Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS

Andrea Isabel Marroquín Guzmán
Asesorado por el Ing. Luis Alberto Arias Solórzano

Guatemala, febrero de 2018

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

ANDREA ISABEL MARROQUÍN GUZMÁN
ASESORADO POR EL ING. LUIS ALBERTO ARIAS SOLÓRZANO

AL CONFERÍRSELE EL TÍTULO DE

INGENIERA EN CIENCIAS Y SISTEMAS

GUATEMALA, FEBRERO DE 2018

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Pedro Antonio Aguilar Polanco
VOCAL I	Ing. Ángel Roberto Sic García
VOCAL II	Ing. Pablo Christian de León Rodríguez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Oscar Humberto Galicia Nuñez
VOCAL V	Br. Carlos Enrique Gómez Donis
SECRETARIA	Inga. Lesbia Magalí Herrera López

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Pedro Antonio Aguilar Polanco
EXAMINADORA	Inga. Mirna Ivonne Aldana
EXAMINADOR	Ing. César Augusto Fernández Cáceres
EXAMINADOR	Ing. Luis Fernando Espino Barrios
SECRETARIA	Inga. Lesbia Magalí Herrera López

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha 01 de mayo de 2017.

Andrea Isabel Marroquín Guzmán

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas

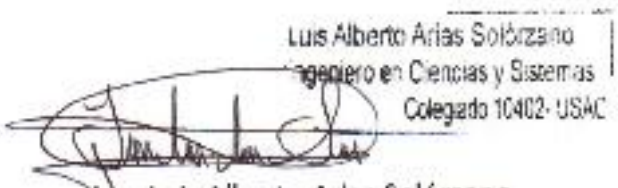
Guatemala, 31 de Julio de 2017

Ingeniero
Carlos Alfredo Azurdia Morales
Coordinador del Área de Trabajos de Graduación

Ingeniero Azurdia:

Por medio de la presente hago constar que la estudiante **ANDREA ISABEL MARROQUÍN GUZMÁN**, quién se identifica con número de carné **200511851**, ha concluido el trabajo final de investigación titulado "***ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS***", habiendo cumplido con el alcance establecido.

Atentamente,



Ing. Luis Alberto Arias Solórzano
Asesor de trabajo de graduación
Colegiado no. 10402

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingenieria en Ciencias y Sistemas

Guatemala, 18 de Octubre de 2017

Ingeniero
Marlon Antonio Pérez Türk
Director de la Escuela de Ingenieria
En Ciencias y Sistemas

Respetable Ingeniero Pérez:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación de la estudiante **ANDREA ISABEL MARROQUÍN GUZMÁN** con camé **200511851 y CUI 1823 16092 0101**, titulado **"ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS"** y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación

ESCUELA DE INGENIERÍA EN CIENCIAS Y SISTEMAS

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN
CIENCIAS Y SISTEMAS
TEL: 24188000 Ext. 1534

El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor con el visto bueno del revisor y del Licenciado en Letras, del trabajo de graduación ***"ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS"****, realizado por el estudiante ANDREA ISABEL MARROQUÍN GUZMÁN aprueba el presente trabajo y solicita la autorización del mismo.*

"ID Y ENSEÑAD A TODOS"

DIRECCION DE INGENIERIA EN CIENCIAS Y SISTEMAS

Ing. Marlon Antonio Pérez Turk
Director
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 13 de febrero de 2018

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 062.2018

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **ANÁLISIS DE METODOLOGÍAS Y PLATAFORMAS DE DESARROLLO DE SOFTWARE PARA LA IMPLEMENTACIÓN DE BUENAS PRÁCTICAS EN LA GESTIÓN DE PROYECTOS TICS,** presentado por la estudiante universitaria: **Andrea Isabel Marroquín Guzmán,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Pedro Antonio Aguilar Polanco
Decano

Guatemala, febrero de 2018

/gdech

ACTO QUE DEDICO A:

Mis padres	Aura Marina y Daniel Humberto, por su amor y apoyo incondicional.
Mis hermanas	Aura y Gabriela, por su cariño y comprensión.
Mis amigos	Por su amistad y buenos momentos que pasamos juntos.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por haberme dado la oportunidad de formar parte de esta casa de estudios.
Facultad de Ingeniería	Por brindarme los conocimientos para convertirme en una profesional.
Ing. Luis Alberto Arias Solórzano	Por sus enseñanzas, colaboración y apoyo durante el desarrollo de este trabajo de graduación.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

GLOSARIO

Desarrollar Realizar o llevar a cabo algo.

Diseño Concepción original de un objeto u obra destinada a la producción en serie.

Empresa Unidad de organización dedicada a actividades industriales, mercantiles o de prestación de servicios con fines lucrativos.

Estándar Que sirve como tipo, modelo, norma, patrón o referencia.

Fase Cada uno de los distintos estados sucesivos de una doctrina o negocio.

Framework Marco de trabajo, estructura real o conceptual destinada a servir de soporte o guía para la construcción de algo que expande la estructura de algo útil.

Master Persona quien tiene el control sobre una situación en particular.

Metodología Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal.

Modelo	Arquetipo o punto de referencia para imitarlo o reproducirlo.
Organización	Asociación de personas regulada por un conjunto de normas en función de determinados fines.
Práctica	Dicho de un conocimiento: Que enseña el modo de hacer algo.
Proyecto	Conjunto de actividades que se desarrolladas por una persona o entidad para alcanzar un determinado objetivo.
Recurso	Conjunto de elementos disponibles para resolver una necesidad.
Software	Conjunto de programas, instrucciones y reglas informáticas que permiten ejecutar distintas tareas en una computadora.

RESUMEN

En el presente trabajo de investigación se analiza el uso de metodologías, buenas prácticas, normas y estándares en la gestión y desarrollo de proyectos de software. Con ello, se identifican los componentes esenciales que permitan garantizar el éxito de estos, tomando en cuenta la experiencia de expertos en las diferentes áreas.

La obtención de los datos que se presentan a continuación, extraída por medio de encuestas realizadas a personas expertas en el ámbito de la gestión y administración de proyectos de software en empresas importantes de la región. Con base a su conocimiento y experiencia, es posible identificar los factores más importantes a considerar durante la ejecución del desarrollo de proyectos de software y los recursos vitales para garantizar el éxito y mejora en estos.

Para determinar el éxito de un proyecto, existen factores que se deben considerar al iniciar un proyecto de software, donde existen diferentes fases e involucrados a lo largo de su desarrollo. Cada uno de los responsables debe conocer las tareas que debe realizar y comprometerse con ellas.

Considerando los recursos y restricciones que se poseen, y tomando en cuenta la necesidad que se desea satisfacer, es importante hacer énfasis en las fases iniciales para la definición de procesos y toma de decisiones en la gestión del proyecto a ejecutar.

Luego del análisis obtenido, tomando en cuenta la teoría y la experiencia de los involucrados, es posible identificar los factores esenciales que deben tomarse en cuenta para una buena administración de proyectos, y garantizar el éxito

cumpliendo con las solicitudes y restricciones definidas por todas las partes involucradas. La falta de aplicación de buenas prácticas, estándares y normas afecta considerablemente el desarrollo y el producto final que, por falta de conocimiento o mala planificación, se ven afectados los miembros que conforman el equipo de trabajo, así como la confianza que exista con los interesados y clientes finales hacia la organización.

Una vez identificados los factores esenciales que permitan garantizar el éxito de la entrega de los productos de calidad, tomando en cuenta las restricciones y modelo del negocio, así como las políticas internas existentes, es posible estipular las claves de éxito que deben de ser prioritarias y establecidas en las organizaciones y equipos de trabajo, beneficiándoles competitivamente.

OBJETIVOS

General

Dar a conocer las ventajas de la buena planificación de desarrollo de software, considerando la metodología por utilizar para el desarrollo, mediante un análisis comparativo de las metodologías y buenas prácticas utilizadas en la actualidad por empresas de desarrollo de software establecidas en Guatemala, resaltando las ventajas y desventajas de cada una de estas.

Específicos

1. Conocer las metodologías de desarrollo más utilizadas en empresas guatemaltecas.

2. Describir las ventajas prácticas de utilizar patrones de diseño en el desarrollo de *software* según expertos en la materia.

3. Identificar la problemática más común según la metodología utilizada tomando en cuenta las condiciones donde se desarrolla.

4. Conocer los antipatrones de diseño más utilizados en el desarrollo de software y las razones detrás de su uso inicial.

5. Conocer los beneficios del uso de las buenas prácticas en la planificación de proyectos de software.

6. Evaluar el costo del cambio de las metodologías utilizadas una vez iniciado el proyecto.

7. Establecer qué metodologías se adaptan según el proyecto de software a desarrollar.

8. Realizar un análisis comparativo entre las diferentes metodologías utilizadas para conocer las ventajas y desventajas en el uso del desarrollo de software.

JUSTIFICACIÓN

Al desarrollar proyectos de software, es necesario definir la metodología por utilizar, por lo que se debe seleccionar una metodología que cumpla con el tipo de desarrollo que se realizará, tomando en cuenta distintos factores, partiendo del recurso humano, las capacidades tecnológicas de la institución, la complejidad del proyecto y el tiempo esperado para la realización del desarrollo.

Existe un conjunto de herramientas y metodologías que pueden ser utilizadas a lo largo del desarrollo del software. Puede que lo componentes seleccionados inicialmente para el desarrollo del proyecto de software no sean los idóneos por lo que se debe considerar un cambio respecto a las técnicas, componentes y metodologías utilizadas una vez iniciado el desarrollo y el realizar este cambio puede ser muy complejo.

Considerando la existencia de diferentes herramientas y metodologías para el desarrollo de software, se pueden obtener diversas métricas, tomando en cuenta la retroalimentación de expertos, aprender de cada uno de los proyectos ejecutados y definir la posibilidad de cambios según la fase en que se encuentre y el impacto que tendrá.

INTRODUCCIÓN

Para la gestión de desarrollo de proyectos de software es necesario el uso de metodologías de desarrollo del mismo, debido a que son útiles, tomando en cuenta los diferentes factores; la selección de estas dependerá de las necesidades y restricciones con las que se cuentan.

Los administradores de proyectos deben saber con toda claridad qué se quieren alcanzar respecto a su desarrollo. Por ello, cuentan con diferentes herramientas a su disposición para la implementación del software, como las metodologías diferentes y plataformas de software, así como la calidad final del producto que se desea obtener.

Dado que existe variedad lineamientos, se puede hacer uso de las buenas prácticas para el desarrollo del software, por lo que es necesario conocer los patrones, así como los anti patrones de diseño para definir las acciones que se ejecutarán durante el desarrollo del software.

Este trabajo pretende dar a conocer las decisiones que se deben tomar respecto al desarrollo de software, la selección adecuada y las ventajas de una buena planificación y selección de las metodologías, según las necesidades y la experiencia previendo evitar cometer problemas comunes.

1. MARCO TEÓRICO

1.1. Procesos de software

También se le conoce como ciclo de vida de desarrollo de software, es la estructura que se utilizará en el producto de software que se desarrollará.

Es un conjunto de elementos que se utilizan para desarrollar el software, como personas, metodologías, herramientas, reglas, procedimientos y políticas de la organización.

1.1.1. Especificación de modelos

Los procesos de software están compuestos por los siguientes elementos:

- Actividad: acciones que se realizan al desarrollar el software.
- Flujo de trabajo: conjunto de actividades y elementos relacionados que generan un resultado.
- Rol: responsable de realizar las actividades del proceso.
- Producto o artefacto: entradas y salidas de las actividades.
- Disciplina: conjunto integrado de actividades enfocadas a una rama de conocimiento específica.

1.1.2. Modelos de desarrollo de software

Existen diferentes modelos para el desarrollo de software, dependiendo de las necesidades y los recursos con los que se cuenten, son los factores a considerar para la selección del mismo.

1.1.2.1. Modelos prescriptivos de proceso

Se refiere al modelo base que se utilizará para el desarrollo de software, el cual está compuesto por comunicación, planeación, modelado, construcción y desarrollo.

Los modelos prescriptivos se refieren al conjunto de elementos del proceso. Son actividades esenciales, proceso de ingeniería de software, tareas, entregables, aseguramiento de calidad y medidas de control de cambios en los proyectos.

Los procesos no deben de ser estáticos, por lo que se busca que puedan acoplarse según las necesidades y los recursos con que se cuentan.

1.1.2.2. Modelo en cascada

También conocido como ciclo de vida clásico. Sugiere el desarrollo de las actividades de forma secuencial, compuesto por las siguientes fases:

- Especificación de requerimientos
- Planeación
- Modelado
- Construcción
- Despliegue

Actualmente, el desarrollo de software no se acopla a este tipo de modelo, dado que se busca realizarlo en menor tiempo y el surgimiento de cambios que se presentan a lo largo de este.

1.1.2.3. Modelos de proceso incrementales

Este modelo se acopla para proyectos de software, en los cuales están bien definidos los requisitos iniciales, donde se descarta el proceso lineal.

1.1.2.4. Modelos de proceso evolutivos

Modelos que busca entregar una versión del desarrollo una vez ha sido completado, hasta llegar al objetivo final.

1.1.2.5. Procesos de desarrollo ágil

Los métodos de desarrollo ágil se enfocan en el desarrollo iterativo e incremental, donde los requisitos y soluciones van cambiando con el tiempo, según se necesite.

1.1.2.5.1. Extreme Programming

El uso de esta metodología ha sido exitoso en compañías de diferentes tamaños y modelos de negocio en el ámbito mundial, dado que hace énfasis en la satisfacción del cliente.

Permite que los desarrolladores respondan a las necesidades cambiantes del cliente. Se enfoca en el trabajo en equipo donde se incluyen gerentes, clientes y desarrolladores, como compañeros colaboradores de un equipo.

Mejora el desarrollo de proyectos basándose en cinco valores; la comunicación, simplicidad, retroalimentación, respeto y coraje. La constante comunicación con el cliente y la retroalimentación permite la entrega lo más pronto posible con los cambios de requerimientos y tecnología sugeridos.

1.1.2.5.2. SCRUM

Es un *framework* para la gestión y control de procesos que reduce la complejidad y se enfoca en la construcción de un producto que cumple con las necesidades del negocio. Es efectivo para la colaboración en equipo de productos complejos. Se basa en los siguientes valores; coraje, enfoque, compromiso, respeto y sinceridad

Existe un equipo conformado por un dueño del producto, equipo de desarrollo y *scrum master*. Los equipos son auto-organizados escogen la mejor manera de realizar su trabajo y tienen las habilidades de cumplir con el mismo sin depender de otros.

1.1.3. Modelos de mejoras de procesos

Existen diferentes modelos y normas que pueden ser aplicados para mejorar la calidad de los proyectos de software, buscando aumentar la productividad de estos.

1.1.3.1. Modelo de madurez de capacidades (CMMI)

Es un modelo de mejora del rendimiento de clase mundial para empresas competitivas que desean alcanzar un alto rendimiento en sus operaciones.

Consiste en la colección de las mejores prácticas diseñadas para promover los comportamientos que guían a la mejora del rendimiento de cualquier organización.

Este modelo maneja cinco niveles, los cuales proveen un método de calificación riguroso capaz de comparar las capacidades de la organización con competidores, industria y esta misma a través del tiempo. Estos niveles de madurez trazan un camino visible para mejorar:

- Inicial. El desarrollo se basa en la responsabilidad de los individuos. No existen procedimientos ni plantillas definidas.

- Gestionado. Normalización de buenas prácticas en el desarrollo de proyectos.

- Definido. La organización está involucrada en un proceso eficiente de proyectos de software.
  - Planillas y métodos bien definidos y documentados.
  - Procesos a nivel organizacional.
  - Definición de proyectos cualitativamente.

- Cuantitativamente gestionado
  - Indicadores estadísticos de la evolución de los proyectos.
  - Almacenamiento de estadísticas para referencias a futuros proyectos.

- Optimizado
  - Posible determinar desviaciones comunes y optimización de procesos.

- Anticipación de problemas y revisión continua, permitiendo la reducción de costos.

1.1.3.2. ISO 9000

Es un conjunto de estándares internacionales para gestión y aseguramiento de la calidad, desarrollado para ayudar a las empresas de forma efectiva en la implementación de sistemas de forma eficiente. Estos no se enfocan en un área específica y se puede aplicar a organizaciones de cualquier tamaño.

Permite ayudar a las empresas para la satisfacción de sus clientes, cumpliendo con los requisitos reglamentarios y alcanzar la mejora continua. Este debe ser tomado como paso inicial, es punto de partida para establecer los niveles de calidad de un sistema.

Se base en siete principios de gestión de calidad que pueden ser aplicados para la mejora de la organización:

- Enfoque en clientes
- Liderazgo
- Compromiso de las personas
- Enfoque basado en procesos
- Mejora
- Toma de decisiones basada en evidencias
- Gestión de relaciones

1.1.3.3. ISO 15504

Norma internacional también denominada SPICE, *Software Process Improvement and Capability Determination*. Su objetivo es evaluar y mejorar la capacidad y madurez de los procesos.

Utilizada en conjunto con la norma ISO 12207 para evaluar y mejorar la capacidad y madurez de los procesos de desarrollo y mantenimiento de software, está definida por un conjunto de buenas prácticas para guiar a la organización.

Está organizada en niveles de capacidad, evaluada con una puntuación en una escala de 0 a 5.

Tabla I. **Niveles de capacidad ISO 15504**

Puntuación	Estado	Descripción
5	En optimización	Cambio de los procesos Mejora continua
4	Predecible	Medición de los procesos Control de los procesos
3	Establecido	Definición de los procesos Recursos de los procesos
2	Gestionado	Gestión de los procesos
1	Realizado	Ejecución del proceso
0	Incompleto	

Fuente: elaboración propia.

1.2. Ingeniería de software

El diseño y construcción de software parte de la toma de requisitos para la comprensión de las necesidades que se subsanarán o los problemas que se resolverán. Se debe definir el ámbito y naturaleza del problema por resolver, continuando con la obtención de las necesidades del cliente. Una vez obtenida la información necesaria se elabora, se modifica, si es necesario, y se afinan los requisitos básicos. Una vez validados y revisados, se asegura que coincide con la percepción del cliente.

1.2.1. Tareas de la ingeniería de requisitos

Es un mecanismo adecuado para la comprensión de lo que desea el cliente, análisis de necesidades, evaluación de factibilidad, negociación de la solución, definición específica de la solución, validación de especificaciones y administración de los requisitos.

- Inicio. Establecer y comprender el problema planteado por el cliente.
- Obtención. Identificar con objetivos para el sistema, satisfaciendo las necesidades del negocio.
- Elaboración. Desarrollo de un modelo técnico donde se afinan las funciones, características y restricciones del software.
- Negociación. Identificación y análisis de riesgos de los requisitos, estimaciones preliminares de esfuerzo y evaluación del impacto en costo y tiempo del proyecto. Establecimiento de requisitos que ya no se tomarán en cuenta, combinación o modificación de estos.
- Especificación. Elaboración de documento escrito, modelado del sistema, casos de uso, prototipo o combinación de éstos.

- Validación. Se examina la especificación, asegurando que no existan inconsistencias, omisiones o errores y la corrección de estos, así como el cumplimiento de lo establecido para el proceso y el producto deseado.
- Gestión. Conjunto de actividades para la identificación, control y rastreo de requisitos y cambios en estos mientras el desarrollo del proyecto.

1.2.2. Procesos de la ingeniería de requisitos

Se refiere al trabajo en conjunto entre los clientes y los ingenieros de software.

- Identificación de los interesados
- Reconocimiento de diferentes puntos de vista
- Trabajo de colaboración
- Formulación de preguntas

Es el proceso de comunicación con las personas interesadas donde se acuerdan reuniones para la recopilación de información, por medio de preguntas y respuesta, recopilación de documentos entre otros.

Se definen los requisitos, agrupándolos según el tipo al que correspondan y los casos de uso identificados con ayuda de los usuarios.

1.2.3. Construcción de modelos de análisis

Existen diferentes tipos de modelado que puede ser utilizado en la fase de análisis, los cuales corresponden a las representaciones de los requerimientos de información, funcionamiento y comportamiento del sistema.

1.2.4. Negociación de requisitos

Se requiere un balance de la funcionalidad, rendimiento y otras características del sistema. Esto se negocia entre el cliente y el desarrollador, tomando en cuenta los costos y el tiempo. Con esto, se realiza un plan de proyecto donde se cumplan con las restricciones del equipo de software y las necesidades del cliente.

1.2.5. Validación de requisitos

Se establecen jerarquías y se agrupan los requisitos, de los cuales se debe tomar en cuenta, que satisfagan las necesidades del cliente y brinden una base sólida para el diseño.

1.3. Modelado de análisis

Es una interpretación técnica inicial de un sistema. Es una combinación de formatos de texto y diagramas para representar los requisitos de datos, funciones y comportamiento para su fácil compresión.

1.3.1. Análisis de requisitos

Se indican diferentes elementos del sistema, como la interfaz y restricciones del software. Los objetivos principales son describir lo que el cliente requiere, establecer una base y definir los requisitos que puedan ser validados, los cuales ayudarán a evaluar la calidad del software, una vez construido.

1.3.2. Concepto de modelado de datos

Es la definición de todos los objetos de datos que serán procesados dentro del sistema y la relación entre ellos. Siendo la representación de la información comprensible por el software, propiedades de los datos y las relaciones entre estos.

1.3.3. Tipos de modelado

Existen diferentes tipos de modelado que se pueden utilizar según el alcance y los recursos disponibles. Cada uno de los tipos se acopla según las capacidades y experiencia que se cuente.

1.3.3.1. Análisis orientado de objetos

El objetivo es definir todas las clases importantes para plantear una solución al problema, así como sus atributos y métodos, relaciones y jerarquía.

1.3.3.2. Modelado basado en escenarios

Se refiere al análisis con UML para la creación de distintos escenarios, representados por medio de casos de uso. Muestra la interacción entre los productores y consumidores de información, diagramas de actividad. El flujo de interacción en un escenario específico y diagramas de carril es una variación de los diagramas de actividad indicando al responsable de la acción.

1.3.3.3. Modelado orientado al flujo

Permite la visión del flujo de los objetos hacia el interior del software, los cuales son procesados y los objetos resultantes fluyen al exterior.

1.3.3.4. Modelado basado en clases

Utiliza las clases orientadas a objetos, las cuales encapsulan atributos y funciones que debe realizar. Las clases pueden ser clasificadas por:

- Entidades externas
- Cosas
- Sucesos o eventos
- Roles
- Unidades organizacionales
- Sitios
- Estructuras

1.4. Ingeniería de diseño

Se refiere a la representación o modelo del software en el que se definen la estructura de datos, arquitectura e interfaces, es decir, los elementos fundamentales necesarios para el desarrollo del software.

1.4.1. Proceso y calidad del diseño

El propósito es producir, de forma eficaz y eficiente, un producto de software, que cumpla con los requerimientos del cliente.

1.4.2. Concepto de diseño

Son el conjunto de componentes necesarios para comprensión del diseño, siendo los siguientes:

- Abstracción
- Refinamiento
- Modularidad
- Arquitectura
- Jerarquía de control
- División estructural
- Estructura de datos
- Procedimientos de software

1.4.3. Modelado de diseño

Está compuesto por cuatro elementos importantes: los datos, arquitectura, componentes e interfaz. Conforme se desarrollan estos elementos, se tiene una visión más completa del diseño.

1.5. Patrones de diseño

Los patrones de diseño son el conjunto de experiencias para la solución de problemas recurrentes y conocidos en el ámbito de la programación.

1.5.1. Clasificación de patrones de diseño

Los patrones de diseño están divididos en tres grupos:

- Patrones de creación: ayuda para que el proceso sea independiente en la creación, composición y representación de sus objetos.
- Patrones estructurales: este tipo de patrones realiza una combinación de clases y objetos para crear estructuras más grandes.

- Patrones funcionales: estos se enfocan en organizar los controles dentro del sistema.

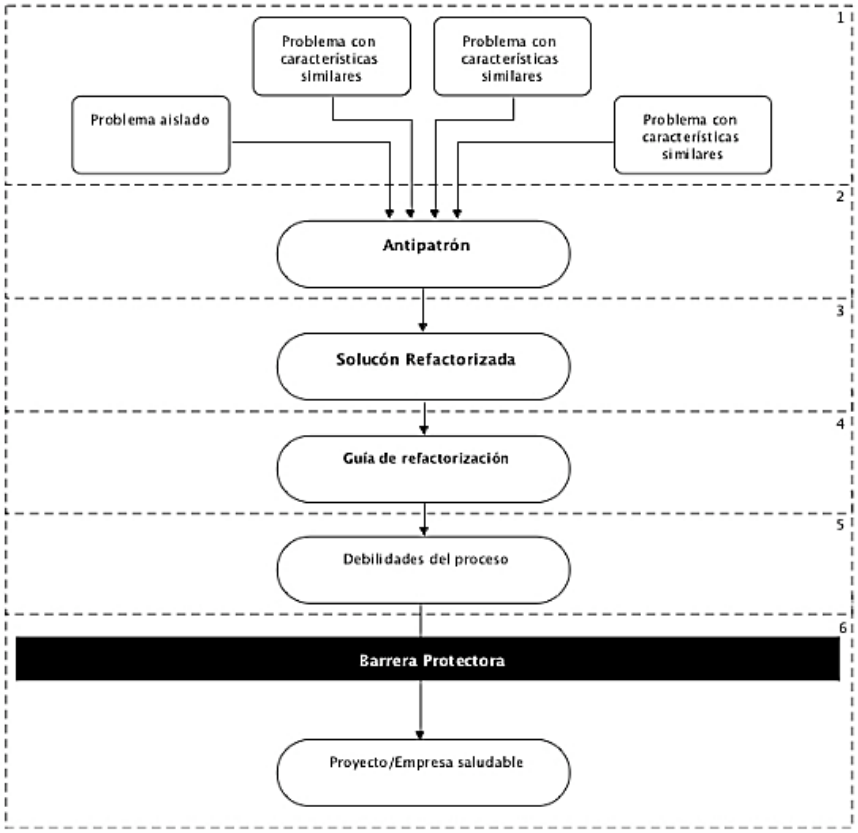
1.6. Antipatrones de diseño

Se refiere a la descripción de situaciones que producen consecuencias negativas en el ámbito del desarrollo de soluciones de software.

Se propone un proceso de para el estudio de los antipatrones:

- Encontrar la falla en el comportamiento
- Identificar las condiciones en que se presenta el problema
- Refactorizar el código
- Publicar la solución
- Identificar debilidades o posibles problemas en el proceso
- Corregir el proceso

Figura 1. **Proceso de estudio de antipatrones**



Fuente: Patrones de Diseño, Refactorización y Anti-patrones. Ventajas y Desventajas de su Utilización en el Software Orientado a Objetos Disponible en Web: http://www.ucasal.edu.ar/htm/ingenieria/cuadernos/archivos/4-p101-Campo.pdf [Consulta: 08 de junio de 2017]

1.7. Administración de proyectos

Parte esencial del éxito de un proyecto de software es la administración correcta del mismo. Es de vital importancia la administración de los proyectos, compuesto por las siguientes tareas: planificación, organización, asignación de personal, dirección, monitoreo, control, innovación y representación.

1.7.1. Características de los proyectos de software

Diferente a otro tipo de proyectos, los proyectos de software deben cumplir con un conjunto de características listadas a continuación:

- Invisibilidad: a diferencia de los proyectos de software, estos no son visibles, dado la naturaleza de los mismos.
- Complejidad: este tipo de proyectos tiende a ser complejos, dada la abstracción y funcionalidad a cumplir.
- Flexibilidad: dado los cambios que pueden existir a lo largo del tiempo, la aplicación de estos es factible en este tipo de proyectos, modificando la funcionalidad con la que se cuenta en cierto momento del tiempo.

1.7.2. Problemas comunes

El desarrollo de proyectos de software produce un conjunto de problemas comunes, dado lo abstracto que puede ser, y a que se trabaja con materia intangible. El desarrollo, como tal, genera problemática desde la forma de utilización de los recursos, tanto de forma técnica y entre los participantes que lo conforman.

A continuación, se listan algunos de los problemas comunes que se pueden presentar:

- Baja calidad del software
- Cambio de requerimientos
- Falta de compromiso
- Mala de comunicación con usuarios y equipo de trabajo
- Retraso de actividades
- Trabajo duplicado

La ocurrencia de estos problemas pueden ser ocasionados por las siguientes situaciones:

- Definición de trabajo inadecuada
- Falta de conocimiento
- Falta de control de calidad
- Falta de estándares
- Falta o documentación inadecuada
- Mala selección de administradores

1.7.3. Paradigmas organizacionales

A nivel organizacional, conforme transcurre el tiempo, se van forjando prácticas sobre cómo manejar las situaciones y la lógica del negocio. De los cuales se pueden listar los siguientes:

- Cerrado. De tipo jerárquico, funcionan bien en proyectos similares a lo que se han trabajado previamente.
- Aleatorio. Se adecuan a proyectos innovadores y adelantos tecnológicos.
- Abierto. Combinación entre el paradigma cerrado y aleatorio, funcionan bien con problemas complejos, pero no eficientemente.
- Sincrónico. Se apoya en la división de un problema y se conforman equipos de trabajo con poca comunicación entre ellos.

1.8. Calidad de software

Se refiere al conjunto de características que cumplen con los requisitos del proyecto de software, ya sean funcionales o no funcionales.

1.8.1. Características de calidad en componentes

Una característica de calidad es un conjunto de propiedades que evalúa y describen la calidad de un producto, la cual se puede dividir en sub características. Por ejemplo, la funcionalidad, fiabilidad o facilidad de uso, entre otros.

Se definen atributos a los cuales se les puede asignar una métrica, la cual evalúa el componente.

Para cumplir con un modelo de calidad, es el conjunto de características y sub características del producto a evaluar.

1.8.2. Propuestas de calidad

Existen estándares previamente definidos para el desarrollo de software, los cuales se basan en un conjunto de características sobre la calidad del producto.

También se puede enfocar en los procesos y en la selección de los componentes para construcción del software, dependiendo del enfoque deseado.

1.8.3. Aseguramiento de la calidad

Se refiere a validar los procesos usados para crear los productos. Es una herramienta de utilidad para administradores y patrocinadores. Esta permite discutir los procesos usados para determinar si los productos creados son razonables.

Es necesario utilizar la metodología o procedimientos estándares para el análisis, diseño, programación y prueba del software que permitan uniformar la filosofía de trabajo, para lograr una mayor confiabilidad, mantenibilidad y facilidad de prueba, a su vez elevar la productividad, tanto para la labor de desarrollo como para el control de la calidad del software. Cuando no se cumplen los estándares o procesos establecidos de la organización o del proyecto se dice se enfrenta una no conformidad. Se espera la ausencia de no conformidades.

2. METODOLOGÍA

2.1. Metodología de investigación

La investigación es de tipo comparativo, debido a que el tema de estudio se centra en el análisis relacionado con el uso de metodologías y buenas prácticas en el desarrollo de software, que abarquen temas de patrones y antipatrones de diseño.

Se busca describir el uso de buenas prácticas y metodologías utilizados durante la construcción del software, considerando el tema de la calidad del software. Estos elementos se deben tomar en cuenta en la toma de decisiones al planificar el software.

2.2. Enfoque de la investigación

La investigación se enmarca en el enfoque cualitativo, dado que se obtendrán las características principales de los temas que se abordarán, así como el análisis comparativo entre las metodologías por investigar. Además, se darán a conocer las ventajas y desventajas, según la naturaleza de la solución planteada.

2.3. Sujeto de estudio

El sujeto de estudio o población lo constituyen metodologías de desarrollo y buenas prácticas de software empleadas por los administradores de proyectos de software.

2.4. Muestra

La investigación se enfoca en la identificación de metodologías de desarrollo de software que se utilizan actualmente, así como el uso de patrones de diseño y las características de calidad para garantizar el éxito en los productos de software.

2.5. Técnicas de recolección de información

La obtención de información bibliográfica utilizada para el desarrollo de la investigación, obtenida de diferentes fuentes, como libros, informes y artículos relacionados con los temas incluidos en esta investigación.

Se acudió a fuentes electrónicas, como revistas, informes en línea, monografías y páginas relacionadas.

También se realizaron entrevistas con expertos en el ámbito de la planificación y gestión de desarrollo de proyectos de software para la obtención de información relacionada con las metodologías, prácticas, patrones y toma de decisiones para el desarrollo de software.

La encuesta realizada tiene dos enfoques definidos, uno es el basado en la experiencia del profesional en su formación y ámbito laboral y el enfoque organizacional, basado en las políticas internas y decisiones propias de esta, la cual se encuentra detallada en el área de apéndices de este documento.

Se busca obtener la información de ambos enfoques, porque para cada uno de estos se tienen diferentes fundamentos para su implementación, que serán de

utilidad para la comprensión de la toma de decisiones en los proyectos de software.

El uso de las fuentes de información se realiza mediante el análisis crítico de toda la información obtenida por los expertos, para ampliar conocimientos y argumentar el contenido de la investigación con base en estudios.

3. ANÁLISIS DE RESULTADOS

A partir de la obtención de información de un grupo de expertos: gerentes, líderes y directores en la gestión y administración de proyectos de software, y con el apoyo del marco teórico aquí presentado, se obtiene la información a continuación presentada.

3.1. Uso y selección de metodologías de desarrollo

El conocimiento y usabilidad de las metodologías de desarrollo de software es variable, según la experiencia y estándares definidos por las organizaciones. Al seleccionar la metodología para desarrollar el software, se debe considerar el tipo de proyecto que se implementará, el tamaño del mismo y, en algunos casos, políticas internas de la organización.

Existen escenarios donde se utiliza más de una metodología de desarrollo de software, es decir se utiliza una combinación de las mismas. Según la experiencia, de ellas se selecciona lo mejor y de mayor utilidad para el proyecto. Es decir que se toman ciertos componentes de utilidad dada la necesidad y giro del negocio.

Al seleccionar e implementar una metodología, puede presentar diferentes problemas, estos son:

- Resistencia al cambio
- Falta de conocimiento
- Mala planificación

- Falta de experiencia

Una vez puesto en marcha el proyecto y mientras más avanzado se encuentre, es muy costoso el cambio de la metodología por utilizar, considerando los riesgos que conlleva en costo, tiempo y recursos. Por ello, se debería definir la planificación en las etapas iniciales, porque esta es la base para llevar a cabo el proyecto.

Una de las ventajas de identificar este tipo de situaciones, es que se obtiene experiencia útil para la definición de futuros proyectos y las restricciones que se presenten.

3.2. Administración de proyectos

Para el desarrollo de software es de vital importancia la administración de proyectos, ya que se definen todos los componentes necesarios y el punto de partida para llevarlo a cabo.

Un gerente de proyecto que controle y coordine las tareas en las fases del proyecto, desde la incepción del mismo hasta su finalización.

Una administración y seguimiento adecuados durante el desarrollo del proyecto garantiza su entrega y éxito al cumplir con los requisitos planteados, el uso adecuado de los recursos y la fecha de entrega del producto.

Uno de los factores más importantes en la ejecución de proyectos de software es en su administración y seguimiento, como la constante y buena comunicación de los involucrados.

3.3. Obtención de requisitos

Es una fase fundamental para los proyectos de software. Aquí se deben obtener los requisitos funcionales y no funcionales, definiendo las restricciones y alcance de lo que se desea ejecutar en el proyecto.

Si esta actividad no se realiza de forma adecuada, su desarrollo tendrá un costo muy elevado, si se identifica la falta de alguno de los requisitos, no se cumpla o comprenda con lo solicitado.

Se debe hacer la validación correspondiente con todos los interesados para confirmar que los requisitos obtenidos concuerden con las necesidades que se deben subsanar y, partiendo de esta información, se dan por aceptados los mismos y continuar con las fases correspondientes para dar inicio al proyecto.

3.4. Modelado y diseño

Una vez se ha concluido con la obtención de requisitos, se debe modelar y diseñar la solución, lo cual facilita y permite a las personas encargadas del desarrollo del software una visión más clara de los requisitos y funcionalidad a cumplir.

Dependiendo de la modalidad de desarrollo seleccionada, así será la documentación que se utilice en esta fase, con los diagramas útiles para el cumplimiento de los requisitos, así como la selección adecuada de infraestructura, conociendo las limitantes y restricciones de recursos.

Dado que existen distintas metodologías, la cantidad de diagramas y detalle de los mismos depende de su selección.

3.5. Patrones y antipatrones de diseño

En la actualidad, se cuenta con un conjunto de patrones de diseño que son un punto de partida para su buen uso durante el desarrollo de proyectos de software, así como los antipatrones. Su identificación permite evitarlos.

La existencia de estos surge gracias a problemas comunes en el ámbito de la codificación de los proyectos de software, los cuales son de utilidad para minimizar riesgos y errores.

Con la ayuda de estos patrones, se pueden definir directrices relacionadas con qué ejecutar y bajo qué condiciones realizarlo. También se define lo que no se debe de realizar, permitiendo un desarrollo ordenado y evitando potenciales problemas.

3.6. Calidad del software

Existe un conjunto de características que se deben cumplir para obtener un producto de calidad. Con la definición adecuada de los requisitos, buena administración del proyecto, apoyo de diferentes librerías, *frameworks* y estándares, se puede realizar un software de calidad, que beneficie la organización en lo relacionado con sus estándares y madurez para el desarrollo de proyectos de software.

El uso de normas y estándares posibilita el logro de la calidad óptima en los proyectos de software para las organizaciones, tomando en cuenta estas recomendaciones y su aplicación es útil para proyectos a futuros y para mejorar los procesos internos.

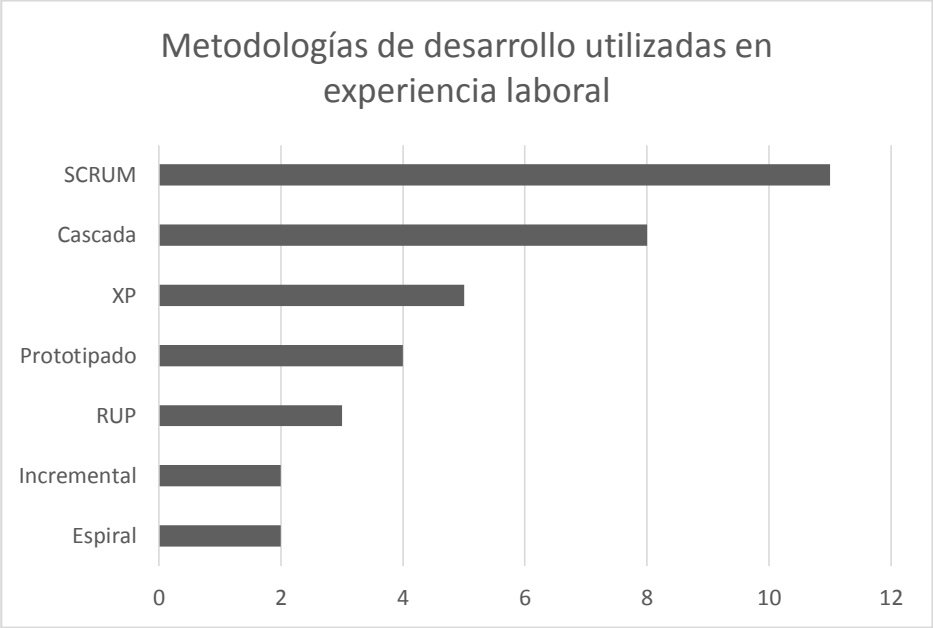
Es importante conocer los beneficios que el uso de estas normas y estándares pueden proporcionar a las organizaciones, ya que les permite formar parte de la cultura organizacional y facilitar el manejo y la entrega de productos de calidad.

4. RESULTADOS

4.1. Enfoque profesional

Pregunta Núm. 1: ¿Qué metodologías de desarrollo de software ha utilizado en su experiencia laboral?

Figura 2. **Resultados pregunta Núm. 1**



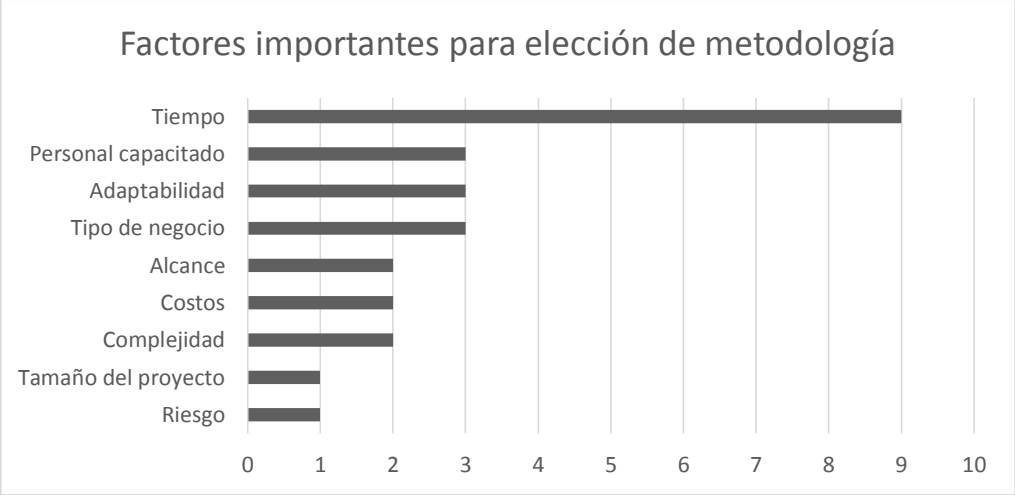
Fuente: elaboración propia.

Las respuestas de los profesionales indican que en el ámbito laboral existe un predominio del uso de metodologías como SCRUM y cascada, porque el proyecto se puede manejar desde su concepción hasta su entrega, de forma más

simple. No obstante, el uso de metodologías complejas es menos frecuente, considerando el trabajo y esfuerzo que conlleva su aplicación.

Pregunta Núm. 2: ¿Cuáles considera los factores más importantes al momento de elegir una metodología de desarrollo?

Figura 3. **Resultados pregunta Núm. 2**

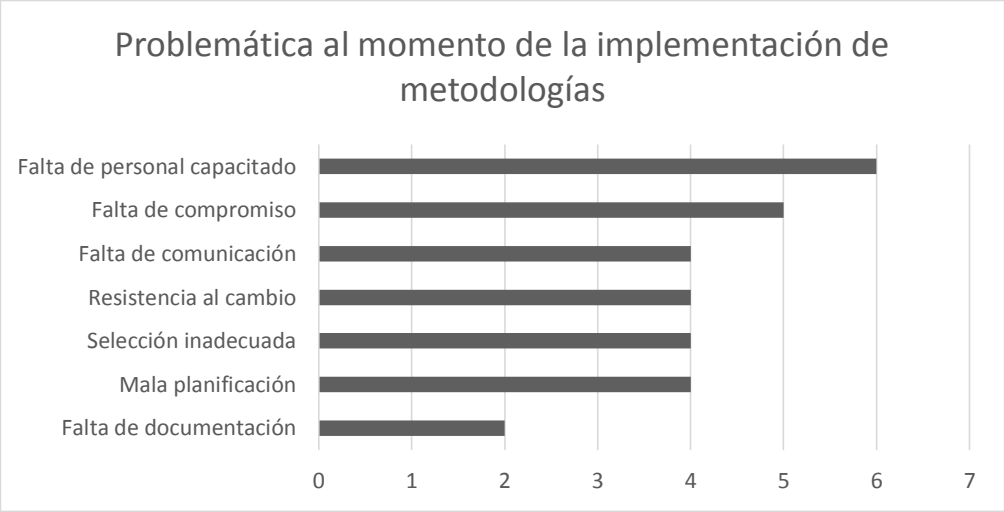


Fuente: elaboración propia.

El factor tiempo es de vital importancia al seleccionar una metodología de desarrollo de software, tomando en cuenta sus características y eficiencia para cumplir con las tareas.

Pregunta Núm. 3: ¿Ha identificado alguna problemática en común al momento de implementar las metodologías de desarrollo de software?

Figura 4. **Resultados pregunta Núm. 3**

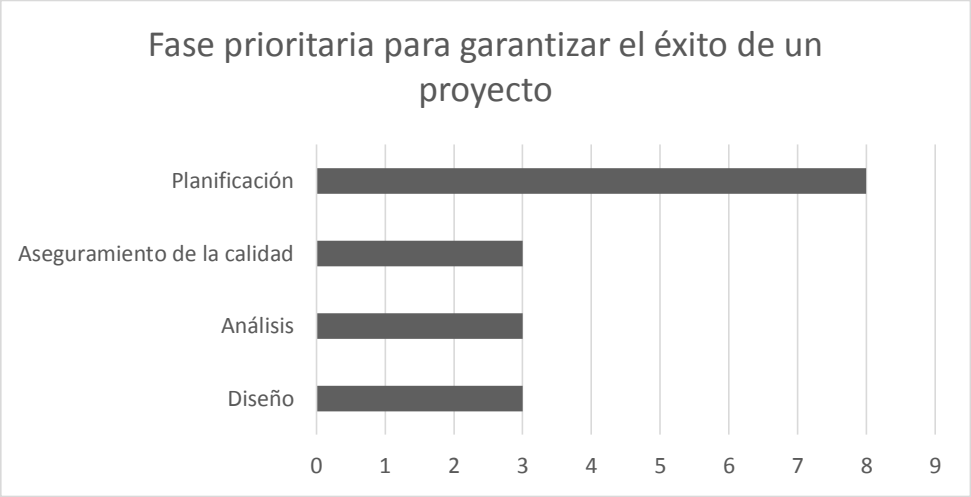


Fuente: elaboración propia.

A lo largo de la implementación de metodologías de desarrollo se presentan distintos problemas. Los más recurrentes son la falta de personal capacitado y el compromiso de estos. Esto desencadena situaciones desfavorables durante el desarrollo del proyecto, por lo que se debe considerar la posibilidad de brindar las capacitaciones correspondientes si se carece de los conocimientos necesarios.

Pregunta Núm. 4: ¿Qué fase del desarrollo de software considera prioritaria para garantizar el éxito de un proyecto?

Figura 5. **Resultados pregunta Núm. 4**



Fuente: elaboración propia.

La fase de planificación es importante para garantizar el éxito del proyecto. En esta se definen las actividades, recursos y tareas por realizar, es el punto de partida para la definición de metas y objetivos del producto por entregar.

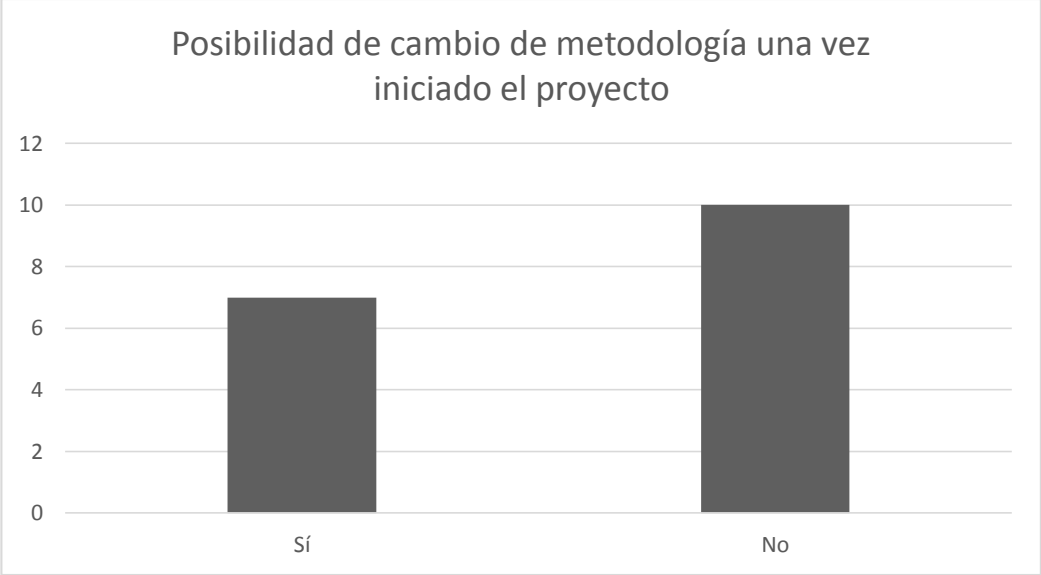
Pregunta Núm. 5: ¿Cómo analiza el impacto y la factibilidad de realizar un cambio dentro del desarrollo de un proyecto de software?

Existen diferentes para la toma de decisiones de la realización de un cambio dentro del proyecto de software, una vez ya se ha puesto en marcha. Es importante conocer la necesidad del cambio solicitado y el impacto que generaría en el proceso con la implementación de este, debido a las dependencias existentes y el esfuerzo que conlleva.

Deben definirse prioridades con las tareas que se están ejecutando, el tiempo que implicaría su realización, los recursos, el esfuerzo y el tiempo para su ejecución.

Pregunta Núm. 6: ¿Una vez iniciado el proyecto, existe la posibilidad de cambiar la metodología empleada una vez se ha iniciado un proyecto?

Figura 6. **Resultados pregunta Núm. 6**

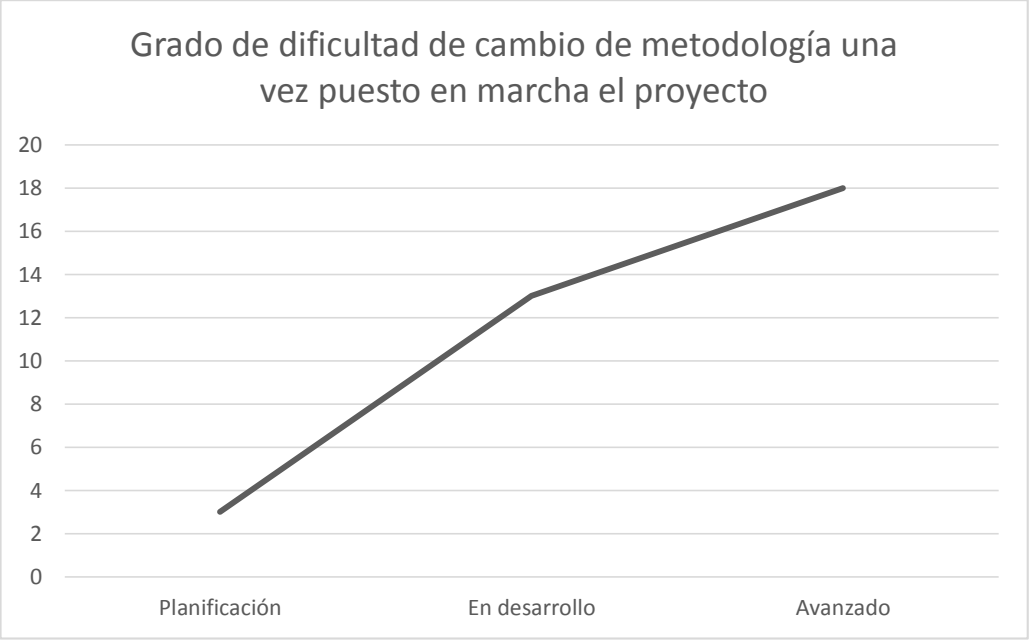


Fuente: elaboración propia.

Cuando se considera posible un cambio de metodología, se deben tomar en cuenta los riesgos que conlleva. Los casos que no se contemplan, se consideran una mala práctica y es un mal uso y desperdicio de recursos.

Pregunta Núm. 7: Dependiendo en la etapa en que se encuentre, ¿cuál es el grado de dificultad de un cambio de metodología una vez iniciado el proyecto de software, y cómo lo haría?

Figura 7. **Resultados pregunta Núm. 7**

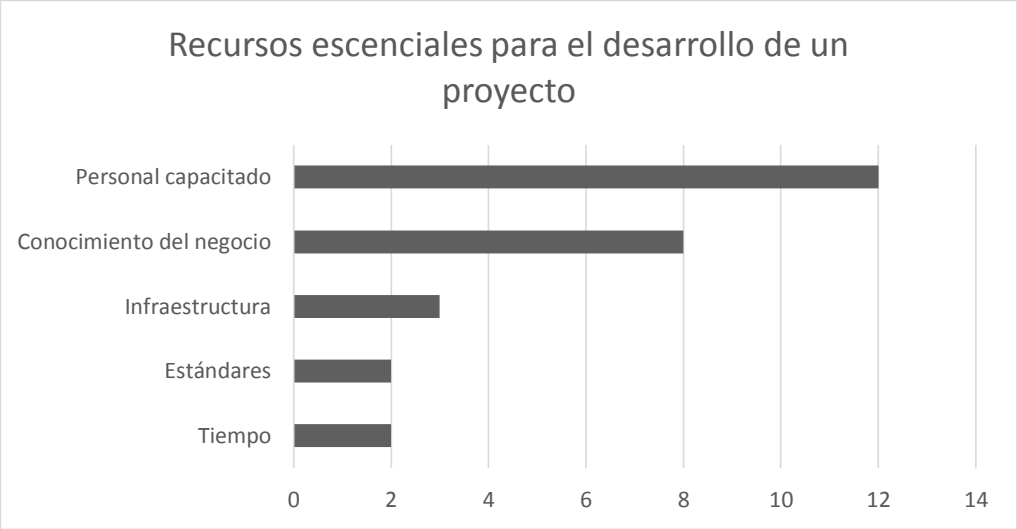


Fuente: elaboración propia.

Es posible un cambio de metodología en etapas iniciales del proyecto, una vez que se ha iniciado, el impacto que se generaría es muy alto, o bien, es posible una transición más simple si la nueva metodología que se desea aplicar comparte valores y se tenga un buen conocimiento de la misma.

Pregunta Núm. 8: ¿Qué recursos considera esenciales para el desarrollo de un proyecto de software?

Figura 8. **Resultados pregunta Núm. 8**

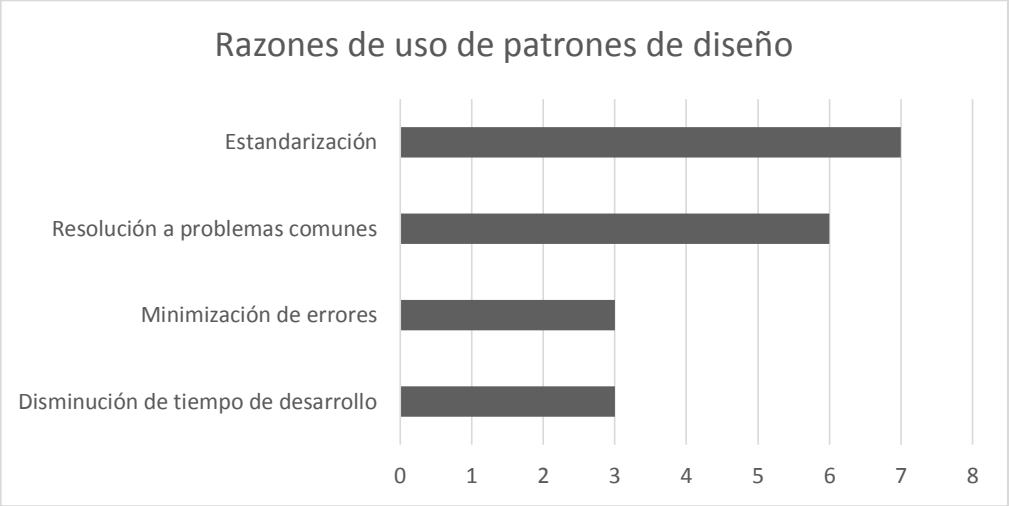


Fuente: elaboración propia.

Para el desarrollo de un proyecto de software se considera esencial contar con personal capacitado, ya que debe ejecutar las tareas y cumplir con las metas establecidas.

Pregunta Núm. 9: ¿Cuáles han sido sus razones para el uso de patrones de diseño del desarrollo de software?

Figura 9. **Resultados pregunta Núm. 9**

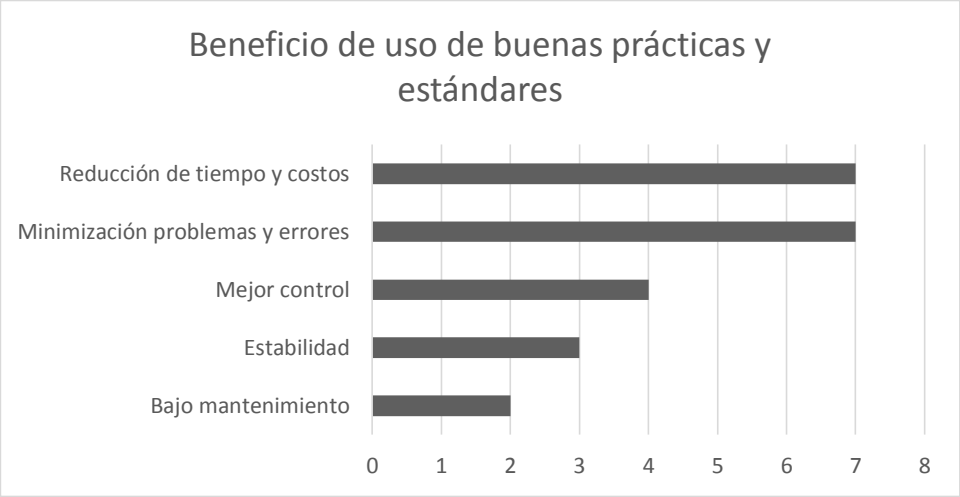


Fuente: elaboración propia.

Una de las razones más frecuentes de utilización de patrones de diseño es la estandarización y resolución de problemas comunes, ya que brindan beneficios al codificar y facilitan la comunicación con los miembros del equipo, para establecer normas y directrices para la ejecución del proyecto.

Pregunta Núm. 10: ¿Qué beneficios le ha brindado la implementación de algún estándar o práctica para asegurar la calidad del software?

Figura 10. **Resultados pregunta Núm. 10**

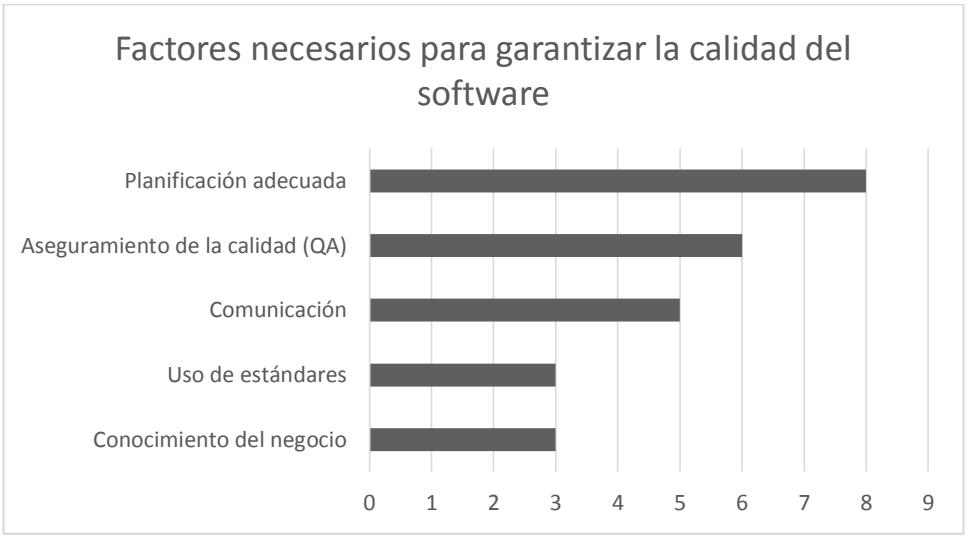


Fuente: elaboración propia.

Al momento de implementar estándares y buenas prácticas se obtienen diferentes beneficios que permiten asegurar la calidad del software, estos ayudan a establecer buenas prácticas y soluciones. Los beneficios de mayor recurrencia son la reducción de tiempo y costo, la minimización de problemas y errores.

Pregunta Núm. 11: ¿Qué factores considera necesarios para garantizar la calidad del software?

Figura 11. **Resultados pregunta Núm. 11**

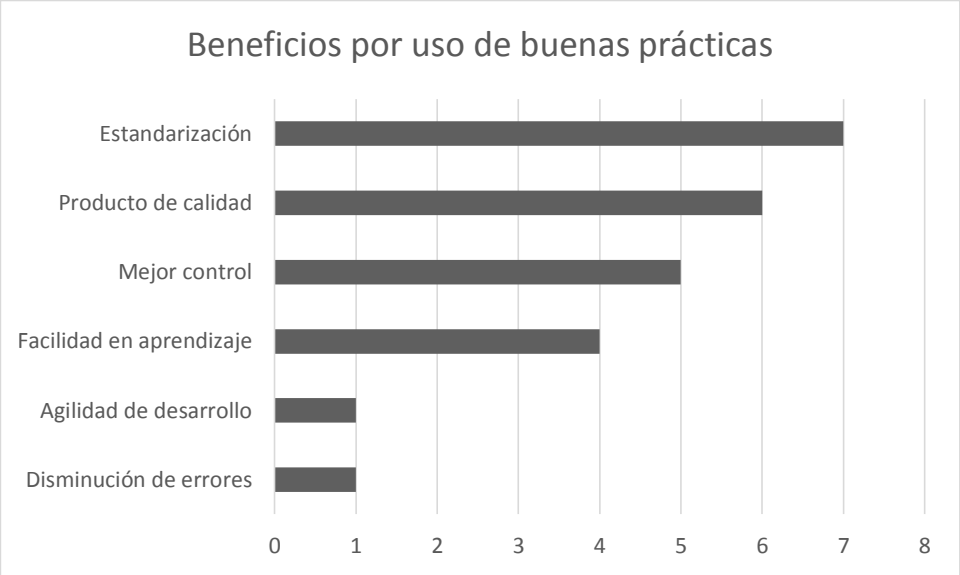


Fuente: elaboración propia.

Uno de los factores de mayor importancia a considerar para garantizar la calidad del software es la realización de una planificación adecuada, de no definir los objetivos adecuadamente desde la incepción del mismo puede generar caos a lo largo de su desarrollo.

Pregunta Núm. 12: ¿Qué beneficios le ha brindado el uso de buenas prácticas en el desarrollo de software?

Figura 12. **Resultados pregunta Núm. 12**



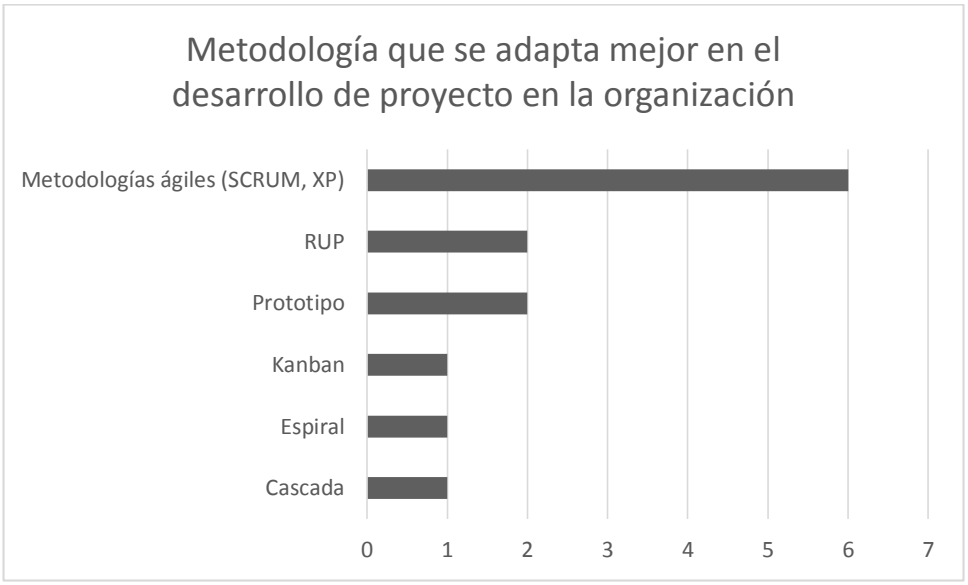
Fuente: elaboración propia.

El uso de buenas prácticas, dada su naturaleza, brinda beneficios. Entre los más recurrentes está la estandarización y la entrega de un producto de calidad. Al definir un estándar, se facilita el uso e implementación, mejorando el proceso y obteniendo un producto estable y de calidad.

4.2. Enfoque organizacional

Pregunta Núm. 13: Basado en su experiencia, ¿qué metodologías de desarrollo considera se adaptan mejor a los proyectos de desarrollo en la organización?

Figura 13. **Resultados pregunta Núm. 13**

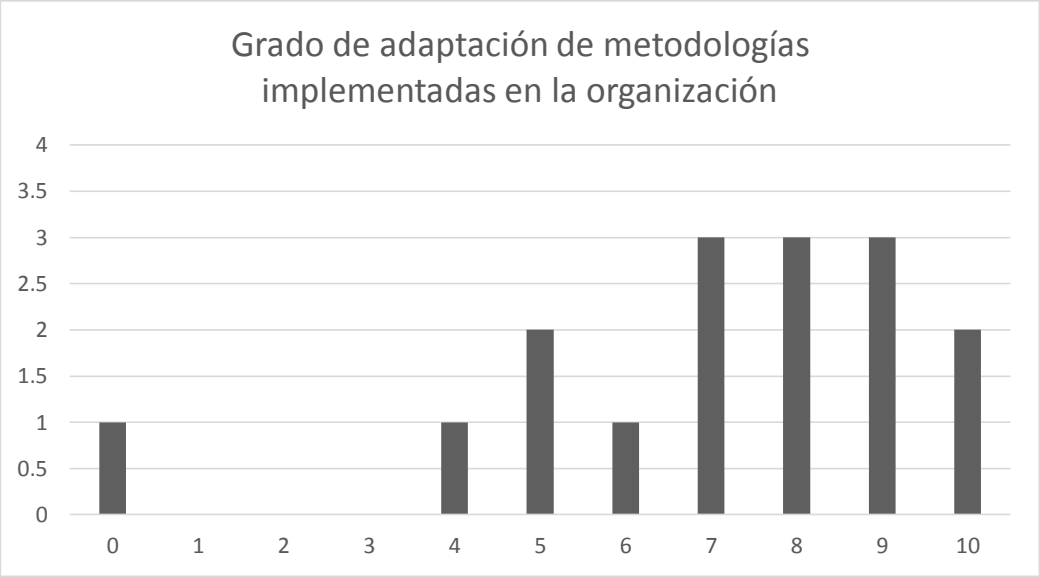


Fuente: elaboración propia.

Existen diferentes metodologías de desarrollo. En la actualidad, ha habido un crecimiento en la utilización de metodologías ágiles, dados los beneficios que estas brindan. Una de las mayores y más frecuentes necesidades es la entrega de productos en un tiempo reducido y los constantes cambios que se a lo largo de la implementación de un proyecto. Estas metodologías son las que mejor se acoplan.

Pregunta Núm. 14: ¿Cuál considera que es el grado de adaptación de las metodologías en el desarrollo de software que ha implementado en la organización, donde 10 se adapta completamente al negocio y 0 no se adapta adecuadamente?

Figura 14. **Resultados pregunta Núm. 14**



Fuente: elaboración propia.

El grado de adaptación de la metodología utilizada en las organizaciones varía según el negocio. La selección de esta se basa en decisiones propias de la organización, por lo que se mantiene un grado medio-alto.

Para aumentar u optimizar el grado de adaptabilidad, se requiere una mejora en la implementación de los procesos y compromiso por parte de los miembros del equipo, contar con el conocimiento y la capacitación adecuada.

Pregunta Núm. 15: ¿Podría nombrar como mínimo 2 y máximo 5 metodologías de desarrollo de software que utilizan en la organización, ordenado de la más utilizada a la menos utilizada?

Figura 15. **Resultados pregunta Núm. 15**

Fuente: elaboración propia.

Existe un dominio en el uso de metodologías ágiles en las organizaciones, dado que se acoplan mejor a las necesidades tan cambiantes y los requerimientos que van variando a lo largo del tiempo.

Pregunta Núm. 16: ¿Qué criterios utilizan para definir los puestos y la asignación tareas de cada uno de los colaboradores en las diferentes tareas para el desarrollo de proyectos de software en la organización?

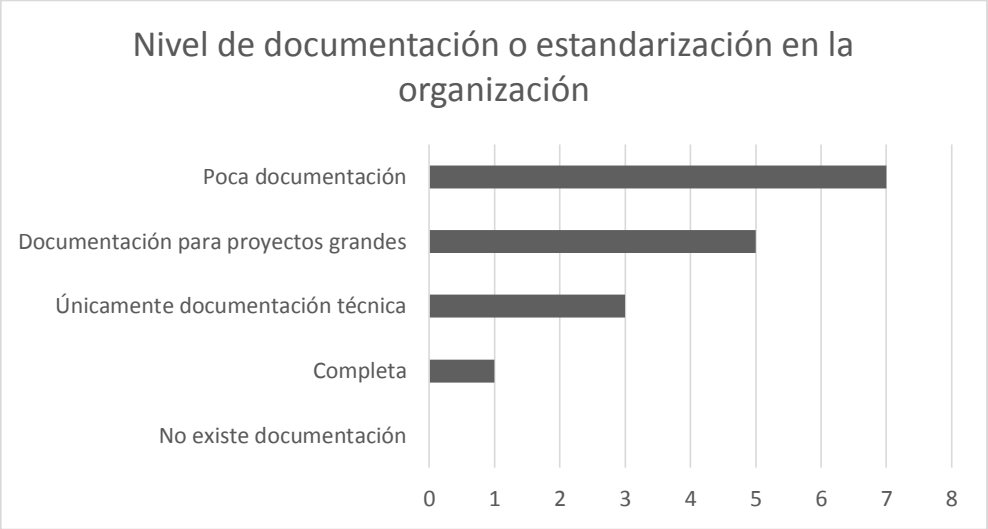
Figura 16. **Resultados pregunta Núm. 16**

Fuente: elaboración propia.

Un criterio importante para la definición de los puestos y asignación de tareas es que el personal tenga la experiencia necesaria para su realización, este es un factor primordial, que facilita su desenvolvimiento, puede tomar el control y compartir el conocimiento con el equipo de trabajo.

Pregunta Núm. 17: ¿Cuál es el nivel de documentación o estandarización que utiliza en la organización durante el desarrollo de software?

Figura 17. **Resultados pregunta Núm. 17**



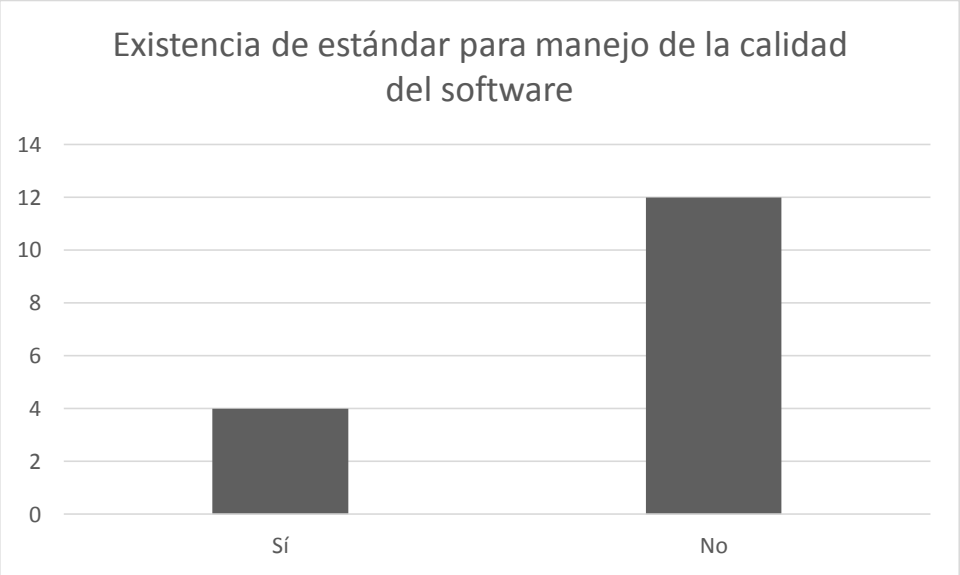
Fuente: elaboración propia.

Se observa que, en términos generales, existe poca documentación en los proyectos de software, o bien solo existe para proyectos voluminosos, los cuales pueden ser considerados de mayor importancia.

Es recomendable mantener un estándar con el uso de la documentación, para contar con un repositorio que se pueda consultar en cualquier momento.

Pregunta Núm. 18: ¿Actualmente la organización cuenta con algún estándar para manejar la calidad del software?

Figura 18. **Resultados pregunta Núm. 18**



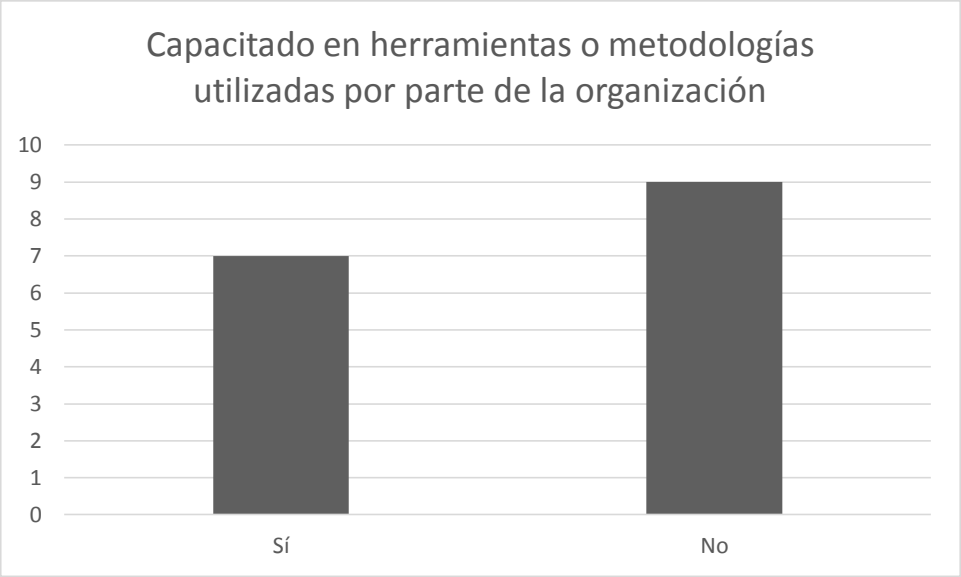
Fuente: elaboración propia.

Muy pocas organizaciones aplican estándares para manejar la calidad del software. La mayoría de los casos se debe a la falta de tiempo para su implementación o no ha sido implementado en su totalidad.

Los miembros de las organizaciones conocen el manejo de estándares de calidad, por lo que en la mayoría de los casos, se está planificando su implementación a corto-mediano plazo.

Pregunta Núm. 19: ¿De parte de la organización ha sido capacitado/certificado en algún *framework*, metodología o herramienta para la planificación y desarrollo de software, y que esté siendo aplicada hoy en día en la organización?

Figura 19. **Resultados pregunta Núm.19**



Fuente: elaboración propia.

Las organizaciones están interesadas en que el personal conozca las herramientas que utilizan internamente. Por ello, han facilitado capacitaciones para el personal. La falta de tiempo ha impedido que se lleven a cabo estas acciones en algunos casos.

Pregunta Núm. 20: ¿Conoce las razones del uso de los patrones de diseño en el desarrollo de software dentro de la organización?

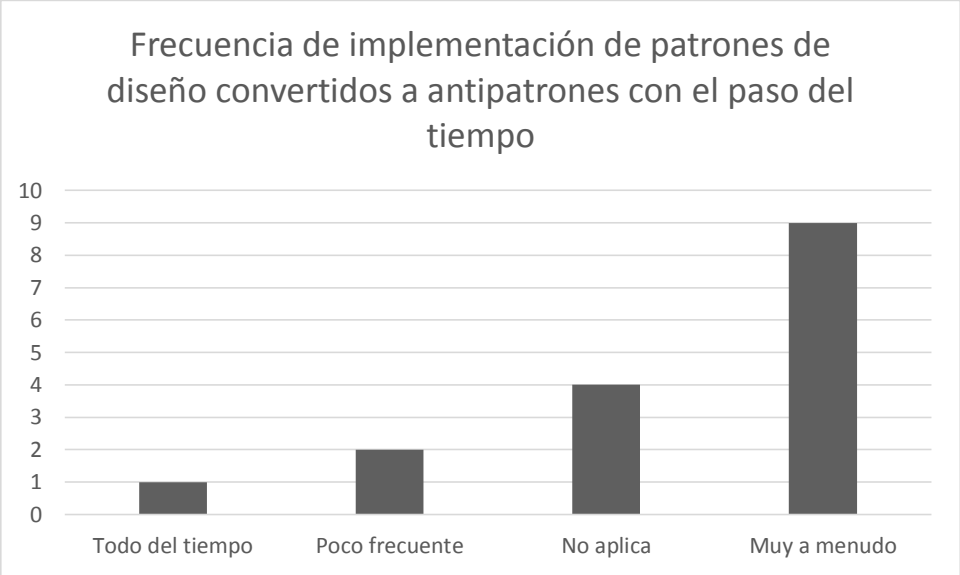
Figura 20. **Resultados pregunta Núm. 20**

Fuente: elaboración propia.

Al utilizar los patrones de diseño, en la mayoría de los casos, se le informa al personal el motivo de su utilización, considerando los beneficios que estos brindan.

Pregunta Núm. 21: ¿Qué tan a menudo, la implementación de patrones de diseño se ha convertido en antipatrones conforme el paso del tiempo dentro de la organización?

Figura 21. **Resultados pregunta Núm. 21**



Fuente: elaboración propia.

El uso de patrones de diseño se convierte en antipatrones de diseño conforme el paso del tiempo, dado que las razones del planteamiento inicial han cambiado considerablemente o los cambios han sido muy variados por lo que la definición inicial no va acorde a las necesidades actuales.

Pregunta Núm. 22: Al momento de identificar alguna eventualidad durante el desarrollo del proyecto de software, indistintamente de la metodología seleccionada, ¿cuáles han sido las mejores prácticas para la resolución de estas eventualidades que usted ha aplicado?

Se debe realizar el análisis de los riesgos que conlleva realizar un cambio, identificar el impacto, establecer prioridades y mantener una comunicación abierta y continua con los clientes para mantenerlos informados, y realizar las negociaciones correspondientes, en caso fuera necesario un cambio en las fechas de entrega o bien agregar esta nueva funcionalidad en un nuevo entregable.

5. DISCUSIÓN DE RESULTADOS

La administración de proyectos, en el ámbito de desarrollo de software es muy cambiante, dado que la ejecución de los proyectos depende de la funcionalidad que sea solicitada y los recursos disponibles para la ejecución de estos.

Es necesario tomar en consideración los factores para garantizar el éxito de los proyectos. Existen diferentes prácticas y metodologías por implementar, que basadas en experiencias previas son útiles para la aplicación en los proyectos de software, facilitan y mejoran la administración y desarrollo de los mismos.

5.1. Selección de metodología

Se debe tomar en cuenta que la selección de la metodología es esencial para el proyecto, de no tomar la decisión adecuada repercute su desarrollo, provocando altos costos y uso inadecuado de los recursos.

Es importante conocer el negocio y los recursos existentes para la ejecución de los proyectos, requerimientos a cumplir en el desarrollo y las restricciones o políticas definidas por la organización o el cliente. Esta información es clave para la selección de la metodología que mejor se acople.

Según la opinión y conocimiento de expertos, no se usa una sola metodología, en varios casos se combina lo mejor de diferentes

metodologías, para lograr un avance, trabajando con sus beneficios, ventajas y fortalezas.

En la actualidad predomina el uso de metodologías ágiles, dada la necesidad de entregas rápidas y cambios constantes en los proyectos de software, dado que estas son más fáciles de adaptar en la presencia de nuevos requerimientos o cambios, así como los valores que se practican. Se hace énfasis en la comunicación e involucramiento constante con el cliente, lo cual pretende asegurar que lo que se está desarrollando cumpla con los requerimientos.

5.2. Problemática en selección de metodología

La problemática más frecuente, al seleccionar la metodología es la falta de información, la cual no se obtienen totalmente, en la fase de toma de requerimiento y análisis, por falta de experiencia y conocimiento del negocio.

Si los requerimientos no son comprendidos en su totalidad y no se considera la funcionalidad esperada de la solución, de forma adecuada, es necesario afinar esta información y conocer las ventajas y usabilidad que tienen las metodologías. Esto es primordial para seleccionar la metodología idónea.

Eventualmente, una vez iniciado con el proyecto, surgen problemas que ligados con la metodología seleccionada. Si el proyecto se encuentra en fases iniciales, es posible cambiarla, porque el impacto es mínimo. En cambio, si se encontrara en fases intermedia o final, el costo sería excesivo y se debería empezar de nuevo.

Si no se selecciona la metodología idónea, se genera un efecto negativo en cadena: el diseño, la codificación, la calidad y el tiempo de ejecución serán inadecuados.

5.3. Patrones y antipatrones de diseño

Como apoyo a la gestión de proyectos, una vez se ha definido el tipo de metodología, es importante considerar el uso de patrones, los cuales proponen y facilitan el desarrollo de proyectos de software.

Una de las mayores ventajas del uso de los patrones de diseño, dada su naturaleza, es que solucionan problemas o situaciones antes del desarrollo, De esta manera se optimizan los recursos y el tiempo, se evitan potenciales dificultades o disminuyen el tiempo para solucionarlas.

El balance, de acuerdo con los resultados, denota que el uso de patrones de diseño es una práctica adecuada que se debe fomentar para el desarrollo de proyectos. Facilitan la comunicación entre equipos y la llegada de nuevos integrantes al equipo de trabajo.

Si se desconoce el uso de los patrones o se pierde interés en su aplicación, con el tiempo y los cambios constantes, pueden dejar de ser beneficiosos y convertirse en una mala práctica, con lo cual se constituyen en antipatrones de diseño.

Es importante identificar a tiempo los antipatrones para corregirlos y minimizar su impacto durante la ejecución del proyecto.

5.4. Calidad del proyecto

Es importante el uso de documentación y estándares para asegurar la calidad del software. Si no se cuenta con este apoyo, no es posible controlarlo y se carecerá de una base que establezca límites y metas. Existen diferentes modelos y normas que proponen el cumplimiento de requerimientos los cuales permiten mantener un estándar de calidad en la entrega de los productos.

Las organizaciones y los integrantes de sus equipos conocen los beneficios que brindan estos modelos. Sin embargo, generalmente, no implementan los modelos dentro de las organizaciones. En algunos casos, se debe a falta de tiempo y capacitación del personal responsable de estas tareas, o bien, aún se encuentran en el planteamiento para implementarlo.

Dado que son evidentes los beneficios que brindará en el futuro el uso de estándares y modelos de calidad, se recomienda su aplicación para la entrega de los productos en aras de subsanar las necesidades de los clientes.

5.5. Administración de proyectos

Es posible la gestión adecuada de los proyectos de software, tomando como base proyectos ejecutados previamente, experiencia con uso de herramientas, metodologías, patrones y estándares. Al carecer de la información necesaria y utilizar incorrectamente estas herramientas, el proyecto puede ser insatisfactorio, se obtendrá un producto que no se cumple con tiempo establecido ni remedia las necesidades planteadas inicialmente.

Las tareas deben entregarse al personal asignado para su cumplimiento, dadas sus habilidades y experiencia. Debe existir definición de puestos y

asignación de tareas específicas para que las personas se responsabilicen de la ejecución de sus atribuciones durante el desarrollo del proyecto.

En la administración del proyecto pueden identificarse puestos claves, como el gerente del proyecto, analista del negocio y líder de tecnología. Son los encargados de la obtención e interpretación de requisitos, dar seguimiento a las tareas asignadas a los equipos de trabajo y asegurarse de que se cumpla con lo establecido.

Con el uso de las herramientas, estándares y buenas prácticas, se facilita la administración de los proyectos, para establecer parámetros y reglas, manteniendo un orden e impidiendo el trabajo empírico, dado que el conocimiento ha sido distribuido para la ejecución adecuada de las tareas.

CONCLUSIONES

1. Las empresas que en Guatemala tienen desarrollo de software utilizan diversas metodologías de desarrollo, que presentan diferentes resultados dependiendo de la aplicación en cada una de las organizaciones. Basado en la experiencia de quienes han implementado estas metodologías, se conocen las ventajas que han brindado. En la actualidad, existe un predomino en el uso de metodologías ágiles, dado los beneficios y la flexibilidad que proveen, tomando en cuenta los cambios constantes que se dan a lo largo del desarrollo del proyecto.

2. Es importante identificar los problemas y situaciones comunes que se pueden enfrentar durante el desarrollo de un proyecto de software, la utilización de los patrones de diseño puede evitar problemas comunes bajo determinados escenarios y situaciones, permitiendo hacer correcciones, facilitar el aprendizaje de los desarrolladores, reducir el tiempo de entrega y mejorar el uso adecuado de los recursos durante la ejecución del proyecto.

3. La problemática común que se encuentra en la selección de una metodología de desarrollo radica en la falta de conocimiento y experiencia del equipo para implementar la metodología, por lo que es necesario contar con el personal calificado para su implementación. Respecto a la utilización de la metodología, existen inconvenientes propios de la misma que de no ser tomados en cuenta nos pueden llevar a establecer lineamientos incorrectos y tomar una decisión no adecuada al momento de elegir e implementarla, afectado el desarrollo del proyecto.

4. Se establece el uso de patrones de diseño basado en experiencias previas, pero a lo largo del tiempo y cambios que se han dado conforme a las necesidades del negocio, estas definiciones dejan de ser de utilidad convirtiéndose en antipatrones de diseño, por lo que es importante identificarlos para que cumplan con las necesidades del negocio y evitar perjuicios para la organización.

5. En la administración de proyectos de software, es importante el uso de diferentes metodologías y buenas prácticas establecidas, dado que estas ayudan al desarrollo y entrega de los proyectos. Pero es necesario conformar un equipo de personas adecuado que conozca suficientemente el proyecto, así es muy probable que haya un buen desempeño durante la gestión de proyectos, disminuye el riesgo al fracaso, aumenta la posibilidad de éxito y asegura la calidad del producto lo cual permita garantizar el éxito al cumplir con los requisitos establecidos, utilizando el tiempo y recursos de forma adecuada.

6. Es posible realizar un cambio de metodología una vez iniciado el proyecto siempre que se encuentre en sus fases iniciales, porque el costo y recursos por utilizar se ven afectados considerablemente, arriesgando el éxito de la ejecución del mismo. La posibilidad del cambio es inversamente proporcional a la etapa donde se encuentre, no es recomendable hacer un cambio en fases avanzadas, dado los riesgos que conlleva.

7. La definición de la selección de una metodología parte de criterios diferentes, como las políticas de las organizaciones, tamaño del proyecto y recursos disponibles. Estos factores son los más determinantes en las empresas guatemaltecas. Es necesario conocer el negocio y sus restricciones, ya que son factores claves para determinar qué metodología

se acopla al proyecto, debido a los potenciales cambios de requerimientos que se puedan presentar a lo largo de su desarrollo. Si son muy frecuentes, se acoplan las metodologías ágiles, incrementales y evolutivas, de lo contrario se puede hacer uso de metodologías más estrictas o pesadas.

8. Las metodologías y su implementación en los proyectos de software han evolucionado con el paso del tiempo. Algunas toman principios de otras y originan nuevas versiones de estas. En la actualidad, hay una predilección por el uso de metodologías más flexibles porque permiten cambios a lo largo de su desarrollo y se basan en la comunicación constante, distinto a las metodologías más estrictas o pesadas que en la actualidad no son de mucha utilidad en entornos dinámicos o giros de negocio cambiantes porque no permiten cambios tan fácilmente o a un bajo costo.

RECOMENDACIONES

1. Debe haber un análisis detrás de la selección de la metodología para un proyecto, el alcance y aplicación dependerá mucho del resultado del análisis que se tenga. Es necesario conocer la funcionalidad del proyecto, restricciones y recursos con los que se cuentan. Cada una de las metodologías cuentan con ventajas y desventajas, por lo que se debe de tener el conocimiento de lo que conlleva su uso implícitamente, siendo esto primordial para el desarrollo del proyecto.

2. Implementar el uso de patrones de diseño, considerada una buena práctica, utiliza estándares y trata de evitar problemas e inconvenientes conocidos, mejora la utilización de recursos y el tiempo para la ejecución del proyecto de desarrollo de software**.** Se debe de poder identificar aquel que cumpla con las características necesarias conforme a las necesidades que se tenga, permitiendo facilidad al momento de su aplicación y un desarrollo más limpio.

3. Analizar profundamente cada uno de los requerimientos y validarlos para realizar el estimado adecuado y cumplir con lo necesitado en el tiempo establecido. Uno de los problemas más comunes en la administración de proyectos de software, se debe a que no son considerados todos los requisitos en su fase inicial, ni cuantificado el tiempo suficiente en el desarrollo del mismo, permitiendo agregar o modificar estos sin planificación alguna, alargando o sobreutilizando los recursos asignados.

4. Mantener un control de las herramientas y estándares utilizados en la organización y del porqué de su implementación, evaluándolas constantemente para determinar si estás aún son de utilidad o bien es necesario realizar un cambio que se acople a las nuevas necesidades del negocio.

5. Conocer y actualizarse sobre de las herramientas para la administración de proyectos de desarrollo de software y, basado en las necesidades y el enfoque del negocio, así como las políticas establecidas, permite la toma de decisiones adecuadas buscando asegurar la calidad y éxito desde la incepción del mismo. Hacer uso de estándares y plantilla dentro de la organización para tener un punto de partida y conocer las metas que se deben cumplir. Es importante que el personal involucrado esté capacitado y tenga el conocimiento del negocio, ya que es fundamental para el éxito de la organización.

6. Tener conocimiento técnico sumando a la experiencia sobre las metodologías, herramientas y técnicas para emplearlas en los proyectos de software y desarrollarlos de forma correcta y exitosa la entrega de los diferentes productos. Se debe realizar una planificación adecuada con todos los elementos involucrados para usar los recursos de la mejor manera.

BIBLIOGRAFÍA

1. Roger S.Pressman. *Ingeniería del Software*. *Un enfoque práctico*. México D.F.2005. Editorial Mc Graw Hill

2. Administración de proyectos de Software. Disponible en Web: http://www.uv.mx/personal/jfernandez/files/2010/07/1_conceptos2012.pdf [Consulta: 10 de mayo de 2017]

3. Patrones de diseño. Disponible en Web: http://es.ccm.net/contents/224-patrones-de-diseno [Consulta: 05 de junio de 2017]

4. Patrones de Diseño, Refactorización y Anti-patrones. Ventajas y Desventajas de su Utilización en el Software Orientado a Objetos Disponible en Web: http://www.ucasal.edu.ar/htm/ingenieria/cuadernos/archivos/4-p101-Campo.pdf [Consulta: 08 de junio de 2017]

5. Métodos de Desarrollo de Software. Disponible en Web: http://www.codecompiling.net/files/slides/IS_clase_13_metodos_y_procesos.pdf [Consulta: 15 de mayo de 2017]

6. Procesos de Software. Disponible en Web: https://sg.com.mx/revista/1/procesos-software#.WRpcdty1vIU [Consulta: 15 de mayo de 2017]

7. CMMI Maturity Levels. Disponible en Web: http://www.tutorialspoint.com/cmmi/cmmi-maturity-levels.htm [Consulta: 21 de mayo de 2017]

8. ISO 9000. Disponible en Web: http://asq.org/learn-about-quality/iso-9000/overview/overview.html [Consulta: 02 de julio de 2017]

9. Aseguramiento de la calidad del software. Disponible en Web:http://sedici.unlp.edu.ar/bitstream/handle/10915/3956/3_-_Aseguramiento_de_la_calidad_del_software.pdf?sequence=11 [Consulta: 02 de julio de 2017]

10. Aseguramiento de la Calidad. Disponible en Web: https://administracionelectronica.gob.es/pae_Home/dms/pae_Home/documentos/Documentacion/Metodologias-y-guias/Metricav3/METRICA_V3_Aseguramiento_de_la_Calidad.pdf [Consulta: 02 de julio de 2017]

11. What is Capability Maturity Model Integration (CMMI)®? Disponible en Web: http://cmmiinstitute.com/capability-maturity-model-integration [Consulta: 02 de julio de 2017]

12. WHAT IS SCRUM? Disponible en Web: https://www.scrum.org/resources/what-is-scrum [Consulta: 02 de julio de 2017]

13. Extreme Programming: A gentle introduction. Disponible en Web: http://www.extremeprogramming.org/ [Consulta: 02 de julio de 2017]

14. La Norma SPICE ISO/IEC 15504. Disponible en Web: https://eqa.es/presentaciones/presentacion_ISO_15504.pdf [Consulta: 02 de julio de 2017]

15. Implantación de ISO 15504 SPICE. Disponible en Web: http://www.kybeleconsulting.com/servicios/evaluacion-y-mejora-de-procesos-software/implantacion-de-iso-15504/ [Consulta: 02 de julio de 2017]

16. MODELO DE EVALUACIÓN (Y MEJORA)DE PROCESOS SOFTWARE ISO 15504. Disponible en Web: https://prezi.com/xqbshgv_dq7t/modelo-de-evaluaciony-mejorade-procesos-software-iso-15504/ [Consulta: 02 de julio de 2017]

APÉNDICES

Apéndice 1. **Encuesta**

Enfoque profesional

1. ¿Qué metodologías de desarrollo de software ha utilizado en su experiencia laboral?

2. ¿Cuáles considera los factores más importantes al momento de elegir una metodología de desarrollo?

3. ¿Ha identificado alguna problemática en común al momento de implementar las metodologías de desarrollo de software?

4. ¿Qué fase del desarrollo de software considera prioritaria para garantizar el éxito de un proyecto?

5. ¿Cómo analiza el impacto y la factibilidad de realizar un cambio dentro del desarrollo de un proyecto de software?

6. ¿Una vez iniciado el proyecto, existe la posibilidad de cambiar la metodología empleada una vez se ha iniciado un proyecto?

7. Dependiendo en la etapa en que se encuentre, ¿cuál es el grado de dificultad de un cambio de metodología una vez iniciado el proyecto de software, y cómo lo haría?

Continuación del apéndice 1

8. ¿Qué recursos considera esenciales para el desarrollo de un proyecto de software?

9. ¿Cuáles han sido sus razones para el uso de patrones de diseño del desarrollo de software?

10. ¿Qué beneficios le ha brindado la implementación de algún estándar o práctica para asegurar la calidad del software?

11. ¿Qué factores considera necesarios para garantizar la calidad del software?

12. ¿Qué beneficios le ha brindado el uso de buenas prácticas en el desarrollo de software?

Enfoque organizacional

13. Basado en su experiencia, ¿qué metodologías de desarrollo considera se adaptan mejor a los proyectos de desarrollo en la organización?

14. ¿Cuál considera que es el grado de adaptación de las metodologías en el desarrollo de software que ha implementado en la organización, donde 10 se adapta completamente al negocio y 0 no se adapta adecuadamente?

Continuación del apéndice 1

15. ¿Podría nombrar como mínimo 2 y máximo 5 metodologías de desarrollo de software que utilizan en la organización, ordenado de la más utilizada a la menos utilizada?

16. ¿Qué criterios utilizan para definir los puestos y la asignación tareas de cada uno de los colaboradores en las diferentes tareas para el desarrollo de proyectos de software en la organización?

17. ¿Cuál es el nivel de documentación o estandarización que utiliza en la organización durante el desarrollo de software?

- Completa
- Poca documentación
- No existe documentación
- Solo existe documentación para proyectos de cierto tamaño
- Únicamente documentación técnica

18. ¿Actualmente la organización cuenta con algún estándar para manejar la calidad del software?

- Sí, detalle
- No, ¿Cuál considera que es la razón por la cual no se tiene?

19. ¿De parte de la organización ha sido capacitado/certificado en algún *framework*, metodología o herramienta para la planificación y desarrollo de software, y que esté siendo aplicada hoy en día en la organización?

- Sí, detalle
- No, ¿por qué no lo considera importante?

Continuación del apéndice 1

20. ¿Conoce las razones del uso de los patrones de diseño en el desarrollo de software dentro de la organización?

21. ¿Qué tan a menudo la implementación de patrones de diseño se ha convertido en antipatrones conforme el paso del tiempo dentro de la organización?

22. Al momento de identificar alguna eventualidad durante el desarrollo del proyecto de software, indistintamente de la metodología seleccionada, ¿cuáles han sido las mejores prácticas para la resolución de estas eventualidades que usted ha aplicado?