UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Diseños de Centro de Testeo para Desarrollo deAplicaciones Móviles

Trabajo de graduación presentado por Dennisse Emilia Escobar Delgado para optar al grado académico de Licenciada en Ingeniería en
Ciencias de la Computación y Tecnologías de la Información

Guatemala

2016

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Diseños de Centro de Testeo para Desarrollo deAplicaciones Móviles

Trabajo de graduación presentado por Dennisse Emilia Escobar Delgado para optar al grado académico de Licenciada en Ingeniería en
Ciencias de la Computación y Tecnologías de la Información

Guatemala

2016

Vo. Bo.

(f) _____

Msc. Mario Porras

Asesor

Tribunal examinador:

(f) _____

Msc. Douglas Barrios

(f) _____

Msc. Lynettte Garcia

(f) _____

Msc. Mario Porras

Fecha de aprobación del examen de graduación:
Guatemala, 08 de diciembre de 2016

PREFACIO

Agradezco primero a Dios, por la bendición y oportunidad de culminar mis estudios académicos de forma exitosa, a mi madre Catalina Delgado que sin ella nada de esto hubiera sido posible, mi motor, mi guía y mi fuerza en todo momento. Quien me dio las alas para volar, pero me recuerda que siempre tengo un nido a donde regresar.

A mis hermanos que me acompañaron en cada momento y todas las personas que han sembrado en mi para llegar hasta aquí.

A todos ellos solo me queda agradecimiento eterno**.**

INDICE

LISTA DE TABLAS

LISTA DE FIGURAS

RESUMEN

Este trabajo de investigación propone y describe el diseño de un centro de testeo para aplicaciones móviles en la Universidad del Valle de Guatemala. De esa cuenta, el estudio demuestra que existe una creciente demanda de servicios especializados en el aseguramiento de la calidad de software móvil, un área poco explorada en el contexto guatemalteco. La investigación desarrolla un modelo integral que abarca aspectos técnicos del testeo y consideraciones de negocio para establecer y operar un centro de testeo eficiente.

Este trabajo aspira a contribuir al desarrollo académico y profesional en el campo del aseguramiento de la calidad de software y a posicionar a la Universidad del Valle de Guatemala como un referente en la prestación de servicios de testeo de aplicaciones móviles. Se espera que la implementación del centro de testeo propuesto tenga un impacto significativo en la industria de desarrollo de software local, para mejorar la calidad de las aplicaciones y fortalecer la competitividad del sector.

Palabras clave:

Testeo, aplicaciones móviles, demanda, calidad, software, centro de testeo, desarrollo académico, competitividad

ABSTRACT

This research work proposes and describes the design of a testing center for mobile applications at the Universidad del Valle de Guatemala. From this account, the study demonstrates that there is a growing demand for specialized services in mobile software quality assurance, an area little explored in the Guatemalan context. The research develops a comprehensive model that encompasses technical aspects of testing and business considerations to establish and operate an efficient testing center.

This work aims to contribute to academic and professional development in the field of software quality assurance and to position the Universidad del Valle de Guatemala as a reference in the provision of mobile application testing services. The implementation of the proposed testing center is expected to have a significant impact on the local software development industry, to improve the quality of applications and strengthen the competitiveness of the sector.

Keywords

Testing, mobile applications, demand, quality, software, testing center, academic development, competitiveness

I. INTRODUCCIÓN

En la era digital, las aplicaciones móviles se han convertido en una parte esencial de la vida cotidiana y de los procesos empresariales. Con el rápido crecimiento de este mercado, la necesidad de garantizar la calidad y fiabilidad de estos productos de software es vital. Este trabajo de investigación propone y describe el diseño de un centro de testeo para aplicaciones móviles en la Universidad del Valle de Guatemala.

Este estudio demuestra que existe una creciente demanda de servicios especializados en el aseguramiento de la calidad de software móvil, un área poco explorada en el contexto guatemalteco. La investigación desarrolla un modelo integral que abarca aspectos técnicos del testeo y consideraciones de negocio para establecer y operar un centro de testeo eficiente.

Uno de los principales propósitos es definir y detallar cuáles son los diseños más utilizados para las pruebas funcionales de aplicaciones móviles y cómo han asegurado la calidad en los centros de testeo. Además, se busca crear una lista de procedimientos que permitan obtener un funcionamiento óptimo de dichos centros, la cual podría facilitar la implementación de un centro de testeo funcional en la Universidad del Valle de Guatemala.

La metodología se basa en una investigación exhaustiva de las prácticas actuales en el testeo de software móvil, lo que incluye el análisis de estándares internacionales, metodologías y modelos de negocio. Se consideran las especificidades del contexto local para asegurar la relevancia y aplicabilidad de las propuestas.

El estudio abarca aspectos críticos del testeo de aplicaciones móviles, incluyendo pruebas de funcionalidad, usabilidad, rendimiento y seguridad. Se presta atención a las particularidades de las diferentes plataformas móviles y a los desafíos específicos que presentan. Además, se exploran tendencias emergentes como la automatización de pruebas y el uso de inteligencia artificial en el proceso de aseguramiento de la calidad.

Este trabajo aspira a contribuir al desarrollo académico y profesional en el campo del aseguramiento de la calidad de software y a posicionar a la Universidad del Valle de Guatemala como un referente en la prestación de servicios de testeo de aplicaciones móviles. Se espera que la implementación del centro de testeo propuesto tenga un impacto significativo en la industria de desarrollo de software local, para mejorar la calidad de las aplicaciones y fortalecer la competitividad del sector.

Las siguientes secciones presentarán un análisis detallado de los resultados obtenidos. Se incluye el modelo de negocio propuesto; la metodología de testeo desarrollada; los sistemas de documentación y seguimiento; las estrategias de evaluación integral. Se explorará cómo estos elementos se integran para formar un centro de testeo coherente y efectivo.

El estudio también aborda las implicaciones económicas y estratégicas de establecer un centro de testeo de esta naturaleza y analiza el potencial impacto en la economía local, la creación de empleos especializados y el fortalecimiento de la posición de Guatemala en el mercado global de desarrollo de software.

Finalmente, se ofrecerán conclusiones y recomendaciones basadas en los hallazgos de la investigación, con el objetivo de guiar la implementación efectiva del centro de testeo propuesto. Estas recomendaciones se enfocarán en aspectos técnicos, operativos y estrategias para asegurar la sostenibilidad y relevancia continua del centro en el campo de la tecnología móvil.

Este proyecto se presenta como una iniciativa crucial en la búsqueda de la excelencia en el desarrollo de software móvil, al combinar una investigación rigurosa con la aplicación prácticade estándares internacionales. Se espera crear bases para un centro de testeo que cumpla con las expectativas actuales y esté preparado para enfrentar los desafíos futuros en el cambiante mundode la tecnología móvil.

II. OBJETIVOS

A. Objetivo general

Proponer el diseño de un centro de testeo para aplicaciones móviles en la Universidad delValle de Guatemala con estándares internacionales en aspectos técnicos, empresariales y éticos.

B. Objetivos específicos

1. Definir y crear diseños de testeo específicos para la certificación de productos de *software* móvil, con buenas prácticas y estándares internacionales.

2. Realizar una lista de procedimientos que se deben seguir para obtener un funcionamiento óptimo de testeo, que, a su vez, permita montar un centro de testeo funcional de móviles en la Universidad del Valle de Guatemala.

3. Diseñar un modelo de negocio detallado para el centro de testeo de la Universidad del Valle de Guatemala, incluyendo el procedimiento para asistir a los clientes en la realización de pruebas.

4. Identificar y proponer metodologías y herramientas actualizadas para el testeo de aplicaciones móviles, posicionando al centro propuesto a la vanguardia de las prácticas de aseguramiento de calidad en *software*.

III. JUSTIFICACIÓN

El desarrollo de un centro de testeo para aplicaciones móviles en la Universidad del Valle de Guatemala responde a una necesidad crítica en el mercado tecnológico actual. El continuo crecimiento de aplicaciones móviles ha generado una demanda creciente de servicios especializados en aseguramiento de calidad, un área aún poco explorada en el contexto guatemalteco.

El proyecto se enfoca en simplificar y optimizar los procesos de testeo, que son fundamentales para mejorar la eficiencia en la certificación de aplicaciones móviles. Se busca simplificar el estudio de patrones de *testing*, ciclos, flujos y pasos adecuados para cada tipo de programa móvil y empresa, y facilitar la comprensión y aplicación de buenas prácticas en el testeo de software.

La implementación de este centro permitiría la mejora de la calidad del software por medio de estándares de calidad en el desarrollo de aplicaciones móviles, y contribuirían a la creación de productos más fiables y eficientes. Con ello se fortalecería la industria local y se proporcionan servicios de testeo de alta calidad, se potenciaría la competitividad de las empresas guatemaltecas de desarrollo de *software* en el mercado global.

También, se incentivaría el desarrollo académico y profesional ya que este serviría como plataforma para la formación de profesionales especializados en testeo de *software*, con lo que se cubriría una brecha importante en el mercado laboral. La innovación y adaptación tecnológica en metodologías y herramientas de testeo mantendrían a la Universidad en la vanguardia en esta nueva rama tecnológica y a la industria local en un campo tecnológico en constante evolución. La implementación de este centro tiene el potencial de generar nuevos empleos especializados y atraer inversiones en el sector tecnológico, el cual impacta en la contribución económica del país.

Asimismo, establecería a la Universidad del Valle de Guatemala como un referente regional en el campo del testeo de aplicaciones móviles, con un posicionamiento estratégico, al asegurar la experiencia del usuario final no solo en la calidad de las

aplicaciones, sino en la seguridad, eficiencia y confiabilidad. Esta iniciativa no solo atiende una necesidad inmediata del mercado, sino que también sienta las bases para el crecimiento sostenido y la innovación en el sector de desarrollo de software móvil en Guatemala.

IV. MARCO TEÓRICO

A. Centro de testeo

Un centro de testeo, también conocido como *testing*, se refiere a una actividad que se desarrolla para evaluar la calidad de un producto, con lo que también ayuda en su mejora al identificar defectos y problemas. El *testing* de *software* consiste en la verificación dinámica del comportamiento de un programa sobre un conjunto finito de casos de prueba, que son seleccionados apropiadamente a partir del dominio de ejecución que usualmente es infinito con relación al comportamiento esperado (Myers, 2011).

Las pruebas representan la única herramienta que puede determinar la calidad de un producto *software*, en otras palabras, es el único método por el que se puede afirmar que un sistema cumple con los requerimientos que ofrece. Ello significa que existe una dependencia directa entre la calidad de un sistema y el valor de este, por lo que las pruebas dotan de valor a los productos *software*. A continuación, se describe el modelo de prueba de *software* (Cristiá, 2009).

La fase de pruebas es una actividad de suma importancia en el proceso de desarrollo de *software*. El propósito fundamental de las pruebas es descubrir errores y asegurar que el comportamiento del programa sea correcto en base a criterios específicos de las pruebas. Se considera que del 40% de los costos totales del desarrollo del *software* se consumen en dicha fase. Esta etapa tiene costos elevados tanto en recursos económicos como humanos, así como tiempo e intensidad de trabajo. No obstante, esa inversión generalmente no implica ninguna contribución en términos de funcionalidad. El escenario optimista indica que las pruebas del *software* deben garantizar la ausencia de fallas, pero en realidad sólo revelan su existencia sin ninguna garantía de su ausencia total (Cristiá, 2009).

B. ANTECEDENTES

La referencia a las pruebas de s*oftware* más antigua corresponde a 1950, pero fue en 1957 que la prueba fue distinguida del *debugging*. Dijkstra en 1970 presentaba una importante limitació, que "La Prueba de *Software* puede ser usada para mostrar la presencia de *bugs*, pero nunca su ausencia" (Amanat, 2004).

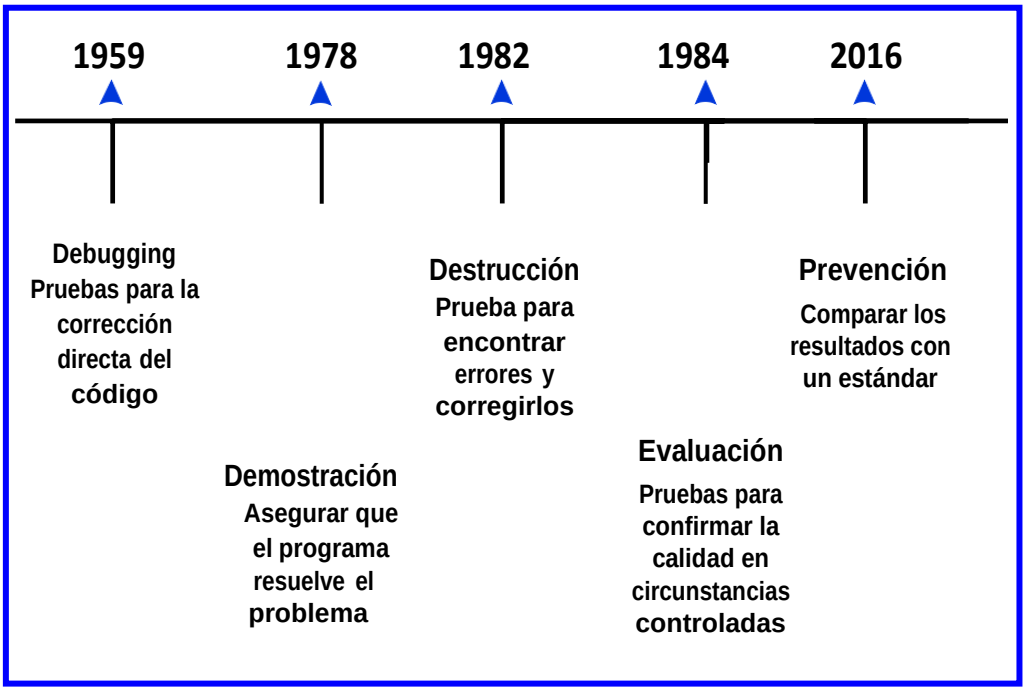
Myers en la segunda edición de su libro *"The art of Software testing"* de 2004, comenta que, en un lapso de tres décadas, la prueba de *software* no se ha convertido en una ciencia exacta,y que aún no se acerca a este nivel, lo que significa que se sabe mucho menos sobre la prueba de*software*, que sobre cualquier otro tema vinculado con el desarrollo de *software*. La prueba pertenece a las "artes oscuras" del desarrollo de *software* (Myers, 2011).

Las pruebas de *software* en el contexto histórico de la ingeniería del *software* pueden ubicarse en varias fases como se observa en la Figura 1, en donde se identifica su evolución (Myers, 2011):

Modelo I. Antes de 1956. Etapa orientada a *debugging*
Modelo II. 1957 - 1978. Etapa orientada a demostración
Modelo III. 1979 - 1982. Etapa orientada a destrucción
Modelo IV. 1983 - 1984. Etapa orientada a evaluación
Modelo V. 1985 actualidad. Etapa orientada a prevención

Figura 1.

Contexto histórico del testing



Nota: El gráfico describe la evolución del testeo de 1959 a 2016. *Fuente: elaboración propia (2016).*

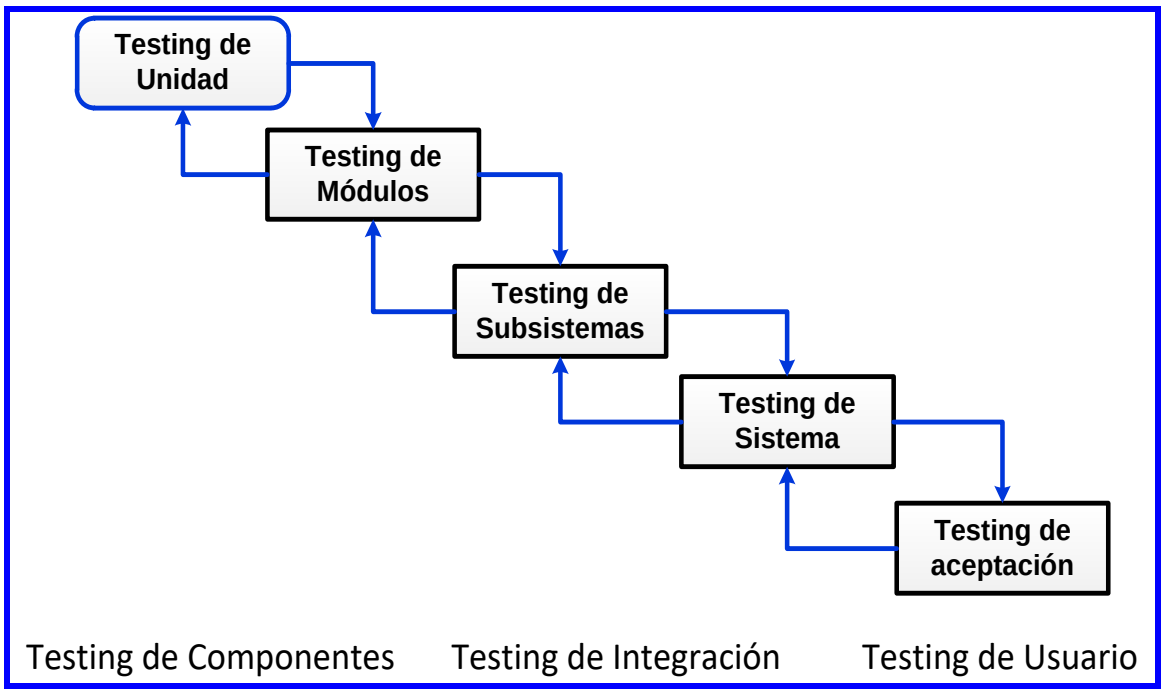
C. Proceso de *testing*

Es casi imposible, excepto para los programas más pequeños, testear un *software* como si fuera una entidad monolítica. Los grandes sistemas de *software* están compuestos de subsistemas, módulos y subrutinas. Se sugiere, por lo tanto, que el proceso de *testing* esté guiado por dicha estructura (Myers, 2011).

Cuando el proceso sugerido se combina adecuadamente con el proceso de desarrollo se habilita la posibilidad de detectar errores de implementación lo más temprano posible, lo que a su vez reduce el costo de desarrollo. El proceso general de *testing* sugerido se puede observar enla Figura 2 (Myers, 2011):

Figura 2.

Proceso general



Nota: El gráfico muestra el proceso general que debe seguirse de forma ordenada para detectar los posibles errores de implementación. Fuente: elaboración propia (2016)

Desde la perspectiva del *testing*, la definición de un proceso puede ser un procedimiento,una política o un estándar. Los modelos de ciclo de vida del *software* sirven como definiciones de alto nivel de las fases que ocurren durante el desarrollo. No tienen como objetivo brindar información detallada, sin embargo, sí proveer las principales actividades y sus interdependencias. Ejemplos de modelos de ciclos de vida son los en cascada, el de desarrollo evolutivo, desarrollo iterativo e incremental y modelo espiral (Amanat, 2004).

Los procesos pueden ser definidos en diferentes niveles de abstracción como actividades, productos (artefactos) y recursos. Estos se adaptan a las necesidades que específicas del contexto organizacional, el tamaño del proyecto, los requerimientos legales,

las prácticas de la industria y la cultura corporativa. Los modelos de procesos son importantes porque proporcionan el orden que debe seguir el proyecto para realizar sus tareas. Existen diferentes modelos de procesos paraelaborar un *software* (Amanat, 2004).

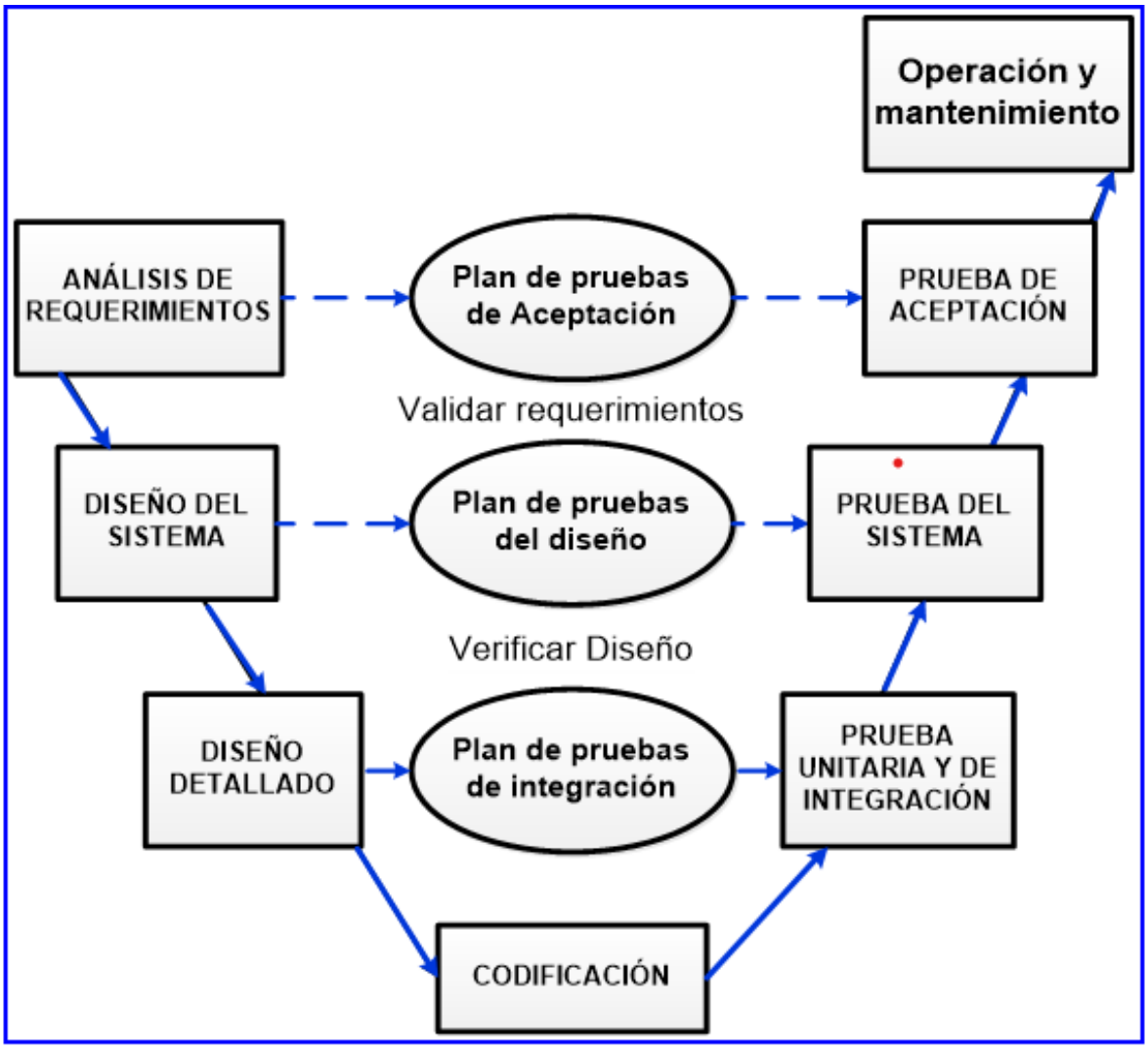
El modelo básico usado en los principios del *software* fue el *"Code & fix"* que tiene dos etapas: La primera consiste en escribir un código que resuelva un problema dado y la segunda, en arreglar los problemas del código. Se codifica y luego se piensa en los requerimientos, diseño, pruebas, mantenimiento. Este modelo no tiene en cuenta las necesidades del cliente y no responden en forma adecuada a cambios en los requerimientos (Amanat, 2004).

El modelo V fue concebido en 1992 como una serie de directivas que describen los procedimientos, los métodos que se aplicarán, y los requerimientos funcionales para las herramientas utilizadas en los sistemas de *software* que se desarrollen para las fuerzas armadas federales alemanas. Creando la idea que el *testing* puede comenzar en cualquier punto del proyecto no necesariamente hasta cuando el producto se encuentre finalizado (Polo, 2013).

El modelo V es una variación del modelo en cascada que demuestra cómo se relacionan las actividades de prueba con las de análisis y diseño. En la Figura 3 se puede ver que la punta de la V es la codificación, con el análisis y el diseño a la izquierda y la prueba y el mantenimientoa la derecha (Polo, 2013).

Figura 3.

Modelo V



Nota: En el gráfico puede observase que el centro de la V se refiere a la codificación, de la cual se desprende el análisis y el diseño del lado izquierdo y del lado derecho la prueba y el mantenimiento. Fuente. Elaboración propia (2016).

El modelo V sugiere que la prueba unitaria y de integración sea hecha también para verificar el diseño y constatar que todos los aspectos del diseño del programa se hayan implementado correctamente en el código. La prueba del sistema debe asegurar que todos los aspectos del diseño estén correctamente implementados (Polo, 2013).

La prueba de aceptación que es dirigida por el cliente, valida los requerimientos. La vinculación entre la parte izquierda y derecha del modelo V implica que, si se encuentran problemas durante la verificación y la validación, el lado izquierdo de la V puede ser ejecutado nuevamente para solucionar el problema y mejorar los requerimientos**,** el diseño y el código antesde retomar las pruebas del lado derecho (Polo, 2013).

D. Tipos de *testing*

En los últimos años, han surgido y evolucionado una variedad de métodos para realizar pruebas de *software*. Las alternativas más significativas en este contexto son las pruebas de Caja Blanca y las pruebas de Caja Negra. Las primeras, pruebas orientadas a la estructura y las segundas, al comportamiento del *software* (Pastor, 2012).

A continuación, se hace una descripción de algunas de las propuestas que plantean cada una de estas alternativas. En la Tabla No. 1 se puede observar las clasificaciones y tipos de *testing* que existen con material de las referencias. Los modelos de pruebas de *software* pueden clasificarse en modelos de pruebas funcionales y pruebas estructurales (Pastor, 2012).

Tabla 1.

Tipos de testing

Tipos	Clasificación
Técnicas de Caja Negra	Partición de Equivalencia Análisis del Valor Límite Tablas de decisión Máquinas de estado finito Grafo Causa Efecto Prueba de Casos de Uso Prueba de Dominios
Técnicas de Caja Blanca	Basadas en el Flujo de Control Basadas en el Flujo de los datos Mutantes

Continuación Tabla No. 1

Técnicas según quién hace la Prueba	Pruebas de aceptación Pruebas alfa y beta Pruebas de usuario Pruebas en pares
Técnicas Basadas en la experiencia	Prueba ad hoc Conjetura de errores *Testing* exploratorio

Nota: La tabla muestra se pueden observar los tipos y clasificaciones de *testing* existentes.
Fuente: Pastor (2012).

Se llama técnicas de Caja Negra o técnicas de prueba funcional a aquellas donde los casos de prueba se derivan a partir de la especificación del sistema. Estas técnicas se describen con mayor profundidad que las otras, ya que el proceso de prueba definido en este trabajo busca realizar una prueba funcional de productos de *software*. Las pruebas de Caja Negra ignoran el interior centrándose únicamente en las entradas y salidas del sistema (Pastor, 2012).

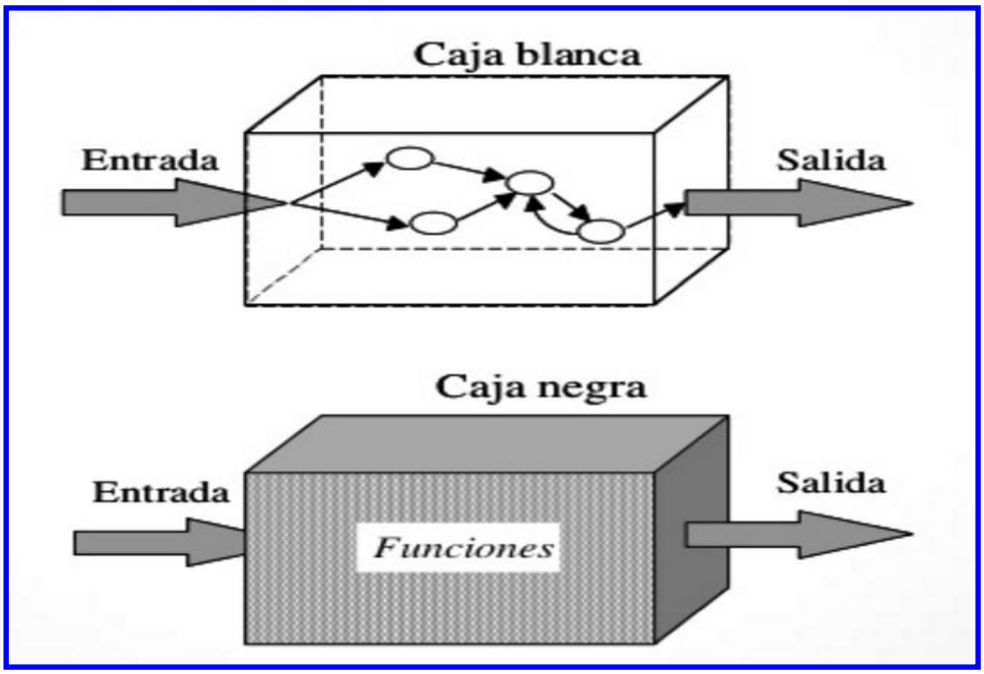
Las pruebas de Caja Blanca comprueban cómo se ejecuta un sistema, es decir, se enfocan en el mecanismo interno del programa al verificar paso a paso cada una de las acciones de este. Son aquellas en donde los casos de prueba se derivan a partir de la estructura del sistema y requieren conocer el código del programa a probar (Pastor, 2012); y centran su atención en los detalles procedimentales del *software.*

Por ello, la implementación de estas pruebas depende en gran medida de la disponibilidad de código fuente. Este tipo de pruebas permiten generar casos para ejercitar y validar los caminos de cada módulo, las condiciones lógicas, los bucles y sus límites, así como también para las estructuras de datos. Las pruebas de Caja Blanca también son conocidas como pruebas de caja de cristal o pruebas estructurales (Pastor, 2012). En la Figura No. 4 se ilustra cómo las pruebas de Caja Blanca comprueban lo que ocurre bajo el

capó, mientras que las pruebas de Caja Negra comprueban los resultados externos de la compilación del software.

Figura 4.

Pruebas de software más comunes



Nota: El gráfico demuestra que la principal diferencia entre las pruebas de caja negra y de caja blanca es lo que se está probando. Fuente: elaboración propia (2016).

Es importante resaltar que la prueba de *software* se realiza con el objetivo de hallar algo que difiera a las especificaciones planteadas para el producto o bien, para detectar la presencia de situaciones que pudieran generar resultados inapropiados (Pastor, 2012).

Aunque a grandes rasgos estas razones pueden orientar el sentido de una prueba, se presentan algunas normas a tomar en cuenta:

- La prueba es un proceso de ejecución de un programa con la intención de descubrir un error.
- Un buen caso de prueba es aquel que tiene alta probabilidad de mostrar un error no descubierto hasta entonces.

- Una prueba tiene éxito si se descubre un error.

Pruebas funcionales

La estrategia de *testing* se define como la forma en la que este se lleva a cabo el para un determinado producto, en donde se define de qué manera y cuándo se analizarán los requerimientos y funcionalidades del producto, y cuándo se diseñarán y se ejecutarán los casos de prueba. Las estrategias más comunes suelen ser las exploratorias, en las que se aprende mientras se ejecutan las pruebas; y el *testing* planificado donde se analiza primero el producto y luego se ejecuta (Pérez, 2011). Al verificar distintas estrategias, se puede tomar lo mejor de cada ella para crear una que se adecúe a las pruebas móviles ya que las existentes se aplicadas al diseño de *software* (Pérez, 2011).

***Testing* diseñado.** Es un conjunto de casos de prueba específicos, que pueden ser ejecutados por cualquier tester para crear validaciones de producciones intermedias, formado a partir de un diseño del requerimiento, el que sirve como base y apoyo en validación, priorización y negociación del alcance de las pruebas de forma objetiva. Este diseño tiene como base el análisis de requerimiento, diseño de casos de prueba, selección y priorización de casos de prueba y ejecución de pruebas (Pérez, 2011).

***Testing* automatizado.** Este tipo de *testing* se puede utilizar solo para ciertos casos con los que sea compatible. Al utilizarlo se debe decidir en cuáles se puede automatizar primero. Se debe tomar en cuenta que es preferible realizarlo con ejecución manual para tomar todos los casos existentes según costo de prueba, frecuencia de ejecución de forma automática y manual. En algunos casos la forma automatizada toma mucho más tiempo que una manual por la cual no es viable decidir automatizarla (Pérez, 2011).

Este tipo de prueba es útil cuando es muy complicado hacerla de manera manual, ya que la incertidumbre de errores de esta forma puede ser grande, por lo que hacerlo de manera automatizada disminuiría proporcionalmente el tiempo de ejecución en práctica. Cada tipo de prueba hace recomendable o no la automatización, ya que es necesario evaluar el proyecto, y revisar las ventajas y desventajas de cierto tipo de prueba según las

características y requisitos (Pérez, 2011).

Existen ciertas pruebas que no se puede realizar de forma automatizada como la funcionalidad, interfaz y presentación de datos. Por el contrario, el performance evalúa aspectos no funcionales del sistema y verifica las respuestas del sistema frente a diferentes situaciones sobre la infraestructura, plataforma y configuraciones, ya que el fin es simular el uso real del sistema para validar que el *software* y *hardware* se soporten entre sí. Otro tipo de pruebas que pueden ser automatizadas son el mantenimiento, portabilidad, entre otros (Pérez, 2011).

En resumen, las pruebas automatizadas pueden ser tomadas en cuenta cuando se realicen pruebas que no sean de interfaz, diseño, muestra o forma de presentar los datos, todo lo que sea cara al usuario. Para obtener los mejores beneficios de dichas pruebas se debe hacer una priorización de estas, de acuerdo con factores como cuáles las más fáciles de automatizar, las que retornen mayor inversión y las que se ejecutarán de manera más frecuente (Pérez, 2011).

Herramientas de automatización

Para elegir una herramienta para automatizar las pruebas es básico que estas aseguren el éxito, por lo que el proceso para la selección debe ser creado con base al fin de la organización, la que será usada de forma efectiva y con el menor costo representativo. No todas las herramientas son útiles para todas las organizaciones, ni existen herramientas que se adecuen a cada proyecto, por lo que debe ser seleccionada por tipo de operatividad (Ciolli, 2007).

Para encontrar la mejor herramienta se aconseja crear una lista de problemas a solucionar para saber cuál se adapta mejor a la organización como, por ejemplo: qué problemas se tiene en el *testing* manual; las pruebas de regresión; la configuración de datos de prueba, ya que son propensos a los errores, si no se cuenta con información sobre lo que se desea probar o si el *testing* manual no es efectivo. También se deben tomar en cuenta las restricciones de la organización con respecto a las herramientas como restricción de ambientes, es decir, si el hardware se encuentra disponible (sistemas operativos, base de

datos, entre otras); integración con el *software* a probar; restricción del proveedor; restricción de costos; políticas y sobre todo por restricción de calidad como fácil de uso; aprendizaje; cantidad de usuarios que soporta; frecuencia de fallas; confiabilidad. Y por último se debe hacer una nueva lista con las características que se desea tener como, por ejemplo: las características que se desean, las que debe contar, las que no interesan, las que cambian de categoría, como las nuevas versiones (Ciolli, 2007).

Las recomendaciones del mercado son de ayuda, pero esto no debe ser base para obtenerlas, ya que como se mencionaba anteriormente son útiles para el tipo de requerimiento. Además de tomar en cuenta el área técnica y el área económica, también se debe tomar considerar a las personas que la utilizarán y sus perspectivas para optar por la que se considere una curva de aprendizaje moderado. La herramienta no debe ser seleccionada cuando el *testing* ya se encuentraen proceso, pues el cronograma y el tiempo se verían afectados. Quien seleccione la herramienta debe contar con habilidades de gestión y conocimiento global del proyecto, pues es quien incitará a la incorporación de la automatización en el testeo (Ciolli, 2007).

Existen diferentes y variados tipos de herramientas las cuales se pueden derivar en:

- Herramientas de diseño
- Herramientas de lógica
- Herramientas de diseño físico
- Herramientas de gestión
- Herramientas de análisis
- Herramientas de análisis de código
- Herramientas de simulación
- Herramientas de performance
- Herramientas de carga
- Herramientas de ejecución y comparación.

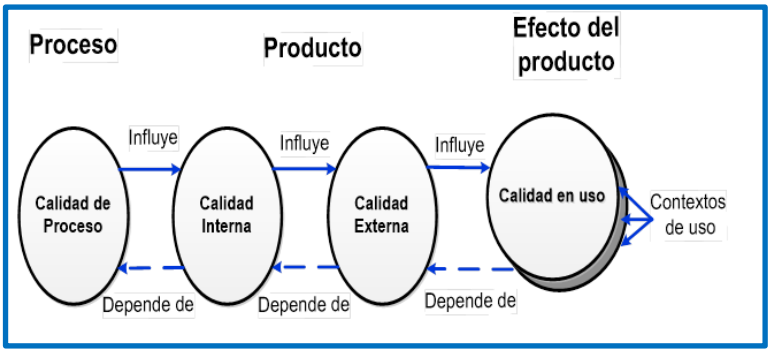
E. Estándares de calidad

El modelo de calidad ISO/IEC 9126 describe un modelo de calidad para un producto de *software*: a) calidad interna y calidad externa, y b) calidad en el uso. La primera parte del modelo especifica seis características para la calidad interna y externa. La segunda parte del modelo especifica cuatro características de calidad en el uso, que es el efecto combinado para el usuario de las seis características de la calidad del producto de *software*. La calidad de proceso contribuye a mejorar la calidad del producto, y la calidad del producto contribuye a mejorar la calidad en el uso. Por lo tanto, evaluar y mejorar el proceso es una forma de mejorar la calidad del producto, y evaluar y mejorar la calidad del producto es una forma de mejorar calidad en uso. De la misma manera, evaluar la calidad en el uso ayuda a mejorar el producto, y la evaluación del producto ayuda a mejorar el proceso (Senlle, 2001).

Cada característica relevante de la calidad del producto de *software* debe ser especificada y evaluada, siempre que sea posible, usando una métrica validada o extensamente aceptada. Se definen tres tipos de métricas que son dependientes e influyentes entre ellas, como se observa en la Figura 5 (Senlle, 2001):

Figura 5.

Calidad en el ciclo de vida



Nota: El gráfico muestra tres clases de métricas que influyen y dependen una de o tras.

Fuente: elaboración propia (2016)

Métricas internas: Se aplican a un producto de *software* no ejecutable (tal como código de la especificación) en el diseño y la codificación, con el propósito primario de asegurar que la calidad externa requerida y la calidad en uso son alcanzadas (Senlle, 2001).

Métricas externas: Son las que utilizan mediciones de un producto de *software* derivadas de las mediciones del comportamiento del sistema del cual es parte, mediante la prueba, operación y observación del *software* o sistema ejecutado (Senlle, 2001).

Métricas de calidad en el uso: Miden el grado en el que un producto resuelve las necesidades de usuarios para alcanzar los objetivos especificados con eficacia, productividad, seguridad y satisfacción en un contexto especificado de uso (Senlle, 2001).

El modelo para la calidad externa e interna categoriza las cualidades del *software* en seis características: funcionalidad, confiabilidad, utilidad, eficacia, capacidad de mantenimiento y portabilidad, que se subdividen en sub características. Estas sub características se manifiestan externamente cuando el *software* se utiliza como parte de un sistema informático, y son el resultado de las cualidades internas del *software*. A continuación, se describe cada una de esas características (Senlle, 2001).

Funcionalidad: Capacidad del producto de *software* para proporcionar las funciones que resuelven las necesidades indicadas e implicadas cuando se utiliza bajo condiciones específicas y acorde con los requerimientos. Como ejemplo de procesos funcionales se encuentra ProTest y *Testing* Ágil (Senlle, 2001).

ProTest: es un proceso de *testing* funcional independiente, orientado a ciclos de prueba y basado en análisis de riesgo del producto. Es decir, se puede aplicar y adaptar este proceso al del de desarrollo que siga cualquier organización y aporta una visión objetiva y tiene la flexibilidad de incorporarse a cualquier altura del proyecto de desarrollo. Este proceso cuenta con cuatro etapas principales: estudio preliminar, planificación, ciclos de prueba y evaluación. El ciclo de prueba se compone de seguimiento, configuración del

entorno, diseño de pruebas, ejecución de las pruebas (Senlle, 2001).

La base principal de este proceso es el ciclo de prueba, el cual se realiza por versión sin modificaciones. Es decir, se verifica el producto en una versión sin sufrir alteraciones durante la ejecución de pruebas y posteriormente se arreglan los incidentes detectados en un ambiente separado, y en una nueva versión con los arreglos y funcionalidades se es liberada para un nuevo ciclo de prueba (Senlle, 2001).

Finalmente, para el manejo de análisis de riesgo, se elabora una lista de riesgo priorizada, se mitiga cada uno de ellos, con una revisión constante que incluye la incorporación de nuevos riesgos. Se considera un riesgo, si representa una amenaza para el producto, organización, negocio, los interesados o compromete el desarrollo del proyecto. Esta lista debe ser elaborada eidentificada por los encargados del *testing*, pero debe ser verificada por el resto de los actores delproyecto (Senlle, 2001).

***Testing* ágil:** es una práctica de *testing* que surge de la necesidad de las organizaciones de adaptar varios métodos de testeo como complemento de las metodologías ágiles. Ya que existen varias limitaciones y obstáculos al momento de incorporar el *testing* en el ciclo de vida de los métodos agiles de desarrollo pues casi siempre existe mala comunicación entre los equipos, carencia de automatización de pruebas en las fases del ciclo de vida del desarrollo.

Esta práctica tiene como prioridad satisfacer al interesado con entregas tempranas del *software*, también promueve la colaboración del equipo de forma y mejora continuas de los procesos para responder al cambio según las expectativas del interesado (Servat, 2005). Las organizaciones que suelen implementar este tipo de metodologías son aquellas que suelen tener integrantes con pensamiento independiente, sin estructura jerárquica y que tiene énfasis en políticas de calidad.

Las empresas que no se encuentran preparadas para este tipo de metodología no logran afrontar los nuevos retos y cambios. El *testing* ágil es una de las metodologías que intenta crear un ambiente colaborativo con interacción intrapersonal, ya que apunta hacia la autogestión y la integración de todos los usuarios que se encuentren en el proyecto, y crea responsabilidad por la calidad del equipo en lugar de crear pertenencia exclusiva del *testing* (Servat, 2005).

Tanto como otras metodologías, el *testing* ágil tiene entre sus prácticas recomendadas las pruebas por pares, pruebas de regresión automatizadas e incluso pruebas de aceptación automatizadas, y la utilización de herramientas adecuadas como un sistema de reporte y seguimiento de incidentes (Servat, 2005).

Esto es fundamental para el éxito del *testing* ágil, ya que el ritmo de desarrollo supera la capacidad de pruebas manuales, por lo que debe ser una estrategia enfocada en la automatización. También, se recomienda como buena práctica, considerar la importancia del rol del usuario o cliente en la elaboración y ejecución de pruebas de aceptación del producto (Servat, 2005).

Confiabilidad: capacidad del producto de *software* para mantener un nivel de desempeño cuando es utilizado bajo las condiciones especificadas.

Usabilidad: capacidad del producto de *software* de ser entendido, aprendido, utilizado y atractivo al usuario, cuando es utilizado bajo condiciones especificadas.

Eficiencia: capacidad del producto de *software* para proporcionar el desempeño apropiado, concerniente a la cantidad de recursos usados, bajo condiciones indicadas.

Mantenibilidad: capacidad del producto de *software* de ser modificado. Las modificaciones pueden incluir correcciones, mejoras o la adaptación del *software* a los cambios en el ambiente, los requerimientos y las especificaciones.

Portabilidad: capacidad del producto de *software* de ser transferido de un ambiente a otro.

El Modelo de Calidad en el uso tiene cuatro características: eficacia, productividad, seguridad física y satisfacción. Estas se describen a continuación:

Eficacia: capacidad del producto de *software* para permitir a los usuarios alcanzar los objetivos con exactitud y completitud, en un contexto especificado de uso.

Productividad: capacidad del producto de *software* para permitir a los usuarios utilizar cantidades apropiadas de recursos en lo referente a la eficacia alcanzada, en un contextoespecificado de uso.

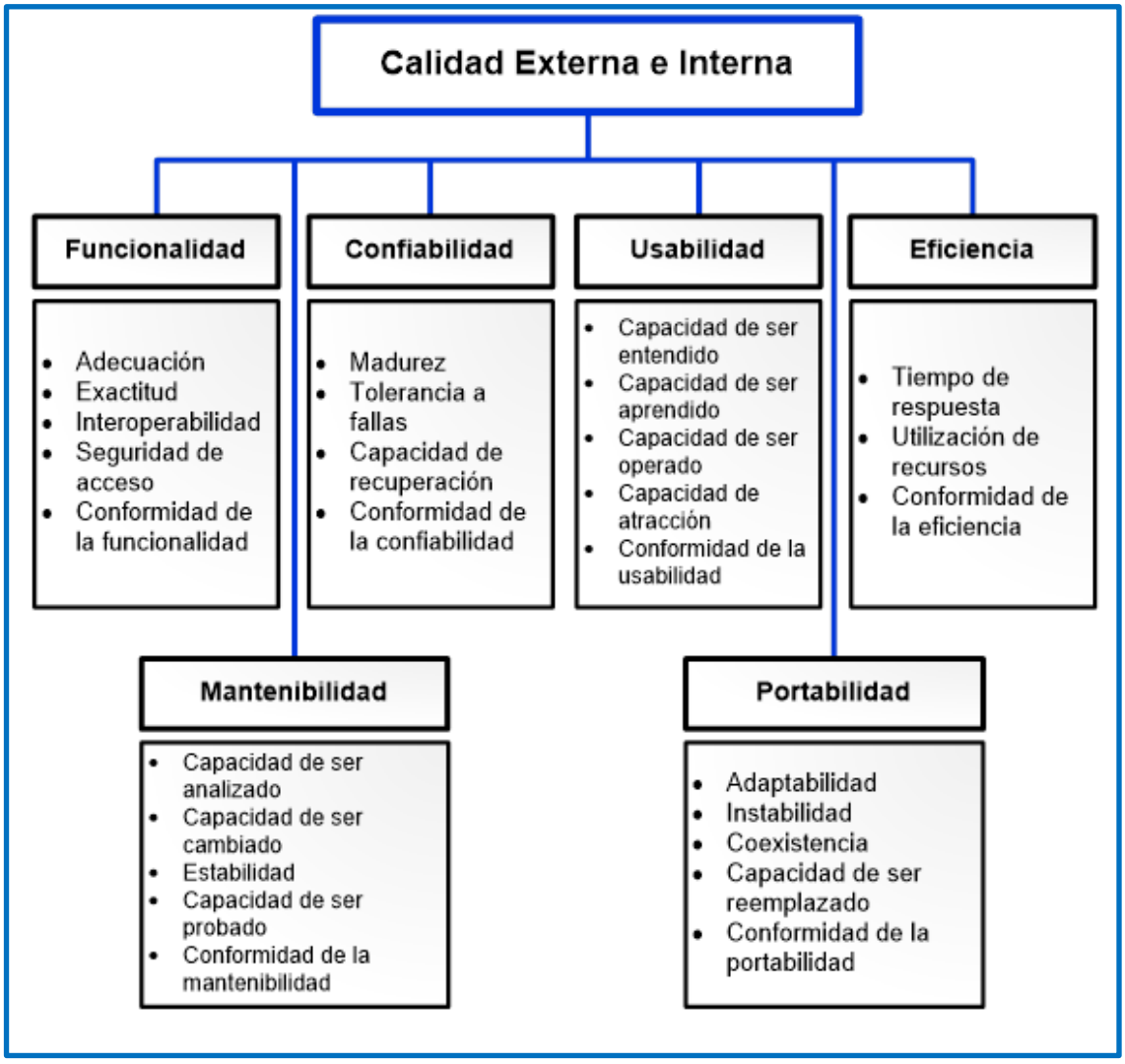
Seguridad física: capacidad del producto de *software* para alcanzar niveles aceptables del riesgo de dañar personas, al negocio, al software o al ambiente, en un contexto especificado de uso.

Satisfacción: capacidad del producto de *software* de satisfacer a los usuarios, en un contexto especificado de uso.

La calidad en uso es la visión del usuario de la calidad, es la capacidad del producto de *software* de permitir a los usuarios alcanzar los objetivos especificados con eficacia, productividad, seguridad y satisfacción en contextos especificados de uso, tal como se describe en la Figura 6 (Servat, 2005):

Figura 6.

Modelo de calidad externo e interno



Nota: El gráfico muestra como la calidad interna y externa se enfocan en los procesos y estándares dentro de la organización y en la satisfacción y percepción del cliente o usuario final. Fuente: Servat (2005)

Certificación de calidad de software

En un mercado globalizado donde las empresas deben innovar y mejorar continuamente para crecer y ser más competitivas es necesario tener acceso a certificaciones de calidad internacionales que les den un respaldo y puedan mantenerse en este mercado. Las certificaciones de calidad en la industria del *software* ayudan a las empresas a ser más productivas al disminuir costos y tiempo en sus desarrollos. El carácter de sin-fronteras de la industria del *software* ha despertado el interés de países latinoamericanos y europeos por impulsar programas de apoyo para el mejoramiento de la calidad en la industria del software. Con el objetivo de lograr mejoras en el desarrollo de *software* y en sus procesos a nivel mundial se han desarrollado modelos de calidad que les permiten a las empresas certificarse y obtener mejores resultados sus productos yen su gestión administrativa y gerencial (Campos, 2005):

La calidad del *software* se refiere a la concordancia con los requerimientos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo documentados y con las características implícitas que se esperan de todo *software* desarrollado profesionalmente. La Calidad del *Software* es una disciplina más dentro de la Ingeniería del *Software*. El principal instrumento para garantizar la calidad de las aplicaciones sigue siendo el Plan de Calidad, el cual se basa en normas o estándares genéricos y en procedimientos particulares. Los procedimientos pueden variar en cada organización, pero lo importante es que estén escritos, personalizados, adaptados a los procesos de la organización y que se sean cumplidos (Campos, 2005).

En este orden de ideas, se puede decir que los requisitos del *software* son la base de las medidas de calidad y que la falta de concordancia con los requisitos es una falta de calidad. Los estándares o metodologías definen un conjunto de criterios de desarrollo que guían la forma en que se aplica la Ingeniería del *Software*. Si no se sigue ninguna metodología siempre habrá falta de calidad. Todas las metodologías y herramientas tienen un único fin producir *software* de alta calidad (Campos, 2005).

F. ISO 9000

El ISO 9000 es una norma internacional de aplicación voluntaria, que establece los requisitos que debe cumplir una empresa para demostrar que tiene la capacidad de cumplir las demandas de sus clientes, que tiene un enfoque proactivo enfocado hacia las causas de falla y control de riesgos y que mejora continua en su desempeño (Campos, 2005).

ISO es acrónimo de la Organización Internacional de Normalización (*Internacional Organization for Standarization*). Se fundó en 1946 con el fin de crear un conjunto común de normas para la manufactura, comercio y comunicaciones. Según los funcionarios de la ISO la organización tomó prestadas las siglas de la palabra griega "isos", que significa igual. Porotra parte, isos es también la raíz del prefijo iso, como en la palabra isométrico (de igual medida o dimensión) y de isonomía (igualdad de leyes o de la gente ante estas). La elección se basó en la ruta conceptual que lleva la palabra "igual","uniforme", "norma" (Campos, 2005).

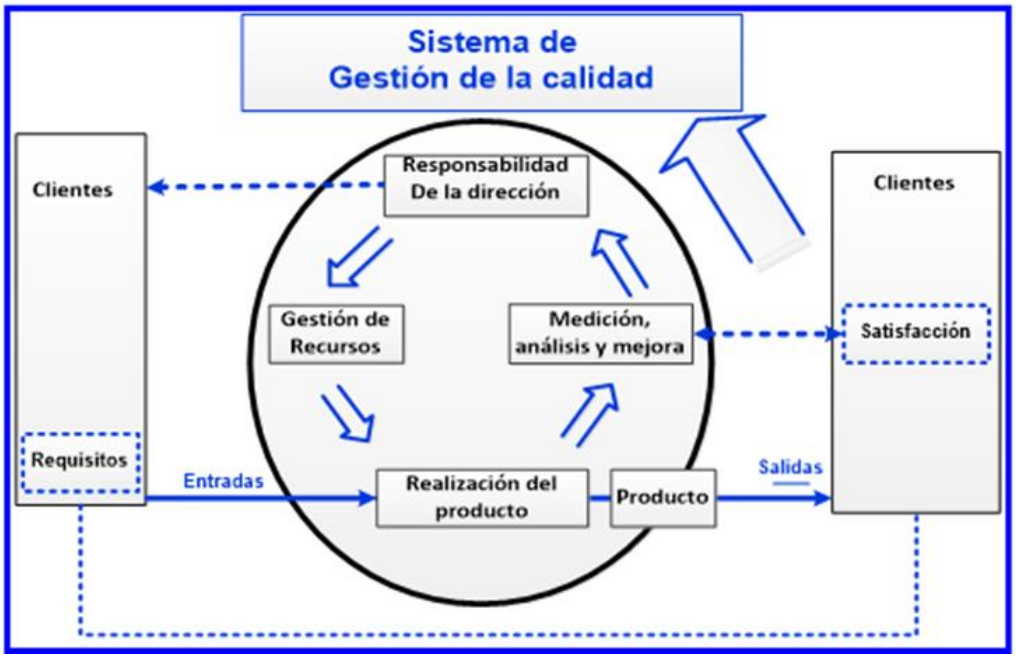
La organización, cuya matriz se encuentra en Ginebra, Suiza, está conformada por más de 130 países. En ISO, cada país está representado por su organismo integrante, la organización nacional que coordina las normas nacionales. Cada organismo tiene derecho a un solo voto sin importar el tamaño del país. En Guatemala el organismo que coordina las normas es la Comisión Guatemalteca de Normas (COGUANOR), que depende del Ministerio de Economía (Campos, 2005).

Todas las normas establecidas por la ISO son voluntarias; no existen requisitos legales para que los países puedan adoptarlas. No obstante, los países y las industrias suelen adoptarlas como normas nacionales. En algunos casos, hay países que le suman requisitos legales a las que han adoptado y, de esta manera, se convierten en obligatorias en esos países. La ISO establece normas para todas las industrias, con excepción de aquellas relacionadas con la ingeniería eléctrica y electrónica. En la Figura 7 se puede observar la gestión de la norma ISO 9000. Las normas de estas áreas corresponden a la Comisión Internacional Electrotécnica IEC *(International Electro Technical Comision*), con sede

también en Ginebra. En la práctica, la ISO y la IEC cooperan muy cerca en sus actividades y publican en conjunto común las directrices que rigen la elaboración de las normas (Campos, 2005).

Figura 7.

Gestión de la calidad



Nota: El gráfico muestra el proceso de gestión de la norma ISO 900 con todos sus componentes, la cual busca la satisfacción del cliente. Fuente: Campos (2005)

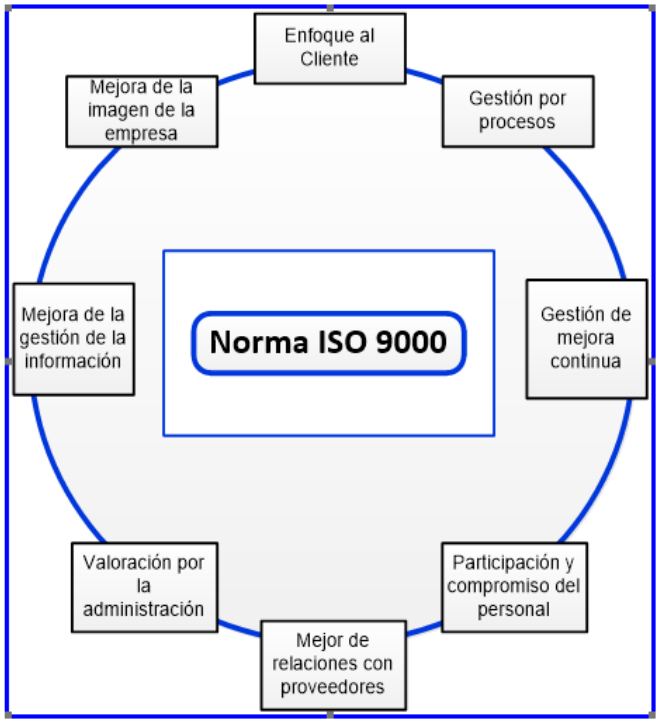
La estructura de la ISO está formada por más de 185 comités técnicos que elaboran el borrador de las normas. Las naciones integrantes constituyen comités técnicos que establecen la postura y las estrategias de negociación del país y seleccionan delegados que aporten sus conocimientos al proceso de elaboración de las normas. Ello permite que la ISO reciba muchas aportaciones y establezca consenso, antes de promulgar una norma, entre la industria, el gobierno y demás entidades interesadas (Campos, 2005).

Familia de las normas ISO 9000. Las normas ISO son un grupo (familia) de normas para la gestión de la calidad en organizaciones productivas y de servicio. Promueve la consistente uniformidad en la producción y servicios, además de confiabilidad (Toledo,

2007). El ISO 9000 se refiere a la capacidad del fabricante o prestador de servicio para producirlos de forma ordenada y confiable, según las necesidades y expectativas del comprador. Por eso no hay "productos ISO 9000" ni está permitido su uso en la certificación en artículos destinados al consumidor final (Toledo, 2007).

Norma ISO 9000. *Sistema de gestión de la calidad-fundamentos y vocabulario:* Describe los fundamentos de los Sistemas de Gestión de la Calidad y específica la terminología de los Sistemas de Gestión de la Calidad. La familia de normas ISO 9000 al ser diseñada para ser consultada por muchos usuarios de diferentes países requiere de un trabajo exhaustivo, sobre todo en los términos y definiciones, ya que el término de una palabra varía de país en país. Para el buen entendimiento de la familia de normas ISO 9000 es necesario entender los términos y definiciones descritas para evitar confusiones o malentendidos. Estas definiciones tienen carácter de obligatoriedad para todos los usuarios de las normas ISO 9000, en especial aquellas organizaciones que buscan la Certificación de un Sistema de Gestión de la Calidad, como se muestra en la Figura 8 (Toledo, 2007)**:**

Figura 8. *Gestión norma ISO 9000*



Nota: El gráfico muestra cada uno de los términos y definiciones que utiliza la Norma ISO

900. Fuente: elaboración propia (2016).

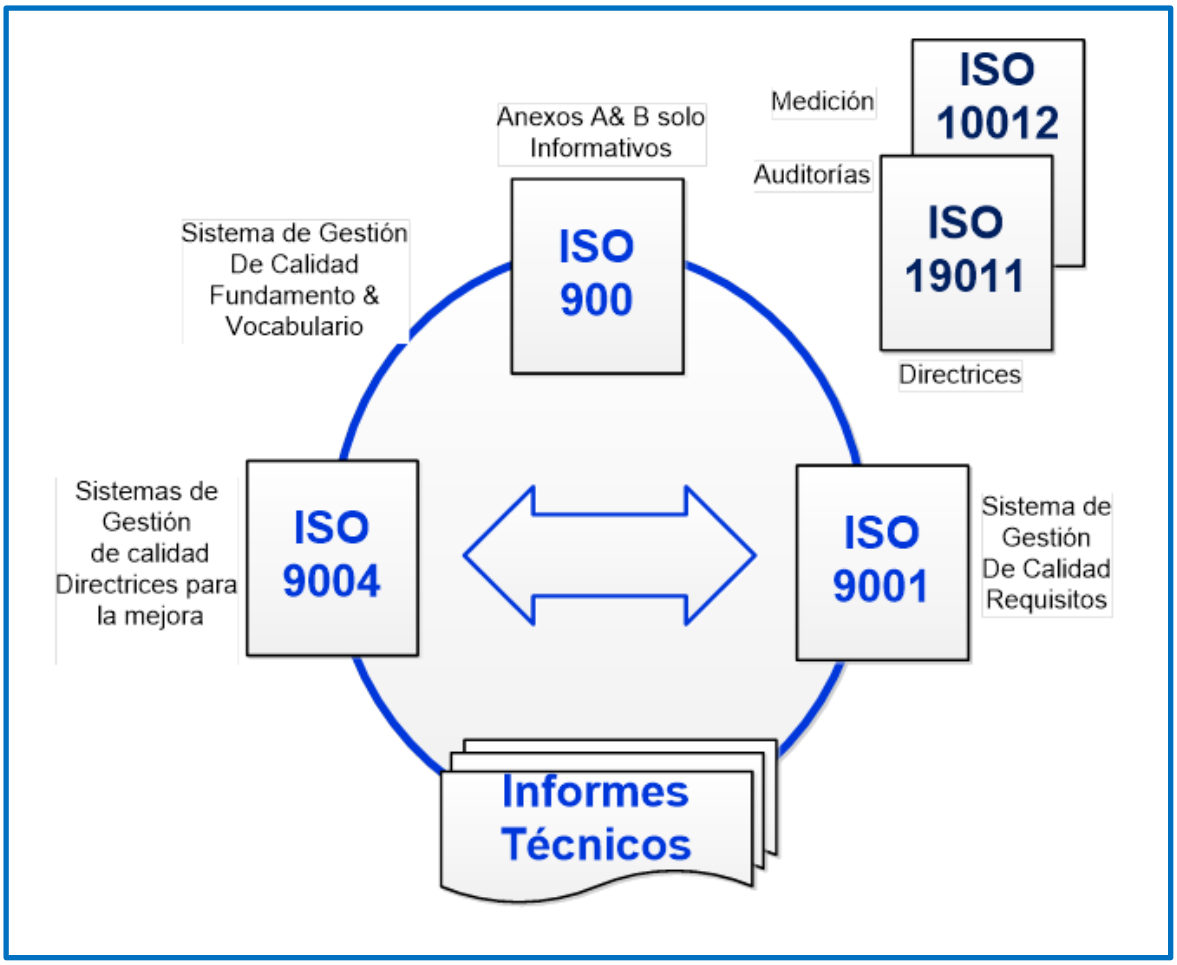
Norma ISO 9001 –Sistema de gestión de la calidad - requisitos: Especifica los requisitos para los Sistemas de Gestión de la Calidad aplicables a toda organización que necesitedemostrar su capacidad para proporcionar productos que cumplan los requisitos de sus clientes y los reglamentos que le sean de aplicación y su objetivo es la consecución de la satisfacción de sus demandas. Es el único estándar de la familia de ISO 9000 contra el cual se puede realizar una auditoría de Certificación (Toledo, 2007).

Norma ISO 9004 –Sistema de gestión de la calidad - directrices para la mejora del desempeño: Proporciona directrices que consideran tanto la eficiencia como la efectividad del sistema de gestión de la calidad. El objetivo de esta norma es la mejora del desempeño de la organización y la satisfacción de los clientes y de las partes interesadas (Toledo, 2007).

Esta norma se recomienda como una guía para aquellas organizaciones cuya alta dirección quiera ir más allá de los requisitos de la norma ISO 9001, por lo que persigue la mejora continua del desempeño. Esta norma no tiene como finalidad la uniformidad de los Sistemas de Gestión de la Calidad ni la documentación de este, su finalidad es identificar y satisfacer las necesidades y expectativas de sus clientes y otras partes interesadas para lograr ventaja competitiva y para hacerlo de una manera eficaz y eficiente. Obtener, mantener y mejorar las prestaciones globales de una organización y sus capacidades (Toledo, 2007).

La norma ISO 9000 no tiene como requisito que exista una persona exclusivamente encargada de la gestión del sistema de calidad, pero si requiere que exista procedimientos, documentación y descriptores de puestos donde se especifiquen cada una de las funciones de las personas involucradas en el sistema, como se muestra en la Figura 9 (Toledo, 2007).

Figura 9. *Familia ISO 9000*

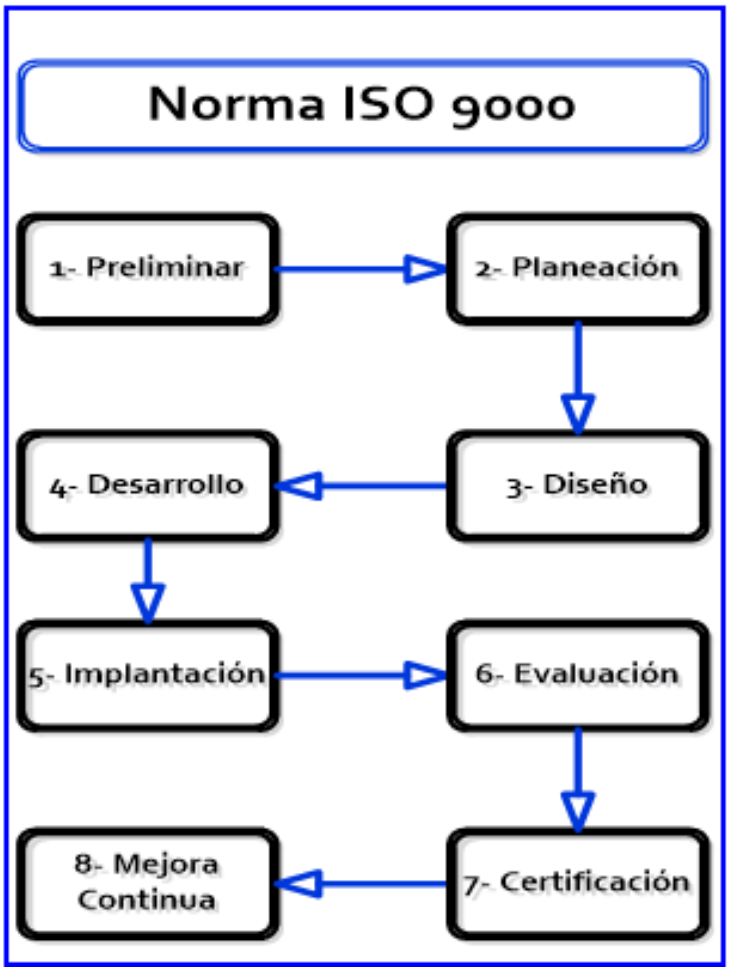


Nota: El gráfico muestra cada uno de los componentes que deben contemplarse para garantizar la gestión adecuada del sistema de calidad. *Fuente: Toledo (2007)*

Proceso de implementación. Existe un proceso de implementación que ha sido tomado de algunos estudios realizados con empresas de tamaño medio. Este proceso consta de ocho fases, las cuales están interrelacionadas, como se detalla en la Figura 10 (Toledo, 2007).

Figura 10.

Proceso de implementación



Nota: En el grafico se muestran cada una de las fases que componen el proceso de implementación. Fuente: elaboración propia (2016)

El proceso de implementación de la norma ISO 9000 supone que se establecerá un equipo de proyecto transfuncional para planear y guiar todo el trabajo, con el apoyo decidido de un comité ejecutivo compuesto por los altos directivos. Los equipos compuestos por elementos más pequeños y por personas con conocimientos serán los que en realidad diseñen el sistema de calidad (Toledo, 2007).

El proceso de certificación se inicia con un diagnóstico de la situación actual de la empresa. En este sentido, se deben determinar cuáles son las condiciones de los sistemas de calidad que existan en ella e identificar los puntos débiles. Asimismo, es necesario considerar elaspecto técnico del proceso de certificación, el aspecto económico implícito en el mismo y por último el aspecto humano. Sobre este último aspecto, es necesario crear en el personal un compromiso de mejora que lleve a la adopción de cambios culturales que orienten las nuevas prácticas hacia la calidad y la satisfacción del cliente (Toledo, 2007).

La mejora continua no se da por sí sola. Es todo un trabajo que puede ser el comienzo de un gran cambio y que involucra a todos los miembros de la organización. Una vez cumplida estaparte, se realizan las auditorías por parte de la Empresa Certificadora. La empresa puede y debe realizar una preauditoria de certificación que, a manera de ensayo final, permita enmendar todos los errores que el nuevo sistema de calidad implantado pueda presentar antes de la evaluación formal realizada ya por la Empresa Certificadora (Toledo, 2007).

Importancia ISO 9000. El crecimiento de la competitividad empresarial obligó a las organizaciones a idear e implementar nuevas y mejores prácticas empresariales relacionadas con la calidad. Estas prácticas, eran muy diversas y dificultaban el intercambio comercial de bienes y servicios entre los diferentes países por poseer cada uno características (costumbres, idioma, idiosincrasia, etc.) particulares y diferentes al resto de países. Los países involucrados se vieron en la necesidad de crear un parámetro internacional que regule las prácticas organizativas y que permita un intercambio confiable de bienes y servicios de calidad. Es así como surgen las normasISO 9000, como estándares que permiten seleccionar, implementar y mantener sistemas que aseguren realmente la calidad de los bienes producidos y que respalden el prestigio de unas empresas frente a otras (Xitumul, 2007).

Esta norma contiene las directrices para seleccionar y utilizar las necesarias para el aseguramiento de la calidad, es decir, es la que permite seleccionar un modelo que lo permita. Entre estas están las ISO 9001/9002/9003 que en la actualidad son resumidas en el ISO 9001. La norma ISO 9004 establece directrices relativas a los factores técnicos,

administrativos y humanos que afectan a la calidad del producto, es decir, establece directrices para la gestión de la calidad (Xitumul, 2007).

La norma ISO 9000 establece directrices relativas a los factores técnicos, administrativos y humanos que afectan a la calidad de los servicios. Las normas ISO 9001/9002/9003 establecen requisitos que determinan los elementos que tienen que comprender los sistemas de calidad, perono es el propósito imponer uniformidad en estos. Son genéricas e independientes de cualquier industria o sector económico concreto. Las nuevas prácticas administrativas están obligando a las empresas a cuidar y controlar los campos relacionados con la calidad, el ambiente y la seguridad y salud ocupacional, pues los efectos de éstos siempre se encuentran interrelacionados.El proceso de implementación de cualquier Sistema de Gestión es largo, tedioso y costoso. Sin embargo, los beneficios que pueden obtenerse de los mismos trascienden todo tipo de esfuerzo y elevan a la organización hacia un nuevo nivel de competitividad (Xitumul, 2007).

Desde el punto de vista de la producción se puede prever una relación positiva entre la norma ISO 9000 y la rentabilidad, debido, al menos a dos factores: en primer lugar, porque las mejoras de calidad pueden ayudar a reducir los costos de producción (como los derivados de inspección y evaluación); y en segundo lugar porque las mejoras en los sistemas de calidad pueden reducir los costes de transacción de producto-mercado en las empresas. Por otro lado, seafirma también que las empresas que cuentan con la norma ISO 9000 pueden ver incrementadassus ventas y su cuota de mercado, y que debido a posibles economías de escala y otros factores relacionados a la rentabilidad de la empresa, puede verse incrementada (Xitumul, 2007).

G. Modelo CMMI

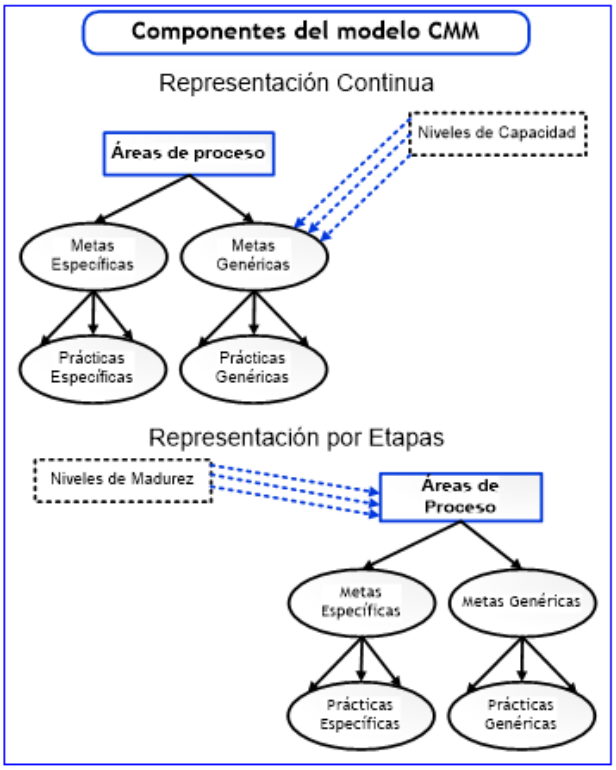
El Instituto de Ingeniería de *Software* de los Estados Unidos (*Software Engineering Institute* o SEI por sus siglas en inglés) es un organismo financiado por el Gobierno Federal de Estados Unidos y operado por la universidad Carnegie Mellon de dicho país (Bush, 2005); el cual apoya a las organizaciones a mejorar el estado de las prácticas de ingeniería,

con la meta de proporcionarles las pautas de actuación necesarias para obtener mejoras observables en su proceso de desarrollo del *software*, de manera que se desarrolla en productos sin defectos que respeten los requerimientos, fechas y costos (Bush, 2005).

Asimismo, persigue acelerar la introducción en las organizaciones de producción de software, de las prácticas y técnicas de ingeniería del software más eficaces y eficientes para identificar, evaluar y mejoraraquellas que se consideren útiles; mantener a largo plazo la competitividad en ingeniería del *software* y en la gestión del cambio tecnológico; habilitar a organizaciones privadas y públicas y trabajar con ellas, para que hagan mejoras en sus prácticas de ingeniería del software; fomentar la adopción y uso continuo de estándares de excelencia en prácticas de ingeniería del *software*. Todo ello, se consigue a través del cumplimiento de cuatro objetivos que se muestra en la Figura 11:

Figura 11.

Modelo CMM



Nota: El gráfico muestra cada uno de los componentes necesarios para alcanzar los objetivos propuestos por SEI. Fuente: Chrissis (2012).

A partir de noviembre de 1986 el SEI, a requerimiento del Gobierno Federal de los Estados Unidos de América, desarrolló una primera definición de un modelo de madurez de procesos en el desarrollo de *software*, que se publicó en septiembre de 1987. Este trabajo evolucionó al modelo CMM o SWCMM (CMM *for Software*), cuya última versión (v1.1) se publicó en febrero de 1993. Este modelo establece un conjunto de prácticas o procesos clave agrupadas en Áreas de Proceso Clave (*KPA - Key Process Area*) (Chrissis, 2012).

Para cada área de proceso define un conjunto de buenas prácticas que habrán de ser:

- Definidas en un procedimiento documentado
- Provistas (la organización) de los medios y formación necesarios
- Ejecutadas de un modo sistemático, universal y uniforme(institucionalizadas)
- Medidas
- Verificadas

A su vez estas áreas de proceso se agrupan en cinco "niveles de madurez", de modo que una organización que tenga institucionalizadas todas las prácticas incluidas en un nivel y sus inferiores, se considera que ha alcanzado ese nivel de madurez. El valor obtenido es un indicador de toda la empresa, aunque puede darse el caso de que en algún departamento tenga un nivel de madurez mayor o inferior al resultante (Chrissis, 2012).

El modelo CMMI (Modelo integrado de madurez de la capacidad) se usa para evaluar el nivel de madurez de una compañía en términos de desarrollo informático. CMMI, una versión más amplia, se basa en CMM, adopta la mayoría de sus conceptos y ofrece los índices de referencia de las mejores prácticas para el desarrollo de *software*. El objetivo es alentar a las compañías para que monitoreen y mejoren continuamente sus procesos, y evalúen el nivel de madurez de estos en una escala de cinco niveles establecida por el CMMI (Chrissis, 2012).

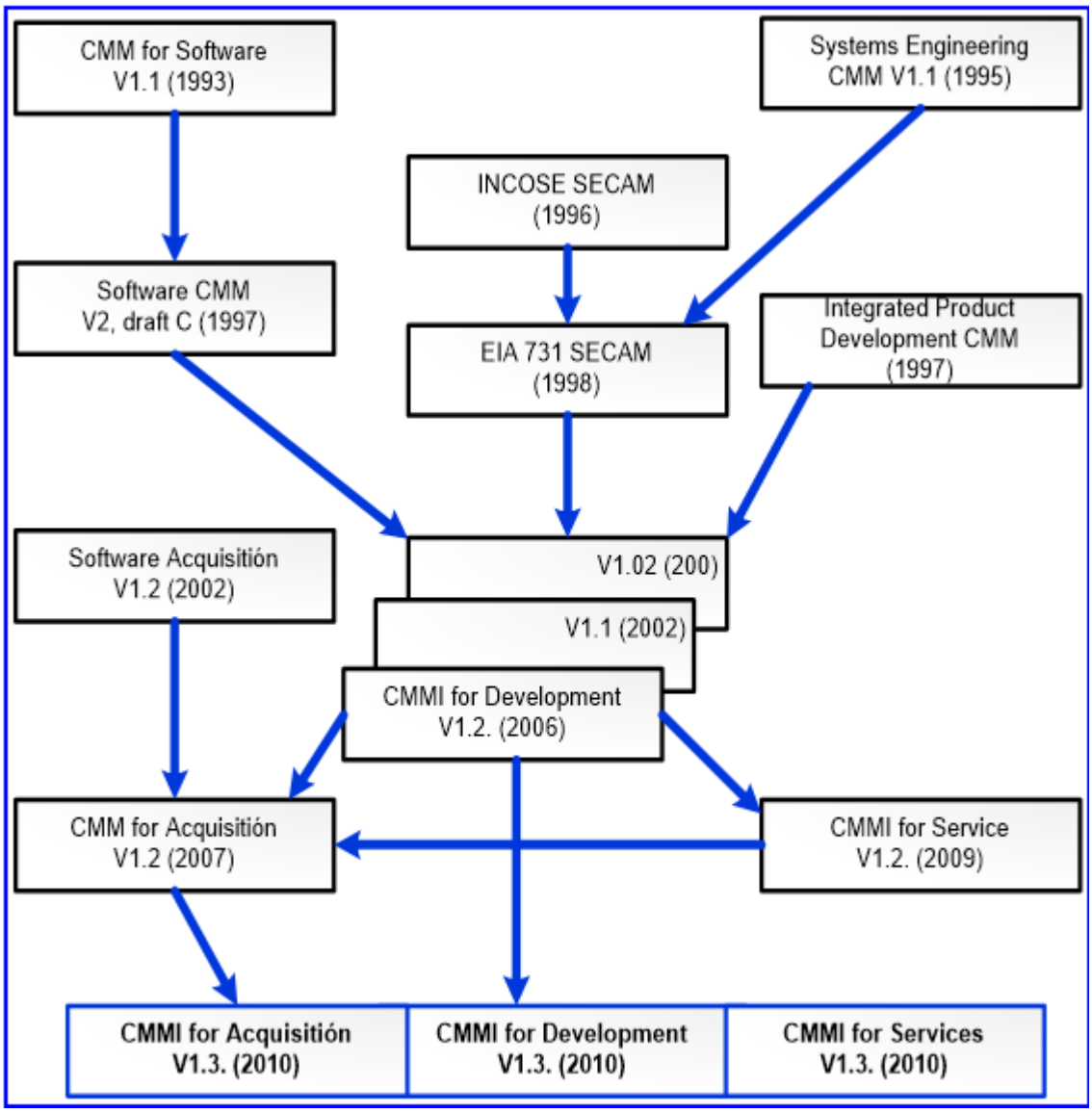
Evolución del modelo CMM. CMMI es la evolución de CMM. CMM Fue desarrollado desde 1987 hasta 1997. En 2002, se lanzó CMMI Versión 1.1, luego en agosto de 2006 siguió la versión 1.2. El objetivo del proyecto CMMI es mejorar la usabilidad de modelos de madurez integrando varios modelos diferentes en un solo marco (*framework*) (Chrissis, 2012). Se desarrolló sobre el principio de calidad de Jurán de solvencia contrastada en la producción industrial: "la calidad del resultado depende principalmente de la calidad de los procesos empleados en su desarrollo" (Chrissis, 2012).

En los años ochenta el Congreso de los Estados Unidos aprobó la creación del Instituto de Ingeniería del *Software* -SEI-, un organismo de investigación para el desarrollo de modelos de mejora para los problemas en el desarrollo de los sistemas de *software*, y para evaluar la capacidad de respuesta y fiabilidad de las compañías que suministran *software* al Departamento de Defensa. El SEI fue desarrollado por el Departamento de Defensa Americano y la Universidad de Carnegie Mellon y empieza a trabajar en un marco de madurez de procesos que permitieron evaluar a las empresas productoras de *software*. La investigación evolucionó hacia el "Modelo de Madurez de las Capacidades (CMM)" (Chrissis, 2012).

En los años noventa, SEI publica la versión 1.0 del Modelo de Madurez de las Capacidades para el *Software* (SW-CMM, *Capability Maturity Model for Software*). En 1993, SEI publica la versión 1.1 de SW-CMM, en 1997 se realiza la publicación de la versión 1.2 y en los 2000 SW-CMM fue integrado y relevado por el nuevo modelo CMMI. La evolución del modelo de CMM se detalla en la Figura 12 (Chrissis, 2012):

Figura 12.

Evolución del modelo CMM



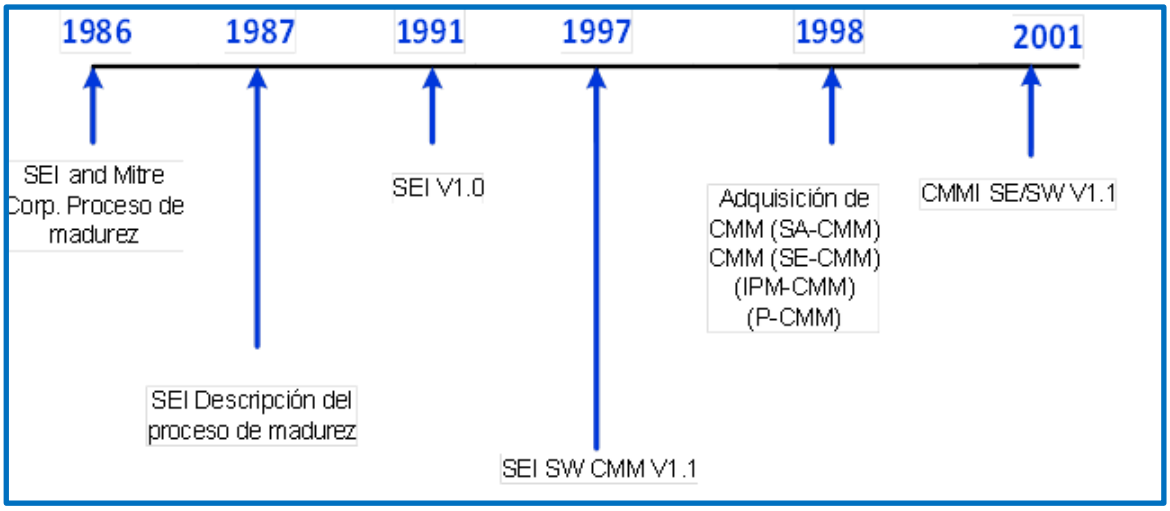
Nota: El gráfico muestra la evolución que ha tenido el Modelo de Madurez de las Capacidades para el *Software* desde 1993 a 2010. Fuente: Chrissis (2012).

El CMMI se desarrolló para facilitar y simplificar la adopción de varios modelos de forma simultánea, y su contenido integra y da relevo a la evolución de sus predecesores, para una visualización la línea del tiempo se detalla en la Figura 13 (Chrissis, 2012):

- CMM-SW (*CMM para software*).
- SE-CMM (Modelo de madurez de la capacidad de sistemas).
- IPD-CMM (Desarrollo integral del producto**).**

Figura 13.

Línea del tiempo CMM

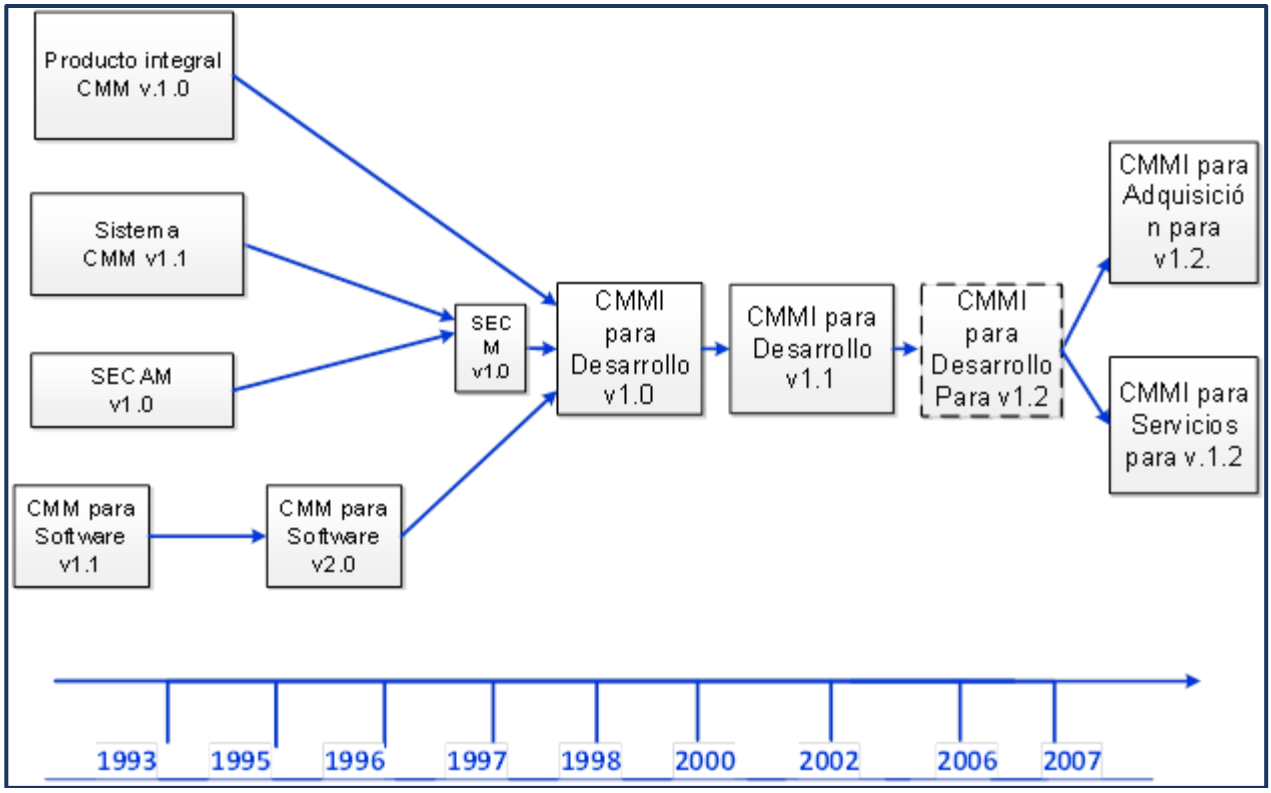


Nota: En el gráfico se muestra la evolución en la línea del tiempo del CMM. Fuente: elaboración propia (2016).

El modelo CMMI constituye un marco de referencia de la capacidad de las organizaciones de desarrollo de *software* en el desempeño de sus diferentes procesos, proporciona una base parala evaluación de la madurez de estas y una guía para implementar una estrategia para la mejora continua de los mismos. CMMI-SW, diseñado por el SEI, como una versión mejorada de modelo CMM, también está compuesto por cinco niveles de madurez, especificando para cada uno de ellos los objetivos que deben ser cubiertos para que una organización pueda ser calificada con el nivel de madurez correspondiente, el desarrollo se puede observar en la Figura 14 (Konrad, 2009):

Figura 14.

Desarrollo del CMM



Nota: El gráfico muestra la evolución del desarrollo de los niveles de madurez entre 1993 y 2007. Fuente: elaboración propia (2016).

Componentes CMM

Los componentes se clasifican en distintos tipos, los cuales se describen a continuación:

Objetivo genérico requerido: los objetivos genéricos asociados a un nivel de capacidad establecen lo que una organización debe alcanzar en ese nivel de capacidad. El logro de cada uno de esos objetivos en un área de proceso significa mejorar el control en la ejecución del área de proceso (Konrad, 2009).

Objetivo específico requerido: los objetivos específicos se aplican a un área de proceso y localizan las particularidades que describen que se debe implementar para satisfacer el propósito del área de proceso (Konrad, 2009).

Los componentes esperados se clasifican en los siguientes tipos:

Práctica genérica: una práctica genérica se aplica a cualquier área de proceso porque puede mejorar el funcionamiento y el control de cualquier proceso.

Práctica específica: una práctica específica es una actividad que se considera importante en la realización del objetivo específico al cual está asociado. Las prácticas específicas describen las actividades esperadas para lograr la meta específica de un área de proceso.

Componentes informativos: propósito, notas introductorias, nombres, prácticas, y productos típicos.

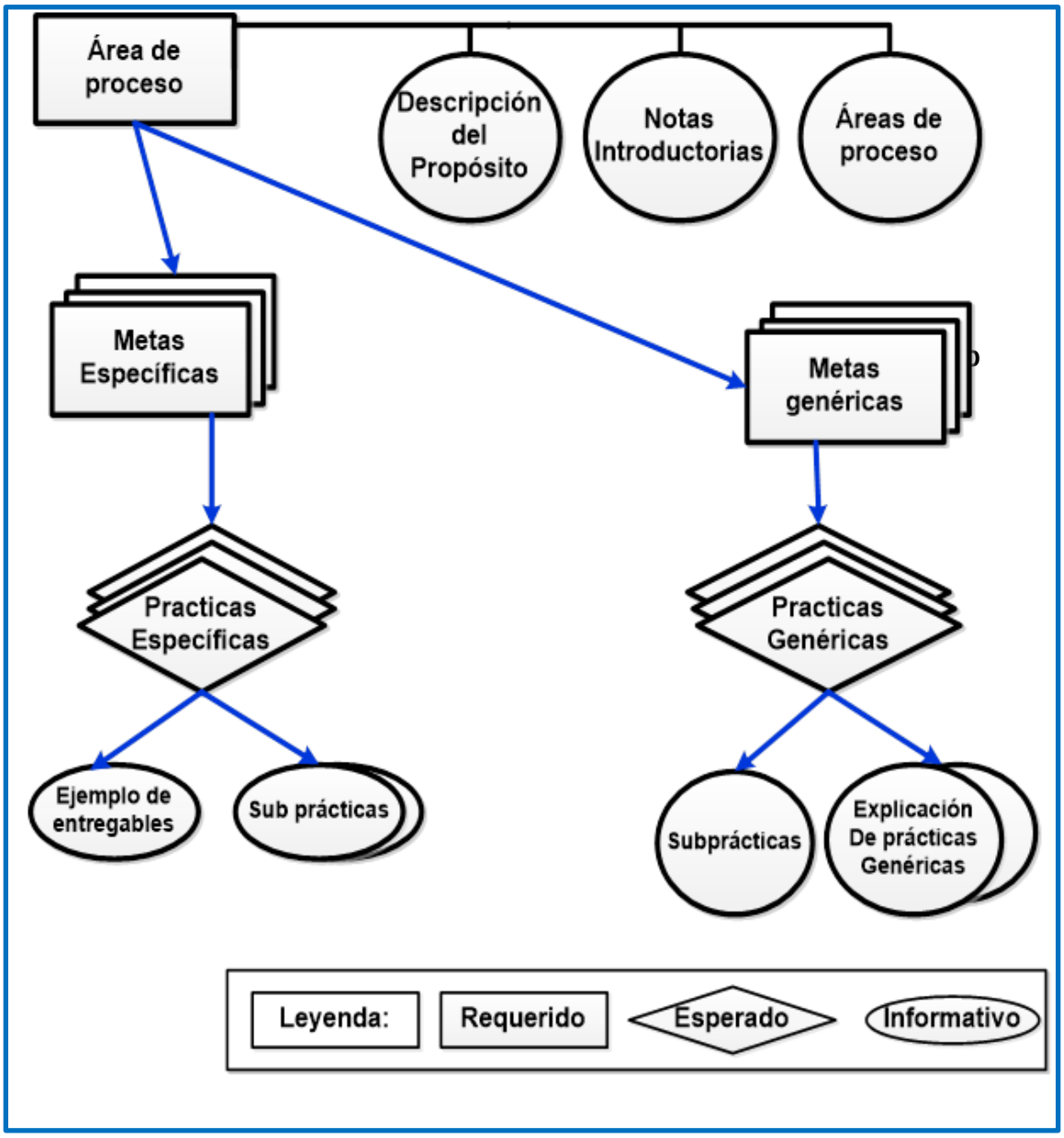
Sub-prácticas: una sub-práctica es una descripción detallada que sirve como guía para la interpretación de una práctica genérica o específica.

Ampliaciones de disciplina: las ampliaciones contienen información relevante de una disciplina particular y relacionada con una práctica específica.

Los componentes esperados se refieren a las cualidades CMMI que describen las actividades que son importantes para lograr un elemento de CMMI requerido y orientan a quienes implementan mejoras o realizan evaluaciones. Los componentes esperados en CMMI son las prácticas específicas y genéricas, como se detalla en la Figura 15. Antes de que las metas puedan considerarse satisfechas, sus prácticas tal y como se describen, o prácticas alternativas aceptables, deben estar presentes en los procesos planificados e implementados de la organización (Konrad,2009).

Figura 15.

Componentes del CMM



Nota: El gráfico muestra las prácticas específicas y genéricas que son necesarias para lograr un elemento del CMMI. Fuente: elaboración propia (2016).

Los componentes informativos son componentes CMMI que ayudan a los usuarios del modelo a comprender los componentes CMMI requeridos y esperados. Estos componentes pueden ser ejemplos en un recuadro, explicaciones detalladas u otras informaciones útiles. Las sub-prácticas, las notas, las referencias, los títulos de metas, los títulos de prácticas, las fuentes, los ejemplos de productos de trabajo y las elaboraciones de prácticas genéricas son componentes informativos del modelo (Konrad, 2009).

El material informativo juega un papel importante en la comprensión del modelo. Frecuentemente, es imposible describir adecuadamente el comportamiento requerido o esperadode una organización usando sólo una meta o la declaración de una práctica. El material del modelo proporciona información necesaria para lograr la correcta comprensión de las metas y prácticas y, por ello, no se puede ignorar (Konrad, 2009).

Ventajas y desventajas del CMM. SEI publicó un informe especial que presenta evidencias cuantitativas creíbles de las mejoras en rendimiento y calidad obtenidas, tanto en coste (ahorros y disminución en coste de encontrar y reparar errores), planificación (disminución del tiempo necesario para terminar tareas y aumento de la fiabilidad de las predicciones sobe estimaciones), calidad (reducción de tasa de defectos) y satisfacción del cliente como retorno dela inversión (Konrad, 2009).

Entre sus fortalezas se destacan las siguientes:

- Inclusión de las prácticas de institucionalización, que permiten asegurar que los procesos asociados con cada área de proceso serán efectivos, repetibles y duraderos.
- Guía paso a paso para la mejora, a través de niveles de madurez y capacidad (frente a ISO).
- Transición del “aprendizaje individual” al “aprendizaje organizacional” por mejora continua, lecciones aprendidas y uso de bibliotecas y bases de datos de proyectos mejorados.

Cabe destacar la reducción del coste en general y de los relacionados con la localización y resolución de defectos. Además, se reduce el rango de variabilidad de costes a medida que evoluciona el proyecto y aumenta la productividad (Konrad, 2009).

También se consigue un ahorro de "tiempo", al mejorar la fiabilidad de la planificación, disminuir el retraso medio de las tareas del 50 al 10 por ciento, incrementar el 30 por ciento la productividad, reducir en un 60 por ciento los trabajos derivados de los resultados de las fases de pruebas, y aumentar la efectividad sobre la planificación realizada (Konrad, 2009).

Se obtiene una mejora en la calidad de producto al reducir el número de defectos que son detectados en las fases tempranas de su ciclo de vida. La mejoría en la satisfacción al cliente y elaumento de las calificaciones obtenidas en las encuestas de satisfacción, así como las mejoras enel ROI (Retorno de la Inversión) son otros de los beneficios de CMMI (Konrad, 2009).

El CMMI influencia en el "proceso de mejora", como qué prácticas genéricas y áreas de proceso de ingeniería aportan un camino de mejora de la capacidad de cada área de proceso en la representación continua, respuesta rápida y guiada a nuevas demandas de las necesidades del negocio, o la inicial priorización de los esfuerzos, o la puesta en marcha de funciones de gestión de proyectos, que darán soporte al resto del proceso (Konrad, 2009).

Algunas de las debilidades que se podrían destacar son:

- El CMMI puede llegar a ser excesivamente detallado para algunas organizaciones.
- Puede ser considerado prescriptivo.
- Requiere mayor inversión para ser completamente implementado.
- Puede ser difícil de entender.

Pros y contras de la aplicación del modelo CMMI en empresas

- La no existencia de una guía a medida de pequeñas organizaciones.
- Reconocen que inicialmente se dirigía a grandes corporaciones, pero la representación continua permite seleccionar sólo aquellas áreas de proceso de interés (adquirido por pequeñas empresas).
- Puede ser demasiado grande para pequeñas organizaciones.
- Crecimiento casi exponencial del número de áreas, prácticas, tiempo, recursos y costes, pero si se alinean los procesos a las necesidades de la organización, se beneficiarán de un proceso estructurado.
- ROI (Retorno de la inversión) no ha sido validado aún en CMMI.
- CMMI resalta la ingeniería de sistemas frente a la ingeniería del *software*.
- SW-CMM exitoso, su mercado son empresas de *software*, pero los interfaces con otros sistemas, con *hardware* o con responsables de otra parte del sistema mejoran el esfuerzo en ingeniería del *software*.
- CMMI es demasiado normativo, en especial con pequeñas organizaciones que, además, funcionan y evolucionan de distinta manera que las grandes.
- CMMI parece ser escrito para organizaciones con madurez y vagamente escrito para ser usado en valoraciones.

Outsourcing. El concepto de *outsourcing* o tercerización empezó a incursionar en el ámbito empresarial y en principio se ofreció como una interesante y atractiva forma de vincular personal para prestación de servicios temporales a particulares u otras empresas. Pronto empezaron los inconvenientes de índole legal, en cuanto al personal contratado, ya que no se le reconocían las prestaciones laborales del caso, porque se le contrataba bajo la modalidad de prestación de servicios (Levison, 2002).

Fundamentalmente se le podría caracterizar en líneas generales como un cierto tipo de contratación indirecta, para el objeto de la prestación de bienes y/o servicios, por parte de una empresa contratante, respecto de una empresa que oferta de los mismos o empresa

contratada (Levison, 2002).

Dada la naturaleza de la necesidad a cubrir por parte del contratante, se abre un generoso abanico de posibilidades de parte de las empresas que ofertan, que van desde el servicio de fotocopias, aseo y limpieza de oficinas, hasta el servicio de correo, entrega puerta a puerta desobres y/o documentos, servicio de transporte vehicular, servicio de vigilancia de empresas, laboratorios, colegios, escuelas, entre otros (Levison, 2002).

Existen al menos cuatro casos tipificados de subcontratación y tercerización laboral.

- Lo cual no pretende ser exhaustivo.
- Intermediación laboral
- Suministro de mano de obra temporal
- Utilización de trabajadores autónomos e independientes

Prueba de integración del modelo de madurez. El *Testing Maturity Model Integration* -TMMI- es el modelo que determina las bases para la mejora de procesos para *testing*, el cual provee la forma de categorizar y describir cómo se debe trabajar el proceso de *testing* que permita conseguir la mejora constante. Está diseñada para evaluar a una organización y verificar si esta cumple con el modelo y el nivel necesario para una certificación para dichos procesos este modelo es popular en la disciplinadel desarrollo de *software* y de *testing* de *software*. Este modelo nace como complemento del CMMI y define una jerarquía de cinco niveles, que abarcan desde las etapas practicas básicas hasta la etapa de gestión de medición y optimización (Rodríguez, 2012).

Los niveles definidos por TMMI con respecto a los procesos son:

Nivel 1: Inicial

En este nivel, el *testing* es casi inexistente, ya que aún no existen procesos definidos y no se jerarquiza su actividad (en este nivel se suele confundir el *testing* con la depuración de código). Los casos de éxito no suelen ser asociados al resultado de la aplicación de un proceso, por lo cual, la situación de éxito no es repetible (Rodríguez, 2012).

El deseo más común en las empresas en el ámbito de calidad en TMMI es liberar productos que puedan funcionar sin errores de gravedad alta. Generalmente los productos, no cubren las expectativas de calidad ya sea en rendimiento, estabilidad, disponibilidad y ausencia de funcionalidades por lo general con satisfacción media del cliente. Es común que las organizaciones en este nivel no dediquen recursos a crear departamentos de pruebas, entornos o ambientes dedicados a pruebas, así como herramientas, o personal capacitado (Rodríguez, 2012).

Nivel 2: Gestionado

Para el nivel dos de TMMI, el *testing* es un proceso gestionado y diferenciado de las actividades de depuración de código. En las organizaciones que adoptan la disciplina del nivel dos, se suele ver que se mantienen las practicas aún en momentos de presión y estrés. El *testing* se realiza en una etapa posterior a la implementación. Los requerimientos se encuentran de forma escrita y en este nivel se incluye tiempo para realizar las actividades de *testing* y se indica por quién será llevada a cabo (Rodríguez, 2012).

Dicho *testing* incluye un plan detallado con el enfoque que tendrá y un análisis de riesgo como, por ejemplo, casos de prueba con base en los requerimientos y documentos generados. El *testing* es realizado en varias etapas y se llevan a cabo diferentes pruebas como integración, componentes del producto y seguimiento que permite crear métodos correctivos que permita solucionar los problemas encontrados (Rodríguez, 2012).

El objetivo principal del *testing* en este nivel es verificar que el producto cumpla con sus requerimientos, pero por la inclusión en etapa tardía del ciclo de vida del desarrollo es común que los defectos se propaguen en el sistema (Rodríguez, 2012).

Nivel 3: Definido

En el nivel tres de TMMI, el *testing* deja de ser considerado como una etapa posterior a la implementación y pasa a estar integrado en el proceso de desarrollo, es decir, en todo el ciclo de vida. Existe planificación temprana de *testing*, y se considera la profesión de *Tester*. Se crean revisiones formales y toman valor al momento de crear los requerimientos ya que los Tester son incluidos en el análisis del proyecto, permitiéndole prestar mayor atención a los requerimientos no funcionales tales como la usabilidad, confiabilidad, entre otros (Rodríguez, 2012).

El nivel tres de TMMI es un mejoramiento del proceso del nivel dos, donde además los procesos, conjunto de estándares de *testing*, y procedimientos son definidos con mayor detalle. Manipular un nivel tres en un proyecto, involucra el tomar en cuenta estándares que mejor apliquen para el producto que se verificara (Rodríguez, 2012).

Nivel 4: Medido

En el nivel cuatro el proceso de *testing* es cuidadosamente definido, fundado y medible rigurosamente. En este nivel se considera el *testing* como una evaluación exhaustiva de todas las actividades por las que pasa el desarrollo del producto y la funcionalidad relacionada. Se debe incluir métricas que controlan y evalúan el rendimiento del proceso de *testing*. Es decir, que la calidad del producto es medible cuantitativamente y permite identificar las necesidades de calidad y las métricas a utilizar (Rodríguez, 2012).

Las revisiones e inspecciones cobran importancia y son consideradas como parte del proceso de *testing*, sobre todo las revisiones por pares. Además, se crean criterios cuantitativos como confiabilidad, usabilidad y mantenibilidad, también el apoyo de

construcción de planes de *testing*, establecimiento de estrategias de como verificar el producto y el enfoque que tendrá el *testing* en todo el proceso (Rodríguez, 2012).

Nivel 5: Optimizado

El nivel cinco de TMMI tiene como objetivos principales la prevención de defectos y la mejora continua a lo largo de todo el proceso del producto. Consta de tres áreas de proceso que se encuentra intrínsecamente relacionadas que son: el control de calidad, la prevención de defectos, y la mejora del proceso de *testing*. Al alcanzar todos los niveles previos, la organización cuenta con un proceso de *testing* medible y definido. Este nivel se encuentra como base una infraestructura organizacional que apoya al *testing*, ya que la entidad es capaz de mejorar su proceso de *testing*, incorporando mejoras incrementales en tecnología, innovación, nuevasherramientas, y metodologías que ayuden a promover la mejora continua en sus productos (Rodríguez, 2012).

Así como también, la mejora de los procesos con apoyo de herramientas, reutilización de documentación de pruebas y ajustando los pasos a seguir para generar nuevos procesos que incentiven el cumplimiento de mejora (Rodríguez, 2012). El papel principal para este nivel es la prevención de los defectos. Es fundamental identificar y analizar las causas más comunes, para establecer acciones preventivas para que no ocurran de nuevo defectos similares. Además, como parte de la prevención es el análisis de los resultados del proceso de control de calidad es necesario para evaluar el rendimiento de procesosy utilizarlo para generar acciones correctivas para un futuro.

El proceso de *testing* es gestionado de forma estática, ya que se maneja un área de proceso de control de calidad, el cual es medido por aspectos de confianza y confiabilidad en las pruebas. En este nivel se espera que el proceso de *testing* sea gestionado, definido, medido, eficiente y sobre todo efectivo. Las pruebas deben ser controladas y llegar a ser predecibles, con un foco deprevención de defectos. Es importante que este proceso siempre sea apoyado por automatización de pruebas, que contribuya a la organización, que se encuentre enfocado en cambios de proceso para lograr la mejora continua y facilite la

reutilización del *testing* (Rodríguez, 2012).

TPI: Mejora del proceso de prueba. El modelo de mejora de procesos TPI se deriva de la aplicación de proceso de *testing* TMAP desarrollado por una de las empresas líderes de *testing*, SOGETI, el cual es definido por pasos de mejora incrementables y controlables y abarca aspectos del proceso desde el uso de herramientas, técnicas de diseño de casos de pruebas, gestión de pruebas, entre otros (Rodríguez, 2012).

TPI se basa en 20 áreas clave y posee diferentes niveles para cada área, para pasar de un nivel a otro se es necesaria mayor eficiencia y efectividad dominada en cada área. El nivel superior se alcanza con la madurez requerida en todos los niveles inferiores. En cada área se establecen metas que deben ser cumplidas para pasar al siguiente nivel. Este proceso destaca que entre cada área existen aspectos de procesos de *testing* que comúnmente se encuentran en los procesos, y ayuda a verificar los aspectos o escenarios no contemplados en el proceso de *testing* (Rodríguez, 2012).

Las 20 áreas clave son:

- Estrategia de prueba
- Modelo de ciclo de vida
- Momento de la participación
- Estimación y planificación
- Técnicas de especificación de pruebas
- Técnicas de pruebas estáticas
- Métricas
- Automatización de pruebas
- Entorno de pruebas
- Entorno de oficina

- Compromiso y motivación
- Funciones de prueba y formación
- Alcance de la metodología
- Comunicación
- Elaboración de informes
- Gestión de defectos
- Gestión del *software* de prueba
- Gestión del proceso de pruebas
- Evaluación
- Pruebas de bajo nivel

H. Certificaciones profesionales de *testing*

ISTQB

El *International Software Testing Qualifications Board* – ISTQB- es una organización decertificación de la calidad del *software* que opera internacionalmente. El ISTQB fue fundado en noviembre de 2002 en India y está legalmente registrado en Bélgica. Esta organización es la encargada de definir un esquema de certificación internacional, en la que se establecen guías de acreditación y evaluación profesional para el cargo de *testing*: Este es evaluado por cada comité a cargo en cada país. El ISTQB ha creado el esquema más exitoso del mundo para la certificación de los probadores de *software*, el cual es implementado por las mejores empresas en *testing*. Hasta junio de 2013, ha certificado a más de 307.000 testers en 70 países en todo el mundo, con una tasa de crecimiento de aproximadamente 12.000 certificaciones por trimestre (Pérez, 2006).

Los esquemas de certificación internacional con base a niveles de madurez y capacidad que comienza en el nivel básico hasta el nivel experto se detallan a continuación:

- ISTQB Nivel Básico (CTFL)
- ISTQB Nivel Avanzado (CTAL)
  - Prueba para gerente
  - Prueba para analista
  - Prueba técnica
- Nivel Experto
  - Prueba de mejora continua
  - Prueba de gestión
  - Prueba de automatización
  - Prueba de seguridad

ASTQB

La certificación *American Software Testing Qualifications Board* – ASTQB- tiene como objetivo promover el profesionalismo en pruebas de *software*. Fue creado en Estados Unidos en 2003, donde un grupo de expertos en testeo de *software* desarrollaron y promocionaron la certificación ISTQB en los Estados Unidos. Es una empresa sin ánimos de lucro donde se paga únicamente el examen y la acreditación. Tiene como objetivo proporcionar la certificación de analista de sistemas y de negocio, proporcionando las mejores herramientas y capacitando a las personas que se desarrollan pruebas de *software*. Es una de las empresas con mayor prestigio quecertifica en móvil tester (Pérez, 2006).

ISQI

El *International Software Quality Institute* -ISQI- es otro de los líderes de certificaciones a través del mundo con más de seis países sedes entre ellos Estados Unidos, Inglaterra, Holanda, entre otros. ISQI provee certificaciones en IT en más de 90 ciudades en los seis continentes y se brinda en 10 lenguajes, posee más de 20,000 certificantes por año, lleva más de 10 años en el mercado. El certificado incluye los estándares globales de *testing* como lo son *Agile, Business Analysis, Mobile App Testing*, Usabilidad y Experiencia de Usuario (Pérez, 2006).

Esta organización se encuentra aliada con más 100 empresas globales, con colaboración especial de *International Organization for Standardization* -ISO-, lo cual le promueve una enseñanza estable y a la vanguardia de la industria para enseñar los mejores modelos y tipos de pruebas de testing (Pérez, 2006).

IIST

Otra importante empresa de certificación es la *International Institute for Software Testing* -IIST-, la cual promueve la certificación *Certified Mobile Software Test Professional* -CMSTP-, fundada en 1999 en Estados Unidos, es una empresa dedica al testeo de software, aseguramiento de calidad y certificación de testeo de *software* (Pérez, 2006).

Fue creada para desarrollar y conocer todas las habilidades, estrategias y necesidades del mercado de aplicaciones móviles. En esta certificación se desarrollan las habilidades y conocimientos requeridos para un efectivo testeo y evaluación sobre aplicaciones móviles ya sea desde páginas web o desde los dispositivos. Dicha certificación tiene la especial tarea de reconocer las necesidades y la ayuda para el control de calidad de manera profesional y enfrentarse a retos de testeo. Es la primera y pionera de la certificación de *testing* móvil y se encuentra basado en el libro- *Mobile Testing Body of Knowledge* -MTBOK- (Pérez, 2006)

Empresas líderes a nivel mundial

En la Tabla 2 se presenta un listado de certificaciones evaluadas, verificadas y más utilizadas para testeo móvil a nivel mundial.

Tabla 2.

Certificaciones que poseen las empresas líderes

EMPRESA	CERTIFICACIÓN
	Metodología de Pruebas Tmap® *Microsoft Partner Gold Competency Status Mobility, ERC & CRM, Collaboration & Content, Application Development, Application Lifecycle Management, Business Intelligence, Devices & Deployment, Management & Virtualization, Data Platform, Server Platform, Communications, Messaging*
	Certificación oficial de ISTQB *(International Software Testing Qualifications Board*
	TMMi y certification official de ISTQB, *Certified Agile Tester*

Continuación Tabla no. 2

	Certificación en CMMI®5
	ISTQB (*International Software Testing Qualifications Board*
	CSQ ISO 9001:2008, ITSM, IQNet
	CSTE *Mobile App Test Automation*

Nota: La tabla muestra un listado de las empresas líderes a nivel mundial, así como certificaciones que poseen cada una de ellas. las Fuente: Testing Magazine, 2015.

También, se detallan a en la Tabla 3, las mejores empresas del mundo que realizan *testing* actualmente:

Tabla 3.

Mejores empresas del mundo en QA testing

	EMPRESA	LOGO	UBICACIÓN
1	**TestCo**		Estados Unidos en Dallas
2	**A1QA**		Estados Unidos en Colorado y Segunda Sede en Reino Unido
3	**BELATRIX – Mobile Testing**		Estados Unidos en Florida, Argentina, Perú
4	**AFOUR Technologies**		India en Pune, en Bangalore, en Koverappa y Estados Unidos en Washington
5	**APPLAUSE**		Estados Unidos en Boston, Berlín, Israel, Polonia e Inglaterra
6	**Tata Consultancy**		Reino Unido

Continuación Tabla no. 3

7	**Eleks**		Ucrania en Lviv, Inglaterra y USA en Las Vegas
8	**Amdocs**		Canadá, USA en Missouri, Australia, Japón, Corea, India, Singapur, Taiwán, Tailandia, Vietnam, Rusia, Alemania y Dubái.
9	**Apica**		Estocolmo y USA en California
10	**Centre4 Testing (Meet Ten10)**		Reino Unido
11	**Test Matick**		Estados Unidos en New York
12	**PerformanceLab**		Estados Unidos en California
13	**PractiTest**		Israel en Rehovot y Estados unidos en New York
14	**QASource**		Estados Unidos en California
15	**QualiTestGroup**		Estados Unidos en California, en Dallas, en Bronx
16	**Soflab Technology**		Polonia en Warszawa

Continuación Tabla 3.

17	**Sogeti**		Reino Unido
18	**Test Insane**		India en Bengaluru

Nota: En la tabla se muestran cuáles son las empresas que en la actualidad hacen el trabajo de *testing*, así como su país de origen. Fuente: Forbes (2015).

Herramientas. En la Tabla 4 se encontrará detalladamente los escenarios de pruebas que se deben tomar en cuenta y el tipo de *testing* que se utilizará:

Tabla 4.

Escenarios de prueba y testing

Aplicación	Tipo
Enterprise Architect	Escenarios de Prueba
Experistes	Reporte y Análisis de Testing
SmartBear	Manejo de Riesgos
Testbird	Reporte de Testing
Tortoise	Repositorio
Visual Studio ALM	Escenarios de Prueba
XBosoft	Reporte y Análisis de *Testing*

Nota: La tabla muestra las aplicaciones y el tipo de *testing* que se debe utilizar en cada una de ellas. Fuente: Moinformática (2016).

En la Tabla 5 se puede ver detalladamente las herramientas que se utilizan para la automatización de pruebas móviles y del sistema operativo compatible con cada una.

Tabla 5.

Automatización de herramientas de testing móvil

Aplicación	IOS	Android	Windows	BlackBerry
Appium	X	X	-	-
BugClipper	X	X	-	-
Calabash	X	X	-	-
DeviceAnywhere	X	X	-	-
eggPlant	X	X	X	-
Frank	X	-	-	-
Google Developers	-	X	-	-
HockeyApp	X	X	X	-
iPad Peek	X	-	-	-
iPhone Tester	X	-	-	-
JMeter	X	X	X	X
KeepItFunctional	X	-	-	-
KeyNote	X	X	X	-
Mobile Labs Trust	X	X	-	-
Monk|ey	-	X	-	-
MonkeyTalk	X	X	-	-
NativeDriver	X	X	X	-
Ranorex	X	X	X	-
Reflector	X	X	-	-
Robotium	-	X	-	-
Scirocco	-	X	-	-
See Test	X	X	X	X

Continuación Tabla 5

Selendroid	-	X	X	-
Silk Mobile	X	X	X	X
SOASTA	X	X	-	-
Tesdroid	-	X	-	-
TestFairy	-	X	X	-
TestFlight	X	-	-	-
TestObject	X	X	-	-
TestStudio	X	-	-	-
Ubertesters	X	X	-	-
UI Automator	-	-	-	-

Nota: La tabla enlista cada una de las herramientas que pueden utilizarse para la automatización de las pruebas y el correspondiente sistema operativo con el que pueden hacerse. Fuente: elaboración propia (2016)

En la tabla 6, se observan las herramientas para hacer pruebas de páginas web.

Tabla 6.

Emulación web

Aplicación	Tipo
Mobile 58one emulator	Emulador Web Móvil
mobiReady	*Rating* del sitio en móvil
ProtoFluis	Prueba de Interfaz
Responsive	Prueba de Interfaz
Screenfly	Prueba de tamaño de pantalla

Nota: La tabla enlista cada una de las herramientas con las que se pueden realizar pruebas a sitios web. *Fuente: Borland (2016)*

En la tabla 7, se detallan las aplicaciones que requieren un pago por el uso y el tipo de pruebas que se pueden realizar en cada una de ellas.

Tabla 7.

Herramientas pagadas

Aplicación	Tipo	Precio
Appsee	Pruebas con *Touch heatmaps*	$250 por mes
AppThwack	Laboratorio virtual	$20 al mes
HockeyApp	Análisis de vulnerabilidades	$10 por hora
Perfecto Mobile	Pruebas en vivo	$25 por hora
User Testign	Pruebas de usabilidad	$250 10 horas
Xamarin Test Cloud	Laboratorio virtual	Está en versión beta

Nota: La tabla enlista aquellas aplicaciones pagadas, así como el tipo de pruebas que realiza y el costo de cada una de ellas. Fuente: elaboración propia (2016)

V. MARCO METODOLÓGICO

Para la ejecución del proyecto planteado, se tomará como base las metodologías que se utilizan en los centros de testeo a nivel mundial y la aplicación en empresas nacionales. Se creará a partir de los estándares de calidad en productos de software, patrones de diseño y esquemas válidos que faciliten el control de calidad en el funcionamiento de productos móvil.

Los recursos serán únicamente investigativos y serán enfocados en empresas que certifican con los estándares de calidad y los requisitos necesarios para conseguir la certificación como empresa de testeo. La factibilidad se medirá al comparar los procedimientos implementados y sugeridos con los estándares de rentabilidad de control de calidad.

Los productos del trabajo consistirán en guías de implementación, manuales de uso para obtener las certificaciones y opciones de herramientas que podrán ser utilizadas para el testeo de alguna aplicación. Con el desarrollo de este trabajo académico se permitirá realizar un centro de testeo y montar uno con base a las guías obtenidas.

Delimitación del tema

En este proyecto no se dará cubertura el área de *software* de inteligencia artificial (Ejemplo: Robótica, *Machine Learning*, Algoritmos Genéticos, entre otros), desarrollo web o algún tipo de producto que sea diferentes a desarrollo aplicativo móvil y será únicamente del área de funcionamiento de la aplicación, es decir, pruebas funcionales. No se realiza el control de calidad previo a la entrega al desarrollo. Se iniciará con el desarrollo finalizado, casos de uso, análisis de diseño y alcance previamente definido.

Diseño de la investigación

El diseño de la investigación es no experimental y transversal, dado que no se manipularon variables y la recolección de datos se realizó en un único momento temporal.

El carácter de la investigación es exploratorio-descriptivo, lo que permitió profundizar en el conocimiento del tema, en el análisis comparativo de estándares y modelos de calidad en el desarrollo de software para proponer soluciones basadas en la evidencia recopilada.

Fases de la investigación

En la fase exploratoria se realizó una extensa revisión bibliográfica que incluyó libros, artículos científicos, tesis y documentación técnica relacionada con el testeo de aplicaciones móviles, estándares de calidad y metodologías de desarrollo de software.

Para la fase analítica se realizó un análisis comparativo de los diferentes estándares, modelos y metodologías identificados en la fase anterior. Este análisis incluyó:

- Estándares de calidad (ISO 9000, CMMI)
- Modelos de madurez de testeo (TMMI, TPI)
- Certificaciones profesionales en testeo (ISTQB, ASTQB, ISQI, IIST)

En la fase de síntesis y propuesta se integraron los hallazgos de las fases previas para desarrollar una propuesta adaptada al contexto de la Universidad del Valle de Guatemala. La propuesta incluyó:

- **Modelo de negocio:** se diseñó un modelo de negocio viable para el centro de testeo en donde se consideraron aspectos operativos, financieros y estratégicos.

- **Procedimientos operativos:** se elaboró una lista detallada de procedimientos estandarizados para garantizar el funcionamiento óptimo del centro de testeo.

- **Herramientas de gestión:** se desarrollaron instrumentos específicos para la operación del centro en donde se incluyen tablas referenciales, formatos de reportes y matrices de evaluación.

Técnicas e instrumentos de recolección de datos

Análisis documental: se empleó para la revisión de laliteratura especializada. Se utilizaron fichas bibliográficas y matrices de análisis para sistematizar la información recopilada.

Análisis comparativo: se realizó un análisis comparativo para evaluar las diferentes metodologías, estándares y herramientas de testeo, utilizando criterios predefinidos de eficacia, eficiencia y adaptabilidad.

***Benchmarking*:** se implementó para identificar y analizar lasmejores prácticas en centros de testeo de aplicaciones móviles a nivel internacional.

Análisis de datos

El análisis de la información recopilada se realizó mediante las siguientes técnicas. Análisis de contenido, para evaluar la relevancia y aplicabilidad de los estándares y modelos estudiados. Análisis comparativo para contrastar las diferentes metodologías y herramientas de testeo y síntesis para integrar los hallazgos y desarrollar la propuesta final.

Criterios de rigor científico

Para asegurar la calidad y credibilidad de la investigación se aplicaron los siguientes criterios: dependencia, para mantener un registro detallado del proceso de investigación; credibilidad, por medio de una triangulación de fuentes y métodos; transferibilidad a través de una descripción detallada del contexto y los participantes del estudio y que puedan ser confirmados para asegurar la objetividad en el análisis y presentación de los resultados.

Consideraciones éticas

La investigación se condujo con respeto a los principios éticos de la investigación académica y se incluyó el reconocimiento adecuado de todas las fuentes consultadas y la objetividad en el análisis y presentación de los resultados.

VI. RESULTADOS

La propuesta realizada a continuación tiene como base la recopilación del marco teórico estudiado, el cual se divide en módulos y procesos a realizar para obtener la mejor metodología que se pueda implementar para la Universidad del Valle de Guatemala. Se determinó un modelo de negocio lineal ya que es el más apropiado para modelar el tipo de negocio a desarrollar. Se estudió el modelo en V y W, pero se descalificó ya que el desarrollo debería ser interno, y el negocio se centra únicamente en las pruebas.

Posteriormente, se desarrolló la lista de procedimientos a realizar, la cual asegura un diseño de *testing* adecuado a la organización. Como implementación de los procedimientos se creó el diseño de testo, el cual surge de una mezcla de *testing* diseñado y exploratorio estilo libre, en el cual se crea un sistema a la medida para la ejecución de pruebas. Este servirá para familiarizarse con el producto, aprender de él, investigarlo y crear las pruebas de humo. Al crear evidencia para los interesados, se dejará constancia de la ejecución con la creación de documentación de respaldo.

El *testing* estará basado en sesiones, con gestión de pruebas e incidentes, control de corrección de incidentes, reportes de entregas por iteración, es decir, intervalos de tiempo dedicados a las pruebas según versiones de programas y correcciones. La duración recomendada será según las entregas y cantidad de incidentes encontrado en cada iteración será registrada en una ficha a completar. Además, se determinarán las tablas referenciales en las que se basará la escala de calificación para cada área verificada.

Para complementar las pruebas se creó una lista de riesgos de forma priorizada donde se detallan los riesgos explorados por cada prueba. Estos riesgos serán determinados por el encargado de pruebas y validado por los demás interesados en el proyecto quienes ajustarán y actualizarán el estado cuando sean mitigados o si se inyectarán nuevos.

Para definir los riesgos las actividades de *testing* se centrará en identificarlos, manejarlos y direccionar las actividades de *testing* para mitigarlos. Se crearon las

categorías de criterios de calidad con diferentes tipos de requerimientos que se desean alcanzar, entre las que se encuentran la capacidad, fiabilidad, usabilidad, *performance*, instalación, compatibilidad, soporte, testeo, mantenibilidad, portabilidad, localización, entre otras. También se creó la lista genérica de riesgos que aplica a cualquier producto.

Se utilizará el enfoque de identificar los riesgos asociados a cada situación, es decir, vulnerabilidades (debilidades o posibles fallas), amenazas (entradas o situaciones que puedan explotar la vulnerabilidad) y víctimas (quien puede ser impactado por la falla). Cuando se esquematice el riesgo se debe crear diagramas, flechas o imágenes donde se encuentre el riesgo para poder ser validado nuevamente.

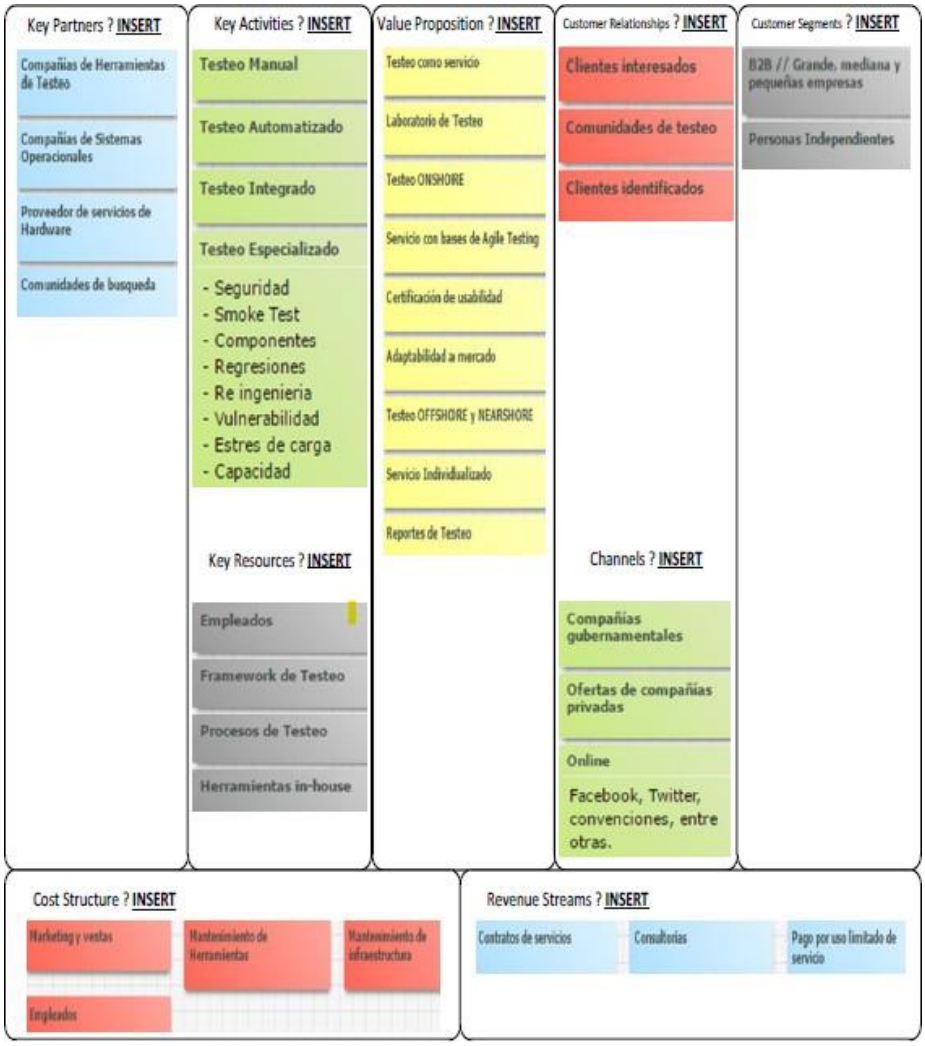
Los reportes se crearán por sesiones con entregas parciales y se culminará en la entrega final, donde se indican el sumario, heurísticas, incidentes, ejecución de pruebas, métricas y resultados.

A. El modelo de negocio

A continuación, se presenta el modelo de negocio para el centro de testeo para aplicaciones móviles y procedimiento a seguir para verificar la factibilidad de establecer dicha entidad dentro de la Universidad del Valle de Guatemala, con el propósito de observar el mercado, verificar si es un plan viable y sobre todo conocer el entorno para verificar si hay retorno de inversión. Es importante resaltar que, con dicha propuesta, se alcanza uno de los objetivos del presente trabajo académico.

Figura 16.

Modelo de negocio para el centro de testeo



Nota: La gráfica muestra el modelo de negocio, junto al tipo de testeo, la relación con los clientes y la segmentación de estos. Fuente: elaboración propia (2016)

Adicionalmente, existen más variantes del modelo de negocio presentado, el cual puede expandirse tomando en cuenta el *testing* para aplicaciones móviles y de aplicaciones web. Se puede realizar variantes creando servicios de manejo o consultoría en cada empresa de forma presencial. Se puede crear *lock-in* de servicio por contrato y manejar tarifas, paquetes o adquisiciones de testeo automatizado ya que al trabajar con alguna empresa se logra obtener el *feedback* y el manejo de la aplicación, por lo cual será más fácil crear

de nuevo pruebas similares o de mejoramiento de la aplicación. Es decir, crear paquetes preferenciales, o de continuidad de revisión para mejoras. Para garantizar el retorno de inversión. Se debe conocer el tipo de negocio y aplicación al que se desea trabajar y qué tipo depropuesta de venta, así como el personal que se utilizará para el manejo del proyecto.

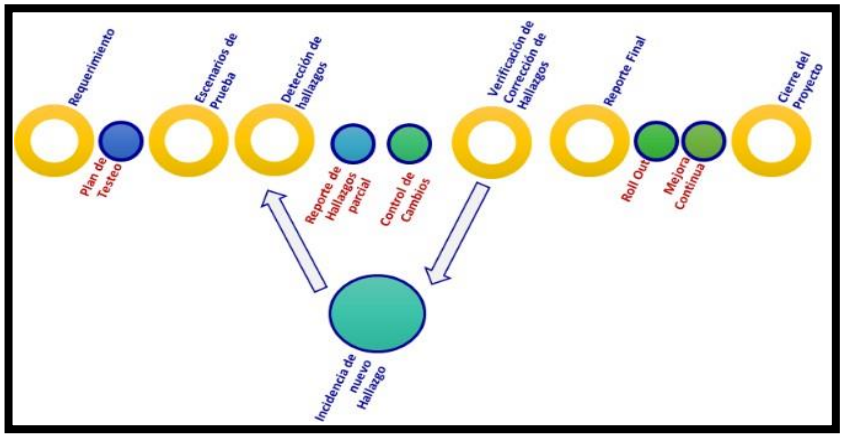
B. Lista de procedimientos

Los procedimientos por seguir en el funcionamiento óptimo de testeo para móvil se definen como los lineamientos o pasos a ejecutar, lo que permite tomar en cuenta todos los escenarios posibles genéricos y especializados para asegurar mayor asertividad en la prueba.

El principal procedimiento es crear escenarios genéricos, es decir, todos aquellos pasos o ejecuciones que se deben realizar obligatoriamente en una aplicación móvil y partir desde esta crear una estrategia efectiva y que se adapte el negocio definido en la Figura 17.

Figura 17.

Modelo de procedimiento de servicio de testeo



Nota: La gráfica muestra los escenarios que deben crearse como procedimiento principal con cada uno de sus componentes en las aplicaciones móviles. Fuente: elaboración propia (2016)

Para cada etapa en la lista de procedimientos se determinaron ciertos documentos a establecerse, los cuales deben ser completados en cada etapa de ejecución de las pruebas tal y como se describe en la tabla 8. Los documentos posteriormente se encuentran detallados, y se propone un ejemplo en el anexo A y B.

Tabla 8.

Documentación según etapas del procedimiento

ETAPA	DOCUMENTOS INVOLUCRADOS
EJECUCIÓN DE PRUEBAS	Casos de Prueba
ANÁLISIS DE INCIDENTES	Informe de Análisis de Defectos
VERIFICACIÓN	Reporte Parcial
INFORME DE RESULTADOS	Reporte Final

Nota: La tabla presenta una lista de las etapas de los procedimientos y la respectiva documentación que debe acompañarlas. Fuente, elaboración propia (2016)

C. Diseño de Testeo

Este diseño permitirá indicar los pasos a seguir para asegurar la certificación de calidad una aplicación móvil, ya que cuenta con una mezcla de los principales pasos y niveles necesarios del aseguramiento de calidad. Para uso explicativo los términos Hallazgo/Incidente/No Conformidad, se tomará como defecto o cualquier eventualidad que no esté de acuerdo con el estándar o funcionamiento requerido por el cliente o funcionalidad de la aplicación.

Para iniciar con el testeo se deben aclarar las precondiciones, o aquellos requerimientos que debe tener el programa para dar inicio con las pruebas.

Precondiciones

a. Conocimiento de situación inicial, conocimiento del requerimiento.

b. Conocer el plan estratégico, alcance o documento de base.

c. Determinar el requerimiento y funciones a validar.

d. Haber seleccionado la gama de plataformas a utilizar.

e. Disponer de dispositivos para las pruebas.

f. Tener valores de prueba validos si la prueba lo necesitara.

Procedimiento de pruebas

Se define como procedimiento de pruebas a los pasos a seguir desde que se solicita la prueba, el manejo del producto, procesos y entrega de resultados para la empresa solicitante. Porello, con base en los estudios anteriores, se determinó un diseño que pueda ser ajustado al tipo de negocio para aplicaciones móviles, el cual se define en la Tabla 9, en donde se indican los pasos a seguir y el desarrollo de cada uno de ellos.

Tabla 9.

Diseño para procedimiento de pruebas

	Pasos a seguir
1	Definir el tipo de proyecto a trabajar. Cronograma. Pruebas para realizar según evaluación y pruebas manuales como automatizadas para los dispositivos definidos.
2	Marcar las prioridades de las funcionalidades para cada área según escala deimportancia. 6.1. Funcionalidad 6.2. Rendimiento 6.3. Seguridad
3	Crear escenarios de prueba
4	Aseguramiento de calidad de usuarios y testes móviles de alto nivel. a. Los comentarios se analizan con una lista de comprobacióndefinida. b. Se compara la interfaz del usuario con aplicaciones de la misma naturaleza que ya se encuentran en el mercado.
5	Identificación y manejo de hallazgos por prioridad y verificar a qué área pertenece.
	a. Manejo de hallazgos en sitio interno para reportar las no conformidades.

Continuación Tabla no. 9

6	Reporte parcial a. Reporte de hallazgos encontrados b. Sumario c. Métricas analizadas Comparar resultados actuales con los esperados
7	Actualización y nueva validación de hallazgos. a. Revisar nuevamente el producto para verificar correcciones de hallazgos *b.* Prueba *End to End.* Prueba de regresión
8	Reporte final a. Estadísticas finales b. Hallazgos totales c. Resultado final a. Conclusiones y recomendaciones
9	Prueba Piloto o *Roll out*
10	Gestión de mejora continua
11	Cierre del proyecto

Nota: La tabla muestra cada uno de los pasos que deben seguirse para la realización de las pruebas requeridas. Fuente: elaboración propia (2016)

Tablas referenciales

Las tablas referenciales hacen mención sobre las mediciones, validaciones, escalas y todaslas implicaciones que sean necesarias para cada área del procedimiento.

Prioridades Funcionales

Las prioridades funcionales se utilizarán para verificar el manejo de las pruebas yseleccionar las prioritarias, permiten crear lista de escenarios a probar por cada área según sea necesario. Según lo investigado, la importancia de las prioridades en la Tabla 10 se basa en las métricas propuestas por la *Standar Measures of the Software*

-IEEE-, donde se indica queporcentaje debería ser el indicado para cada área, para obtener unas buenas prácticas de testeo.

Tabla 10.

Prioridades en escala de importancia según tipos y características

Nota: La tabla enlista los detalles de cada una de las prioridades funcionales y su grado de importancia. Fuente: IEEE Standar Measures of the Software (2006)

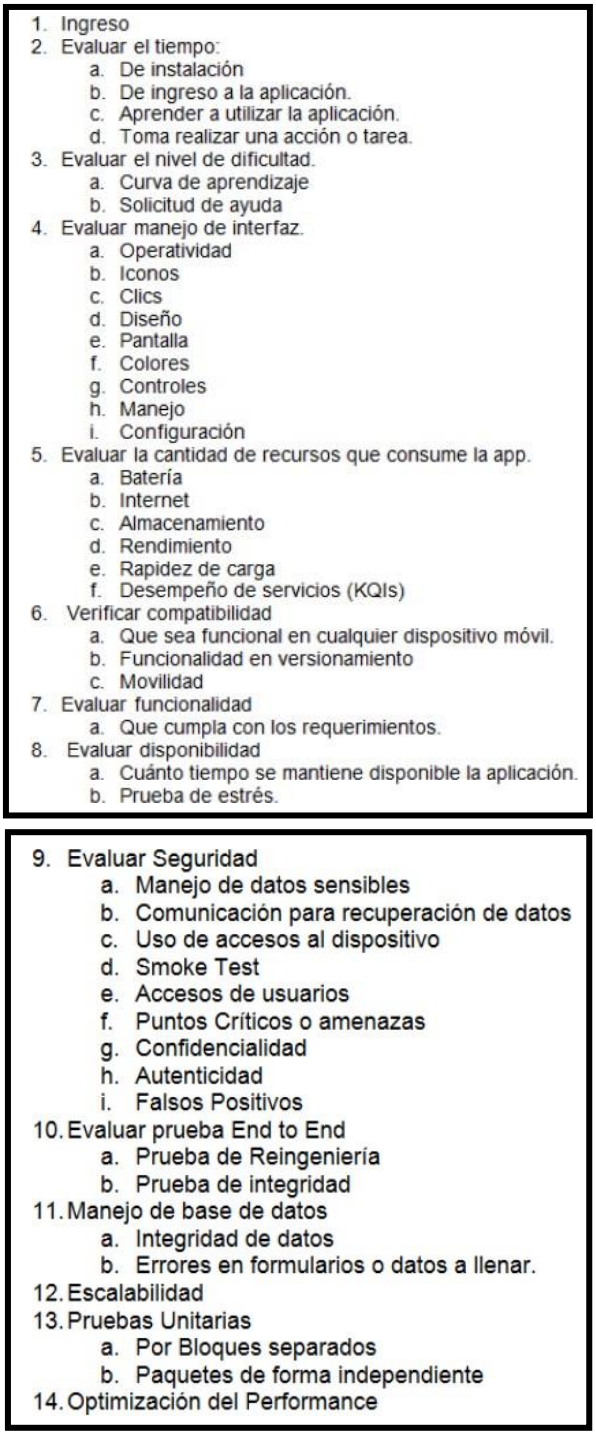
Implicación de cada área	Detalle	Importancia
Funcionalidad/Usabilidad	Interoperabilidad. Ciclo de vida de la aplicación. Gestión de redes y gestión de interfaz.	48%
Rendimiento/Portabilidad	Disponibilidad. Rendimiento de red. Integración del sistema y rendimiento *back-end.*	20%
Seguridad	Puntos críticos y amenazas. Confidencialidad, integridad. Autenticidad. Autorización. Disponibilidad, rechazo y falsos positivos.	32%

Escenarios de Prueba

Los escenarios de prueba son listas reutilizables con base en los casos posibles de prueba para las aplicaciones móviles. Estos deben ser acordes al requerimiento. La siguiente lista se deriva de la aplicación de varias técnicas de diseño de pruebas para las aplicaciones móviles más utilizadas mostradas en la Figura 18, la cual permite crear una lista de estándares que generalmente se debe cumplir al momento de realizar *testing*.

Figura 18.

Detalle de escenarios de pruebas



Nota: La gráfica muestra la lista de los estándares que debe cumplir al realizar un testing.

Fuente: elaboración propia (2016)

Lista de comprobación de usuario para pruebas no-funcionales.

La siguiente lista muestra una lista estándar de comprobación para pruebas no-funcionales, las cuales abarcan los comentarios necesarios y esenciales para conocer la funcionalidad de la aplicación, tal y como se muestra la lista de comprobación de la Tabla 11.

Dicha lista se aplicará cuando el cliente solicite que dentro de las pruebas a realizar le interese las pruebas sobre interfaz gráfica. La calificación será listada o metas determinando la cantidad de aciertos y desaciertos de la aplicación. Dichas pruebas deben ser de forma manual por ser subjetiva.

Tabla 11.

Comprobación para tester y usuarios no-funcional

ESCENARIOS DE PRUEBA	MÉTRICA
EFECTIVIDAD	- Tiempo - Tareas completadas al primer intento - Tiempo de ingreso a la aplicación - Tiempo para aprender
EFICIENCIA	- Tiempo en realizar una tarea - Numero de teclas presionadas por tarea - Tiempo transcurrido en cada pantalla - Eficiencia con usuarios expertos
SATISFACCIÓN	- Nivel de dificultad - Preferencias

Continuación Tabla no. 11

FACILIDAD	- Tiempo usado para terminar una tarea la primera vez - Cantidad de entrenamiento o ayuda - Curva de aprendizaje
MEMORIA	- Numero de pasos, clics usados para terminar una tarea después de no usar la aplicación por un periodo de tiempo.
ACCESIBILIDAD	- Manejo en diferentes dispositivos y sistemas operativos.
SEGURIDAD	- Control de usuarios (Cuando aplique). - Número de incidentes – Reglas de seguridad
PORTABILIDAD	- Nivel de configuración
CONTEXTO	- Grado de conectividad
USABILIDAD	- Rendimiento - Recursos consumidos - Porcentaje de batería usada - Estrés

Nota: La tabla presenta una lista de cada uno de los escenarios de prueba y las métricas que estos deben comprobar. Fuente: elaboración propia, (2016)

Comprobación Subjetiva

La lista de la Tabla 12 muestra una propuesta estándar de comprobación subjetiva para los usuarios. La forma de cotejo será calificar las respuestas con un listado con un método comparativo, para determinar el acierto como la satisfacción. Dichas pruebas deben ser medidasde forma manual debido a que son respuestas subjetivas.

Tabla 12.

Lista de comprobación subjetivas

Subjetivas	
Satisfacción	- Tiempo de salida - Tamaño del uso de la pantalla - Ayuda - Contenido - Interfaz

Nota: La tabla muestra los elementos que deben tomarse en cuenta en la comprobación subjetiva. Fuente: elaboración propia (2016)

Reporte de Incidencias

La siguiente lista determina un listado estándar de las incidencias que se deben validar en una aplicación móvil para promover un buen con el control de calidad. Donde se determina la prioridad de la incidencia, creando una lista de prioridades como la que se muestra en la Tabla 13, donde se tomaron las incidencias más frecuentes y las que deberían realizarse siempre. Dichas incidencias se agruparon por categorías de criterios de calidad, que se puedan ajustar a diferentes tipos de requerimientos. Y así asegurar que se cubrieron las funcionalidades básicas de la aplicación y permitan responder las siguientes interrogantes:

a. Funcionalidad: ¿Funcionará en el dispositivo, se adapta a las necesidades?

b. Usabilidad: ¿Qué tan fácil le es al usuario poder utilizarlo?

c. Rendimiento: ¿Funciona bien y si resiste a fallas en ciertas situaciones?

d. Base de Datos: ¿Es funcional la aplicación?

e. Seguridad: ¿Los datos se encuentran seguros?

f. Requeridas: ¿Hace lo que debe hacer?

Y según las prioridades, se determinan que tan crítica es la incidencia encontrada y quetan pronto debe ser solucionada.

- Urgente: debe ser solucionada de inmediato, es bloqueante.
- Alta: no cumple con requerimiento del cliente.
- Media: necesidades del cliente a satisfacer, puede crear vulnerabilidades.
- Baja: análisis subjetivo del cliente según preferencias.

Tabla 13.

Lista de reporte de incidencias

HALLAZGO	INCIDENCIA	PRIORIDAD
FUNCIONALIDAD	a. Control Alta b. Almacenamiento c. Opciones de manejo d. Interfaz e. Compatibilidad	ALTA
USABILIDAD	a. Interacción b. Estructura c. Optimización del Performance d. Formato e. Tiempos Disponibilidad	BAJA

Continuación Tabla no. 13

RENDIMIENTO	a. Consumo de batería b. Rapidez de carga c. Movilidad d. Desempeño de servicios e. Consumo de internet f. Soporte de carga g. Prueba de estrés	MEDIA
BASE DE DATOS	a. Accesibilidad b. Interoperabilidad de redes c. Versiona miento d. Integridad de datos e. Escalabilidad	MEDIA
SEGURIDAD	a. Vulnerabilidades b. Smoke Test I c. Integridad referencial d. Autenticidad e. Autorización f. Falsos positivos g. Datos Sensibles h. Confidencialidad	ALTA
REQUERIDAS	a. Integración b. Funcionalidad del requerimiento c. Funciones especificas d. Prueba End to End e. Pruebas unitarias (paquetes de forma independiente)	URGENTE

Nota: La tabla enlista cada uno de los hallazgos que se encuentren durante el testing, la incidencia que ellos tiene y su grado de prioridad. Fuente: elaboración propia (2016*)*

Reporte de Riesgo

Existe una lista universal genérica para cualquier sistema, entre las que son aplicablespara aplicaciones móviles se encuentran las siguientes:

a. Complejidad: complejidad de entendimiento.

b. Nuevo: todo lo que no se encuentra en documentación del producto.

c. Cambiado: cualquier elemento que haya sido manipulado o mejorado.

d. Dependencia: elementos cuya falla puede causar fallas en cascada al resto del sistema.

e. Sensibilidad: algo sensible a las fallas del resto del sistema.

f. Crítico: todo lo que al fallar pueda causar un daño de importancia.

g. Preciso: cualquier elemento que falla según el requerimiento.

h. Popular: algo que sería muy usado.

i. Estratégico: efecto de importancia en el negocio, como ser una funcionalidad que distingue al producto de los competidores.

j. De terceros: algo que fue desarrollado fuera del proyecto.

k. Distribuido: cualquier elemento que se debe trabajar juntos.

l. Con errores: algo que devolvió muchos errores.

m. Falla reciente: nueva inyección de error por cascada.

En la Tabla 14 se observa la categorización de riesgos, como ejemplo de la lista genérica.

Tabla 14.

Categorías de riesgos

RIESGO	DEFINICIÓN
INSTALACIÓN	Archivos temporales
	Archivos viejos después de instalación Archivos innecesarios Instalado en lugar incorrecto
EFECTOS SECUNDARIOS	Un archivo comparte con otra aplicación es afectado
	Archivos no pertenecientes se eliminan. Es afectado por la aplicación
HARDWARE	No es suficiente para la aplicación.
	Afecta pantalla
	No detecta si ya está instalada

Nota: La tabla enlista cada uno de los riesgos y su categorización dentro de la lista genérica. Fuente: elaboración propia (2016)

Reportes

Los reportes serán la prueba de pruebas y sumario general de los incidentes encontrados durante las mismas, en donde se detalla la información encontrada, con resultados estadísticos y manejo de versiones con iteraciones.

Reporte Parcial

El reporte parcial ayuda al seguimiento de defectos, mejorando la trazabilidad de las pruebas y un manejo más claro del proyecto, para dar a conocer los resultados parciales y determinar las tareas pendientes.

El reporte tendrá como base los siguientes apartados:

- Sumario del estatus del proyecto

El sumario del estatus del proyecto servirá para conocer los avances obtenidos en el proyecto, la información general, las iteraciones realizadas hasta el momento de entrega y si existe algún incidente encontrado relevante que deba ser tomado en cuenta.

- Objetivos cumplidos

Los objetivos cumplidos conforman la lista de objetivos que se han llevado a cabo del requerimiento base con respecto al producto, el estatus descriptivo de cada entrega y del avance en el que se encuentra el proyecto en ese momento y se indica el porcentaje de avance y métricas medidas hasta el momento. Un resumen de las pruebas que se han ejecutado hasta el momento. En los objetivos cumplidos se debe agregar la lista de incidentes encontrados parcialmente por prioridades, gráficas de incidentes por prioridades, estado de los incidentes, resumen general, entre otros.

- Actividades planeadas pendientes

Las actividades pendientes serán la lista de actividades que aún falta realizarse del proyecto, lista de las pruebas que aún falta por realizarse para cumplir con el proyecto y cerrarlo o avanzar a la siguiente iteración según sea el caso.

- Recepción de contenido parcial

Tabla que debe llenarse como formulario de entrega, donde los interesados se encuentran de acuerdo con la información proporcionada y como prueba de haber recibido el documento. En el Anexo A se encuentra el ejemplo de reporte parcial como ejemplo de implementación para tener una idea base de como poder llenarlo.

Reporte Final

El reporte final ayuda a conocer el estatus final del producto, con un detalle exhaustivo sobre los defectos encontrados, con métricas, historial, riesgos, y graficas sobre el proyecto en general. El reporte final tendrá como base los siguientes apartados:

- Sumario del proyecto

El sumario del proyecto servirá para dar a conocer la duración del proyecto, el número de iteraciones realizadas, las versiones y el listado de pruebas que fueron realizadas al producto. En cada iteración realizada se detallará un resumen de que se realizó en cada una y si existiera algún retraso en el tiempo planeado explicando a que se debió.

- Objetivos cumplidos

Los objetivos cumplidos conforman la lista de objetivos que se llevaron a cabo del requerimiento base con respecto al producto, el estatus descriptivo de cada entrega y como

se fue realizado el avance del proyecto. Además, se incluirá un resumen de las pruebas que se ejecutaron. En los objetivos cumplidos se debe agregar la lista de incidentes encontrados totales por prioridades, graficas de incidentes por prioridades, estado final de los incidentes, resumen general, entre otros.

- Métrica de hallazgos

Se mostrarán todos los hallazgos encontrados durante toda la ejecución de las pruebas.

- Riesgos

Se mostrarán todos los riesgos que se determinaron al finalizar la evaluación según la lista genérica de riesgos o todos aquellos se consideraron y no fueron resueltos. Se debe proponer soluciones que se consideren sean necesarias para la mitigación de estos.

- Recomendaciones

Se detallarán las recomendaciones que se crean necesarias según el producto evaluado de mejores, versiones o productos que se encuentren en el mercado para manejarlo como mejoras.

- Recepción de contenido

Tabla que debe llenarse como formulario de entrega, donde los interesados se encuentran de acuerdo con la información proporcionada y como prueba de haber recibido el documento.

- Historial de versiones

Cuadro que se implementará cuando se manejen diferentes versiones del producto o cuando se trabajen más de un producto con el mismo cliente.

- Carta de conformidad

Carta que debe ser entregada según formato de la institución, donde se valide que se acepta y se finaliza el proyecto para la empresa solicitante, en donde se certifica que la aplicación cumple con los requisitos de las pruebas realizadas y se detalla cada una y según los criterios de aceptación del centro de la Universidad, el cual da por aprobada y usable la aplicación. Además, se debe adjuntar un resumen general de los incidentes, riesgos y vulnerabilidades encontradas en total.

Esta carta debe ser firmada por el analista que se encargó de la revisión del proyecto y el gerente del proyecto que avale la Universidad.

- Anexo

Se adjuntarán como anexos las métricas que evalúan y determinan las herramientas utilizadas para el testeo como muestra que se realizaron las pruebas y estas fueron satisfactorias.

En el Anexo A se encuentra el ejemplo de reporte final como ejemplo de implementaciónpara tener una idea base de como poder llenarlo.

Costos

En la Tabla 15, se presenta un listado de costos como referencia para determinar cuánto se gastaría en certificaciones, desde la perspectiva para una compañía y de la perspectiva personal para certificar a los testers, para poder montarlo.

Tabla 15.

Costos

Costos de registro de marca Testing		
1	*Gartner*	$40,000
2	*D&B Credibility Corp.*	$8,000
Costos de las certificaciones		
3	ISO/IEC 27001 y 200000	$48,000
4	ISTQB, es renovable	$1,500
5	TMMi	$16,500
6	ISO 9001:2008, con más de 200 empleados	$22,800
7	ISQI, por persona	$1,400
8	IIST, por persona	$1,100
9	UX, depende de la universidad impartida	$3,200 a $17,000
10	CSTE, examen por persona	$420
11	ASTQB, Examen es renovable por persona	$250

Nota: La tabla hace una lista de los costos en los que se incurre en cada uno las certificaciones que deben realizarse. Fuente: elaboración propia (2016)

Pruebas Automatizadas

Se definió el proceso de automatización con la tecnología a utilizar y los detalles técnicos que deben seguirse para realizar las pruebas de manera efectiva. Las etapas que se deben llevar acabo se encuentran intrínsecamente relacionadas. Con cada una de las etapas involucradas se debe seguir el proceso para conseguir pruebas exitosas. En la Tabla 16 se indican los pasos a seguir y que incluye cada paso. En este módulo, los pasos deben ser ejecutados. Pero el proceso es flexible y se pueden realizar actividades extras, documentar procesos y realizar actividades opcionales en el proyecto.

Tabla 16.

Diseño para pruebas automatizadas

Pasos para seguir	
1	Seleccionar herramientas a utilizar. a. Se debe seleccionar las herramientas y crear un informe con cuales seutilizarán.
2	Seleccionar Pruebas. a. Casos de Prueba
3	Preparación de Automatización
4	Implementación de Pruebas Automatizadas
5	Ejecución de pruebas automatizadas a. Informe de incidentes
6	Análisis de errores detectados a. Análisis y reporte de incidencias
7	Mantenimiento de automatización
8	Crear detalle en reporte parcial a. Documentos de resultados de automatización
9	Verificar y actualizar documentación
10	Cierre del proyecto

Nota: En la tabla se presentan los pasos que deben seguirse en las pruebas automatizas del testing. Fuente: Elabración propia (2016)

VII. ANÁLISIS DE RESULTADOS

El estudio para establecer un centro de testeo de software móvil en la Universidad del Valle de Guatemala ha arrojado resultados prometedores y comprensivos. A continuación, se presenta un análisis detallado de los hallazgos:

Se ha conceptualizado el centro de testeo con un modelo lineal enfocado exclusivamente en servicios de testeo. Esta especialización permite una concentración de recursos y experiencia en un área específica, lo que puede resultar en un servicio de alta calidad. Además, el modelo contempla:

El potencial de expansión en donde identifica la posibilidad de ampliar los servicios a aplicaciones web y ofrecer consultoría *in situ*. Esta flexibilidad permite la adaptación a las cambiantes necesidades del mercado tecnológico.

La propuesta de paquetes preferenciales y contratos continuos, para establecer relaciones a largo plazo con los clientes. Esto podría asegurar un flujo de ingresos estable al crear así una estrategia de fidelización.

La naturaleza flexible y escalable del modelo que facilite su ajuste a las demandas del mercado. Esto es crucial en un sector tan dinámico como el de la tecnología móvil, pues es adaptable a las pruebas móviles.

La metodología propuesta se distingue por su enfoque integral y su adherencia a estándares internacionales. Se incorpora una combinación de métodos que incluyen pruebas diseñadas y exploratorias, lo que facilita una evaluación exhaustiva y profunda de las aplicaciones sometidas a análisis.

El marco operativo se fundamenta en una estructura que comprende precondiciones, procedimientos y prioridades funcionales, basadas en los estándares IEEE. Este enfoque brinda una base sólida y reconocida para las operaciones del centro.

La incorporación de pruebas automatizadas en la metodología refleja un compromiso con la eficiencia y la modernización de los procesos. La solidez de este enfoque metodológico, jun su alineación con estándares internacionales, contribuye significativamente a la credibilidady eficacia del servicio propuesto.

El sistema de documentación propuesto es exhaustivo y bien estructurado. Se han definido documentos específicos para cada etapa del proceso, lo que facilita el seguimiento y la gestión del proyecto. Asimismo, los reportes estructurados que permiten la implementación de informes parciales y finales con estructuras bien definidas mejoran la comunicación y la transparencia conlos clientes. Se incluye un sistema de categorización de riesgos, lo que demuestra un enfoque proactivo en la gestión de proyectos lo que facilita el seguimiento y mejora la comunicación con los clientes, con lo que aumentan la transparencia y confiabilidad del servicio.

El enfoque de evaluación propuesto es comprehensivo. Se propone una diversidad de herramientas y se establecen tablas referenciales y listas de comprobación para diversos aspectosde las aplicaciones móviles. Se han creado evaluaciones holísticas tanto objetivas como subjetivas, lo que permite una valoración más completa de las aplicaciones. El enfoque abarca tanto aspectos técnicos como de experiencia de usuario, proporcionando una evaluación integralde las aplicaciones.

La propuesta para el centro de testeo de software móvil en la Universidad del Valle de Guatemala demuestra ser comprehensiva y bien estructurada con un modelo de negocio flexible y escalable; una metodología de testeo robusta alineada con estándares internacionales; mientrasque los procesos exhaustivos de documentación y evaluación proporcionan una base sólida parael proyecto.

Sin embargo, el éxito y la sostenibilidad a largo plazo del centro dependen crucialmente de un análisis financiero más profundo y detallado. Este estudio financiero deberá incluir:

- Proyecciones de ingresos basadas en un análisis de mercado actualizado y específico para Guatemala y la región centroamericana.

- Estimaciones detalladas de costos operativos, incluyendo personal, equipamiento, licencias de software, y actualizaciones tecnológicas regulares.

- Análisis de punto de equilibrio para determinar la viabilidad económica del centro endiferentes escenarios de demanda.

- Evaluación de posibles fuentes de financiamiento que incluyen fondos universitarios, subvenciones gubernamentales y posibles asociaciones con empresas tecnológicas.

- Estrategias de *pricing* competitivas que consideren tanto la calidad del servicio ofrecido como las realidades económicas del mercado local.

- Plan de contingencia financiera para manejar posibles fluctuaciones en la demanda ocambios en el panorama tecnológico.

La realización de este estudio financiero exhaustivo no solo ayudará a determinar la viabilidad del proyecto, sino que también proporcionará una hoja de ruta para su implementación y crecimiento sostenible. Con esta base financiera sólida, y con el aprovechamiento de las fortalezas identificadas en el modelo propuesto, la implementación exitosa de este centro tiene el potencial de posicionar a la Universidad del Valle de Guatemala como un referente en el campodel testeo de software móvil en la región, lo que contribuiría significativamente al desarrollo tecnológico y económico de Guatemala y Centroamérica.

VIII. CONCLUSIONES

1. En Guatemala no existen empresas que brinden el servicio de *testing* para aplicaciones móviles únicamente, por lo que este trabajo académico representa un aporte para el país sobre la temática abordada.

2. Se logró simplificar, detallar y definir los diseños más utilizados para las pruebas funcionales de aplicaciones móviles. El estudio ha permitido identificar y proponer un modelo que asegura la calidad en los centros de testeo y proporciona una base sólida para la implementación de un centro de testeo en la Universidad del Valle de Guatemala.

3. Se definieron y crearon diseños de testeo específicos para la certificación de productos de software móvil. Estos diseños incorporan las mejores prácticas y estándares internacionales, lo que contribuye a la robustez y credibilidad del proceso de certificaciónpropuesto.

4. Se desarrolló una lista comprehensiva de procedimientos para obtener un funcionamiento óptimo de testeo. Esta lista proporciona una guía clara y estructurada que facilitará la implementación y operación de un centro de testeo funcional de móviles en la Universidad del Valle de Guatemala.

5. El estudio ha proporcionado una base sólida para la implementación de un centro de testeo, y abarca desde aspectos técnicos hasta consideraciones de negocio. Esto ofrece una visión integral que puede guiar la toma de decisiones y la planificación estratégica para el establecimiento del centro.

6. La investigación ha permitido identificar y proponer metodologías y herramientas actualizadas para el testeo de aplicaciones móviles, lo que posiciona al centro propuesto a la vanguardia de las prácticas de aseguramiento de calidad en software.

IX. RECOMENDACIONES

1. A la Universidad del Valle de Guatemala que promueva en los estudiantes de ingeniería la importancia del tema de certificación de calidad en *software*.

2. A la facultad de ingeniería, que incluya en el pénsum de la carrera de Ciencia de la Computación y Tecnologías de la Información cursos sobre diseño de herramientas parapruebas de *software*.

3. Que el presente trabajo de investigación sea una herramienta útil para los estudiantes de la Universidad del Valle de Guatemala, así como material didáctico para el personal docente.

X. BIBLIOGRAFÍA

Libros

Amanat, Chauddhry. 2004. *Gas Well Testing Handbook [Manual de pruebas de pozos de gas]*. 1ª ed. USA: Butterworth- Heinemann. 864 págs.

Ander-Egg Ezequiel. 2003. *Métodos y técnicas de investigación social*. 24ª ed. Argentina: Lumen. 425 págs.

Bush, Marilyn. 2005. *CMMI Assessments: Motivating Positive Change. [Evaluaciones CMMI: motivando el cambio positivo]*. 1ª ed. U.S.A.: Addison-Wesley. 432 págs.

Chrissis, Mary Beth. 2012. *CMMI para desarrollo. Guía para la integración de procesos y la mejora de productos*. 2ª ed. España: Editorial Universitaria Ramón Areces. 630 págs.

Eco, Umberto. 2009. *Cómo se hace una tesis*. 6ª ed. España: Gedisa. 240 págs.

Firtman, Maximiliano. 2012. *Jquery mobile: Aplicaciones html5 para móviles.* 1ª ed. España: Anaya. 272 págs.

García Sánchez, Álvaro. 2012. *Despliegue de aplicaciones Web*. 2ª ed. México: Garceta Grupo Editorial. 414 págs.

Glenford, J Myers. 2011. *The Art of Software Testing. [**El arte de las pruebas de software**]* 3ª ed. U.S.A.: Wiley. 253 págs.

Guerrero, Margarita. 2008. *Implementación del Sistema Integrado de Gestión en la Empresa de Diseño e Ingeniería de Cienfuegos*. 1ª ed. México: Universidad de Cienfuegos Carlos Rafael Rodríguez. 127 págs.

Hernández Roberto. 2006. *Metodología de la investigación*. 1ª ed. México: McGraw-Hill. Ciudad Juárez. 497 págs.

Konrad, Mike. 2009. *CMMI: Guía para la integración de procesos y la mejora de productos.* 2ª ed. U.S.A.: Addison-Wesley. 630 págs.

Levinson, William. 2002. *ISO 9000 En primera Línea*. 1ª ed. España: Acribia. 124 págs.

Martínez, David. 2010. *Aplicaciones Web: Un enfoque práctico*. 1ª ed. España: RA-MA S.A Editorial y Publicaciones. 296 págs.

Roma, Pastor, César San Juan. 2012. *Programación multimedia y dispositivos móviles*. 1ª ed. México: Garceta Grupo Editorial. 496 págs.

Polo Usaola, Macario. 2013. *Técnicas combinatorias y de mutación para Testing de Sistemas Software*. 1ª ed. España: Roma. 170 págs.

Sánchez Asenjo, Jorge. 2015. *Implantación de aplicaciones web.* 1ª ed. México: Garceta Grupo Editorial. 476 págs.

Santiago Nolasco, Jorge. 2015. *Desarrollo de aplicaciones móviles con Android*. 1ª ed. España: Roma. 512 págs.

Senlle Szodo, Andrés. 2001. *ISO 9000/2000 Calidad y excelencia todo lo que tiene que conocer para implantar y mantener un sistema de gestión de la calidad.* 1ª ed. México: Gestión 2000. 240 págs.

Servat, Alberto. 2005. *Calidad Metodología para documentar el ISO 9000 Versión 2000*. 1ª ed. Argentina: Ariel. 202 págs.

Campos, Julio Francisco. 2005. *Documentación e Implementación de Procedimientos para los Departamentos de Gerencia Administrativa Financiera, Simop e ISO 9000 Cgc-Bid de Cámara Guatemalteca de la Construcción Requeridos por el Sistema de Gestión de la Calidad basado en la Norma ISO 9001:2000.* Universidad de San Carlos de Guatemala. 216 págs.

Ciolli, María Elena. 2007. *Testing de migración de aplicaciones distribuidas a entornos Web.* Universidad Nacional de la Plata. 217 págs.

Mérida Méndez, Mario Rodrigo. 2007. *Aplicaciones y servicios de un Sistema de Telefonía Móvil GSM, sobre una plataforma de Transferencia de Datos GPRS, en la República de Guatemala.* Universidad de San Carlos de Guatemala. 172 págs.

Minera Baldizón, Manuel Alberto. 2011. *Definición de una Metodología para Evaluar el Grado de Seguridad de una Aplicación Web*. Universidad de San Carlos de Guatemala. 113 págs.

Pérez Vásquez, Luis Miguel. 2011. *Análisis de plataformas populares de Desarrollo de Aplicaciones para dispositivos móviles*. Universidad de San Carlos de Guatemala. 126 págs.

Pérez, Alejandro. 2008. *Especificación de un marco de pruebas asociado a Genexus con adaptación de funcionalidades de FIT1*. Universidad de la República, Montevideo. 267 págs.

Pérez, Beatriz. 2006. *Proceso de Testing Funcional Independiente*. Universidad de la República, Montevideo. 164 págs.

Pinto Agustín, Adriana Lizzeth. 2014. *Desarrollo de una aplicación móvil para*

asistencia médica para usuarios con VIH, Integrada con la plataforma de la Red Social Twitter Vinculada a Onusida Guatemala. Universidad de San Carlos de Guatemala. 131 págs.

Rodríguez Fernández, Ana Patricia. 2012. *Implementación de CMMI y la norma ISO spice 15504 para la mejora de procesos de las empresas de desarrollo de software guatemaltecas*. Universidad de San Carlos de Guatemala. 120 págs.

Rodríguez Sánchez, Sergio Adalberto. 2013. *Desarrollo e implementación de una Aplicación Web para el Control de la Evaluación de los Estudiantes del Área de EPS en la Facultad de Agronomía*. Universidad de San Carlos de Guatemala. 126 págs.

Sanz Moyano, Sergio. 2009 *implantación de CMMI en pequeñas empresas de desarrollo de software*. Universidad Politécnica de Valencia. 109 págs.

Toledo García, Denis Enrique. 2007. *Diseño del Sistema de Medición de la Satisfacción del Cliente para una Empresa Productora de Detergentes en Polvo de la Ciudad Capital de Guatemala, en el Marco de la Norma ISO 9000:2000.* Universidad de San Carlos de Guatemala. 171 págs.

Xitumul Ruiz, Delmi Marilú. 2007. *Normas ISO 9000 Vs. CMMI - SW como Estándar de Calidad en el desarrollo del Software y el proceso de obtención de la Certificación en cada estándar.* Universidad de San Carlos de Guatemala. 129 págs.

Sitio web

Cristiá, M. (1 de enero de 2009) Testing de Software: https://www.academia.edu/292809/Introducci%C3%B3n_Al_Testing_De_Software

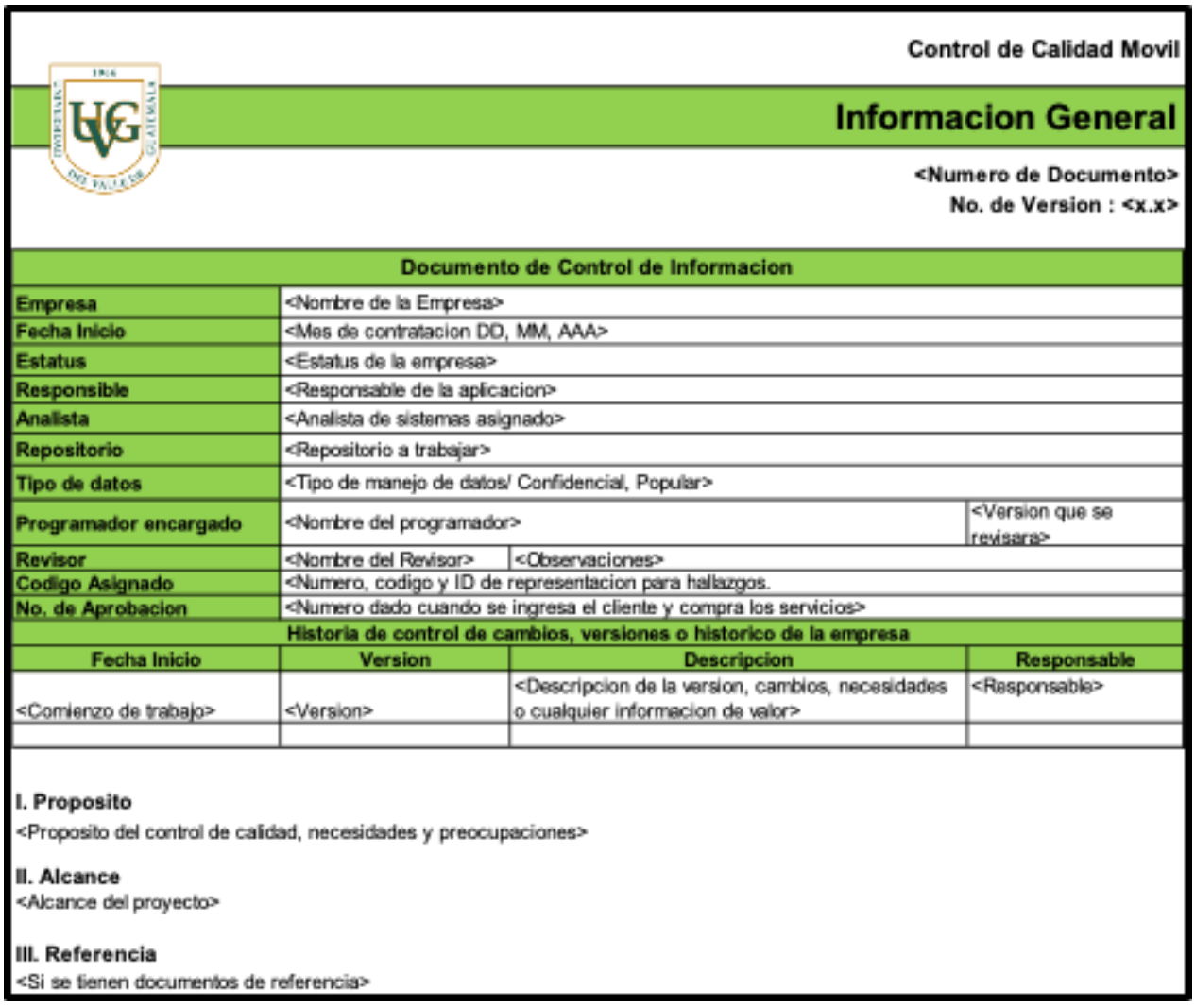
XI. ANEXOS

Anexo 1. A continuación se presentan una serie de ejemplos de la documentación que acompaña los distintos momentos en el proceso de *testing*:

En la Figura 19, se visualiza el ejemplo de encabezado de la información general del proyectoa evaluar por parte del revisor.

Figura 19.

Ejemplo encabezado de información general



Control de Calidad Movil

Informacion General

<Numero de Documento>
No. de Version : <x.x>

Documento de Control de Informacion			
Empresa	<Nombre de la Empresa>		
Fecha Inicio	<Mes de contratacion DD, MM, AAA>		
Estatus	<Estatus de la empresa>		
Responsible	<Responsable de la aplicacion>		
Analista	<Analista de sistemas asignado>		
Repositorio	<Repositorio a trabajar>		
Tipo de datos	<Tipo de manejo de datos/ Confidencial, Popular>		
Programador encargado	<Nombre del programador>		<Version que se revisara>
Revisor	<Nombre del Revisor>	<Observaciones>	
Codigo Asignado	<Numero, codigo y ID de representacion para hallazgos.		
No. de Aprobacion	<Numero dado cuando se ingresa el cliente y compra los servicios>		

Historia de control de cambios, versiones o historico de la empresa			
Fecha Inicio	Version	Descripcion	Responsable
<Comienzo de trabajo>	<Version>	<Descripcion de la version, cambios, necesidades o cualquier informacion de valor>	<Responsable>

I. Proposito
<Proposito del control de calidad, necesidades y preocupaciones>

II. Alcance
<Alcance del proyecto>

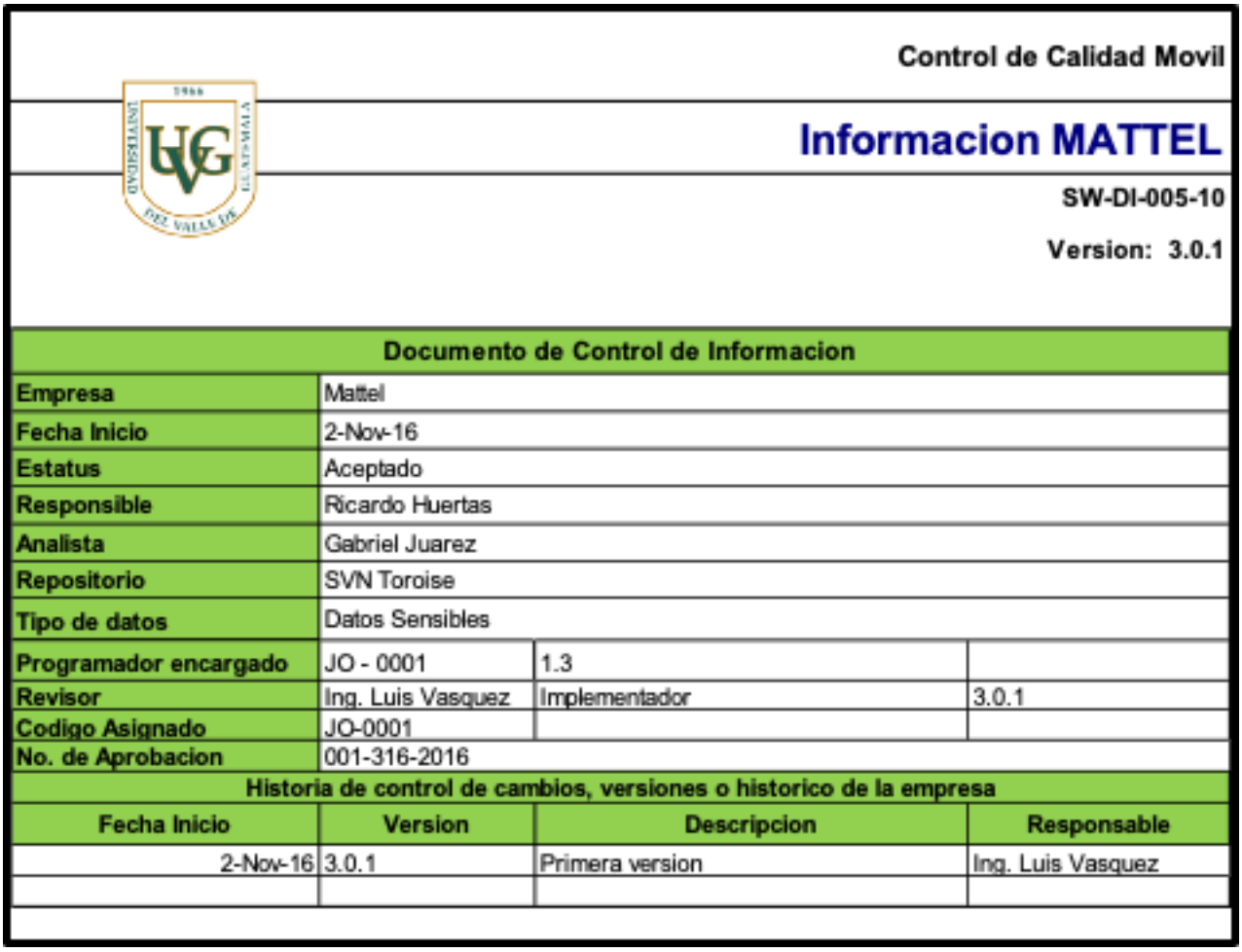
III. Referencia
<Si se tienen documentos de referencia>

Nota: La gráfica muestra la información general del proyecto que se evaluará por parte de quien realizará la revisión. Fuente: elaboración propia (2016)

En la figura 20, se visualiza el ejemplo de encabezado de la información general delproyecto a evaluar por parte de la empresa o dueño del producto a revisar.

Figura 20.

Ejemplo encabezado de información general con información de ejemplo



Control de Calidad Movil

Informacion MATTEL

SW-DI-005-10

Version: 3.0.1

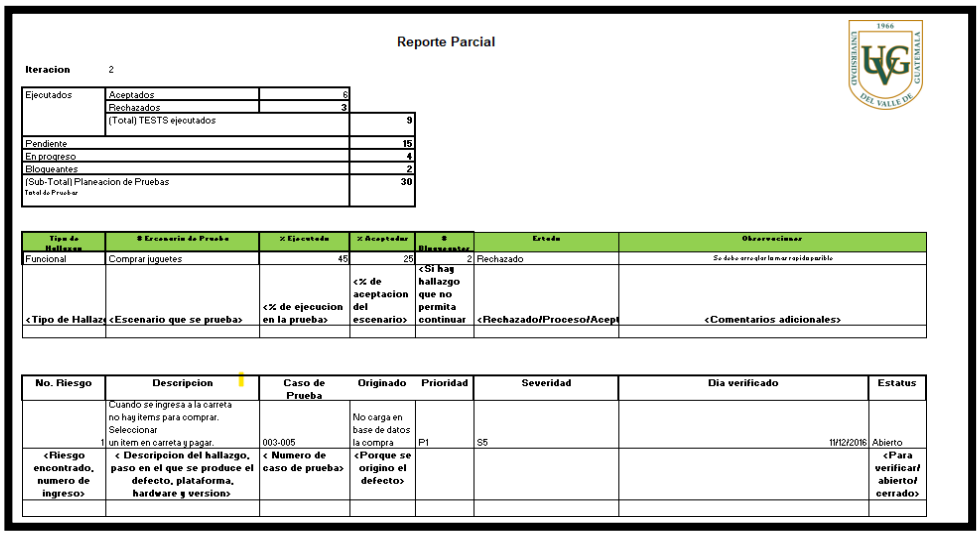
Documento de Control de Informacion			
Empresa	Mattel		
Fecha Inicio	2-Nov-16		
Estatus	Aceptado		
Responsible	Ricardo Huertas		
Analista	Gabriel Juarez		
Repositorio	SVN Toroise		
Tipo de datos	Datos Sensibles		
Programador encargado	JO - 0001	1.3	
Revisor	Ing. Luis Vasquez	Implementador	3.0.1
Codigo Asignado	JO-0001		
No. de Aprobacion	001-316-2016		
Historia de control de cambios, versiones o historico de la empresa			
Fecha Inicio	Version	Descripcion	Responsable
2-Nov-16	3.0.1	Primera version	Ing. Luis Vasquez

Nota: La gráfica muestra la información general del proyecto que se evaluará por parte de quien solicita la revisión. Fuente: elaboración propia (2016)

En la Figura 21, se visualiza el ejemplo de encabezado de la información general delproyecto a evaluar por parte de la empresa o dueño del producto a revisar.

Figura 21.

Ejemplo de información del reporte parcial



Reporte Parcial

Iteracion 2

Ejecutados	Aceptados	6	
	Rechazados	3	
	(Total) TESTS ejecutados		9
Pendiente			15
En progreso			4
Bloqueantes			2
(Sub-Total) Planeacion de Pruebas			30
Total de Pruebas			

Tipo de Hallazgo	# Escenario de Prueba	% Ejecutado	% Aceptados	# Bloqueantes	Estado	Observaciones
Funcional	Comprar juguetes	45	25	2	Rechazado	Se debe arreglar lo mas rapido posible
<Tipo de Hallaz	<Escenario que se prueba>	<% de ejecucion en la prueba>	<% de aceptacion del escenario>	<Si hay hallazgo que no permita continuar	<Rechazado/Proceso/Acept	<Comentarios adicionales>

No. Riesgo	Descripcion	Caso de Prueba	Originado	Prioridad	Severidad	Dia verificado	Estatus
1	Cuando se ingresa a la carreta no hay items para comprar. Seleccionar un item en carreta y pagar.	003-005	No carga en base de datos la compra	P1	S5	11/12/2016	Abierto
<Riesgo encontrado, numero de ingreso>	< Descripcion del hallazgo, paso en el que se produce el defecto, plataforma, hardware y version>	< Numero de caso de prueba>	<Porque se origino el defecto>				<Para verificar/ abierto/ cerrado>

Nota: La gráfica muestra la información inicial que debe contener un informe parcial.
Fuente: elaboración propia (2016)

En la Figura 22, se visualiza la clasificación según la severidad y la prioridad para colocaren el reporte a completar.

Figura 22.

Clasificación de severidad y prioridad para el reporte parcial

Severidad	
S1	Baja
S2	Media
S3	Media - Alta
S4	Alta
S5	Urgente

Prioridad	
P1	Principal
P2	Alta
P3	Media - Alta
P4	Media
P5	Baja

Nota: La gráfica enlista la clasificación y el grado de severidad que representa.
Fuente: elaboración propia (2016)

En la Figura 23, se visualiza el listado de hallazgos, clasificados por tipo y por prioridad seleccionados previamente**.**

Figura 23.

Listado de tipos de hallazgos y prioridad para completar el reporte

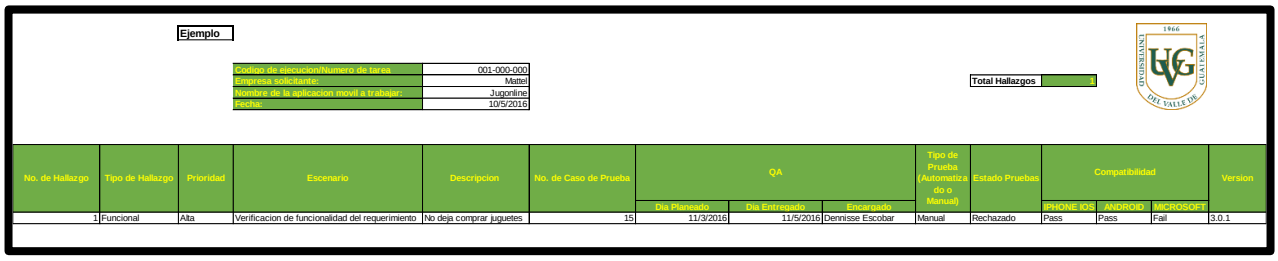
Tipo de Hallazgo	Hallazgos	Prioridad
Funcionalidad	Control	**Critico**
	Almacenamiento	
	Opciones de manejo	
	Interfaz	
	Compatibilidad	
Usabilidad	Interaccion	**Baja**
	Estructura	
	Optimizacion del Performance	
	Formato	
	Tiempos	
	Disponibilidad	
	Buena experiencia	
Rendimiento	Consumo de bateria	**Media**
	Rapidez de carga	
	Movilidad	
	Desempeno de servicios	
	Consumo de internet	
	Soporte de carga	
	Prueba de estres	
Base de datos	Accesibilidad	**Media**
	Interoperabilidad de redes	
	Versionamiento	
	Integridad de datos	
	Escalabilidad	
Seguridad	Vulnerabilidades	**Alta**
	Smoke Test	
	Integridad referencial	
	Autenticidad	
	Autorizacion	
	Falsos positivos	
	Datos Sensibles	
	Confidencialidad	
Requeridas	Integracion	**Bloqueante**
	Funcionalidad del requerimiento	
	Funciones especificas	
	Prueba End to End	
	Pruebas unitarias (paquetes de forma independiente)	

Nota: El gráfico muestra el tipo de hallazgos que pueden encontrarse y su grado de prioridad. Fuente: elaboración propia (2016)

En la Figura 24, se visualiza el ejemplo de reporte de hallazgos con la aplicación de filtrosde tipo de hallazgo y criticidad.

Figura 24.

Ejemplo de reporte de hallazgo



Ejemplo

Codigo de ejecucion/Numero de tarea	001-000-000
Empresa solicitante:	Mattel
Nombre de la aplicacion movil a trabajar:	Jugonline
Fecha:	10/5/2016

Total Hallazgos 1

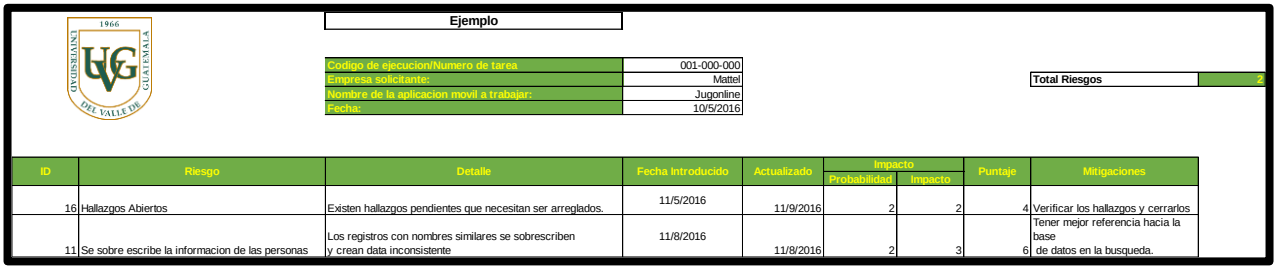
No. de Hallazgo	Tipo de Hallazgo	Prioridad	Escenario	Descripcion	No. de Caso de Prueba	QA			Tipo de Prueba (Automatizado o Manual)	Estado Pruebas	Compatibilidad			Version
						Dia Planeado	Dia Entregado	Encargado			IPHONE IOS	ANDROID	MICROSOFT	
1	Funcional	Alta	Verificacion de funcionalidad del requerimiento	No deja comprar juguetes	15	11/3/2016	11/5/2016	Dennisse Escobar	Manual	Rechazado	Pass	Pass	Fail	3.0.1

Nota: La gráfica muestra el reporte de tipo de hallazgos y su grado de importancia crítica. Fuente: elaboración propia (2016)

En la Figura 25, se visualiza el ejemplo de reporte de riesgos encontrados de formadetallada.

Figura 25.

Ejemplo de reporte de riesgos



Ejemplo

Codigo de ejecucion/Numero de tarea	001-000-000
Empresa solicitante:	Mattel
Nombre de la aplicacion movil a trabajar:	Jugonline
Fecha:	10/5/2016

Total Riesgos 2

ID	Riesgo	Detalle	Fecha Introducido	Actualizado	Impacto		Puntaje	Mitigaciones
					Probabilidad	Impacto		
16	Hallazgos Abiertos	Existen hallazgos pendientes que necesitan ser arreglados.	11/5/2016	11/9/2016	2	2	4	Verificar los hallazgos y cerrarlos
11	Se sobre escribe la informacion de las personas	Los registros con nombres similares se sobrescriben y crean data inconsistente	11/8/2016	11/8/2016	2	3	6	Tener mejor referencia hacia la base de datos en la busqueda.

Nota: La gráfica muestra el reporte detallado de riesgos que se encuentren. Fuente: elaboración propia (2016)

Ejemplo de reporte parcial

En la Figura 26, se muestra un ejemplo de reporte parcial en donde se pueda crear el detalledel proyecto según las bases especificada en la propuesta.

Figura 26.

Documento completo de información del cliente del reporte parcial

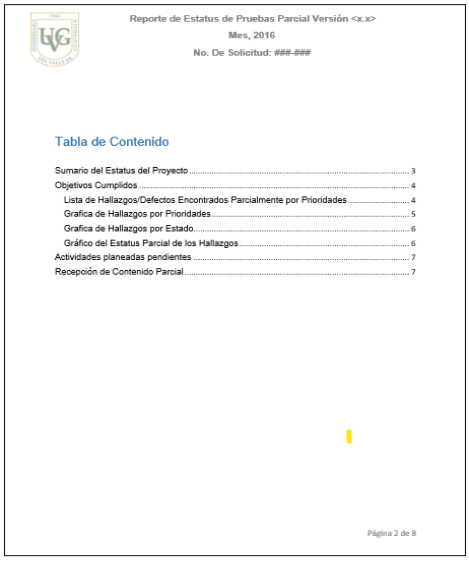
Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Aseguramiento de Calidad para Móviles

Información del Cliente {Nombre del Cliente}	
ID del Proyecto	#####-######
Documentos de Referencia	Código asignado para identificar el cliente.
Numero de Iteraciones	No. de Iteraciones de las cuales se reportan
Título de la aplicación verificada	Nombre de la aplicación evaluada.
Fecha	Fecha de entrega del documento

Analista de sistemas
Abalado por: Universidad del Valle de Guatemala
Dirección: 18 Avenida 11-95 zona 15, Vista Hermosa 3, Guatemala 01015

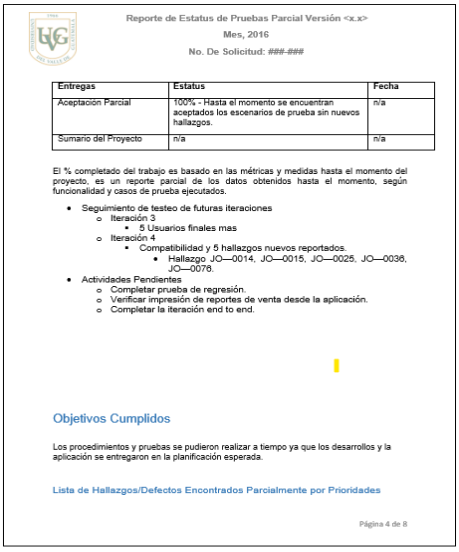
Página 1 de 8



Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Tabla de Contenido

Página 2 de 8



Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Entregas	Estatus	Fecha
Aceptación Parcial	100% - Hasta el momento se encuentran aceptados los escenarios de prueba sin nuevos hallazgos.	n/a
Sumario del Proyecto	n/a	n/a

El % completado del trabajo es basado en las métricas y medidas hasta el momento del proyecto, es un reporte parcial de los datos obtenidos hasta el momento, según funcionalidad y casos de prueba ejecutados.

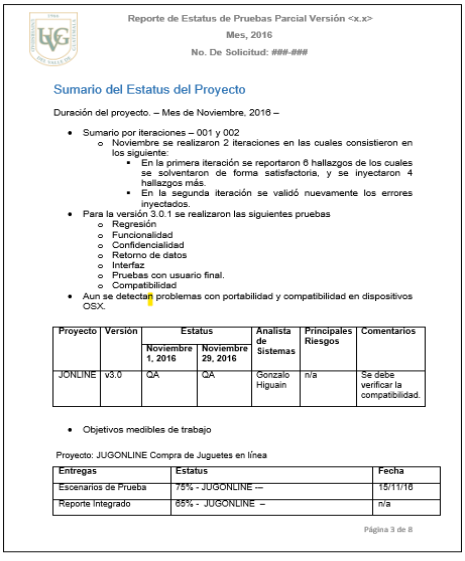
- Seguimiento de testeo de futuras iteraciones
  - Iteración 3
    - 5 Usuarios finales mas
  - Iteración 4
    - Compatibilidad y 5 hallazgos nuevos reportados.
      - Hallazgo JO—0014, JO—0015, JO—0025, JO—0036, JO—0076.
- Actividades Pendientes
  - Completar prueba de regresión.
  - Verificar impresión de reportes de venta desde la aplicación.
  - Completar la iteración end to end.

Objetivos Cumplidos

Los procedimientos y pruebas se pudieron realizar a tiempo ya que los desarrollos y la aplicación se entregaron en la planificación esperada.

Lista de Hallazgos/Defectos Encontrados Parcialmente por Prioridades

Página 4 de 8



Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Sumario del Estatus del Proyecto

Duración del proyecto. – Mes de Noviembre, 2016 –

- Sumario por iteraciones – 001 y 002.
  - Noviembre se realizaron 2 iteraciones en las cuales consistieron en los siguiente:
    - En la primera iteración se reportaron 6 hallazgos de los cuales se solventaron de forma satisfactoria, y se inyectaron 4 hallazgos más.
    - En la segunda iteración se validó nuevamente los errores inyectados.
- Para la versión 3.0.1 se realizaron las siguientes pruebas
  - Regresión
  - Funcionalidad
  - Confidencialidad
  - Retorno de datos
  - Interfaz
  - Pruebas con usuario final.
  - Compatibilidad
- Aun se detectan problemas con portabilidad y compatibilidad en dispositivos OSX.

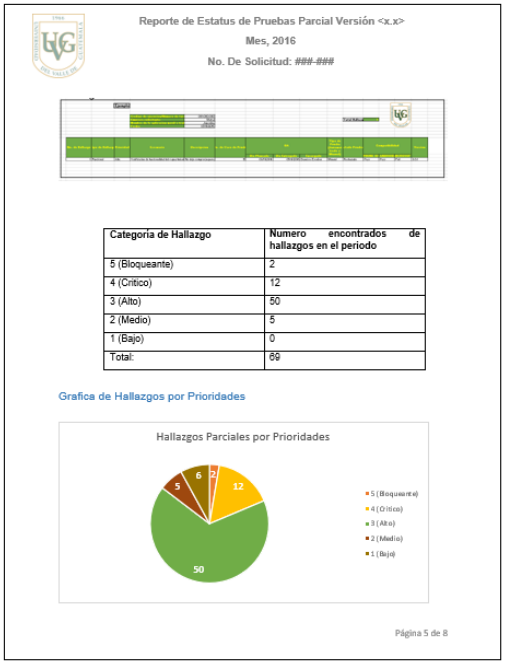
Proyecto	Versión	Estatus		Analista de Sistemas	Principales Riesgos	Comentarios
		Noviembre 1, 2016	Noviembre 29, 2016			
JONLINE	v3.0	QA	QA	Gonzalo Higuain	n/a	Se debe verificar la compatibilidad.

- Objetivos medibles de trabajo

Proyecto: JUGONLINE Compra de Juguetes en línea

Entregas	Estatus	Fecha
Escenarios de Prueba	75% - JUGONLINE —	15/11/16
Reporte Integrado	65% - JUGONLINE –	n/a

Página 3 de 8

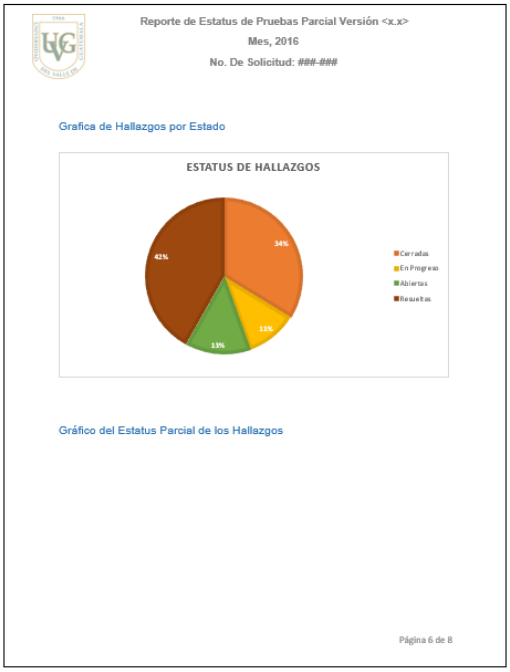


Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Categoría de Hallazgo	Numero encontrados de hallazgos en el periodo
5 (Bloqueante)	2
4 (Critico)	12
3 (Alto)	50
2 (Medio)	5
1 (Bajo)	0
Total:	69

Grafica de Hallazgos por Prioridades

Página 5 de 8

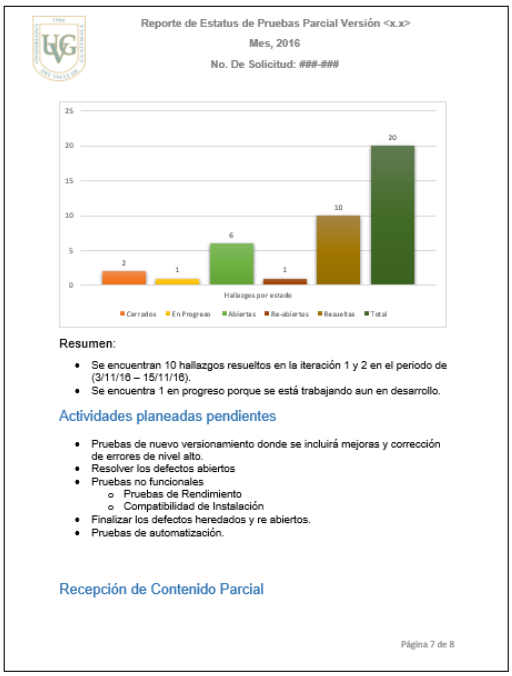


Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Grafica de Hallazgos por Estado

Gráfico del Estatus Parcial de los Hallazgos

Página 6 de 8



Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Resumen:

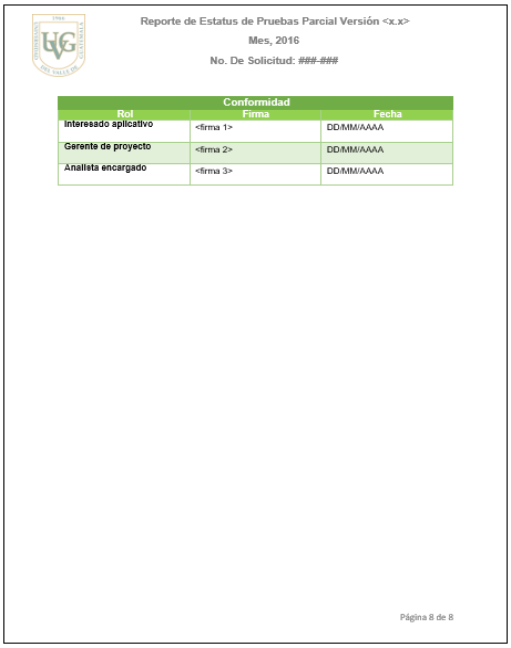
- Se encuentran 10 hallazgos resueltos en la iteración 1 y 2 en el periodo de (3/11/16 – 15/11/16).
- Se encuentra 1 en progreso porque se está trabajando aun en desarrollo.

Actividades planeadas pendientes

- Pruebas de nuevo versionamiento donde se incluirá mejoras y corrección de errores de nivel alto.
- Resolver los defectos abiertos
- Pruebas no funcionales
  - Pruebas de Rendimiento
  - Compatibilidad de Instalación
- Finalizar los defectos heredados y re abiertos.
- Pruebas de automatización.

Recepción de Contenido Parcial

Página 7 de 8



Reporte de Estatus de Pruebas Parcial Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Conformidad		
Rol	Firma	Fecha
Interesado aplicativo	<firma 1>	DD/MM/AAAA
Gerente de proyecto	<firma 2>	DD/MM/AAAA
Analista encargado	<firma 3>	DD/MM/AAAA

Página 8 de 8

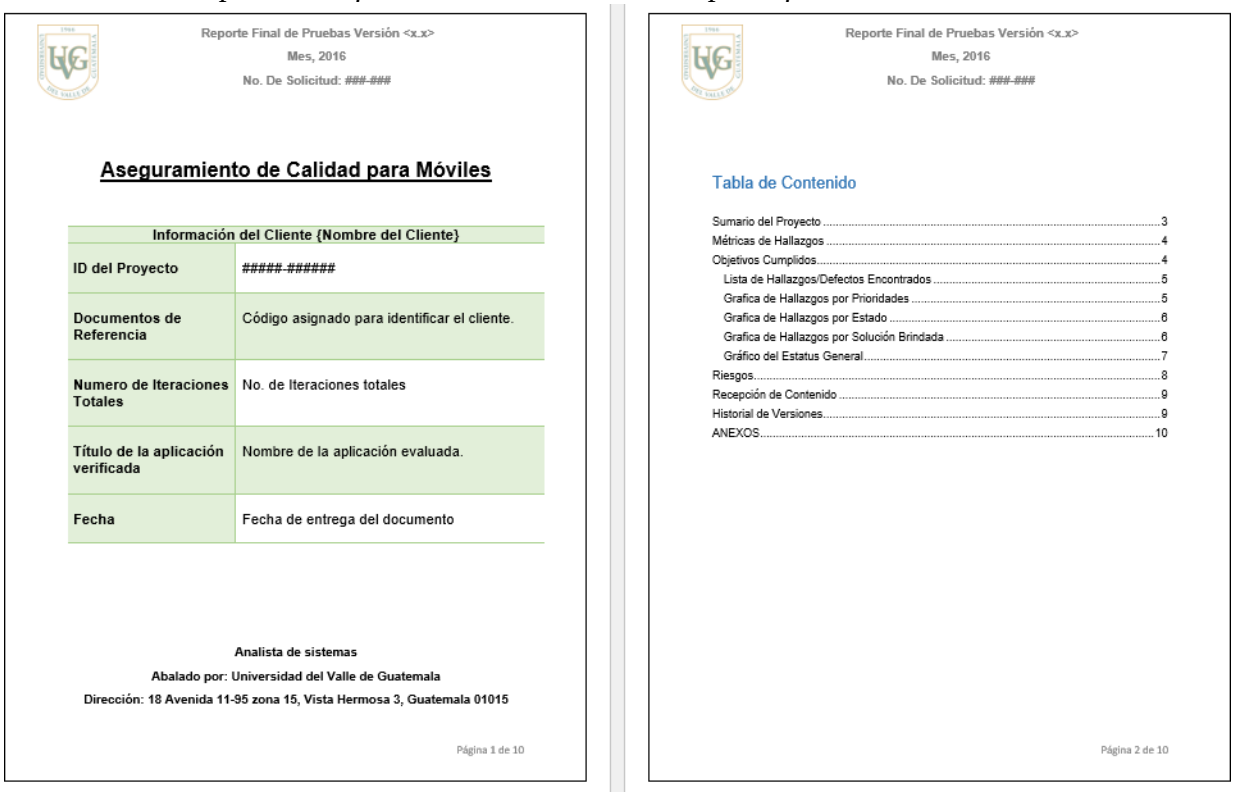
Nota: El gráfico es un ejemplo de un documento completo del informe parcial que se presenta al cliente. Fuente: elaboración propia (2016)

Ejemplo de reporte final

En la figura 27, se observa un ejemplo de reporte final en donde se pueda crear un detalle,resumen, métricas del proyecto según las bases especificada en la propuesta.

Figura *27.*

Documento completo de información del cliente del reporte final



Reporte Final de Pruebas Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Aseguramiento de Calidad para Móviles

Información del Cliente {Nombre del Cliente}	
ID del Proyecto	#####-######
Documentos de Referencia	Código asignado para identificar el cliente.
Numero de Iteraciones Totales	No. de Iteraciones totales
Título de la aplicación verificada	Nombre de la aplicación evaluada.
Fecha	Fecha de entrega del documento

Analista de sistemas
Abalado por: Universidad del Valle de Guatemala
Dirección: 18 Avenida 11-95 zona 15, Vista Hermosa 3, Guatemala 01015

Página 1 de 10

Reporte Final de Pruebas Versión <x.x>
Mes, 2016
No. De Solicitud: ###-###

Tabla de Contenido

Página 2 de 10

Sumario del Proyecto

Duración del proyecto: – Inicio Mes de Noviembre, 2016 – Finalización Mes de Marzo, 2017 -

- Sumario por iteraciones – 001 y 002
  - Noviembre se realizaron 2 iteraciones en las cuales consistieron en los siguiente:
    - En la primera iteración se reportaron 6 hallazgos de los cuales se solventaron de forma satisfactoria, y se inyectaron 4 hallazgos más.
    - En la segunda iteración se validó nuevamente los errores inyectados.
- Para la versión 3.0.1 se realizaron las siguientes pruebas
  - Regresión
  - Funcionalidad
  - Confidencialidad
  - Retorno de datos
  - Interfaz
  - Pruebas con usuario final.
  - Compatibilidad
  - Seguridad
  - Usabilidad
  - Base de Datos

Proyecto	Versión	Estatus		Analista de Sistemas	Principales Riesgos	Comentarios
		Noviembre 1, 2016	Marzo 29, 2017			
JONLINE	v3.0	QA	QA	Gonzalo Higuain	n/a	Se finalizó satisfactoriamente las pruebas.

- Objetivos medibles de trabajo

Proyecto: JUGONLINE Compra de Juguetes en línea

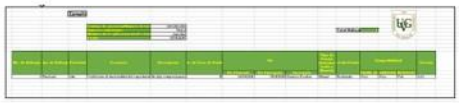
Entregas	Estatus	Fecha
Escenarios de Prueba	100% - JUGONLINE —	15/11/16
Reporte Integrado	100% - JUGONLINE –	n/a
Aceptación	100% -	n/a
Sumario del Proyecto	100%	n/a

Métricas de Hallazgos

Objetivos Cumplidos

Los procedimientos y pruebas se pudieron realizar a tiempo ya que los desarrollos y la aplicación se entregaron en la planificación esperada. Se finalizó satisfactoriamente las pruebas móviles para dicho proyecto.

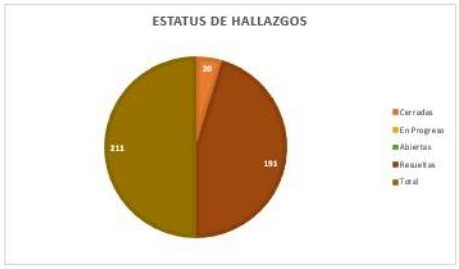
Lista de Hallazgos/Defectos Encontrados



Categoría de Hallazgo	Numero encontrados de hallazgos en el periodo
5 (Bloqueante)	20
4 (Critico)	35
3 (Alto)	57
2 (Medio)	34
1 (Bajo)	65
Total:	211

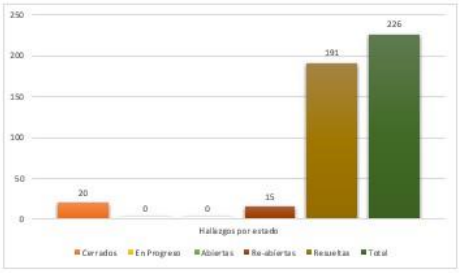
Grafica de Hallazgos por Prioridades

Grafica de Hallazgos por Estado



Grafica de Hallazgos por Solución Brindada

Gráfico del Estatus General



Resumen:

- Se encuentran solucionaron todas las incidencias, cerradas las que no procedían, resueltas en su totalidad.

Riesgos

- Dependencias de datos, Datos de estado de curación, recuperaciones de datos existentes de ensayo de impacto.
- Se han publicado fases que no fueron analizadas por control de calidad lo que podría dejar algunas funciones sin haberse probado completas.
- Impactos de tiempo de inactividad dejando logueado al usuario.
- Problemas de validación no se recuperan los datos tal y como se ha introducido.
- Existe el riesgo de una fuga de información personal o envío de correo basura ya que se utilizan correos de personas reales.
  - Registro de usuarios inválidos e inexistentes.
  - Permite hacer el login.
- Se llena de nuevos todos los campos al ingresar un usuario existente.

Recomendaciones

- Se debe realizar un nuevo versionamiento para dar de baja ciertos riesgos que ponen en riesgo la seguridad de los datos del cliente.
- Realizar pruebas funcionales a todo el sistema no solo las nuevas inyecciones y módulos para minimizar los riesgos y vulnerabilidades.

Recepción de Contenido

Conformidad		
Rol	Firma	Fecha
Interesado aplicativo	<firma 1>	DD/MM/AAAA
Gerente de proyecto	<firma 2>	DD/MM/AAAA
Analista encargado	<firma 3>	DD/MM/AAAA

Historial de Versiones

Versión #	Implementador	Día de Revisión	Aprobado por:	Día Aprobación	Motivo
1.0	*<Nombre del implementador>*	*DD/MM/AAAA*	*<Nombre quien aprobó>*	*DD/MM/AAAA*	*<Motivos de cambio y nuevo versionamiento>*

Carta de Conformidad

Por este medio se da por aceptado y por finalizado el requerimiento numero #####, para la empresa $$$$$, en la cual se certifica que la aplicación cumple con los requisitos de pruebas funcionales, usabilidad y de seguridad según los criterios de aceptación del centro de testeo móvil de la Universidad del Valle de Guatemala. Teniendo como resultado:

Hallazgos: ##

Riesgos: ###

Vulnerabilidades: ###

_____

Firma Analista del Sistema

ANEXOS

Se pueden adjuntar como anexos las métricas que evalúan y determinan las herramientas utilizadas para el testeo como muestra que se realizaron las pruebas y estas fueron satisfactorias.

Nota: El gráfico es un ejemplo de un documento completo del informe final que se presenta al cliente Fuente: elaboración propia (2016)

XII. GLOSARIO

Aplicación	Es una solicitud. En programa de *software*, servicio o sitio web interactivo que se ejecuta en un dispositivo.
Automatización	Pruebas que pueden ser implementadas que se ejecuten de forma automática.
Calidad	Conjunto de propiedades inherentes a una cosa que permite caracterizarla y valorarla con respecto a las restantes de su especie.
Casos de Prueba	Unidad de verificación sobre un *software*.
Capability Maturity Model (CMM)	Marco de trabajo de cinco niveles que describen los elementos clave de un proceso *software* efectivo. El modelo CMM cubre las mejores prácticas para la planificación, ingeniería y gestión del desarrollo y el mantenimiento del *software*.
Capability Maturity Model Integration (CMMI)	Marco de trabajo que describe los elementos clave para un proceso de desarrollo y mantenimiento efectivo de un producto. El modelo CMMI cubre las mejores prácticas para la planificación, la ingeniería y gestión del desarrollo y el mantenimiento del producto.

Código	Instrucciones de ordenador y definiciones de datos expresados en un lenguaje de programación o en una forma de salida generada por un ensamblador, compilador u otro traductor.
Configuración	Composición de un componente o de un sistema definido como el número, naturaleza e interconexiones de las partes que lo constituyen
Conjunto de pruebas base	Conjunto de casos de prueba derivados de la estructura interna de un componente o especificación para asegurar que será alcanzado el 100% de un criterio de cobertura especificado.
Criterios de aceptación	Los criterios de salida que un componente o sistema debe satisfacer para ser aceptado por un usuario, cliente u otra entidad autorizada.
Emulador	Es un programa de *software* que proporciona una plataforma virtual en la que se ejecute una aplicación móvil.
Equipo Ágil	Son los expertos en áreas técnicas y dominio del negocio, que trabajan orientados al mismo fin en pequeños ciclos de tiempo llamados iteraciones

Funcionalidad	Funcionamiento de un producto de *software* existente.
ISO	Es la Organización Internacional para la Estandarización, que regula una serie de normas para fabricación, comercio y comunicación, en todas las ramas industriales.
ISO 9000	Es un conjunto de normas sobre calidad y gestión de calidad, establecidas por la Organización Internacional de Normalización (ISO). Se pueden aplicar en cualquier tipo de organización o actividad orientada a la producción de bienes o servicios.
Manual *testing*	Es una prueba de funcionamiento en el que una persona entra en las entradas de prueba, observa que los resultados reales y los registros de veredictos.
Metodología Ágil	Proceso de desarrollo de *software* que tiene como objetivo construir aplicaciones de manera rápida, teniendo como base las comunicaciones cara a cara en vez de documentación.
Mobile app	Es un programa de *software* que se ejecuta en un dispositivo móvil.

Mobile device	Es un pequeño ordenador portátil con capacidad de comunicación inalámbrica incorporada, por ejemplo, un teléfono celular, smartphone o tablet PC. Por lo general, de uso manual, pero puede ser embebido en otros sistemas.
Proceso	Es una serie de pasos que involucran actividades, restricciones y recursos que producen una determinada salida esperada
Proceso de desarrollo de *software*	Suele denominarse ciclo de vida del *software* y describe la vida de un producto de *software* desde su concepción hasta su entrega, utilización y mantenimiento.
Prueba	Es una actividad realizada para evaluar, validar la funcionalidad y la calidad del producto para mejorarla, identificando defectos y problemas.
Prueba de Humo/Smoke *Test*	Testing rápido que se realiza sobre aspectos funcionales para asegurar la funcionalidad básica del *software*, que se encuentre estable.
Prueba de *software*	Es la verificación dinámica del comportamiento de un programa contra el comportamiento esperado, usando un conjunto finito de casos de prueba, seleccionados de manera adecuada desde el dominio infinito de ejecución.

Resultado real	El comportamiento producido/observado cuando un componente o sistema es probado.
Requerimiento	Descripción de la funcionalidad de un sistema
Software	Conjunto de programas y rutinas que permiten a la computadora realizar determinadas tareas.
***Software* a medida**	*Software* desarrollado específicamente para un conjunto de usuarios o clientes. Lo opuesto es el *software* de distribución masiva.
Tester	Persona que puede desempeñar varios roles en un proyecto de *software*, tiene habilidades técnicas y destrezas tanto como para el desarrollo como la ejecución de las pruebas
Test case.	Es un conjunto de valores para las variables de una acción y un resultado esperado
Test model.	Es una descripción estructurada y simplificada de la AUT que hace que sea fácil de producir casos de prueba
Test plan.	Es una colección documentada de excursiones, casos de uso, y variantes.

Testing	Son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada.
Validación	Proceso de evaluación de un sistema o componente durante o al final del proceso de desarrollo para determinar cuándo se satisfacen los requerimientos especificados.
Valor límite	Valor de entrada o de salida que se encuentra en la frontera de una partición de equivalencia o a la mínima distancia incremental a cualquier lado de la frontera, por ejemplo, el valor mínimo o máximo de un rango
Verificación	Proceso de evaluación de un sistema o componente para determinar si un producto de una determinada fase de desarrollo satisface las condiciones impuestas al inicio de la fase.