UNIVERSIDAD DON BOSCO

VICERRECTORÍA DE ESTUDIOS DE POSTGRADO

TRABAJO DE GRADUACIÓN

UN FRAMEWORK DE BUENAS PRÁCTICAS PARA LOS REQUISITOS DE SOFTWARE DE APLICACIONES WEB APLICABLES AL CONTEXTO SALVADOREÑO

PARA OPTAR AL GRADO DE

MAESTRO EN ARQUITECTURA DE SOFTWARE

ASESOR:

EVA CAMPOS DE BERGANZA

PRESENTADO POR:

FLORENCE GUADALUPE VILLA-ALTA GARAY

OSCAR ARMANDO CASTILLO RODRÍGUEZ

Antiguo Cuscatlán, La Libertad, El Salvador, Centroamérica.

Enero de 2018

Contenido

Índice de tablas

Índice de figuras

Introducción

El propósito de esta investigación nace de identificar que dentro de la fase de análisis del ciclo de vida de desarrollo de software se encuentran las principales causas por las que los proyectos fallan. Inicialmente el marco conceptual define los términos con los que se formuló la base conceptual que rige la investigación y que dan una antesala importante al proceso de revisión bibliográfica.

La investigación "*Un framework de buenas prácticas para los requisitos de software de aplicaciones Web aplicables al contexto salvadoreño",* tiene como dominio, las actividades que conforman los requisitos de software, entre las que están: obtención, análisis, especificación, validación y gestión de requisitos.

Con el objetivo de construir un framework de buenas prácticas relacionadas con los requisitos de software de aplicaciones Web aplicables al contexto salvadoreño, se estableció utilizar una metodología de investigación cuantitativa-cualitativa para obtener las bondades de ambos métodos. El *enfoque cuantitativo* se utilizó con el objetivo de explorar y entender el problema de estudio, y el *enfoque cualitativo* fue utilizado con el objetivo de profundizar en los hallazgos, utilizando como instrumento el cuestionario guiado.

Esta investigación se dividió en dos partes: la primera etapa se refiere específicamente a la revisión bibliográfica entre las que fueron incluidas fuentes como: *CMMI-DEV, SWEBOK, PMBOK, NASA Systems Engineering Handbook*, entre otras bibliografías relacionadas a metodologías y buenas prácticas de requisitos. De estas fuentes se extrajo buenas prácticas que fueron seleccionadas y analizadas, partiendo de una base de 201 buenas prácticas que fueron sintetizadas a un total de 72, de las cuales se seleccionó las principales buenas prácticas, obteniendo así 27 buenas prácticas que formaron lo que constituye el *framework propuesto* en

esta investigación. Posteriormente, en la segunda etapa se realizó la investigación de campo, donde se buscó establecer si las prácticas del framework propuesto son utilizadas en el contexto salvadoreño en el desarrollo de aplicaciones Web, con la colaboración de una muestra intencional de expertos.

Los resultados de la investigación de campo arrojaron valiosa información, identificando *el grado en que las buenas prácticas son o no utilizadas* y en consecuencia permitió identificar si las prácticas son *conocidas y utilizadas* en el contexto.

Finalmente, el cumplimiento de los objetivos de investigación permitió brindar un aporte valioso de parte de esta investigación ya que se construyó un documento en el cual se expone cada buena práctica desde la perspectiva de las fuentes documentales exploradas. Este documento anexo se ha de identificar como *Framework propuesto*.

1. Marco Conceptual

Este apartado facilita la comprensión del dominio del problema, haciendo explícitos los conceptos, definiciones claves y sus relaciones, para el trabajo de investigación: "*Un framework de buenas prácticas para los requisitos de software de aplicaciones Web aplicables al contexto salvadoreño".*

Esta investigación se centra en las buenas prácticas de requisitos de software, por lo que interesa definir lo que se entenderá por buenas prácticas, software y requisitos de software:

1.1 Buenas Prácticas

"Una actividad o proceso comprobado que ha sido utilizado con éxito por múltiples empresas y que ha demostrado producir resultados confiables" (ISACA, 2017).

1.1.1 Fuentes de buenas prácticas

Las buenas prácticas pueden ser encontradas en frameworks, modelos, metodologías, estándares, entre otras.

1.1.1.1 Framework

Klaus Pohl (2010, pág. 42) define un framework como "el marco que consolida varios resultados de investigación que se han desarrollado sobre la base de declaraciones de problemas prácticos y que han sido validados con éxito y transferidos a la industria".

Basados en la definición anterior, para esta investigación se entenderá por framework de buenas prácticas para los requisitos de software a la estructura base que organiza las buenas prácticas alrededor de las *actividades de requisitos de software*, que puedan encontrarse en fuentes como frameworks de desarrollo de software, de gestión de proyectos, de definición y mejora de procesos; así como modelos, estándares y métodos o metodologías relacionadas a los requisitos de software. Por ejemplo: CMMI-DEV 1.3, PMBOK V5, SWEBOK V3, entre otros.

1.1.1.2 Modelos

Representan sistemas, procesos, servicios de tecnologías de la información, elementos de configuración, entre otros. Son empleados para ayudar a entender o predecir comportamientos futuros (Lucio, Corona, & Debenedet, 2011, pág. 72). Ver ejemplos de modelos de ciclo de vida del desarrollo de software en la sección 1.2.3.

1.1.1.3 Metodologías

Se define metodología como "un sistema de prácticas, técnicas, procedimientos y normas utilizadas por quienes trabajan en una disciplina" (PMI, 2013, pág. 454). Ver ejemplos de metodologías de desarrollo de software en la sección 1.2.2.

1.1.1.4 Estándares

"Documento que provee, para uso común y repetitivo, las reglas, pautas o características que deberían cumplir las actividades (o sus resultados), a fin de obtener un óptimo grado de orden en un contexto dado" (PMI, 2013, pág. 541). Por ejemplo: ISO 9000, ISO 29110, PMBOK, entre otros.

Debido a que las buenas prácticas que el framework propuesto en la investigación identifica son para los requisitos de software a continuación la definición de software y proceso de desarrollo de software:

1.2 Software

Según Pressman (2010, pág. 3) el software es:

> "1) instrucciones (programas de cómputo) que cuando se ejecutan proporcionan las características, función y desempeño buscados; 2) estructuras de datos que permiten que los programas manipulen la información, y 3) información descriptiva tanto en papel como en formas virtuales que describe la operación y uso de los programas".

En la actualidad existen siete grandes categorías de software (Pressman, 2010, pág. 6): software de sistemas, software de aplicación, software de ingeniería y ciencias, software incrustado, software de línea de productos, aplicaciones Web y software de inteligencia artificial.

Ya que la investigación está orientada a las aplicaciones Web, esta se define como: una aplicación en la cual un usuario por medio de un navegador realiza peticiones a un servidor Web accesible a través de Internet (o a través de Intranet) y que recibe una respuesta que se muestra en el propio navegador (Mora, 2002, págs. 47-51).

Los tipos de aplicaciones Web son: aplicaciones interactivas, transaccionales, colaborativas, portales Web, Web social, orientadas a servicios, ubicuas (móviles) y semánticas (Sanchez Zuaín & Duran, 2016, págs. 628-629).

1.2.1 Proceso de desarrollo de software

Dado que es habitual entrar en discusión con los términos: proceso de desarrollo de software y proceso de software, es importante indicar que se entenderán como sinónimos para esta investigación.

Entendiéndose como proceso de desarrollo de software al "conjunto coherente de políticas, estructuras organizativas, tecnologías, procedimientos, actividades y artefactos que se necesitan para concebir, desarrollar, implantar y mantener un producto de software" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 22).

A pesar de la variedad de propuestas de procesos de software, existen cuatro actividades fundamentales que son comunes y que se retomarán para esta investigación (Sommerville, 2011, pág. 28):

1.2.1.1 Especificación del software

Define la funcionalidad y restricciones operacionales que debe cumplir el software. Esta fase comprende a los requisitos de software.

1.2.1.2 Diseño e implementación del software

Se diseña y construye el software de acuerdo con las especificaciones.

1.2.1.3 Validación del software

Se valida el software para asegurar que cumpla con lo que quiere el cliente.

1.2.1.4 Evolución del software

El software debe evolucionar para satisfacer y adaptarse a las necesidades cambiantes del cliente.

1.2.2 Metodologías de desarrollo de software

El proceso de desarrollo de software es implementado por medio de metodologías de desarrollo del software, a través de la definición de pautas a seguir y restricciones a cumplir (Pantaleo & Rinaudo, 2014, pág. 54).

Se define metodologías de desarrollo de software como una forma de trabajo para desarrollar software, donde se especifica las tareas a llevar a cabo, los artefactos a generar y las relaciones entre ambos (Pantaleo & Rinaudo, 2014, pág. 54).

Las metodologías de desarrollo de software se clasifican como metodologías conducidas por los planes y ágiles (Pantaleo & Rinaudo, 2014, pág. 55). Siendo cada enfoque adecuado para los diferentes tipos de software (Sommerville, 2011, pág. 29).

1.2.2.1 Metodologías conducidas por los planes

"Los procesos de software dirigidos por un plan son aquellos donde todas las actividades del proceso se planean por anticipado y el avance se mide contra dicho plan" (Sommerville, 2011, pág. 29).

1.2.2.2 Metodologías ágiles

> *"*Los métodos ágiles son métodos de desarrollo incremental donde los incrementos son mínimos y, por lo general, se crean las nuevas liberaciones del sistema, y cada dos o tres semanas se ponen a disposición de los clientes. Involucran a los clientes en el proceso de desarrollo para conseguir una rápida retroalimentación sobre los requerimientos cambiantes. Minimizan la cantidad de documentación con el uso de comunicaciones informales, en vez de reuniones formales con documentos escritos*"* (Sommerville, 2011, pág. 58).

Las metodologías de desarrollo de software se estructuran sobre modelos de ciclo de vida del desarrollo de software, los cuales se detallan en la siguiente sección.

1.2.3 Modelos de ciclo de vida del desarrollo de software

Los modelos de ciclo de vida de desarrollo definen las fases por las que trascurren los proyectos de desarrollo de software y las dependencias entre ellas (Sánchez, Sicilia, & Rodríguez, 2012, pág. 36).

Existen varios modelos. Los que se describirán a continuación no son los únicos, pero si los que han tenido mayor relevancia histórica (Sánchez, Sicilia, & Rodríguez, 2012, pág. 39).

1.2.3.1 Modelo en cascada

Es el modelo más difundido y caracterizado por llevar a cabo distintas fases de desarrollo en secuencia, comenzando cada una de ellas en el punto que terminó la anterior (Sánchez, Sicilia, & Rodríguez, 2012, pág. 40).

"Éste toma las actividades fundamentales del proceso de especificación, desarrollo, validación y evolución, y luego, los representa como fases separadas del proceso, tal como especificación de requerimientos, diseño de software, implementación, pruebas, etc." (Sommerville, 2011, pág. 29) (Ver figura 1).

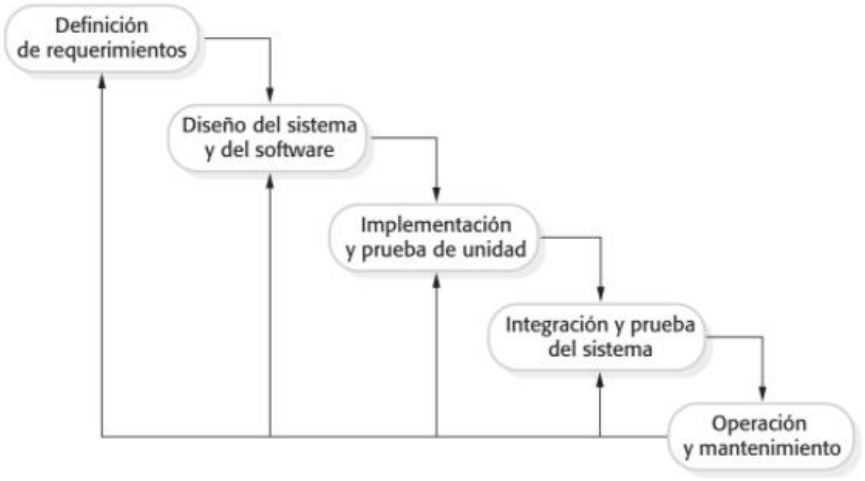

Figura 1. Modelo en cascada. Tomado de Ingeniería del Software (Sommerville, 2011, pág. 30).

1.2.3.2 Modelo en V

El modelo con forma de "V", es una evolución del modelo en cascada haciendo énfasis en actividades de verificación y validación. En su parte izquierda representa las fases de desarrollo de software y en la derecha los tipos de prueba a considerar previo a pasar a la siguiente fase de desarrollo (Sánchez, Sicilia, & Rodríguez, 2012, pág. 41). La figura 2, muestra la formulación típica del modelo.

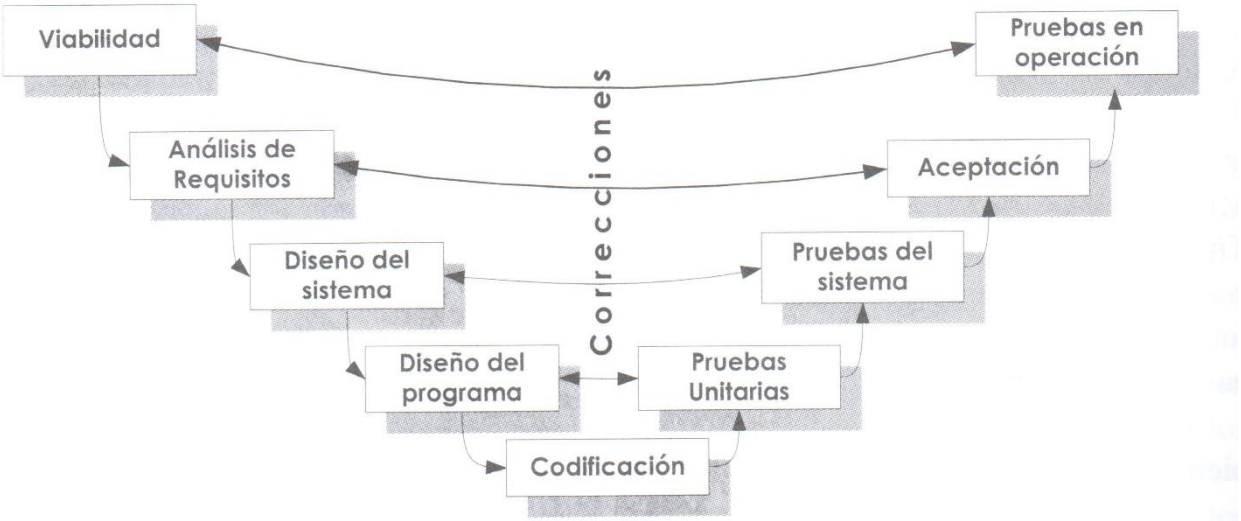

Figura 2. Modelo en "V". Tomado de Ingeniería del Software, un enfoque desde la guía del SWEBOK (Sánchez, Sicilia, & Rodríguez, 2012, pág. 41).

1.2.3.3 Modelo basado en prototipos

Este modelo sugiere crear un prototipo de software que implemente solo una pequeña parte funcional, permitiendo en fases tempranas del proyecto tener una visión y entendimiento común entre clientes, usuarios y desarrolladores. Este modelo puede combinarse con otros, como el modelo en cascada, simplemente incluyendo un prototipo en alguna fase temprana del proceso de desarrollo de software (Sánchez, Sicilia, & Rodríguez, 2012, pág. 42).

La utilización del modelo basado en prototipos mejora el entendimiento común de los requisitos al principio del desarrollo, evitando que los fallos se propaguen hasta el código e identificarlos antes del final del proyecto (Sánchez, Sicilia, & Rodríguez, 2012, pág. 44).

1.2.3.4 Modelo en espiral

Reúne las mejores características del uso del modelo basado en prototipos y de cascada, manteniendo que la gestión de riesgos debe guiar el proceso de desarrollo, es decir evaluar los riesgos de forma regular en el proyecto. La esencia de dicho modelo es la división de cuatro

cuadrantes que representa un tipo de actividad diferente (Ver figura 3) (Sánchez, Sicilia, & Rodríguez, 2012, págs. 44-45).

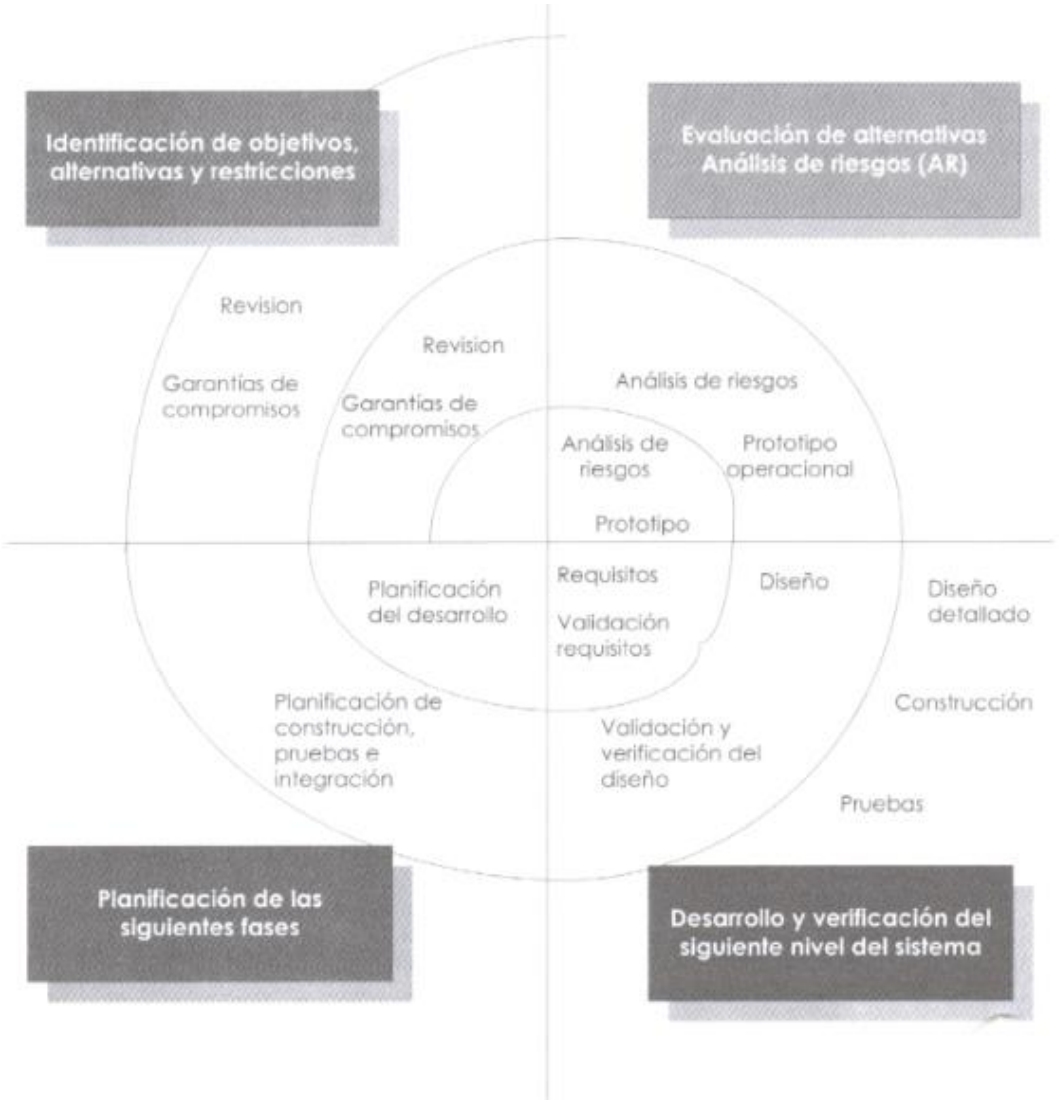

Figura 3. Modelo en espiral. Tomado de Ingeniería del Software, un enfoque desde la guía del SWEBOK (Sánchez, Sicilia, & Rodríguez, 2012, pág. 45).

Todos los modelos y metodologías mencionados brindan pautas para mejorar la calidad del software por lo que se define a continuación:

1.2.4 Calidad del software

Según IEEE (1990, pág. 60) la calidad del software se define como*:* "a) El grado en que un sistema, componente o proceso cumple con los requisitos especificados. b) El grado en que un sistema, componente o proceso cumple con las necesidades o expectativas del cliente o usuario". Para complementar lo anterior Humphrey Watts, conocido como el Padre de la Calidad del Software menciona que el software de calidad es aquel que cumple con las necesidades funcionales del usuario, con la fiabilidad y consistencia de realizar el trabajo del usuario (2009, pág. 2).

Es fácil identificar en la definición anterior que los requisitos de software son un elemento predominante en la calidad del software, y es parte del porqué de esta investigación. Por eso es primordial comprender con claridad ¿qué es un requisito de software?

1.3 Requisitos de Software

SWEBOK proporciona una definición general de requisito de software como "la propiedad que un software desarrollado o adoptado debe tener para resolver un problema concreto" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 116). Somerville y Peter Sawyer, citados por Wiegers (2013, pág. 6) dicen que "los requisitos son una especificación de lo que debe implementarse. Son descripciones de cómo debe comportarse el sistema, o de una propiedad o atributo del sistema. Puede ser una limitación en el proceso de desarrollo del sistema".

Un requisito debe cumplir los siguientes criterios (Kirk Kandt, 2006, págs. 127-128):

- No ambigüedad: el requisito debe permitir una sola interpretación.
- Integridad: se establecen todas las condiciones relativas a la aplicabilidad de un requisito.
- Concisión: el requisito es declarado de forma simple.

- Coherencia: el requisito no entra en conflicto con otro.
- Corrección: el requisito indica con precisión una necesidad del usuario.
- Viabilidad: el requisito puede lograrse dentro de las limitaciones existentes.
- Necesidad: el requisito es esencial para satisfacer las necesidades reales de los usuarios.
- Prioridad: el requisito se clasifica para reflejar su importancia relativa.
- Legibilidad: el requisito debe ser fácil de leer y basado en palabras y frases sencillas.
- Trazabilidad: se identifica la fuente del requisito y se puede remontar a requisitos de nivel superior, casos de prueba funcional y un subsistema u otro artefacto de diseño arquitectónico.
- Unicidad: el requisito no duplica otro requisito.
- Verificabilidad: se puede verificar el requisito (por ejemplo, una prueba podría demostrar la correcta implementación de un requisito).

1.3.1 Tipos de requisitos de software

Dado que existen diferentes clasificaciones de requisitos de software representados por distintos autores, para esta investigación se hará referencia a la definida por Sánchez, Sicilia y Rodríguez (2012, pág. 121):

"Los dos tipos de requisitos que tradicionalmente se han identificado atendiendo a criterios de funcionalidad son: requisitos funcionales y requisitos no funcionales".

1.3.1.1 Requisitos funcionales

Un requisito funcional "especifica una función que un sistema o componente de un sistema debe ser capaz de llevar acabo" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 121). Es decir que "especifican los comportamientos que el producto de software exhibirá bajo condiciones

específicas, describen lo que los desarrolladores deben implementar para permitir que los usuarios realicen sus tareas (requisitos de usuario), satisfaciendo así los requisitos del negocio" (Wiegers & Beatty, 2013, pág. 9).

La mayoría de requisitos funcionales provienen directamente de los requisitos de usuario, y son descritos en un lenguaje natural a través de técnicas de *obtención de requisitos* como la entrevista (Sánchez, Sicilia, & Rodríguez, 2012, pág. 122).

Los requisitos de usuario son aquellos que "describen metas o tareas que los usuarios deben poder realizar con el producto de software que proporcionará valor a alguien. El dominio de los requisitos de usuario también incluye descripción de los atributos o características del producto que son importantes para satisfacción del usuario" (Wiegers & Beatty, 2013, pág. 9). Las formas de representar estos requisitos incluyen casos de uso, historias de usuario y tablas de respuesta a eventos" (Wiegers & Beatty, 2013, pág. 9).

Por otra parte, los *requisitos de negocio* son los que describen por qué la organización está implementando el sistema, así como los beneficios empresariales que la organización espera lograr (objetivos del negocio). Estos requisitos suelen venir del patrocinador o cliente, gerente de los usuarios o un visionario del producto. Generalmente estos se registran en un documento de visión y alcance (Wiegers & Beatty, 2013, pág. 8).

1.3.1.2 Requisitos no funcionales

"Son aquellos que especifican aspectos técnicos que debe incluir el sistema, y que pueden clasificarse en restricciones y calidades" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 121).

Dentro de las restricciones se encuentra cualquier limitación a la que se enfrenten los desarrolladores del sistema, y dentro de las calidades se encuentran todas aquellas características

que importan al usuario final o cliente, que son relevantes para establecer el grado de satisfacción con el sistema final (Sánchez, Sicilia, & Rodríguez, 2012, págs. 121-122).

Una clasificación amplia de los requisitos no funcionales permite identificar tres categorías (Sánchez, Sicilia, & Rodríguez, 2012, pág. 123):

- Requisitos del producto: "aquellos que detallan limitaciones o comportamientos exigidos al producto resultantes del desarrollo (Sánchez, Sicilia, & Rodríguez, 2012, pág. 123)". Tales como el rendimiento, fiabilidad, seguridad y usabilidad (Sommerville, 2011, pág. 88).
- Requisitos de la organización: "aquellos relacionados con las normativas de funcionamiento de la organización que lleva acabo el desarrollo, sus procedimientos y políticas. Por ejemplo, los estándares de desarrollo, la documentación a entregar junto con el producto, los plazos de entrega, etc." (Sánchez, Sicilia, & Rodríguez, 2012, pág. 123).
- Requisitos externos: "cubren aspectos externos al sistema y a su proceso de desarrollo. Por ejemplo, interoperabilidad con otros sistemas, requisitos legales, etc." (Sánchez, Sicilia, & Rodríguez, 2012, pág. 123).

1.3.2 Actividades de requisitos de software

Según la guía SWEBOK el área de los requisitos de software, se ocupa de la obtención (elicitación), análisis, especificación, y validación de los requisitos de software así como la gestión de los requisitos durante todo el proceso de desarrollo de software, con la denominación *genérica de actividades de requisitos* (Sánchez, Sicilia, & Rodríguez, 2012, págs. 116-117).

A continuación, se muestra una infografía de las actividades de requisitos de software:

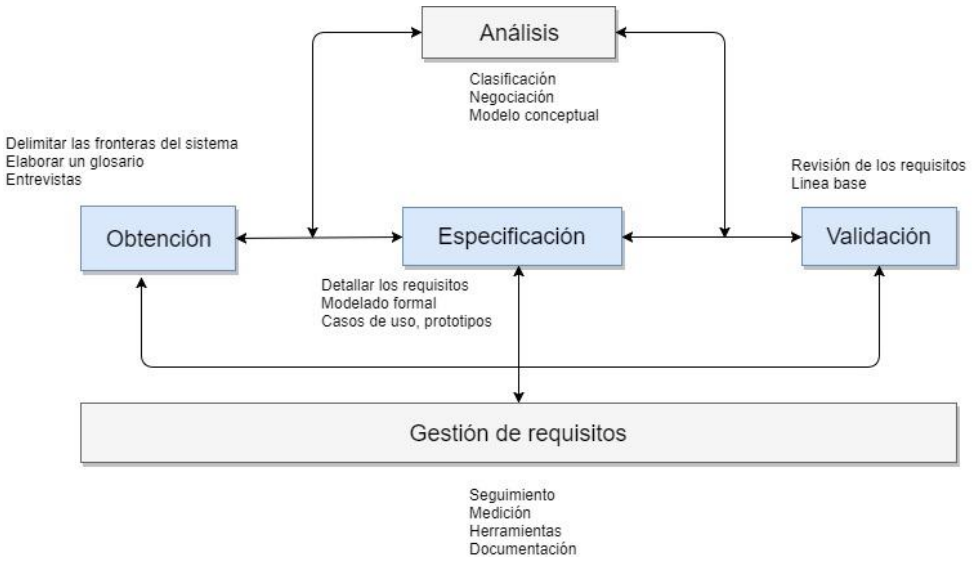

Figura 4. Actividades de requisitos de software. Adaptado de Ingeniería del Software, un enfoque desde la guía del SWEBOK (Sánchez, Sicilia, & Rodríguez, 2012, pág. 127).

1.3.2.1 Obtención de requisitos

Es el proceso en el cual se recolecta la información necesaria para comenzar a entender el problema que el software debe resolver, e identificar qué es lo que el cliente necesita para poder empezar a definir el rumbo del proyecto a realizar (IEEE, 2014, págs. 5-7).

"Consiste en capturar el propósito y funcionalidades del sistema desde la perspectiva del usuario" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 117).

La obtención de requisitos abarca actividades relacionadas con el descubrimiento de requisitos tales como: entrevistas, talleres, análisis de documentos, creación de prototipos, entre otros (Wiegers & Beatty, 2013, pág. 16).

Las acciones claves de la obtención de requisitos son (Wiegers & Beatty, 2013, pág. 16):

- Identificar las partes interesadas.
- Comprender las tareas y objetivos de usuario, junto a los objetivos de negocio con los que se alinean estas tareas.

- Aprender sobre el entorno en el que se utilizará el producto.
- Trabajar con los individuos que representan cada clase de usuarios para entender las necesidades de funcionalidad y expectativas de calidad.

1.3.2.2 Análisis de requisitos

"Es el proceso de estudiar las necesidades del usuario para obtener una definición detallada de los requisitos" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 117). Es decir que implica llegar a una comprensión más rica y precisa de cada requisito, representando conjuntos de requisitos de múltiples maneras (Wiegers & Beatty, 2013, pág. 16).

El proceso de analizar los requisitos se utiliza para (IEEE, 2014, pág. 7):

- Detectar y resolver conflictos entre requisitos.
- Descubrir los límites del software y cómo debe interactuar con su entorno organizacional y operativo.
- Elaborar requisitos del sistema para derivar los requisitos de software.

Las acciones claves del análisis de requisitos son (Wiegers & Beatty, 2013, pág. 16):

- Analizar la información recibida de los usuarios.
- Descomponer los requisitos de alto nivel a un nivel más detallado y apropiado.
- Obtención de requisitos funcionales a partir de la información de otros requisitos.
- Comprender la importancia relativa de los atributos de calidad.
- Asignación de requisitos a los componentes de software definidos en la arquitectura del sistema.
- Negociar prioridades de implementación.
- Identificar requisitos innecesarios en relación con el alcance definido.

1.3.2.3 Especificación de requisitos

"Es el proceso de documentar el comportamiento requerido de un sistema software, a menudo utilizando una notación de modelado u otro lenguaje de especificación" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 117).

El resultado de la especificación de requisitos es "un conjunto de modelos donde se representan los requisitos desde varias perspectivas (del usuario, estructural, de comportamiento, de flujo de información, entre otros) (Sánchez, Sicilia, & Rodríguez, 2012, págs. 117-118).

Esta actividad de requisitos de software implica la representación y almacenamiento de los conocimientos recopilados de forma persistente y bien organizada (Wiegers & Beatty, 2013, pág. 17). Este documento es denominado *especificación de requisitos del software*, que se conoce comúnmente por sus siglas en inglés como *SRS* (Software Requeriments Specification) (Sánchez, Sicilia, & Rodríguez, 2012, pág. 138).

1.3.2.4 Validación de requisitos

Confirma que se cuenta con la información correcta de requisitos que permitirá a los desarrolladores crear una solución que satisfaga los objetivos de negocio (Wiegers & Beatty, 2013, pág. 17).

Por medio de la examinación de los requisitos se asegura que estos definan el sistema que el cliente y usuarios desean (Sánchez, Sicilia, & Rodríguez, 2012, pág. 118). Wiegers y Beatty (2013, pág. 17), recomiendan como actividades centrales las siguientes:

- La revisión de los requisitos documentados para corregir cualquier problema antes que el grupo de desarrollo los acepte.
- Desarrollo de pruebas de aceptación.

1.3.2.5 Gestión de requisitos

"La gestión de requisitos incluye todas las actividades que mantienen la integridad, exactitud y la circulación de los acuerdos de requisitos a lo largo del proyecto" (Wiegers & Beatty, 2013, pág. 458).

Entre las actividades de gestión de requisitos se encuentran (Wiegers & Beatty, 2013, págs. 17-18):

- Definir la línea base de los requisitos, una instantánea en el tiempo que representa un conjunto acordado, revisado y aprobado de requisitos funcionales y no funcionales, a menudo para una versión de producto especifica o una iteración de desarrollo.
- Evaluar el impacto de los cambios de requisitos propuestos e incorporar cambios aprobados en el proyecto de manera controlada.
- Mantener los planes del proyecto actualizados con los requisitos a medida evolucionan.
- Negociación de nuevos compromisos basados en el impacto estimado de cambios en los requisitos.
- Definir las relaciones y dependencias que existen entre los requisitos.
- Seguimiento de los requisitos individuales a sus correspondientes diseños, códigos fuente y pruebas.
- Seguimiento del estado de los requisitos y cambio de actividad en todo el proyecto.

2. Marco Metodológico

2.1 Tema de Investigación

"Un framework de buenas prácticas para los requisitos de software de aplicaciones Web aplicables al contexto salvadoreño".

2.2 Objetivos

2.2.1 Objetivo general

Construir un framework de buenas prácticas relacionadas con los requisitos de software de aplicaciones Web aplicables al contexto salvadoreño.

2.2.2 Objetivos específicos

1. Identificar las buenas prácticas relacionadas con los requisitos de software por medio de la revisión bibliográfica.
2. Elaborar un framework propuesto de buenas prácticas relacionadas con los requisitos de software por medio de criterios de selección que fueron identificados y definidos durante la investigación (Ver sección 2.8.1.2).
3. Evaluar el framework propuesto de buenas prácticas relacionadas con los requisitos de software, mediante el juicio de expertos, para establecer su utilización en el desarrollo de aplicaciones Web en pequeñas y medianas empresas de El Salvador.

2.3 Preguntas de Investigación

1. ¿Qué buenas prácticas relacionadas a los requisitos de software se identifican en la revisión bibliográfica?
2. ¿Qué buenas prácticas de las identificadas en la revisión bibliográfica pueden ser consideradas para el framework propuesto y qué criterios de caracterización y selección de las buenas prácticas pueden utilizarse?

3. ¿Qué buenas prácticas del framework propuesto consideran los expertos nacionales son utilizadas en el desarrollo de aplicaciones Web en la pequeña y mediana empresa salvadoreña?

2.4 Justificación

F. F. Brooks afirma que "lo más complicado de construir un sistema software es decidir exactamente qué construir […] Ninguna otra parte del trabajo afecta tan negativamente al resultado si se hace mal. Ninguna es tan difícil de rectificar" (Sánchez, Sicilia, & Rodríguez, 2012, pág. 111). Si no se corrigen errores en los requisitos de software de forma temprana, afectará incrementando los costos del proyecto, en el peor de los casos las correcciones implicarán cambios de la arquitectura que requieren comenzar de nuevo, también ocurren escenarios en los que el software se finaliza pero no cumple con las necesidades del usuario final (Wiegers & Beatty, 2013, págs. 19-20).

Varios estudios sugieren que los errores introducidos durante las actividades de los requisitos representan del 40 al 50 por ciento de todos los defectos encontrados en un producto de software (Wiegers & Beatty, 2013, pág. 4). Por lo anterior, es crucial darle la importancia que se debe a los requisitos de software y a las prácticas relacionadas a ellos ya que esto aporta una ventaja competitiva desde el punto de vista de un mercado creciente en el área de desarrollo de software, que deriva en mayor eficiencia en el proceso de desarrollo y calidad en el producto de software (Wiegers & Beatty, 2013, págs. 19-23).

Es relevante el aporte de esta investigación ya que sintetizará la riqueza de buenas prácticas existentes y el aporte de expertos nacionales, interesados en mejorar la calidad de los requisitos de software.

La adopción del instrumento resultante de la investigación, un framework de buenas prácticas enfocadas a los requisitos de software, podría beneficiar a profesionales y empresas involucradas e interesadas en el proceso de requisitos de software, logrando a mediano y largo plazo efectos como: la optimización del uso de los recursos, aumento de la productividad y por ende, una mayor ventaja competitiva.

En El Salvador existen empresas dedicadas a la producción y comercialización del software, otras poseen áreas o departamentos destinados a su construcción para su uso interno, estas empresas deben afrontar retos como la competencia en el mercado, y principalmente el de adaptar las buenas prácticas que han resultado ser útiles en otros contextos, lo que se considera un factor importante en la calidad del software que se produce.

Además, facilitará a las empresas adoptar un framework de buenas prácticas sintetizado por lo que le será más simple de seguir y comprender, siguiendo las líneas generales sobre requisitos de software y de cada una de las actividades que lo componen.

2.5 Alcances

1. Se identificarán y seleccionarán las buenas prácticas relacionadas con los requisitos de software, establecidas en las actividades de obtención, análisis, especificación, validación y gestión de los requisitos del software.
2. Las buenas prácticas serán identificadas de algunas fuentes como: frameworks de desarrollo de software, de gestión de proyectos, de definición y mejora de procesos; así como modelos, estándares y métodos o metodologías relacionadas al desarrollo de software con fecha de publicación superior al año 2000.
3. Se construirá una propuesta de framework de buenas prácticas para los requisitos de software procesadas de la revisión bibliográfica de normas, modelos, frameworks,

metodologías, métodos, recopilaciones de buenas prácticas, investigaciones, entre otras fuentes. Las buenas prácticas serán caracterizadas y seleccionadas de acuerdo a criterios que serán identificados y definidos durante la investigación.

El framework propuesto se usará para construir el instrumento de la investigación cuantitativa-cualitativa: el cuestionario dirigido, cuyo objetivo será determinar si cada buena práctica del framework propuesto se usa o no en el contexto que la investigación contempla y también dará lugar al encuestado a justificar o brindar explicaciones.

4. Se evaluarán, utilizando el cuestionario dirigido, bajo el juicio de expertos cada una de las buenas prácticas de requisitos de software, con el objetivo de validar su utilización en el desarrollo de aplicaciones Web en el contexto salvadoreño, haciendo énfasis en pequeñas y medianas empresas. Los expertos serán profesionales que residan en El Salvador y trabajen en pequeñas y medianas empresas salvadoreñas.

2.6 Limitaciones

El 72.7% de las fuentes bibliográficas están en idioma inglés, este hecho no altera los resultados más si podría generar diferencias en cuanto a la traducción.

2.7 Método y Diseño de Investigación

La primera etapa de investigación consistió en la *revisión bibliográfica*, donde se consultó y obtuvo las fuentes relevantes para el problema de investigación. En la segunda etapa se realizó una *investigación de campo,* en la cual se utilizó el método *cuantitativo-cualitativo,* también conocido como método mixto.

La meta de la investigación mixta no es reemplazar a la investigación cuantitativa ni a la investigación cualitativa, sino utilizar las fortalezas de ambos tipos de indagación combinándolas

y tratando de minimizar sus debilidades potenciales (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 544).

La *investigación cuantitativa* se utilizó con el objetivo de explorar y entender el problema de estudio, esto debido a que en el enfoque cuantitativo "la literatura representa un papel crucial, guía a la investigación. Es fundamental para la definición de la teoría, el diseño y demás etapas del proceso" (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 12). Por otra parte, la *investigación cualitativa* fue utilizada con el objetivo de profundizar en los hallazgos, ya que este método proporciona profundidad a los datos, dispersión, riqueza interpretativa y contextualización (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 12).

Ya que ambas investigaciones (cuantitativa-cualitativa) se realizaron al mismo tiempo se considera del tipo *concurrente.*

La utilización de ambos métodos es debido a la conveniencia de tener diversas fuentes y tipos de datos, métodos de recolección de datos y la posibilidad de triangulación de los datos para validarlos (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, págs. 549-551).

El enfoque cuantitativo de la investigación, tuvo un alcance descriptivo, ya que tiene como propósito la descripción de eventos, situaciones representativas de un fenómeno o unidad de análisis especifica (Baray, 2006, pág. 44). A la vez se clasificó en un diseño *no experimental* ya que, registró y analizó los resultados obtenidos de los aportes de expertos en un momento determinado y único. También, se consideró de tipo transversal (Hernández Sampieri, Fernández Collado, & Baptista Lucio, pág. 151) porque indagó la incidencia y los valores que manifiesta una o más variables en un grupo de personas (Baray, 2006, pág. 44).

El alcance de la investigación cualitativa tuvo un diseño fenomenológico ya que se busca comprender la perspectiva de los participantes acerca del fenómeno, profundizar en su experiencia y opiniones (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 364).

2.8 Descripción de la Metodología

Para lograr los objetivos, la investigación se subdividió en dos etapas:

2.8.1 Primera etapa

La primera etapa, se realizó por medio de la revisión bibliográfica, la cual dio inicio con la selección de las *fuentes bibliográficas*, de las que se extrajo un *banco de buenas prácticas* de requisitos de software, el cual permitió obtener un *listado* que se utilizó para la construcción del instrumento de investigación y *framework propuesto*.

La revisión bibliográfica supone hacer una investigación documental. Baena citado por Baray, (2006, pág. 49) la define como "una técnica que consiste en la selección y recopilación de información por medio de la lectura crítica de documentos y materiales bibliográficos de bibliotecas, hemerotecas, centros de documentación e información*"*.

2.8.1.1 Selección de fuentes bibliográficas

Consistió en la identificación de la bibliografía que define frameworks de desarrollo de software, de gestión de proyectos, de definición y mejora de procesos, así como modelos, estándares y métodos o metodologías relacionadas a los requisitos de software.

Las fuentes bibliográficas fueron seleccionadas a partir de los criterios siguientes:

Tabla 1
Criterios de selección de las fuentes

Criterios	Descripción
Última versión publicada	Toda la bibliografía deberá contar con fecha de publicación superior al año 2000 y ser la última edición publicada.
Alcance	La fuente deberá proporcionar buenas prácticas enfocadas o relacionadas a los requisitos de software.

Nota. Elaboración propia.

Las fuentes bibliográficas se detallan en el capítulo 3, sección 3.1.

2.8.1.2 Creación del banco de buenas prácticas de requisitos de software

De las fuentes bibliográficas seleccionadas en la etapa anterior, se extrajo las buenas prácticas por autor, se codificaron y clasificaron de acuerdo con las actividades de requisitos de software (obtención, análisis, especificación, validación y gestión). El banco de buenas prácticas se detalla en el capítulo 3, sección 3.2.

Para la selección de las buenas prácticas, se hizo uso del criterio de elegibilidad siguiente:

Tabla 2
Criterio de selección de buenas practicas

Criterio	Descripción
Alcance	Toda buena práctica debe estar relacionada a los requisitos de software y a alguna de sus actividades.

Nota. Elaboración propia.

2.8.1.3 Análisis y síntesis de buenas prácticas

Se utilizó la técnica *síntesis documental*, para organizar la información tabularmente, para eliminar duplicidades, con la finalidad de integrar y sintetizar las buenas prácticas a una única definición que contiene los elementos esenciales de cada autor (Ver sección 3.3).

2.8.1.4 Listado de buenas prácticas

El *listado de buenas prácticas* se obtuvo de las principales buenas prácticas sintetizadas, basado en su frecuencia, las cuales se utilizaron para establecer el framework propuesto a validarse (Ver sección 3.4).

2.8.2 Segunda etapa

En la segunda etapa se realizó una *investigación de campo,* utilizando un cuestionario guiado para la recolección de datos, con el cual se buscó establecer la utilización de las buenas prácticas en el desarrollo de aplicaciones Web en el contexto salvadoreño. Los participantes fueron profesionales de pequeñas y medianas empresas (Ver sección 2.8.2.2), quienes se seleccionaron intencionalmente a partir de criterios de inclusión que se establecieron en la investigación (Ver sección 2.8.2.3), para garantizar un nivel de experiencia y conocimientos aceptables.

2.8.2.1 Generalidades de la investigación de campo

La investigación de campo se realizó bajo las siguientes generalidades:

- **Dominio de la investigación:** se tomó como dominio de la investigación cada una de las actividades que componen a los requisitos de software, como lo son: obtención, análisis, especificación, validación y gestión de requisitos.
- **Unidad de análisis (objeto de la investigación):** buenas prácticas de requisitos de software.
- **Variables:** utilización de las buenas prácticas.
- **Valores posibles:** no se utiliza, se utiliza.
- **Técnica:** juicio de expertos.

2.8.2.2 Población y muestra

- **Población:** profesionales de las pequeñas y medianas empresas en El Salvador, con experiencia en proyectos de desarrollo de aplicaciones Web, ya que en este ámbito es en el que interesó comprobar la utilización de las buenas prácticas de requisitos de software.
- **Muestra:** se realizó seleccionando una muestra no probabilística, intencional, basado en criterios establecidos con el fin de incluir expertos que resulten relevantes (Ver sección 2.8.2.3).

Este estudio se enfocó en los profesionales de las pequeñas y medianas empresas como parte de la población que cuenta con acceso a computadoras e internet más representativo y que necesita mayor apoyo en asesoramiento que permita aumentar su productividad utilizando los recursos tecnológicos con que cuenta (SELA, 2005, pág. 32) .

Según el censo de la DIGESTYC (Dirección General de Estadística y Censo) (2012, pág. 21) la micro empresa representa un 95.8%, la pequeña y mediana empresa representan el 3.9% y la gran empresa el 0.3% del total de empresas salvadoreñas.

Según encuesta a PYMES en Centroamérica 2004, de las microempresas solo el 46% tiene acceso a una computadora y solo el 35% cuenta con acceso a internet (SELA, 2005, pág. 31), razón por la que se ha excluido a las microempresas.

También se ha excluido a la gran empresa porque la adopción de las TICs (Tecnologías de la Información y Comunicación) está siendo llevada a cabo, primeramente, por parte de las empresas grandes, lo que tiende a acrecentar aún más la brecha en productividad que existe entre las empresas grandes y pequeñas o medianas (SELA, 2005, pág. 3). En El Salvador la gran

empresa representa un 0.3% del total de empresas, mucho menos representativo (MINEC - DIGESTYC, 2012, pág. 6).

2.8.2.3 Caracterización de la muestra

Según Almenara & Llorente (2013, pág. 14) *"*la selección del número de expertos depende de aspectos como la facilidad para acceder a ellos o la posibilidad de conocer expertos suficientes sobre la temática objeto de la investigación*"*.

Los criterios para la selección de los participantes en la investigación fueron:

- Cumplir con al menos seis años de experiencia laboral enfocado en el desarrollo de software.
- Haber desempeñado algún rol como: analista de negocio/procesos, analista de sistemas, arquitecto de software, programador, administrador de proyectos, asegurador de la calidad, probador de software e implementador.
- Ser profesionales que residan en El Salvador y trabajen en empresas salvadoreñas, catalogadas como pequeñas o medianas, de cualquier sector.

2.8.2.4 Tipo de muestreo

El tipo de muestra se estableció como *no probabilístico e intencional* para ambos métodos de investigación, ya que no se busca generalizar los hallazgos en otro contextos. Los profesionales fueron seleccionados según el propósito de la investigación, sin pretender que los casos fueran representativos de la población (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, págs. 189-190).

2.8.2.5 Tamaño de la muestra

Se determinó una muestra de *diez* expertos, tomando en consideración los tamaños recomendados para la estrategia de investigación seleccionada (Ver inciso 2.7), que según

Creswell pueden ser entre 5 y 25 participantes; y según Morse debe ser al menos de 6 participantes (Mason, 2010).

2.9 Instrumento de Recolección de Datos

Inicialmente se planteó el utilizar dos instrumentos para la recolección de datos: un cuestionario y una posterior entrevista para profundizar los datos obtenidos del cuestionario, pero se llegó a identificar que un *cuestionario guiado con discusión para entender el fenómeno* cumplía los propósitos tanto del cuestionario como de la entrevista y complementó de mejor forma las necesidades de información, permitiendo un mejor contacto con el experto.

Para poder aplicar el cuestionario guiado se optó por el contexto de la entrevista, orientando en el tránsito del instrumento y proporcionando explicaciones breves pero suficientes (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 239).

El tipo de preguntas definidas para el instrumento fueron del tipo mixta: cerradas y abiertas, donde para establecer la utilización de las buenas prácticas de requisitos de software por medio de una descripción numérica se utilizó las del tipo cerrada, y para incluir las opiniones de los expertos y dar una mejor respuesta a las preguntas de investigación se utilizó las preguntas del tipo abierta por cada actividad de requisitos.

2.9.1 Cuestionario guiado

El diseño del cuestionario guiado utilizando un formato de entrevista (Ver anexo No. 1), se realizó en base al *listado de buenas prácticas con mayor frecuencia* obtenidas en la primera etapa de la investigación, el cual fue administrado con el siguiente protocolo:

- Descripción de la investigación: se le presentó al entrevistado la descripción general de la investigación y las indicaciones para desarrollar el cuestionario. Consultándole si permitía realizar la grabación de la entrevista.

- Desarrollo del cuestionario: se le expresó cada pregunta al entrevistado, retroalimentándolo donde él lo solicitara y tomando apuntes de los comentarios compartidos por el experto.
- Cierre de la entrevista: se le agradeció la participación y tiempo proporcionado.

2.9.2 Análisis y organización de datos

En el proceso cuantitativo primero se recolectan todos los datos y posteriormente se analizan, mientras que en la investigación cualitativa no es así, la recolección y el análisis ocurren prácticamente en paralelo (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 439). Dada la naturaleza mixta de la investigación, la recolección de datos fue simultánea (datos cuantitativos y cualitativos), mientras que el análisis se realizó al final, debido a la influencia del proceso cuantitativo.

En la parte cualitativa se recibieron datos a los que se les tuvo que dar una estructura (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 439). Y para los datos cuantitativos se utilizó la *distribución de frecuencia y gráficas* debido a que se analizan con métodos descriptivos.

A la vez se definió que si al analizar cada caso (cuestionario) se encontraban inconsistencias o falta de claridad en el entendimiento del problema planteado, se regresaría a campo a recolectar más datos (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010, pág. 441).

2.10 Procesamiento de la Información

La información se alojó en una herramienta de cálculo, donde se procesó los datos cuantitativos y se obtuvo las frecuencias, porcentajes y gráficas que representan la proporción en el que cada buena práctica es utilizada según los expertos y de *las preguntas abiertas* se

identificó aspectos relevantes y de interés, así como las razones de la no utilización de una buena práctica en el contexto salvadoreño en pequeñas y medianas empresas.

La información resultante se analizó y se relacionó (resultados cualitativos y cuantitativos), mediante la triangulación de datos y de esto se describió los resultados que lanzaron elementos claves identificados.

2.11 Entregables

La presente investigación producirá los siguientes entregables:

1. **Marco conceptual:** expone los conceptos que aborda el tema de investigación, el cual permite comprender y analizar el tema propuesto (Capitulo 1 del entregable 5).
2. **Marco metodológico:** contiene la explicación de los mecanismos a utilizarse para el análisis de la problemática de investigación. Es decir, el tipo de investigación que se realizó, métodos, técnicas y procedimientos aplicados (Capitulo 2 del entregable 5).
3. **Resultados de la revisión bibliográfica:** identifica las fuentes de buenas prácticas de requisitos de software que forman el banco de conocimiento de la investigación, se clasifican y organizan para cada actividad de la fase de requisitos de software (Capitulo 3 del entregable 5).
4. **Resultados de la investigación de campo:** presenta los resultados de la investigación cuantitativa y cualitativa (Capitulo 4 del entregable 5).
5. **Tesina:** en este documento final se integra todos los entregables anteriores y los resultados de la investigación.
6. **Artículo:** es un resumen de toda la tesina para su publicación.

2.12 Matriz de Congruencia

En el Anexo No. 2, se presenta la matriz de congruencia de la investigación.

3. Resultados de la Revisión Bibliográfica

3.1 Fuentes Bibliográficas de Buenas Prácticas de Requisitos de Software

La revisión bibliográfica permitió identificar una serie de fuentes de información documental, las cuales fueron incluidas para la extracción de las buenas prácticas de los requisitos de software. La siguiente tabla identifica la documentación utilizada en su última versión publicada a la fecha de la investigación, el identificador utilizado en el banco de buenas prácticas y el tipo de clasificación del documento.

Tabla 3
Fuentes de información documental seleccionadas

Identificación documental	Identificador	Tipo	Año de publicación
CMMI DEV, Versión 1.3, SEI.	CMMI	Framework y Modelo	2010
Guía de los Fundamentos para Dirección de Proyectos, Quinta edición, PMI.	PMBOK	Framework y Estándar	2012
Requirements Engineering-Fundamentals, Principles, and Techniques, Klaus Pohl, Primera edición.	Klaus Pohl	Framework	2010
Agile Software Requeriments-Lean Requirements Practices For Teams, Programs, And The Enterprise, Dean Leffingwell, primera edición.	Dean Leffingwell	Libro	2011
Rational Unified Process: Overview, Rational Software Corporation, 2001.	RUP	Metodología	2001
Guía Práctica de Gestión de Requisitos, INTECO.	INTECO	Framework	2008
Software Engineering Quality Practices, Ronald Kirk Kandt, primera edición.	Ronald Kirk	Libro	2006
Software Engineering Best Practices, Capers Jones, primera edición.	Casper Jones	Libro	2010
Software Requirements, Wiegers & Beatty, tercera edición.	Wieger	Libro	2013
System Engineering Handbook-NASA/SP-2016-610S Rev 2, NASA.	NASA	Framework	2017
SWEBOK, Version 3, IEEE Computer Society.	SWEBOK	Framework	2014

Nota. Elaboración propia, a partir del Anexo 3.

Algunas fuentes fueron excluidas por no proporcionar buenas prácticas directamente relacionadas a los requisitos de software (Ver anexo No. 3), aunque estén dirigidas a mejorar el

proceso de desarrollo de software, solo describen a nivel general aspectos como: conformación de equipos de desarrollo de software, distribución de roles, gestión del tiempo, mejora de la productividad de los programadores, definición de actividades y procesos, referencias a otros estándares y modelos principales que ya fueron incluidos, como CMMI DEV, SWEBOK, PMBOK, entre otros.

Entre las excluidas se encuentran: MoProsoft (Modelo de Procesos para la Industria del Software), ISO 9001:2000, ISO/IEC 29110, PSP (Personal Software Process), TSP (Team Software Process), Ingeniería del software un enfoque práctico desde la guía SWEBOK - Salvador Sánchez, Ingeniería del software - Ian sommerville e Ingeniería del software un enfoque práctico - Roger Pressman.

3.2 Banco de Buenas Prácticas

De las *fuentes de información documental seleccionadas* se extrajo y clasificó cada buena práctica según la actividad de requisitos de software por medio del criterio definido en la tabla 2 de la sección 2.8.1.2, haciendo un total de 201 buenas prácticas que se detallan en el Anexo No. 4, y cuyo resumen es el siguiente:

Tabla 4
Buenas prácticas de las fuentes bibliográficas

Actividad de requisitos	Total de buenas prácticas
Obtención	63
Análisis	33
Especificación	30
Validación	38
Gestión	37

Nota. Elaboración propia, a partir del Anexo 4.

Luego de extraer las buenas prácticas de las fuentes bibliográficas se creó un documento llamado “Banco buenas prácticas”, separando en hojas cada actividad de requisitos de software con la siguiente estructura (Ver anexo No. 5):

- Correlativo de ordenamiento: utilizado para una numeración general.
- Autor: identifica cada una de las fuentes documentales.
- Código banco: establecido para la identificación de cada buena práctica en el banco.
- Página: especifica el número de página del origen de la buena práctica.
- Código síntesis: identifica las buenas prácticas con diversos orígenes del banco de buenas prácticas con significado común.
- Enunciado de la buena práctica: redacción de la buena práctica por actividad de requisitos de software.

3.3 Síntesis de Buenas Prácticas

Dado a que algunos autores utilizan terminologías diferentes para enunciar las buenas prácticas con el mismo significado, se identificó las prácticas comunes o relacionadas por medio del código llamado “Código síntesis”, que sirvió para la depuración y posterior reescritura de cada buena práctica. Se obtuvieron un total de 72 buenas prácticas sintetizadas que se detallan en el Anexo No. 6 y se resumen en la tabla siguiente:

Tabla 5
Buenas prácticas sintetizadas

Actividad de requisitos	Total de buenas prácticas
Obtención	16
Análisis	16
Especificación	11
Validación	15
Gestión	14

Nota. Elaboración propia, a partir del Anexo 6.

La estructura tabular para el ordenamiento de las buenas prácticas sintetizadas fue la siguiente:

- Código síntesis: identifica las buenas prácticas con diversos orígenes del banco de buenas prácticas con significado común.
- Enunciado de la buena práctica: redacción de las buenas prácticas.
- Autor/Fuente documental: identifica el "código banco" equivalente con la buena práctica sintetizada.
- Frecuencia: identifica la frecuencia de aparición de la buena práctica sintetizada por fuente documental.

3.4 Listado de Buenas Prácticas del Framework Propuesto

Luego de obtener el listado de las buenas prácticas sintetizadas, manteniendo el orden de las actividades de requisitos, se extrajeron las buenas prácticas en base a una frecuencia de aparición mayor o igual a tres.

Debido a la relevancia identificada en algunas buenas prácticas, en la actividad de obtención y gestión de los requisitos de software se decidió incluir dos buenas prácticas adicionales por formar parte de la guía SWEBOK y por ser referenciadas dentro de otras prácticas principales.

El resultado final fueron 27 buenas prácticas (Ver anexo No. 7) las cuales representan el framework propuesto.

A continuación, se detallan las buenas prácticas por actividad de requisitos de software:

3.4.1 Listado de buenas prácticas de obtención de requisitos

- Utilizar técnicas para obtención de requisitos (entrevistas, cuestionarios, observación, historia de usuarios, casos de uso, escenarios, prototipos, talleres de facilitación,

análisis de documentos, análisis de tormentas de ideas, examinar informes de problemas, etc.)

- Identificar, verificar e involucrar a las partes interesadas (stakeholders) y administrar los diferentes puntos de vista.
- Definir y documentar claramente el alcance y visión del proyecto.
- Identificar y evaluar todas las fuentes potenciales de requisitos (políticas de negocio, procedimientos, estándares, lecciones aprendidas, archivos de proyectos anteriores, requisitos del entorno del negocio, componentes de productos heredados, estatutos reguladores, etc.)
- Establecer y mantener acuerdos con los stakeholders (negociación y resolución de conflictos).
- Comunicar de manera efectiva y utilizar modelos de comunicación.

3.4.2 Listado de buenas prácticas de análisis de requisitos

- Priorizar los requisitos de software.
- Analizar los riesgos y viabilidad de los requisitos.
- Construcción de modelos técnicos (diagramas de casos de uso, modelos de flujo de datos, modelos de estado, modelos basados en objetivos, interacciones de usuarios, modelos de objetos, modelos de datos, entidad relación, transición de estado, entre otros), simulaciones y prototipos.
- Desarrollar modelado conceptual (contexto).
- Descomposición lógica (clasificación, relación y jerarquía de requisitos).

3.4.3 Listado de buenas prácticas de especificación de requisitos

- Crear artefactos (documentar la visión y arquitectura del negocio, reglas del negocio, requisitos funcionales y no funcionales, casos de uso y objetos del negocio, atributos de calidad, etc.)
- Documentar los requisitos de forma correcta (completos, consistentes, dentro del alcance del proyecto, capaces de ser probados) en un lenguaje natural (descripción no técnica) y al mismo tiempo en lenguaje de requisitos formal utilizando reglas de especificación de requisitos (eliminar todos los pronombres de la especificación de requisitos, sustituyéndolos por los sujetos).
- Definir los requisitos teniendo en cuenta la perspectiva del usuario.
- Adoptar plantillas de documentos de requisitos (visión y alcance, casos de uso, especificación de requisitos – SRS, etc.)
- Escribir especificación de requisitos no ambigua (cada requisito debe ser comprensible por sí solo).

3.4.4 Listado de buenas prácticas de validación de requisitos

- Establecer para cada requisito criterios de validación y aceptación de las partes interesadas.
- Validar la documentación de los requisitos funcionales y técnicos en su contenido, documentación y acuerdos.
- Analizar y validar los requisitos (trazabilidad, supuestos válidos, esenciales, consistentes con el diseño).
- Revisar los requisitos y utilizar inspecciones de requisitos formales tanto con usuarios como con proveedores.

- Utilización de prototipos para validar la interpretación de los requisitos de software (las características clave de las nuevas aplicaciones) y para obtener nuevos requisitos.

3.4.5 Listado de buenas prácticas de gestión de requisitos

- Gestionar de forma efectiva y eficiente el control de cambios de los requisitos (nuevos, cambios y eliminación), utilizando un plan y herramientas de configuración y control de cambios.
- Llevar el seguimiento de la vida de un requisito, utilizando la técnica de matriz de trazabilidad de requisitos.
- Establecer la línea base de los requisitos para asegurar que cualquier modificación en los requisitos que cambien la línea base se trate como cambios de alcance.
- Supervisar y dar seguimiento al estado de los requisitos de software (especificado, verificado, analizado, entre otros) mediante un proceso definido.
- Medir el número y la gravedad de los defectos en los requisitos definidos.
- Formar a los analistas de requisitos para asegurar que tienen el conocimiento, entre otros aspectos, de cómo escribir y gestionar buenos requisitos.

3.5 Framework Propuesto

Se diseñó un documento el cual describe cada buena práctica de los requisitos de software, según las fuentes de información documental seleccionadas (Ver anexo No. 8).

4. Resultados de la Investigación de Campo

Para la investigación se estableció inicialmente se entrevistarían ocho expertos (Ver sección 2.8.2.5), pero en el proceso se depuro dos entrevistas por no contar con respuestas completas. Sin embargo, debido a que algunas respuestas no satisfacían la necesidad de información para cumplir con los objetivos de la investigación, se decidió entrevistar dos expertos más, alcanzando una muestra de d*iez* expertos distribuidos en siete de mediana empresa y tres de pequeña empresa salvadoreña.

El cuestionario guiado fue dividido en dos secciones (Ver anexo No. 1):

- La primera sección identifica el experto: nivel educativo, tamaño de empresa, roles desempeñados, participación en las actividades de requisitos y años de experiencia.
- La segunda sección establece la utilización de las buenas prácticas, clasificadas por actividad de requisitos de software, por medio de 27 preguntas del tipo cerradas y cinco abiertas.

Las respuestas obtenidas por los participantes pueden ser consultadas en el anexo No. 9.

A continuación, se proporcionan los resultados obtenidos.

4.1 Parte I. Identificación del Experto

4.1.1 Nivel educativo

El nivel educativo de la mayoría de los participantes fue de ingeniería (60%), seguido de licenciatura (30%) y maestría (10%).

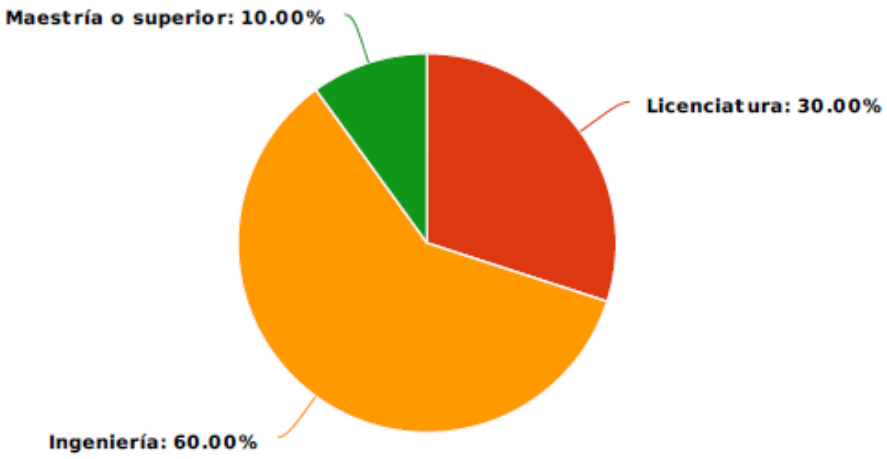

Figura 5. Nivel educativo. Elaboración propia.

4.1.2 Tamaño de la empresa donde labora

El 70% de los participantes labora en la mediana empresa y el 30% restante en la pequeña empresa.

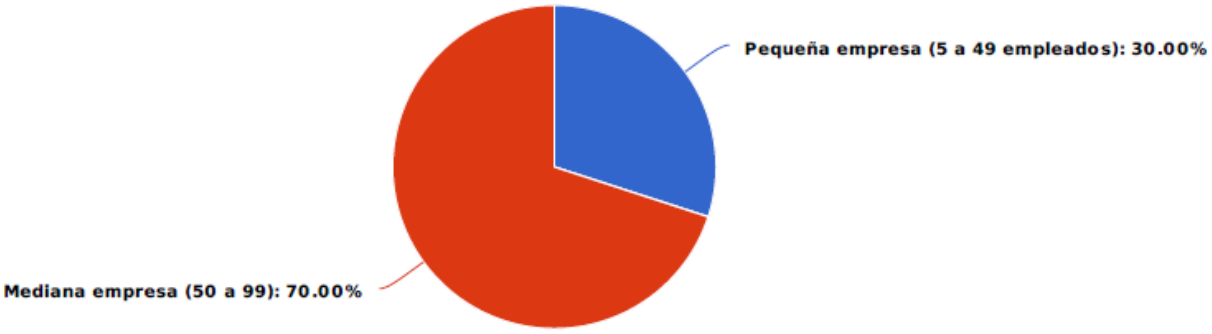

Figura 6. Distribución por tamaño de la empresa. Elaboración propia.

4.1.3 Roles que ha desempeñado en su carrera profesional

Los roles que han desempeñado o desempeñan actualmente los participantes de la investigación son los que se mencionan a continuación (ordenados de mayor a menor): analistas de sistemas, programador, analista de negocios/procesos, arquitecto de software e implementador.

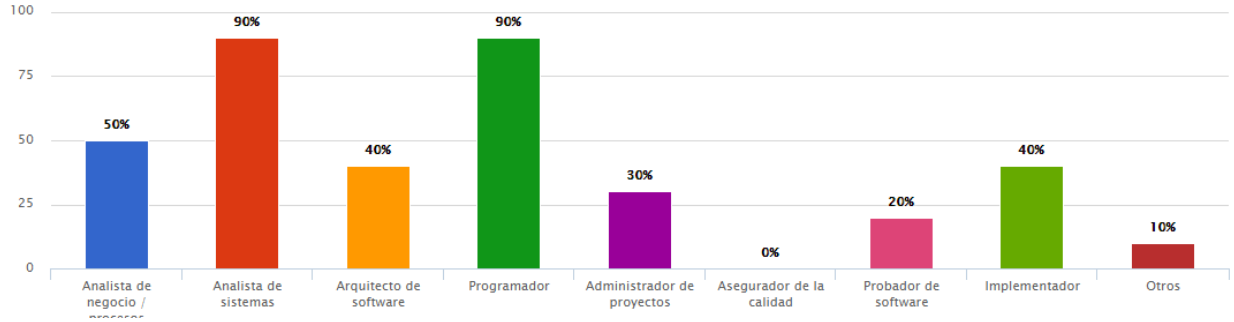

Figura 7. Distribución por roles desempeñados. Elaboración propia.

4.1.4 Participación en actividades de requisitos de software

Todos los participantes afirman haber participado en la actividad de obtención y análisis de requisitos de software, seguido de un 80% en la especificación, 70% en la validación y solo un 50% en la gestión.

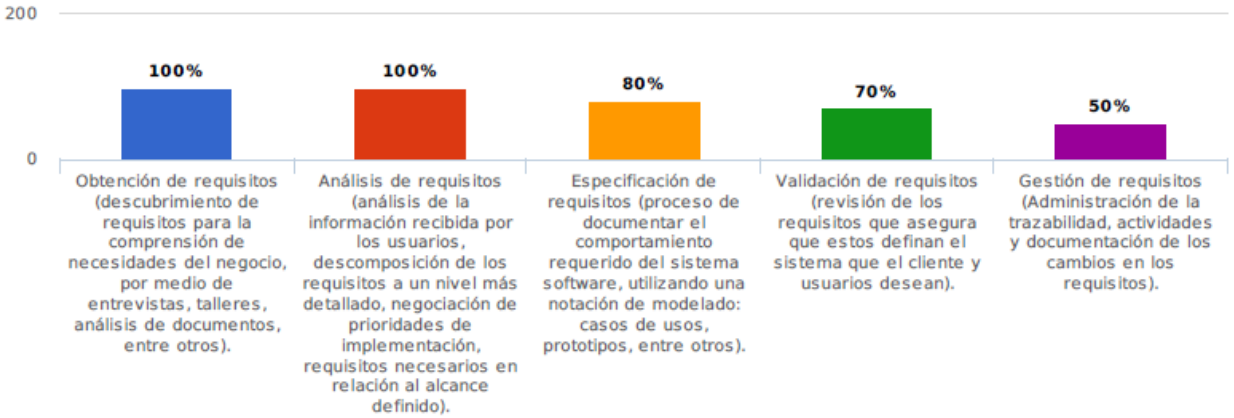

Figura 8. Distribución por participación en actividades de requisitos. Elaboración propia.

4.1.5 Años de experiencia desempeñando puestos o roles relacionados a requisitos de software

Todos los participantes se encontraron dentro del rango de 6 a 10 años de experiencia relacionada a requisitos de software.

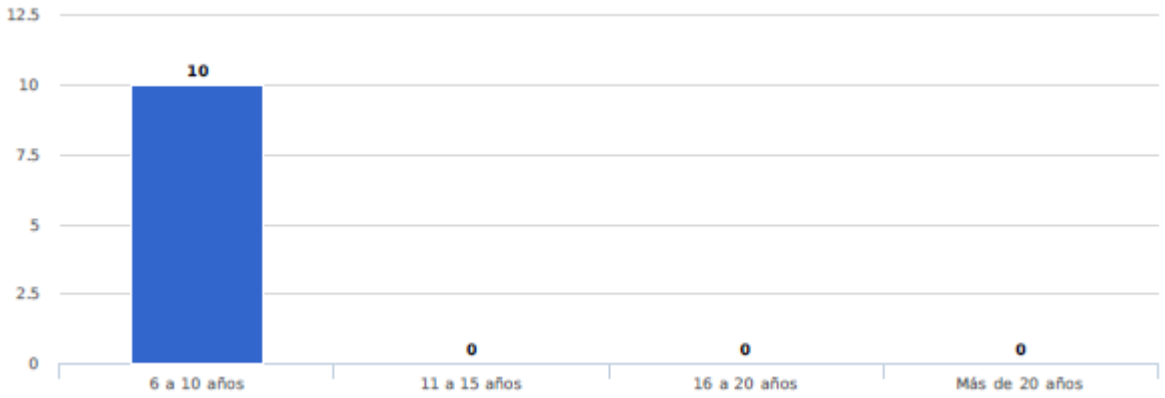

Figura 9. Distribución por años de experiencia. Elaboración propia.

El detalle en años de experiencia de los participantes es el siguiente:

- 6 años: cuatro participantes.
- 7 años: dos participantes.
- 8 años: dos participantes.
- 9 años: un participante.
- 10 años: un participante.

4.2 Parte II. Buenas Practicas de Requisitos de Software

Esta sección tuvo por objetivo establecer en base a la experiencia laboral del participante, que buenas prácticas de requisitos de software se usan en el desarrollo de aplicaciones Web en la pequeña y mediana empresa de El Salvador, y justificar las causas o impedimentos de su utilización.

La siguiente figura se presenta la participación de los participantes (experiencia en la actividad) versus el porcentaje en que afirman utilizar las buenas prácticas propuestas, en cada actividad de requisitos de software.

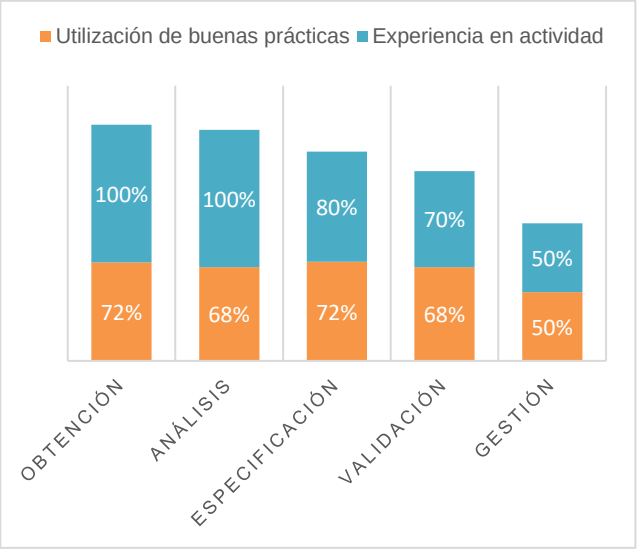

Figura 10. Experiencia vs Utilización. Elaboración propia.

La experiencia de los participantes sobre las actividades de requisitos esta principalmente enfocadas a la obtención (100%) y análisis (100%), en segundo lugar, a la especificación (80%), en tercer lugar, a la validación (70%) y en último lugar a la gestión (50%).

Según los participantes las buenas prácticas relacionadas a las actividades de obtención (71.66%) y especificación (72%) de requisitos son las más utilizadas. Las buenas prácticas dentro de las actividades de análisis y validación son menos utilizadas (68%) que las anteriores, pero en su mayoría no parecieron ser un tema desconocido para los expertos, en contraste con la gestión de requisitos de la cual se observó desconocimiento de la mayoría de buenas prácticas que les fueron presentadas (50%).

4.2.1 Actividad de obtención

4.2.1.1 Utilizar técnicas para obtención de requisitos

Entre las técnicas de obtención de requisitos se encuentran las entrevistas, cuestionarios, observación, historias de usuarios, casos de uso, escenarios, prototipos, talleres de facilitación, análisis de documentos, análisis de tormentas de ideas, examinar informes de problemas, etc.

Los participantes respondieron que la utilización de esta buena práctica es del 100%.

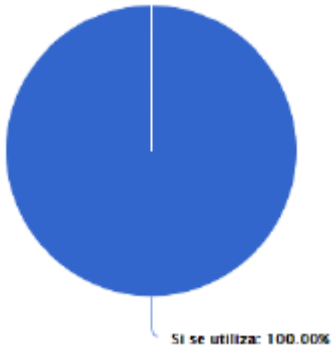

Figura 11. Utilizar técnicas para obtención de requisitos. Elaboración propia.

Los participantes expresaron que la utilización de técnicas de obtención de requisitos permite indagar en la diversidad de pensamiento e interpretaciones de los usuarios, identificando *"puntos críticos secundarios"* que impactan a los requisitos y que no son tan visibles para el usuario, ejemplo de ellos: leyes fiscales, procesos internos particulares aplicados al proceso estándar, entre otros. Así mismo dichas técnicas permiten corroborar el requisito, ya que el usuario proporciona inicialmente una idea general del problema lo que se soluciona con la retroalimentación para aterrizar la idea.

Aunque todos afirmaron utilizarlo, algunos aclararon contar con la dificultad del poco involucramiento del usuario.

4.2.1.2 Identificar, verificar e involucrar a las partes interesadas (stakeholders) y administrar los diferentes puntos de vista

La mayoría de los participantes (80%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

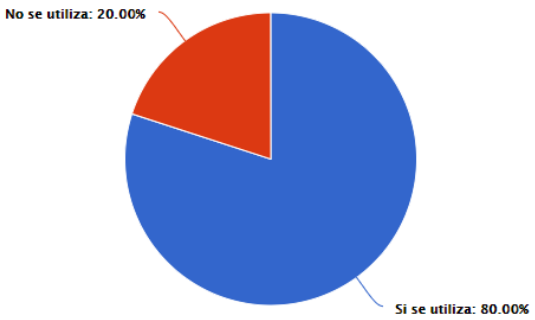

Figura 12. Identificar, verificar e involucrar a las partes interesadas y administrar los diferentes puntos de vista. Elaboración propia.

Los participantes mencionan que entre los beneficios que trae utilizar esta buena práctica están: obtener aportes o puntos de vista desde diferentes experiencias en el negocio, se eliminan posibles contratiempos para pasar a producción, disminuye el retrabajo y se logra una visión integrada del proyecto con los involucrados, diferente a cuando los requisitos se ven con una sola persona. Por otro lado, quienes no la utilizan lo atribuyen a no contar con procesos de comunicación dentro de las empresas, ni personas que velen por que se cumplan.

4.2.1.3 Definir y documentar claramente el alcance y visión del proyecto

El 50% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 50% opina que no se utiliza.

Figura 13. Definir y documentar claramente el alcance y visión del proyecto. Elaboración propia.

Los participantes concuerdan que esta buena práctica les permite plasmar las ideas y objetivos a cumplir en el proyecto, este documento se utiliza muchas veces como parte del contrato.

Entre las argumentaciones del porque no se utiliza esta buena práctica los participantes expresaron lo siguiente: 1) que en la mayoría de los casos el usuario final o dueño del proceso no expresa claramente el problema a solucionar, y 2) el no contar con una metodología bien definida pasa por alto el control de esta actividad.

Adicionalmente concuerdan que una de las principales consecuencias de no administrar bien el alcance, es que al finalizar el software este termina siendo muy diferente a lo establecido inicialmente.

4.2.1.4 Identificar y evaluar todas las fuentes potenciales de requisitos

Entre las fuentes potenciales de requisitos se encuentran las políticas de negocio, procedimientos, estándares, lecciones aprendidas, archivos de proyectos anteriores, requisitos del entorno del negocio, componentes de productos heredados, estatutos reguladores, entre otros.

El 70 % de los participantes opina que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

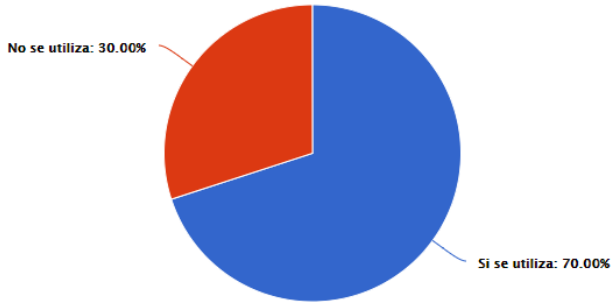

Figura 14. Identificar y evaluar todas las fuentes potenciales de requisitos. Elaboración propia.

Los participantes hacen mención que una de las causas de su no utilización es que generalmente los analistas de sistemas no buscan precedentes de proyectos anteriores y que unos de los puntos claves para especificar un buen requisito es la identificación de estatutos regulatorios y estándares con los que habría que alinearse o a tomar en cuenta.

Así mismo agregan que esta buena práctica ayuda a identificar aspectos técnicos como la infraestructura, los servicios Web, componentes y otros recursos heredados que pueden ser reutilizados para agilizar el desarrollo del software.

4.2.1.5 Establecer y mantener acuerdos con los stakeholders (negociación y resolución de conflictos).

El 80 % de los participantes opina que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 20% opina que no se utiliza.

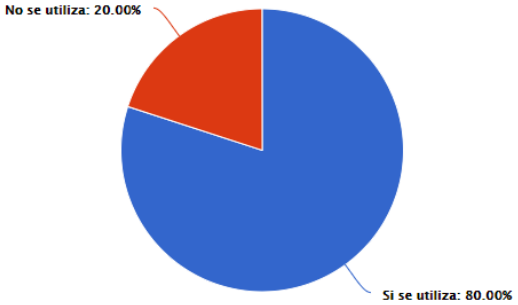

Figura 15. Establecer y mantener acuerdos con los stakeholders. Elaboración propia.

Los participantes mencionan que el establecer acuerdos con los stakeholders ayuda a definir los límites para el alcance del proyecto, a la vez les facilita establecer y aprobar nuevos requisitos que pueden derivar de cambios organizacionales.

Los participantes hacen mención que todos los acuerdos y conflictos deben ser comunicados a todas las partes del proyecto, y en la medida de lo posible evitar figuras autoritarias que solo se encargan de imponer los requisitos sin tomar en cuenta la opinión de las otras partes involucradas. Quienes no la utilizan justifican que no cuentan con una metodología para el desarrollo de software y que no se da el involucramiento de los usuarios.

4.2.1.6 Comunicar de manera efectiva y utilizar modelos de comunicación.

El 50% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 50% opina que no se utiliza.

Figura 16. Comunicar de manera efectiva y utilizar modelos de comunicación. Elaboración propia.

Los participantes mencionan que uno de los principales problemas de su no utilización en los proyectos de desarrollo de software es que la comunicación no fluye correctamente a todos los involucrados y esta se queda en ciertos niveles organizacionales, para poder aplicar esta buena práctica debe definirse un plan de comunicación y delegar a una persona como responsable de su cumplimiento.

4.2.2 Actividad de análisis

4.2.2.1 Priorizar los requisitos de software

El 90% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 10% opina que no se utiliza.

Figura 17. Priorizar los requisitos de software. Elaboración propia.

Los participantes expresan que priorizar los requisitos les permite establecer una ruta, identificando cuales son los requisitos más críticos (mayor impacto e importantes para la organización), así como una estrategia para desarrollarlos. A su vez hacen mención que esta buena práctica les ayuda a realizar una designación más equitativa, según la experiencia del personal. Quienes no utilizan la priorización consideran se debe a que no evalúan el impacto ni la importancia de los requisitos de software.

4.2.2.2 Analizar los riesgos y viabilidad de los requisitos.

El 70% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

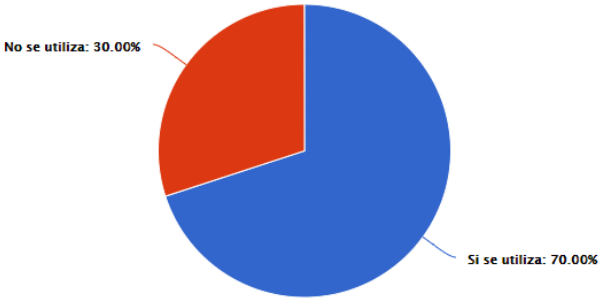

Figura 18. Analizar los riesgos y viabilidad de los requisitos. Elaboración propia.

Los que afirman utilizar la buena práctica, no profundizaron en como realizan el proceso, en algunos casos afirmaron no ser ellos los responsables de efectuarlo y si se realiza no es a conciencia sino por cumplir un requisito que permita pasar a producción. Los restantes participantes manifiestan que esta buena práctica muchas veces no se utiliza debido a las limitaciones de tiempo por lo que los riesgos se van identificando una vez se ha empezado el desarrollo del software.

4.2.2.3 Construir modelos técnicos, simulaciones y prototipos

Entre los modelos técnicos se encuentran los diagramas de casos de uso, modelos de flujo de datos, modelos de estado, modelos basados en objetivos, interacciones de usuarios, modelos de objetos, modelos de datos, entidad relación, transición de estado, entre otros.

El 70% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

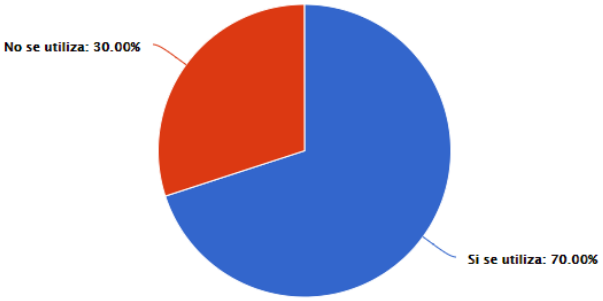

Figura 19. Construir modelos técnicos, simulaciones y prototipos. Elaboración propia.

La mayoría de participantes dicen realizar modelado o diagramación porque les permite un mejor entendimiento de los procesos de negocio y les sirve para definir el problema a solucionar, estimar los tiempos en el plan de trabajo del equipo de desarrollo con una mejor visión de las actividades.

Los participantes hacen hincapié que esta actividad debería ser realizada en conjunto por el arquitecto de software y analista de sistemas, porque les permite realizar un análisis para el diseño e identificar que patrones pueden utilizarse.

Los factores que identifican los participantes del por qué no se utiliza esta buena práctica son: 1) la mayoría de empresas quieren el software para ayer, es decir de urgencia, 2) el recurso humano limitado, una sola persona realiza diferentes roles, la diagramación o modelado no se completa o queda desactualizada y no concuerda con lo desarrollado.

Se identificó que el 30% de los participantes tienden a confundir el modelado técnico con el modelado conceptual.

4.2.2.4 Desarrollar modelado conceptual (contexto)

El 50% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 50% opina que no se utiliza.

Figura 20. Desarrollar modelado conceptual. Elaboración propia.

La mitad de los participantes manifiestan que este tipo de diagrama o modelado les permite tener mejor entendimiento del problema, comprender la lógica del negocio y es el más popular utilizado para cualquier tamaño de empresa o proyecto.

Algunos participantes mencionan que no utilizan este tipo de modelo debido a que se realiza con un alto nivel de abstracción, por lo que lo consideran incompleto e inexacto, ya que puede ignorar aspectos importantes, por lo prefieren utilizar modelados más técnicos para definir límites e interfaces del sistema a desarrollar.

4.2.2.5 Descomposición lógica (clasificación, relación y jerarquía de requisitos)

El 60% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 40% opina que no se utiliza.

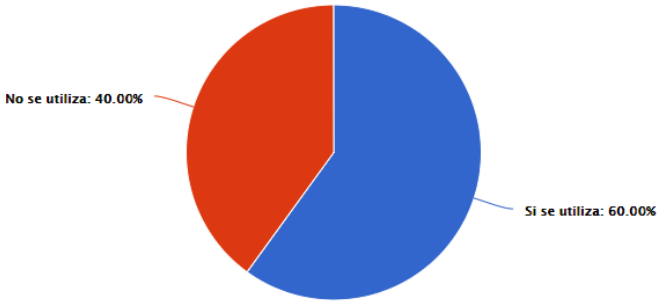

Figura 21. Descomposición lógica. Elaboración propia.

Los participantes indican que tener clasificados los requisitos permite llevar un seguimiento y entendimiento de los mismos, es decir llevar las ideas ordenadas.

Un factor por lo que esta buena práctica no se utiliza es no contar con un proceso definido para realizarlo y que generalmente no se clasifican o identifican los requisitos no funcionales.

4.2.3 Actividad de especificación

4.2.3.1 Crear artefactos

Entre los artefactos se encuentran el documentar la visión y arquitectura del negocio, reglas del negocio, requisitos funcionales y no funcionales, casos de uso y objetos del negocio, atributos de calidad, entre otros.

El 70% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

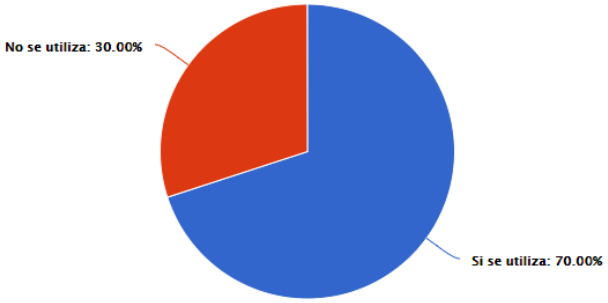

Figura 22. Crear artefactos. Elaboración propia.

En cuanto a la creación de artefactos los participantes concuerdan que les permite tener centralizado el conocimiento, mejorando el tiempo en curva de aprendizaje en los miembros del equipo del proyecto. A la vez expresan que esta tarea la realizan analistas de sistemas, muy pocas veces el programador y que resulta ser muy útil en los casos de rotación del personal, cambios o mantenimiento del software y para realizar buenas prácticas de reutilización.

Algunos participantes opinan que causas de su no utilización son que esta información generalmente se mantiene desactualizada o incompleta. Por otra parte, expresan que en algunas empresas el documentar no es trabajo de valor, por el mal liderazgo, la cultura y porque no hay procedimientos claros y bien definidos.

4.2.3.2 Documentar los requisitos de forma correcta en un lenguaje natural y al mismo tiempo en lenguaje de requisitos formal utilizando reglas de especificación de requisitos

El 70 % de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

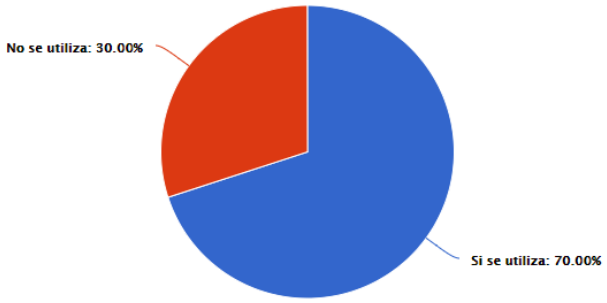

Figura 23. Documentar los requisitos de forma correcta. Elaboración propia.

Los participantes que afirman utilizar esta buena práctica aseguran que para ello utilizan formatos y validan con una segunda persona la redacción. Como resultado les proporciona un entendimiento común sobre los requisitos y a su vez tener una base para negociar con el cliente, y ya que no manejan reglas específicas, reconocen que no siempre se definen correctamente los requisitos, otros ven esta práctica como algo que debe realizar el analista o líder únicamente. Quienes no utilizan esta práctica se debe a que no documentan los requisitos y justifican que no es algo que se les exija y tampoco se asigna tiempo para realizarlo.

4.2.3.3 Definir los requisitos teniendo en cuenta la perspectiva del usuario

El 70 % de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza

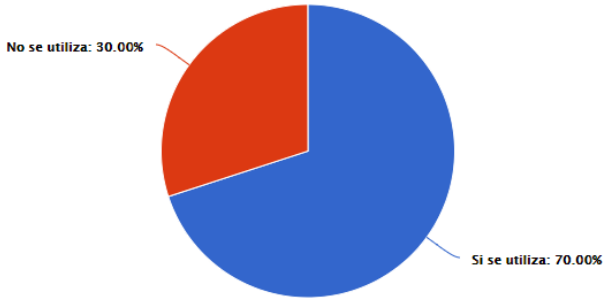

Figura 24. Definir los requisitos teniendo en cuenta la perspectiva del usuario. Elaboración propia.

Los que utilizan la práctica afirman que está condicionada a que el usuario tenga un buen manejo del negocio. Quienes no lo utilizan aducen que es porque sus usuarios no tienen definido lo que necesitan o como debería funcionar el software y es el desarrollador en algunos casos quien define el funcionamiento en base a su experiencia.

4.2.3.4 Adoptar plantillas de documentos de requisitos (visión y alcance, casos de uso, especificación de requisitos - SRS)

La mayoría de los participantes (70%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

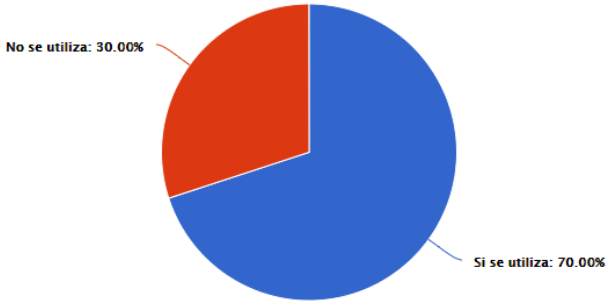

Figura 25. Adoptar plantillas de documentos de requisitos. Elaboración propia.

Quienes utilizan plantillas de documentos de requisitos lo encuentran beneficioso, ya que les ayuda en la búsqueda de información, cambios en los requisitos y el entendimiento de los requisitos, además les permite aplicar reutilización y mejora la comunicación para la definición del producto de software. Quienes no la utilizan aducen que se debe a la falta de procedimientos en el desarrollo de software.

4.2.3.5 Escribir especificación de requisitos no ambigua (cada requisito debe ser comprensible por sí solo)

La mayoría de los participantes (80%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 20% opina que no se utiliza.

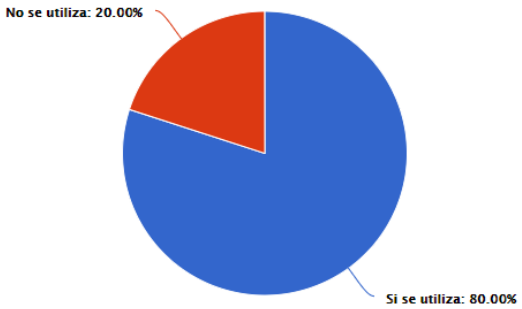

Figura 26. Escribir especificación de requisitos no ambigua. Elaboración propia.

Los participantes reconocen los beneficios de esta buena práctica, ya que, con una especificación no ambigua, todos comprenden lo mismo y tienen claras las funcionalidades esperadas en el software. Coinciden en que utilizar la técnica de solicitar a una segunda persona darles lectura a los requisitos para validarlos, ayuda a que sea claro y completo, adicional a eso comentan que mantener a la misma persona que levanta los requisitos en todo el proyecto ayuda a no producir ambigüedades, también hacen énfasis en que el conocimiento técnico y del negocio es algo indispensable para quien levanta los requisitos. Quienes no lo utilizan no proporcionaron más explicación que la de simplemente no tener documentadas las especificaciones de requisitos y por ende las ambigüedades son parte de su día a día.

4.2.4 Actividad de validación

4.2.4.1 Establecer para cada requisito criterios de validación y aceptación de las partes interesadas

La mayoría de los participantes (90%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 10% opina que no se utiliza.

Figura 27. Establecer criterios de validación y aceptación de las partes interesadas. Elaboración propia.

La mayoría de los participantes aseguran utilizar esta práctica y no iniciar el desarrollo sin contar con criterios de aceptación, sin embargo, consideran que no lo realizan de la mejor forma y en algunos casos dejan de hacerlo porque el encargado o jefe exige se realice el desarrollo en tiempo muy escaso o cuando tienen un requisito muy técnico, en este caso no lo validan con las partes interesadas. Por otro lado, quienes no lo utilizan argumentan se debe al no contar con la documentación que les permita validar los requisitos.

4.2.4.2 Validar la documentación de los requisitos funcionales y técnicos en su contenido, documentación y acuerdos

La mayoría de los participantes (70%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

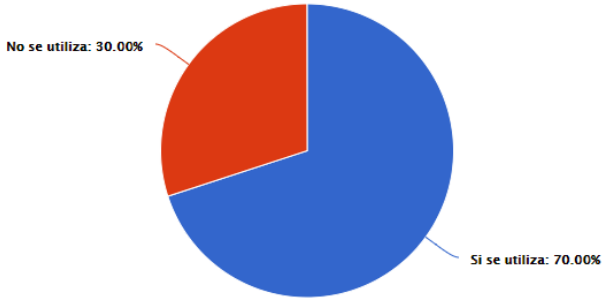

Figura 28. Validar la documentación de los requisitos funcionales y técnicos. Elaboración propia.

Los participantes afirman que esta buena práctica la realizan: el responsable de asegurar la calidad, los involucrados y jefes. Pero quienes no la utilizan lo atribuyen a mala comunicación y a que algunas personas se adueñan de los requisitos y documentación. Se notó en los participantes cierto desconocimiento, algunos afirmando utilizarla, más no indicaron el proceso, asumen que alguien más dentro de su empresa lo hace.

4.2.4.3 Analizar y validar los requisitos (trazabilidad, supuestos válidos, esenciales y consistentes con el diseño)

La mayoría de los participantes (70%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

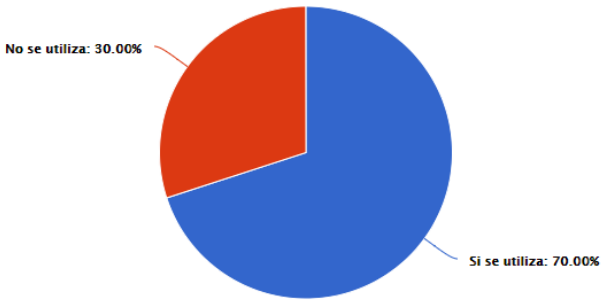

Figura 29. Analizar y validar los requisitos. Elaboración propia.

Quienes dicen utilizar la buena práctica, al cuestionarlos sobre la forma en que la realizan, la mayoría solo mencionaron algunas actividades, por ejemplo: la validación del diseño junto a su documentación. Quienes no utilizan la práctica, argumentan que sin una buena planificación y no contar con herramientas y/o metodologías, no es posible llevarla a cabo.

4.2.4.4 Revisar los requisitos y utilizar inspecciones de requisitos formales tanto con usuarios como con proveedores

La mayoría de los participantes (40%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 60% opina que no se utiliza

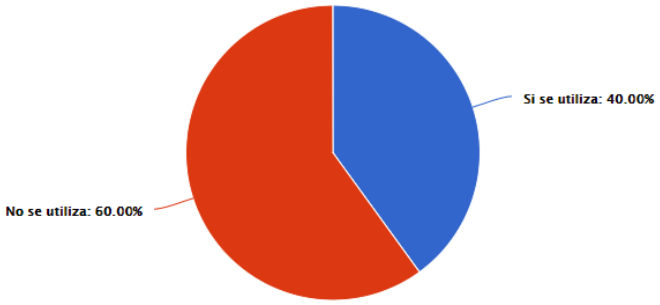

Figura 30. Revisar los requisitos y utilizar inspecciones de requisitos formales tanto con usuarios como con proveedores. Elaboración propia.

Quienes realizan esta práctica lo hacen asignando a una persona para que valide el requisito, distinto a quien lo ha documentado, y esto les sirve para identificar errores en la documentación de los requisitos. Quienes no lo utilizan consideran que es por omitir personas claves, limitándose a realizarla con un encargado o representante del negocio, esto por directrices de los líderes en la empresa. La mayoría concuerda que los requisitos no se revisan, solo se hacen pruebas cuando ya está finalizado el desarrollado, con método de prueba y error, lo que causa muchas veces retrabajo.

4.2.4.5 Utilización de prototipos para validar la interpretación de los requisitos de software (las características clave de las nuevas aplicaciones) y para obtener nuevos requisitos

La mayoría de los participantes (70%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

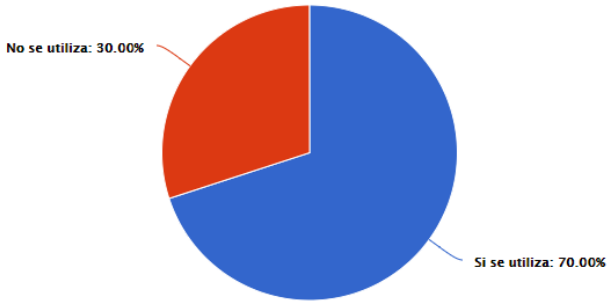

Figura 31. Utilización de prototipos para validar la interpretación de los requisitos de software y para obtener nuevos requisitos. Elaboración propia.

Los participantes lo asocian a la parte inicial de obtención del requisito y como un prototipo no funcional, como un bosquejo, un diseño básico de pantallas, etc. Les apoya para retroalimentar con los usuarios (incluyendo a todos los involucrados) para tener una idea clara del funcionamiento esperado que tendrá el software. Los que no lo utilizan consideran que no siempre es necesario que depende de la complejidad ya que esto genera tiempo y costos.

4.2.5 Actividad de gestión

4.2.5.1 Gestionar de forma efectiva y eficiente el control de cambios de los requisitos (introducción, cambios y eliminación), utilizando un plan y herramientas de configuración y control de cambios

La mayoría de los participantes (50%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 50% opina que no se utiliza.

Figura 32. Gestionar de forma efectiva y eficiente el control de cambios de los requisitos, utilizando un plan y herramientas de configuración y control de cambios. Elaboración propia.

La mitad de los participantes aseguran que utilizar esta buena práctica les ha permitido: minimizar riesgos, cumplir con temas legales, verificar y llevar un mejor control del desarrollo, lo que a su vez se traduce en eficiencia en la administración. Los que no la utilizan lo atribuyen al desconocimiento, ausencia de la documentación y al alto costo de la inversión que representaría para las empresas contar con un área especializada y herramientas.

4.2.5.2 Llevar el seguimiento de la vida de un requisito, utilizando la técnica de matriz de trazabilidad de requisitos.

El 30% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 70% opina que no se utiliza.

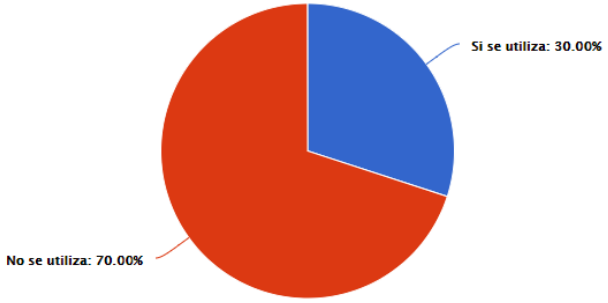

Figura 33. Llevar el seguimiento de la vida de un requisito, utilizando la técnica de matriz de trazabilidad de requisitos. Elaboración propia.

Algunos participantes afirman que utilizan la buena práctica y consideran que llevar el seguimiento de la vida de los requisitos por medio de la matriz de trazabilidad les es útil para el control de los proyectos, sin embargo, al consultarles en las razones de por qué no es utilizada

esta técnica, sus respuestas no corresponden con lo que es una matriz de trazabilidad de requisitos. Mientras los que no la utilizan argumentan que la causa principal es el desconocimiento y la falta de personal suficiente.

4.2.5.3 Establecer la línea base de los requisitos para asegurar que cualquier modificación en los requisitos que cambien la línea base se trate como cambios de alcance.

La mayoría de los participantes (70%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 30% opina que no se utiliza.

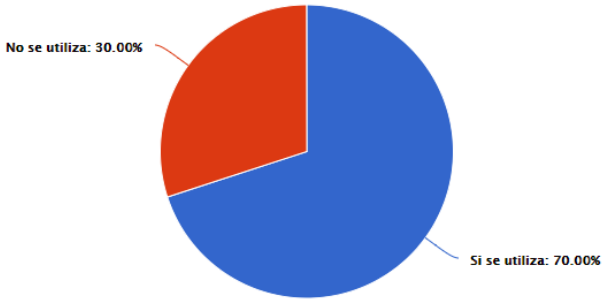

Figura 34. Establecer la línea base de los requisitos para asegurar que cualquier modificación en los requisitos que cambien la línea base se trate como cambios de alcance. Elaboración propia.

Los participantes coinciden en que esta práctica se establece y utiliza porque va relacionada a la negociación, que involucra costos y tiempos para el desarrollo del producto de software. El utilizar metodologías ha apoyado que se establezca la línea base de requisitos. Los participantes coinciden en que, si no se guían por una metodología, no establecen formalmente la línea base de requisitos, no controlan cambios y simplemente van modificando según surgen las necesidades. Los que no la utilizan responden que la empresa no maneja metodologías de desarrollo y que generalmente no documentan los requisitos por lo que no controlan cambios en el alcance.

4.2.5.4 Supervisar y dar seguimiento al estado de los requisitos de software (especificado, verificado, analizado, entre otros) mediante un proceso definido

La mayoría de los participantes (60%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 40% opina que no se utiliza.

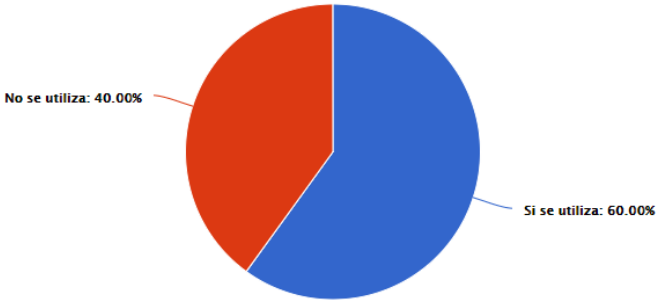

Figura 35. Supervisar y dar seguimiento al estado de los requisitos de software mediante un proceso definido. Elaboración propia.

Los participantes que afirman utilizar esta buena práctica comentan que no la realizan por medio de un proceso formal, en algunos casos se manejan por medio de correos electrónicos, en empresas pequeñas donde no hay un encargado de la planificación difícilmente se lleva control de los estados por medio de un proceso definido. En otro caso el estado no se maneja a nivel de cada requisito sino del proyecto que se esté realizando, esto les permite ir avanzando en las fases del desarrollo teniendo la autorización. Quienes no la utilizan esta buena práctica, conocen el estado de un requisito hasta que concluye el software o eventualmente cuando se supervisa el proyecto.

4.2.5.5 Medir el número y la gravedad de los defectos en los requisitos definidos

El 50% de los participantes respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 50% opina que no se utiliza.

Figura 36. Medir el número y la gravedad de los defectos en los requisitos definidos. Elaboración propia.

Los participantes que aseguran utilizarla comentan que es un indicador de que el requisito fue claro, algunos realizan una medición definiendo una métrica de defectos (Bugs) localizados durante las pruebas. Sin embargo, consideran que no se le da la importancia o seguimiento necesario. Quienes no la utilizan reconocen que es por ello por lo que repiten los mismos errores, otros justifican que el usuario no sabe lo que quiere o que los procesos del negocio no se encuentran definidos.

4.2.5.6 Formar a los analistas de requisitos para asegurar que tienen el conocimiento, entre otros aspectos, de cómo escribir y gestionar buenos requisitos

Los participantes (40%) respondieron que esta buena práctica si se utiliza en el desarrollo de aplicaciones Web en El Salvador, mientras que el 60% opina que no se utiliza.

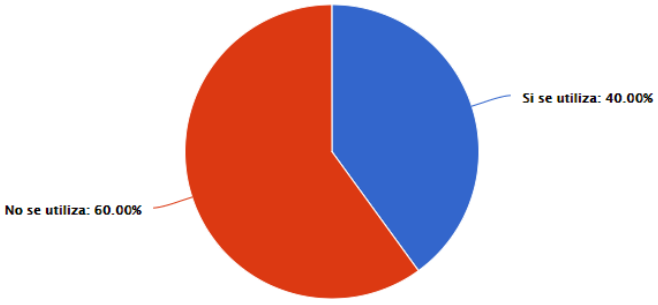

Figura 37. Formar a los analistas de requisitos para asegurar que tienen el conocimiento, entre otros aspectos, de cómo escribir y gestionar buenos requisitos. Elaboración propia.

Los participantes que dicen utilizar esta buena práctica consideran que es igual de importante la parte técnica, así como conocer sobre los procesos de negocio, lo que evita retrasos y reduce costos. La mayoría de participantes coinciden que en nuestro país las empresas no forman a los profesionales y que en muchos casos estos aprenden de las experiencias, ya que se cree erróneamente que el desarrollador es conocedor de cómo escribir y gestionar buenos requisitos. A su vez opinan que la mayoría de empresas no tienen una visión estratégica por el desconocimiento de quienes las dirigen, lo que produce que no se valore y no se le dé importancia a capacitar a los analistas de requisitos.

5. Conclusiones

El objetivo general de la investigación se alcanzó luego de construir un framework de buenas prácticas relacionadas con los requisitos de software de aplicaciones Web aplicables al contexto salvadoreño. Por medio del logro de los objetivos específicos siguientes:

Del proceso de 1) *identificar las buenas prácticas relacionadas con los requisitos de software por medio de la revisión bibliográfica,* se obtuvieron 11 fuentes documentales entre las que se encuentran: CMMI V1.3, SWEBOK V3, PMBOK V5, NASA-Handbook, entre otros.

Para así 2) *elaborar un framework propuesto de buenas prácticas relacionadas con los requisitos de software*, por medio de un banco de buenas prácticas que se sometió a un proceso de homologación y síntesis cuyo resultado fue un listado de buenas prácticas organizadas bajo cada actividad de requisitos de software. Finalmente se buscó 3) *evaluar el framework propuesto de buenas prácticas relacionadas con los requisitos de software, mediante el juicio de expertos, para establecer su utilización en el desarrollo de aplicaciones Web en pequeñas y medianas empresas de El Salvador,* cuyos resultados obtenidos permitió identificar elementos comunes y relevantes que impiden que estas buenas prácticas sean utilizadas.

Esta investigación abre un camino en investigaciones hechas en el país sobre los requisitos de software, al tiempo que puede servir como referencia a futuras posibles investigaciones, en torno a las técnicas y herramientas de gestión de requisitos que podrían ser utilizadas en nuestro contexto, las diferentes metodologías y su aporte en gestión de requisitos, la creación y el establecimiento de procesos para la gestión de requisitos, desarrollo de aplicaciones que faciliten la gestión de requisitos aplicando buenas prácticas, entre otras.

En el desarrollo de esta investigación se identificó que las buenas prácticas de requisitos de software, pueden ser utilizadas indistintamente tanto en el desarrollo de aplicaciones web como de escritorio.

A continuación se concluye cuáles son las actividades de requisitos de software que son más y menos utilizadas.

La actividad de obtención de requisitos, la más reconocida y utilizada.

Fue notorio en la bibliografía consultada el énfasis que se da a esta actividad. Un indicador sobre este fenómeno es que las buenas prácticas de obtención encontradas en fuentes bibliográficas, duplicaban a las identificadas en las demás actividades, encontrando mucho realce por los autores, dado que es el primer paso se considera crítico para el establecimiento de las siguientes actividades.

De los resultados de la investigación de campo y según lo expresado por los expertos sobre su experiencia utilizando las buenas prácticas de requisitos de software, podemos deducir que hay un mayor nivel de conciencia sobre la importancia de las buenas prácticas de la actividad de obtención, ya que fue donde mostraron mayor dominio de ellas.

La gestión de requisitos la actividad olvidada por las áreas de desarrollo de software.

En esta actividad los participantes mostraron mayor desconocimiento, siendo notorio que no tenían claro en qué consistían algunos procesos como el control de cambios y configuraciones, control de estados o técnicas como la matriz de trazabilidad de requisitos. Entre las causas expresadas por los expertos están: no especificar requisitos, la falta de capacitación en la empresa sobre cómo gestionar requisitos y el desconocimiento de las técnicas y herramientas existentes. Lo que ha permitido que las empresas consideren un costo el tiempo invertido en gestionar requisitos o contratar a una persona para esta actividad.

De los resultados de investigación se determinó, las buenas prácticas que son más y menos utilizadas en cada actividad de requisitos.

Obtención de requisitos de software

De las buenas prácticas en la actividad de obtención de requisitos las más utilizadas son:

1. Utilizar técnicas para obtención de requisitos entre las que se encuentran entrevistas, cuestionarios, observación, historia de usuarios, casos de uso, escenarios, prototipos, talleres de facilitación, análisis de documentos, análisis de tormentas de ideas, entre otros. Esta práctica es utilizada por todos los participantes.
2. Identificar, verificar e involucrar a las partes interesadas (stakeholders) y administrar los diferentes puntos de vista y,
3. Establecer y mantener acuerdos con los stakeholders (negociación y resolución de conflictos).

Las practicas menos utilizadas por los participantes son:

1. Definir y documentar claramente el alcance y visión del proyecto y,
2. No comunicar de manera efectiva ni utilizar modelos de comunicación.

Análisis de requisitos de software

La buena práctica más utilizada en la actividad de análisis de requisito de software es la priorización de requisitos. Las menos utilizadas son:

1. Desarrollar modelado conceptual (contexto).
2. Descomposición lógica de los requisitos de software (clasificación, relación y jerarquía de requisitos).

Especificación de requisitos de software

En la actividad de especificación de requisitos de software la buena práctica más utilizada es escribir especificaciones de requisitos no ambigua.

Las demás prácticas en buena medida son utilizadas por los participantes, incluyendo la creación de artefactos, documentar de forma correcta los requisitos, definición de los requisitos tomando en cuenta la perspectiva de los usuarios y la adopción de plantillas de requisitos.

Validación de requisitos de software

En la actividad de validación de requisitos de software la buena práctica más utilizada es la del establecimiento por cada requisito los criterios de validación y aceptación de las partes interesadas, y la menos utilizada es la de revisar los requisitos y utilizar inspecciones formales tanto con usuarios como con proveedores.

Gestión de requisitos de software

En la actividad de gestión de requisitos de software, la buena práctica más utilizada es establecer la línea base de los requisitos de software.

En la actividad de gestión de requisitos de software la mayoría de buenas prácticas no son utilizadas, entre ellas:

1. Llevar el seguimiento de la vida de un requisito, utilizando la técnica de matriz de trazabilidad de requisitos.
2. Formar a los analistas de requisitos para asegurar que tienen el conocimiento entre otros aspectos, de escribir y gestionar buenos requisitos.
3. Medir el número y la gravedad de los defectos en los requisitos de software definidos.
4. Gestionar de forma efectiva y eficiente el control de cambios de los requisitos de software.

5. Supervisar y dar seguimiento a los estados de los requisitos.

Al profundizar con los expertos sobre las razones de la no utilización de las buenas prácticas que les fueron presentadas, se identificaron las siguientes causas:

- Poco involucramiento de los usuarios.
- Omitir personas claves en la obtención y validación de requisitos.
- No contar con procesos de comunicación dentro de las empresas, ni personas que velen por que se cumplan.
- La comunicación no fluye a todos los interesados.
- Los usuarios o dueños de los procesos no expresan claramente el problema a solucionar
- No contar con una metodología de desarrollo.
- Desconocimiento de las buenas prácticas de gestión de requisitos.
- No utilizar herramientas para la gestión de requisitos.
- No utilizar precedentes de proyectos anteriores.
- No delegar a una persona como responsable del cumplimiento de buenas prácticas.
- La no identificación de estatutos regulatorios y estándares organizacionales.
- Los requisitos son impuestos por figuras autoritarias.
- Mala definición del alcance de los proyectos o desarrollos.
- Mala planificación que no establece tiempo para el análisis.
- Recurso humano limitado.
- Contar con documentación de requisitos incompleta o desactualizada
- Mal liderazgo de los responsables de áreas de desarrollo de software.
- No documentar los requisitos de software.

- Alto costo de la inversión para establecer un área especializada en requisitos de software.
- Los procesos de negocio no se encuentran definidos.
- Las empresas no tienen una visión.
- No capacitar al personal en temas de requisitos de software.

Finalmente, no nos queda más que recordar las palabas de F. F. Brooks “lo más complicado de construir un sistema software es decidir exactamente qué construir […] Ninguna otra parte del trabajo afecta tan negativamente al resultado si se hace mal. Ninguna es tan difícil de rectificar” (Sánchez, Sicilia, & Rodríguez, 2012, pág. 111).

6. Bibliografía

Almenara, C. J., & Llorente, C. M. (2013). La aplicación del juicio de experto como técnica de evaluación de las tecnologías de la información (TIC). *Revista de Tecnología de Información y Comunicación en Educación*, 11-22. Obtenido de http://tecnologiaedu.us.es/tecnoedu/images/stories/jca107.pdf

Baray, H. L. (2006). *Introducción a la metodología de la investigación.* (Edicion Electrónica. ed.). MEXICO: eumed.net. Recuperado el 6 de Diciembre de 2016, de www.eumed.net/libros/2006c/203/

Bernal Torres, C. A. (2010). Metodología de la investigación. En C. A. Torres, *Metodología de la investigación para administración, economia, umanidad y ciencias sociales.* Colombia: PEARSON EDUCACIÓN.

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, M. (2010). *Metodología de la investigación.* México D.F.: McGRAW-HILL.

Humphrey, W. (2009). The Software Quality Profile. Recuperado el 18 de Junio de 2017, de http://www.sei.cmu.edu/library/assets/softwarequalityprofile.pdf

IEEE. (1990). *Standard Glossary of Software Engineering Terminology.* New York. Recuperado el 6 de Enero de 2017, de http://www.mit.jyu.fi/ope/kurssit/TIES462/Materiaalit/IEEE_SoftwareEngGlossary.pdf

IEEE. (2014). *SWEBOK-Guide to the Software Engineering Version 3.0* (Segunda ed.). Nueva Jersey: IEEE Computer Society Products and Services.

ISACA. (10 de Octubre de 2017). *ISACA trust in, and value from, information system*. Obtenido de https://www.isaca.org/Pages/Glossary.aspx?tid=2015&char=G

Kirk Kandt, R. (2006). *Software Engineering Quality Practices* (Primera ed.). New York: Auebach Publication. Recuperado el 12 de Agosto de 2017

Lucio, T., Corona, M., & Debenedet, A. (2011). *Glosario y abreviaturas de ITIL*. España: AXELOS Limited. Obtenido de https://www.axelos.com/Corporate/media/Files/Glossaries/ITIL_2011_Glossary_ES-(Latin-America)-v1-0.pdf

Mason, M. (Septiembre de 2010). *Sample Size and Saturation in PhD Studies Using Qualitative Interviews*. Recuperado el 14 de Octubre de 2017, de Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 11(3), Art. 8: http://www.qualitative-research.net/index.php/fqs/article/view/1428/3027)

MINEC - DIGESTYC. (2012). *Directorio de unidades económicas 2011-2012.* Dirección general de estadisticas y censos, Ciudad Delgado. Recuperado el 20 de Marzo de 2017, de http://www.digestyc.gob.sv/index.php/novedades/avisos/aviso-empresa/264-directorio-de-unidades-economicas-2011-2012.html

Mora, S. L. (2002). *Programación de aplicaciones web: Historia, principios básicos y clientes web.* San Vicente, España: Editorial Club Universitario.

Pantaleo, G., & Rinaudo, L. (2014). *Ingenieria del Software* (Primera ed.). Argentina: Alfaomega.

PMI. (2013). *Guía de los fundamentos para dirección de proyectos* (Quinta ed.). Pensilvania, Estados Unidos: Project Management Institute Inc.

Pressman, R. (2010). *Ingenieria del software un enfoque práctico* (Séptima ed.). México, D.F: McGraw-Hill.

Sanchez Zuaín, S., & Duran, E. (Octubre de 2016). Taxonomía de Requisitos para Aplicaciones Web. *XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016).*, 626-635. Recuperado el 10 de Abril de 2017, de http://hdl.handle.net/10915/56725

Sánchez, S. A., Sicilia, M., & Rodríguez, D. (2012). *Ingeniería del software - un enfoque desde la guía SWEBOK* (Primera ed.). Madrid: Alfaomega, Garceta.

SEI. (2010). *CMMI for Development, Version 1.3.* Carnegie Mellon University. Recuperado el 23 de Abril de 2017, de https://resources.sei.cmu.edu/asset_files/TechnicalReport/2010_005_001_15287.pdf

SELA. (2005). Capítulo III .Acceso a las Tecnologías de la Información y la Comunicación (TICs) . En R. Monge Gonzales, C. Alfaro Azofeifa, & J. I. Alfaro Chamberlain, *TICs en las PYMES de Centroamérica: impacto de la adopción de las tecnologías de la información y la comunicación en el desempeño de las empresas* (pág. 270). International Development Research Centre. Recuperado el 5 de Abril de 2017, de http://www.sela.org/media/262053/t023600002508-0-tics_en_las_pymes_de_centroam%C3%A9rica_parte_i.pdf

Sommerville, I. (2011). *Ingeniería del software* (Novena ed.). México: Pearson educación.

Wiegers, K., & Beatty, J. (2013). *Software Requirements, 3rd Edition.* Redmond: Microsoft Press.

7. Anexos

7.1 Anexo No.1 – Cuestionario Guiado

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo _1_Cuestionario_guiado.pdf*

7.2 Anexo No. 2 - Matriz de Congruencia

Tabla 6
Matriz de congruencia.

#	Preguntas de investigación	Objetivo	Metodología	Técnicas	Instrumentos	Dominio	Variables

1	1) ¿Qué buenas prácticas relacionadas a los requisitos de software se identifican en la revisión bibliográfica?	1) Identificar las buenas prácticas relacionadas con los requisitos de software por medio de la revisión bibliográfica.	ETAPA 1.Revision bibliográfica
			• Revisión bibliográfica
			• Selección de fuentes bibliográficas.
			• Creación del banco de buenas prácticas de requisitos de software.
			  ○ Selección de buenas prácticas.
			  ○ Análisis y síntesis
			• Listado de buenas prácticas.
			• Framework propuesto.
				Revisión bibliográfica	Ficha de cotejo con criterios de selección de las fuentes.	• Obtención de requisitos.
						• Análisis de requisitos.
						• Especificación de requisitos.
						• Validación de requisitos.
						• Gestión de requisitos.
							Utilización de las buenas prácticas.

2	2) ¿Qué buenas prácticas de las identificadas en la revisión bibliográfica pueden ser consideradas para el framework propuesto y qué criterios de caracterización y selección de las buenas prácticas pueden utilizarse?	2) Elaborar un framework propuesto de buenas prácticas relacionadas con los requisitos de software por medio de criterios de selección que serán identificados y definidos durante la investigación.		Análisis documental Síntesis documental	Criterios de selección Herramienta de tabulación y cálculo		

3	3) ¿Qué buenas prácticas del framework propuesto consideran los expertos nacionales son utilizadas al desarrollo de aplicaciones Web en la pequeña y mediana empresa salvadoreña?	3) Evaluar el framework propuesto de buenas prácticas relacionadas con los requisitos de software, mediante el juicio de expertos, para establecer su utilización en el desarrollo de aplicaciones Web en pequeñas y medianas empresas de El Salvador.	ETAPA 2. Investigación de campo
			• Generalidades de la investigación de campo.
			• Caracterización de la muestra
			• Criterios de selección de la muestra.
			• Tipo de muestreo.
			• Tamaño de la muestra.
			• Instrumento de recolección de datos
			  ○ Cuestionario guiado
			  ○ análisis y organización de datos.
			• Procesamiento de la información.
				Juicio de expertos, análisis e interpretación documental y de resultados
				Triangulación entre métodos
					Criterios de selección de expertos.
					Cuestionario guiado, protocolo de entrevista, libreta de notas y grabadora.
					Herramienta de cálculo.		

Nota. Elaboración propia.

7.3 Anexo No. 3 – Selección de Fuentes Bibliográficas

Tabla 7.
Selección de fuentes bibliográficas.

Identificación documental	Identificador	Última versión publicada	Fecha publicación superior al año 2000	Alcance
CMMIDEV, Versión 1.3, SEI/2010.	CMMI	X	X	X
Guía de los Fundamentos para Dirección de Proyectos, Quinta edición, PMI/2012.	PMBOK	X	X	X
Requirements Engineering-Fundamentals, Principles, and Techniques, Klaus Pohl, Primera edición/2010.	Klaus Pohl	X	X	X
Agile Software Requeriments-LeanRequirements Practices For Teams, Programs, And The Enterprise, Dean Leffingwell, primera edición/2011.	Dean Leffingwell	X	X	X
Rational Unified Process: Overview, Rational Software Corporation/2001.	RUP	X	X	X
Guía Práctica de Gestión de Requisitos, INTECO/2008.	INTECO	X	X	X
Software Engineering Quality Practices, Ronald Kirk Kandt, primera edición/2006.	Ronald Kirk	X	X	X
Software Engineering Best Practices, Capers Jones, primera edición/2010.	Casper Jones	X	X	X
Software Requirements, Wiegers & Beatty, tercera edición/2013.	Wieger	X	X	X
System Engineering Handbook-NASA/SP-2016-610S Rev 2, NASA/2017.	NASA	X	X	X
SWEBOK, Version 3, IEEE Computer Society/2014.	SWEBOK	X	X	X
Modelo de Procesos para la Industria del Software, MoProsoft /2005.	MoProsoft	X	X	
ISO 9001:2015, Normas sobre calidad y gestión de calidad.	ISO 9000	X	X	
ISO/IEC 29110:2016, Ingeniería de sistemas y software - Perfiles de ciclo de vida para entidades muy pequeñas (VSE).	ISO29110	X	X	
Personal Software Process, PSP/1997.	PSP	X		
Team Software Process, TSP/2005.	TSP	X	X	
Ingeniería del software un enfoque práctico desde la guía SWEBOK - Salvador Sánchez, primera edición/2011.	Sánchez	X	X	
Ingeniería del software, Ian sommerville, novena edición/2011.	Ian	X	X	
Ingeniería del software un enfoque práctico, Roger Pressman, séptima edición/2010.	Pressman	X	X	

Nota. Elaboración propia. Solo se seleccionaron aquellas fuentes bibliográficas que cumplieron todos los criterios: última versión publicada con fecha superior al año 2000 y alcance. Para ampliar información sobre los criterios ir a la sección 2.8.1.2.

7.4 Anexo No. 4 – Selección de Buenas Prácticas de Fuentes Bibliograficas

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo _4 _Seleccion_de_buenas_practicas_de_fuentes_bibliograficas.xlsx*

7.5 Anexo No. 5 – Banco Buenas Prácticas

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo_5_Banco_buenas_practicas.xlsx*

7.6 Anexo No. 6 – Síntesis Buenas Prácticas

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo_6_Sintesis_buenas_practicas.xlsx*

7.7 Anexo No. 7 – Listado Buenas Prácticas

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo_7_Listado_buenas_practicas.xlsx*

7.8 Anexo No. 8 - Framework Propuesto

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo _8_Framework_propuesto.pdf*

7.9 Anexo No. 9 – Resultados del Cuestionario

Véase el documento en el CD, carpeta Tesina_ UDB _RSW, *Anexo_9_Resultados_del_cuestionario.pdf*