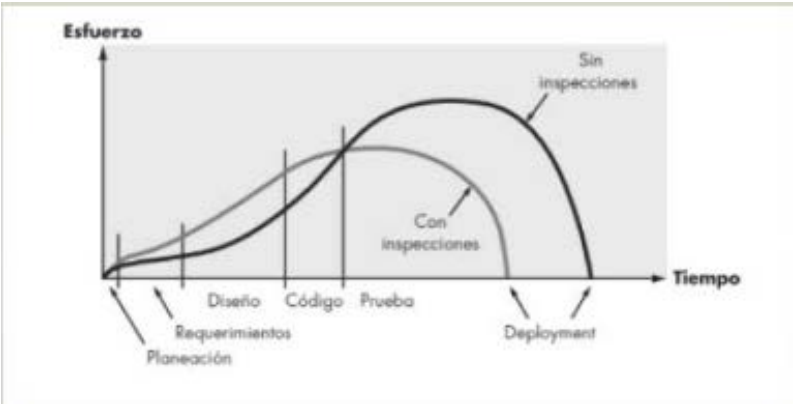


Figura 28. Esfuerzo realizado con y sin revisiones

Fuente: (Roger Pressman, 2010)

3.2.3 Revisiones informales

Un tipo de revisión informal es una reunión imprevista de más de dos personas para revisar productos del trabajo software en busca de errores que luego son registrados y resueltos por el desarrollador. (Roger Pressman, 2010)

La programación por pares usada en el proceso XP es un tipo de verificación de escritorio continua que actúa como un mecanismo para detectar y resolver problemas, y asegurar la calidad en tiempo real. La redundancia en las tareas en la programación por parejas se justifica por los ahorros relacionados con una mayor calidad del producto software como por ejemplo evitar los reprocesos.

3.2.4 Revisiones formales

Es una actividad para el control de calidad que pretende descubrir errores en la implementación, lógica y funcionamiento del software. Verifica que el software cumple los requerimientos de los participantes, garantiza que el software fue desarrollado de manera uniforme y en base a estándares predefinidos, y contribuye a hacer los proyectos más manejables. (Roger Pressman, 2010)

Las revisiones formales se centran en una parte específica del producto para tener una mayor probabilidad de detectar errores, participan pocos desarrolladores que deben tener un conocimiento previo del producto software a revisar.

Figura 29. Modelo para revisiones formales

Fuente: (Roger Pressman, 2010)

3.3 ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE.

El objetivo del aseguramiento de la calidad del software es producir software a tiempo y de alta calidad, reducir la cantidad de repeticiones y disminuir el tiempo para salir al mercado. Es una actividad que garantiza que se hacen las tareas correctas, en el momento correcto y en la forma correcta. (Schulmeyer & McManus, 1998)

Un sistema de aseguramiento de la calidad se define como una estructura organizacional con responsabilidades, procedimientos, procesos y recursos necesarios para implementar una administración de la calidad. Los sistemas de aseguramiento de la calidad ayudan a las organizaciones a asegurar que sus productos y servicios satisfagan las expectativas de los clientes cumpliendo sus especificaciones. (Roger Pressman, 2010)

Las actividades de la administración de la calidad son los siguientes:

- *Estándares*: asegurar que los estándares se sigan.
- *Revisiones y auditorías*: detectar errores.
- *Pruebas*: planear de forma apropiada.
- *Colección y análisis de los errores*: medir como se está haciendo algo.
- *Administración del cambio*: controlar los cambios.
- *Educación*: mejorar prácticas de ingeniería de software.
- *Administración de los proveedores*: incorporación de cláusulas de calidad en el contrato.

- *Administración de la seguridad*: utilizar tecnología apropiada.
- *Seguridad*: disminuir el riesgo.
- *Administración de riesgos*: establecer planes de contingencia.
- *Actividades de apoyo*: se produzcan con calidad. (Roger Pressman, 2010)

3.3.1 Metas del aseguramiento de la calidad

El grupo de aseguramiento de la calidad del software tiene la responsabilidad de planear, analizar y hacer reportes acerca de la calidad del proceso software.

Las tareas del grupo de aseguramiento de la calidad del software son:

- Preparar el plan de aseguramiento de la calidad.
- Participar en la selección del proceso de software.
- Revisar las actividades de la ingeniería de software.
- Audita los productos para verificar las correcciones.
- Asegura que el trabajo y sus productos se documenten.
- Registra la falta de cumplimiento y reporta a la alta dirección.
- Coordina el control del cambio.
- Ayuda a recolectar métricas. (Roger Pressman, 2010)

Tabla 4. Metas, atributos y métricas de la calidad del software

META	ATRIBUTO	MÉTRICA
Calidad de requerimientos	Ambigüedad	Número de modificadores ambiguos
	Completitud	Número de historias completas
	Comprensibilidad	Número de secciones y subsecciones
	Volatilidad	Números de cambios por requerimiento
	Trazabilidad	Número de requerimientos no trazables hasta el diseño
	Claridad del modelo	Número de modelos UML
Calidad del diseño	Integridad arquitectónica	Existencia del modelo arquitectónico
	Completitud de componentes	Número de componentes que se siguen hasta el modelo arquitectónico
	Complejidad de la interfaz	Número promedio de pasos para llegar a una función
	Patrones	Número de patrones utilizados
Calidad del código	Complejidad	Complejidad ciclomática
	Facilidad de mantenimiento	Factores de diseño
	Comprensibilidad	Porcentaje de comentarios internos
	Reusabilidad	Porcentaje de componentes reutilizados
	Documentación	Índice de legibilidad
Eficacia del control de calidad	Asignación de recursos	Porcentaje de personal por hora y por actividad
	Tasa de finalización	Tiempo de terminación real versus lo planeado
	Eficacia de la revisión	Medición de la revisión
	Eficacia de las pruebas	Número de errores de importancia crítica encontrados

Fuente: (Roger Pressman, 2010)

3.3.2 Confiabilidad del software

Es la probabilidad de que un programa opere sin fallas en un ambiente específico en un tiempo determinado. Las fallas se producen por falta de conformidad con los requisitos, pueden rastrearse a problemas en el diseño o la codificación, y pueden ser leves o catastróficas. (Musa citado en Roger Pressman, 2010, p. 376)

Un par de medidas de la confiabilidad de un sistema son:

Tiempo medio entre fallas (TMEF) = Tiempo medio para la falla (TMPF) + Tiempo medio para la reparación (TMPR)

Fallas en el tiempo (FET) = es una medición estadística del número de fallas en un componente en mil millones de horas de operación.

La disponibilidad del software mide la probabilidad de una aplicación software de funcionar conforme a los requisitos en un momento determinado:

Disponibilidad = (TMPF / (TMPF + TMPR))

3.3.3 Plan de aseguramiento de la calidad del software

El IEEE (1993) ha publicado una norma para el aseguramiento de la calidad del software que identifica los siguientes aspectos:

- Propósito y alcance del plan
- Descripción de los productos de trabajo
- Normas y prácticas aplicables
- Acciones de revisión y auditoría
- Herramientas y métodos que apoyen a las tareas
- Procedimientos para la administración de la configuración
- Métodos para mantener los registros
- Roles y responsabilidades relacionados con la calidad

3.4 NORMAS DE CALIDAD ISO 9000.

Un sistema de gestión de la calidad es un conjunto de elementos interrelacionados para establecer políticas de la calidad, objetivos de la calidad y procesos de calidad, con el propósito de facilitar la consecución de la visión y los objetivos del negocio y aumentar la satisfacción del cliente. ("ISO 9000:2015(es), Sistemas de gestión de la calidad — Fundamentos y vocabulario," n.d.)

Los principios básicos de un sistema de gestión de la calidad son:

- *Orientación al cliente*: cumplir y superar sus expectativas.
- *Enfoque a procesos*: actividades y recursos gestionados y controlados.
- *Liderazgo*: la alta dirección debe definir las políticas de la calidad.
- *Compromiso de los empleados*: involucración en el proceso de la calidad.
- *Mejora continua*: rendimiento de la organización.
- *Toma de decisiones objetiva*: basadas en el análisis de información.
- *Gestión de las relaciones con proveedores*: basadas en el mutuo beneficio. (Sánchez et al., 2012)

La ISO 9001:2015 está fundamentada en el rendimiento con un enfoque en lo que tiene que conseguirse en lugar de como conseguirlo, combina el enfoque basado en procesos con el pensamiento basado en el riesgo utilizando el ciclo Planificar – Hacer - Verificar - Actuar (PHVA) para gestionar cada uno de los procesos y el sistema de

gestión de la calidad como un todo, de tal manera que se impulse la mejora. ("isofocus_113.pdf," n.d., p. 9)

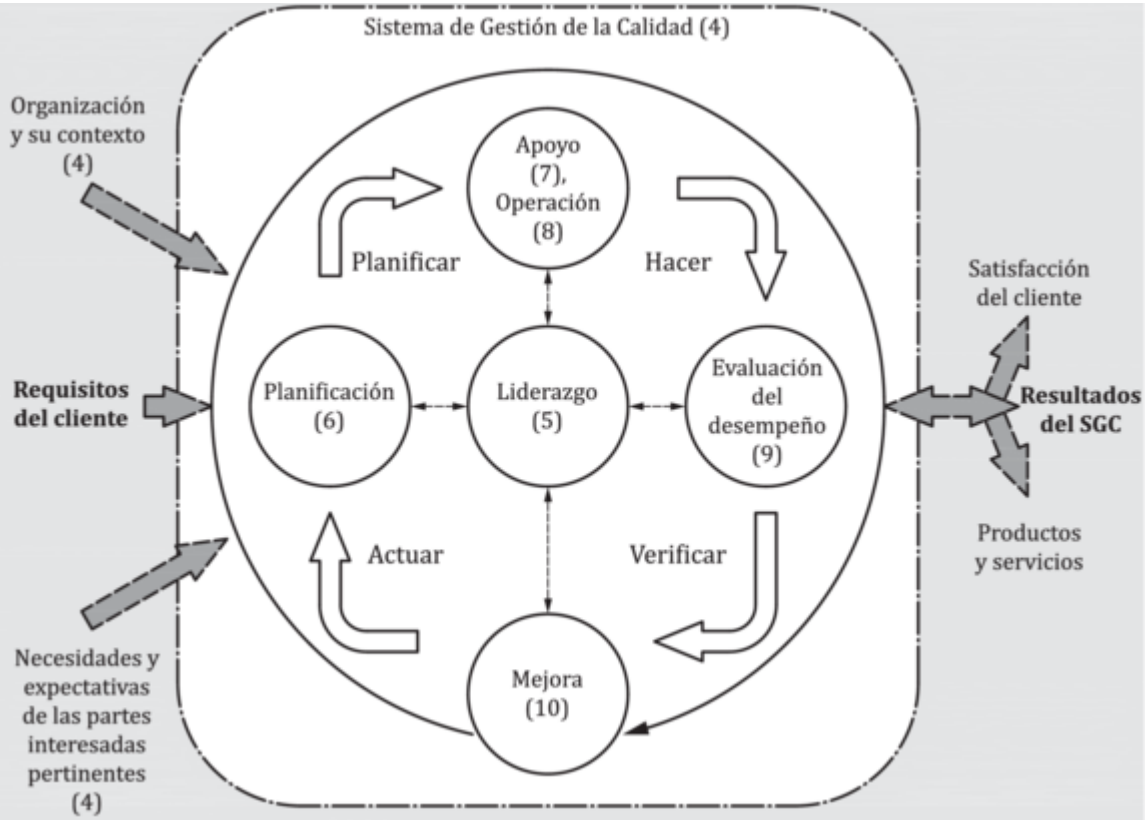


Figura 30. ISO 9001:2015 y el ciclo PHVA

Fuente: ("ISO 9001:2015(es), Sistemas de gestión de la calidad — Requisitos," n.d.)

El ciclo PHVA se describe a continuación:

- Planificar: aquí se establecen los objetivos del sistema y los recursos necesarios para obtener resultados conforme a los requisitos del cliente.
- Hacer: se realiza lo planificado.
- Verificar: se ejecuta el monitoreo y medición de los procesos y los productos y servicios resultantes.

- Actuar: se efectúan tareas para mejorar el desempeño. (“ISO 9001:2015(es), Sistemas de gestión de la calidad — Requisitos,” n.d.)

“El enfoque a procesos implica la definición y gestión sistemática de los procesos y sus interacciones, con el fin de alcanzar los resultados previstos de acuerdo con la política de la calidad y la dirección estratégica de la organización.” (“ISO 9001:2015(es), Sistemas de gestión de la calidad — Requisitos,” n.d.)

La norma ISO/IEC 90003:2014 Directrices para la aplicación de la norma ISO 9001: 2008 para el Proceso de Software, cubre una variedad de actividades del ciclo de vida del producto software que incluye: planeación, control, medición, pruebas, informes y mejora de los niveles de calidad en todo el proceso de desarrollo de software. (“ISO/IEC 90003:2014(en), Software engineering — Guidelines for the application of ISO 9001:2008 to computer software,” n.d.)

3.5 MODELO CMMI-DEV V1.3.

El modelo CMMI-DEV V1.3 (Modelo Integrado de Capacidad de Madurez para Desarrollo) suministra un estándar para la gestión de la calidad, el mejoramiento de procesos y el empleo de buenas prácticas en el desarrollo de software, se concentra en las acciones para desarrollar productos y servicios de calidad con el fin de satisfacer las necesidades de los clientes. (Software Engineering Institute, 2010)

Las organizaciones normalmente se enfocan en las tres dimensiones siguientes: personas, procedimientos y equipamiento, los **procesos** son los elementos que mantienen todo junto y permiten organizar la forma de trabajar, incorporar el conocimiento, aprovechar los recursos y analizar tendencias.

Un **enfoque de procesos** brinda la base y la seguridad necesarias para maximizar el rendimiento de las personas y la utilización de la tecnología para alcanzar los objetivos del negocio de una manera más consistente.

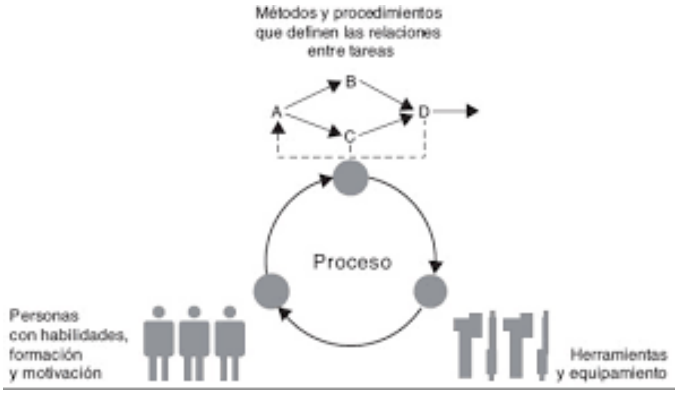


Figura 31. Las tres dimensiones críticas para las organizaciones

Fuente: (Software Engineering Institute, 2010)

El Software Engineering Institute (2010, p. 29) indica que "la calidad de un sistema o producto está muy influenciada por la calidad del proceso empleado para desarrollarlo y mantenerlo".

El *modelo CMMI-DEV se concentra en optimizar los procesos de una organización* y representa una ruta progresiva de mejora desde procesos inmaduros a procesos disciplinados con calidad. (Software Engineering Institute, 2010)

Los componentes del modelo CMMI-DEV se agrupan en tres categorías:

- Componentes requeridos: son esenciales para lograr la mejora de procesos.
    - Son las Metas específicas y Metas genéricas
- Componentes esperados: describen las actividades importantes para lograr un componente requerido.
    - Son las Prácticas específicas y Prácticas genéricas

- Componentes informativos: ayudan a comprender los componentes CMMI requeridos y esperados.
    - Son la Declaración del propósito, Notas introductorias, Áreas de proceso relacionadas, Ejemplo de productos de trabajo, Subprácticas, Elaboraciones de la práctica genérica.

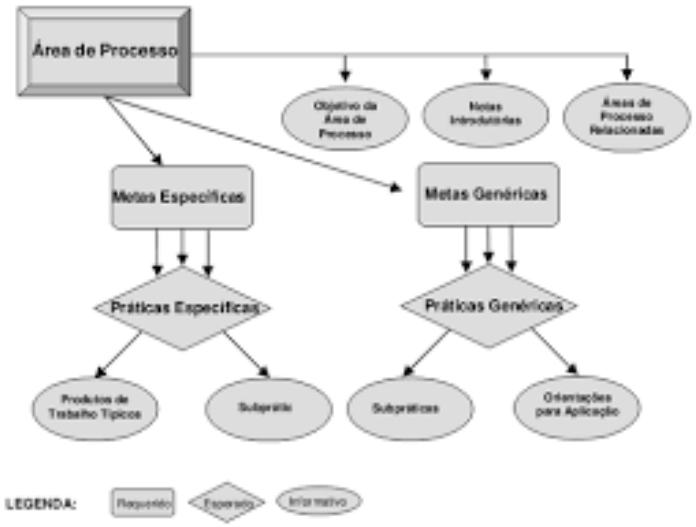


Figura 32. Componentes del modelo CMMI

Fuente: (Software Engineering Institute, 2010)

Un **Área de Proceso** es un conjunto de prácticas relacionadas que cuando se efectúan en grupo satisfacen metas importantes para mejorar esa área. En este contexto, los procesos indican “qué hacer” y no “cómo hacerlo”.(Marcal & Freitas, 2008)

CMMI brinda soporte a dos caminos de mejora de procesos usando niveles, la utilización de la representación continua permite lograr “niveles de capacidad” y la utilización de la representación por etapas permite lograr “niveles de madurez”. Los

niveles de capacidad describen el logro de la mejora de procesos de una organización en áreas de proceso individuales. Los **niveles de madurez** describen el logro de la mejora de procesos de una organización en múltiples áreas de proceso.

Tabla 5. Comparación de los niveles de capacidad y niveles de madurez

Nivel	**Representación continua** **Niveles de capacidad**	**Representación por etapas** **Niveles de madurez**
Nivel 0	Incompleto	
Nivel 1	Realizado	Inicial
Nivel 2	Gestionado	Gestionado
Nivel 3	Definido	Definido
Nivel 4		Gestionado cuantitativamente
Nivel 5		En optimización

Fuente: (Software Engineering Institute, 2010)

- Niveles de capacidad: se consigue un nivel cuando se cumplen todas las metas genéricas hasta ese nivel para un área de proceso.
    - Nivel 0. Incompleto: es un proceso que se realiza parcialmente.
    - Nivel 1. Realizado: es un proceso que cumple el trabajo necesario para producir productos.
    - Nivel 2. Gestionado: es un proceso realizado que se planifica y se hace según la política.
    - Nivel 3. Definido: es un proceso gestionado que se adecua desde procesos estándar.

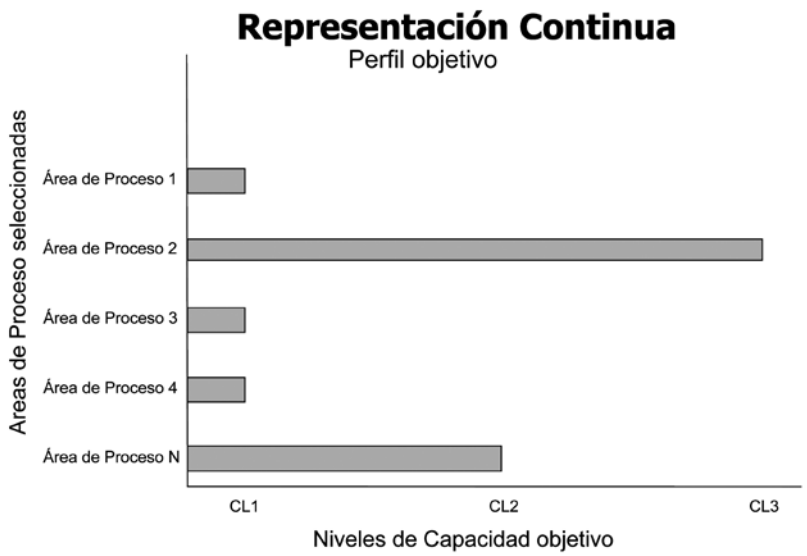


Figura 33. Representación continua

Fuente: (Software Engineering Institute, 2010)

- Niveles de madurez: constan de prácticas específicas y genéricas relacionadas para un conjunto predefinido de áreas de proceso.
    - Nivel 1. Inicial: los procesos son habitualmente ad hoc y confusos.
    - Nivel 2. Gestionado: los procesos se planifican y se realizan según las políticas.
    - Nivel 3. Definido: los procesos están bien determinados y entendidos, y se describen en estándares.
    - Nivel 4. Gestionado cuantitativamente: se implanta objetivos cuantitativos para la calidad y el rendimiento.
    - Nivel 5. En optimización: una organización mejora continuamente sus procesos apoyándose en un entendimiento cuantitativo de sus objetivos de negocio.

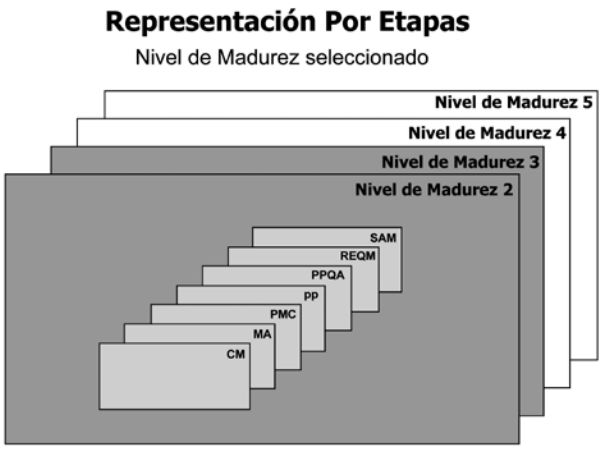


Figura 34. Representación por etapas

Fuente: (Software Engineering Institute, 2010)

Las 22 áreas de proceso se clasifican en cuatro categorías: Gestión de Proyectos, Gestión de Procesos, Soporte e Ingeniería.

Tabla 6. Áreas de proceso con categorías y madurez

Área de Proceso	Categoría	Nivel de Madurez
Definición de Procesos de la Organización (OPD)	Gestión de procesos	3
Enfoque en Procesos de la Organización (OPF)	Gestión de procesos	3
Formación en la Organización (OT)	Gestión de procesos	3
Rendimiento de Procesos de la Organización (OPP)	Gestión de procesos	4
Gestión del Rendimiento de la Organización (OPM)	Gestión de procesos	5
Gestión de Acuerdos con Proveedores (SAM)	Gestión de proyectos	2
Gestión de Requisitos (REQM)	Gestión de proyectos	2
Monitorización y Control del Proyecto (PMC)	Gestión de proyectos	2
Planificación del Proyecto (PP)	Gestión de proyectos	2
Gestión de Riesgos (RSKM)	Gestión de proyectos	3
Gestión Integrada del Proyecto (IPM)	Gestión de proyectos	3
Gestión Cuantitativa del Proyecto (QPM)	Gestión de proyectos	4
Desarrollo de Requisitos (RD)	Ingeniería	3
Integración del Producto (PI)	Ingeniería	3
Solución Técnica (TS)	Ingeniería	3
Validación (VAL)	Ingeniería	3
Verificación (VER)	Ingeniería	3
Aseguramiento de la Calidad del Proceso y del Producto (PPQA)	Soporte	2
Gestión de Configuración (CM)	Soporte	2
Medición y Análisis (MA)	Soporte	2
Análisis de Decisiones y Resolución (DAR)	Soporte	3
Análisis Causal y Resolución (CAR)	Soporte	5

Fuente: (Software Engineering Institute, 2010)

- Metas genéricas y prácticas genéricas: la institucionalización de un proceso indica que es la forma normal de trabajo en la organización y es consistente en su ejecución. El grado de institucionalización está asociado a las metas genéricas:
    - GG1 Lograr las metas específicas.
        - GP1.1 Realizar las prácticas específicas
    - GG2 Institucionalizar un proceso gestionado
        - GP2.1 Establecer una política de la organización
        - GP2.2 Planificar el proceso, está relacionada con PP.

- GP2.3 Proporcionar recursos, está relacionada con PP.
- GP2.4 Asignar responsabilidad, está relacionada con PP.
- GP2.5 Formar al personal, está relacionada con PP y OT.
- GP2.6 Controlar los productos de trabajo, está relacionada con CM.
- GP2.7 Identificar e involucrar a las partes interesadas relevantes, está relacionada con PP, PMC, IPM.
- GP2.8 Monitorizar y controlar el proceso, está relacionada con PMC y MA.
- GP2.9 Evaluar objetivamente la adherencia, está relacionada con PPQA.
- GP2.10 Revisar el estado con el nivel directivo, está relacionada con PMC.

o GG3 Institucionalizar un proceso definido

- GP3.1 Establecer un proceso definido, está relacionada con IPM y OPD.
- GP3.2 Recoger experiencias relativas al proceso, está relacionada con IPM, OPF, OPD.

4 CAPÍTULO IV: ANÁLISIS SITUACIONAL.

4.1 MISIÓN.

Actualmente, el Departamento de Desarrollo de Software de la PUCE no tiene definida una misión específica sino que comparte la visión de la Dirección de Informática de la PUCE que indica lo siguiente:

> En el marco de la misión de la PUCE, de formar íntegramente a las personas, la Dirección de Informática brinda servicios tecnológicos a nivel académico, científico y administrativo en:
>
> - Implementación de Soluciones Informáticas.
> - Consultoría informática para clientes internos
> - Apoyo y asesoría a usuarios en sus necesidades informáticas.
> - Instrucción informática para clientes internos y externos.
>
> A fin de elevar el nivel de la cultura informática propia e irradiarla hacia el ambiente externo, generar recursos para la autogestión, posicionarse como una unidad que se rige por estándares internacionales de calidad y mantener una imagen de prestigio ante la comunidad. (PUCE, 2014b)

A su vez la misión de la Dirección de Informática tiene su marco en la misión de la PUCE que se presenta a continuación:

> COMO UNIVERSIDAD:
> Considera misión propia el contribuir, de un modo riguroso y crítico, a la tutela y desarrollo de la dignidad humana y de la herencia cultural, mediante la investigación, la docencia y los diversos servicios ofrecidos a las comunidades locales, nacionales e internacionales.
> En dicha misión, asume el deber de prestar particular atención a las dimensiones éticas de todos los campos del saber y del actuar humano, tanto a nivel individual como social. En este marco propugna el respeto a la dignidad y a los derechos de la persona humana, y a sus valores trascendentes, y apoya y promueve la implantación de la justicia en todos los órdenes de la existencia.
> Goza de aquella autonomía institucional que le es necesaria para cumplir sus funciones eficazmente.

> Garantiza a sus miembros la libertad académica, salvaguardando los derechos de la persona y de la comunidad dentro de las exigencias de la verdad y del bien común.
> Dirige su actividad hacia la persona integral, para superar una formación meramente profesionalizante. Por ello trata de formar a sus miembros intelectual y moralmente, para el servicio a la sociedad.
> Examina a fondo la realidad con los métodos propios de cada disciplina académica, estableciendo después un diálogo entre las diversas disciplinas que las enriquezca mutuamente. Con ello pretende la integración del saber.
> Promueve el compromiso de todos los miembros de la comunidad universitaria para la consecución de los fines institucionales, a través del diálogo y la participación.
> COMO UNIVERSIDAD CATÓLICA
> Se inspira en los principios cristianos; propugna la responsabilidad del ser humano ante Dios, el respeto a la dignidad y derechos de la persona humana y a sus valores trascendentales; apoya y promueve la implantación de la justicia en todos los órdenes de la existencia; propicia el diálogo de las diversas disciplinas con la fe, la reflexión sobre los grandes desafíos morales y religiosos, y la praxis cristiana.
> COMO UNIVERSIDAD DIRIGIDA POR LA COMPAÑIA DE JESÚS
> Promueve la implantación y el desarrollo de la pedagogía ignaciana en todas sus actividades académicas. (PUCE, 2008, p. 5)

4.2 VISIÓN DE FUTURO.

De igual forma el Departamento de Desarrollo de Software de la PUCE no tiene definida una visión específica, sino que comparte la visión de la Dirección de Informática de la PUCE que dice así:

> La Dirección de Informática quiere ser la unidad líder y modelo en la provisión de excelentes servicios que apoyen los campos de acción de la Pontificia Universidad Católica del Ecuador (académicos, científicos, de investigación y de gestión), mediante tecnología de vanguardia, sustentada por un recurso humano altamente calificado y comprometido. (PUCE, 2014b)

De la misma manera la visión de la Dirección de Informática tiene su marco de acción en la visión de la PUCE:

> En los próximos años, la PUCE, fundamentada en el pensamiento y en las directrices pedagógicas ignacianas, se consolidará como un sistema nacional integrado competitivo y autosostenible, con infraestructura tecnológica de vanguardia.
> Será reconocida por su gestión ética en servicio de la comunidad, y por su estructura académica moderna para la formación de profesionales con responsabilidad social.
> Será también reconocida por los resultados de la investigación científica desarrollada en sus unidades académicas, por realizar su gestión con el apoyo de un sistema técnico, innovador y efectivo, con procesos eficientes y recursos humanos capacitados y comprometidos con la misión institucional. (PUCE, 2008, p. 6)

4.3 VALORES INSTITUCIONALES.

Los valores institucionales de la Pontificia Universidad Católica del Ecuador que son compartidos también por la Dirección de Informática y por el Departamento de Desarrollo de Software son los siguientes:

- Justicia
- Integridad
- Responsabilidad Social
- Equidad
- Innovación
- Igualdad de oportunidades
- Diversidad
- Reconocimiento del mérito individual
- Sentido de pertenencia a la institución
- Orientación de servicio
- Mejoramiento continuo
- Trabajo en equipo
- Puntualidad
- Disciplina. (PUCE, 2008, p. 7)

4.4 OBJETIVOS ESTRATÉGICOS.

De acuerdo con el Plan Estratégico de Desarrollo Institucional 2014 - 2018 de la PUCE los siguientes son los Objetivos Estratégicos:

1. En los próximos 5 años, ofrecer a la sociedad profesionales integrales: conscientes, competentes, compasivos y comprometidos en el conocimiento de nuestra realidad, a través del fortalecimiento académico, investigativo y de vinculación con la colectividad.
2. En los próximos 5 años, perfeccionar el sistema de gestión integral de la PUCE a través de la implementación de buenas prácticas de gestión de infraestructura física, tecnológica, educativa y financiera, que apoyen de manera transversal y eficiente a la comunidad universitaria en sus tareas fundamentales.
3. Implementar en los próximos 5 años un sistema de gestión administrativa y laboral que asegure la eficiencia en el desempeño del talento humano de la PUCE, a través del fortalecimiento de competencias, desarrollo profesional y seguimiento continuo que propendan al desarrollo institucional.
4. Para el año 2018, implementar estándares de evaluación y acreditación nacional e internacional, haciendo uso de las mejores metodologías de gestión universitaria, en búsqueda del mejoramiento y crecimiento continuo de la PUCE.
5. En los próximos 5 años aprovechar la cooperación nacional e internacional para la consecución de los objetivos de la PUCE. (Pontificia Universidad Católica del Ecuador, 2014)

En el marco del Despliegue de los Objetivos Estratégicos de la PUCE, la Dirección de Informática y en concreto el Departamento de Desarrollo de Software tiene la responsabilidad de colaborar en la ejecución de los siguientes Objetivos Tácticos, Estrategias y Acciones:

- **OT 1.1**: Consolidar en los siguientes 5 años el modelo educativo de la PUCE como sustento del quehacer académico de la PUCE.
  - E 1.1.2: Optimizar el proceso de gestión académica de pre y posgrado en la PUCE.
    - *A 1.1.2.1*: Diagnóstico y propuesta de estandarización de los procesos de admisión, evaluación docente y de titulación en las carreras y programas de la PUCE.
- **OT 1.2**: Incrementar en los siguientes 2 años la producción científica de la PUCE, con trabajos y publicaciones que se articulen a las necesidades del país, y

que sean reconocidas nacional e internacionalmente en el medio académico y productivo.

- E 1.2.3: Gestionar el seguimiento de la investigación en la PUCE.
  - *A 1.2.3.1:* Identificación y control de proyectos de investigación en curso.

- **OT 1.3**: Incrementar en los siguientes 2 años la participación de la comunidad universitaria en los procesos de: acción social, pasantías, prácticas preprofesionales y servicios de extensión del sistema de vinculación con la colectividad de la PUCE.
  - E 1.3.1: Articular los procesos de vinculación con la colectividad identificados en la PUCE.
    - *A 1.3.1.1:* Estructuración de un sistema sólido para el impulso, planificación, y evaluación de programas y proyectos de vinculación.
- **OT 2.1**: Gestionar en los próximos 2 años la infraestructura física, tecnológica y educativa de la PUCE de tal manera que cumpla eficientemente con estándares nacionales e internacionales.
  - E 2.1.1: Adecuar la infraestructura física, tecnológica y educativa en función de los objetivos estratégicos en toda la PUCE.
    - *A 2.1.1.3:* Fortalecimiento bibliográfico relacionado a los programas de grado y posgrado.
- **OT 2.2**: Invertir en los siguientes 5 años eficientemente los recursos financieros en las áreas estratégicas de la PUCE.
  - E 2.2.1: Rediseñar los procesos financieros de soporte en función de las áreas estratégicas de la PUCE.
    - *A 2.2.1.1:* Elaboración de un estudio técnico que permita establecer los costos por carrera, profesor y estudiante en la PUCE.
- **OT 3.2**: Optimizar en los siguientes 2 años el desempeño del talento humano a través de una adecuada selección, capacitación, evaluación y desarrollo profesional que aporte en los diferentes ámbitos misionales y de soporte a la PUCE.
  - E 3.2.2: Articular subprocesos de talento humano a las áreas estratégicas de la PUCE.
    - *A 3.2.2.1:* Implementación del escalafón docente.
- **OT 4.1**: Desarrollar en los próximos 5 años en la PUCE carreras y programas pertinentes a la realidad nacional.
  - E 4.1.1: Investigar y desarrollar nuevas carreras y programas académicos para la PUCE.
    - *A 4.1.1.2:* Desarrollar un programa de promoción y mercadeo de la oferta académica existente en la PUCE.
- **OT 4.3**: Lograr en los próximos 5 años que la comunicación interna y externa de la PUCE sea eficaz.
  - E 4.3.1: Articular las relaciones públicas y la comunicación interna y externa a las áreas estratégicas de la PUCE.
    - *A 4.3.1.2:* Diseñar un sistema de comunicación intrainstitucional que articule los procesos misionales, estratégicos y de soporte de la PUCE. (Ponce León, 2016, p. 22)

4.5 ESTRUCTURA INTERNA ORGANIZACIONAL.

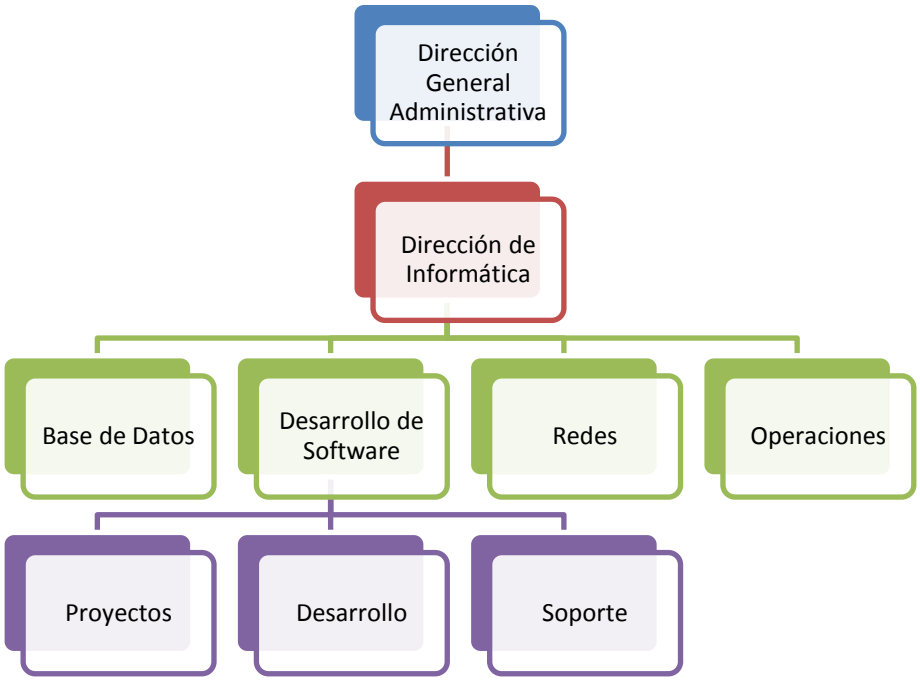


Figura 35. Organigrama Estructural

Fuente: El autor de la tesis

La Dirección de Informática (DI) es la unidad de servicios autorizada para proporcionar a la PUCE las soluciones informáticas y los recursos tecnológicos necesarios para satisfacer las necesidades y apoyar a la gestión de la comunidad universitaria. Depende de la Dirección General Administrativa y su función principal es garantizar el uso y funcionamiento adecuado de la tecnología en la PUCE, así como asesorar a las instancias docentes y administrativas en la gestión de la información. La DI tiene a su cargo definir todas las políticas tecnológicas, delinear el desarrollo informático y liderar proyectos de tecnologías de la información y la comunicación. (PUCE, 2014b)

Se compone de cuatro departamentos:

- **Departamento de Base de Datos**: tiene la responsabilidad de la administración centralizada, contingencia externa y la implementación de sistemas de administración y monitoreo de las bases de datos de la PUCE.
- **Departamento de Desarrollo de Software:** es responsable del análisis, diseño, codificación, pruebas, despliegue y mantenimiento de todas las aplicaciones informáticas de la PUCE.
    - Desde Enero del 2014 el departamento se organiza internamente en tres grupos de trabajo: proyectos, desarrollo y soporte.
- **Departamento de Redes:** administra, monitorea y respalda la información de los servidores de la PUCE, del correo electrónico y gestiona la seguridad informática.
- **Departamento de Operaciones:** es responsable de dar soporte técnico a los usuarios, administra el centro de cómputo, administra el uso de software legal y el antivirus, define los estándares de hardware y software. (PUCE, 2014b)

4.6 PRODUCTOS Y SERVICIOS.

El Departamento de Desarrollo de Software cuenta con muchas aplicaciones desarrolladas in-house y otras adquiridas a terceros, las cuales administra y brinda el mantenimiento funcional necesario.

Los siguientes son algunos de los productos y servicios más importantes que provee el departamento:

- Sistemas para la Gestión Académica y de los estudiantes
    - Sistema Académico UXXI
    - Sistema UXXI Investigación
    - Gestión del Proceso de Inscripción
    - Seguimiento académico (Tutorías)
    - Planificación y registro de la Programación académica
    - Registro de Disertaciones de grado
    - Gestión de los programas microcurriculares
    - Seguimiento y evaluación de programas microcurriculares
    - Hoja de actualización de datos personales vía web
    - Registro de actividades académicas del profesor
    - Sistema para registro de información de posgrados
    - Gestión de la Biblioteca y préstamo de libros
    - Gestión de los centros de documentación de la PUCE
    - Gestión de usuarios y aulas virtuales (Moodle)

- Registro y carnetización de egresados PUCE
- Generación de informes mediante Business Objects
- Encuestas de satisfacción de los servicios
- Sistema de Indicadores Estadísticos PUCE (SIEPUCE)
- Gestión de solicitudes de certificados de los estudiantes
- Evaluación Docente General
- Evaluación Docente Específica
- Medición de las habilidades gerenciales

- Sistemas para la Gestión Administrativa y Comunicación Institucional
  - Gestión de Recursos Humanos y Nómina (ADAM)
  - Gestión de parqueaderos
  - Gestión documental para el proceso de acreditación
  - Sistema de retroalimentación a directivos
  - Evaluación del Clima laboral
  - Emisión de carnés
  - Control automático de asistencias del Centro Médico y de la DI
  - Seguimiento y control de ingreso de los profesores en aulas de clase
  - Sistema de Pantallas informativas
  - Sistema de despacho de correspondencia
  - Gestión del Archivo central
  - Reservas de uso de Playa ancha
  - Inventarios de materiales de promoción DGE
  - Evaluación de la atención del Centro Médico
  - Integración de los sistemas de la PUCE (PISP)

    - Consulta del avance de los planes de mejora de las sedes
    - Portal web de la PUCE
    - Servicios a través de la Intranet
    - Portal del ISP
    - Desarrollo e implementación de páginas web
- Sistemas para la gestión de la Dirección de Informática
    - Administración de servidores y aplicativos
    - Sistema de Administración de Recursos Tecnológicos (SARS)
    - Despliegue de Pop ups en la intranet
- Sistemas para la Gestión Financiera
    - Solicitud de trámites para la DGF
    - Sistema financiero BAAN ERP
    - Ingreso de datos informativos de proveedores
    - Participación en el proyecto de implementación de facturación electrónica. (PUCE, 2014a, p. 85)

4.7 ANÁLISIS FODA.

El Departamento de Desarrollo de Software de la PUCE presenta las siguientes fortalezas, debilidades, oportunidades y amenazas que serán analizadas para proponer acciones de mejora considerando siempre que estén alineadas a los objetivos estratégicos de la PUCE.

Fortalezas:

- Buen ambiente laboral
- Equipamiento de última generación
- Experiencia del recurso humano
- Desarrollo de aplicaciones personalizadas
- Buen análisis de requerimientos
- Buena comunicación con el cliente

Debilidades:

- Falta de comunicación interna
- Falta un plan de capacitación en nuevas tecnologías
- Falta de distribución equitativa del trabajo
- Falta de incentivos para realizar investigación
- Falta de autogestión para conseguir recursos económicos
- Falta de divulgación de los servicios prestados

- Falta de definición formal de procesos, roles y responsabilidades
- No hay un plan estratégico específico para el departamento

Oportunidades:

- Venta de servicios de asesoría tecnológica
- Venta de servicios de desarrollo de software
- Institucionalización de los procesos del área
- Presentación de proyectos en la PUCE

Amenazas:

- Contratación de empresas externas que ofrecen servicios de desarrollo de software similares
- Problemas financieros de la PUCE
- El rechazo al cambio de los usuarios ante nuevos sistemas
- La PUCE es burocrática

Estrategia ofensiva (FO):

- Aprovechar la buena comunicación con el cliente para vender servicios de asesoría tecnológica
- *Valerse de la experiencia del recurso humano para institucionalizar los procesos del área.*

Estrategia defensiva (FA):

- Cultivar el buen análisis de requerimientos para evitar el rechazo al cambio de los usuarios.
- Manejar la buena comunicación con el cliente para evitar la contratación de empresas externas.

Estrategia de reorientación (DO):

- Brindar capacitación a los técnicos en nuevas tecnologías para potenciar la venta de servicios de desarrollo de software.
- *Definir formalmente procesos, roles y responsabilidades para fomentar la mejora de los procesos del área.*

Estrategia de supervivencia (DA):

- Implementar mecanismos de divulgación de los servicios prestados para que las unidades no contraten servicios de desarrollo de software externo.
- Autogestionar la venta de servicios y productos tecnológicos para conseguir recursos económicos.

4.8 ANÁLISIS DEL NIVEL DE CONFORMIDAD DE LOS PROCESOS ACTUALES CON LOS COMPONENTES REQUERIDOS DEL MODELO CMMI-DEV.

La metodología de desarrollo actual que se utiliza en el departamento de Desarrollo de Software de la PUCE es el modelo en Cascada que tiene un enfoque secuencial con un flujo lineal en donde las fases del ciclo de vida del desarrollo de software se realizan una a continuación de la otra hasta obtener el producto terminado, el problema con esta metodología es que para iniciar el proyecto todos los requisitos deben estar completos y la puesta en producción del producto terminado demora demasiado tiempo, a esto se suma la falta de procedimientos definidos, aprobados y documentados, lo que provoca inestabilidad en la entrega de los productos del trabajo software y datos históricos inexistentes o incompletos.

Las metodologías ágiles de software eliminan estos problemas ya que tienen un enfoque iterativo e incremental que permite:

- la entrega de software funcional de forma temprana y continua,
- el cambio en los requerimientos aún después de comenzado el proyecto,
- el trabajo conjunto entre el cliente y los desarrolladores durante el proyecto.(Potter, 2010)

Una organización ágil de software puede implementar Scrum correctamente, pero fallar en obtener los beneficios reales debido a la falta de consistencia y la insuficiente ejecución de procesos de ingeniería y de gestión; CMMI puede ayudar a dichas

organizaciones a institucionalizar metodologías ágiles más disciplinadamente y entender los procesos a abordar con mayor prioridad. A la inversa, una organización puede cumplir con CMMI, pero fallar en alcanzar un rendimiento óptimo debido a una inadecuada implementación de los procesos, Scrum y otras metodologías ágiles de software pueden guiar a dichas organizaciones hacia una implementación más eficiente y adaptable de los requerimientos de CMMI.(Jakobsen & Johnson, 2008)

Los proyectos que combinan metodologías ágiles de software con CMMI son más exitosos en producir software de alta calidad que satisfacen las necesidades cambiantes de los clientes de manera más efectiva y a un ritmo más rápido. (Sutherland, Jakobsen, & Johnson, 2008)

Según Sutherland (2008) una organización CMMI nivel 5 ha reducido el trabajo redundante en un 42%, las desviaciones en las estimaciones de planificación han disminuido a menos del 10%, y el 92% de los hitos son entregados a tiempo. En resumen, dicha organización es capaz de entregar lo que el cliente desea, en tiempo, costo y calidad usando el 69% del esfuerzo comparado con una organización CMMI nivel 1. Adicionalmente, si se reemplazan algunos procesos con la metodología ágil Scrum baja el costo del desarrollo en 34%, disminuye la sobrecarga de procesos en 50%, y bajan los defectos en un 40%.

En consonancia con lo expuesto anteriormente, para definir el marco de trabajo para el análisis de los procesos actuales y complementar el trabajo de revisión bibliográfica realizada, se entrevistó a dos profesores expertos en Ingeniería de Software, Procesos y

Calidad de la Escuela de Sistemas de la PUCE: el Ing. Oswaldo Espinosa y el Ing. Fabián de la Cruz, los cuales confirmaron mi idea de seguir la línea del desarrollo ágil de software junto con el modelo de mejoramiento de procesos CMMI-DEV.

Con el propósito de obtener una visión real y objetiva de la manera en que se llevan a cabo los procesos de desarrollo de software en el departamento, se realizó una encuesta anónima electrónica a 15 analistas de sistemas del departamento de Desarrollo de Software de la PUCE utilizando una encuesta creada con “Google Forms” que contiene 22 preguntas, una pregunta por cada área de proceso del modelo CMMI-DEV V1.3. (ver Anexo 1. “Encuestas y Resultados”)

El modelo CMMI-DEV V1.3 tiene un conjunto de componentes requeridos: metas genéricas y metas específicas, que son esenciales para lograr la mejora de procesos en el área de proceso dada.

A continuación, se detalla el nivel de conformidad de los procesos actuales versus los componentes requeridos del modelo CMMI-DEV V1.3.

Tabla 7. Nivel de conformidad de los componentes requeridos del modelo

Áreas de Proceso y Metas	Nivel de Madurez	Categoría	Nivel de Conformidad
Planificación del Proyecto (PP)	**2**	Gestión de Proyectos	
PP SG1 Establecer las estimaciones			Rara vez 69%
PP SG2 Desarrollar un plan de proyecto			Rara vez 69%
PP SG3 Obtener el compromiso con el plan			Rara vez 69%
Monitorización y Control del Proyecto (PMC)	**2**	Gestión de Proyectos	
PMC SG1 Monitorizar el proyecto frente al plan			Rara vez 69%
PMC SG2 Gestionar las acciones correctivas hasta su cierre			Rara vez 69%
Gestión de Acuerdos con Proveedores (SAM)	**2**	Gestión de Proyectos	
SAM SG1 Establecer acuerdos con proveedores			Frecuentemente 54%
SAM SG2 Satisfacer los acuerdos con los proveedores			Frecuentemente 54%
Gestión de Requisitos (REQM)	**2**	Gestión de Proyectos	
REQM SG1 Gestionar los requisitos			Rara vez 46%
Aseguramiento de la Calidad del Proceso y del Producto (PPQA)	**2**	Soporte	
PPQA SG1 Evaluar objetivamente los procesos y productos de trabajo			Nunca 31%
PPQA SG2 Proporcionar una visión objetiva			Nunca 31%
Gestión de Configuración (CM)	**2**	Soporte	
CM SG1 Establecer las líneas base			Rara vez 38%
CM SG2 Seguir y controlar los cambios			Rara vez 38%
CM SG3 Establecer la integridad			Rara vez 38%
Medición y Análisis (MA)	**2**	Soporte	
MA SG1 Alinear las actividades de medición y análisis			Rara vez 62%
MA SG2 Proporcionar los resultados de la medición			Rara vez 62%
Definición de Procesos de la Organización (OPD)	**3**	Gestión de procesos	
OPD SG1 Establecer los activos de proceso de la organización			Rara vez 69%
Enfoque en Procesos de la Organización (OPF)	**3**	Gestión de procesos	
OPF SG1 Determinar las oportunidades de mejora de procesos			Rara vez 62%
OPF SG2 Planificar e implementar las acciones			Rara vez 62%

de proceso			
OPF SG3 Desplegar los activos de proceso de la organización e incorporar las experiencias			Rara vez 62%
Formación en la Organización (OT)	**3**	Gestión de procesos	
OT SG1 Establecer una capacidad de formación de la organización			Rara vez 69%
OT SG2 Proporcionar formación			Rara vez 69%
Gestión de Riesgos (RSKM)	**3**	Gestión de Proyectos	
RSKM SG1 Preparar la gestión de riesgos			Rara vez 38%
RSKM SG2 Identificar y analizar los riesgos			Rara vez 38%
RSKM SG3 Mitigar los riesgos			Rara vez 38%
Gestión Integrada del Proyecto (IPM)	**3**	Gestión de Proyectos	
IPM SG1 Utilizar el proceso definido del proyecto			Rara vez 77%
IPM SG2 Coordinar y colaborar con las partes interesadas relevantes			Rara vez 77%
Desarrollo de Requisitos (RD)	**3**	Ingeniería	
RD SG1 Desarrollar los requisitos del cliente			Frecuentemente 61%
RD SG2 Desarrollar los requisitos de producto			Frecuentemente 61%
RD SG3 Analizar y validar los requisitos			Frecuentemente 61%
Integración del Producto (PI)	**3**	Ingeniería	
PI SG1 Prepararse para la integración del producto			Frecuentemente 69%
PI SG2 Asegurar la compatibilidad de las interfaces			Frecuentemente 69%
PI SG3 Ensamblar los componentes de producto y entregar el producto			Frecuentemente 69%
Solución Técnica (TS)	**3**	Ingeniería	
TS SG1 Seleccionar soluciones de componentes de producto			Frecuentemente 77%
TS SG2 Desarrollar el diseño			Frecuentemente 77%
TS SG3 Implementar el diseño del producto			Frecuentemente 77%
Validación (VAL)	**3**	Ingeniería	
VAL SG1 Preparar la validación			Frecuentemente 54%
VAL SG2 Validar el producto o los componentes de producto			Frecuentemente 54%
Verificación (VER)	**3**	Ingeniería	

VER SG1 Preparar la verificación			Rara vez 54%
VER SG2 Realizar las revisiones entre pares			Rara vez 54%
VER SG3 Verificar los productos de trabajo seleccionados			Rara vez 54%
Análisis de Decisiones y Resolución (DAR)	**3**	Soporte	
DAR SG1 Evaluar las alternativas			Rara vez 54%
Rendimiento de Procesos de la Organización (OPP)	**4**	Gestión de procesos	
OPP SG1 Establecer las líneas base y los modelos de rendimiento			Rara vez 62%
Gestión Cuantitativa del Proyecto (QPM)	**4**	Gestión de Proyectos	
QPM SG1 Preparar la gestión cuantitativa			Rara vez 54%
QPM SG2 Gestionar el proyecto cuantitativamente			Rara vez 54%
Gestión del Rendimiento de la Organización (OPM)	**5**	Gestión de procesos	
OPM SG1 Gestionar el rendimiento del negocio			Nunca 77%
OPM SG2 Seleccionar las mejoras			Nunca 77%
OPM SG3 Desplegar las mejoras			Nunca 77%
Análisis Causal y Resolución (CAR)	**5**	Soporte	
CAR SG1 Determinar las causas de los resultados seleccionados			Rara vez 46%
CAR SG2 Tratar las causas de los resultados seleccionados			Rara vez 46%
GG1 Lograr las metas específicas	**2**		Rara vez 57%
GG2 Institucionalizar un proceso gestionado	**2**		Rara vez 57%
GG3 Institucionalizar un proceso definido	**3**		Rara vez 59%

Fuente: El autor de la tesis

Como resultado de las encuestas se presenta la siguiente figura y la siguiente tabla que muestran el porcentaje de conformidad de los componentes requeridos agrupados por su Nivel de Madurez correspondiente.

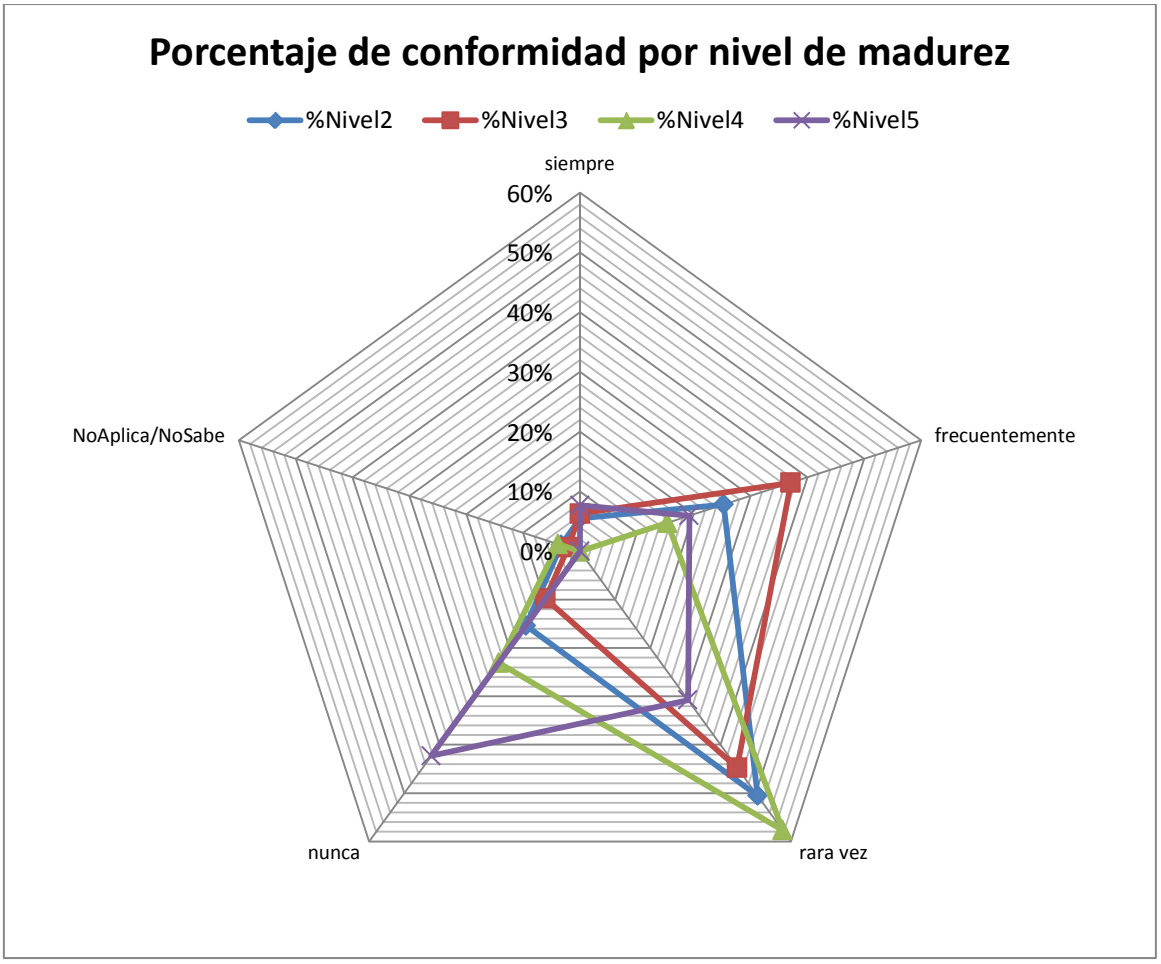


Figura 36. Porcentaje de conformidad por nivel de madurez

Fuente: El autor de la tesis

Tabla 8. Porcentaje de conformidad por nivel de madurez

Opciones	Nivel2	Nivel3	Nivel4	Nivel5
siempre	5%	6%	0%	8%
frecuentemente	25%	37%	15%	19%
rara vez	51%	45%	58%	31%
nunca	15%	10%	23%	42%
NoAplica/NoSabe	3%	2%	4%	0%

Fuente: El autor de la tesis

Se ha encontrado que los procesos del nivel 2 de madurez se cumplen solamente “rara vez” según el 51% de los encuestados, siendo el proceso Planificación del Proyecto (PP) el que más aporta a esta medida, mientras que el 25% de los encuestados opinan que estos procesos se ejecutan “frecuentemente”.

El 45% de los encuestados indican que los procesos del nivel 3 de madurez se cumplen “rara vez”, siendo el proceso Gestión Integrada del Proyecto (IPM) el que más aporta a esta medida, también el 37% de los encuestados opinan que estos procesos se ejecutan “frecuentemente”.

El 58% de las personas encuestadas dijeron que los procesos del nivel 4 de madurez se ejecutan solamente “rara vez”, siendo el proceso Rendimiento de Procesos de la Organización (OPP) el que más aporta a esta medida, además el 23% de los encuestados opinan que estos procesos no se ejecutan “nunca”.

El 42% de las respuestas indican que los procesos del nivel 5 de madurez se no practican “nunca”, siendo el proceso Gestión de Rendimiento de la Organización (OPM) el que más aporta a esta medida, mientras que el 31% de los encuestados opinan que estos procesos se hacen “rara vez”.

A partir de los datos hallados, se puede distinguir que la mayor parte de los procesos de los niveles de madurez 2 y 4 se realizan solamente “rara vez”, lo que muestra una debilidad en la gestión, documentación y medición cuantitativa de los proyectos software.

De la misma manera, se puede observar que la mayoría de los procesos del nivel de madurez 5 no se realizan “nunca”, lo que evidencia una debilidad grande en la optimización continua de los procesos software.

También, se puede ver que los procesos del nivel 3 de madurez se desempeñan en su mayoría solamente “rara vez”, lo que indica una debilidad en la definición y aprobación de procesos estándar que sirvan como base para el desarrollo de los proyectos software.

Por lo tanto, de las debilidades detectadas se concluye que la mayoría de los procesos actuales del departamento de Desarrollo de Software de la PUCE están en el nivel de madurez inicial, en el cual los procesos son ad hoc sin un soporte estable y su cumplimiento depende del profesionalismo y actitud de los empleados, que con frecuencia se exceden en el presupuesto y en los plazos disponibles.

Igualmente, como resultado de las encuestas se presenta la siguiente figura y la siguiente tabla que muestran el porcentaje de conformidad de los componentes requeridos agrupados por su Categoría de proceso correspondiente.

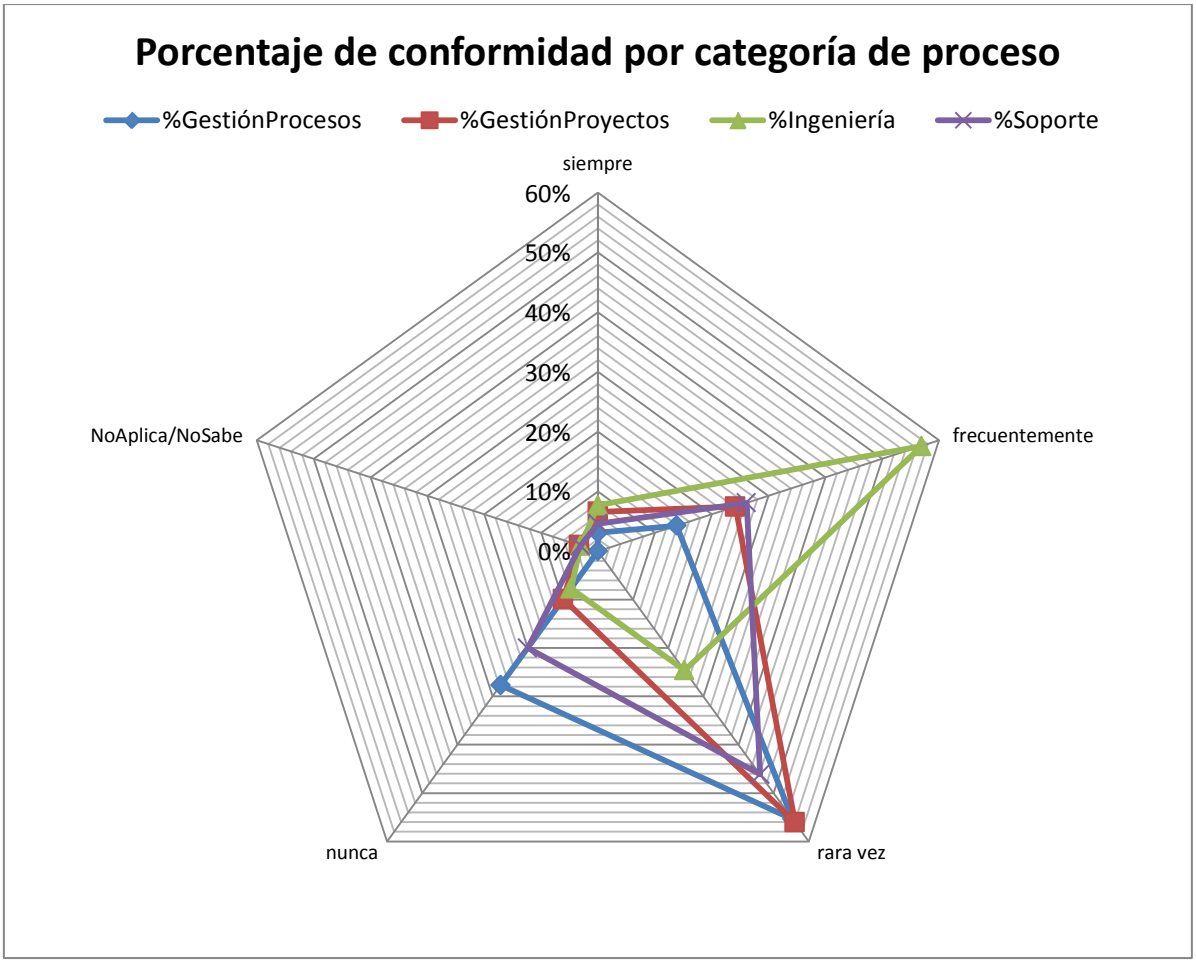


Figura 37. Porcentaje de conformidad por categoría de procesos

Fuente: El autor de la tesis

Tabla 9. Porcentaje de conformidad por categoría de procesos

Opciones	Gestión de Procesos	Gestión de Proyectos	Ingeniería	Soporte
siempre	3%	7%	8%	5%
frecuentemente	14%	24%	57%	26%
rara vez	55%	56%	25%	46%
nunca	28%	10%	8%	20%
NoAplica/NoSabe	0%	3%	3%	3%

Fuente: El autor de la tesis

Para los procesos de la categoría Ingeniería se encontró que el 57% de los encuestados indican que se cumplen “frecuentemente”, siendo el proceso Solución Técnica (TS) el que más aporta a esta medida, también el 25% de los encuestados opinan que estos procesos se ejecutan solamente “rara vez”.

Se ha encontrado también que el 56% de las personas encuestadas dijeron que los procesos de la categoría Gestión de Proyectos se ejecutan “rara vez”, siendo el proceso Gestión Integrada del Proyecto (IPM) el que más aporta a esta medida, además el 24% de los encuestados opinan que estos procesos se ejecutan “frecuentemente”.

Los procesos de la categoría Gestión de Procesos se efectúan solamente “rara vez” según el 55% de los encuestados, siendo el proceso Definición de los Procesos de la Organización (OPD) el que más aporta a esta medida, también el 28% de los encuestados opinan que estos procesos no se ejecutan “nunca”.

El 46% de las respuestas indican que los procesos de la categoría Soporte se practican “rara vez”, siendo el proceso Medición y Análisis (MA) el que más aporta a esta medida, mientras que el 26% de los encuestados consideran que estos procesos se realizan “frecuentemente”.

De los datos encontrados, se puede notar que la mayor parte de los procesos de las categorías de Gestión de Proyecto y Gestión de Procesos se realizan “rara vez”, lo que indica una debilidad en la planificación, monitorización y control de los proyectos software, además de la falta de definición e institucionalización de procesos estándar.

Se puede apreciar que la mayoría de los procesos de la categoría Ingeniería se efectúan “frecuentemente”, lo que evidencia una fortaleza en el análisis y diseño de requisitos, la

construcción del código de los componentes software, y la integración, validación y verificación de los productos software entregables.

Además, se puede observar que los procesos de la categoría Soporte se realizan en su mayoría “rara vez”, lo que muestra una debilidad en el aseguramiento de la calidad tanto del proceso como del producto y la falta de gestión en la configuración del producto software.

Por consiguiente, de las fortalezas detectadas se concluye que los procesos involucrados directamente en la Ingeniería del producto software se están llevando a cabo frecuentemente, pero no con el nivel de madurez necesario y se pueden volver fácilmente inestables y caóticos en momentos de presión, al no contar consistentemente con el sustento sólido de los procesos de Soporte, ni la guía y control de los procesos de Gestión.

4.8.1 Identificación de oportunidades de mejora

En base a todo el análisis previo, se identifica que un buen inicio para el diseño e implantación del sistema de gestión de la calidad y mejoramiento continuo de los procesos del departamento de Desarrollo de Software de la PUCE, es el estudio y definición detallada de los procesos para las siete áreas de proceso del nivel de madurez 2 del modelo CMMI-DEV con un enfoque en metodologías ágiles de desarrollo de software, de tal manera que se cuente con una plataforma estable sobre la cual se desarrollaran adecuadamente los demás procesos.

En el nivel de madurez 2 (Gestionado) los proyectos se realizan y gestionan siguiendo planes documentados, los procesos se planifican y ejecutan acorde con las políticas de la organización, se dispone de personal calificado y recursos apropiados, se involucra a las partes interesadas relevantes, se monitorizan y controlan los productos de trabajo, se evalúa la adherencia de las actividades a las descripciones de procesos, la dirección puede revisar los productos de trabajo en puntos predefinidos, todo esto ayuda para que los procesos se mantengan organizados aún en períodos bajo presión. (Torrecilla, 2012)

Las metodologías ágiles de desarrollo de software como Scrum proveen buenas prácticas de gestión e ingeniería, y en conjunto con CMMI que provee disciplina y consistencia, se puede alcanzar una sinergia positiva para alcanzar el nivel de madurez 2 de CMMI, que ayuda a organizaciones de nivel medio a tomar ventaja de técnicas más ordenadas y flexibles. (Diaz, Garbajosa, & Calvo-Manzano, 2009)

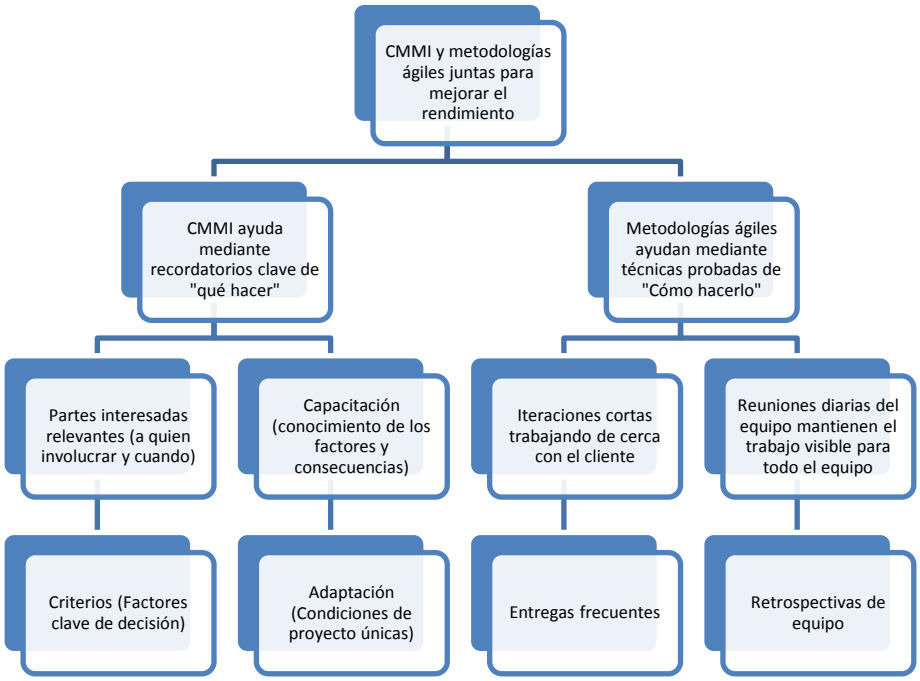


Figura 38. Metodologías ágiles y CMMI juntos para mejorar el rendimiento

Fuente: (McMahon, 2011)

A partir de la implantación de las buenas prácticas del modelo CMMI-DEV con un enfoque ágil para las áreas de proceso del nivel de madurez 2, se puede solicitar a los organismos acreditadores una evaluación formal SCAMPI (Método de evaluación estándar de CMMI para la mejora de procesos) para obtener la certificación correspondiente, y a futuro conseguir niveles de madurez mayores.

5 CAPÍTULO V: DISEÑO DEL SISTEMA DE GESTIÓN DE LA CALIDAD.

5.1 ENFOQUE METODOLÓGICO PARA EL DISEÑO DE UN SGC BAJO EL MODELO CMMI-DEV V1.3 UTILIZANDO TÉCNICAS DE DESARROLLO ÁGIL DE SOFTWARE.

El enfoque metodológico para el diseño del Sistema de Gestión de la Calidad sigue el enfoque de mejora del ciclo de Deming (PHVA) que indica que un sistema debe seguir las siguientes fases: Planificar, Hacer, Verificar y Actuar.

El hecho de que la metodología ágil Scrum y CMMI tengan raíces comunes en el trabajo de Deming ayuda en la integración de dichos modelos, dado que Deming ya tenía la idea de que los elementos de gestión duros (CMMI) y los elementos de gestión suaves (Scrum) pueden ser combinados para lograr el máximo rendimiento de los procesos del negocio. (Preis, 2012)

En la fase de Planificación se ha realizado el análisis de los procesos actuales versus los requeridos por el modelo CMMI-DEV V1.3, se ha determinado el nivel de conformidad de los procesos actuales versus los componentes requeridos del modelo CMMI-DEV y se han identificado las oportunidades de mejora.

En la fase de Hacer se obtendrá el compromiso del Jefe del Departamento de Desarrollo de Software como el patrocinador del proceso de mejora basado en CMMI-DEV, se definirán la política y los objetivos de calidad, y se detallará para cada proceso del nivel 2 de madurez: el propósito, declaración de la política, roles y responsabilidades, actividades requeridas, medidas, controles y productos de trabajo.

En la fase de Verificación se deberán realizar evaluaciones al sistema de gestión de la calidad utilizando el método de evaluación SCAMPI. Esta fase no forma parte del alcance del presente estudio.

En la fase de Actuación se deberán acoger las observaciones y recomendaciones de mejora de procesos, resultado de las evaluaciones al sistema de gestión de la calidad. Esta fase no forma parte del alcance del presente estudio.

5.2 CARTA DE COMPROMISO DE LA DIRECCIÓN CON EL SISTEMA DE GESTIÓN DE LA CALIDAD.

"Yo, Oswaldo Luna Bastidas, Jefe del Departamento de Desarrollo de Software de la Pontificia Universidad Católica del Ecuador, me comprometo a respaldar la implementación del sistema de gestión de la calidad para el Departamento de Desarrollo de Software, su mantenimiento y mejoramiento continuo.

El compromiso con la implementación del sistema de gestión de la calidad incluye la asignación de los recursos necesarios, la planificación y revisión constante de los resultados de los procesos, establecer acuerdos con los proveedores y comunicar permanentemente al equipo de trabajo la importancia de satisfacer las necesidades de los clientes.

Ing. Oswaldo Luna Bastidas

Jefe del Departamento de Desarrollo de Software

PUCE

(Original firmado)"

5.3 POLÍTICA DE LA CALIDAD.

La política de la calidad constituye el marco dentro del cual se desarrolla el sistema de gestión de la calidad.

"El Departamento de Desarrollo de Software de la PUCE lleva a cabo su actividad de desarrollo de aplicaciones informáticas para la comunidad universitaria de la PUCE, mediante la prestación ágil y eficiente de soluciones tecnológicas de alta calidad, enfocado en satisfacer las necesidades de sus clientes, con una utilización eficiente de recursos y mejoramiento continuo de sus procesos."

5.4 OBJETIVOS DE LA CALIDAD.

Los objetivos de la calidad deben ser medibles y coherentes con la política de la calidad.

a) Desarrollar ágil y eficientemente soluciones tecnológicas de calidad, reduciendo las no conformidades en un 25% semestral.
b) Satisfacer las necesidades de las partes interesadas relevantes, incrementando la trazabilidad entre los requisitos y los productos de trabajo en un 20% semestral.
c) Utilizar eficientemente los recursos, incrementando el cumplimiento de los hitos planificados en un 30% semestral.

d) Mejorar continuamente los procesos, disminuyendo el reproceso de las actividades software en un 15% semestral.

5.5 MAPA DE PROCESOS PROPUESTO PARA EL DEPARTAMENTO DE DESARROLLO DE SOFTWARE DE LA PUCE.

En las organizaciones los procesos se clasifican en tres grupos: gobernantes, cadena de valor y apoyo.

Los procesos gobernantes determinan las políticas, objetivos y estrategias a seguir por las organizaciones. Los procesos gobernantes propuestos para el Departamento de Desarrollo de Software de la PUCE son: Gestión de Procesos y Gestión de Proyectos.

Los procesos de cadena de valor añaden valor al servicio prestado desde la óptica del cliente externo. Los procesos de cadena de valor propuestos para el Departamento de Desarrollo de Software de la PUCE son los de Ingeniería: Desarrollo de Requisitos, Solución Técnica, Verificación, Integración del Producto, Validación.

Los procesos de apoyo sirven de soporte a los demás procesos. Los procesos de apoyo propuestos para el Departamento de Desarrollo de Software de la PUCE son los de Soporte: Gestión de Configuración, Medición y Análisis, Aseguramiento de la Calidad del Proceso y del Producto, Análisis de Decisiones y Resolución, Análisis Causal y Resolución.

A continuación, se presenta el mapa de procesos propuesto para el Departamento de Desarrollo de Software de la PUCE:

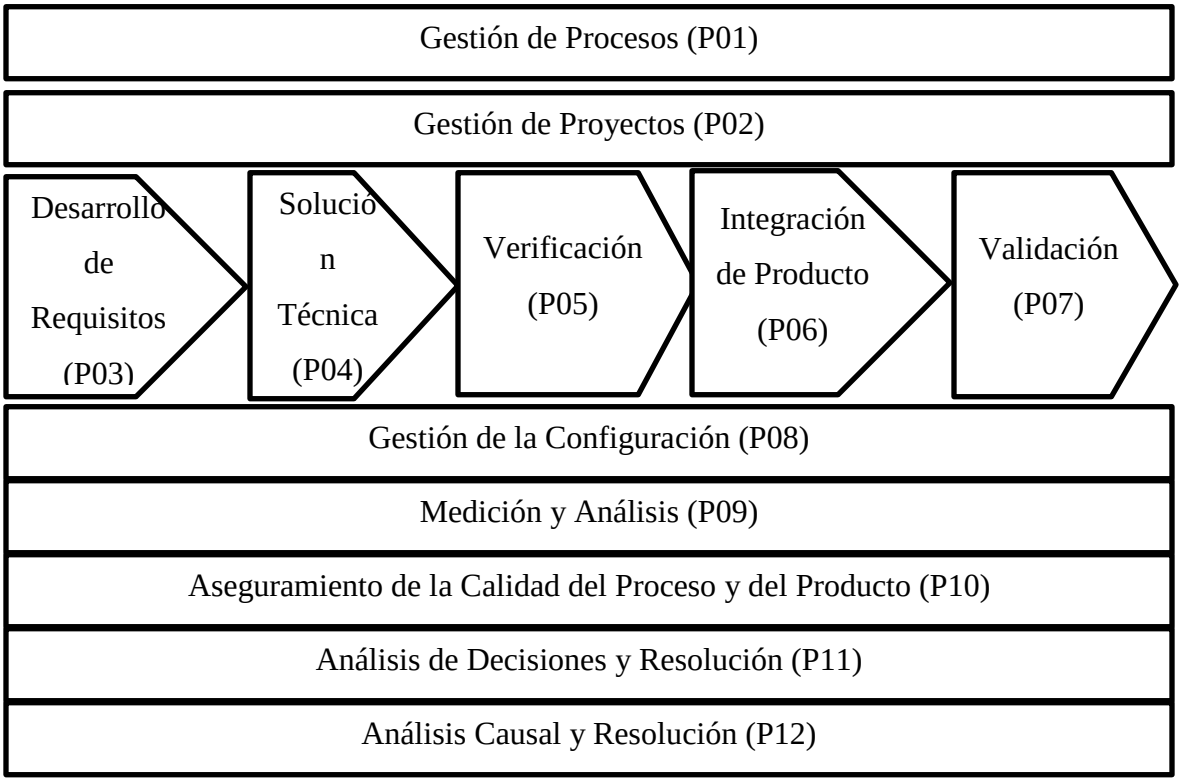


Figura 39. Mapa de Procesos Propuesto

Fuente: El autor de la tesis

5.6 DEFINICIÓN DETALLADA DE LOS PROCESOS DEL NIVEL 2 DE MADUREZ.

La definición detallada del funcionamiento de los procesos del nivel de madurez 2 (gestionado) se realizará mediante descripciones documentadas cuyo contenido será el siguiente:

- Fecha e información de la versión o revisión
- Autor, Experto y aprobaciones requeridas
- Declaración del Propósito del proceso
- Declaración de la Política para el proceso
- Identificación de los Roles y Responsabilidades
- Procedimiento que describe la secuencia lógica de actividades mediante un diagrama de flujo el cual contiene: entradas, actividades, controles, salidas o productos resultantes y partes interesadas relevantes del proceso.
- Definiciones de los términos propios del proceso
- Indicadores y medidas para monitorear la evolución del rendimiento
- Referencias a otros documentos o formatos

Las descripciones de los procesos del nivel de madurez 2 se han agrupado de acuerdo al "Mapa de Procesos Propuesto", y el detalle de su funcionamiento se presenta en el Anexo 2 "Manual de Procesos".

5.6.1 Procesos Gobernantes

5.6.1.1 Gestión de Procesos (P01)

Las descripciones de los procesos en este apartado permiten establecer un conjunto de activos de proceso de la organización, estándares del entorno de trabajo, planificar su mejoramiento, desarrollar habilidades en las personas, comprender y gestionar cuantitativamente el rendimiento del proceso para satisfacer los objetivos del negocio.

Debido a que ninguno de estos procesos tiene un nivel de madurez 2, no serán descritos en el presente trabajo, y podrán ser descritos a detalle en futuras investigaciones.

A continuación, se listan los subprocesos de la Gestión de Procesos con el respectivo código de identificación:

Tabla 10. Subprocesos Gestión de Procesos

Subproceso	Nivel de Madurez	Código
Definición de Procesos de la Organización (OPD)	3	P01-01
Enfoque en Procesos de la Organización (OPF)	3	P01-02
Formación en la Organización (OT)	3	P01-03
Rendimiento de Procesos de la Organización (OPP)	4	P01-04
Gestión del Rendimiento de la Organización (OPM)	5	P01-05

Fuente: El autor de la tesis

5.6.1.2 Gestión de Proyectos (P02)

Las descripciones de los procesos en este apartado permiten establecer los planes que definen las actividades del proyecto, aportar una visión del progreso, gestionar los requisitos del producto, gestionar la adquisición de productos con proveedores, identificar problemas potenciales antes de que ocurran, establecer el involucramiento de las partes interesadas, y gestionar cuantitativamente el proyecto.

Los procesos de la Gestión de Proyectos a describir detalladamente son los que tienen un nivel de madurez 2: Planificación del Proyecto (PP), Monitorización y Control del Proyecto (PMC), Gestión de Requisitos (REQM), Gestión de Acuerdos con Proveedores (SAM); el detalle de su funcionamiento se presenta en el Anexo 2 “Manual de Procesos”.

A continuación, se listan los subprocesos de la Gestión de Proyectos con el respectivo código de identificación:

Tabla 11. Subprocesos Gestión de Proyectos

Subproceso	Nivel de Madurez	Código
Planificación del Proyecto (PP)	2	P02-01
Monitorización y Control del Proyecto (PMC)	2	P02-02
Gestión de Requisitos (REQM)	2	P02-03
Gestión de Acuerdos con Proveedores (SAM)	2	P02-04
Gestión de Riesgos (RSKM)	3	P02-05
Gestión Integrada del Proyecto (IPM)	3	P02-06
Gestión Cuantitativa del Proyecto (QPM)	4	P02-07

Fuente: El autor de la tesis

5.6.2 Procesos de la Cadena de Valor

5.6.2.1 Desarrollo de Requisitos (P03)

La descripción del proceso en este apartado permite establecer los requisitos del cliente y del producto.

El proceso de Desarrollo de Requisitos (P03) no se describe detalladamente debido a que corresponde a un nivel de madurez 3, y podrá ser descrito a detalle en futuras investigaciones.

5.6.2.2 Solución Técnica (P04)

La descripción del proceso en este apartado permite implementar soluciones para los requisitos.

El proceso de Solución Técnica (P04) no se describe detalladamente debido a que corresponde a un nivel de madurez 3, y podrá ser descrito a detalle en futuras investigaciones.

5.6.2.3 Verificación (P05)

La descripción del proceso en este apartado permite controlar que los productos software están conformes los requisitos especificados.

El proceso de Verificación (P05) no se describe detalladamente debido a que corresponde a un nivel de madurez 3, y podrá ser descrito a detalle en futuras investigaciones.

5.6.2.4 Integración de Producto (P06)

La descripción del proceso en este apartado permite ensamblar el producto y asegurar que se comporta correctamente.

El proceso de Integración del Producto (P06) no se describe detalladamente debido a que corresponde a un nivel de madurez 3, y podrá ser descrito a detalle en futuras investigaciones.

5.6.2.5 Validación (P07)

La descripción del proceso en este apartado permite demostrar que un producto cumple con su uso previsto.

El proceso de Validación (P07) no se describe detalladamente debido a que corresponde a un nivel de madurez 3, y podrá ser descrito a detalle en futuras investigaciones.

5.6.3 Procesos de Apoyo

5.6.3.1 Gestión de la Configuración (P08)

La descripción del proceso en este apartado permite establecer la integridad de los productos de trabajo.

El proceso de Gestión de la Configuración (P08) corresponde a un nivel de madurez 2 por lo que se describe detalladamente en el Anexo 2 “Manual de Procesos”.

5.6.3.2 Medición y Análisis (P09)

La descripción del proceso en este apartado permite desarrollar medidas base y derivadas para dar soporte a los objetivos de medición.

El proceso de Medición y Análisis (P09) corresponde a un nivel de madurez 2 y se describe detalladamente en el Anexo 2 “Manual de Procesos”.

5.6.3.3 Aseguramiento de la Calidad del Proceso y del Producto (P10)

La descripción del proceso en este apartado permite monitorizar la adherencia de los procesos ejecutados y de los productos obtenidos con respecto a los planificados.

El proceso de Aseguramiento de la Calidad del Proceso y del Producto (P10) corresponde a un nivel de madurez 2 y se describe detalladamente en el Anexo 2 “Manual de Procesos”.

5.6.3.4 Análisis de Decisiones y Resolución (P11)

La descripción del proceso en este apartado permite analizar las decisiones utilizando un sistema de evaluación formal.

El proceso de Análisis de Decisiones y Resolución (P11) no se describe detalladamente debido a que corresponde a un nivel de madurez 3, y podrá ser descrito a detalle en futuras investigaciones.

5.6.3.5 Análisis Causal y Resolución (P12)

La descripción del proceso en este apartado permite identificar las causas de los resultados presentados e iniciar acciones correctivas para su solución.

El proceso de Análisis Causal y Resolución (P12) no se describe detalladamente debido a que corresponde a un nivel de madurez 5, y podrá ser descrito a detalle en futuras investigaciones.

6 CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES.

6.1 CONCLUSIONES.

- CMMI-DEV es un modelo que provee estabilidad y disciplina en la ejecución de los procesos del ciclo de vida del software.
- Las metodologías ágiles de desarrollo de software proveen una implementación eficiente y flexible de las actividades del desarrollo de aplicaciones informáticas, lo que permite alcanzar un aprovechamiento óptimo de los recursos disponibles.
- La implantación de un sistema de gestión de la calidad basado en la combinación de las buenas prácticas de CMMI-DEV con la gestión ágil de las metodologías de software ligeras como Scrum, provoca un efecto de sinergia con propiedades emergentes que permiten balancear armónicamente las bondades de ambos enfoques.
- La implantación de los procesos del nivel 2 de madurez CMMI (gestionado) es un inicio adecuado para el sistema de gestión de la calidad, ya que proporcionará una plataforma consistente, para la mejora continua de los procesos de desarrollo de software del Departamento de Desarrollo de Software de la PUCE.
- La capacitación oportuna y suficiente de las personas que serán responsables de la implantación del sistema de gestión de la calidad basado en CMMI-DEV con metodologías ágiles es imprescindible para el éxito del proyecto.

6.2 RECOMENDACIONES.

- Describir detalladamente los procesos del Departamento de Desarrollo de Software de la PUCE en los niveles de madurez 3, 4 y 5 de CMMI-DEV con metodologías ágiles de software en futuras investigaciones.
- Investigar a futuro nuevas combinaciones de estándares para la mejora de procesos software con nuevas metodologías de desarrollo ligero de software.
- Obtener certificaciones internacionales en modelos CMMI y SCAMPI (Método de evaluación estándar de CMMI para la mejora de procesos), para poder asesorar y certificar a organizaciones regionales interesadas.

7 BIBLIOGRAFÍA.

Agile Alliance :: Home. (n.d.). Retrieved November 8, 2014, from http://www.agilealliance.org/

Beck, K. (2004). *Extreme Programming Explained: Embrace Change* (segunda). Addison-Wesley.

Bessin, J. (2004). The Business Value of Quality. IBM DevelopersWorks.

Diaz, J., Garbajosa, J., & Calvo-Manzano, J. (2009). Mapping CMMI Level 2 to Scrum Practices: An Experience Report. *CCIS 42*, *EuroSPI 2009*, 93–104.

IEEE. (1993). IEEE Standards Collection: Software Engineering. IEEE Standard.

ISO 9000:2015(es), Sistemas de gestión de la calidad — Fundamentos y vocabulario. (n.d.). Retrieved April 23, 2016, from https://www.iso.org/obp/ui/#iso:std:iso:9000:ed-4:v1:es

ISO 9001:2015(es), Sistemas de gestión de la calidad — Requisitos. (n.d.). Retrieved April 23, 2016, from https://www.iso.org/obp/ui/#iso:std:iso:9001:ed-5:v1:es

isofocus_113.pdf. (n.d.). Retrieved from http://www.iso.org/iso/es/isofocus_113.pdf

ISO/IEC 25010:2011(en), Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. (n.d.). Retrieved April 23, 2016, from https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en

ISO/IEC 90003:2014(en), Software engineering — Guidelines for the application of ISO 9001:2008 to computer software. (n.d.). Retrieved April 23, 2016, from https://www.iso.org/obp/ui/#iso:std:iso-iec:90003:ed-2:v1:en

Jakobsen, C., & Johnson, K. (2008). Mature Agile with a twist of CMMI.

Manifiesto por el Desarrollo Ágil de Software. (n.d.). Retrieved April 23, 2016, from http://agilemanifesto.org/iso/es/

Marcal, A. S., & Freitas, B. (2008). Blending Scrum practices and CMMI project management process areas. *Innovations Syst Softw Eng*, 17–29. http://doi.org/10.1007/s11334-007-0040-1

McCall, J., Richards, P., & Walters, G. (1977). *Factors in Software Quality*.

McMahon, P. (2011). *Integrating CMMI and Agile Development*. Boston: Addison-Wesley.

Moniruzzaman, A., & Hossain, S. (2012). Comparative Study on Agile software development methodologies.

Ponce León, F. (2016). Cuenta y razón. Rendición de cuentas del período 2015. Pontificia Universidad Católica del Ecuador.

Pontificia Universidad Católica del Ecuador. (2014). Plan Estratégico de Desarrollo Institucional 2014 - 2018 PUCE. PUCE.

Potter, N. (2010). Comparing Scrum and CMMI. The Process Group. Retrieved from www.processgroup.com

Preis, A. (2012). Integration Evaluation of Scrum and CMMI.

PUCE. (2008). Plan Estratégico de Desarrollo Institucional.

PUCE. (2014a). Revista Actualidad n.º 27. Retrieved November 29, 2014, from http://es.calameo.com/read/00002700854ec0b582a07

PUCE. (2014b). Universidad. Retrieved November 29, 2014, from http://www.puce.edu.ec/portal/content/Universidad/104?link=oln30.redirect

Roger Pressman. (2010). *Ingeniería del software* (Séptima edición). México D.F.: McGraw-Hill.

Sánchez, S., Sicilia, M., & Rodríguez, D. (2012). *Ingeniería del software*. México D.F.: AlfaOmega Grupo Editor.

Schulmeyer, G. C., & McManus, J. I. (1998). *Handbook of Software Quality Assurance* (tercera). Prentice Hall.

Schwaber, K., & Sutherland, J. (2013). La Guía de Scrum. Retrieved March 13, 2016, from http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-ES.pdf

Software Engineering Institute. (2010). Spanish Technical Report CMMI V 1 3.pdf. Retrieved March 13, 2016, from http://www.sei.cmu.edu/library/assets/whitepapers/Spanish%20Technical%20Report%20CMMI%20V%201%203.pdf

Surveys Exploring The Current State of Information Technology Practices. (n.d.). Retrieved April 23, 2016, from http://www.ambysoft.com/surveys/

Sutherland, J., Jakobsen, C., & Johnson, K. (2008). Scrum and CMMI Level 5: The Magic Potion for Code Warriors. *Hawaii International Conference on System Sciences*, *41*, 1–9. http://doi.org/10.1109/HICSS.2008.384

Torrecilla, C. (2012). A Scrum-based approach to CMMI maturity level 2 in Web Development environments.

8 ANEXOS.

8.1 ANEXO 1. ENCUESTA Y RESULTADOS.

Encuesta de Procesos de Desarrollo de Software

La presente encuesta tiene como objetivo recopilar información acerca del estado actual de los Procesos Software que se llevan a cabo en el Departamento de Desarrollo de Software de la PUCE bajo la óptica del estándar CMMI para Desarrollo v1.3

La encuesta es anónima con el propósito de recoger respuestas transparentes y honestas.

Esta encuesta consta de 22 preguntas de opción múltiple, la duración aproximada es de alrededor de 5 minutos.

La presente encuesta forma parte de la tesis de posgrado: "Diseño de un Sistema de Gestión de la Calidad para el Departamento de Desarrollo de Software de la PUCE".

1. El proceso de establecer y mantener un plan de proyecto como base para gestionar el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

2. El proceso de establecer y gestionar el proyecto software y la involucración de las partes interesadas relevantes, de acuerdo a un proceso integrado y definido, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

3. El proceso de establecer y mantener las descripciones de los modelos de ciclo de vida y los estándares del entorno de trabajo del proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

4. El proceso de establecer un plan táctico de formación para que las personas desempeñen sus roles con eficacia en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca

- NoAplica/NoSabe

5. El proceso de evaluar y clasificar cada riesgo usando las categorías y los parámetros definidos para la gestión del riesgo, y determinar su prioridad relativa en el proyecto software, se cumple:
   - siempre
   - frecuentemente
   - rara vez
   - nunca
   - NoAplica/NoSabe

6. El proceso de desarrollar los requisitos de producto y sus atributos de calidad, en base a la priorización y refinación de los requisitos del cliente en el proyecto software, se cumple:
   - siempre
   - frecuentemente
   - rara vez
   - nunca
   - NoAplica/NoSabe

7. El proceso de gestionar los requisitos, mantener su trazabilidad bidireccional e identificar las inconsistencias de los requisitos con los planes y productos de trabajo del proyecto software, se cumple:
   - siempre

- frecuentemente
- rara vez
- nunca
- NoAplica/NoSabe

8. El proceso de asegurar que los acuerdos con el proveedor se formalizan antes de aceptar el producto adquirido en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

9. El proceso de seleccionar, diseñar e implementar soluciones para los componentes de producto, en base a criterios establecidos en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

10. El proceso de realizar las revisiones entre pares de los productos de trabajo en base a criterios de verificación establecidos en el proyecto software, e identificar las cuestiones resultantes de estas revisiones, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

11. El proceso de decisión basado en una evaluación formal de soluciones alternativas para tratar los problemas en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

12. El proceso de establecer y mantener un sistema de gestión de configuración y de gestión de cambios para controlar los productos de trabajo en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca

- NoAplica/NoSabe

13. El proceso de monitorizar los valores reales de los parámetros de planificación del proyecto software, frente al plan de proyecto, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

14. El proceso de recolectar y almacenar los datos de medición del proceso en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

15. El proceso de gestionar el proyecto software utilizando medidas cuantitativas para determinar si se alcanzarán los objetivos de calidad y de rendimiento del proceso, se cumple:
    - siempre
    - frecuentemente
    - rara vez

- nunca
- NoAplica/NoSabe

16. El proceso de analizar el rendimiento de los procesos del proyecto software y establecer y mantener las líneas base de rendimiento de procesos, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

17. El proceso de evaluar objetivamente la conformidad de los procesos realizados y de los productos de trabajo en el proyecto software, con las descripciones de proceso, estándares y procedimientos aplicables, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

18. El proceso de seleccionar los componentes de producto a validar y los métodos de validación a utilizar en el proyecto software, para asegurar que los productos son adecuados para su utilización en el entorno de operación previsto, se cumple:
    - siempre

- frecuentemente
- rara vez
- nunca
- NoAplica/NoSabe

19. El proceso de ensamblar los componentes de producto verificados del proyecto software, y entregar el producto integrado, verificado y validado, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

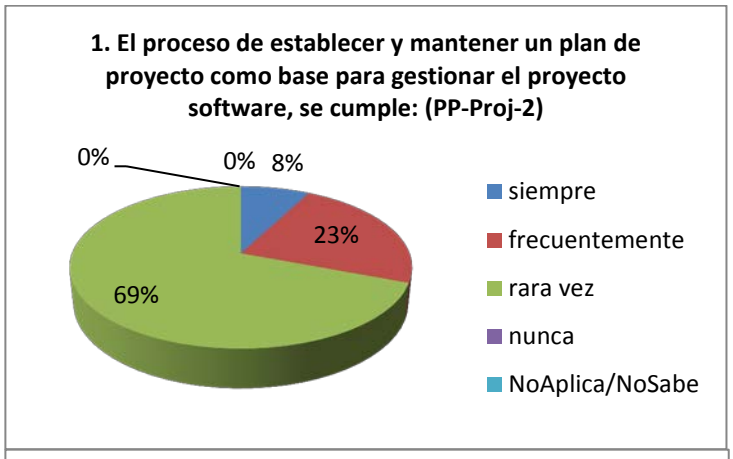
20. El proceso de realizar el análisis causal de los problemas en el proyecto software y proponer acciones para tratarlos, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

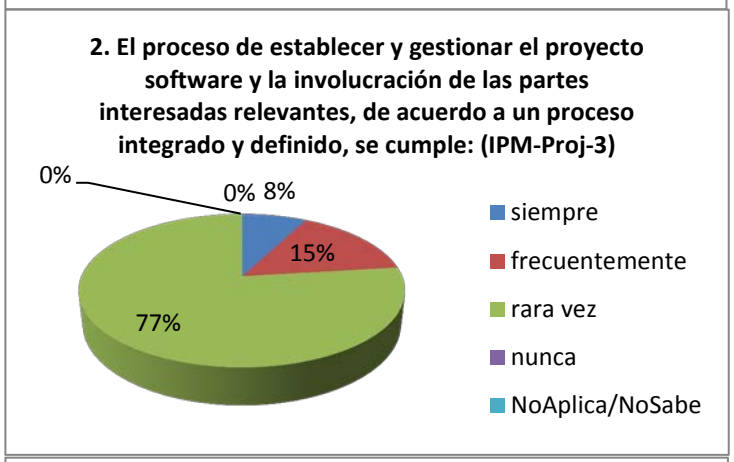
21. El proceso de gestionar el rendimiento de los procesos en el proyecto software utilizando técnicas estadísticas para comprender carencias de rendimiento e identificar áreas para la mejora de procesos, se cumple:

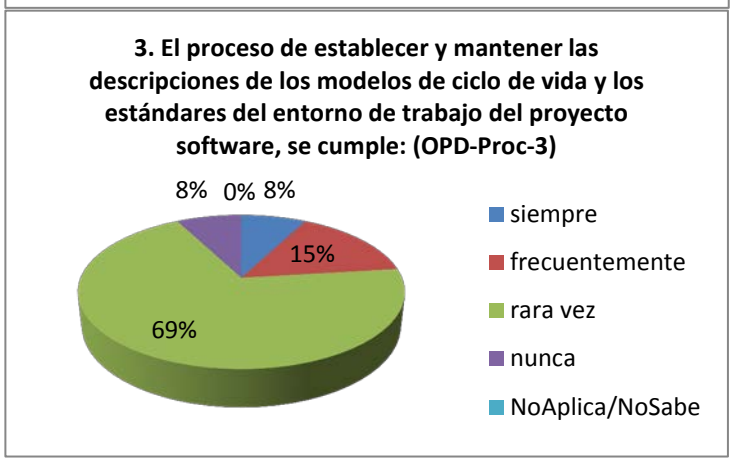
- siempre
- frecuentemente
- rara vez
- nunca
- NoAplica/NoSabe

22. El proceso de determinar fortalezas, debilidades y oportunidades de mejora de procesos en el proyecto software, se cumple:
    - siempre
    - frecuentemente
    - rara vez
    - nunca
    - NoAplica/NoSabe

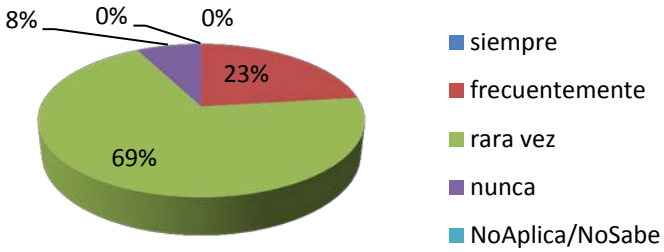
Resultados de la Tabulación de las Encuestas



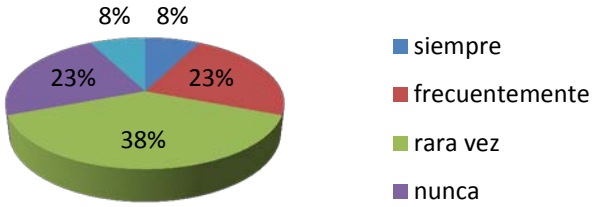




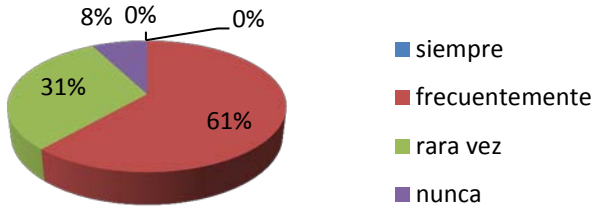
4. El proceso de establecer un plan táctico de formación para que las personas desempeñen sus roles con eficacia en el proyecto software, se cumple: (OT-Proc-3)



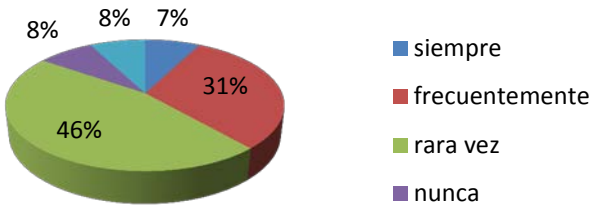
5. El proceso de evaluar y clasificar cada riesgo usando las categorías y los parámetros definidos para la gestión del riesgo, y determinar su prioridad relativa en el proyecto software, se cumple: (RSKM-Proj-3)



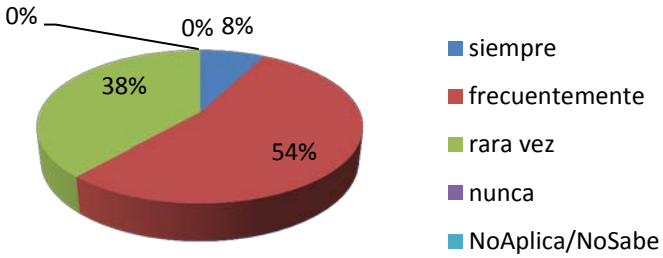
6. El proceso de desarrollar los requisitos de producto y sus atributos de calidad, en base a la priorización y refinación de los requisitos del cliente en el proyecto software, se cumple: (RD-Eng-3)



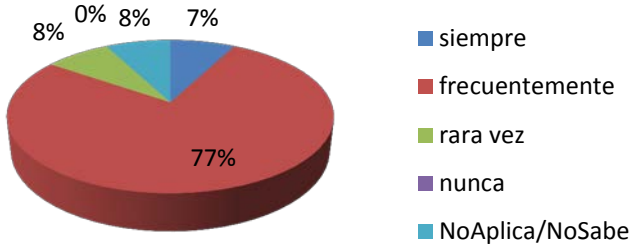
7.El proceso de gestionar los requisitos, mantener su trazabilidad bidireccional e identificar las inconsistencias de los requisitos con los planes y productos de trabajo del proyecto software, se cumple: (REQM-Proj-2)



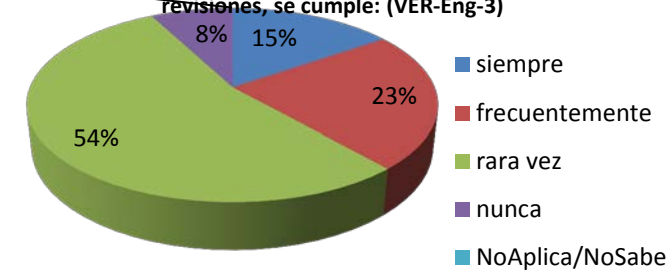
8.El proceso de asegurar que los acuerdos con el proveedor se formalizan antes de aceptar el producto adquirido en el proyecto software, se cumple: (SAM-Proj-2)



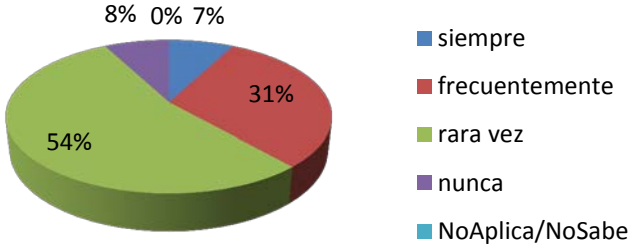
9.El proceso de seleccionar, diseñar e implementar soluciones para los componentes de producto , en base a criterios establecidos en el proyecto software, se cumple: (TS-Eng-3)



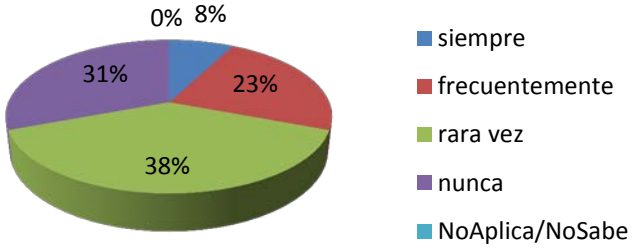
10.El proceso de realizar las revisiones entre pares de los productos de trabajo en base a criterios de verificación establecidos en el proyecto software, e identificar las cuestiones resultantes de estas revisiones, se cumple: (VER-Eng-3)



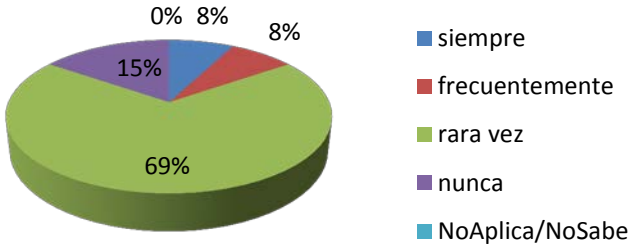
11.El proceso de decisión basado en una evaluación formal de soluciones alternativas para tratar los problemas en el proyecto software, se cumple: (DAR-Sup-3)



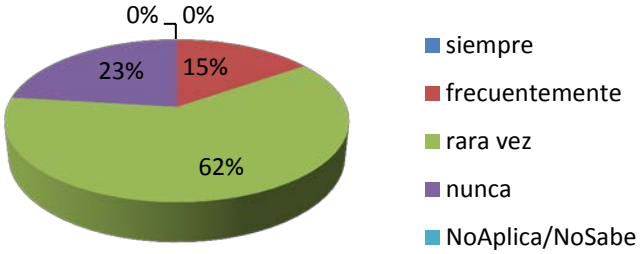
12.El proceso de establecer y mantener un sistema de gestión de configuración y de gestión de cambios para controlar los productos de trabajo en el proyecto software, se cumple: (CM-Sup-2)



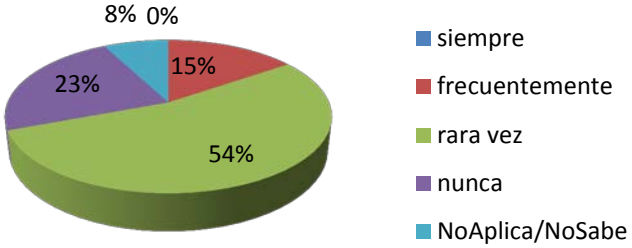
13.El proceso de monitorizar los valores reales de los parámetros de planificación del proyecto software, frente al plan de proyecto, se cumple: (PMC-Proj-2)



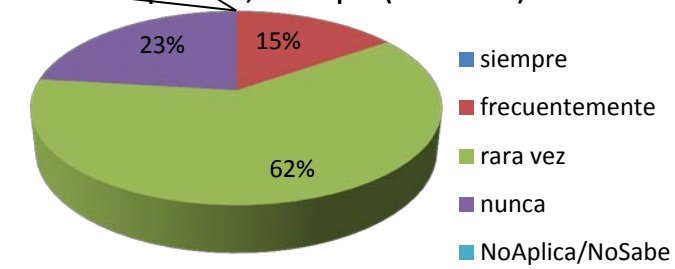
14.El proceso de recolectar y almacenar los datos de medición del proceso en el proyecto software, se cumple: (MA-Sup-2)



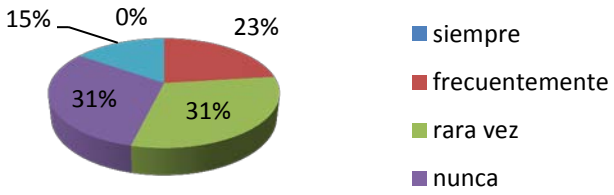
15.El proceso de gestionar el proyecto software utilizando medidas cuantitativas para determinar si se alcanzarán los objetivos de calidad y de rendimiento del proceso, se cumple: (QPM-Proj-4)



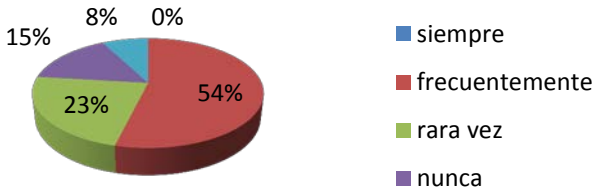
16.El proceso de analizar el rendimiento de los procesos del proyecto software y establecer y mantener las líneas base de rendimiento de procesos, se cumple: (OPP-Proc-4)

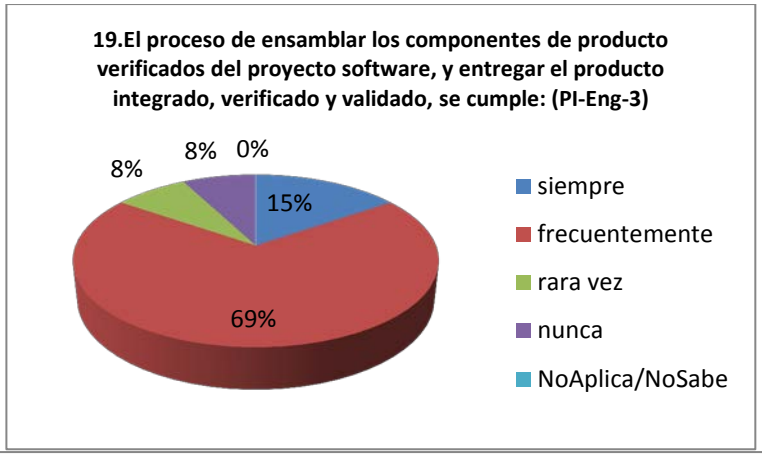


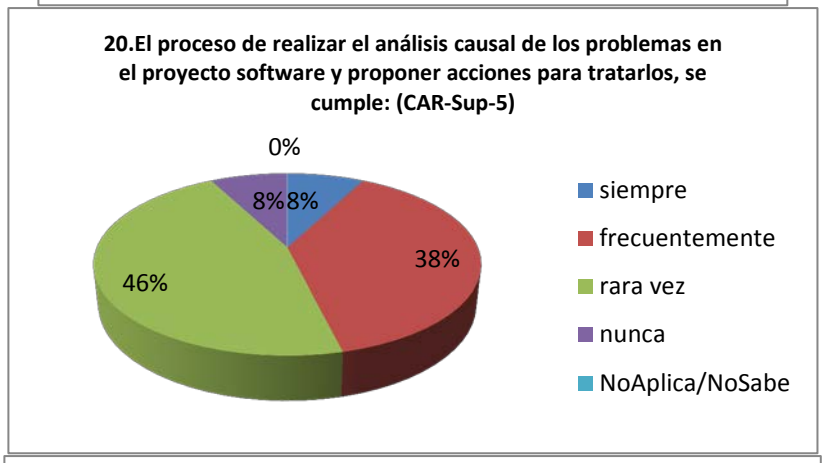
17.El proceso de evaluar objetivamente la conformidad de los procesos realizados y de los productos de trabajo en el proyecto software, con las descripciones de proceso, estándares y procedimientos aplicables, se cumple: (PPQA-Sup-2)

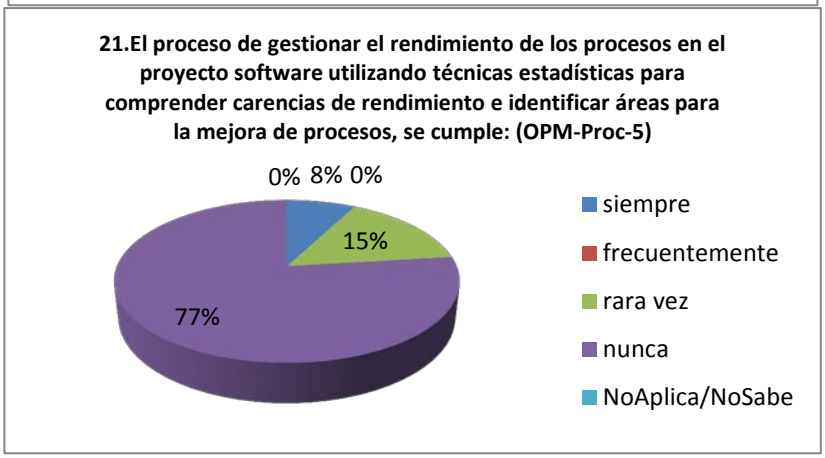


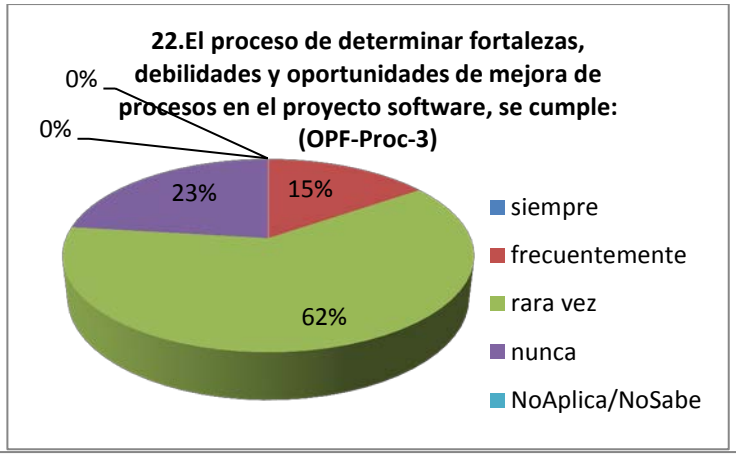
18.El proceso de seleccionar los componentes de producto a validar y los métodos de validación a utilizar en el proyecto software, para asegurar que los productos son adecuados para su utilización en el entorno de operación previsto, se cumple: (VAL-Eng-3

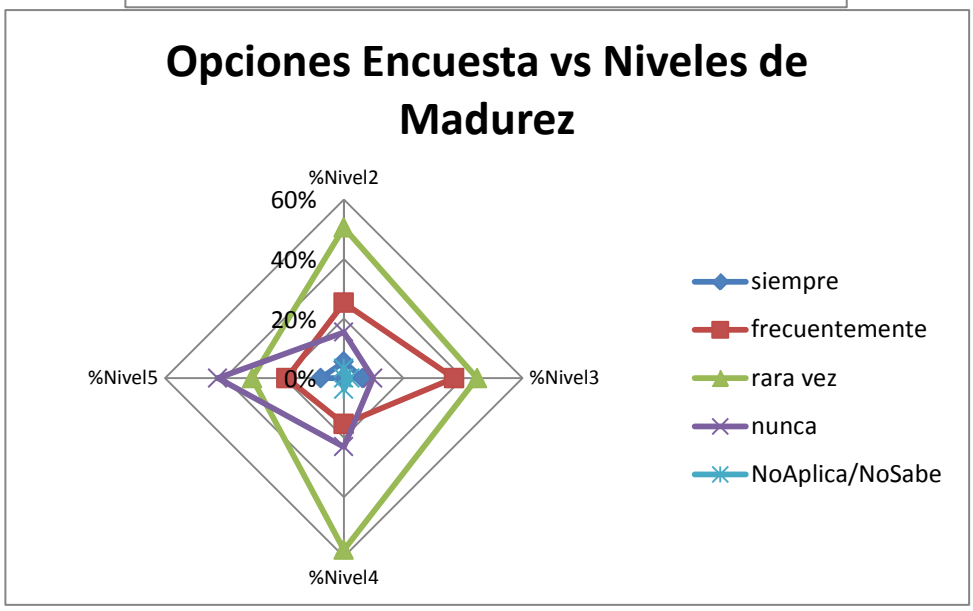


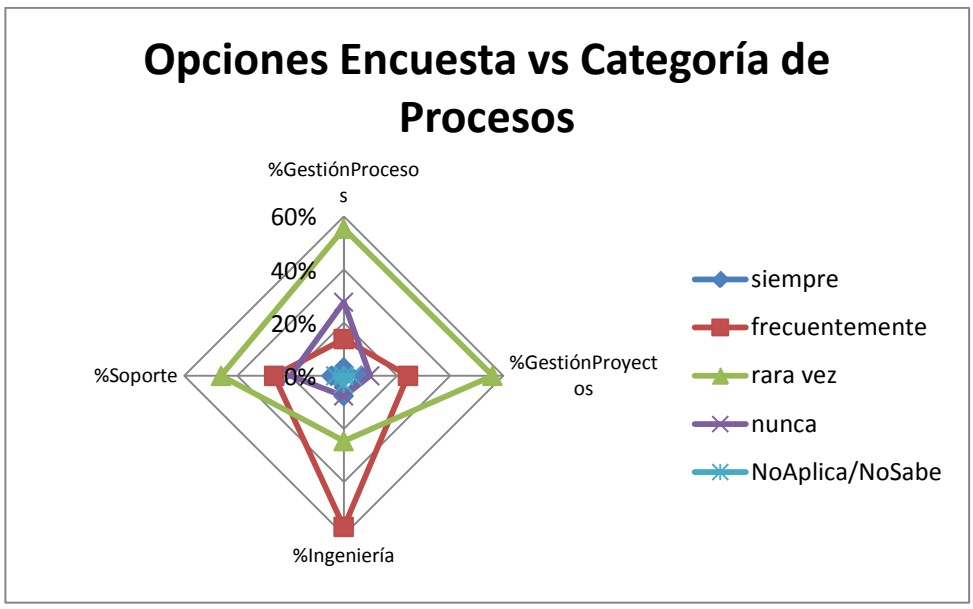

19.El proceso de ensamblar los componentes de producto verificados del proyecto software, y entregar el producto integrado, verificado y validado, se cumple: (PI-Eng-3)
8%
8%
0%
15%
69%
siempre
frecuentemente
rara vez
nunca
NoAplica/NoSabe


20.El proceso de realizar el análisis causal de los problemas en el proyecto software y proponer acciones para tratarlos, se cumple: (CAR-Sup-5)
0%
8%
8%
38%
46%
siempre
frecuentemente
rara vez
nunca
NoAplica/NoSabe


21.El proceso de gestionar el rendimiento de los procesos en el proyecto software utilizando técnicas estadísticas para comprender carencias de rendimiento e identificar áreas para la mejora de procesos, se cumple: (OPM-Proc-5)
0%
8%
0%
15%
77%
siempre
frecuentemente
rara vez
nunca
NoAplica/NoSabe


22.El proceso de determinar fortalezas, debilidades y oportunidades de mejora de procesos en el proyecto software, se cumple: (OPF-Proc-3)
0%
0%
23%
15%
62%
siempre
frecuentemente
rara vez
nunca
NoAplica/NoSabe


Opciones Encuesta vs Niveles de Madurez
%Nivel2
60%
40%
20%
0%
%Nivel5
%Nivel3
%Nivel4
siempre
frecuentemente
rara vez
nunca
NoAplica/NoSabe


Opciones Encuesta vs Categoría de Procesos
%GestiónProcesos
60%
40%
20%
0%
%Soporte
%GestiónProyectos
%Ingeniería
siempre
frecuentemente
rara vez
nunca
NoAplica/NoSabe

8.2 ANEXO 2. MANUAL DE PROCESOS.

1. PROPÓSITO

Evaluar objetivamente los procesos realizados y los productos de trabajo resultantes con respecto a las descripciones de proceso y estándares, identificar y tratar las no conformidades y comunicar los resultados de la evaluación al personal del proyecto y a la gerencia.

2. POLÍTICA

El aseguramiento de la calidad del proceso y del producto se lo realizará de manera ágil mediante evaluaciones objetivas valiosas y eficientes llevadas a cabo diariamente sobre los procesos y productos de trabajo en desarrollo durante el sprint.

3. RESPONSABLES

Equipo de planificación.- es el responsable de establecer las actividades de la calidad que asegurarán el cumplimiento de los objetivos de la calidad. Está conformado por el Jefe de Desarrollo y el Dueño del Producto.

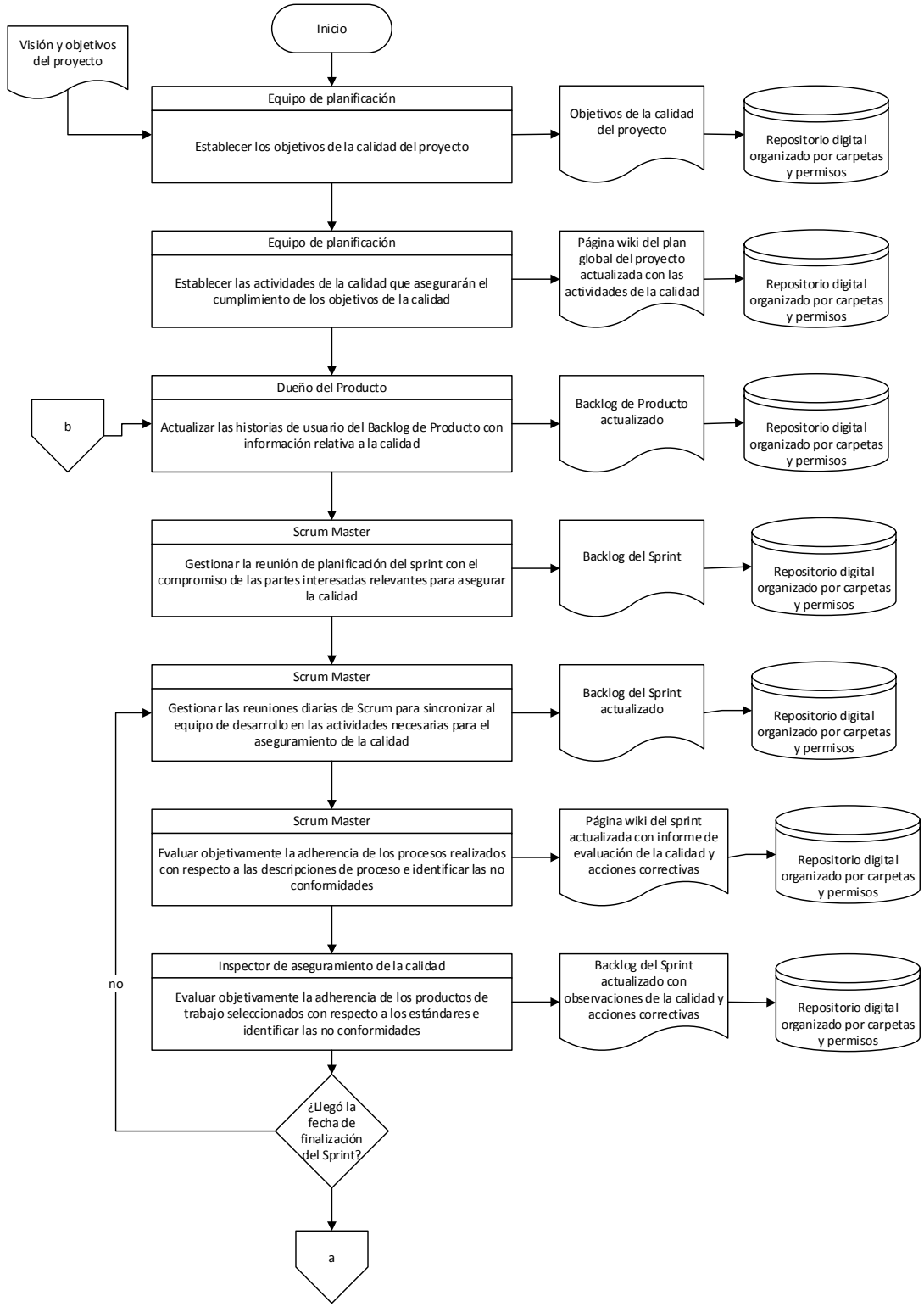
Dueño del producto.- es el responsable de actualizar las historias de usuario del Backlog del Producto con información relativa a la calidad.

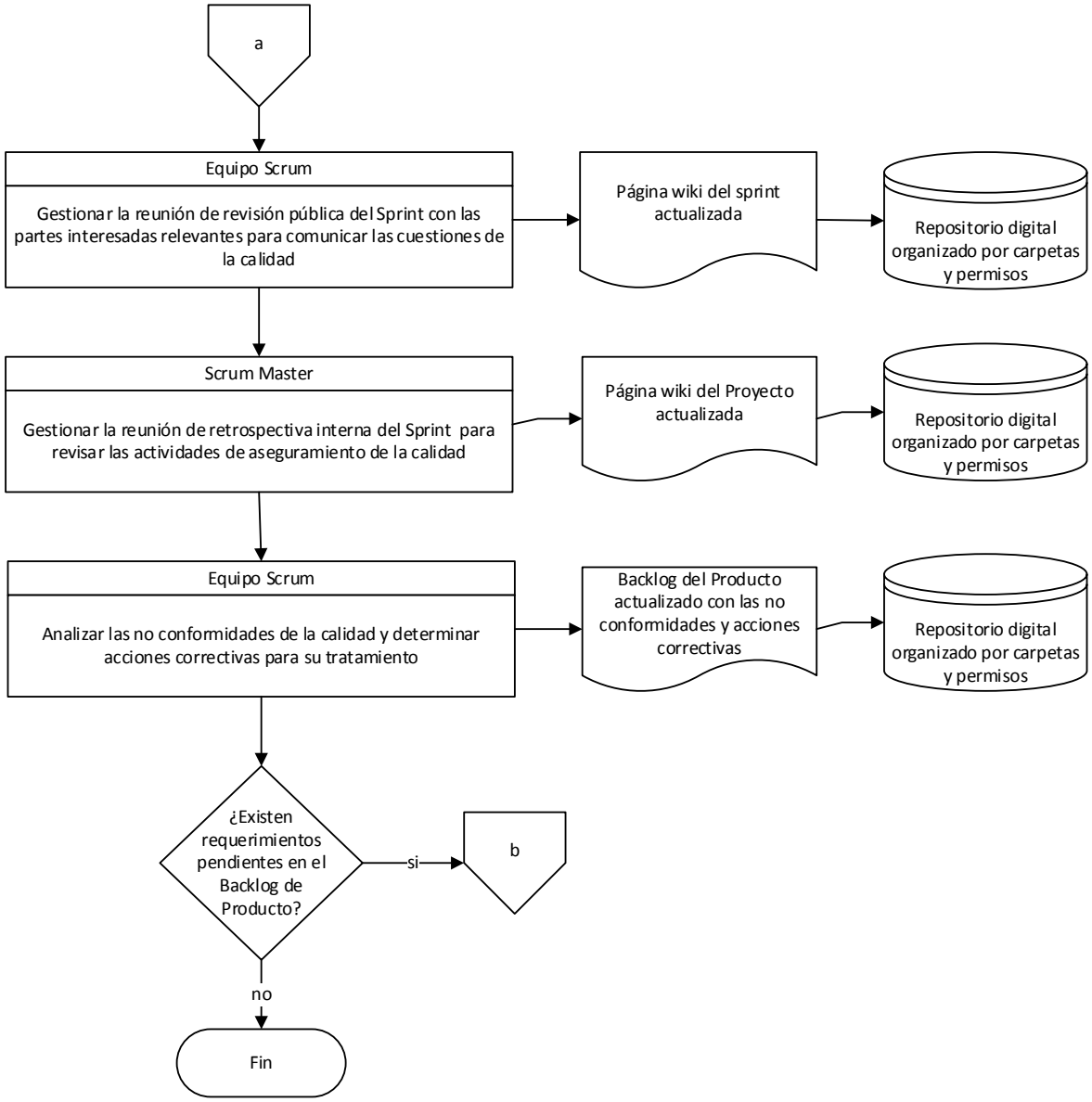
Scrum Master.- es el responsable de evaluar objetivamente la adherencia de los procesos realizados con respecto a las descripciones de proceso e identificar las no conformidades.

Inspector de aseguramiento de la calidad.- es el responsable de evaluar objetivamente la adherencia de los productos de trabajo seleccionados con respecto a los estándares e identificar las no conformidades. El inspector de aseguramiento de la calidad es normalmente asignado de forma rotativa de entre los desarrolladores con más experiencia que no hayan desarrollado el producto de trabajo seleccionado.

Equipo Scrum.- es el responsable de analizar las no conformidades de la calidad y determinar acciones correctivas para su tratamiento.

4. PROCEDIMIENTO




a
Equipo Scrum
Gestionar la reunión de revisión pública del Sprint con las partes interesadas relevantes para comunicar las cuestiones de la calidad
Página wiki del sprint actualizada
Repositorio digital organizado por carpetas y permisos
Scrum Master
Gestionar la reunión de retrospectiva interna del Sprint para revisar las actividades de aseguramiento de la calidad
Página wiki del Proyecto actualizada
Repositorio digital organizado por carpetas y permisos
Equipo Scrum
Analizar las no conformidades de la calidad y determinar acciones correctivas para su tratamiento
Backlog del Producto actualizado con las no conformidades y acciones correctivas
Repositorio digital organizado por carpetas y permisos
¿Existen requerimientos pendientes en el Backlog de Producto?
si
b
no
Fin

5. DEFINICIONES

Página de Información del Proyecto.- es una página wiki con información actualizada del avance del proyecto con la siguiente información: objetivo y visión del proyecto, el Backlog del Producto, la velocidad estimada del equipo, el calendario con la duración del proyecto, presupuesto, la infraestructura de soporte, compromiso de las partes interesadas relevantes y el gráfico burndown de proyecto.

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Solicitante de la historia de usuario, Responsable de la historia de usuario, Estado de la historia de usuario, Fecha del estado de historia de usuario, ID producto de trabajo, ID seguimiento de errores.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado), Fecha del estado de la tarea, ID producto de trabajo, ID de la historia de usuario asociada.

Página de Información del Sprint.- es un página wiki con información actualizada del avance del Sprint con la siguiente información: el objetivo del sprint, el Backlog del sprint, la velocidad estimada del equipo, el calendario con la duración del sprint, hora del scrum diario, fecha para la demo del sprint, disponibilidad de los miembros del equipo y el compromiso de las partes interesadas relevantes.

Objetivos de la calidad del proyecto.- son los resultados del aseguramiento de la calidad de productos y procesos que se esperan alcanzar al finalizar el proyecto.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P10-I1	Densidad de defectos	Mensual	Número de defectos detectados por cada fase del ciclo de vida / Tamaño del producto de trabajo
P10-I2	Grado de consecución de hitos	Mensual	Número de hitos cumplidos / Número total de hitos planificados

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P10-R1	Formato de Backlog de Producto
P10-R2	Formato de Backlog de Sprint
P10-R3	Formato de página wiki con información del proyecto
P10-R4	Formato de página wiki con información del Sprint
P10-R5	Estructura de organización del repositorio digital

1. PROPÓSITO

Gestionar la adquisición de productos y servicios de proveedores que pueden ser entregados al cliente del proyecto.

2. POLÍTICA

La gestión de acuerdos con los proveedores se lo realizará de manera ágil mediante la utilización de técnicas de contratación simplificadas para el llamado a selección de proveedores.

3. RESPONSABLES

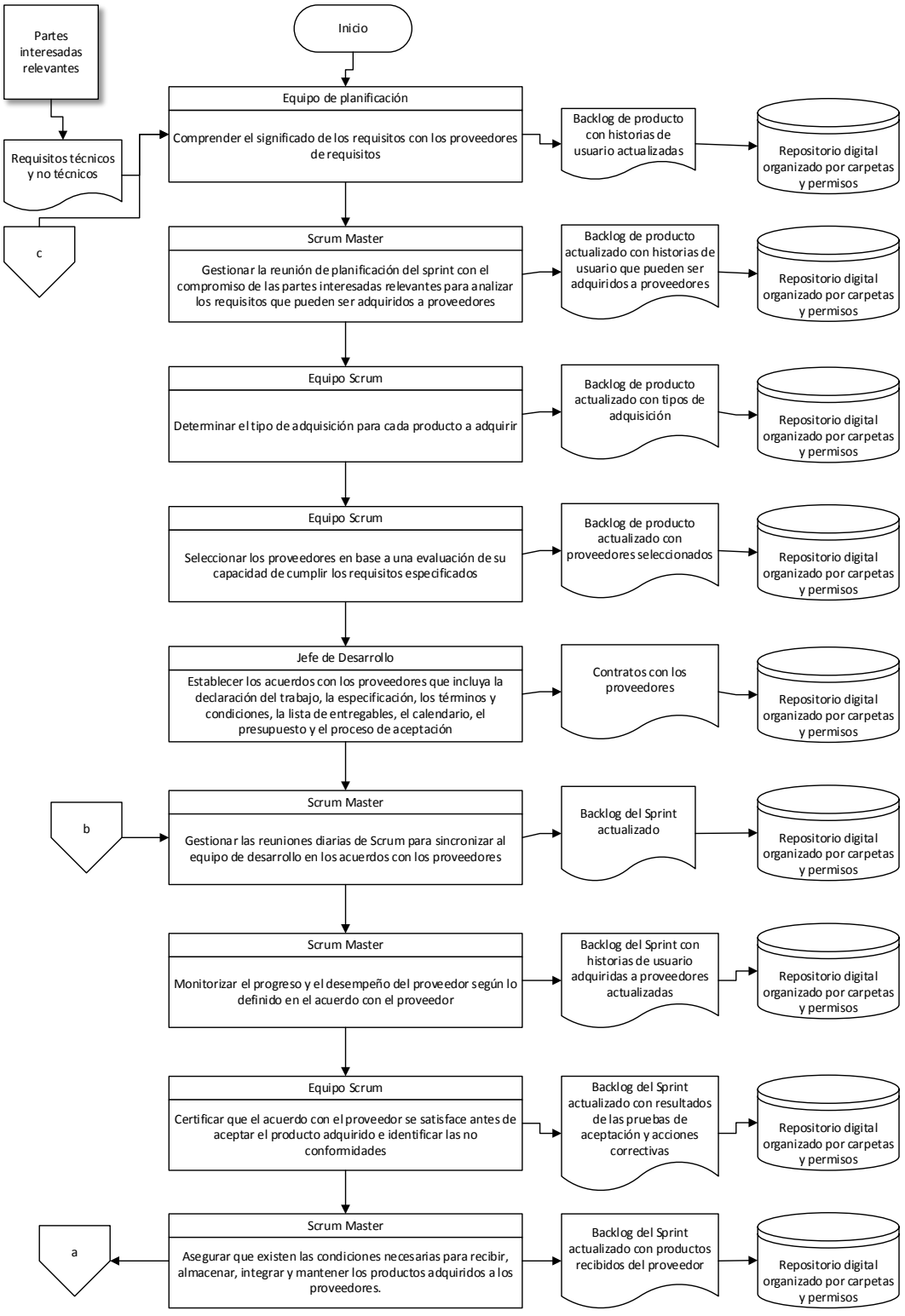
Equipo de planificación.- es el responsable de comprender el significado de los requisitos con los proveedores de requisitos. Está conformado por el Jefe de Desarrollo y el Dueño del Producto.

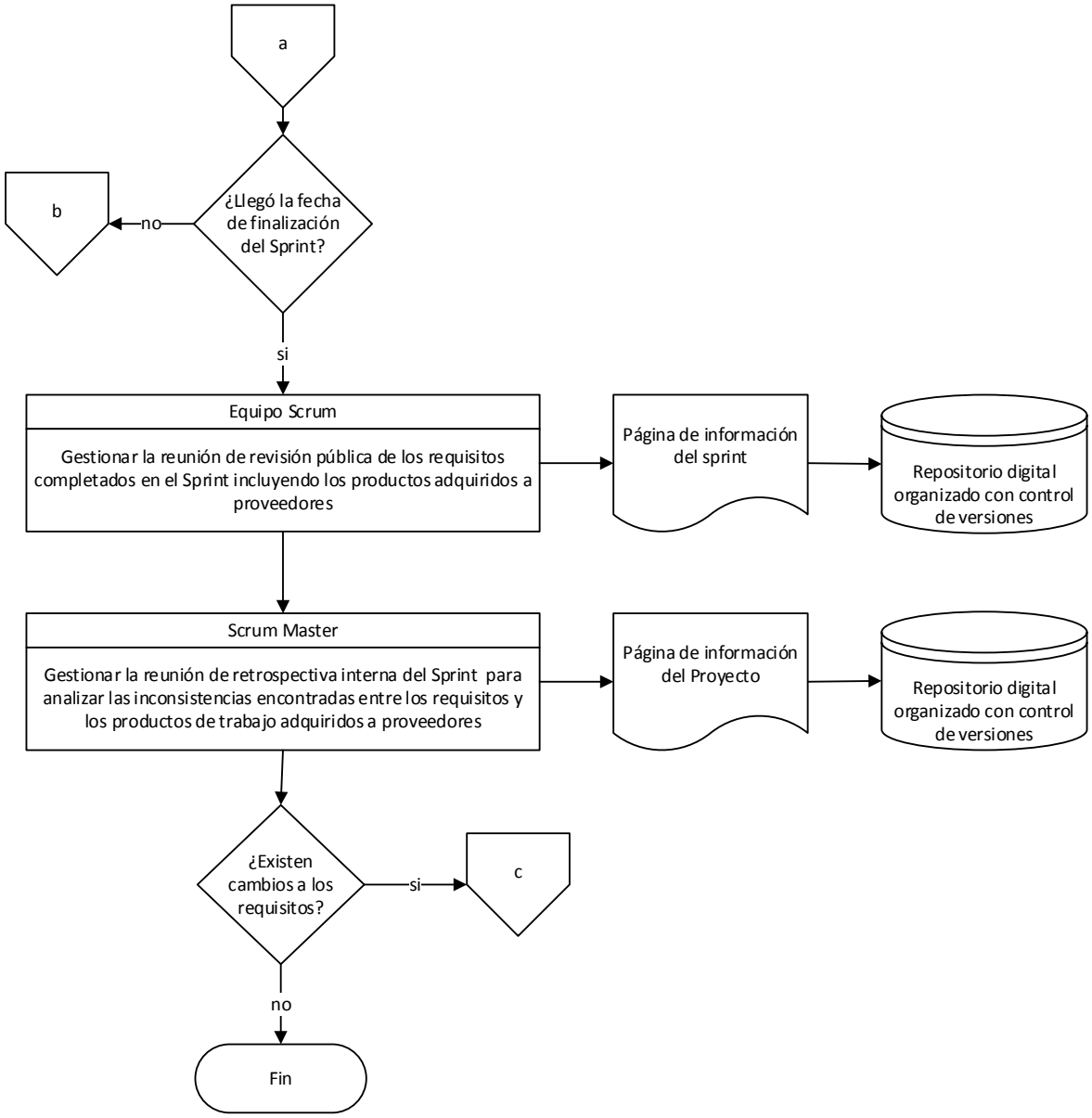
Scrum Master.- es el responsable de monitorizar el progreso y el desempeño del proveedor, y asegurar que existen las condiciones necesarias para recibir, almacenar, integrar y mantener los productos adquiridos a proveedores.

Equipo Scrum.- es el responsable de determinar el tipo de adquisición para cada producto a adquirir, seleccionar los proveedores en base a una evaluación de su capacidad y certificar que el acuerdo con el proveedor se satisface antes de aceptar el producto adquirido e identificar las no conformidades. Está conformado por el Dueño del Producto, el Scrum Master y el Equipo de Desarrollo.

Jefe de desarrollo.- es el responsable de establecer los acuerdos con los proveedores que incluya la declaración del trabajo, la especificación, los términos y condiciones, la lista de entregables, el calendario, el presupuesto y el proceso de aceptación.

4. PROCEDIMIENTO




a
¿Llegó la fecha de finalización del Sprint?
no
b
si
Equipo Scrum
Gestionar la reunión de revisión pública de los requisitos completados en el Sprint incluyendo los productos adquiridos a proveedores
Página de información del sprint
Repositorio digital organizado con control de versiones
Scrum Master
Gestionar la reunión de retrospectiva interna del Sprint para analizar las inconsistencias encontradas entre los requisitos y los productos de trabajo adquiridos a proveedores
Página de información del Proyecto
Repositorio digital organizado con control de versiones
¿Existen cambios a los requisitos?
si
c
no
Fin

5. DEFINICIONES

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Solicitante de la historia de usuario, Responsable de la historia de usuario, Estado de la historia de usuario, Fecha del estado de historia de usuario, ID producto de trabajo, ID seguimiento de errores.

Contratos con los proveedores.- es un acuerdo escrito entre la organización dueña del proyecto y el proveedor que puede ser un contrato, licencia, acuerdo de nivel de servicio o memorando de acuerdo y debe contener: declaración del trabajo, la especificación, los términos y condiciones, la lista de entregables, el calendario, el presupuesto y el proceso de aceptación.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado), Fecha del estado de la tarea, ID producto de trabajo, ID de la historia de usuario asociada.

Página de Información del Proyecto.- es una página wiki con información actualizada del avance del proyecto con la siguiente información: objetivo y visión del proyecto, el Backlog del Producto, la velocidad estimada del equipo, el calendario con la duración del proyecto, presupuesto, la infraestructura de soporte, compromiso de las partes interesadas relevantes y el gráfico burndown de proyecto.

Página de Información del Sprint.- es un página wiki con información actualizada del avance del Sprint con la siguiente información: el objetivo del sprint, el Backlog del sprint, la velocidad estimada del equipo, el calendario con la duración del sprint, hora del scrum diario, fecha para la demo del sprint, disponibilidad de los miembros del equipo y el compromiso de las partes interesadas relevantes.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P02-04-I1	Tasa de productos adquiridos a proveedores	Mensual	Número de productos adquiridos a proveedores / Número total de productos del sprint
P02-04-I2	Tasa de costo real de productos adquiridos a proveedores	Mensual	Costo real de los productos adquiridos a proveedores / Costo estimado de los productos adquiridos a proveedores

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P02-04-R1	Formato de Backlog de Producto
P02-04-R2	Formato de Backlog de Sprint
P02-04-R3	Formato de página wiki con información del proyecto
P02-04-R4	Formato de página wiki con información del Sprint
P02-04-R5	Estructura de organización del repositorio digital
P02-04-R6	Formato de contratos con los proveedores

1. PROPÓSITO

Establecer la integridad de los productos de trabajo que componen las líneas base en momentos determinados, mediante la identificación y control de la configuración y las evaluaciones de la configuración.

2. POLÍTICA

La gestión de la configuración del proyecto se la realizará de manera ágil mediante la automatización del sistema de gestión de la configuración para soportar la integración y prueba continua de los productos de trabajo durante el sprint y la entrega de software incremental al final de cada sprint.

3. RESPONSABLES

Equipo de planificación.- es el responsable de seleccionar los elementos de configuración y los productos de trabajo que los componen para ser controlados por el sistema de gestión de la configuración. Está conformado por el Jefe de Desarrollo y el Dueño del Producto.

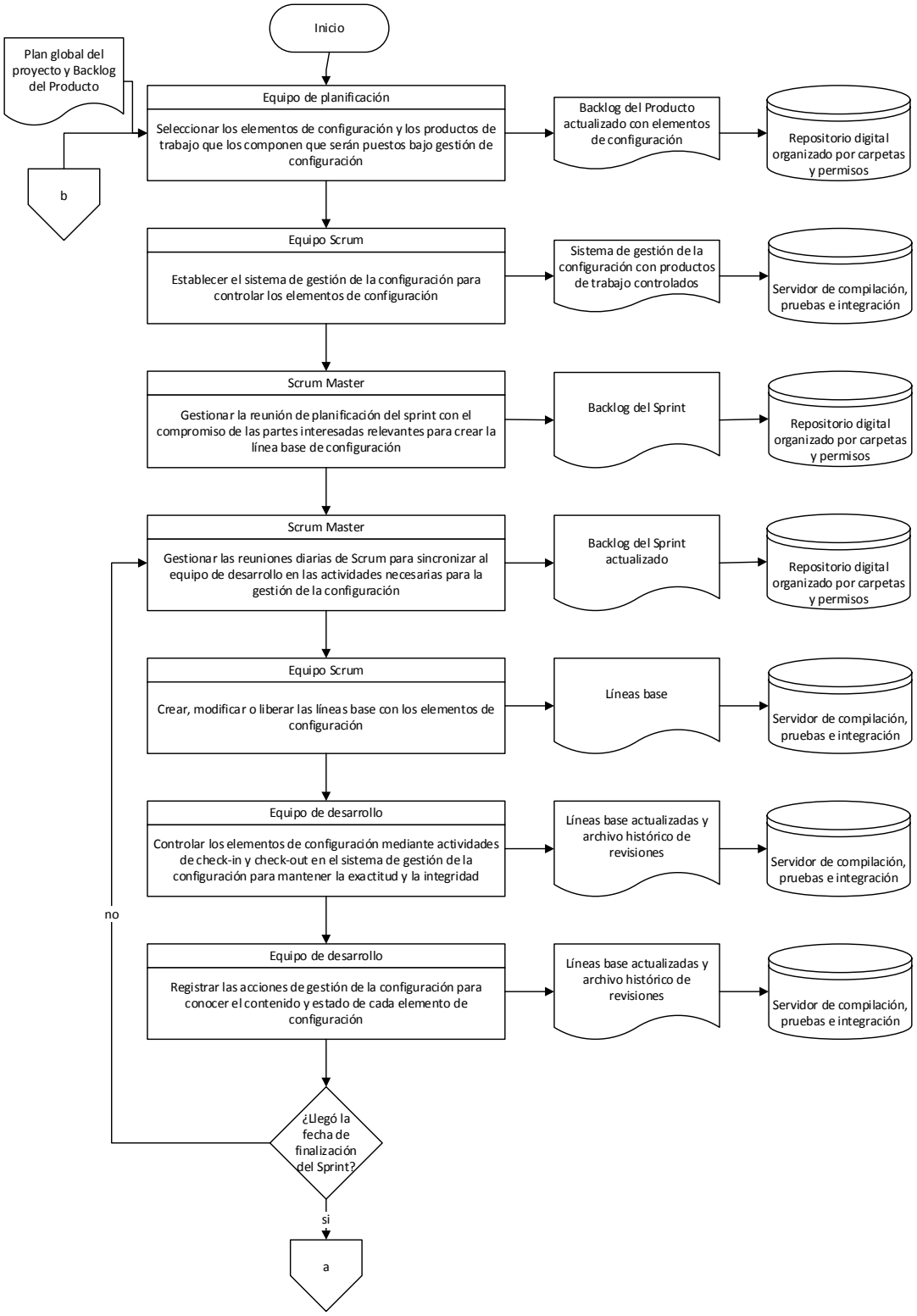
Equipo Scrum.- es el responsable de establecer el sistema de gestión de la configuración y crear, modificar o liberar las líneas base. Está conformado por el Dueño del Producto, el Scrum Master y el Equipo de Desarrollo.

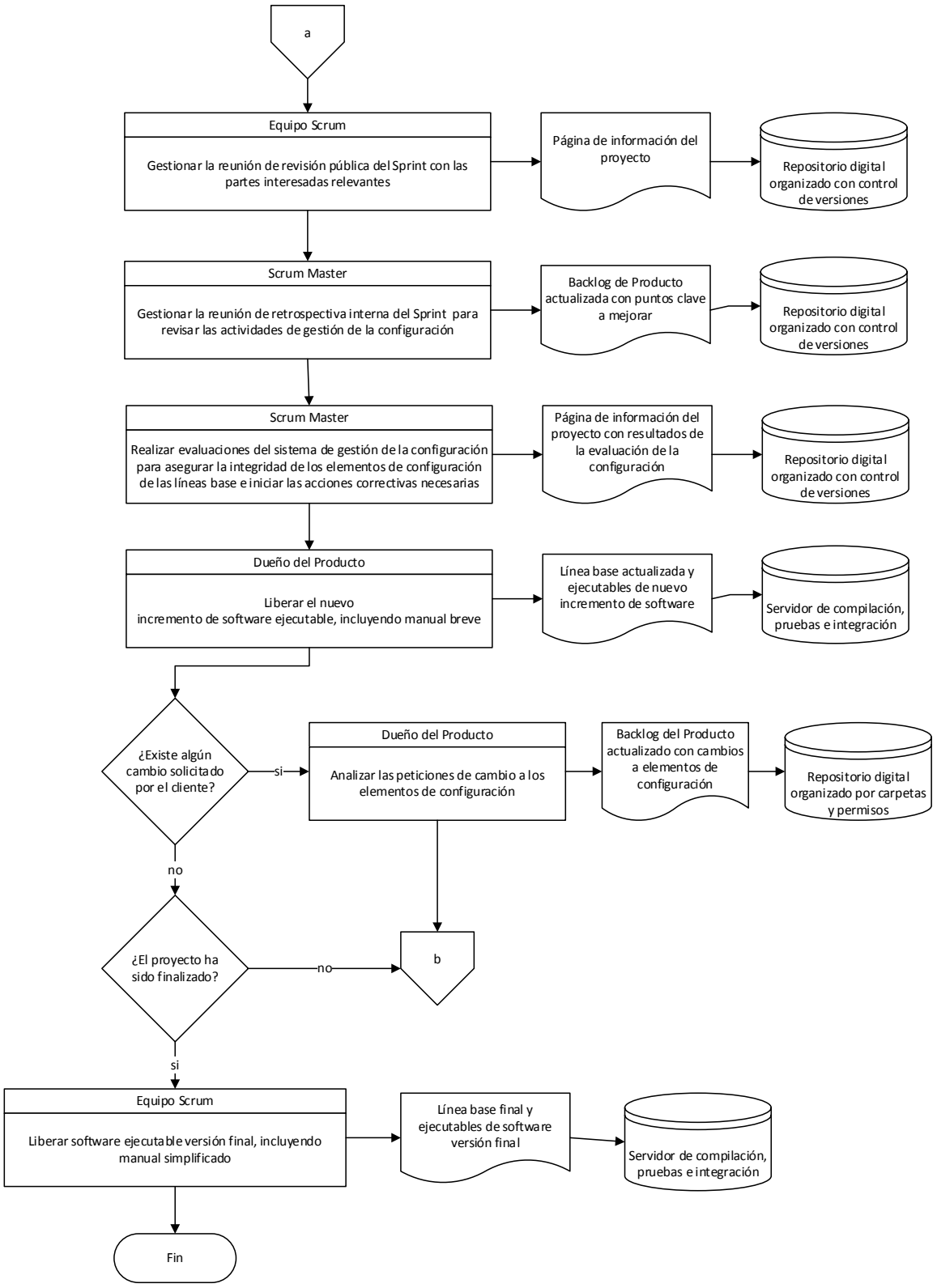
Equipo de desarrollo.- es el responsable de controlar y registrar los cambios en los elementos de configuración para mantener la exactitud e integridad de los productos de trabajo.

Scrum Master.- es el responsable de gestionar las reuniones del sprint y de resolver los impedimentos que afecten al equipo de desarrollo, y realizar evaluaciones del sistema de gestión de la configuración para asegurar la integridad de los elementos de configuración.

Dueño del Producto.- es el responsable de autorizar la liberación de los nuevos incrementos de software y analizar las peticiones de cambio a los elementos de configuración.

4. PROCEDIMIENTO




a
Equipo Scrum
Gestionar la reunión de revisión pública del Sprint con las partes interesadas relevantes
Página de información del proyecto
Repositorio digital organizado con control de versiones
Scrum Master
Gestionar la reunión de retrospectiva interna del Sprint para revisar las actividades de gestión de la configuración
Backlog de Producto actualizada con puntos clave a mejorar
Repositorio digital organizado con control de versiones
Scrum Master
Realizar evaluaciones del sistema de gestión de la configuración para asegurar la integridad de los elementos de configuración de las líneas base e iniciar las acciones correctivas necesarias
Página de información del proyecto con resultados de la evaluación de la configuración
Repositorio digital organizado con control de versiones
Dueño del Producto
Liberar el nuevo incremento de software ejecutable, incluyendo manual breve
Línea base actualizada y ejecutables de nuevo incremento de software
Servidor de compilación, pruebas e integración
¿Existe algún cambio solicitado por el cliente?
si
Dueño del Producto
Analizar las peticiones de cambio a los elementos de configuración
Backlog del Producto actualizado con cambios a elementos de configuración
Repositorio digital organizado por carpetas y permisos
no
¿El proyecto ha sido finalizado?
no
b
si
Equipo Scrum
Liberar software ejecutable versión final, incluyendo manual simplificado
Línea base final y ejecutables de software versión final
Servidor de compilación, pruebas e integración
Fin

5. DEFINICIONES

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Solicitante de la historia de usuario, Responsable de la historia de usuario, Estado de la historia de usuario, Fecha del estado de historia de usuario, ID producto de trabajo, ID seguimiento de errores.

Sistema de gestión de la configuración.- es un conjunto de elementos interrelacionados para controlar las líneas base con sus elementos de configuración y los productos de trabajo asociados para mantener la integridad y la calidad del software.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado), Fecha del estado de la tarea, ID producto de trabajo, ID de la historia de usuario asociada.

Líneas base.- es un conjunto de elementos de configuración revisados y aprobados que sirve como base para el siguiente incremento de software y cualquier cambio debe ser aprobado por el dueño del producto.

Página de Información del Proyecto.- es una página wiki con información actualizada del avance del proyecto con la siguiente información: objetivo y visión del proyecto, el Backlog del Producto, la velocidad estimada del equipo, el calendario con la duración del proyecto, presupuesto, la infraestructura de soporte, compromiso de las partes interesadas relevantes y el gráfico burndown de proyecto.

Ejecutables de software.- son los archivos de instalación del software que se entregan al cliente para obtener nuevas funcionalidades o la versión completa, junto con un manual simplificado de usuario.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

Servidor de compilación, pruebas e integración.- es una computadora con buena capacidad que permite al equipo de desarrollo realizar y registrar las pruebas, compilación e integración de los productos de trabajo asignados en el sprint.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P08-I1	Tasa de líneas base no autorizadas	Diaria	Número de Líneas base liberadas sin autorización / Número total de líneas base liberadas
P08-I2	Tasa de inconsistencias en los productos de trabajo	Diaria	Número de productos de trabajo con inconsistencias en la versión / Número de total de productos de trabajo

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P08-R1	Formato de Backlog de Producto
P08-R2	Formato de Backlog de Sprint
P08-R3	Estructura del Sistema de gestión de la configuración
P08-R4	Guía para guardar las líneas base
P08-R5	Formato de página wiki con información del proyecto
P08-R6	Guía de instalación del incremento de software
P08-R7	Estructura de organización del repositorio digital
P08-R8	Estructura del servidor de compilación, pruebas e integración

1. PROPÓSITO

Gestionar los requisitos de los productos incluyendo los requisitos técnicos y no técnicos, y asegurar la trazabilidad entre los requisitos, los planes y los productos de trabajo.

2. POLÍTICA

La gestión de los requisitos se la realizará de manera ágil mediante el seguimiento de los requisitos en un Backlog de Producto por medio del cual se comunican y se registran los compromisos del Equipo Scrum y se ajustan diariamente en base al progreso obtenido durante el Sprint, la trazabilidad y la consistencia entre los requisitos se revisarán en las reuniones de planificación y de revisión del sprint.

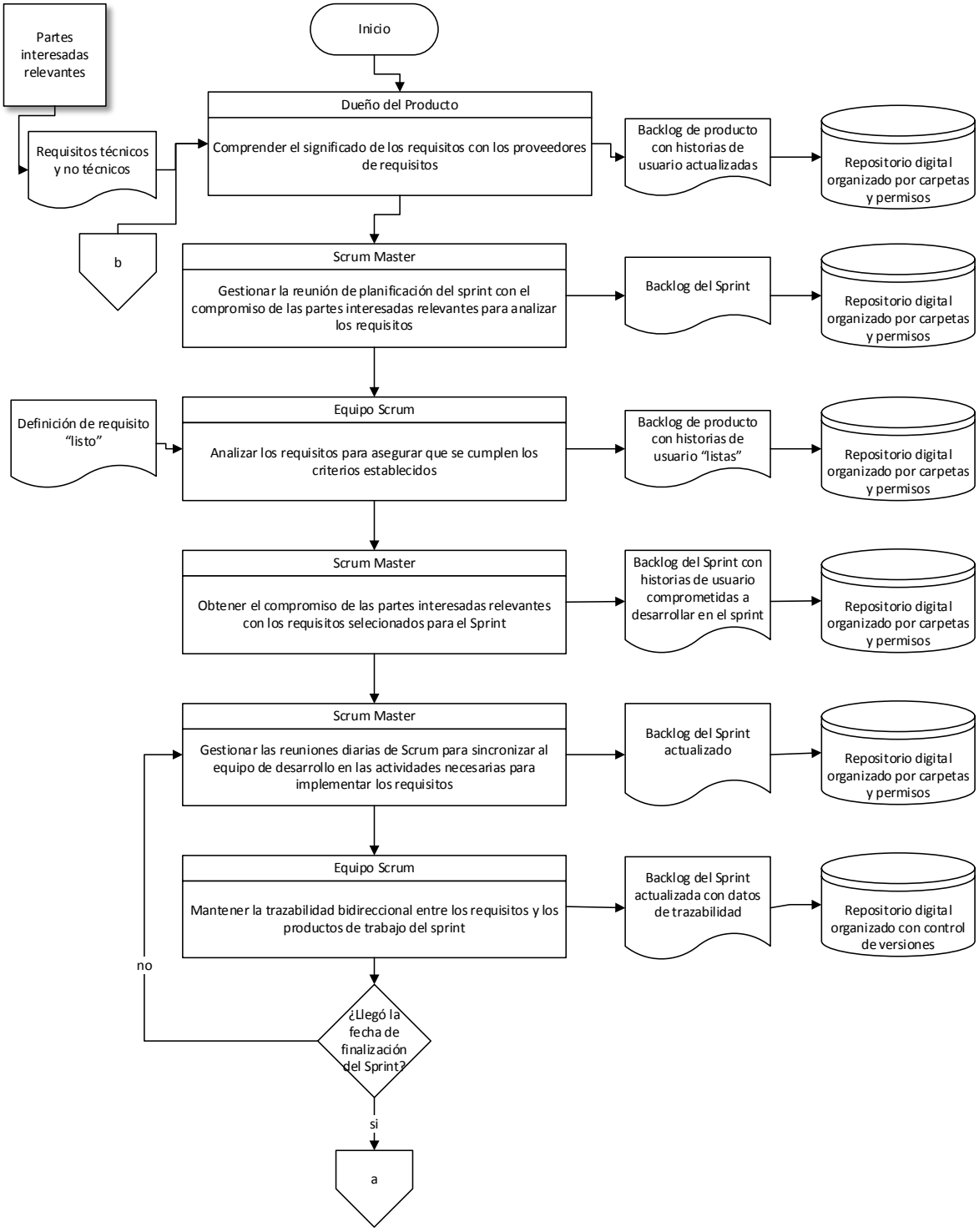
3. RESPONSABLES

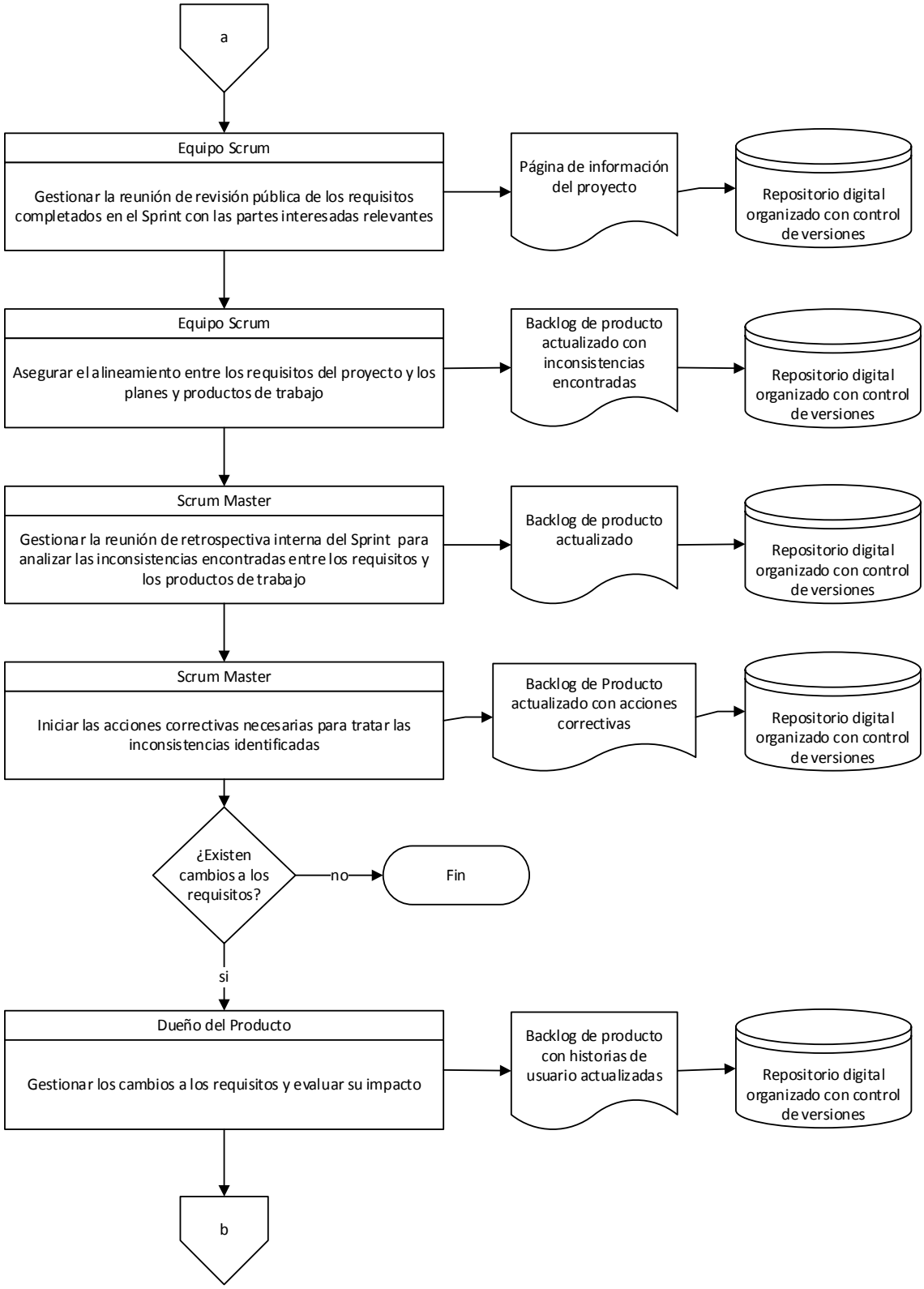
Dueño del Producto.- es el responsable de comunicarse con los proveedores de requisitos para comprender el significado completo de los requisitos y sus cambios.

Scrum Master.- es el responsable de obtener el compromiso de las partes interesadas relevantes con los requisitos seleccionados para ser desarrollados en el sprint e iniciar las acciones correctivas en caso de encontrar inconsistencias entre los requisitos y los productos de trabajo.

Equipo Scrum.- es el responsable de analizar los requisitos para revisar si cumplen con los criterios establecidos para su aceptación, mantener la trazabilidad bidireccional, y asegurar el alineamiento entre los requisitos y los productos de trabajo.

4. PROCEDIMIENTO




a
Equipo Scrum
Gestionar la reunión de revisión pública de los requisitos completados en el Sprint con las partes interesadas relevantes
Página de información del proyecto
Repositorio digital organizado con control de versiones
Equipo Scrum
Asegurar el alineamiento entre los requisitos del proyecto y los planes y productos de trabajo
Backlog de producto actualizado con inconsistencias encontradas
Repositorio digital organizado con control de versiones
Scrum Master
Gestionar la reunión de retrospectiva interna del Sprint para analizar las inconsistencias encontradas entre los requisitos y los productos de trabajo
Backlog de producto actualizado
Repositorio digital organizado con control de versiones
Scrum Master
Iniciar las acciones correctivas necesarias para tratar las inconsistencias identificadas
Backlog de Producto actualizado con acciones correctivas
Repositorio digital organizado con control de versiones
¿Existen cambios a los requisitos?
no
Fin
si
Dueño del Producto
Gestionar los cambios a los requisitos y evaluar su impacto
Backlog de producto con historias de usuario actualizadas
Repositorio digital organizado con control de versiones
b

5. DEFINICIONES

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Solicitante de la historia de usuario, Estado de la historia de usuario, Fecha del estado de historia de usuario, ID producto de trabajo, ID seguimiento de errores.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

Definición de requisito "listo".- es un lista de chequeo simple y corta para determinar cuándo una historia de usuario esta lista para ser seleccionada en el próximo sprint.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado), Fecha del estado de la tarea, ID producto de trabajo, ID de la historia de usuario asociada.

Página de información del proyecto.- es un reporte dirigido para lograr el compromiso e involucramiento de la alta dirección y muestra el progreso del proyecto al final de cada sprint, incluyendo elementos como: historias de usuario finalizadas, velocidad del equipo, gráficos burndown, impedimentos relevantes y resultados de las medidas establecidas.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P02-03-I1	Tasa de requisitos listos	Mensual	(Número de requisitos que cumplen con la definición de "listo" / Número total de requisitos del Backlog de Producto)
P02-03-I2	Tasa de cambios a los requisitos	Mensual	Número de requisitos que registran cambios / Número total de requisitos del Backlog de Producto

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P02-03-R1	Formato de Backlog de Producto
P02-03-R2	Estructura de organización del repositorio digital
P02-03-R3	Lista de Chequeo simple para la definición de requisito “listo”
P02-03-R4	Formato de Backlog de Sprint
P02-03-R5	Formato de página wiki con información del proyecto

1. PROPÓSITO

Especificar los objetivos de medición y el análisis de medidas para alinearlos con las necesidades de información y con los objetivos del proyecto.

2. POLÍTICA

La medición y análisis de los datos del proyecto se la realizará de manera ágil mediante el establecimiento de objetivos de medición y medidas durante el sprint inicial, en las reuniones diarias de scrum se recogerán, revisarán y analizarán los datos para construir los gráficos burndown de sprint y de proyecto, que servirán para monitorizar el progreso de los objetivos del negocio.

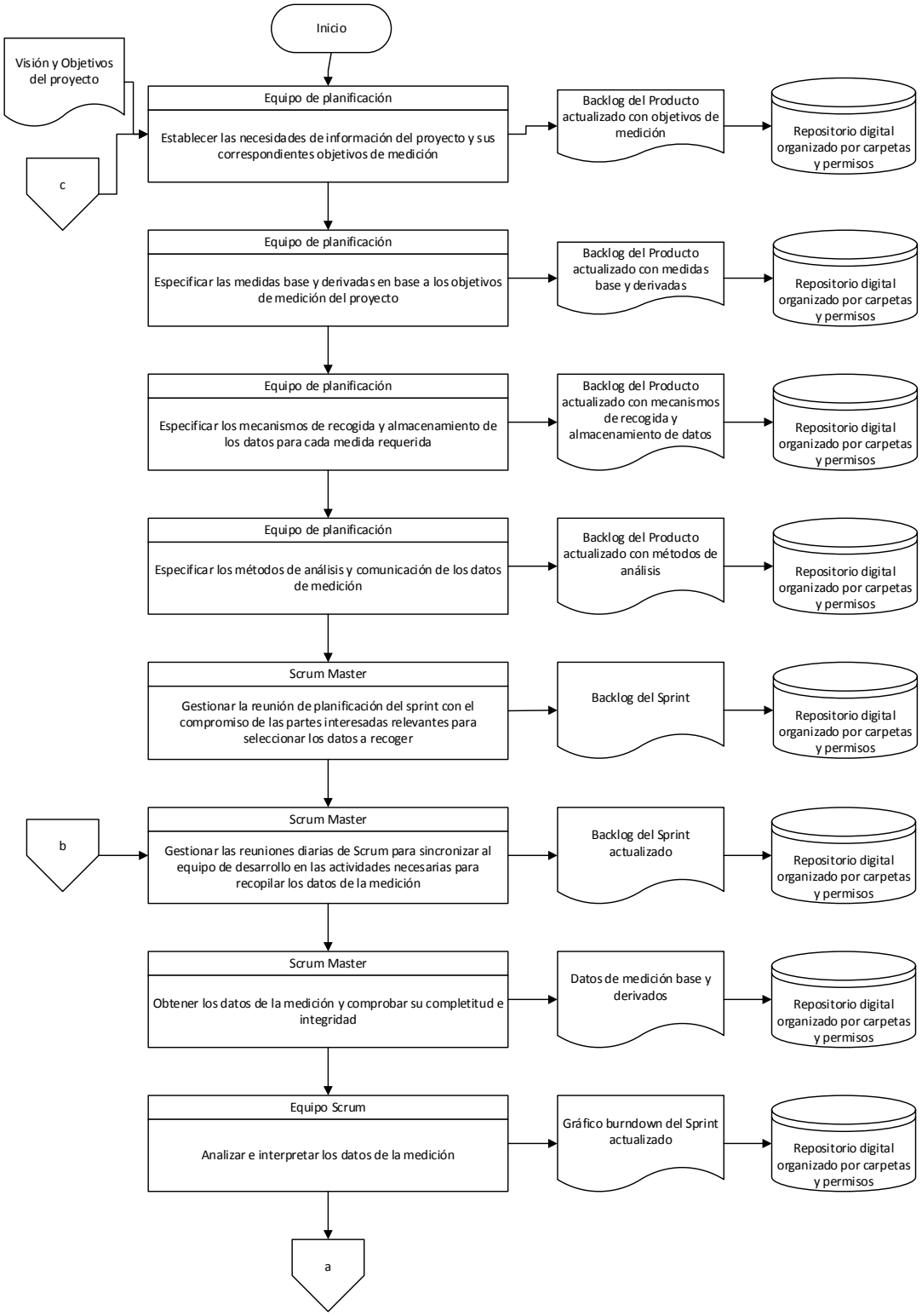
3. RESPONSABLES

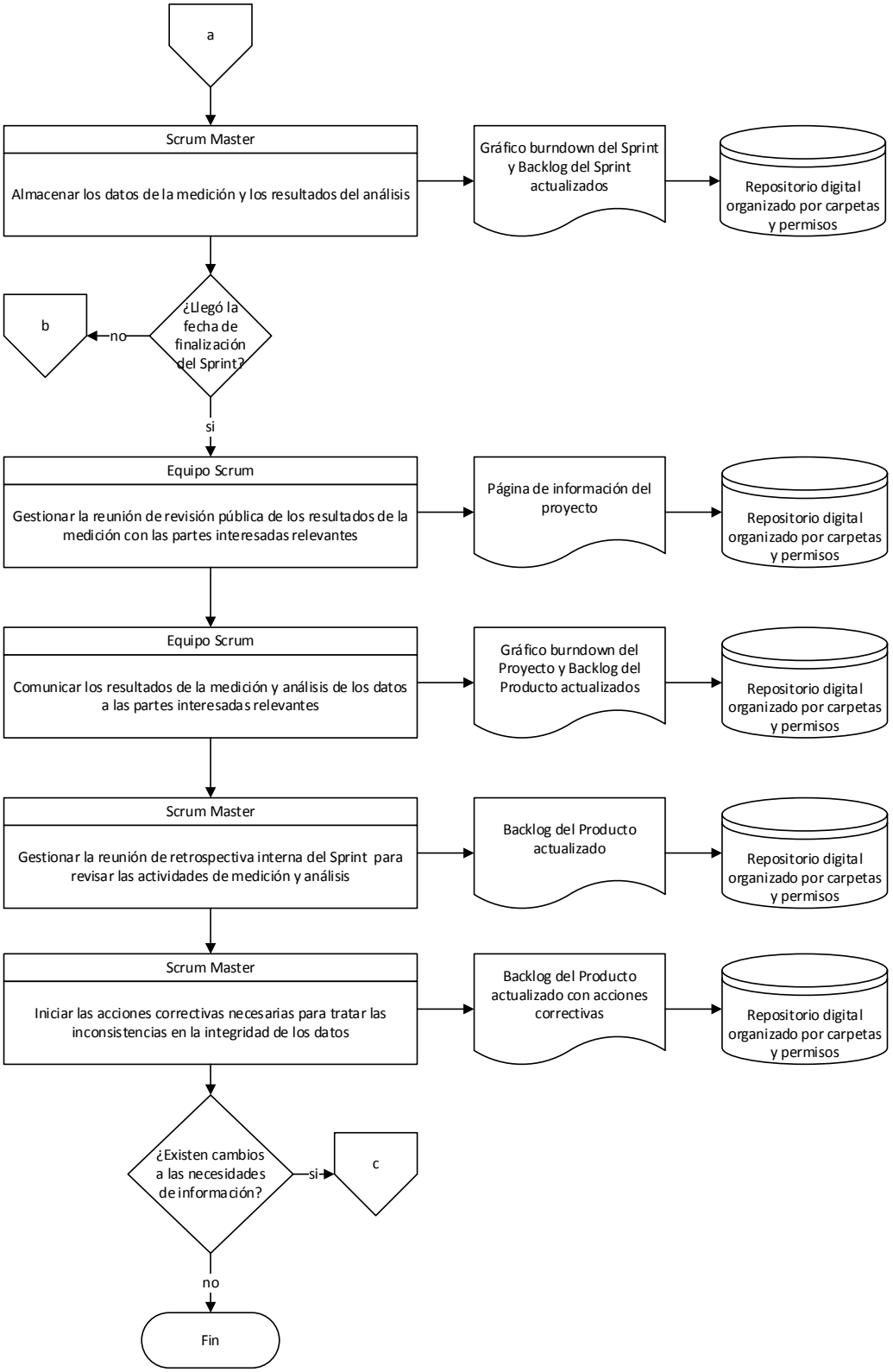
Equipo de planificación.- es responsable de establecer las necesidades de información y sus correspondientes objetivos de medición, especificar las medidas base y derivadas y sus mecanismos de recogida, almacenamiento, análisis y comunicación. Está conformado por el Jefe de Desarrollo y el Dueño del Producto.

Scrum Master.- es responsable de gestionar las reuniones de sincronización del equipo scrum, obtener los datos de la medición y almacenarlos junto con el análisis y los resultados, e iniciar las acciones correctivas en caso de encontrar inconsistencias en la información.

Equipo Scrum.- es responsable de analizar e interpretar los datos de la medición, y comunicar los resultados de la revisión a las partes interesadas relevantes. Está conformado por el Dueño del Producto, el Scrum Master y el Equipo de Desarrollo.

4. PROCEDIMIENTO




a
Scrum Master
Almacenar los datos de la medición y los resultados del análisis
Gráfico burndown del Sprint y Backlog del Sprint actualizados
Repositorio digital organizado por carpetas y permisos
¿Llegó la fecha de finalización del Sprint?
no
b
si
Equipo Scrum
Gestionar la reunión de revisión pública de los resultados de la medición con las partes interesadas relevantes
Página de información del proyecto
Repositorio digital organizado por carpetas y permisos
Equipo Scrum
Comunicar los resultados de la medición y análisis de los datos a las partes interesadas relevantes
Gráfico burndown del Proyecto y Backlog del Producto actualizados
Repositorio digital organizado por carpetas y permisos
Scrum Master
Gestionar la reunión de retrospectiva interna del Sprint para revisar las actividades de medición y análisis
Backlog del Producto actualizado
Repositorio digital organizado por carpetas y permisos
Scrum Master
Iniciar las acciones correctivas necesarias para tratar las inconsistencias en la integridad de los datos
Backlog del Producto actualizado con acciones correctivas
Repositorio digital organizado por carpetas y permisos
¿Existen cambios a las necesidades de información?
si
c
no
Fin

5. DEFINICIONES

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Solicitante de la historia de usuario, Estado de la historia de usuario, Fecha del estado de historia de usuario, ID producto de trabajo, ID seguimiento de errores.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado), Fecha del estado de la tarea, ID producto de trabajo, ID de la historia de usuario asociada.

Datos de medición base y derivados.- son todos los datos recogidos de las actividades del proyecto principalmente durante las reuniones diarias de Scrum.

Gráfico Burndown del Sprint.- en el eje X van los días del sprint, en el eje Y va el trabajo restante del Sprint en puntos de historia, en cada reunión diaria de scrum se actualiza los puntos en el gráfico según los puntos de historia completados.

Página de información del proyecto.- es un reporte dirigido para lograr el compromiso e involucramiento de la alta dirección y muestra el progreso del proyecto al final de cada sprint, incluyendo elementos como: historias de usuario finalizadas, velocidad del equipo, gráficos burndown, impedimentos relevantes y resultados de las medidas establecidas.

Gráfico Burndown del Proyecto.- en el eje X van los sprints del proyecto, en el eje Y va el trabajo restante del proyecto en puntos de historia, en cada reunión de planificación del Sprint se actualiza los puntos en el gráfico según los puntos de historia completados.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P09-I1	Tasa de medidas alineadas con los objetivos de medición	Mensual	Número de medidas base trazables a un objetivo de medición / Número total de medidas base
P09-I2	Tasa de consistencia de los datos recogidos	Mensual	Número de datos libres de inconsistencias / Número total de datos recogidos

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P09-R1	Formato de Backlog de Producto
P09-R2	Estructura de organización del repositorio digital
P09-R3	Formato de Backlog de Sprint
P09-R4	Formato de página wiki con información del proyecto
P09-R5	Formato de gráfico Burndown de proyecto
P09-R6	Formato de gráfico Burndown de Sprint
P09-R7	Formato de almacenamiento de los datos de medición

1. PROPÓSITO

Monitorizar y controlar el progreso de las actividades del proyecto para tomar las acciones correctivas apropiadas cuando el estado real del proyecto se desvíe significativamente de los valores esperados en el plan global del proyecto.

2. POLÍTICA

La monitorización y el control del proyecto se los realizará de manera ágil mediante la comprensión continua del rendimiento del proyecto, comparando el esfuerzo, el costo y el cronograma real con el plan del proyecto, se involucrará continuamente al cliente y a los usuarios finales en el desarrollo del incremento de software, si se detecta una desviación significativa en el rendimiento se tomarán ágilmente las acciones correctivas necesarias.

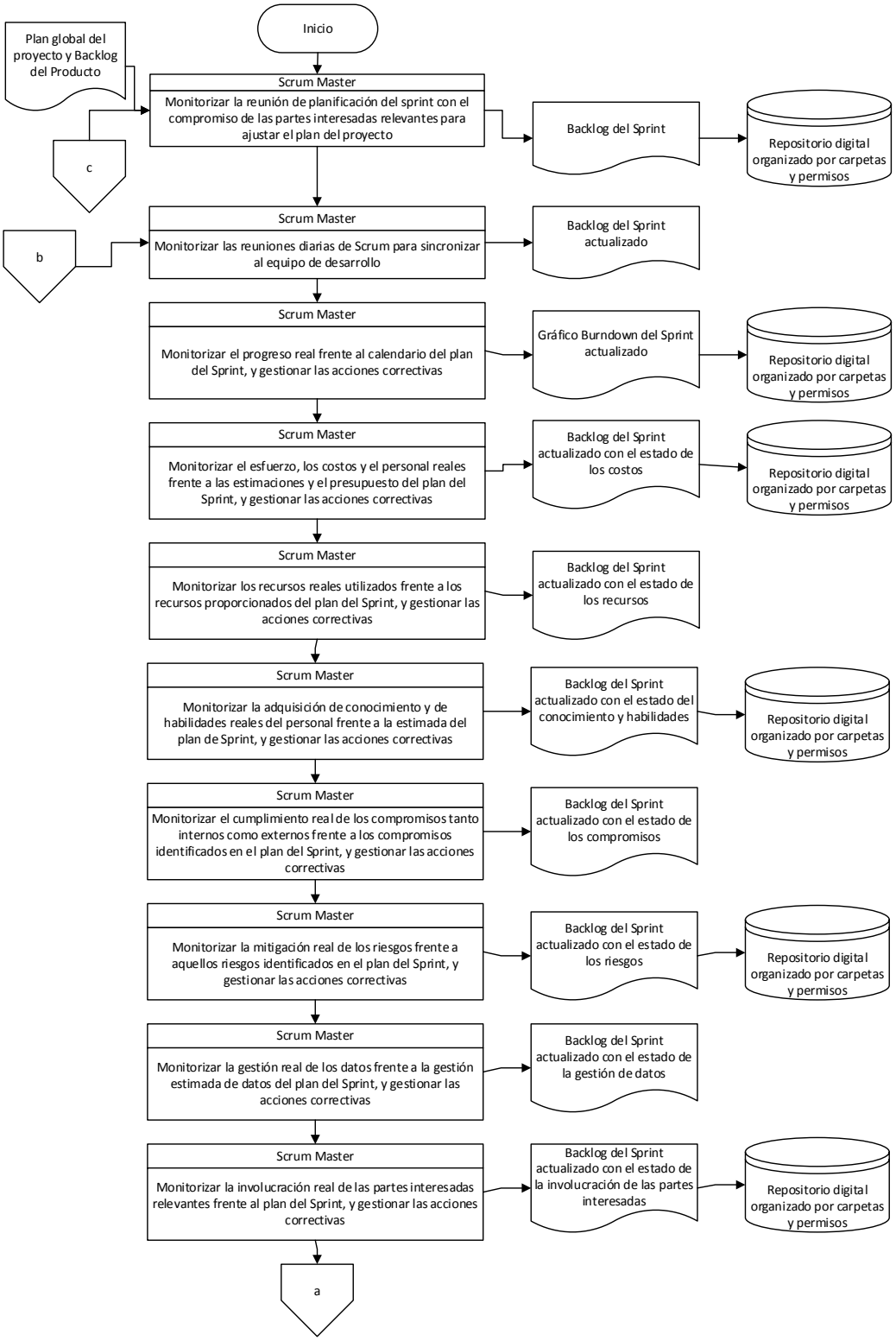
3. RESPONSABLES

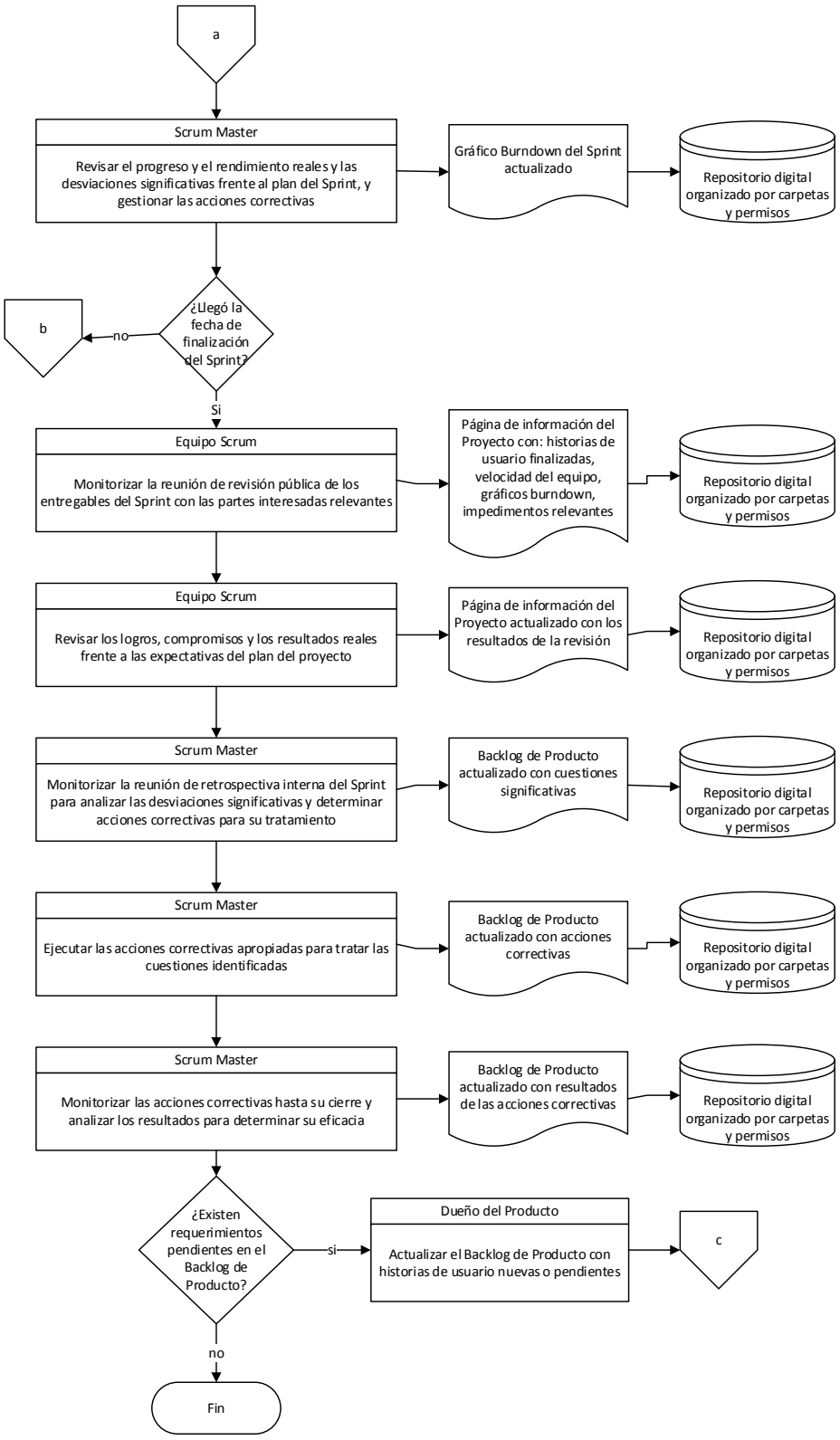
Scrum Master.- es responsable de monitorizar el progreso real del proyecto frente al plan global del proyecto, en especial es responsable de resolver los impedimentos que pueda tener el equipo de desarrollo tan pronto como sea posible, ejecutando las acciones correctivas apropiadas.

Equipo Scrum.- es responsable de llevar a cabo las acciones correctivas necesarias sobre las cuestiones identificadas. Está conformado por el Dueño del Producto, el Scrum Master y el Equipo de Desarrollo.

Dueño del Producto.- es responsable de mantener organizado, priorizado y actualizado el Backlog de Producto incluyendo o modificando las historias de usuario según se requiera.

4. PROCEDIMIENTO




a
Scrum Master
Revisar el progreso y el rendimiento reales y las desviaciones significativas frente al plan del Sprint, y gestionar las acciones correctivas
Gráfico Burndown del Sprint actualizado
Repositorio digital organizado por carpetas y permisos
¿Llegó la fecha de finalización del Sprint?
no
b
Si
Equipo Scrum
Monitorizar la reunión de revisión pública de los entregables del Sprint con las partes interesadas relevantes
Página de información del Proyecto con: historias de usuario finalizadas, velocidad del equipo, gráficos burndown, impedimentos relevantes
Repositorio digital organizado por carpetas y permisos
Equipo Scrum
Revisar los logros, compromisos y los resultados reales frente a las expectativas del plan del proyecto
Página de información del Proyecto actualizado con los resultados de la revisión
Repositorio digital organizado por carpetas y permisos
Scrum Master
Monitorizar la reunión de retrospectiva interna del Sprint para analizar las desviaciones significativas y determinar acciones correctivas para su tratamiento
Backlog de Producto actualizado con cuestiones significativas
Repositorio digital organizado por carpetas y permisos
Scrum Master
Ejecutar las acciones correctivas apropiadas para tratar las cuestiones identificadas
Backlog de Producto actualizado con acciones correctivas
Repositorio digital organizado por carpetas y permisos
Scrum Master
Monitorizar las acciones correctivas hasta su cierre y analizar los resultados para determinar su eficacia
Backlog de Producto actualizado con resultados de las acciones correctivas
Repositorio digital organizado por carpetas y permisos
¿Existen requerimientos pendientes en el Backlog de Producto?
si
Dueño del Producto
Actualizar el Backlog de Producto con historias de usuario nuevas o pendientes
c
no
Fin

5. DEFINICIONES

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Estado de la historia de usuario, Seguimiento de errores.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado).

Gráfico Burndown del Proyecto.- en el eje X van los sprints del proyecto, en el eje Y va el trabajo restante del proyecto en puntos de historia, en cada reunión de planificación del Sprint se actualiza los puntos en el gráfico según los puntos de historia completados.

Gráfico Burndown del Sprint.- en el eje X van los días del sprint, en el eje Y va el trabajo restante del Sprint en puntos de historia, en cada reunión diaria de scrum se actualiza los puntos en el gráfico según los puntos de historia completados.

Página de información del proyecto.- es un reporte dirigido para lograr el compromiso e involucramiento de la alta dirección y muestra el progreso del proyecto al final de cada sprint, incluyendo elementos como: historias de usuario finalizadas, velocidad del equipo, gráficos burndown, impedimentos relevantes y resultados de las medidas establecidas.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P02-02-I1	Tasa real de Burndown de Sprint	Diaria	(Puntos de historia completados reales / días transcurridos del Sprint)
P02-02-I2	Tasa real de Burndown del Proyecto	Mensual	(Puntos de historia completados reales / días transcurridos del Proyecto)
P02-02-I3	Tasa de finalización de acciones correctivas	Mensual	Número de acciones correctivas cerradas / número total de acciones correctivas

P02-02-I4	Tasa de cumplimiento de actividades planificadas	Mensual	Número de actividades cumplidas / Número total de actividades planificadas

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P02-02-R1	Formato de Backlog de Producto
P02-02-R2	Formato de Backlog de Sprint
P02-02-R3	Formato de gráfico Burndown de proyecto
P02-02-R4	Formato de gráfico Burndown de Sprint
P02-02-R5	Formato para reunión de retrospectiva del Sprint
P02-02-R6	Formato para reunión de revisión del Sprint
P02-02-R7	Estructura de organización del repositorio digital
P02-02-R8	Formato de página wiki con información del proyecto

1. PROPÓSITO

Establecer la planificación para las actividades del proyecto, definidas en las siguientes fases del ciclo de vida del proyecto:

- Planificación: se establece la visión del proyecto, los requerimientos de las partes interesadas relevantes, y los fondos presupuestarios para la ejecución del proyecto.
- Puesta en escena: identifica y prioriza los requerimientos para el siguiente sprint, descompone el Backlog de producto en Sprints acorde con la priorización previa considerando la productividad del equipo de desarrollo.
- Desarrollo: implementa el sistema en sprints de 30 días, y al final del sprint un incremento del producto es mostrado a las partes interesadas relevantes.
- Entrega: cubre el despliegue e implementación de las funcionalidades completadas.

2. POLÍTICA

La planificación se la realizará de manera ágil llevando a cabo un desarrollo incremental y evolutivo en donde los equipos planifican, monitorizan y reconcilian los planes en cada iteración. En el Sprint inicial se organiza el entorno de trabajo, se diseña la arquitectura básica y se definen los atributos de calidad y funcionalidad del producto.

3. RESPONSABLES

Equipo de planificación.- es responsable de establecer la visión y los objetivos del proyecto, el Backlog de producto de alto nivel, los recursos, el presupuesto y calendario inicial del proyecto. Está conformado por el Jefe de Desarrollo y el Dueño del Producto.

Jefe de Desarrollo.- es responsable de establecer la página de información del proyecto y proveer de los recursos necesarios para llevar a cabo el proyecto.

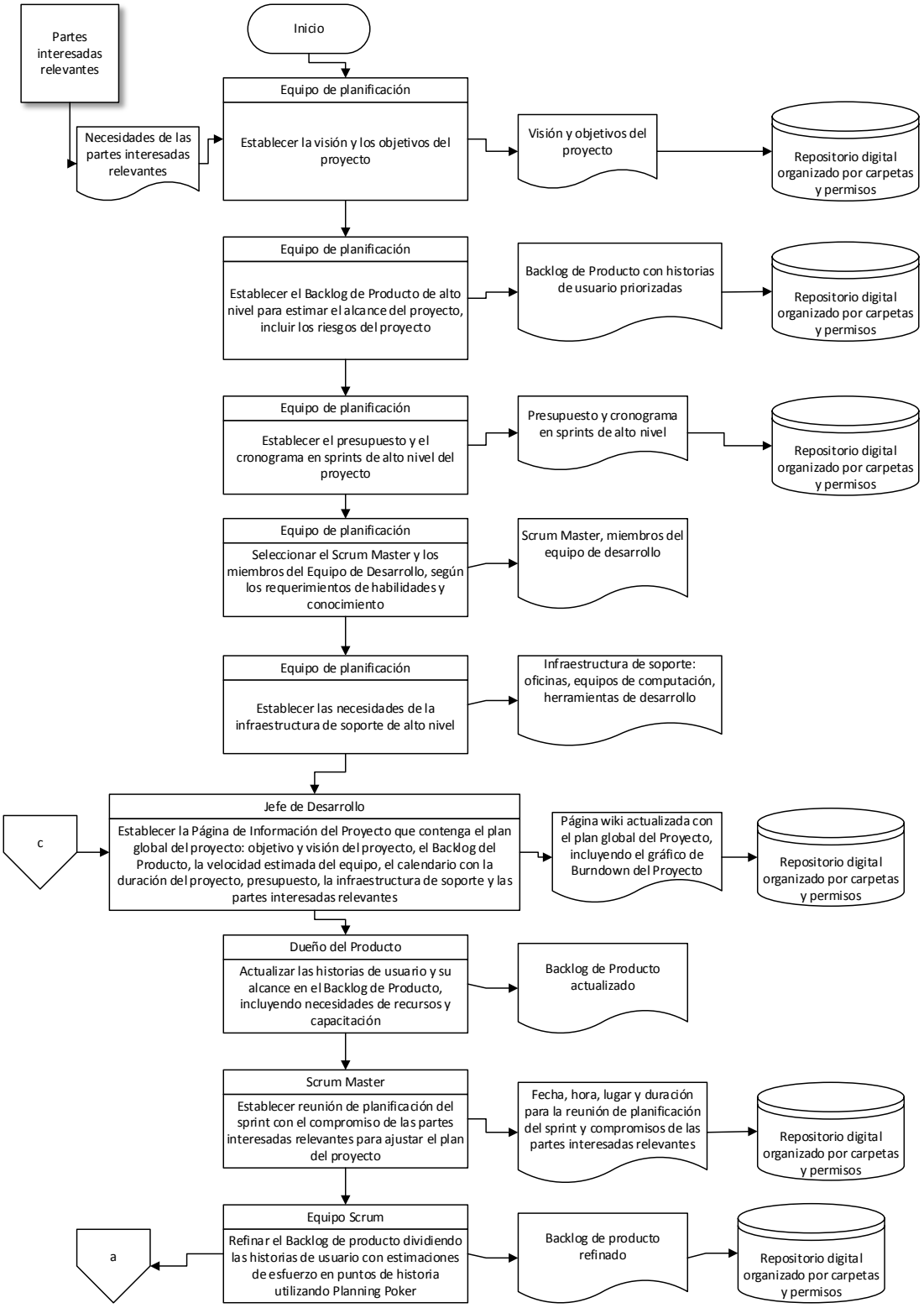
Dueño del Producto.- es responsable de mantener actualizado el Backlog de Producto y autorizar la liberación de incrementos entregables de software.

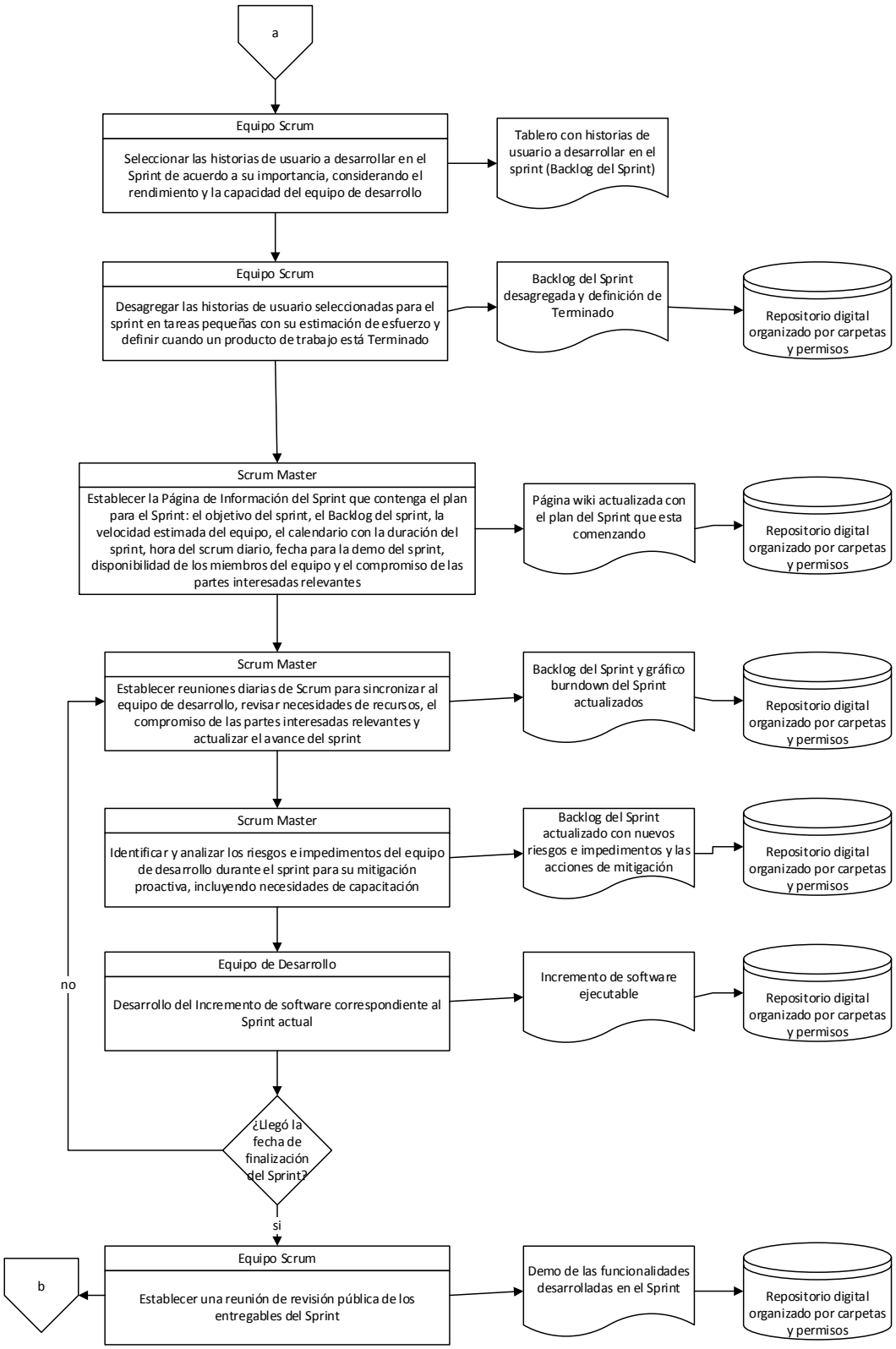
Scrum Master.- es responsable de establecer las reuniones de planificación del sprint, las reuniones diarias de scrum, la reunión de retrospectiva del sprint y actualizar la página de información del Sprint.

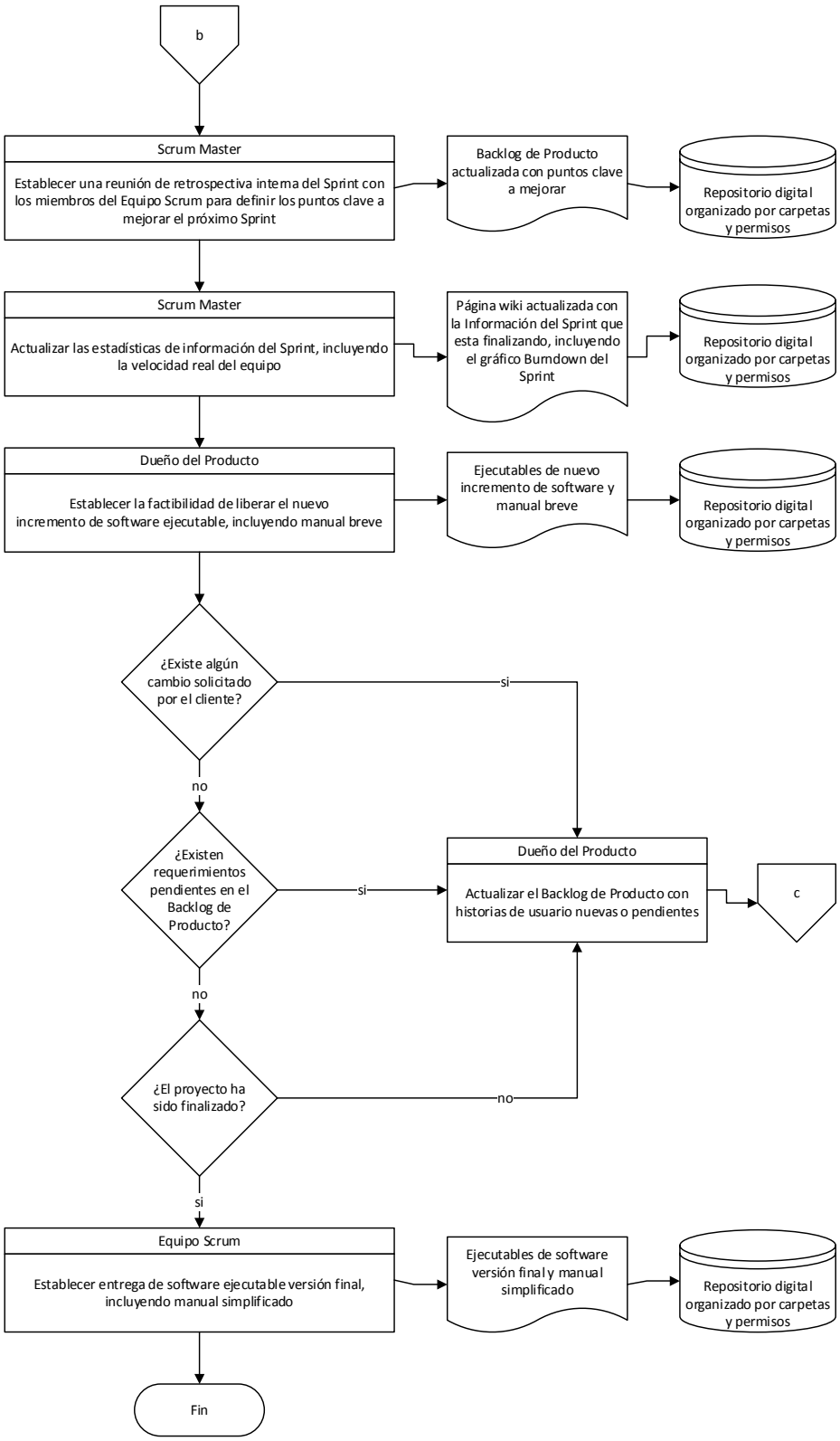
Equipo Scrum.- es responsable de refinar el Backlog de producto, seleccionar y desagregar las historias de usuario a desarrollar en el sprint, llevar a cabo la reunión de revisión de los entregables del sprint. Está conformado por el Dueño del Producto, el Scrum Master y el Equipo de Desarrollo.

Equipo de Desarrollo.- es responsable de desarrollar el incremento de software del sprint.

4. PROCEDIMIENTO




a
Equipo Scrum
Seleccionar las historias de usuario a desarrollar en el Sprint de acuerdo a su importancia, considerando el rendimiento y la capacidad del equipo de desarrollo
Tablero con historias de usuario a desarrollar en el sprint (Backlog del Sprint)
Equipo Scrum
Desagregar las historias de usuario seleccionadas para el sprint en tareas pequeñas con su estimación de esfuerzo y definir cuando un producto de trabajo está Terminado
Backlog del Sprint desagregada y definición de Terminado
Repositorio digital organizado por carpetas y permisos
Scrum Master
Establecer la Página de Información del Sprint que contenga el plan para el Sprint: el objetivo del sprint, el Backlog del sprint, la velocidad estimada del equipo, el calendario con la duración del sprint, hora del scrum diario, fecha para la demo del sprint, disponibilidad de los miembros del equipo y el compromiso de las partes interesadas relevantes
Página wiki actualizada con el plan del Sprint que esta comenzando
Repositorio digital organizado por carpetas y permisos
Scrum Master
Establecer reuniones diarias de Scrum para sincronizar al equipo de desarrollo, revisar necesidades de recursos, el compromiso de las partes interesadas relevantes y actualizar el avance del sprint
Backlog del Sprint y gráfico burndown del Sprint actualizados
Repositorio digital organizado por carpetas y permisos
Scrum Master
Identificar y analizar los riesgos e impedimentos del equipo de desarrollo durante el sprint para su mitigación proactiva, incluyendo necesidades de capacitación
Backlog del Sprint actualizado con nuevos riesgos e impedimentos y las acciones de mitigación
Repositorio digital organizado por carpetas y permisos
Equipo de Desarrollo
Desarrollo del Incremento de software correspondiente al Sprint actual
Incremento de software ejecutable
Repositorio digital organizado por carpetas y permisos
no
¿Llegó la fecha de finalización del Sprint?
si
b
Equipo Scrum
Establecer una reunión de revisión pública de los entregables del Sprint
Demo de las funcionalidades desarrolladas en el Sprint
Repositorio digital organizado por carpetas y permisos


b
Scrum Master
Establecer una reunión de retrospectiva interna del Sprint con los miembros del Equipo Scrum para definir los puntos clave a mejorar el próximo Sprint
Backlog de Producto actualizada con puntos clave a mejorar
Repositorio digital organizado por carpetas y permisos
Scrum Master
Actualizar las estadísticas de información del Sprint, incluyendo la velocidad real del equipo
Página wiki actualizada con la Información del Sprint que esta finalizando, incluyendo el gráfico Burndown del Sprint
Repositorio digital organizado por carpetas y permisos
Dueño del Producto
Establecer la factibilidad de liberar el nuevo incremento de software ejecutable, incluyendo manual breve
Ejecutables de nuevo incremento de software y manual breve
Repositorio digital organizado por carpetas y permisos
¿Existe algún cambio solicitado por el cliente?
si
no
¿Existen requerimientos pendientes en el Backlog de Producto?
si
Dueño del Producto
Actualizar el Backlog de Producto con historias de usuario nuevas o pendientes
c
no
¿El proyecto ha sido finalizado?
no
si
Equipo Scrum
Establecer entrega de software ejecutable versión final, incluyendo manual simplificado
Ejecutables de software versión final y manual simplificado
Repositorio digital organizado por carpetas y permisos
Fin

5. DEFINICIONES

Backlog de Producto priorizado.- debe contener la siguiente información: ID de la historia de usuario, Nombre de la historia de usuario, Estimación en puntos de historia, Cómo Probarlo "Cómo X, quiero Y, así que Z", Observaciones, Tipo de la historia de usuario, Solicitante de la historia de usuario, Responsable de la historia de usuario, Estado de la historia de usuario, Fecha del estado de historia de usuario, ID producto de trabajo, ID seguimiento de errores.

Planning Poker.- es un método de estimación de historias de usuario en base a puntos de historia con 13 cartas impresas con los siguientes valores: 0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100, ?, café; mientras más alto el valor de la carta más complejidad tendrá el desarrollo de la historia de usuario para un miembro del equipo Scrum, se determina la estimación de la historia de usuario al llegar a un consenso entre todos los miembros del equipo.

Backlog de Sprint priorizado.- debe contener la siguiente información de las tareas del Sprint: ID de la tarea, Nombre de la tarea, Estimación de la tarea, Responsable de la tarea, Cómo Probarlo, Observaciones, Tipo de la tarea, Estado de la tarea (Pendiente, En Curso, Terminado), Fecha del estado de la tarea, ID producto de trabajo, ID de la historia de usuario asociada.

Gráfico Burndown del Proyecto.- en el eje X van los sprints del proyecto, en el eje Y va el trabajo restante del proyecto en puntos de historia, en cada reunión de planificación del Sprint se actualiza los puntos en el gráfico según los puntos de historia completados.

Gráfico Burndown del Sprint.- en el eje X van los días del sprint, en el eje Y va el trabajo restante del Sprint en puntos de historia, en cada reunión diaria de scrum se actualiza los puntos en el gráfico según los puntos de historia completados.

Página de Información del Proyecto.- es una página wiki con información actualizada del avance del proyecto con la siguiente información: objetivo y visión del proyecto, el Backlog del Producto, la velocidad estimada del equipo, el calendario con la duración del proyecto, presupuesto, la infraestructura de soporte, compromiso de las partes interesadas relevantes y el gráfico burndown de proyecto.

Definición de Terminado.- es un lista de chequeo simple y corta para determinar cuándo una historia de usuario está completamente terminada y puede ser incorporada en la siguiente entrega.

Página de Información del Sprint.- es un página wiki con información actualizada del avance del Sprint con la siguiente información: el objetivo del sprint, el Backlog del sprint, la velocidad estimada del equipo, el calendario con la duración del sprint, hora del scrum

diario, fecha para la demo del sprint, disponibilidad de los miembros del equipo y el compromiso de las partes interesadas relevantes.

Reunión de retrospectiva del Sprint.- es una reunión para determinar las oportunidades de mejora en base a la siguiente información: qué hicimos bien, qué pudo haberse hecho mejor, qué necesita ser mejorado. Como resultado de esta reunión se actualiza el Backlog de producto con los puntos a mejorar en el siguiente sprint.

Ejecutables de software.- son los archivos de instalación del software que se entregan al cliente para obtener nuevas funcionalidades o la versión completa, junto con un manual simplificado de usuario.

Repositorio digital.- es la unidad de almacenamiento digital centralizada que permite la organización y control de archivos y carpetas con los respectivos niveles de accesos.

Visión y objetivos del proyecto.- son los resultados que se esperan alcanzar al finalizar el proyecto.

6. INDICADORES

No	Nombre	Frecuencia	Fórmula de cálculo
P02-01-I1	Tasa de planificación de hitos del proyecto	Mensual	Número de hitos planificados / número total de hitos en el ciclo de vida del proyecto
P02-01-I2	Velocidad estimada del equipo de desarrollo	Mensual	(Velocidad real del equipo / días-persona disponibles reales)*días-persona disponibles estimado

7. REFERENCIAS

CODIGO	NOMBRE DOCUMENTO
P02-01-R1	Formato de Backlog de Producto
P02-01-R2	Guía de Planning Poker
P02-01-R3	Formato de Backlog de Sprint
P02-01-R4	Formato de gráfico Burndown de proyecto
P02-01-R5	Formato de gráfico Burndown de Sprint
P02-01-R6	Formato de página wiki con información del proyecto
P02-01-R7	Formato de página wiki con información del sprint
P02-01-R8	Lista de Chequeo simple para la definición de terminado
P02-01-R9	Formato para reunión de retrospectiva del Sprint

P02-01-R10	Guía de instalación del software
P02-01-R11	Formato del manual simplificado de usuario
P02-01-R12	Formato seguimiento de errores
P02-01-R13	Formato productos de trabajo
P02-01-R14	Estructura de organización del repositorio digital