PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

# UN MODELO DE MEJORAMIENTO DEL PROCESO DE TESTING EN PEQUEÑAS EMPRESAS

**FRANCISCO JAVIER OYARCE VALDERRAMA**

INFORME FINAL DEL PROYECTO
PARA OPTAR AL TÍTULO PROFESIONAL DE

INGENIERO CIVIL EN INFORMÁTICA

SEPTIEMBRE 2011

Pontifica Universidad Católica de Valparaíso
Facultad de Ingeniería
Escuela de Ingeniería Informática

# UN MODELO DE MEJORAMIENTO DEL PROCESO DE TESTING EN PEQUEÑAS EMPRESAS

**FRANCISCO JAVIER OYARCE VALDERRAMA**

Profesor Guía: **Rodolfo Humberto Villarroel Acevedo**

Profesor Co-referente: **Silvana Roncagliolo de La Horra**

Carrera: **Ingeniería Civil en Informática**

Septiembre 2011

A Dios, mi familia, amigos, compañeros, profesores y todo aquel que influyó en mi formación universitaria durante estos años.

# Índice

# Resumen

El proceso de testing pretende verificar y determinar que los productos software cumplan con el fin para el que fueron creados. Así, se descubren fallos en su implementación, funcionalidades y aspectos no funcionales. Actualmente, la industria del software sigue la tendencia de evaluarse en modelos que acrediten la calidad de sus procesos y productos. El problema de estas evaluaciones es que son, en su mayoría, orientadas a grandes empresas, por lo que su implantación en organizaciones pequeñas puede resultar difícil y a veces utópica.

Debido a esta situación, es que se propone generar un modelo de mejoramiento para el testing, diseñado para las necesidades y recursos de las pequeñas empresas. Este modelo, se ha desarrollado en base al estudio de otros modelos de mejoramiento de testing, en su mayoría para grandes organizaciones.

Este modelo propuesto ha sido implementado en una organización real, permitiendo formalizar un proceso de testing dentro del ciclo de desarrollo de los productos.



*Palabras-claves: testing, calidad, pequeña empresa, modelo de mejoramiento*.



# Abstract

The testing process is intended to verify and determine whether the software products are accomplishing the purpose for which they were created. Thus, it can finds defects in its implementation, functionality and nonfunctional aspects. Currently, the software industry follows the tendency to be evaluated in models in order to prove the quality of its processes and products. The problem with these assessments is that they are mostly aimed at large companies, so its implementation in small organizations can be difficult and sometimes utopian.

Because of this situation, it is proposed to generate a testing improvement model, designed for the needs and resources of small businesses. This model has been developed based on the study of other models to improve testing, mainly for big organizations.

This proposed model has been implemented in a real organization, allowing a formal testing process in the cycle of product development.



*Key words: testing, quality, small organization, improvement model.*

# Lista de Figuras

# Lista de Tablas

# 1 Introducción

El proceso de pruebas en el software, en inglés testing, tiene por objetivo verificar y determinar si los productos desarrollados cumplen con el fin para el que fueron creados. Normalmente, el software se prueba para descubrir fallos en su implementación y funcionalidades. Sin embargo, además de determinar si el producto realiza las tareas para los cuales fue construido, el proceso de testing puede evaluar la calidad del producto y de los procesos involucrados en su desarrollo. Por lo que, someter al software a un proceso completo y adecuado de testing es de suma relevancia para poder contar con un producto de alto nivel.

Actualmente, las organizaciones proveedoras de bienes y servicios siguen la tendencia de poder certificar sus procesos, y en consecuencia la calidad de sus productos, a través de la adopción de estándares o la obtención de certificaciones de distintos aspectos de calidad. La industria del software no es la excepción a este estilo, y por esto hoy podemos ver que existen una serie de certificaciones, normas y modelos (por ejemplo: ISO o CMMI) que permiten corroborar la calidad pretendida en los procesos de desarrollo, y específicamente en los procesos de testing en las empresas. La gran desventaja, desde un cierto punto de vista, que tienen estas certificaciones es que están dirigidas a grandes organizaciones y adaptarlas a una pequeña empresa puede resultar complejo y excesivamente costoso para su realidad [1]. Por lo que se implementan de forman inadecuada o su implementación es descartada debido a las dificultades que trae.

Debido a esta última situación, es que resulta necesario generar un modelo que permita mejorar el proceso de testing en las pequeñas empresas, de tal manera que se aseguren los procedimientos de pruebas de sus productos, y así brindar mayor calidad en sus proyectos. Para esto, se necesita analizar los modelos existentes y determinar las dificultades específicas que éstos generan y rescatar sus aspectos positivos. De esta forma, se podrá elaborar la propuesta que se pretende aplicar en un caso práctico.

Este documento corresponde al informe final del trabajo realizado para proponer un modelo de mejora para el proceso de testing. En primer lugar, se expone un capítulo sobre la metodología que se aplica, se definen objetivos y se presenta la planificación utilizada en el trabajo. Posteriormente, se expone el desarrollo del marco teórico, en dónde se presentan algunos de los modelos más utilizados actualmente para la mejora de procesos de testing y algunas técnicas a considerar. Luego, un capítulo que presenta una comparación entre los modelos descritos. Después, se entrega un modelo de mejoras desarrollado en base al trabajo realizado en el marco teórico, seguido de la aplicación del modelo en un entorno real junto a los resultados obtenidos de esta experiencia. Por último, se exponen las principales conclusiones respecto al trabajo abarcado a lo largo de este proyecto.

# 2 Metodología

## 2.1 Objetivos

### 2.1.1 Objetivo General

- Generar un modelo de mejoramiento del proceso de testing para pequeñas empresas.

### 2.1.2 Objetivos Específicos

- Realizar un estudio sobre la situación actual de los modelos existentes para procesos de testing.
- Crear un modelo base para el proceso de testing pretendido, a partir de las propuestas existentes.
- Aplicar el modelo en un caso práctico y evaluar sus resultados.

## 2.2 Planificación

El proyecto consistió, primeramente, en el estudio de los conceptos involucrados en torno al testing. Para esto, fueron necesarias consultas bibliográficas y un trabajo de investigación para elaborar un marco teórico adecuado y actual. De esta forma, se analizaron los modelos actuales más utilizados en el testing actualmente y según las experiencias obtenidas en casos prácticos se estudiaron sus fortalezas y debilidades. Para luego considerar las prácticas que han dado buenos resultados, en la elaboración del modelo de mejoras.

Cabe destacar que no sólo se han considerado modelos en esta etapa, sino que también se analizaron distintas metodologías o ideas que están surgiendo en torno a la mejoras del testing. También, depende de los resultados de sus implementaciones, la inclusión de éstos en el modelo deseado.

Posteriormente, en base al estudio de las técnicas y modelos existentes, y las ideas propias del proyecto se propuso un modelo de mejoras al testing que cumpla con las características esperadas.

Por último, el modelo se aplicó a un caso práctico con el fin de obtener resultados sobre su implementación, las cuales pueden entregar información del mismo modelo y de tópicos para un trabajo futuro relacionados al testing.

# 3 Marco Teórico

## 3.1 CMM-CMMI

### 3.1.1 Capability Maturity Model (CMM)

Es un modelo construido por el SEI (Software Engineering Institute) y financiado por el Departamento de Defensa de los EE.UU. a principios de los años 80, como medida ante la llamada “crisis del software”. Este modelo tiene como objetivo llevar a las organizaciones que desarrollan y mantienen software a un nivel de madurez, que potencien de manera eficiente sus capacidades de tal manera de obtener proyectos exitosos y de calidad [2].

El modelo se basa en conceptos de calidad y de mejoramiento continuo, esto debe ser entendido y aplicado por toda la organización para alcanzar el nivel máximo de madurez. Un proceso se dice que es maduro si cumple con: estar definido, estar documentado, su personal está entrenado, practicar lo entrenado, ser mantenido, estar controlado, ser verificado, ser validado, ser medido y poder ser mejorado.

Cabe destacar, que se existe toda una familia CMM. El más popular, y el que se comentará en este trabajo, es el SW-CMM que normalmente es conocido simplemente como CMM.

El progreso de madurez se mide de acuerdo a los “niveles de madurez”. Cada uno de los cinco niveles posee Áreas de proceso clave (Key Process Area, KPA) que deben cumplirse, esto es medido mediante la satisfacción de metas. A su vez, para cumplir con las KPA deben llevarse a cabo ciertas prácticas definidas. En la figura 3.1 se grafican los cinco niveles de CMM y la tabla 3.1 muestra las KPAs por cada nivel CMM.

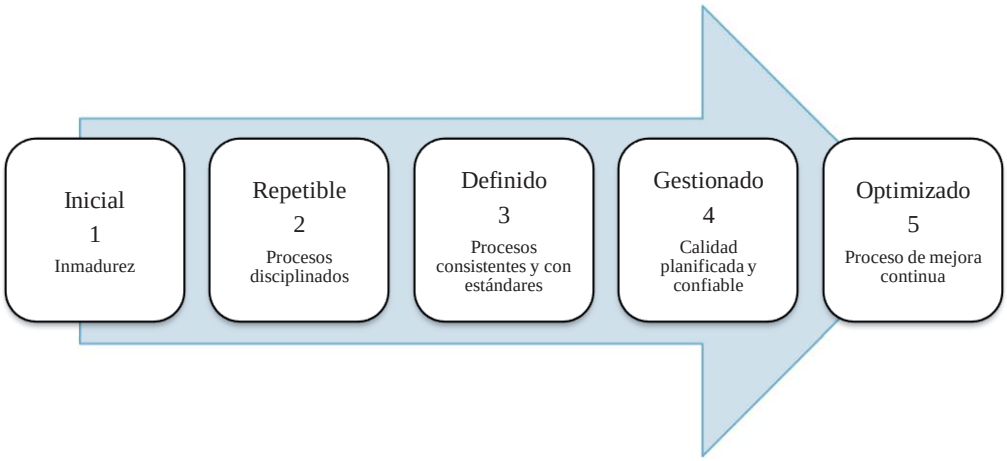


Figura 3.1 Niveles de Madurez CMM

Tabla 3.1 KPAs por niveles CMM.

| Nivel | KPAs |
|---|---|
| **Nivel 1** | No hay. |
| **Nivel 2** | Gestión de requisitos.<br>Planificación de proyectos del software.<br>Seguimiento y supervisión del proyecto software.<br>Gestión de subcontratación de software.<br>Aseguramiento de calidad de software.<br>Gestión de configuración del software. |
| **Nivel 3** | Enfoque en el proceso organizacional.<br>Definición del proceso organizacional.<br>Programa de entrenamiento.<br>Gestión integrada de software.<br>Ingeniería de productos de software.<br>Coordinación entre grupos.<br>Revisión de pares. |
| **Nivel 4** | Gestión cuantitativa del proceso.<br>Gestión de la calidad del software. |
| **Nivel 5** | Prevención de defectos.<br>Gestión de cambio de tecnología.<br>Gestión de cambio de proceso. |

Una breve descripción de los niveles de CMM es la siguiente:

1. Nivel Inicial: Este es el estado inicial de toda organización. A este nivel los proyectos no se pueden planear, los requerimientos no están controlados, no existe control de versiones, la calidad se considera burocracia y el personal vive en constante estrés y frustración. Resulta común que la inversión realizada en un proyecto no se vuelva a usar en futuras tareas, pues prácticamente no existe el concepto de reutilización.

2. Nivel Repetible: Ya existe la noción de administración de proyectos, de control de requerimientos, versiones del producto y se comienza a utilizar la subcontratación. El personal hace uso de su experiencia y trabajo en los proyectos futuros (reutilización). Es común que a este nivel las mejoras no sean implementadas a nivel total de la organización, y que existan proyectos en un nivel inicial de inmadurez.

3. Nivel Definido: Existen procedimientos y estándares comunes debidamente documentados a seguir y el personal está instruido en el uso de ellos. También, se conocen pautas y criterios para adaptar los modelos a las características propias de cada proyecto. Se logra disminuir la dependencia respecto a personas irreemplazables, pues todo es documentado.

4. Nivel Gestionado: La organización es capaz de medir la calidad de sus productos y procesos, y para esto existen métricas claras. De esta forma, se pueden estimar los rendimientos y tomar medidas cuando éstos disminuyen.

5. Nivel Optimizado: Se trata de una mejora continua en la organización. Todo el personal está consciente de la importancia de las mejoras y conocen sus beneficios. Se analizan los defectos existentes en proyectos para determinar causas y evitarlos a futuro.

### 3.1.2 Capability Maturity Model Integration (CMMI)

En los años 90 la familia CMM ya contaba con variados modelos para la mejora y medición de la madurez, en distintas áreas, de las organizaciones, por lo que era común encontrarse con casos en que se implementaron simultáneamente distintos modelos en una misma organización. CMMI se creó con la intención de permitir la adopción simultánea de modelos.

Actualmente, existen tres áreas cubiertas por distintos modelos: CMMI-DEV [3] para el desarrollo -explicado en este documento-, CMMI-ACQ para la adquisición y CMMI-SVC para servicios.

CMMI-DEV consiste en la implementación de una serie de buenas prácticas, acordes a ciertas metas, que permitirán a una organización mejorar los procesos para su desarrollo de bienes y servicios. Para esto, se trabajan 22 áreas de procesos.

Para describir o esbozar el camino hacia la mejora de procesos (la situación ideal) CMMI utiliza una escala de niveles. Para esto, existen dos tipos: Escala de madurez (llamada representación escalonada) y la escala de capacidades (llamada representación continua). La diferencia entre ambas escalas, a grandes rasgos, es que CMMI aplica un camino para alcanzar un mejor nivel de capacidad de un área de procesos de la organización o un mejor nivel de madurez de la organización. En la tabla 3.2 se muestra la escala de niveles para ambos enfoques.

Tabla 3.2 Niveles CMMI.

| Nivel | Capacidad | Madurez |
|---|---|---|
| **0** | Incompleto | - |
| **1** | Ejecutado | Inicial |
| **2** | Gestionado | Gestionado |
| **3** | Definido | Definido |
| **4** | | Cuantitativamente Gestionado |
| **5** | | Optimizado |

En otras palabras, los niveles de capacidad son aplicados para la obtención, en una organización, de mejoras en áreas de procesos individuales y los niveles de madurez son aplicados a las mejoras de la organización en múltiples áreas de procesos.

#### 3.1.2.1 **Niveles de Capacidad**

Un nivel de capacidad se alcanza cuando todas las metas genéricas del nivel son alcanzadas. Cada área de proceso, se trabaja hasta un cierto nivel de capacidad según las necesidades de la organización.

- Incompleto: Se trata de cuando los procesos no se están realizando o sólo se realizan parcialmente. Por lo tanto, no se satisfacen una o más de las metas específicas de las áreas de procesos y no existen las metas genéricas.

- Ejecutado: A este nivel se realizan los procesos. Esto quiere decir que los procesos cumplen las necesidades de trabajo para producir los productos o servicios de la organización. Se cumplen las metas específicas.

- Gestionado: El nivel 2 de capacidades se caracteriza por tener procesos gestionados. Es decir, se realizan los procesos según los planes y la política establecida por la organización.

- Definido: Los procesos son gestionados y además se definen a medida de la organización; se cuenta con una descripción de mantención.

#### 3.1.2.2 **Niveles de Madurez**

El motivo por el cual la escala de madurez en sus niveles 2 y 3 utilice los mismo nombres de la escala de capacidad es debido a que niveles de madurez y niveles de capacidad son complementarios. Es decir, los niveles de madurez indican las mejoras de una organización en torno a un conjunto de áreas de procesos, y los niveles de capacidad las mejoras de una organización en torno a un área de procesos en particular. A diferencia de los niveles de capacidad, los niveles de madurez tienen establecidas cuales áreas de procesos deben ser trabajadas para alcanzar el siguiente nivel

- Inicial: En este nivel los procesos se dice que se realizan de forma caótica, no existe orden ni claridad dentro de la organización. Esto se debe, entre otros motivos, a que es la misma organización la que no provee un ambiente necesario para el buen desarrollo de los procesos. Las organizaciones que se encuentran en este nivel pueden desarrollar sus productos, sin embargo es común exceder los costos y plazos estimados. Además, es frecuente el abandonar los procesos cuando éstos son incontrolables y los sucesos obtenidos son difíciles de repetir.

- Gestionado: En este nivel los proyectos son planeados y ejecutados de acuerdo con las políticas establecidas en la organización. Los procesos de disciplinas alcanzados en este nivel ayudan a mantener las buenas prácticas durante los períodos de sobrecarga de trabajo, por lo que se puede considerar a la organización como una más robusta.

- Definido: En este nivel los procesos se entienden y se describen bajo estándares, procedimientos, herramientas y métodos. Éstos deben revisarse y mejorarse en el tiempo. En comparación al nivel 2, en este nivel los estándares y descripciones de procesos no varían según cada proyecto, sino que éstos se adaptan a la naturaleza propia de cada proyecto. Por lo tanto, se dice que los procesos del nivel 3 son más rigurosos que en niveles anteriores, pues los procesos están claramente definidos al igual que los resultados que se esperan de ellos.

- Cuantitativamente Gestionado: La organización establece objetivos cuantitativos para medir la calidad y controlar sus proyectos. Estos objetivos se basan en las necesidades del cliente, de usuarios finales y los ejecutores de los procesos. Se puede establecer una línea base y modelos para ayudar a establecer la calidad de los procesos y sus objetivos a nivel organizacional. Alcanzado este nivel, la realización de ciertos procesos se puede predecir, debido a que el control de procesos se realiza usando técnicas estadísticas y cuantitativas que permiten anticipar resultados.

- Optimizado: Se trata del nivel en que la organización busca una mejora continua de sus procesos. Las mejoras se basan en objetivos cuantitativos en base a las necesidades del negocio, y las decisiones sobre las nuevas mejoras son decididas en base a métodos cuantitativos. De modo global, los estándares mismos de la organización y el soporte tecnológico están sujetos a mejoras medibles de sus actividades.

## 3.2 Testability Maturity Model (TMM)

Desarrollado por Software Quality Engineering (SQE), consultora del estado de Florida de EE.UU. Como su nombre lo sugiere, TMM está basado, o al menos comparte cierta ideología, con CMM, pues entrega un marco de trabajo para guiar la mejora de prácticas basadas en información y relaciones con áreas claves de testing. El objetivo es identificar los obstáculos que la organización debe superar para realizar un mejor testing. [4]

### 3.2.1 Niveles TMM

TMM posee tres niveles que miden que tan efectivo ha sido el esfuerzo de la organización por superar los obstáculos del testing. Es decir, en la medida que los obstáculos son identificados y superados se aumenta de nivel TMM.

- Nivel 1: Existe un apoyo débil para el testing, reflejado en que pocos aspectos del testing han sido abordados.
- Nivel 2: Se alcanza un apoyo básico para la testing, esto es que los aspectos básicos del testing son gestionados debidamente.
- Nivel 3: Hay un gran apoyo para la testing, es decir, todos los aspectos del testing son considerados y gestionados.

En la figura 3.2 se muestran los niveles de TMM.

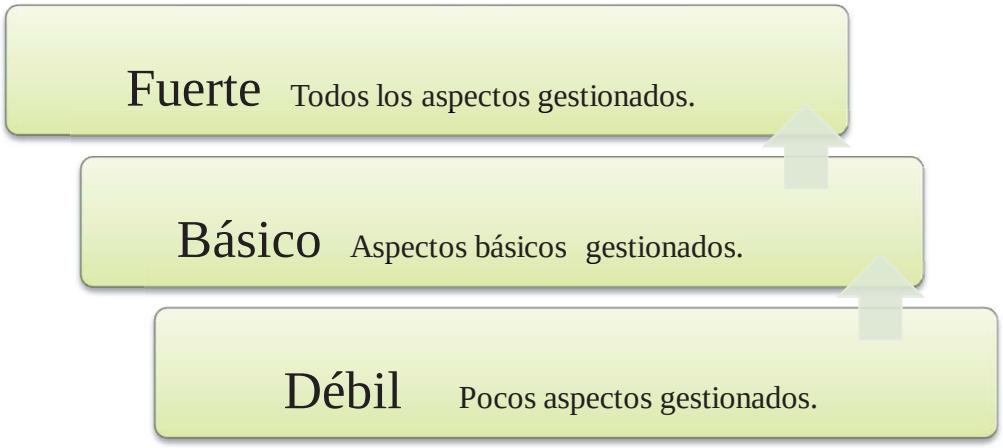


Figura 3.2 Niveles de TMM.

### 3.2.2 Áreas Clave de Apoyo

Así como CMM tiene sus KPAs, TMM se basa en sus Áreas Clave de Apoyo (Key Support Areas, KSA) las que en la medida que son gestionadas e implementadas en la organización determinan el nivel TMM alcanzado. La tabla 3.3 muestra las KSAs y sus ítems implicados.

Tabla 3.3 KSAs de TMM.

| KSA | Ítems implicados |
|---|---|
| **Infraestructura amigable para el testing.** | Gestión de configuración efectiva para el software y testware.<br>Preservar y reutilizar conjuntos de prueba.<br>Automatización del ciclo de vida del testing.<br>Metodologías y métodos efectivos.<br>Detección de patrones de problemas.<br>Entrenamiento en testing. |
| **Planeación consciente de los proyectos de testing.** | Obtención de información de los requerimientos del producto.<br>Determinación de peligros en el producto.<br>Comunicación y cooperación con desarrollo.<br>Distribución de recursos y planeación de tareas acorde al testing.<br>Testers incluidos en el control de cambios.<br>Balance entre análisis, revisiones y testing del producto.<br>Medición de cambios en el costo-efectividad del testing. |

| KSA | Ítems implicados |
|---|---|
| **Información amigable del testing del producto.** | Estabilidad en la estructura y contenido de los requerimientos.<br>Patrones de uso del producto.<br>Peligro en aplicación del producto.<br>Conocimiento detallado de las relaciones de distintos aspectos del producto.<br>Diseño técnico para accesibilidad y comprensión del tester. |
| **Diseño consciente del testing del software.** | Diseño funcional que guía las elecciones del usuario, incluyendo visibilidad y mecanismos de control.<br>Diseño técnico que apoye la simulación y ejecución continua, minimización de alternativas y limitación de la carga.<br>Código accesible y comprensible por el tester.<br>Datos y códigos con costos alcanzables y efectivos. |
| **Testware (procedimientos, software de soporte, sets de datos y la documentación utilizada en el testing).** | Reutilización de testware debe ser apoyada por: Gestión de configuración del testware, modularidad e independencia del testware, guía hacía los objetivos y uso de herramientas automatizadas. |
| **Entorno de diseño de testing.** | Apoyo automatizado para la ejecución y evaluación de resultados del testing.<br>Flexibilidad para simular situaciones críticas.<br>Gran usabilidad. |

### 3.2.3 Relación de Niveles CMM-TMM

Existe una relación no necesariamente recíproca entre los niveles alcanzados en una organización por ambos modelos. Esto es, que un alto nivel TMM implica un alto nivel CMM (o al menos las condiciones de CMM), pues resulta necesario para cumplir con los aspectos de mejoras del testing tener un alto nivel de maduración de las capacidades en la organización. Por otro lado, si se tiene un alto nivel CMM no se puede garantizar un alto nivel TMM, dado que CMM no especifica aspectos orientados a la testabilidad.

## 3.3 Test Maturity Model Integration (TMMi)

Es un marco de trabajo desarrollado por la fundación TMMi dirigido a la mejora de procesos de testing y complementario a CMMI [5]. Su principal fuente o modelo base es el TMM, además del mismo CMMI.

Utiliza una escala con niveles de madurez para medir las mejoras conseguidas en la organización, regidas por sus metas y prácticas correspondientes. Así, en la medida que la organización que aplique TMMi aumente de nivel, sus procesos de testing mejorarán e impactarán directamente en la calidad de sus productos.

Cabe destacar que por ser de naturaleza complementaria al CMMI, normalmente un nivel TMMI sólo hará referencia a un mismo o menor nivel CMMI. Por este motivo, entonces,

algunas prácticas necesarias para el TMMi serán referenciadas desde el CMMI para poder utilizarlas en el contexto del testing.

### 3.3.1 Niveles de Madurez TMMi

En la medida que la organización vaya aprendiendo y aplicando las distintas prácticas entregadas por el modelo, ésta irá aumentando su nivel de madurez TMMi. Existen cinco niveles que determinan el estado de los procesos de testing de la organización. Cada uno de los niveles tiene un conjunto de áreas de procesos que se deben mejorar para alcanzar un determinado nivel. En la figura 3.3 se muestran los niveles de madurez de TMMi.

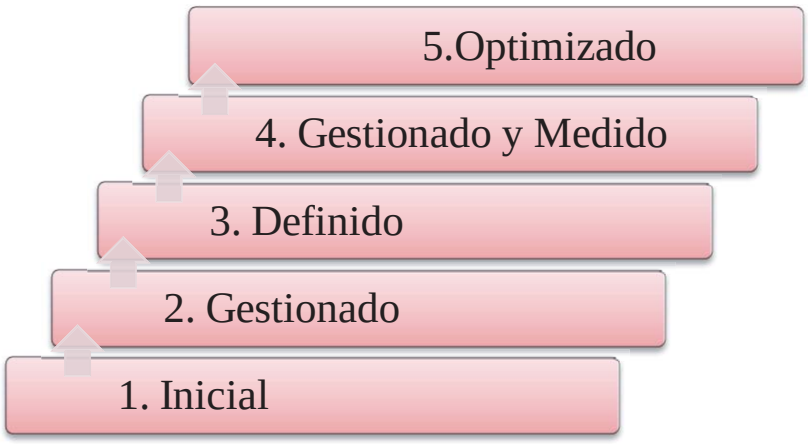


Figura 3.3 Niveles de madurez TMMi.

**Inicial**

Se trata de una organización en la que sus procesos de testing se pueden catalogar como caóticos. Los procesos de testing no están definidos y probablemente tampoco exista un entorno adecuado para éstos. A este nivel el testing prácticamente sólo existe para permitir la liberación del producto sin grandes errores y como consecuencia se resiente la calidad de éste y puede no cumplir con todos los requerimientos exigidos por el cliente.

Existe una falta de herramientas y recursos, así como de capacitación adecuada para el personal. Además es común que procesos de testing se abandonen antes de terminar y la reutilización es prácticamente nula. Todo esto, puede provocar retrasos en la entrega del producto.

**Gestionado**

En este nivel comienza a gestionarse el testing, además se inicia toda una estrategia de testing para la empresa. Lo anterior implica el desarrollo de planes que permitirán gestionar los riesgos que puedan amenazar el desarrollo de las pruebas.

Se dispone de casos de prueba los cuales se seleccionan y/o adaptan de acuerdo a los proyectos que se tengan. A pesar de las mejoras, se sigue detectando un retraso en la inclusión del testing en el ciclo de vida del producto. Las pruebas se realizarán a distintos niveles (unitarias, integración, de sistema y aceptación) cada una con sus objetivos claramente definidos por una estrategia global de la empresa.

Los productos aún tendrán deficiencias de calidad, esto debido al retraso de la integración del testing en el ciclo de vida, sin embargo a este nivel de madurez los productos cumplirán los requerimientos especificados por los clientes.

Las áreas de procesos que comprenden este nivel son:

- Política y estrategia de testing: Desarrollar y establecer una política y estrategia de organización en la cual se definan los alcances de las pruebas.

- Planeamiento de testing: Definir un enfoque basado en la identificación de riesgos y la estrategia definida.

- Monitoreo y control de testing: Proveer un entendimiento del progreso del testing y la calidad del producto.

- Diseño y ejecución de testing: Mejorar las capacidades del proceso de testing durante su diseño y ejecución.

- Ambiente de testing: Establecer y mantener un ambiente adecuado.

**Definido**

Una organización con este nivel de madurez en sus procesos de testing, incluirá el diseño de pruebas desde la toma de requerimientos del proyecto. Las pruebas ya pasan a formar una parte importante de la organización, pues se profesionaliza y se valora su importancia para alcanzar productos de calidad.

El personal profesional de testing tiene implicancia en distintas etapas del proyecto, por ejemplo en la especificación de requerimientos. Además, las pruebas comienzan a incluir más aspectos a evaluar como la usabilidad o confiabilidad de los productos.

A diferencia del nivel de madurez anterior, la documentación, procedimientos y estándares ya son a nivel total de la organización y no a medida de cada proyecto.

Las áreas de proceso en el nivel 3 son:

- Organización del testing: Identificar y organizar un grupo de personas altamente calificada para ser responsables del testing.

- Programa de entrenamiento para el testing: Desarrollar un programa de entrenamiento, que facilite el desarrollo de conocimientos y habilidades para las personas encargadas de las tareas de testing.

- Integración del testing con el ciclo de vida: Establecer y mantener un conjunto de procesos organizacionales de testing y estándares de trabajo para integrarlos al ciclo de vida del proyecto.

- Testing no funcional: Mejorar la calidad del proceso de testing planificando, diseñando y ejecutando enfoques de testing que evalúen aspectos no funcionales del producto.

- Revisiones de pares: Verificar que el producto cumple con los requerimientos especificados.

**Gestionado y medido**

Para el nivel 4 los objetivos del testing se definen de forma cuantitativa de forma de poder medir de manera clara su alcance en el producto. Es más, las métricas se hacen parte de toda la organización, lo que favorece la toma de decisiones basándose en hechos concretos.

Por otro lado, las revisiones e inspecciones pasan a formar parte del proceso de testing y también se controlan por métricas definidas a nivel organizacional. Los productos al ser evaluados cuantitativamente pueden ser mejor medidos tanto funcional como no funcionalmente y garantizan de mejor forma su eficacia y eficiencia.

Las áreas de proceso para este nivel son:

- Métricas de testing.

- Evaluación de la calidad del producto.

- Revisiones avanzadas de pares.

**Optimizado**

En el nivel 5 de madurez el testing pasa a ser un proceso completamente definido y fundamental en el desarrollo. El testing está constantemente siendo mejorado, pues de él depende en gran parte el control de los costos de los proyectos.

A partir de este punto el testing está basado en métricas objetivas, y se introducen al proceso nuevas secciones como “Prevención de defectos” y “Control de calidad”. Y las herramientas que apoyan el testing se definen en base a procesos establecidos.

El testing ha pasado a ser un proceso que comienza en prevenir los defectos.

Las áreas de procesos en el nivel 5 son:

- Prevención de defectos.

- Optimización del proceso de testing.

- Control de calidad.

## 3.4 Test Improvement Model (TIM)

TIM, o en español Modelo de Mejora de Pruebas, es una guía para la mejora del procesos de pruebas que usa de base CMM y TMM [6]. Este modelo está dirigido a cinco áreas clave (Keys Area, KA): Organización, Planificación y Seguimiento, Casos de Pruebas, Testware y Comentarios.

Este modelo es usado de dos formas. En primer lugar, para definir las prácticas que se llevan a cabo en las áreas clave definidas. Y por otro lado, también se usa para definir o esbozar el camino hacia "el destino", es decir, la situación de mejora que se quiere alcanzar.

Se forma a partir de dos componentes principales: un marco de trabajo, que incluye una escala de cuatro niveles y las áreas clave ya mencionadas, y un procedimiento de evaluación. Cada uno de los niveles de la escala, es nombrado según la meta total que se pretende alcanzar en dicho nivel, a su vez cada meta total está compuesta por sub-metas que deben ser completadas en su totalidad para cumplir la meta total. Las cinco áreas clave son medidas por la misma escala, y en cada área deben realizarse determinadas actividades para conseguir las metas.

### 3.4.1 Niveles TIM

La escala de niveles del marco de trabajo se muestra, en orden creciente, en la tabla 3.4.

Tabla 3.4 Niveles y sub-metas TIM.

| Nivel | Sub-metas |
|---|---|
| **Base Inicial** | Estándares, políticas, procedimientos, métodos y metodologías documentadas.<br>Una función básica de testing. |
| **Costo-Eficacia** | Costos de reprocesos y gestión del cambio minimizados.<br>Pruebas rápidas y sistemáticas.<br>Las tareas se realizan de "forma correcta".<br>Minimización de desarrollo de artefactos existentes. |
| **Disminución de Riesgos** | Apoyo desde todos los niveles de gestión.<br>Detección de errores peligrosos.<br>Riesgo dirigido por la programación y asignación de los recursos. |
| **Optimización** | Pruebas gestionadas y documentadas a nivel total de acuerdo a la política de calidad de la organización.<br>Cambio cultural en cuanto a la mejora y prevención continua.<br>Testing integrado totalmente con el ciclo de vida del proyecto.<br>Las decisiones se basan en hechos. |

La Base Inicial es donde se implementarán los cambios más básicos, pero que permitirán seguir evolucionando en los siguientes niveles de TIM. Aquí, se debe formar un grupo dedicado al testing encargado de establecer los roles, responsabilidades y documentos

necesarios para iniciar el mejoramiento de las prácticas. Luego, una vez que en este primer nivel estén establecidos los esfuerzos, éstos se pueden comenzar a destinar a la disminución de costos o en incrementar la relación de su efectividad. Posteriormente, la disminución de riesgos permitirá actuar y estar preparado ante situaciones indeseadas que puedan poner en peligro los proyectos. Por último, la optimización será capaz de asegurar la calidad permitiendo mantener una mejora continua en los procesos del testing. En la tabla 3.5 se muestran las estrategias propuestas para alcanzar cada nivel.

Tabla 3.5 Estrategias TIM

| Nivel | Estrategias |
|---|---|
| **Base Inicial** | Documentación de estándares, métodos y políticas.<br>Análisis y clasificación de los problemas del producto. |
| **Costo-Eficacia** | Detección temprana.<br>Tareas de testing asistidas por computador.<br>Entrenamiento.<br>Reutilización. |
| **Disminución de Riesgos** | Mejora temprana.<br>Justificación de gastos.<br>Análisis de problemas del producto y procesos.<br>Medición del producto, procesos y recursos.<br>Análisis y gestión del riesgo.<br>Comunicación entre todas las partes del proyecto. |
| **Optimización** | Conocimiento y entendimiento a través de la experimentación y modelado.<br>Mejora continua.<br>Análisis de raíz.<br>Testing equilibrado en cuanto a necesidades del producto y políticas de calidad.<br>Cooperación entre todas las partes del proyecto y en todas las fases de desarrollo. |

### 3.4.2 Áreas Clave

Cada área se relaciona con actividades específicas distribuidas en los cuatro niveles de TIM.

#### Organización

La capacidad organizacional será fundamental para el mejoramiento de las prácticas en el testing. Esto incluye un equipo, prácticas, estándares y documentación organizada ligada al testing que permita la mejora continua y así alcanzar hasta el máximo nivel. Las actividades sugeridas por nivel se muestran en la tabla 3.6.

**Planificación y seguimiento**

La planificación implica definir estrategias y objetivos para la conducción de proyectos. El seguimiento es necesario para chequear el progreso del trabajo realizado. La tabla 3.7 muestra las actividades por nivel.

**Casos de prueba**

Los procesos de testing deberían ser realizados en base a casos de prueba que permitan diseñar debidamente el proceso, pues éstos determinarán, en cierta forma, la calidad del proceso. La tabla 3.8 contiene las actividades por nivel.

**Testware**

Se entiende por testware a los procedimientos, el software de soporte, los sets de datos y la documentación de respaldo que se utiliza en el testing. Es evidente, que la calidad y capacidad del testware son claves para alcanzar un buen nivel de testing. La tabla 3.9 muestra las actividades por cada nivel.

**Revisiones**

Son todas las técnicas que involucren lectura inspecciones visuales de documentación del software. En la tabla 3.10 se muestran las actividades recomendadas para cada nivel.

Tabla 3.6 Actividades KA organización TIM.

| Nivel | Actividades |
|---|---|
| **Base Inicial** | La función de testing está organizada según los estándares documentados.<br>Existe un líder y un núcleo en el equipo de testing. |
| **Costo-Eficacia** | El personal clave trabaja a tiempo completo en un solo proyecto y los recursos no se comparten.<br>La función de testing es independiente.<br>La estructura del equipo está basada en la disponibilidad y necesidades del proyecto.<br>Las responsabilidades están claramente definidas, documentadas y comprendidas.<br>El grupo de testing dispone de espacios y herramientas adecuadas.<br>Entrenamiento para conocer las necesidades de la empresa. |
| **Disminución de Riesgos** | La asignación de recursos y planeación de las tareas se realiza en todos los niveles.<br>Participación de testers en todas las fases de desarrollo y mejoras en el desarrollo de testing.<br>Rotación del personal entre desarrollo y testing. |
| **Optimización** | Los tester son parte de equipos multidisciplinarios.<br>Mejora organizacional continúa.<br>Existencia de grupos para la mejora de procesos. |

Tabla 3.7 Actividades KA planificación y seguimiento TIM.

| Nivel | Actividades |
|---|---|
| **Base Inicial** | Se realiza una planeación básica.<br>Las condiciones de entrada y salida están definidas para todos los niveles del testing.<br>Los resultados de las pruebas son debidamente documentados, procesados y distribuidos.<br>Los planes son realizados siguiendo estándares, y sus modificaciones son documentadas y justificadas.<br>La planificación es realizada de acuerdo a las políticas documentadas.<br>Los planes pueden cambiar con los requerimientos.<br>Los cambios al plan de pruebas están hechos de acuerdo a la política documentada. |
| **Costo-Eficacia** | La planificación y su seguimiento es asistida computacionalmente.<br>Se tienen planes genéricos.<br>La elección del método y nivel de pruebas es acorde a los objetivos del testing y los objetos a probar.<br>El progreso se mide y se compara con la planificación.<br>La planificación es realizada por un planeador o tester capacitado.<br>La planificación se realiza de forma evolutiva.<br>La predicción de recursos es realizada de acuerdo a la política documentada. |
| **Disminución de Riesgos** | El análisis de riesgo depende de la planificación y la disposición de los recursos.<br>Las partes afectadas deben aprobar los planes incluyendo los objetivos de testing.<br>Se monitorea el cumplimiento de los objetivos de testing.<br>La planificación es hecha por un tester o planificador experto. |
| **Optimización** | Los modelos están hechos de costos, recursos, riesgos, etc.<br>Las actividades de planificación y seguimiento son mejoradas continuamente basándose en análisis y mediciones de la experiencia.<br>Una vez terminado el proyecto se realizan mediciones y sus resultados son debidamente distribuidos. |

Tabla 3.8 Actividades KA casos de prueba TIM.

| Nivel | Actividades |
|---|---|
| **Base Inicial** | Los casos de prueba son revisados cuando cambian los requerimientos.<br>Los casos de prueba están basados en los requerimientos.<br>Los casos de prueba son diseñados y documentados de acuerdo a la política documentada.<br>Los casos de prueba son ejecutados de acuerdo a las instrucciones documentadas. |
| **Costo-Eficacia** | Los casos de prueba son diseñados usando técnicas documentadas.<br>Los aspectos de pruebas influyen los requerimientos y diseño.<br>Los casos de prueba son registrados y reutilizados.<br>Los casos de prueba son organizados con respecto al nivel de prueba, los requerimientos, los objetos a probar y objetivos. |
| **Disminución de Riesgos** | Los casos de prueba se ordenan y seleccionan de acuerdo a su orden de relevancia.<br>Los casos de prueba son diseñados en base a los riesgos. |
| **Optimización** | Las técnicas de diseño de los casos de prueba son medidas, revisadas y mejoradas. |

Tabla 3.9 Actividades KA testware TIM.

| Nivel | Actividades |
|---|---|
| **Base Inicial** | El reporte de problemas es computarizado.<br>Se usan archivos de sistema para la "gestión de configuración" |
| **Costo-Eficacia** | Seguimiento de problemas, trazabililidad y ejecución del testing asistidos computacionalmente.<br>Recolección, procesamientos y reportes de casos de pruebas asistidos computacionalmente.<br>Se utilizan herramientas para el análisis de cobertura.<br>Simulaciones para el hardware y software.<br>Bases de datos para la gestión de configuración del testware.<br>Herramientas de pruebas son evaluadas antes de ser adquiridas y puestas en acción.<br>Se utilizan herramientas de análisis estático. |
| **Disminución de Riesgos** | Análisis de riesgo asistido computacionalmente.<br>Testware y software están bajo un control de gestión efectivo.<br>Pruebas de regresión asistidas computacionalmente.<br>Utilización exclusiva de tecnología madura. |
| **Optimización** | Comprobación de resultados y evaluaciones asistidos computacionalmente.<br>Diseño de testware asistido computacionalmente.<br>Desarrollo de test integrado.<br>Experimentación con herramientas. |

Tabla 3.10 Actividades KA revisiones TIM.

| Nivel | Actividades |
|---|---|
| **Base Inicial** | Los requerimientos se revisan.<br>Los líderes de revisión son capacitados en técnicas de revisión.<br>Se invierte en soporte y capacitación.<br>Utilización de técnicas de revisión estandarizadas y documentadas. |
| **Costo-Eficacia** | Todo el equipo de revisión es capacitado en técnicas de revisión.<br>Se elaboran y se utilizan listas de tareas y escenarios.<br>Se revisan documentos de diseño y codificación.<br>La organización puede optar entre algunas técnicas de revisión. |
| **Disminución de Riesgos** | Se revisa el testware y los casos de prueba.<br>Las técnicas de revisión son revisadas.<br>Se miden las revisiones de procesos, productos y recursos. |
| **Optimización** | Se mejoran continuamente las técnicas de revisión.<br>La selección del equipo y técnicas de revisión se basan en hechos. |

### 3.4.3 Procedimiento de Evaluación

Consiste en dos instancias. En primer lugar, se determina la situación actual de la organización, para luego establecer el desarrollo de un plan de mejoras.

Para la evaluación inicial se deben seguir los siguientes pasos:

1. El modelo y el procedimiento de evaluación se explican a las personas en la organización. Esto incluye personal de todos los niveles de la organización.

2. La evaluación se debe realizar a través de entrevistas y cuestionarios que permitan obtener apreciaciones de personas claves en la organización.

3. Los resultados obtenidos se analizan para obtener un "perfil maduro".

Luego, para desarrollar el plan de mejora se sugiere:

1. Identificar la solución: A través del "perfil maduro" se deben proponer estrategias para la mejora de la organización.

2. Análisis de solución: Se discuten las estrategias propuestas y se selecciona la mejor en base a los costos, riesgos y otros factores relevantes para la organización.

3. Presentación: La propuesta de mejora y sus tareas a seguir se presentan al personal.

## 3.5 TestPAI

Se define como un área de proceso de pruebas integrado con CMMI. Se sitúa en el nivel 3 junto a los procesos de ingeniería. Nace como respuesta a la necesidad de las pequeñas y

medianas empresas de un marco de trabajo para lograr una mejora del proceso de prueba en un contexto idóneo a su realidad (costos y tiempo). El motivo de que sea integrado a CMMI, por supuesto, es debido a que muchas organizaciones están trabajando con CMMI para mejorar sus procesos ya existentes, y de esta manera se logra una cierta coherencia entre los métodos usados.

TestPAI define cinco metas específicas (Specific Goals, SG), cada una se logra a través de sus prácticas específicas (Specific Practices, SP). La tabla 3.11 muestra dichas metas y sus prácticas.

Tabla 3.11 Metas y prácticas de TestPAI.

| Metas | | Prácticas |
|---|---|---|
| **Establecer Objetivos y Políticas** | Se deben definir objetivos para cada tipo de prueba (aceptación, sistema, integración y unitarias). | • Establecer objetivos de pruebas.<br>• Establecer política de pruebas. |
| **Planificar Pruebas** | La planificación se debe realizar al comienzo del proyecto, de manera de obtener una correcta estimación de costos y tiempos requeridos. | • Establecer el alcance.<br>• Establecer estrategia de pruebas.<br>• Identificar y establecer recursos.<br>• Establecer presupuesto y calendario.<br>• Establecer programa de formación.<br>• Desarrollar y mantener el plan de pruebas. |
| **Especificación de Casos de Prueba** | Cada tipo de prueba necesita su propia especificación | • Especificación del diseño de pruebas.<br>• Especificación de los casos de prueba.<br>• Especificación del procedimiento de pruebas. |
| **Ejecución Exitosa de las Pruebas** | Entre mejor se desarrolle el proceso de pruebas, más probabilidad hay de encontrar defectos en el sistema. | • Preparación del entorno.<br>• Generación de casos de prueba.<br>• Ejecución de pruebas. |
| **Análisis y Reporte de Resultados** | Analizar, evaluar y comunicar los resultados obtenidos del proceso de pruebas. | • Establecer informe de pruebas.<br>• Evaluar el proceso de pruebas.<br>• Determinar acciones correctivas. |

Para mayor especificación de TestPAI, de sus objetivos generales y prácticas específicas se recomienda revisar su documentación oficial [7].

## 3.6 Plan de Testing-TUTELKÁN

Tutelkán es un proyecto conjunto de universidades (UTFSM y UCHILE), asociaciones gremiales y empresas privadas, y financiado por CORFO que busca generar contenidos de procesos de software y de transferencia metodológica, que sean públicamente accesibles, mantenidos y entregados a través de una plataforma tecnológica propia con el propósito de lograr la incorporación de estándares de calidad internacionalmente aceptados a organizaciones (principalmente PyMEs) en sus procesos de desarrollo y mantención de software [8].

El proyecto Tutelkán propone una plantilla llamada "Plan de Testing" [9] que se plantea como una guía para el testing de cada proyecto. Los datos solicitados en la plantilla se completan dependiendo de las especificaciones propias de cada proyecto.

A continuación se explican las secciones que componen el plan de testing:

**Introducción**

Se define el propósito y ámbito del testing. El propósito se expresa en términos de objetivos de requerimientos, estrategias, recursos, estimación de esfuerzos y elementos entregables. El ámbito contempla una descripción escrita sobre quien recibirá el plan a desarrollar, definiciones sobre testing unitarios y testing funcional, además de los riesgos que se pueden presentar y sus acciones a seguir, un catastro de la documentación disponible.

Además, se formalizan las abreviaciones y modismos a utilizar, así como también bibliografía que se utilice en el proceso.

**Requerimientos para el test**

Se exponen detalladamente los requerimientos del testing, para esto se definen los casos de prueba en forma global. Y se detallan los requerimientos funcionales y no funcionales.

**Estrategia de pruebas**

En esta sección se establece la forma en que se realiza el testing. Se especifican los tipos de pruebas en dónde se asegura desde la integridad de la base de datos y se definen condiciones especiales de las pruebas funcionales. Esta sección incluye pruebas de rendimiento y seguridad.

**Recursos**

Se especifican los recursos para llevar a cabo las pruebas, incluyendo los profesionales destinados a las distintas tareas del testing tales como: Diseñadores, testeadores,

administradores e implementadores. Además, se define el ambiente del testing que incluye el hardware, software y recursos comunicacionales para llevar a cabo los procesos.

**Hitos del proyecto de testing**

Se establecen las fechas de inicio y término de las principales actividades del plan, diseño, implementación, ejecución y evaluación de las pruebas. Se realiza en forma de una tabla que además considera los responsables y los recursos involucrados en las actividades.

**Entregables**

Se detallan los documentos que serán creados a partir de las pruebas, incluyendo datos del autor, destinatarios y fechas de entrega.

**Anexos**

El primer anexo, titulado "Tareas del proyecto" es un cheklist que será completado en la medida que se realizan las tareas programadas.

Además, en esta sección se adjuntan anexos para el apoyo de las pruebas de rendimiento y seguridad. Estableciendo objetivos, consideraciones especiales y otros aspectos relevantes.

## 3.7 ISO/IEC 29110

Se origina debido a la necesidad de obtener certificaciones de calidad para organizaciones pequeñas. ISO a través del grupo de trabajo WG24 trabajó en la realización del estándar ISO/IEC 29110, cuyo objetivo es la mejora de los procesos en el ciclo de vida del software en las VSEs (Very Small Enterprises). Su nombre oficial es "Ingeniería de Software – Perfiles de ciclo de vida para entidades muy pequeñas" [10].

### 3.7.1 Estructura

El estándar ISO 29110 se divide, según el público objetivo, en las secciones indicadas en la tabla 3.12 y la figura 3.4.

Tabla 3.12 ISO/IEC 29110 Público Objetivo

| ISO/IEC 29110 | Título | Público Objetivo |
|---|---|---|
| **Parte 1 (ISO/IEC 29110-1)** | Información General | VSEs, asesores, productores estándar, vendedores de herramientas y vendedores de metodologías. |
| **Parte 2 (ISO/IEC 29110-2)** | Marco y Taxonomía | Productos estándar, vendedores de herramientas y vendedores de metodología. |
| **Parte 3 (ISO/IEC 29110-3)** | Guía de Evaluación | Asesores y VSEs. |

| ISO/IEC 29110 | Título | Público Objetivo |
|---|---|---|
| **Parte 4 (ISO/IEC 29110-4)** | Especificaciones de Perfiles | Productores estándar, vendedores de herramientas y vendedores de metodologías. |
| **Parte 5 (ISO/IEC 29110-5)** | Guía de Gestión e Ingeniería | VSEs. |

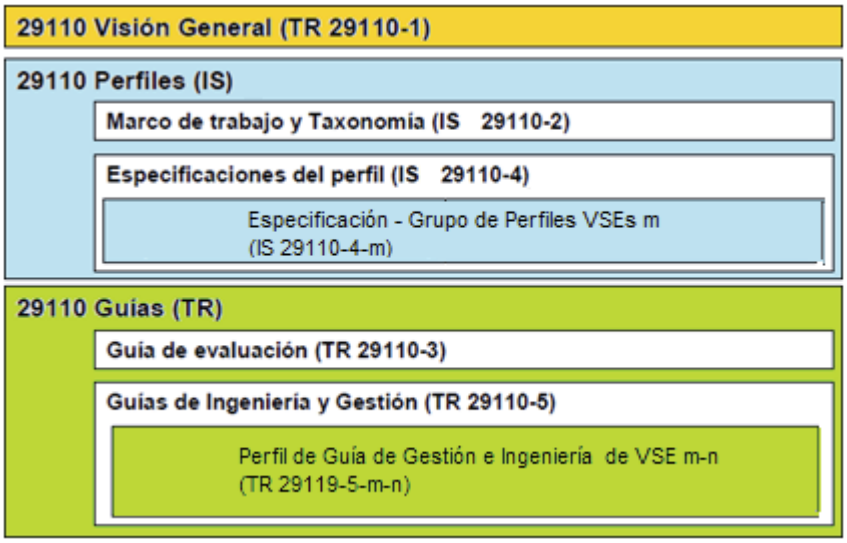


Figura 3.4 Estructura ISO 29110.

La numeración del estándar funciona de la siguiente forma: Si se añade un nuevo perfil para su especificación, ISO/IEC 29110-4 e ISO/IEC 29110-5 se pueden modificar sin alcanzar el resto del estándar, pero modificando las secciones 4 y 5 de la forma ISO/IEC 29110-4-m e ISO/IEC 29110-5-m-n.

#### 3.7.1.1 ISO/IEC 29110-1 Información general

Se definen los términos del negocio relacionados a las VSEs. Se explican procesos, ciclos de vida y conceptos de estandarización y a nivel general todo el estándar ISO/IEC 29110. También se presentan características y requerimientos de las VSEs, y se explican los perfiles específicos, documentos y guías para las VSEs.

#### 3.7.1.2 ISO/IEC 29110-2 Marco y taxonomía

Se presentan los conceptos de Perfiles Estandarizados de Ingeniería de Software y términos comunes de los Conjuntos de Documentos de Perfiles para VSEs. Se establece la lógica tras la definición y aplicación de perfiles estandarizados. Además, se especifican los elementos comunes para la estandarización de perfiles (estructura, conformidad y evaluaciones) y se presenta la taxonomía de los perfiles ISO/IEC 29110.

#### 3.7.1.3 **ISO/IEC 29110-3 Guía de evaluación**

Se definen las guías de los procesos de evaluación y cumplimiento de requerimientos necesarios para alcanzar las mejoras. Además, contiene información que puede ser útiles para desarrolladores de métodos y herramientas de mejoras. ISO/IEC 29110-3 está dirigido a personas relacionadas con procesos de evaluación.

#### 3.7.1.4 **ISO/IEC 29110-4 Especificación de perfiles**

Provee la especificación para todos los perfiles del *Grupo de Perfiles Genéricos*. Este grupo es aplicable a las VSEs que no desarrollan productos críticos de software. Los perfiles se basan en sub-grupos que elementos de estándar. Los perfiles de VSEs se aplican a autores y proveedores de guías, herramientas y materiales de asistencias.

#### 3.7.1.5 **ISO/IEC 29110-5 Guía de gestión e ingeniería**

Contienen directrices de aplicación sobre cómo aplicar los procesos sugeridos. Incluye guías de gestión y de ingeniería para la aplicación de perfiles. A continuación, de describe en más detalle.

### 3.7.2 Información General

El perfil básico de Gestión e Ingeniería de VSEs se aplica a una entidad muy pequeñas (empresa, organización, departamento o proyecto de hasta 25 personas) dedicada al desarrollo de software. Los proyectos pueden tratarse de desarrollo interno o para clientes externos. El contrato interno entre el equipo del proyecto y su cliente no tiene que ser explícito.

La Guía ofrece gestión de proyectos y procesos de aplicación de programas que integran las prácticas de mejoras, ciclo de vida del software, procesos de software y contenido de sistemas. ISO/IEC 29110-5 está destinada a ser utilizada por las VSEs para establecer procesos e implementar cualquier enfoque o metodología de desarrollo, incluyendo, por ejemplo: ágil, prueba evolutiva, incremental, impulsado desarrollo, entre otros.

A través del uso de la guía se espera obtener beneficios en los siguientes aspectos:

- Producto en sintonía con los requerimientos del usuario.
- Un proceso de gestión disciplinado que proporciona visibilidad de los proyectos y acciones correctivas del proyecto ante problemas y desviaciones.
- Un proceso sistemático de implementación de software que satisfaga las necesidades del cliente y garantice la calidad productos.

Para utilizar la guía se necesita cumplir con las siguientes bases:

- Declaración del trabajo del proyecto documentado.
- Equipo de trabajo, incluyendo jefe de proyecto, asignado y entrenado
- Contar con buenos servicios e infraestructura para comenzar el proyecto.

El propósito de los procesos de gestión de proyecto es establecer y realizar en forma sistemática las tareas propias de la implementación de un proyecto software, de forma de cumplir con los objetivos, calidad, tiempos y costos esperados.

## 3.8 ISO 29119

Actualmente en desarrollo, su principal objetivo es generar un estándar que reúna vocabulario, procesos, documentación y técnicas para el ciclo de vida del testing [11]. Este estándar estará dirigido para cualquier proyecto de desarrollo o mantención de software en una pequeña o mediana empresa. Estará compuesto de cuatro partes.

### 3.8.1 Conceptos y Vocabulario

El objetivo de esta primera parte es dar una visión general de la norma y de los conceptos generales de pruebas de software con el fin de proporcionar un vocabulario de términos relacionados al proceso de testing a lo largo de todo el ciclo de vida de los proyectos. Se prevé que en esta parte se incluyan, en su versión final, los siguientes temas:

- Introducción al testing
- Rol de verificación y validación
- Testing como heurística
- Pruebas exhaustivas
- Testing en el contexto de una organización y proyectos
- El proceso de testing
- Proceso generic de testing en el ciclo de vida
- Desarrollo de sub-procesos en proyectos y sus resultados
- Curso de mantenimiento y sus resultados

- Procesos de soporte para el ciclo de vida
- Testing basado en riesgos
- Sub-procesos del testing
- Objetivos del testing
- Ítems de testing
- Características de testing de calidad
- Testing básico
- Re-testing (repeticiones) y testing regresivo
- Técnicas de testing
- Enfoques de testing, incluyendo: basado en riesgo, basado en requerimientos, testing analítico, basado en modelos, planeado, no planeado y automático
- Métricas y medidas
- Testing en diferentes modelos de ciclos de vida, incluye:, ágil, evolutivo y secuencial

### 3.8.2 Proceso de Testing

Define un modelo de proceso de pruebas genérico a utilizar dentro de proyectos de desarrollo. Este proceso cubre los siguientes aspectos:

- Especificaciones del testing organizacional
- Gestión de pruebas
- Proceso dinámico de pruebas (incluye diseño e implementación de testing, instalación de entorno de pruebas, ejecución y reporte de pruebas)

### 3.8.3 Documentación de Testing

Esta parte cubrirá la documentación de prueba a través del ciclo de vida completo testing. Utilizará como base el estándar IEEE 829 [12]. Incluirá plantillas que se pueden personalizar y que abarcan todas las fases del proceso de pruebas, incluyendo:

- Documentación del proceso de testing organizacional
    - Política de testing organizacional
    - Estrategia de testing organizacional

- Documentación de la gestión del proceso de testing
    - Planeación de testing
    - Reporte del estado de testing
    - Informe final de testing

- Documentación del proceso de testing dinámico
    - Especificación de diseño de testing
    - Especificación de casos de pruebas
    - Especificación de procedimientos de pruebas
    - Requerimientos de datos de pruebas
    - Detalle de requerimientos de entorno de pruebas
    - Informe de disponibilidad de entorno de pruebas
    - Pruebas de resultados
    - Resultados de pruebas
    - Registro de ejecución de pruebas
    - Reporte de incidentes de pruebas

### 3.8.4 Técnicas de Testing

Se abordan las técnicas para los distintos tipos de testing, tales como: revisiones estáticas, pruebas dinámicas, test funcional y no funcional. La base para esto es la norma BS-7925-1/2 [13] de La Sociedad Británica de Computación. Incluirá:

- Técnicas de testing basadas en especificaciones
- Técnicas de testing estructuradas

Además, proveerá información en cuanto a definiciones de una variedad relaciones de calidad con distintos tipos de testing.

## 3.9 TPI

El modelo TPI fue desarrollado basándose en conocimiento práctico y ofrece un punto de vista en la madurez de los procesos de testing dentro de la organización. Es así, que este modelo ayuda a definir pasos para una mejora gradual y controlada de los procesos del testing [14].

El modelo define dos niveles de testing que se deben realizar dependiendo de los requerimientos de las pruebas:

- Pruebas de bajo nivel: Este nivel de pruebas es ejecutado, principalmente, por los mismos desarrolladores. Se trata de pruebas a componentes separados del sistema o

en combinación entre éstos. En esta etapa se realizan las pruebas unitarias y modulares.
En la medida que los componentes testeados cumplen con los requerimientos establecidos, se suman más componentes del sistema de forma de realizar pruebas de integración.

- Pruebas de alto nivel: Una vez que las pruebas de bajo nivel han sido ejecutadas y se han corregido los defectos encontrados, se realizan pruebas de sistemas para determinar globalmente el cumplimiento de los requerimientos. Una vez que las pruebas del sistema se terminan, el cliente tiene acceso a un sistema de pruebas para realizar las pruebas de aceptación. La ejecución de las pruebas de aceptación requiere un ambiente que debe ser representativo del entorno de producción.

### 3.9.1 Problemas del Testing

TPI detecta tres problemas principales relacionados al testing que dificultan su correcta ejecución en las organizaciones:

- Testing primitivo: Se trata del proceso de testing que comienza justo antes de iniciarse la codificación del sistema. Este proceso termina una vez que no se detectan más defectos, lo que puede provocar una falsa sensación de satisfacción de requerimientos, y más tarde puede significar grandes costos de mantenimiento y re-procesos.

- Estado actual: El test se planea y prepara antes de su ejecución y se basa en las especificaciones de requerimientos. El testing suele ser breve en el tiempo, realizado por pocas personas y no dispone de grandes recursos. El testing es considerado tarde en el ciclo de vida del proyecto y por esto suele traer costos adicionales. Debido a estas situaciones, no se puede determinar la calidad del sistema una vez que finaliza el testing.

- Nuevos desarrollos: Hoy en día, los procesos de desarrollo deben ser más rápidos para cumplir con la demanda del mercado. Sin embargo, al haber más desarrollo no existe certeza de que exista una disminución de errores en el proceso. Es decir, el testing podría no avanzar al mismo ritmo que se desarrolla, lo que a futuro puede traer graves problemas de calidad en los productos.

### 3.9.2 Pasos para la Mejora del Proceso de Testing

- Determinar el objetivo y área a considerar: ¿Qué se busca? ¿un testing más rápido? ¿Más barato o de mayor alcance?

- Determinar la situación actual: Determinar fortalezas y debilidades del proceso actual de testing.

- Determinar la situación requerida: Se determina la situación requerida basándose en los pasos anteriores.

- Cambios de implementación: Las acciones de mejoras sugeridas se deben llevar a cabo de acuerdo a un plan y verificando que se cumplan los objetivos propuestos.

### 3.9.3 Estructura de TPI

TPI considera veinte *áreas clave* que en la medida que la organización gestione, éstas aumentarán su *nivel de madurez*. Para cada nivel de madurez, se debe alcanzar un *objetivo* el que debe ser comprobado mediante los *puntos de control*. Un punto de control, es la generación de una pregunta en base al logro del objetivo pretendido en el nivel de cierta área clave. En la tabla 3.13 se muestran las áreas claves y sus objetivos para cada nivel.

Tabla 3.13 Áreas claves y niveles TPI.

| | **Nivel** | | | |
|---|---|---|---|---|
| **Área Clave** | A | B | C | D |
| **Estrategia del testing** | Estrategia para test simple de alto nivel | Estrategia combinada para pruebas de alto nivel | Estrategia combinada para pruebas de alto y bajo nivel | Estrategia combinada para todas las pruebas y evaluaciones |
| **Modelo del ciclo de vida** | Planeación, especificación y ejecución | Planeación, preparación, especificación, ejecución y finalización | - | - |
| **Momento de participación** | Finalización de base del testing | Inicio de la base del testing | Inicio definición de requerimientos | Iniciación del proyecto |
| **Estimación y planeamiento** | Estimación y planificación justificadas | Estimación y planificación estadísticamente justificada | - | - |
| **Especificaciones técnicas del testing** | Técnicas informales | Técnicas formales | - | - |
| **Técnicas estáticas de testing** | Inspección a la base del testing | Listas de chequeos | - | - |
| **Métricas** | Métricas para el producto | Métricas para procesos | Métricas para el sistema | Métricas organizacionales |
| **Herramientas de testing** | Herramientas de planeación y control | Herramientas de análisis y ejecución | Automatización extensiva de los procesos de testing | - |

| Área Clave | Nivel | | | |
|---|---|---|---|---|
| | A | B | C | D |
| **Entorno de pruebas** | Entorno gestionado y controlado | Testing en el ambiente adecuado | Entorno a pedido | - |
| **Entorno de oficina** | Entorno adecuado y puntual | - | - | - |
| **Compromiso y motivación** | Asignación de presupuesto y tiempo | Testing integrado en la organización del proyecto | Ingeniería de testing | - |
| **Funciones y capacitación de testing** | Encargado del testing y testers | Gestión metódica, técnica y con apoyo | Aseguramiento de calidad | - |
| **Ámbito de metodología** | Proyecto específico | Genérico de la organización | Optimización organizacional | - |
| **Comunicación** | Comunicación interna | Comunicación del proyecto (defectos, control de cambios) | Comunicación entre la organización sobre la calidad del testing | - |
| **Reportes** | Defectos | Progresos, actividades y defectos prioritarios | Riesgos y recomendaciones | Rol de procesos de mejoras para el software |
| **Gestión de defectos** | Gestión interna de defectos | Gestión de defectos extendida | Gestión de defectos de proyectos | - |
| **Gestión de testware** | Gestión interna de testware | Gestión externa de la base y objetivos del testing | Testware reutilizable | Requerimientos de trazabilidad para los casos de pruebas |
| **Gestión de proceso de testing** | Planeación y ejecución | Planeación, ejecución, monitoreo y ajustes | Monitoreo y ajustes organizacional | - |
| **Evaluación** | Técnicas de evaluación | Estrategias de evaluación | - | - |
| **Testing de bajo nivel** | Ciclo de vida: planeación, especificación y ejecución | Técnicas de caja blanca | Estrategia de testing | - |

Además, para determinar la escala de madurez a nivel organizacional (escala de 13 niveles), se debe cumplir con ciertos niveles en cada área clave. Esto se presenta en la "*Matriz de Madurez de Test"* presentada en tabla 3.14.

Tabla 3.14 Matriz de madurez de test.

| Área Clave | Escala Organizacional | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| **Estrategia del testing** | | A | A | A | A | A | B | B | B | B | C | C | D | D |
| **Modelo del ciclo de vida** | | A | A | A | B | B | B | B | B | B | B | B | B | B |
| **Momento de participación** | | | A | A | A | A | B | B | B | B | C | C | D | D |
| **Estimación y planeamiento** | | | | A | A | A | A | A | A | A | B | B | B | B |
| **Especificaciones técnicas del testing** | | A | A | B | B | B | B | B | B | B | B | B | B | B |
| **Técnicas estáticas de testing** | | | | | A | A | B | B | B | B | B | B | B | B |
| **Métricas** | A | A | A | A | A | A | A | A | B | B | B | C | C | D |
| **Herramientas de testing** | | | | A | A | A | A | B | B | B | C | C | C | C |
| **Entorno de pruebas** | | | | A | A | A | A | B | B | B | B | B | B | C |
| **Entorno de oficina** | | | | A | A | A | A | A | A | A | A | A | A | A |
| **Compromiso y motivación** | | A | A | A | A | B | B | B | B | B | B | C | C | C |
| **Funciones y capacitación de testing** | | | | A | A | A | B | B | B | C | C | C | C | C |
| **Ámbito de metodología** | | | | | A | A | A | A | A | A | B | B | B | C |
| **Comunicación** | | | A | A | B | B | B | B | B | B | B | C | C | C |
| **Reportes** | | A | A | B | B | B | C | C | C | C | C | D | D | D |
| **Gestión de defectos** | | A | A | A | A | B | B | C | C | C | C | C | C | C |
| **Gestión de testware** | | | A | A | A | B | B | B | B | C | C | C | C | D |
| **Gestión de proceso de testing** | | A | A | B | B | B | B | B | B | B | B | C | C | C |
| **Evaluación** | | | | | | | A | A | A | B | B | B | B | B |
| **Testing de bajo nivel** | | | | | A | A | B | B | C | C | C | C | C | C |

## 3.10 Comparativa de Modelos

Ya revisado los principales modelos de testing, a continuación se presenta un esquema comparativo entre ellos. Esto, con el objetivo de poder obtener las ventajas que un modelo puede ofrecer sobre otro y así contar con una mejor herramienta que permita discriminar los elementos que debe contener el modelo a proponer. En la tabla 3.15 se presentan siete modelos de testing, descritos anteriormente, y distintos aspectos relacionados a la mejora y gestión del testing, de esta forma se grafica qué modelos contemplan los aspectos considerados para la comparación.

Tabla 3.15 Comparación de aspectos de modelos de testing.

| | TMM | TMMi | TIM | TestPAI | Tutelkán | ISO 29119 | TPI |
|---|---|---|---|---|---|---|---|
| **Pequeña Empresa** | | | | ✔ | ✔ | ✔ | |
| **Ambiente** | ✔ | ✔ | ✔ | | ✔ | | |
| **Planeación** | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| **Información/Documentación** | ✔ | | ✔ | | ✔ | ✔ | ✔ |
| **Diseño de pruebas** | ✔ | ✔ | ✔ | ✔ | | | |
| **Testware** | ✔ | | ✔ | | ✔ | | ✔ |
| **Política Organizacional** | | ✔ | ✔ | ✔ | | ✔ | ✔ |
| **Control de testing** | | ✔ | ✔ | ✔ | | | |
| **Ejecución de testing** | | ✔ | | ✔ | | ✔ | |
| **Organización de personal** | | ✔ | ✔ | | ✔ | | |
| **Capacitación de personal** | | ✔ | ✔ | | | | ✔ |
| **Integración con el ciclo de vida** | | ✔ | ✔ | | | ✔ | |
| **Testing no funcional** | | ✔ | | | ✔ | ✔ | ✔ |
| **Métricas** | | ✔ | ✔ | | | | |
| **Prevención de defectos** | ✔ | | ✔ | | | | ✔ |
| **Optimización continua** | ✔ | ✔ | ✔ | | | | |
| **Reutilización de testing** | | | ✔ | ✔ | ✔ | | ✔ |
| **Gestión de riesgos** | | | ✔ | ✔ | | ✔ | |
| **Necesidades organizacionales** | ✔ | ✔ | ✔ | ✔ | | | |
| **Revisiones estáticas** | | | ✔ | | ✔ | ✔ | ✔ |
| **Pruebas funcionales** | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| **Adaptación al ciclo de vida** | | | | | | ✔ | ✔ |
| **Comunicación** | ✔ | ✔ | ✔ | | | | ✔ |
| **Testing de bajo nivel** | ✔ | ✔ | ✔ | | | | ✔ |

## 3.11 Pruebas Funcionales Evolutivas

Se sabe que la cantidad de casos de pruebas que se pueden generar para un sistema es demasiado grande (posiblemente infinito), es por esto que se comienzan a utilizar nuevas técnicas para automatizar la generación de los casos de prueba y conseguir pruebas significativas para el sistema. Así, optimización estocástica y búsquedas inteligentes se han aplicado con fines de investigación (éstas últimas mediante algoritmos evolutivos). La idea es simple, se considera la prueba deseada como un problema de optimización y mediante algoritmos evolutivos se buscan datos idóneos para las pruebas [15].

En concreto, se define el objetivo a optimizar (lo que se quiere probar en el sistema) y se genera un conjunto inicial de datos (generalmente aleatorio), con estos datos el sistema se somete a prueba y luego se analizan los resultados para determinar el valor de adecuación necesario. Los datos según su valor de adecuación se someten a procesos de combinación y mutación para obtener datos hijos y así se vuelve a probar el sistema. El proceso se repite hasta alcanzar el objetivo deseado o alguna otra condición de parada definida. En otras palabras, al realizar pruebas funcionales evolutivas se buscan datos de prueba con los cuales el sistema testeado sea defectuoso. Cuando el óptimo es alcanzado es porque se ha descubierto un defecto en el sistema.

Existen herramientas para el entorno de estas pruebas evolutivas, por ejemplo el proyecto EvoTest [16] implementa un "Marco de trabajo evolutivo para testing", el cual es una extensión de Eclipse, que utiliza un generador evolutivo.

A pesar de los prometedores resultados que se han obtenido a nivel de investigación, la gran desventaja que se tiene es que la implementación de pruebas funcionales requiere de gran tiempo y costos para su implantación en el proceso de desarrollo, además de personal con grandes conocimientos de computación evolutiva. Por otro lado, los casos prácticos realizados hasta el momento sólo se han limitado a experimentos realizados en grandes organizaciones, específicamente en la industria automotriz en Estados Unidos.

## 3.12 Testing y Metodologías Ágiles

Según [17] se definen cuatro etapas para el proceso de pruebas en métodos convencionales de desarrollo: pruebas unitarias (PU), pruebas de integración (PI), pruebas de aceptación (PA) y pruebas de sistema (PS). Si bien, hoy en día se da cada vez más importancia al testing, éste depende directamente del resto de las actividades del ciclo de vida, lo que en consecuencia provoca que el testing sea un complemento al proceso de desarrollo [18]. La figura 3.5 representa el enfoque tradicional de testing, en donde las pruebas se enfocan en un orden inverso a la concepción del sistema.

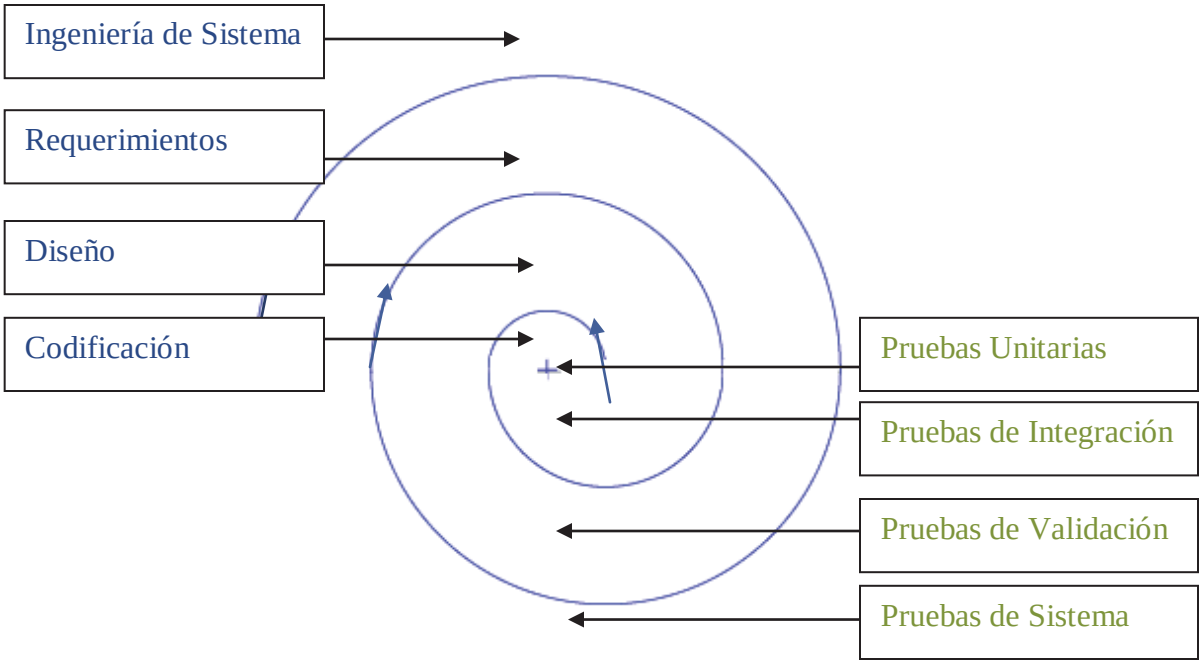


Figura 3.5 Enfoque tradicional de testing.

Normalmente, en procesos de desarrollo de metodologías ágiles, particularmente XP, los requerimientos del usuario se representan por historias de usuario que a su vez definen criterios de aceptación y que éstos son la base de los casos de prueba. Por lo tanto, las pruebas de aceptación pueden formularse desde una temprana etapa del ciclo de desarrollo. Es así, que el testing pasa a ser la base del desarrollo en las metodologías ágiles. Estas metodologías no consideran el testing como barreras a superar para alcanzar la validación del sistema, sino que presentan distintos enfoques del proceso de desarrollo que vienen determinados por los tipos de pruebas que se realizan.

Las metodologías ágiles utilizan las pruebas unitarias y de aceptación como principales herramientas para dar soporte a los enfoques de pruebas. Y por otro lado, las pruebas de integración y las pruebas de sistema se aplican de forma incremental en el proceso de desarrollo, esto es, que en la medida que nuevo código se desarrolla, éste debe ser sometido a testing junto al resto del sistema.

Se hace necesario disponer de recursos tecnológicos que permitan automatizar la ejecución de las pruebas, dado la gran cantidad de tiempo invertido en testing. Además, será necesario automatizar, en la medida que sea posible, la toma de medidas de calidad, la generación de documentación y otras actividades vitales para el testing.

# 4 Modelo Propuesto

El modelo de mejora propuesto se basa en siete secciones de aplicación: Política Organizacional, Testing de Bajo Nivel, Diseño, Integración al proyecto, Prevención, Testing no Funcional y Mejora Continua. Éstas deben ser gestionadas para mejorar la calidad del testing en la pequeña empresa. Las secciones de aplicación se proponen en un cierto orden de aplicación, sin embargo éste no es estricto y puede variar dependiendo de las condiciones y necesidades de cada organización. Además, existe una relación bidireccional entre cada sección, pues en la medida que se trabajen nuevas secciones las anteriores requerirán modificaciones para alcanzar una coordinación a nivel total de la empresa. El trabajo de mejoras en las distintas secciones está basado en dos recursos fundamentales para el mejoramiento del testing de toda organización: Organización de Personal y Documentación. Estos recursos son elegidos tomando en cuanta que son los activos más accesibles en las organizaciones y permitirán una adecuada implementación y formalización de las actividades de mejoras de testing en una pequeña empresa.

La figura 4.1 muestra las secuencia (flexible) de secciones de mejoras de testing y su relación con las recursos fundamentales.

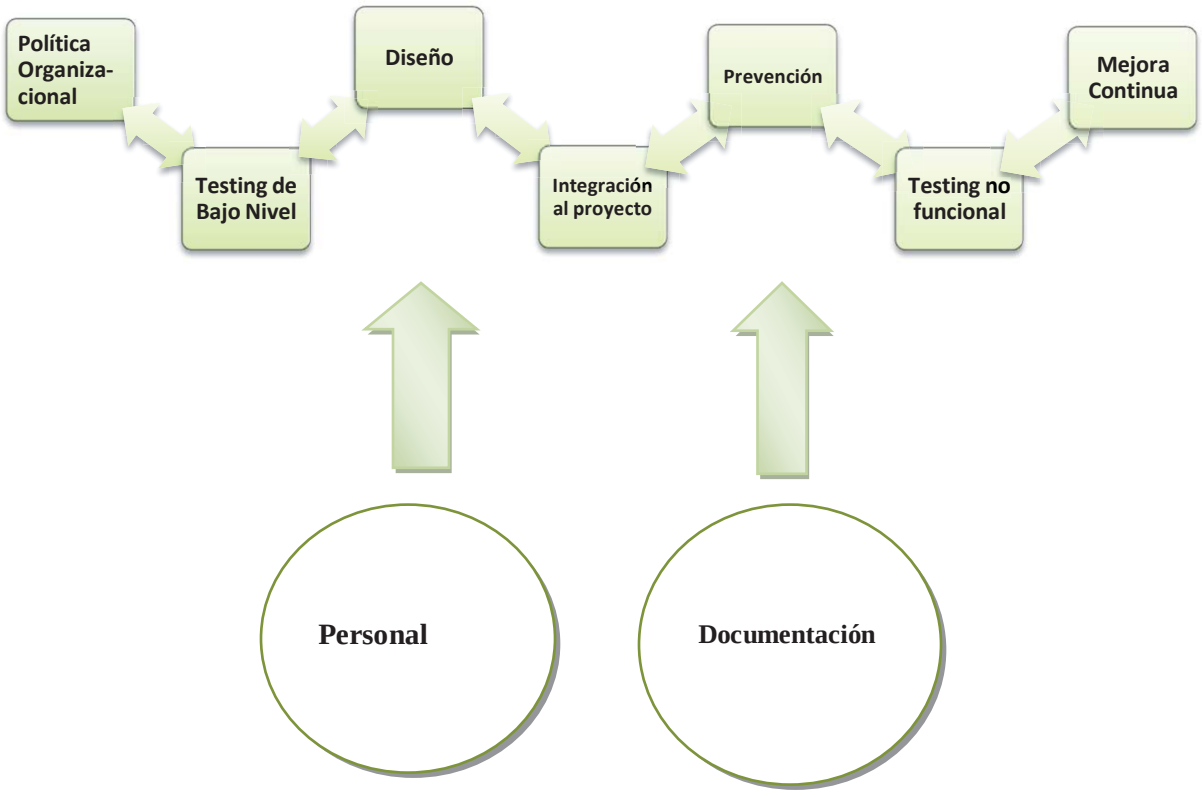


Figura 4.1 Estructura del modelo propuesto.

## 4.1 Recursos Fundamentales

Estas prácticas son la base del modelo de mejoramiento propuesto, es decir, cada una de las áreas que se presentan basarán sus actividades de mejoras en la gestión del personal y los métodos de documentación.

Algunas áreas tienen una mayor o menos relación con las prácticas fundamentales dependiendo de sus actividades y objetivos.

### 4.1.1 Personal

Se entiende que el equipo humano son las personas encargadas de trabajar en las mejoras del testing propuesto. Es por esto, que una adecuada organización del personal y sus recursos permitirá que las mejoras se puedan implementar de forma correcta y que el camino a la mejora se complete y no quede detenido por falta de personal o por el desconocimiento de roles de los trabajadores en el mejoramiento.

La organización debe estar preparada para colocar sus recursos en las actividades correctas y poder incorporar nuevos elementos, según sea necesario, además de buscar reemplazo de personal cuando se haga necesario.

La gestión del personal es relevante, ya que es necesario lograr que el equipo se comprometa con el proceso de mejoramiento y que muchas de sus prácticas habituales sean corregidas para alcanzar el estado deseado.

### 4.1.2 Documentación

Formalizando cada uno de los cambios y acciones realizadas, se puede mantener una comunicación y retroalimentación entre todas las partes involucradas, es por esto que en cada una de las etapas se debe implementar un adecuado método de documentación que permita cumplir con este objetivo.

La inversión de recursos en la mejora del material es fundamental para dejar registro del comportamiento de la organización durante el camino a la mejora. Además, una adecuada documentación permitirá que en la medida que nuevas personas se unan a la organización, éstas puedan comprender de forma correcta cómo se realizan las pruebas.

## 4.2 Secciones de Mejora

La elección de las áreas de mejora se realizó en base a adaptaciones de los distintos modelos descritos en el capítulo 3 y experiencias como las de [1]. Las áreas son distintos aspectos en los que se propone enfocar las mejoras en un determinado momento, pero tal y como ya se ha señalado, la cronología de aplicación es tan sólo una propuesta, ya que ésta es flexible y puede ser adaptada según las necesidades propias de la organización donde se

aplique. Por lo tanto, éste no es un modelo de madurez y las distintas áreas no representan necesariamente niveles a alcanzar.

Para comprender bien el funcionamiento de este modelo, se debe señalar que al ser un modelo de mejoras, se asume que ya existe un proceso de testing en la organización que debe ser mejorado y no se pretende comenzar el testing desde cero, es decir, en la medida que el modelo se aplique las viejas prácticas deberán convivir con las prácticas propuestas. Lo anterior, quiere decir que el estar enfocando los esfuerzos en una determinada área del testing no implica que el resto de los conceptos o áreas no se estén implementando de alguna manera.

Las áreas de mejoras se presentan con una descripción de qué es lo que representan en el testing y su eventual relación con otras áreas, luego se analizan sus relaciones con las prácticas fundamentales y finalmente se formalizan algunas actividades recomendadas para su gestión.

### 4.2.1 Política Organizacional

La política organizacional es la base del proceso de mejoramiento del testing, pues la política organizacional determina qué es lo que se quiere mejorar, cuáles son las metas y expectativas de la mejora, con qué recursos se cuenta para mejorar el testing, es decir, se define la política organizacional respecto al tema.

En un comienzo, la política organizacional tenderá a ser amplia e incluso ambigua debido a que muchas de las decisiones de las mejores del testing se tomarán en base a las experiencias obtenidas en la medida que se vaya mejorando el proceso. Esto, no quiere decir que la política no deba ser respetada y que pueda pasarse sobre ella, ya que para una buena mejora de testing se requiere un compromiso y coordinación total de la organización.

La política organizacional podrá variar en la medida de que se mejoren las demás áreas de mejoras y que las tecnologías varíen y permitan la inclusión de nuevas herramientas para los procesos de pruebas. En base a esto, se espera que la variación en la política sea más frecuente en las primeras etapas de mejoras de testing y luego, en la medida que las mejoras se consoliden, los cambios tiendan a desaparecer y originarse para situaciones de reformas muy puntuales.

Esta área de mejora deberá definir los recursos que se está dispuesto a utilizar en las mejoras que se quieren lograr, considerando de ser necesaria la contratación de personal de apoyo directo para las mejoras. Además, la política organizacional debe definir los criterios que definan una correcta implantación de las mejoras, ya que de no cumplirse las expectativas deberá echarse pie atrás y asumir los errores cometidos que no llevaron a la meta deseada.

#### Personal

La política Organizacional tiene directa relación con la organización del personal, ya que dependiendo de las decisiones tomadas se deberá organizar el personal disponible para la mejora del testing.

Una propuesta inicial es que la política organizacional sea capaz de instaurar una nueva forma de ver las pruebas en el proceso de desarrollo que se realiza hasta ese momento. El personal debe aprender a valorar las mejoras que se proponen y ver más allá de un simple trámite o la búsqueda de alguna certificación que permita publicitar mejor su negocio. Por otro lado, se deben definir de manera clara los roles de los distintos actores en el ciclo de desarrollo y qué tareas le corresponde a cada uno en la mejora del testing.

El conseguir una correcta organización de las personas, permitirá administrar correctamente gran parte de los recursos y los esfuerzos para lograr las mejoras pretendidas.

**Documentación**

Toda política organizacional debe estar correctamente documentada y disponible para todos los involucrados, ya que es la mejor forma de que las decisiones organizacionales sean consideradas a nivel global de la organización y que éstas sean respetadas y consideradas en las mejoras implementadas.

La política organizacional deberá definir de qué manera se documentan los hechos y trabajos que se realicen para las mejoras, de tal manera que la documentación permita la reutilización de casos de pruebas, por ejemplo, en futuros proyectos. Además, los errores cometidos en la mejora y ejecución de pruebas deben ser archivados y estar disponibles para evitar repetir esfuerzos ya realizados.

**Actividades recomendadas**

- Establecer políticas y objetivos del mejoramiento de testing
- Establecer política de planificación de pruebas
- Definición de estándares
- Política de organización de personal
- Política de organización de recursos
- Definición de evaluación de mejoras
- Control de política organizacional

### 4.2.2 Testing de Bajo Nivel

Una vez que se han realizado los lineamientos que pretenden llevar a la organización a la mejora del testing se debe empezar a ejecutar cambios más concretos que impliquen un cambio en la realización normal de las tareas. Para esto, una primera propuesta es la de mejorar el testing de bajo nivel, en específico el testing unitario. La idea es que en esta área se permita a los desarrolladores contar con un grado de libertad tal, que los mismos

programadores puedan definir la forma en que realizan las pruebas unitarias a sus módulos desarrollados.

Los programadores, de captar la nueva filosofía de hacer las cosas de mejor forma en la Política Organizacional, deben sentirse comprometidos a mejorar la calidad de sus trabajos. La idea es que los programadores prueben debidamente sus trabajos antes de liberarlos y que sean sometidos a pruebas de integración y de sistema. Esto, se logra permitiendo al desarrollador decidir qué herramientas implementar (dentro de los recursos disponibles), el tiempo dedicado y a la vez, en medida de los resultados obtenidos, poder ayudar en la mejora del testing de mayor nivel.

El testing unitario debidamente planeado y ejecutado, permite una detección temprana de fallas en el sistema, y por lo tanto podrá prevenir grandes costos de mantención de problemas que se descubran en etapas más avanzadas del proceso de desarrollo. El personal dedicado a estas tareas deberá tener una constante preocupación por mejorar sus prácticas y contribuir a la mejora general de la organización. Una buena forma de permitir a estos testers mejorar sus conocimientos es permitirles poder capacitarse en técnicas de testing, aunque esto no necesariamente signifique para la empresa tener que invertir en estudios y cursos, ya que se puede sacar provecho a la gran cantidad de información disponible en la red.

Se podría tender a pensar que ésta nueva tarea de los desarrolladores puede poner en riesgo su productividad, a favor se tiene que entre mejor sea la planeación y ejecución de las pruebas y una adecuada documentación del trabajo los futuros proyectos deberán ser apoyados por la experiencia adquirida en la buenas prácticas y en el tiempo reducir tiempos de trabajo en aspectos que ya se manejen de forma correcta.

En la medida que se consolide la nueva manera de realizar el testing unitario, se pueden agregar aspectos como la revisión de pares para fomentar el diálogo entre los trabajadores y definir nuevas ideas que puedan aportar a las mejoras.

**Personal**

La política organizacional deberá definir con claridad al personal a cargo de esta área, es decir, en la medida que evolucione esta área se debe verificar el estado del área anterior. La propuesta es que sean los mismos desarrolladores quienes implementen esta área, sin embargo si existe personal experto en pruebas y con pocas habilidades en desarrollo no tendría sentido forzar un cambio de esquema.

Se debe contemplar los recursos disponibles para el personal en la mejora del testing unitario, como por ejemplo la compra de herramientas de software o capacitación pagada para los testers. Además, se puede establecer como estrategia de mejora la disposición de algún tiempo libre de trabajo de los testers para poder dedicarlo a la elaboración de ideas o de estudio sobre mejoras del testing y su trabajo.

**Documentación**

La documentación es vital para la mantención de las mejoras que se logren con la gestión de mejoras de las pruebas unitarias. Las estrategias decididas por los trabajadores, la

utilización de software y otras herramientas deben ser reportadas e idealmente poder ser explicadas a otros miembros del equipo con el fin de generar una retroalimentación y avanzar juntos en la mejora del testing.

Una correcta aplicación de la documentación puede permitir con el tiempo la reutilización de material y una mejor adaptación al nuevo personal que pueda integrarse a la organización. Incluso, la documentación puede ser dispuesta para ser revisada por los otros niveles de testing para poder detectar debilidades que se hagan evidentes en etapas más maduras de las pruebas.

**Actividades recomendadas**

- Capacitaciones individuales
- Capacitaciones organizacionales
- Evaluaciones de pares
- Herramientas de apoyo
- Documentación de testing
- Planeación de testing

### 4.2.3 Diseño

Las pruebas de mayor nivel (de integración, de sistema y de aceptación) requieren de mayor preparación antes de ser ejecutadas, ya que deben representar las necesidades y requerimientos del usuario para poder dar de alta, si son superadas, al sistema que se está desarrollando.

El diseño debe incluir la elaboración de casos de prueba genéricos para ser adaptados a los distintos proyectos que se desarrollen en la organización. Además un buen diseño de pruebas debe considerar un esfuerzo en cuanto a la definición del procedimiento mismo de las pruebas, es decir, la forma en que las pruebas deben ejecutarse y cómo se evaluarán sus resultados.

Los requisitos del usuario tienen un rol fundamental en el diseño de las pruebas de alto nivel, ya que estas pruebas deberán definir si el sistema que se está desarrollando cumple con las necesidades y expectativas del cliente. Debido a esto, el diseño deberá estar relacionado con el levantamiento de requisitos que los analistas realicen al comienzo del proyecto, es decir, los diseñadores de testing deberán saber interpretar los requerimientos formalmente aceptados en la etapa de análisis.

**Personal**

Para el personal a cargo del diseño de pruebas es ideal que sea con dedicación exclusiva a las tareas de testing. Un líder asignado a la tarea, debe definir qué tan especifico debe ser el diseño de pruebas, aunque es lógico suponer que entre más detallado y mayor tiempo se le dedique los resultados sean mejores. Lo recomendado es que en la medida que las mejoras se vayan implementando en la organización, aumentar el personal en ésta área o invertir en mejorar la preparación de éste.

**Documentación**

Todo diseño bien documentado mejora la explicación de éste, es más si los diseños están bien documentado podrán ser reutilizados en trabajos futuros. La documentación debe incluir casos de pruebas, especificaciones técnicas sobre su implementación y otros aspectos. Es muy relevante generar un documento de pruebas y requerimientos que permita definir requerimientos comunes de los clientes y las pruebas que deben superar para ser considerados cumplidos.

Tal y como es común en las áreas anteriores, la documentación al estar disponible a los pares, y al resto de la organización, permite generar un ambiente de crecimiento en conjunto y da la posibilidad a mejoras globales.

**Actividades recomendadas**

- Diseño de casos de pruebas
- Documentación pruebas-requerimientos
- Especificación de procedimientos de ejecución
- Control de diseño

### 4.2.4 Integración al Proyecto

Hasta las áreas anteriores se ha subentendido al testing como una etapa que se hace presente en el ciclo de desarrollo a partir de la implementación. Se entenderá, entonces, por integración al concepto de hacer del testing una etapa transversal del ciclo de desarrollo, es decir, el testing debe ser considerado desde la toma de requerimientos hasta la ejecución de las pruebas y la eventual aprobación por parte del cliente.

Se debe entender que los testers son parte del personal del área de calidad de la empresa, bajo este supuesto es lógico hacerlos parte desde etapas tempranas del ciclo de desarrollo. Su experiencia debe aportar la detección temprana de falencias que el sistema puede contener y ayudar en la estimación de costos del proyecto, ya que sus conocimientos les permitirán saber, por ejemplo, que tan complicado es satisfacer determinados requerimientos de sus clientes.

Dependiendo de cómo se gestione la integración y de las prácticas propias de la organización, el diseño de pruebas tenderá a variar y a hacerse más específico en relación a los requerimientos del proyecto.

La integración trae como consecuencia una unión explicita entre distintos niveles del ciclo de desarrollo, y que permitirá poder mejorar la comunicación a nivel general de la organización lo que dará lugar a mayores revisión de pares, evaluaciones conjuntas y otras prácticas de mejoramiento.

Cabe destacar que en organizaciones en que el ciclo de desarrollo sea de algún estilo ágil, la integración será aún más necesaria e incluso será una de las áreas más importantes a abordar de este modelo. Como bien se explicó en el marco teórico de este informe, las metodologías ágiles requieren de un correcto proceso de testing que apruebe los requerimientos formales del usuario y permita el avance rápido en el ciclo de desarrollo.

**Personal**

Para la correcta integración del testing al ciclo de desarrollo todo el personal involucrado en él deberá asumir una actitud de trabajo que probablemente sea nueva para la organización. Se necesita gran comunicación entre los distintos niveles del proyecto y serán necesarias reuniones de coordinación para proyectos específicos y de preparación para futuros clientes.

No se requieren, en principio, el aumento de recurso humano, sino más bien potenciar sus habilidades blandas de manera de manera de lograr acuerdos en grupos de trabajo que normalmente no estén muy integrados.

**Documentación**

La documentación será necesaria para formalizar los acuerdos que puedan llegar a tomarse entre los distintos grupos de trabajo, lo que se podrá ver reflejado, además, en cambios en la Política Organizacional.

A partir de la mejora de esta área la documentación de los proyectos, a nivel general, debido al cambio que implicará en la forma de desarrollar la documentación estará orientada más hacia el desarrollo en base a la aceptación de los requerimientos de los clientes y cómo se medirán éstos. Por lo tanto, un documento oficial de métricas se hace necesario.

**Actividades Recomendadas**

- Reuniones de organización entre distintos niveles
- Revisiones avanzadas de pares
- Diseño de pruebas basado en requerimientos
- Documento de métricas

### 4.2.5 Prevención

Por prevención se entenderá al enfoque que hará del testing un proceso, que además de detectar fallas en los sistemas que se están desarrollando, que permita prevenir los defectos desde la concepción misma de los proyectos. Por lo tanto, resulta evidente la cercanía que tiene esta área con la el Diseño e Integración. Además, la Política Organizacional también sufre modificaciones debido a esta nueva forma de ver y comprender el testing.

La prevención implica un cambio de enfoque en lo que, probablemente, sea el objetivo del testing en la organización. Requiere de una nueva forma de comprender el testing y la utilidad que éste presta al desarrollo. El personal debe instruirse en los métodos actuales respecto a la prevención de fallos. El gran peligro, es creer que una adecuada prevención implica la inexistencia de errores, pues como bien se sabe todo producto software tiene errores en su implementación. Es más, en la medida que la prevención se vaya adaptando al proceso de desarrollo, se hará necesario que el diseño se dedique a idear casos de pruebas más específicos que busquen poner a prueba de un modo más agresivo al sistema, ya que la cantidad de errores fáciles de detectar tenderá a desaparecer.

**Personal**

No se requieren grandes variaciones en la organización del personal de la organización, sin embargo se debe considerar la posibilidad de separar al equipo actual de diseño de testing, en un grupo encargado de la prevención de fallas y otro para la mejora de pruebas en busca de errores durante las pruebas. Idealmente, de crearse esta división, se puede considerar la rotación de roles entre estos nuevos grupos de tal manera que se potencien sus habilidades y el personal logre una mayor experiencia en el concepto global del testing.

**Documentación**

Está muy relacionado con los cambios que se definan en la política organizacional sobre cómo almacenar los métodos que permitirán adelantarse a la fallas que los sistemas puedan presentar.

Resulta muy importante almacenar en forma histórica errores que se detecten en módulos que por estrategia de testing se puedan considerar libres de fallas, pero que al ser testeados en pruebas de sistema se descubran problemas no contemplados.

**Actividades recomendadas**

- Rotación de equipos (prevención-detección)
- Capacitación en prevención
- Revisión de pares
- Creación de casos de pruebas “agresivos”

### 4.2.6 Testing No Funcional

En las áreas anteriores no se consideran las pruebas orientadas a satisfacer requerimientos no funcionales de los usuarios. El testing no funcional, además de evaluar el cumplimiento de requerimientos no funcionales, permite darle mayor valor al sistema desarrollado, pues las pruebas no funcionales pueden demostrar fortalezas del sistema en cuanto a aspectos como la robustez, confiabilidad, usabilidad y otros aspectos relacionados a la calidad.

Dependiendo de los recursos disponibles para la realización de estas tareas, debe definirse hasta qué punto se quiere testear el sistema por sobre los aspectos no funcionales explícitamente considerados en el levantamiento de requisitos. Por lo tanto, se deben discutir y decidir estos aspectos como una Política Organizacional.

El testing no funcional no debe alcanzar un mayor protagonismo que el testing funcional, sino que debe lograrse un equilibrio que permita alcanzar el nivel deseado de calidad con que la organización desea entregar sus productos a su cliente. El nivel de calidad, se verá reflejado, además, en la necesidad de soporte que los usuarios manifiesten, por lo que será relevante poder concluir ideas en base a la satisfacción expresada por parte de los clientes.

El equipo de esta área, deberá establecer un canal de comunicación directo con el área de programación, debido a que durante la implementación es que generarán los sistemas que cumplan o no con los atributos no funcionales. En otras palabras, este enlace entre los equipos permitirá aumentar la Prevención de fallos de aspectos no funcionales.

**Personal**

El personal debe ser dotado de las herramientas que permitan realizar en forma correcta el testing no funcional, los recursos destinados a estas pruebas dependerán del alcance que tengan.

Eventualmente, si la organización no cuenta con personal experto en el tema, debe considerar la posibilidad de recibir apoyo de alguien especializado. Sin embargo, normalmente toda empresa cuenta con su área de testing no funcional y con la experiencia que implica trabajar en esa área basta para el propósito inicial de este nuevo enfoque.

**Documentos**

Se deberán formalizar las pruebas que se realicen y sus resultados, así como establecer métricas para evaluar aspectos no funcionales que deben estar claramente definidas a nivel organizacional.

Es fundamental para el equipo de esta área contar con los documentos formales de requerimientos del proyecto de manera de conocer en forma clara los aspectos a evaluar para cumplir con las solicitudes del cliente.

**Actividades recomendadas**

- Métricas de pruebas no funcionales
- Diseño de testing no funcional
- Estrategia organizacional de testing no funcional
- Implementación de herramientas
- Reuniones de colaboración interdisciplinaria

### 4.2.7 Mejora Continua

Finalmente, tras gestionar las áreas anteriores es momento de que las mejoras alcanzadas en el testing sean controladas periódicamente y los esfuerzos apunten a mantener los cambios y buscar nuevas alternativas para obtener nuevas mejoras. La organización no debe conformarse con los logros alcanzados, ni debe dar por superada sus problemáticas de testing.

Se deben programar reuniones periódicas con todo el equipo de desarrollo para evaluar la situación y analizar los nuevos desafíos para la organización. Además, se puede realizar una evaluación de satisfacción de los clientes con el fin de determinar el verdadero valor que se ha agregado a los productos y cuanto ha aumentado la calidad en el desarrollo de tal manera de sacar provecho a los logros.

Las pruebas deben estar altamente gestionadas y documentadas a nivel total de acuerdo a la Política Organizacional, y los proyectos no deben ser dados de alta sin cumplir con los requisitos acordados con los clientes. El trabajo de mantenimiento debiese apuntar a la prevención y no en corregir fallos descubiertos por el uso del sistema en su ambiente de explotación.

El cambio cultural en las personas se consolida y las mejoras en el testing son vistas como una necesidad ya que permite un mejor desarrollo y una disminución en reparación de sistemas luego de ser entregados.

Las decisiones se basan en hechos, es decir, la experiencia de la empresa ya debe ser tal que le permita definirse en sus proyectos en base a los trabajos previos y criterios objetivos que le permitan predecir las situaciones futuras.

Si las áreas anteriores han sido correctamente aplicadas, ésta última será alcanzada y gestionada como una consecuencia natural de todo el proceso de mejora propuesto en este modelo, ya que el concepto de testing estará plenamente adquirido en la organización y ya será imposible concebir el desarrollo sin todos los conceptos de testing implementados.

**Personal**

La organización debe fomentar la comunicación entre todo el equipo involucrado en la mejora de testing y en caso de que exista desvinculación laboral de personal se deberá buscar un reemplazo que esté dispuesto a hacerse parte de los procesos de mejoras establecidos.

En la medida que este modelo esté plenamente implementado puede evaluarse la posibilidad de realizar rotación entre los distintos roles, siempre que las habilidades técnicas del personal lo permita, ya que permitirá al personal tener una visión más amplia de las mejoras.

**Documentación**

La formalización de todas las mejoras, causas y efectos de problemas estarán documentados. Esto reflejará la condición de tomar las decisiones basándose en hechos propuestos para esta área.

Las métricas establecidas para la evaluación de los distintos criterios de calidad están disponibles en un repositorio para todo el personal y el desarrollo de los proyectos es documentado en forma progresiva.

**Actividades recomendadas**

- Rotación de roles
- Reuniones de evaluación
- Mantenimiento preventivo
- Capacitación constante
- Encuestas de satisfacción
- Determinación de valor agregado

## 4.3 Modelo de Diagnóstico

El modelo de diagnóstico es la herramienta que permite conocer la situación inicial de una organización que desea aplicar las mejoras presentadas. Para la evaluación, por lo tanto, se toma como referencia el modelo de mejoras propuesto para así obtener una impresión (comparación) que permita conocer el real estado del proceso de testing. Por lo tanto, un adecuado diagnóstico permite conocer las deficiencias y fortalezas que presenta un determinado caso de aplicación para así poder enfocar de forma correcta los esfuerzos que conducirán a mejoras en el proceso de testing.

Para la correcta aplicación de las mejoras, el modelo de diagnóstico propuesto consta de tres fases: Evaluación de la organización, propuesta de mejoras y el acuerdo de mejoras.

### 4.3.1 Evaluación de la Organización

Esta fase es la encargada de obtener los datos necesarios sobre el estado de la organización. Esto, se hace evaluando por separado el estado organizacional de cada una de las secciones de aplicación, además de un estudio global del entorno organizacional.

Se requiere de un evaluador único que dirija el proceso de evaluación y recopile la información pertinente del caso. El proceso se basa principalmente en la resolución de un breve cuestionario que permite, en primer lugar, obtener una respuesta certera (sí o no) para luego extraer datos adicionales en la sección de comentarios.

El evaluador también debe solicitar a los encargados una descripción detallada del proceso de testing que se aplica inicialmente, de modo de poder comprender el funcionamiento global del proceso. Para esto, se necesitan reuniones y encuentros con los distintos actores del proceso de desarrollo para captar sus impresiones referentes al tema, conocer el material documentado existente (si lo hay) y otros aspectos relevantes. También se deben tomar en cuenta las propuestas de la organización para posteriormente ayudar en la elaboración de las "Propuestas de Mejoras".

Las tablas 4.1 a 4.7 muestran el cuestionario a aplicar para la evaluación de cada una de las secciones de aplicación.

Tabla 4.1 Cuestionario sección "Pólitica Organizacional".

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Existen políticas y objetivos para el testing? | | |
| **2** | ¿Existe política de planificación de testing? | | |
| **3** | ¿Existen estándares definidos para testing? | | |
| **4** | ¿La política de asignación de personal al testing es clara? | | |
| **5** | ¿La política de asignación de otro tipo de recursos es conocida? | | |
| **6** | ¿Existe un control para la efectividad del testing? | | |

Tabla 4.2 Cuestionario sección "Testing de Bajo Nivel".

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Existen capacitaciones para testing? | | |
| **2** | ¿Existen instancias comparativas entre pares? | | |
| **3** | ¿Se cuenta con herramientas (tecnológicas) para el testing? | | |
| **4** | ¿Las pruebas se documentan? | | |
| **5** | ¿Las pruebas se planean? | | |

Tabla 4.3 Cuestionario sección diseño.

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Existe un diseño de pruebas? | | |
| **2** | ¿Se conocen pruebas para determinados requerimientos? | | |
| **3** | ¿Los procedimientos de ejecución de pruebas son conocidos? | | |
| **4** | ¿Existe un control para el diseño? | | |

Tabla 4.4 Cuestionario sección "Integración al Proyecto".

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Existen instancias de encuentros entre testers y analistas/diseñadores? | | |
| **2** | ¿Existen instancias de encuentro entre testers y programadores? | | |
| **3** | ¿Las pruebas se basan en requerimientos? | | |
| **4** | ¿Existe un documento cuantitativo de aprobación de pruebas? | | |
| **5** | ¿Existen instancias comparativas de pares? | | |

Tabla 4.5 Cuestionario sección "Prevención".

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Los equipos (análisis, diseño, programación, testers) rotan sus roles? | | |
| **2** | ¿Existen capacitaciones orientadas a la prevención? | | |
| **3** | ¿Existe revisión entre pares? | | |
| **4** | ¿Existe revisión entre los distintos roles? | | |
| **5** | ¿Las pruebas detectan grandes fallas del sistema? | | |

Tabla 4.6 Cuestionario sección "Testing No Funcional".

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Existen pruebas no funcionales? | | |
| **2** | ¿Existen métricas cuantitativas para el testing no funcional? | | |
| **3** | ¿Existe mantenimiento preventivo para los proyectos? | | |
| **4** | ¿La organización cuenta con capacitación? | | |
| **5** | ¿Existen herramientas de evaluación de satisfacción de los clientes y usuarios finales? | | |
| **6** | ¿Se busca entregar un valor agregado a los productos? | | |

Tabla 4.7 Cuestionario sección "Mejora Continua".

| N° | Pregunta | Respuesta (S/N) | Comentarios |
|---|---|---|---|
| **1** | ¿Existe rotación de roles/tareas entre los distintos actores? | | |
| **2** | ¿Existen instancias de evaluación para el proceso de testing? | | |
| **3** | ¿Las pruebas se basan en requerimientos? | | |
| **4** | ¿Existe un documento cuantitativo de aprobación de pruebas? | | |
| **5** | ¿Existen instancias comparativas de pares? | | |

### 4.3.2 Propuesta de Mejora

Una vez terminada la etapa de evaluación, los datos deben ser trabajados y comparados con el modelo de mejoras. Esto, permite determinar cuáles aspectos del modelo están siendo trabajados, en qué grado, y cuáles no. Lo anterior, se traduce en un informe de propuestas destinado a la organización, el que contiene aspectos a mejorar, determinando plazos de trabajo, recursos y otros aspectos relevantes.

La propuesta debe abarcar la mayor cantidad de mejoras posibles, pues será en la siguiente etapa en donde se determine en conjunto con la organización cuál será el trabajo que realmente se ejercerá y con cuáles recursos se podrá contar para el trabajo de mejoras.

Los resultados de cada cuestionario de las distintas secciones de aplicación deben ser entregados según el formato propuesto por la tabla 4.8.

Tabla 4.8 Resultado de cuestionarios

| Sección de aplicación | Respuestas positivas (%) | Respuestas negativas (%) |
|---|---|---|
| **Política Organizacional** | | |
| **Testing de Bajo Nivel** | | |
| **Diseño** | | |
| **Integración al Proyecto** | | |
| **Prevención** | | |
| **Testing No Funcional** | | |
| **Mejora Continua** | | |

Para formalizar los contenidos mínimos del informe se establecen los siguientes:

- Recursos estimados para la ejecución del plan de mejoras.
- Fortalezas y debilidades por cada sección de aplicación.
- Fortalezas y debilidades a nivel organizacional.
- Plan de aplicación de mejoras por sección (puede trabajarse simultáneamente más de una sección).

### 4.3.3 Acuerdo de Mejoras

Una vez que el informe con la "Propuesta de Mejora" es revisado por la organización, se debe decidir en qué aspectos del testing se está dispuesto a invertir los recursos. Al tratarse de un modelo desarrollado para pequeñas empresas, la flexibilidad de éste es fundamental para su implementación en forma correcta. Por lo tanto, en gran parte depende del criterio de los encargados organizacionales

De esta etapa debe obtenerse un documento formal de los acuerdos de trabajo establecidos entre el evaluador y el personal encargado de la organización.

# 5 Caso de Aplicación

## 5.1 Contexto de Aplicación

La organización en que se aplicó el modelo es una empresa formada hace cuatro años originada para entregar servicios a empresas en las áreas de redes informáticas, diseño y tecnologías de la información y en el último tiempo en el desarrollo de software a medida. En el área de desarrollo el equipo de trabajo es constituido por sus dos socios quienes ejercen el papel de analistas y diseñadores, y por otra parte, se cuenta con un equipo en promedio de tres programadores que son quienes ejecutan el testing.

Entre sus principales clientes se encuentra el rubro hotelero (Queen Royal, Mediterráneo, entre otros), firmas de abogados y el centro de formación técnica de la Universidad de Valparaíso.

A futuro, se planea orientar sus negocios hacia el desarrollo de software casi en su totalidad. Por lo mismo, se encuentran trabajando en algunos proyectos de mayor importancia a la experiencia anterior, por lo que requieren mejorar sus procesos de desarrollo y testing para ofrecer productos de mayor calidad y garantías.

## 5.2 Diagnóstico

### 5.2.1 Modelo de Evaluación

Para registrar la evaluación realizada a la organización en donde se implementará el modelo, se presenta las respuestas obtenidas de las encuestas presentadas anteriormente, esto se hace presentado las preguntas de cada sección de aplicación en una tabla por separado y la cantidad de respuestas afirmativas y negativas. Además, en base a los comentarios de dichas respuestas y al análisis hecho en terreno por el evaluador se procederá a evaluar la organización con respecto al modelo de mejoras propuesto. Cabe destacar que los cuestionarios se aplicaron a cuatro personas de la organización. Los resultados de los cuestionarios se presenten en las tablas de 5.1 a 5.7.

Tabla 5.1 Resultado cuestionario sección "Política Organizacional".

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Existen políticas y objetivos para el testing? | 0 | 4 |
| **2** | ¿Existe política de planificación de pruebas? | 0 | 4 |
| **3** | ¿Existen estándares definidos para testing? | 0 | 4 |
| **4** | ¿La política de asignación de personal al testing es clara? | 1 | 3 |
| **5** | ¿La política de asignación de otro tipo de recursos es conocida? | 0 | 4 |
| **6** | ¿Existe un control para la efectividad de las pruebas? | 1 | 3 |

Tabla 5.2 Resultado cuestionario sección "Testing de Bajo Nivel"

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Existen capacitaciones para testing? | 0 | 4 |
| **2** | ¿Existen instancias comparativas entre pares? | 0 | 4 |
| **3** | ¿Se cuenta con herramientas (tecnológicas) para el testing? | 3 | 1 |
| **4** | ¿Las pruebas se documentan? | 2 | 2 |
| **5** | ¿Las pruebas se planean? | 4 | 0 |

Tabla 5.3 Resultado cuestionario sección "Diseño".

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Existe un diseño de pruebas? | 0 | 4 |
| **2** | ¿Se conocen pruebas para determinados requerimientos? | 1 | 3 |
| **3** | ¿Los procedimientos de ejecución de pruebas son conocidos? | 0 | 4 |
| **4** | ¿Existe un control para el diseño? | 0 | 4 |

Tabla 5.4 Resultado cuestionario sección "Integración al Proyecto".

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Existen instancias de encuentros entre testers y analistas/diseñadores? | 0 | 4 |
| **2** | ¿Existen instancias de encuentro entre testers y programadores? | 1 | 3 |
| **3** | ¿Las pruebas se basan en requerimientos? | 3 | 1 |
| **4** | ¿Existe un documento cuantitativo de aprobación de pruebas? | 0 | 4 |
| **5** | ¿Existen instancias comparativas de pares? | 0 | 4 |

Tabla 5.5 Resultado cuestionario sección "Prevención".

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Los equipos (análisis, diseño, programación, testers) rotan sus roles? | 0 | 4 |
| **2** | ¿Existen capacitaciones orientadas a la prevención? | 0 | 4 |
| **3** | ¿Existe revisión entre pares? | 0 | 4 |
| **4** | ¿Existe revisión entre los distintos roles? | 1 | 3 |
| **5** | ¿Las pruebas detectan grandes fallas del sistema? | 0 | 4 |

Tabla 5.6 Resultado cuestionario sección "Testing No Funcional".

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Existen pruebas no funcionales? | 1 | 3 |
| **2** | ¿Existen métricas cuantitativas para el testing no funcional? | 0 | 4 |
| **3** | ¿Existe mantenimiento preventivo para los proyectos? | 0 | 4 |
| **4** | ¿La organización cuenta con capacitación? | 0 | 4 |
| **5** | ¿Existen herramientas de evaluación de satisfacción de los clientes y usuarios finales? | 0 | 4 |
| **6** | ¿Se busca entregar un valor agregado a los productos? | 4 | 0 |

Tabla 5.7 Resultado cuestionario sección "Mejora Contínua".

| N° | Pregunta | Respuesta Afirmativas | Respuestas Negativas |
|---|---|---|---|
| **1** | ¿Existe rotación de roles/tareas entre los distintos actores? | 0 | 4 |
| **2** | ¿Existen instancias de evaluación para el proceso de testing? | 0 | 4 |
| **3** | ¿Las pruebas se basan en requerimientos? | 0 | 4 |
| **4** | ¿Existe un documento cuantitativo de aprobación de pruebas? | 0 | 4 |
| **5** | ¿Existen instancias comparativas de pares? | 0 | 4 |

En la tabla 5.8 se exponen algunos comentarios relevantes al respecto de ciertas respuestas.

Tabla 5.8 Comentarios de respuestas.

| N° Tabla | N° Pregunta | Comentarios |
|---|---|---|
| **5.1** | **2** | NO. Pero existen períodos establecidos de marcha blanca planificados por contrato. |
| **5.2** | **1** | NO. Pero existe un interés en informarse en la medida que es posible. |
| **5.2** | **2** | NO. Cada uno es responsable de sus tareas. |
| **5.2** | **4** | NO. Existe un registro pobre a través de e-mail. |
| **5.2** | **5** | SÍ. Hay un programador que planifica las pruebas.<br><br>SÍ. Al conocer los requerimientos se piensa en cómo probar. |
| **5.3** | **2** | SÍ. En desarrollo web al conocer los requerimientos ya se sabe qué probar, ya que muchos errores son de compatibilidad de browsers. |

| N° Tabla | N° Pregunta | Comentarios |
|---|---|---|
| **5.4** | **3** | SÍ. Siempre |
| **5.5** | **5** | NO. Errores pequeños generalmente.<br><br>NO. Errores de requerimientos principalmente. |
| **5.6** | **1** | SÍ. Usabilidad sobretodo. Fácil y rápido para todos. |
| **5.6** | **6** | SÍ. Usabilidad.<br><br>SÍ. Estabilidad y cosas que convenzan al cliente de que somos la mejor opción. |

Durante las visitas a la empresa se pudo conocer el estilo de trabajo que se aplica allí. Existen una serie de deficiencias que impiden poder llevar a cabo un proceso de testing dedicado y de calidad.

El estilo de desarrollo es el siguiente: Los analistas se reúnen con los clientes para obtener los requerimientos del producto, luego estos mismos analistas se encargan de diseñar el sistema para luego, una vez que terminan, entregar las especificaciones a los programadores. Los programadores se encargan de probar individualmente sus módulos, para luego hacer en conjunto a los analistas-diseñadores pruebas de sistema (sin planificación y sin registros relevantes de sus resultados) para pasar a un período de marcha blanca con el cliente que por lo general termina siendo la entrega de un prototipo, pues en este período se detectan cambios o problemas de requerimientos y el sistema debe ser modificado y nuevamente probado en marcha blanca por el usuario antes de realizar la entrega final.

Uno de los programadores, planifica sus pruebas, sin embargo no existe diseño de éstas lo que implica principalmente dos cosas: Las pruebas no son lo suficientemente efectivas y las pruebas se vuelven a preparar para futuros proyectos (no existe reutilización de testing). Los otros programadores, no preparan de forma alguna las pruebas a realizar, y aplican una suerte de improvisación lo que trae como consecuencia que muchas veces son los usuarios finales los que prueban realmente el sistema y descubren sus vulnerabilidades y deben reportarlas.

Las pruebas se realizan en su mayoría sin herramientas, a excepción de una aplicación que utiliza uno de los programadores para cargar datos y poner a prueba la estabilidad y robustez de sus aplicaciones (pruebas de estrés). El resto realiza todo manualmente, uno por uno lo que impide probar de forma certera las capacidades de las aplicaciones.

A nivel general, queda claro que el proceso de testing es pobre con respecto al modelo de mejoras establecido y es una actividad no valorada, sin embargo y contradictoriamente se manifiesta un interés en querer mejorar los procesos de desarrollo en general, y en especifico el testing. La organización se ha mostrado dispuesta a colaborar con la evaluación y dicen

estar dispuestos a cumplir con el trabajo necesario, en la medida que sea posible, para mejorar en la calidad de sus productos.

### 5.2.2 Propuesta de Mejoras

La tabla 5.9 en forma resumida pretende entregar el grado de aplicación de cada una de las secciones de aplicación.

Tabla 5.9 Aplicación de secciones.

| Sección de aplicación | Respuestas positivas (%) | Respuestas negativas (%) |
|---|---|---|
| **Política Organizacional** | 8,33% | 91,67% |
| **Testing de Bajo Nivel** | 45% | 55% |
| **Diseño** | 6,25% | 93,75% |
| **Integración al Proyecto** | 20% | 80% |
| **Prevención** | 5% | 95% |
| **Testing No Funcional** | 20,83% | 79,17% |
| **Mejora Continua** | 0% | 100% |

#### 5.2.2.1 **Recursos estimados para la ejecución del plan de mejoras**

- Recursos económicos: Para mejorar el proceso de testing se deberán contemplar recursos monetarios para invertir en la adquisición de herramientas de automatización de testing y la contratación de nuevo personal para dedicación de las pruebas.

- Recursos de personal: Se requiere de personal que pueda dedicar su tiempo a la elaboración y diseño de pruebas.
- Recursos de tiempo: Los proyectos deberán retrasarse en su entrega de manera de poder cumplir con un proceso de testing adecuado.

#### 5.2.2.2 **Fortalezas y debilidades por cada sección de aplicación**

- Política Organizacional: En base al cuestionario se aprecia un pobre política con respecto al proceso de testing, sin embargo ésta no es nula lo que evidencia que existe una intención de mantener un proceso estable y que las pruebas son una etapa necesaria dentro del ciclo de desarrollo. En específico, al estar establecidos por contratos los períodos de marcha blanca indican que es ineludible un período para testear el software.

- Testing de bajo nivel: El testing de bajo nivel es la sección que mejor está mejor gestionada dentro del proceso de testing, pues los programadores tienen la posibilidad de probar libremente sus desarrollos, aunque la debilidad de esta libertad es que no existen exigencias mínimas formalmente. Existe la vocación por parte de un programador de planificar sus pruebas desde el inicio del desarrollo, lo que cual es un activo que se debe aprovechar y replicar en el resto del equipo de trabajo. Las debilidad más importante es que no existen instancias de comparación entre los distintos miembros del equipo llegando incluso a definirse que las responsabilidades son individuales.

- Diseño: El diseño deficiente o prácticamente nulo se condice con la falta de políticas organizacionales. Sin embargo, la existencia de pruebas predeterminadas según ciertos requerimientos entrega nociones de un proceso de diseño de pruebas vago, pero existente. Las principales debilidades de esta sección son la falta de trabajo en sí misma.

- Integración al proyecto: Un factor crítico que no permite una integración del testing en todo el ciclo de desarrollo tiene que ver con el flujo de trabajo entre analistas-diseñadores y desarrolladores, que al no tener grandes instancias de encuentro el testing queda aislado del resto del ciclo. Por otro lado, existe una convicción de que las pruebas deben estar basadas en los requerimientos presentados por los clientes.

- Prevención: Esta sección se encuentra altamente descuidada, pues no existe una ideología clara del testing o bien ésta no está correctamente enfocada a producir productos de calidad. No obstante, la historia demuestra que no existen grandes errores de programación los que se han producido, sino más bien de toma de requerimientos.

- Testing no funcional: El testing no funcional es un tema que si bien no está gestionado formalmente, es un tema importante para la organización y se demuestra en la búsqueda de valor agregado que se le quiere entregar a los productos y en las pruebas no funcionales aplicadas (pruebas no planificadas ni diseñadas). Las principales debilidades son por lo tanto la falta de métricas, capacitaciones y todo lo que abarca esta sección para su correcta gestión.
- Mejora continua: Esta sección es la única que no obtuvo ninguna respuesta afirmativa, evidenciando la falta de madurez en el proceso de testing. Por lo tanto, el

mantener las mejores que se logren tras la aplicación de las mejoras, será un tema relevante de tratar para no perder los esfuerzos que se realicen.

#### 5.2.2.3 **Fortalezas y debilidades a nivel organizacional**

En base a las visitas, cuestionarios y entrevistas aplicadas a la organización se concluye que la gran fortaleza es la disposición a mejorar el proceso de testing. Además, a pesar de la falta de un proceso formal para el testing, sí existe al menos un conocimiento y un intento por aplicar buenas prácticas para mejorar dicho proceso.

Una oportunidad que no se debe dejar pasar es que el giro de la empresa está cambiando hacia el desarrollo de software como fuente primaria de proyectos, lo que se traduce en la ganancia de experiencia y la posibilidad de ir aplicando mejoras en la medida que se abarcan nuevos proyectos.

Las debilidades por otro lado, se ven por el lado de la falta de personal que pueda dedicarse a llevar un testing gestionado, así como también la falta de tiempo para dedicarse a las mejoras. Lo anterior sumado a las pocas posibilidades de inversión en cuanto a herramientas de apoyo, capacitaciones hacen de esta organización un caso ideal para aplicar el modelo de mejoras, pues cualquiera de los grandes modelos existentes sería prácticamente imposible de implementar.

Como todo proceso de mejoras, éste se puede ver amenazado por que los efectos no se vean evidenciados en el corto plazo y todas las prácticas que se apliquen se vean como una burocratización de procesos ya establecidos. Sin embargo, si se realiza un plan de trabajo acordado y con metas claras, éstas se deberán reflejadas con avances en plazos determinados.

#### 5.2.2.4 **Plan de aplicación de mejoras para las secciones**

- Política Organizacional: Esta sección será la primera que debe trabajar, ya que aquí se definirán las limitaciones para el trabajo futuro del resto de las secciones. Se deberá definir los objetivos para el proceso de testing y políticas que permitan definir de forma correcta la planificación de pruebas, así como también estándares. Por lo anterior, deberá acordarse los recursos de los que se dispondrá para las mejoras así como el personal que participará en el cambio. Esto debe quedar acordado en la primera semana de trabajo.

- Testing de bajo nivel: El testing de bajo nivel al ya estar implementado, sólo se deberá mejorar y ejercer ciertas tareas que permitan mejorarlo. Se deberán propiciar instancias de revisiones entre distintos desarrolladores y la adquisición de herramientas que permitan automatizar las pruebas. Por otro lado, todas las pruebas deberán ser registradas y documentadas para ser reutilizadas en el futuro.
- Diseño: Se empleará un esquema para diseñar pruebas en base a los requerimientos expresados por los usuarios, así como también se debe especificar los procesos de ejecución para su correcta implementación. Esta sección requerirá de mayor tiempo, pues deberán implementarse en distintos proyectos, por lo menos cuatro semanas.

- Integración al proyecto: Para integrar el testing a todo el ciclo de desarrollo de los productos se deberá establecer un trabajo conjunto entre los distintos actores del ciclo. Así, se fomentaran instancias de evaluación entre pares que permitan mejorar las tareas que implica el testing. Además, se debe establecer un documento con métricas para la aprobación de los requerimientos de los usuarios. Esta sección se recomienda ser gestionada sólo después que el diseño esté mejorado.

- Prevención: Se fomentará el cambio de pensamiento desde el comienzo del proceso de mejoras. La idea es que en la medida que se entiendan las mejoras del testing, la organización sea capaz de valorar la importancia de la prevención por sobre la búsqueda de errores y fallas de los productos. Por lo mismo, se deberán mejorar las instancias de pruebas para desarrollar modelos de pruebas que permitan encontrar errores críticos del sistema. Esto tomará tanto como dure el proceso de mejoras en general, es decir, una estimación de más de un mes.

- Testing no funcional: Esta sección sólo requiere llevar a la práctica las intenciones existentes en la organización. Se establecerá dentro de las políticas una formalización sobre los atributos no funcionales que se desee entregar a los clientes, al mismo tiempo de establecer métricas que los comprueben. Para eso, será necesario la adquisición de herramientas y establecer planes de mantenimientos preventivos que permitan mejorar las capacidades de los sistemas y evitar problemas futuros tras su entrega. Esta sección debe ser trabajada en conjunto al diseño.

- Mejora continua: Una vez que se mejoren todas las demás secciones, se establecerá un plan de mejoras continuas para poder mantener e incluso seguir avanzando en las mejoras alcanzadas. Se deberá establecer un proceso período de evaluación para el proceso de testing y estas prácticas deben establecerse dentro de las semanas siguientes a la aplicación de las mejoras en el resto de las secciones.

### 5.2.3 Acuerdo de Mejoras

A modo de resumen, en la tabla 5.10 se enumeran las mejoras propuestas y se confirman las aceptadas.

Tabla 5.10 Mejoras propuestas y aceptadas.

| Sección de aplicación | Mejoras |
|---|---|
| **Política Organizacional** | Definir los objetivos para el proceso de testing. ✔<br><br>Definir políticas de planificación de pruebas. ✔<br><br>Definir estándares de testing. ✔<br><br>Establecer recursos económicos para las mejoras.<br><br>Establecer recursos de personal para las mejoras.<br><br>Establecer recursos de tiempo para las mejoras. ✔<br><br>Establecer personal que participará en el cambio. ✔ |
| **Testing de Bajo Nivel** | Generar instancias de revisiones entre distintos desarrolladores.<br><br>Adquisición de herramientas para automatizar pruebas. ✔ *(sólo herramientas libres de costos)*<br><br>Registrar y documentar pruebas. ✔ |
| **Diseño** | Establecer un esquema para diseñar pruebas en base a los requerimientos. ✔<br><br>Especificar los procesos de ejecución de pruebas. |
| **Integración al Proyecto** | Integrar el testing a todo el ciclo de desarrollo.<br><br>Establecer un trabajo conjunto entre los distintos actores del ciclo.<br><br>Fomentaraá instancias de evaluación entre pares.<br><br>Establecer un documento de métricas para la aprobación de los requerimientos. ✔ |
| **Prevención** | Generar planificación de pruebas en fases de diseño. ✔<br><br>Desarrollar modelos de pruebas que permitan encontrar errores críticos del sistema. ✔ |

| Sección de aplicación | Mejoras |
|---|---|
| **Testing No Funcional** | Establecer políticas sobre los atributos no funcionales a entregar a los clientes. ✔<br><br>Establecer métricas para el testing no funcional<br><br>Adquisición de herramientas.<br><br>Establecer planes de mantenimientos preventivos. ✔ |
| **Mejora Continua** | Establecer un plan de mejoras continuas. ✔<br><br>Establecer un proceso periódico de evaluación para el proceso de testing. ✔ |

Luego de conversar con los dueños de la empresa para establecer el plan de trabajo se ha llegado a un acuerdo de trabajo que, como se muestra en las tablas, no han aceptado algunas sugerencias, entre ellas destacan que:

- No se destinarán recursos económicos para la adquisición de herramientas ni de personal

- El testing de bajo nivel será inmediatamente mejorado, sin embargo sólo en cuanto a diseño y planificación, pues no se considera gran interacción entre los distintos programadores para la mejora del testing. Esto, se debe principalmente a la forma de trabajo que hace complicado generar estas instancias de forma permanente.

- El testing no se incluirá de forma formal al proceso entero de desarrollo, debido a la complicación de involucrar a todo el personal de forma permanente en el proceso, sin embargo existe el compromiso de que el testing esté relacionado plenamente a los requerimientos del usuario.

- El testing no funcional se formalizará, sin embargo no se invertirá en la mejora de pruebas ni en su forma de aprobación. No obstante, dentro de un proceso de mantenimiento preventivo se incluirá la revisión de estos aspectos por parte de los usuarios finales.

## 5.3 Trabajo Práctico

A continuación, se detalla el trabajo realizado como parte de la aplicación del modelo de mejoras. De esta forma, se entrega un testimonio de cómo se realizó dicha experiencia y las mejoras que se implementaron en el proceso de testing de la organización.

### 5.3.1 Política Organizacional

#### 5.3.1.1 **Definir los objetivos para el proceso de testing**

Los objetivos específicos que se definieron para cumplir el proceso de testing son los siguientes:

- Verificar y validar los sistemas desarrollados por la organización.
- Corregir los errores detectados durante los períodos de desarrollo de productos.
- Prevenir defectos desde etapas tempranas del desarrollo de productos.

#### 5.3.1.2 **Definir políticas de planificación de pruebas**

Las políticas definidas en la planificación si bien no son específicas y tienden a ser superficiales se deben a que estas políticas se formalizarán en la medida que el nuevo proceso de testing se implemente y se obtenga una claridad de los alcances que éste tendrá. Por lo tanto, inicialmente las políticas básicas para la planificación de las pruebas son:

- Testing unitario implementado por propios programadores.
- Testing de integración dirigido por jefes de proyecto.
- Definición de condiciones especiales al inicio del proyecto.

#### 5.3.1.3 **Definir Estándares de Testing**

Lo estándares definidos en esta primera aplicación, solo consideran aspectos relativos al alcance de las pruebas o la dedicación que éstas tendrán:

- Las pruebas deben ser aprobadas en su 100%
- Por cada 5 horas de codificación corresponde un mínimo de 1 hora de trabajo dedicado al proceso de testing.

#### 5.3.1.4 **Establecer recursos de tiempo para las mejoras**

- Normalmente un trabajo de 2 horas semanales presencialmente en el lugar de trabajo por parte del encargado de implementar las mejoras.
- Se definen tareas a cumplir por parte de la organización y del encargado de mejoras. Los cuales deberán cumplir los plazos que se establezcan para el correcto funcionamiento de las mejoras.

#### 5.3.1.5 **Establecer personal que participará en el cambio**

- Analistas/diseñadores (2 trabajadores).
- Programadores (2 trabajadores).

### 5.3.2 Testing de Bajo Nivel

#### 5.3.2.1 **Adquisición de herramientas para automatizar pruebas (sólo herramientas libres de costos)**

**PHP CodeSniffer**

Si bien la idea era automatizar la ejecución de pruebas en sí, tras la recolección de información que permitiera encontrar las herramientas adecuadas para la mejora de las pruebas se llegó a la conclusión de que no es prioridad la automatización. Sin embargo, se implementó la utilización de una herramienta que permita controlar de mejor forma los estándares en la sintaxis de programación (estándares en desarrollo). Dado el mayor orden obtenido se consiguió mayor facilidad en la comprensión de códigos lo que se traduce en beneficios a la hora de corrección de módulos ante defectos detectados en las pruebas.

**PHPUnit**

Se experimenta con la automatización de las pruebas de unitarias a través del uso de este framework. Considerando que el proceso formal de pruebas dentro de la organización aún es inmaduro, el uso de esta herramienta hasta este punto se usó de forma experimental, mientras no se definan completamente que componentes del proceso de pruebas es el correcto para el desarrollo obtenido.

#### 5.3.2.2 **Registrar y documentar pruebas**

Para mantener un correcto historial de las pruebas de bajo nivel que se realizan se decidió que las pruebas deben contener ciertos aspectos que se detallan a continuación:

**Consideraciones sobre el plan de pruebas**

Antes de la explicitar el plan de pruebas de un proyecto, es necesario conocer como está conformado el producto a testear para poder establecer una línea base que permita la correcta planificación, alcance, ejecución y control de las pruebas.

Un resumen de las funcionalidades del sistema puede ayudar para la comprensión de esta documentación en el futuro, sin tener que recurrir a todo el material destinado al proyecto total. Es importante considerar los módulos que contempla y su interacción con otros. Estas consideraciones deben incluir aspectos relevantes tanto para las inspecciones de código como para las pruebas en sí.

**Formulario de inspección**

La tabla 5.11 es el documento que permitirá tener información sobre las inspecciones que se hayan realizado a algún trozo de código, funcionalidad o módulo. Se describe a continuación.

Tabla 5.11 Formulario de inspección.

| | |
|---|---|
| Responsable: | Nombre de quien inspecciona el código. |
| Sección: | Especificación del número de líneas del código fuente o funcionalidad o módulo a inspeccionar. |
| Fecha: | Fecha en que se realiza la inspección. |
| Resultados: | Conclusiones de la inspección y/o explicitación de los defectos detectados. Especificar incongruencias entre la codificación y documentos de diseño. |

**Ambiente del sistema**

El ambiente de pruebas debe ser, en lo posible, idéntico al ambiente en donde el sistema será explotado finalmente. De no ser así, se debe especificar las diferencias y dejar constancia las posibles consecuencias en la ejecución del sistema. Sin embargo, lo importante es registrar las condiciones del ambiente de prueba.

**Calendario de pruebas**

El ideal es que exista un calendario a seguir desde el inicio del proyecto. No obstante, cualquier cambio que se produzca debe ser documentado, con mayor motivo si el período de pruebas se extiende debido a fallas de implementación detectadas en el proceso de pruebas. El calendario debe cumplir con el mínimo establecido en las políticas organizacionales.

**Formulario de pruebas**

La tabla 5.12 es el documento que permitirá tener información sobre las pruebas que se hayan realizado en algún módulo. Entendiendo la gran cantidad de pruebas a realizar y apelando a la libertad propuesta para testear de cada programador, debe ser usado con el fin de registrar los casos de prueba más relevante.

Tabla 5.12 Formulario de descripción/resultados de la prueba.

| |
|---|
| Nº Prueba: Número de la prueba. |
| Encargado: Persona que realiza la prueba. |
| Fecha: Fecha cuando se realiza la prueba. |
| Sección: Función, módulo o conjunto de módulos a probar. |
| Descripción: Breve descripción de la prueba. |
| Resultados esperados: Resultados que el usuario espera del elemento a probar. |
| Entradas: Datos que fueron ingresados. |
| Salidas: Datos que se entregaron como resultado a las entradas. |
| Actividades: pasos que se realizaron durante la prueba. |
| Observaciones: explicitación de defectos, situaciones anómalas, suspensiones o simples comentarios de la prueba y/o elemento probado. |
| Nombre Firma |

### 5.3.3 Diseño

#### 5.3.3.1 **Establecer un esquema para diseñar pruebas en base a los requerimientos**

Para poder diseñar las pruebas en base a los requerimientos es necesario que los requerimientos de cada proyecto estén bien definidos, entendidos y convenidos tanto por los clientes como por la organización desarrolladora. Para esto, se ha debido generar una estructura que defina la forma en que se captan los requerimientos de los clientes para acordar el alcance de los proyectos.

La estructura considera los siguientes aspectos:

- Definición del Problema y Objetivos.
    - Descripción del Problema.
    - Objetivo General.
    - Objetivos Específicos.

- Especificación de Requerimientos.
    - Requerimientos Generales.
    - Requerimientos Específicos del Usuario.
    - Requerimientos No Funcionales.
    - Casos de Uso Gráfico.

Una vez formalizados los requerimientos y expectativas del usuario se establece un esquema para diseñar las pruebas que permitan probar lo que corresponda a cada proyecto. La tabla 5.13 explica los contenidos mínimos que debe cumplir los casos de pruebas según el requerimiento otorgado. Cabe destacar que este esquema se aplica a los requerimientos específicos de cada proyecto (funcionalidades propias del software), por lo que para las pruebas de bajo nivel que buscan la verificación de un determinado módulo genérico (como el validar una fecha, nombre, u otro dato) se aplica el criterio establecido inicialmente para las pruebas de bajo nivel.

Tabla 5.13 Esquema para diseño de pruebas.

| |
|---|
| Requerimiento: *Se específica el requerimiento especificado por el usuario* |
| Tipo de prueba: *Definir si será de tipo caja blanca o caja negra, en base a la funcionalidad/requisito a testear.* |
| Especificaciones de la prueba: *En qué consiste la prueba.* |
| Requerimientos para la prueba: *Necesidades de las prueba* |
| Proceso de obtención de datos: *Especificación de cómo se obtendrán los datos y qué tipo de datos son necesarios.* |

## 5.3.4 Integración al Proyecto

### 5.3.4.1 **Establecer un documento de métricas para la aprobación de los requerimientos**

Este documento tiene por objetivo involucrar en forma directa el proceso de planeación del testing en etapas tempranas del proyecto, logrando una integración del proceso de testing a lo largo del ciclo de vida. Específicamente, lo que pretende es que el analista, al momento de captar los requerimientos de los clientes, pueda definir un esbozo de las pruebas que serán necesarias para conseguir la aprobación de los requerimientos. Este documento debe definir en parámetros medibles ciertas condiciones que han de cumplir las pruebas para considerarlo satisfactorias.

Esta modalidad solo se utilizó en uno de los proyectos en que se implementaron mejoras de testing. Contextualizando se trataba de un sitio web, que debía exhibir a los clientes

propiedades disponibles para arriendo o venta, y debía poder ser administrado por la corredora de propiedades. Lo que se utilizó se resume en la tabla 5.14. Cabe destacar que no todos los requerimientos involucrados en el proyecto son considerados para este documento..

Tabla 5.14 Métricas para aprobación de requerimientos

| Requisito | Pruebas | Métrica |
|---|---|---|
| **Recomendar propiedades vía e-mail a conocidos** | Enviar a múltiples usuarios<br>Evitar filtro de spam | Lograr llegar a la bandeja de entrada a 5 dominios relevantes (Hotmail.com (y sus variaciones), gmail.com, yahoo.es, terra.cl, vtr.net) |
| **Solicitar cita para visita de propiedad** | Recepción de correo electrónico al corredor<br><br>Solicitar citas a una misma hora y día | Confirmar 1 cita, tras la solicitud de 2 (o más) concurrentes. |
| **Enviar propiedad para evaluación** | Subir múltiples imágenes digitales.<br><br>Indicar características de propiedad | Subida exitosa de 3 formatos: jpg, npg, bmp.<br><br>Comprobar en búsqueda con resultado 100% de coincidencia |
| **Buscador de propiedades por características** | Establecer características para propiedades<br><br>Obtener resultados de búsqueda correctos | Obtener resultados precisos en un 100%<br><br>Sugerir máximo 2 propiedades que no cumplan características. (rango de precios siguiente o sector habitacional cercano) |

## 5.3.5 Prevención

### 5.3.5.1 **Generar planificación de pruebas en fases de diseño**

Con el fin de prevenir futuras complicaciones en los productos que sean detectadas durante la fase de pruebas, se debe comenzar a planificar las pruebas mientras se esté diseñando el sistema. Esto, tendrá un impacto relevante en la medida que más proyectos documentados existan, pues en la medida que los diseños dispongan de mayor documentación pasada para planear las pruebas resultará más fácil el predecir ciertos errores típicos del pasado.

Dado que esta es la primera implementación de mejoras en el proceso de testing, no existe la documentación necesaria para realizar esta actividad, por lo que queda pendiente como tarea para la organización en la medida que sigan aplicando en nuevos proyectos el modelo de mejoras de testing aquí presentado. Sin embargo, durante esta primera aplicación se

realizó el ejercicio de discutir experiencias pasadas que permitieran predecir errores que pudieran parecer típicos en ciertos tipos de proyectos. Esto, trajo como resultado el evitar ciertos trabajos innecesarios y decisiones más rápidas en cuanto a tipos de tecnologías utilizadas en los desarrollos, lo que permitió ahorrar tiempo y esfuerzo por parte de los programadores.

#### 5.3.5.2 **Desarrollar modelos de pruebas que permitan encontrar errores críticos del sistema**

A diferencia de los tipos de pruebas aplicados normalmente, que solo buscan probar las funcionalidades de los distintos módulos, lo que se pretende ahora es desarrollar pruebas con la intención de "botar" el sistema desarrollado, de modo de encontrar errores críticos que no permitan la operación del producto testeado. Esta etapa del testing, se da como consecuencia de un proceso de testing correctamente gestionado en los otros aspectos.

Este aspecto del testing no fue posible implementarse durante la aplicación del modelo de mejoras, principalmente a causa de falta de tiempos debido a atrasos inesperados (aunque comunes) en los proyectos que participaron en las mejoras. Sin embargo, se declaró como forma de trabajo la siguiente metodología:

- El sistema debe ser testeado como un todo durante esta etapa.
- El sistema en lo posible según las capacidades organizacionales debe ser testeado por un programador ajeno al proyecto.
- Dentro de la ejecución de pruebas se pueden premeditar situaciones que ponga en riesgo la consistencia del sistema (por ejemplo: manipulación errónea en la base de datos, manipulación directa de los datos).
- Cualquier falla detectada debe ser reparada antes de dar de alta el proyecto.
- Los errores detectados en un proyecto que puedan estar replicados en proyectos pasados, implica la revisión obligatoria de dichos trabajos pasados (mantención preventiva).

### 5.3.6 Testing No Funcional

#### 5.3.6.1 **Establecer políticas sobre los atributos no funcionales a entregar a los clientes**

Las políticas pueden variar según la naturaleza de cada proyecto, sin embargo con el fin de establecer un sello característico en los productos elaborados por la organización se definen ciertas características no funcionales que deben considerarse en los desarrollos. Estas políticas, para cumplirse, implican que deben implementarse cambios en las etapas de desarrollo para así como en el diseño de pruebas para comprobar su buena aplicación.

**Especificaciones de entrega**

No se requieren pruebas específicas para comprobar el cumplimiento de este aspecto. Basta con verificar si se cumplen plazos, condiciones, formas y otros detalles que han de establecerse con el cliente.

**Usabilidad**

En vista de que gran parte de los desarrollos son con interfaz de un sitio web, lo que implican interacción con usuarios que no estarán familiarizados con el sistema, se hace muy necesario entregar sistemas con un gran nivel de usabilidad.

Para la verificación de este aspecto, mediante el uso de pruebas, se deben añadir evaluaciones heurísticas en los procesos que permitan detectar problemas en las interfaces. Para esto, el encargado de las mejoras ha debido instruir con casos prácticos sobre el cómo abordar las heurísticas implicadas en este proceso.

La tabla 5.15 es un resumen de problemas detectados en un proyecto de sitio web promocional de un hotel luego de realizar una evaluación heurística con cuatro evaluadores (personal de la organización que fue capacitado). En la columna del "principio no cumplido" se utiliza la letra "S" representa el promedio de severidad del problema (en una escala de 0 a 4) y la letra "F" el promedio de la frecuencia del problema (en una escala de 0 a 4).

Tabla 5.15 Resumen de evaluación heurística.

| N° | Explicación | Ejemplo | Principio no cumplido |
|---|---|---|---|
| 1 | La página principal sobrecargada de información. Lo que dificulta la ubicación de lo que se busca. | Si se busca el e-mail de contacto cuesta mucho encontrarlo. | Diseño estético y minimalista.<br>S:3,25<br>F: 2 |
| 5 | Inconsistencia en el uso de menús en distintos navegadores. | El menú de accesos directos en Mozilla Firefox lleva al link seleccionado al hacerle click. En Internet Explorer se elige el link y luego se clickea el botón Ir para ir a la página seleccionada. | Consistencia y estándares.<br>S: 2<br>F :3 |

| N° | Explicación | Ejemplo | Principio no cumplido |
|---|---|---|---|
| 7 | Distinta vista en navegadores. | En Mozilla Firefox se muestran 4 links en la parte inferior izquierda de la página principal. En Internet Explorer no se ven. | Consistencia y estándares.<br>S: 2,25<br>F :2 |
| 10 | Menú izquierdo muy extenso (11 opciones) | Muestra demasiadas opciones de una sola vez | Diseño estético y minimalista<br>S:2,25<br>F :2 |
| 11 | Menú izquierdo finaliza con opción inexistente | El menú izquierdo pareciera que no tiene fin, no está bien delimitado | Consistencia y Estándares<br>S: 1,5<br>F :1 |
| 17 | Imagen “Bienvenidos” del menú izquierdo no conduce a nada | La frase se confunde con un link | Compatibilidad entre el sistema y el mundo real.<br>Consistencia y Estándares<br>S: 3,75<br>F :1 |

| N° | Explicación | Ejemplo | Principio no cumplido |
|---|---|---|---|
| 19 | Links se abre en ventana nueva | Los menú abren en nuevas ventanas, lo que podría complicar la navegación por el sitio web | El usuario debe tener el control y libertad,<br>S: 2<br>F :3 |
| 23 | Espacio perdido, desaprovechado. En blanco. | A la derecha de la página, queda un espacio bastante grande que no se utiliza | Otro<br>S: 2,25<br>F :2 |
| 24 | No existe mapa del sitio | | Consistencia y Estándares<br>S: 3,25<br>F :1 |

**Eficiencia**

Para cumplir con estos requerimientos los sistemas deben ser sometidos a pruebas de "estrés". De esta forma, se pretende sobrecargar de datos con el fin de determinar si los tiempos de respuesta del sistema, mediante las bases de datos, responden con los requisitos planteados para el proyecto. De esta forma, se medirán la capacidad de respuestas ante consultas, ingresos y modificaciones en los datos.

Además de las pruebas mencionadas, dado el interés de asegurar la eficiencia, adicionalmente el desarrollo de los sistemas deberá incluir una especificación sobre el hardware necesario para mantener su correcto funcionamiento. Esto, implica nuevas pruebas que permitan establecer cuáles son las características que deben cumplir los equipos en que se instalarán los sistemas.

**Fiabilidad**

En la medida que el testing detecte errores que no sean de codificación, se deben establecer proporciones de fiabilidad de los sistemas, con el fin de determinar si se alcanzan los requerimientos del cliente en este aspecto. Por otro lado, se deben establecer los criterios de respaldo y resguardo de los sistemas, y probar las capacidades de recuperación de los sistemas antes una caída o pérdida de datos. Estas pruebas se ejecutan de forma manual y en base a casos específicos que puedan presentarse más casos poco probables.

**Portabilidad**

Si bien los sistemas son normalmente especificados para ciertos tipos de equipos para ser instalados y ejecutados, se realizarán pruebas en entornos similares (principalmente otros sistemas operativos a través de virtualización) para establecer capacidad de funcionamiento en entornos similares. Esto, con el fin de disminuír potenciales reclamos en caso de cambios no notificados en los equipos por parte de los usuarios.

#### 5.3.6.2 **Establecer planes de mantenimientos preventivos**

Para adelantarse a reportes de errores detectados por los usuarios una vez que ya ha sido entregado con el sistema se establecen planes de mantenimientos de tipo preventivos para los productos en funcionamiento.

Los plazos y ciclos para estos mantenimientos dependen de cada proyecto, sin embargo a modo general se establecen los aspectos mínimos que evaluarán estas tareas:

- Pruebas de estrés y consistencia. Orientado a las bases de datos y su integridad
- Testing a nivel de integración en base a conocimientos adquiridos por otros proyectos. Por ejemplo: errores detectados en proyectos similares de posterior desarrollo
- Actualizaciones de tecnologías (bases de datos principalmente) lo que conlleva a nuevas pruebas de compatibilidad
- Comprobación de copias de respaldo. Pruebas de restauración de sistema

### 5.3.7 Mejora Continua

#### 5.3.7.1 **Establecer un proceso periódico de evaluación para el proceso de testing**

Para verificar los alcances del proceso de testing y proponer modificaciones se establece que cada año, es decir, a partir de este momento, cada agosto de cada año evaluar el proceso de testing. Así, se debe decidir cuales aspectos no están funcionando como se espera que lo hagan y qué acciones se pueden tomar para corregirlos.

Este trabajo no se resume simplemente a una revisión anual, pues se entiende que por cada desarrollo que es sometido al proceso de testing se pueden obtener lecciones y propuestas que podrán ser evaluadas año a año para generalizarlas y hacerlas partes del proceso de testing.

La evaluación no contará con el encargado de mejoras de testing, por lo que requiere de un compromiso de parte de la organización de valorar las mejoras alcanzadas con el fin seguir mejorando los procesos en general y en específico las pruebas.

En concreto, se acuerda que la evaluación debe realizarse con actores de los distintos niveles del ciclo de desarrollo y que debe contar con un mínimo de dos semanas en donde se dediquen espacios idóneos para los debates y profundización de propuestas que hagan. Un

aspecto relevante de este ejercicio es analizar la inclusión dentro del testing de las actividades propuestas que no fueron aceptadas por la organización en el punto **5.2.3** con el fin de robustecer el proceso ya implementado.

## 5.4 Resultados

Antes de analizar los resultados obtenidos con la implementación del modelo de mejoras se debe hacer un resumen cronológico sobre cómo se llevó a cabo el trabajo. Lo primero, como resulta evidente fue el área de Política Organizacional, lo cual no llevó más allá de una semana en donde se tomaron las decisiones necesarias, posteriormente se mejoró el testing de bajo nivel aplicando las actividades planteadas en los proyectos que ya se encontraban en desarrollo, y posteriormente sumándose los nuevos proyectos trabajados durante la implementación de las mejoras. De forma simultánea se gestionaron dos áreas del modelo de mejoras: Testing No Funcional y Diseño, esto porque en cuanto a características no funcionales se podía avanzar sin la necesidad de estar en un proyecto en particular, por otro lado las normas de diseño tampoco requerían enfrentar un proyecto en particular por lo que se pudo terminar sin grandes inconvenientes estas dos áreas. Finalmente y en el siguiente orden se gestionaron las áreas restantes: Integración al Proyecto, Prevención y Mejora Continua.

Posterior a todas las mejoras implementadas se establecieron reuniones con la organización de modo de obtener la retroalimentación necesaria para comprender y dimensionar los alcances que tuvo la aplicación del modelo. De estos encuentros se establecen las siguientes conclusiones:

### 5.4.1 Ciclo definido de testing

Si se compara la situación inicial del proceso de testing existente con la situación lograda tras la aplicación del modelo de mejoras, se encontrarán diferencias determinantes en cuanto al nivel de procesos de desarrollos de los productos. La organización no poseía un proceso de testing definido dentro de su ciclo de desarrollo, ya que solo se dedicaban tiempos a pruebas, pero éstos no debían cumplir ningún estándar, políticas o expectativas planteadas.

Tras la aplicación del modelo de mejoras, se ha logrado consolidar un proceso de testing definido en tiempo, alcance y otros aspectos relevantes. La organización ya cuenta con un proceso formal y los productos desarrollados son sometidos a éste, lo que permite la fabricación de software con un mejor control de calidad pudiendo satisfacer el cumplimiento de requerimientos funcionales y no funcionales de parte de los clientes.

La organización se declara satisfecha con este primer paso, que si bien aún puede ser mejorado, ya representa una mejora a nivel global del funcionamiento de su negocio. Tanto así, que los socios fundadores plantean la disposición a formalizar todo el proceso de desarrollado utilizado en su empresa.

### 5.4.2 Disminución de repetición de trabajo

Al documentar las pruebas, tanto en su planeación, diseño y ejecución, se cuenta con información valiosa para futuros proyectos. Este aspecto, se pudo comprobar inmediatamente en los proyectos que fueron parte de la implementación de las mejoras, ya que permitió la reutilización de casos de prueba en piezas de software vinculadas; y la omisión de pruebas en piezas idénticas de código reutilizado.

Un ejemplo claro de esta situación se da con la creación de un sitio web que hubo que desarrollar posteriormente al proyecto con un hotel. Las pruebas heurísticas realizadas en el primer proyecto y resumidas en 5.3.6.1 permitieron un significativo ahorro de trabajo tras descubrir errores en el primer proyecto, que se tuvo gran cuidado de no volver a cometer en los proyectos siguientes.

Por otro lado, las pruebas realizadas a bajo nivel han permitido hoy un aumento significativo en la producción por parte de los programadores en base a dos factores: el primero como es evidente es la no repetición de los mismo errores de codificación y la segunda y más interesante se da al cometer errores de programación y no recordar cómo resolverlos (situación que siempre se ha dado en la organización), sin embargo hoy existe una documentación que permite chequear la solución y solucionar problemas que antes incluso tardaban días, sobre todo cuando es otro programador el que se ve enfrentado a la situación.

En este aspecto quedan en absoluta evidencia las mejoras entregadas a nivel organizacional la aplicación correcta de un proceso de testing. En conversaciones con programadores, manifiestan que este aspecto es altamente relevante para ellos y sienten que su trabajo es mucho más eficiente, puesto que normalmente no es parte de sus funciones lidiar con los conflictos que manifiestan los usuarios tras la entrega, pero sí es cotidiano lidiar con sus propios errores y frustraciones a la hora de codificar los sistemas.

### 5.4.3 Disminución de costos en detección de errores

Debido a la aplicación de pruebas más rigurosas, mejor planeadas y diseñadas. Se han podido descubrir errores de implementación con mayor frecuencia, si bien esto implica un costo de re-implementación que no ayuda a la agilidad de los desarrollos, es un costo muy inferior al tener que lidiar con un cliente insatisfecho o que descubre fallas una vez que le es entregado el producto.

La organización dentro de su historia, nunca ha tenido que enfrentarse a situaciones críticas que comprometan el funcionamiento de sus productos tras la entrega final a sus clientes, sin embargo se han dado ocasiones en que los usuarios son los que denuncian el hallazgo de pequeñas fallas y piden su corrección. El costo de tiempo, de personal y de oportunidad que implica el tener que “retomar” el desarrollo de un proyecto finalizado es tremendo para la organización, es por esto que el poder estar cada vez más seguros de que el cliente no manifestará problemas tras la entrega de su producto es tremendamente relevante para el buen funcionamiento de la empresa.

De los tres proyectos que fueron parte de la implementación del modelo de mejoras, tras casi dos meses de entrega del producto con más tiempo de funcionamiento no han existido

reclamos posteriores por parte de los usuarios, al menos en el aspecto de funcionalidades. Esto marca un hito, históricamente el mayor número de solicitudes por parte de los clientes se da durante el primer mes de uso de su producto.

Toda esta detección temprana y su ahorro en costos han permitido a la organización poder continuar avanzando en nuevos proyectos sin correr el peligro de retomar viejos proyectos que pudieran atentar con la correcta atención que merece un cliente. Este estado sin "clientes insatisfechos" debe ser chequeado periódicamente a través de las mantenciones preventivas con el fin seguir consolidando su buena relación con los clientes.

### 5.4.4 Baja en insatisfacción de clientes

Ligado al punto anterior la baja en la insatisfacción de clientes no solo se ha visto reflejada por el no descubrimientos de fallas por parte de ellos. Si el sistema cumple con sus requerimientos los usuarios estarán relativamente satisfechos, pero si además se hacen los esfuerzos por verificar que se cumplan los mínimos aspectos no funcionales, los usuarios tienen una actitud aún más positiva con el nuevo sistema.

Acá las pruebas de estrés fueron muy valoradas por la organización, ya que a pesar de siempre estar consideradas nunca se habían realizado debido a que no existía un procedimiento claro sobre cómo abordarlas. Su realización trajo como consecuencia bases de datos más estables y mejores tiempos, o al menos suficientes, de espera ante las distintas transacciones de los sistemas. Esto da como resultado usuarios más a gusto con los nuevos sistemas y una mayor satisfacción de parte del cliente.

Durante los últimos encuentros de evaluaciones con la organización han manifestado los logros alcanzados en cuanto a la mejoras de usabilidad en sus proyectos de desarrollo web. Ya que al ser sistemas que interactúan con usuarios sin preparación, éste aspecto se torna importante. Estas nuevas características integrados en sus desarrollos y comprobadas a través de pruebas especialmente diseñadas con ese fin, han permitido disminuir los plazos de entrega, en hasta un 10%, ya que los usuarios quedan satisfechos con los primeros prototipos o solo hacen pequeñas observaciones, lo cual en comparación a experiencias pasadas era totalmente distinto.

Otro factor que ha influida en la disminución de insatisfacción, a criterio de los dueños de la organización, es la formalización a través de documentos de la toma de requerimientos de los clientes. La estructura mencionada en 5.3.3.1 ha permitido mejorar la comunicación entre analistas y clientes, ya que existe un método más claro de hacer entender al cliente que se ha interpretado. Además, este documento evita de cierta forma al usuario al contradecir sus requerimientos y a los desarrolladores les permite hacer mejor su trabajo y a los testers planear de manera mejor y con más tiempo las pruebas a la que los sistemas serán sometidos.

### 5.4.5 Mejora Parcial en la valoración del testing

La mejora de testing dentro de la organización ha traído ventajas evidentes en muchos aspectos del desarrollo de software, sin embargo el convencimiento por parte de la

organización, principalmente por los programadores, aún no es total. Esto se debe principalmente a la burocratización que implica la adopción o mejora de un proceso de testing. En principio se tuvo que elaborar todos los formatos de documentación, aprender a seguir las reglas, documentar todo, re-hacer trabajo por no cumplir los acuerdos y otros aspectos que implican desmotivación por parte del personal.

Por otro lado, los resultados obtenidos hasta ahora, y en la medida que se han ejecutado nuevos proyectos, siguiendo las indicaciones del proceso de testing, han hecho valorar la inversión de trabajo realizado, ya que se entregan productos mejores y, gracias a la reutilización, en algunos casos con un menor esfuerzo.

Haciendo el balance entre ambas perspectivas se determina que el testing es valorado en cuanto a sus resultados casi totalmente, pero sus métodos no son del todo apreciados. Esta situación debería cambiar en función de que el proceso de testing sea totalmente integrado y deje de ser visto como nuevas exigencias organizacionales. Para esto, habrá que esperar a que se desarrollen más proyectos y a que se realicen las primeras evaluaciones del proceso para ver de qué manera se puede mejorar y tal vez desburocratizar las actividades.

Estas aprensiones en contra del testing son originadas principalmente por los programadores, no así por los analistas quienes son los dueños de la empresa y de alguna forma son capaces de ver a nivel global las consecuencias de estas implementaciones.

### 5.4.6 Retraso en la Elaboración de Productos

Este aspecto se aborda de manera independiente al punto anterior, pues indiscutiblemente los proyectos (sobre todo los primeros) se vieron retrasados por la implementación de una etapa formal de testing en el ciclo de desarrollo. Sin embargo, este aumento en el ciclo de producción solo es notorio en las primeras aplicaciones de testing, pues en la medida que existan más antecedentes previos a las pruebas los tiempos se van recuperando, como ya ha sucedido con los proyectos web.

Con el tiempo, los tiempos deberían recuperarse y la organización deberá decidir si estos recursos son destinados a desarrollo más rápido de sistemas, y por lo tanto abordar más proyectos o se destina a evaluar y seguir mejorando el proceso de testing con el fin de alcanzar productos de mayor calidad para sus clientes.

Si se contrastan los tiempos de desarrollo antes de la aplicación del modelo con los actuales, es evidente la diferencia, sin embargo se deben valorar los beneficios obtenidos tanto a nivel de procesos de desarrollo como a nivel de entrega final a los clientes. Por lo tanto, y en pleno consenso a los encargados de proyectos, se considera que el retraso (o inversión en tiempo) es necesaria y es rentable para los objetivos del negocio, ya que permite mejorar a largo plazo los proyectos y en el corto plazo la satisfacción de los clientes.

# 6 Conclusiones

La bibliografía estudiada ha demostrado que la situación que motiva la realización de este trabajo es verdadera y requiere de una solución hecha a la medida de las pequeñas empresas. Esto queda más de manifiesto con la implementación del modelo en la organización, pues se evidenció que la mejora de procesos involucrados en el desarrollo de testing son necesarias y los modelos actuales no son fáciles de aplicar en estos entornos pequeños.

La propuesta de mejoras presentada fue confeccionada en base al estudio que comprendido como marco teórico, por lo que se puede determinar que el trabajo realizado en la elaboración del marco teórico ha rendido un fruto. Probablemente, si hoy se volviera a desarrollar el marco teórico habría nuevos conceptos e ideas que se podrían agregar que permitan nutrir la solución planteada. Esto, representa un punto relevante para la implementación del modelo, el buscar una actualización constante, que como se ve en algunos modelos es una tendencia.

Con en el marco teórico desarrollado se puede determinar que los modelos, en general, tienen componentes y estructuras similares. Es común la utilización de escalas para calificar el estado de las organizaciones y sus mejoras. Además, existen relaciones explícitas entre modelos, por ejemplo cuando un modelo se elaboró basándose en otro. Por otro lado, existen modelos que en una primera impresión pueden resultar complejos debido a la gran cantidad de aspectos y tareas que implican, sin embargo las propuestas que entregan son sumamente interesantes y merecen ser analizadas para idear un modelo como el que se pretende generar.

Tras este informe final, se han cumplido los tres objetivos específicos, y por lo tanto el objetivo general. El último objetivo específico se logra implementando el modelo de mejoras en un entorno real, en este caso una pequeña empresa desarrolladora de software. Esto último, plantea nuevos desafíos para perfeccionar el modelo, pues surgen nuevas actividades e ideas que son dignas de ser consideradas. Por ejemplo, hay que volver a evaluar la planificación de aplicación del modelo, pues existen ocasiones en que los tiempos acostumbrados de la organización no son totalmente compatibles.

El modelo de diagnóstico resultó práctico para esta implementación, pues pudo reflejar bien la situación inicial y determinar el camino a seguir para alcanzar las mejoras que la organización deseaba. Sin embargo, se plantea la necesidad de conjugar de mejor manera la propuesta de mejoras que el encargado hace con las mejoras que la organización finalmente acepte.

Los alcances totales del modelo de mejoras no podrán ser medidos y dimensionados hasta que no se apliquen todas las actividades propuestas, para esto es necesario que se apliquen evaluaciones al proceso de testing establecido en la organización, pero esto escapa a los límites fijados para este trabajo.

Los desafíos más importantes se pueden dar en el ámbito de lograr que el modelo de mejoras pueda ser aplicado más ágilmente, con el fin de no causar la impresión de que se están burocratizando los procesos, aunque esto depende del proceso de testing a mejorar. En este

caso, el proceso de testing mejorado era bastante básico y poco formal, por lo que el modelo debió aplicarse en una organización poco estructurada en este sentido.

Otro desafío importante para seguir comprobando la eficacia del modelo sería poder aplicarlo en otras organizaciones de características similares en cuanto a personal y recursos, pero con otros estilos de trabajo. Así, se podría ver la capacidad real de la adaptación a distintos entornos del modelo. Es por esto, que el modelo debería seguir mejorando constantemente en función de las distintas realidades en la que sea aplicado.

La planificación programada se ha cumplido. Con respecto al factor tiempo se debe añadir una acotación: el tiempo de evaluaciones (reuniones posteriores a la aplicación del modelo) resulta más escaso que el tiempo dedicado a la implementación, pues al no haber un incentivo real para la organización se tiende a disminuir la disposición al trabajo en este tema, sin embargo esto no imposibilitó que se puedan haber extraído las conclusiones necesarias con respecto a la experiencia y que fueron explicadas en este documento.

# 7 Referencias


1. Tanja Vos, Jorge Sánchez, Maximiliano Mannise, *Mejorando el testeo en las PYME. ¿Cómo empezar?*, Software Quality Usability and Certification, Proceedings JTS 2008, páginas 71-80.

2. Mark Paulk, Charles Weber y Mary Beth Chrissis, *The Capability Maturity Model for Software,* Software Engineering Institute, USA, Julio 1993.

3. *CMMI for Development, version 1.3 (CMMI-DEV V1.3)*, Carnegie Mellon University-Software Engineering Institute, noviembre de 2010. Disponible vía web en: http://www.sei.cmu.edu/reports/10tr033.pdf. Revisada por última vez el 21 de febrero de 2012.

4. David Gelperin y Alden Hayashi, *"How to support better software testing"*, Application Development Trends, Mayo 1996, páginas 42-48.

5. *Test Maturity Model Integration (TMMi) Version 2.0*, TMMi Foundation. Disponible vía web en: http://www.tmmifoundation.org/downloads/tmmi/TMMi%20Framework.pdf. Revisada por última vez el 12 de septiembre de 2010.

6. Thomas Ericson, Anders Subotic y Stig Ursing, *TIM – A Test Improvement Model*, 1997.

7. *TestPAI: Un Área de Proceso de Ingeniería de Nivel de Madurez 3*. Disponible vía web en: http://sel.inf.uc3m.es/asanz/testpai/testpai_pa_fullversion_esp.pdf. Revisada por última vez el 12 de septiembre de 2010.

8. Proyecto Tutelkán, *Tutelkán- Descripción General del Proyecto*, mayo 2009. Disponible vía web en: http://www.tutelkan.org/blog/wp-content/uploads/2009/06/tutelkan-descripcion-general-del-proyecto.pdf. Revisada por última vez el 7 de noviembre de 2010.

9. Proyecto Tutelkán, *Plantillas de Apoyo al Proceso de Implementación. Plan de Pruebas,* 2010. Disponible vía web en: http://www.tutelkan.org/blog/wp-content/uploads/2009/06/tip-plantillas-y-guias.rar. Revisada por última vez el 7 de noviembre de 2010.

10. ISO/IEC 29110-5-1-2 Software Engineering – Lifecycle profiles for Very Small Entities (VSEs) – Management and engineering guide: Generic profile group: Basic Profile, 15-5-2011.

11. ISO/IEC 29119 Software Testing, *Four Parts Standard*. Disponible vía web en: http://www.softwaretestingstandard.org. Revisada por última vez el 9 de mayo de 2012.

12. IEEE 829-2008, *Standard for Software Test Documentation*, IEEE, 2008.

13. BS-7925-1/2, *Component Testing*, The British Computer Society, 1998.

14. Jari Andersin, TPI- a model for test Process Improvement, Seminario en Modelos de Calidad para Ingeniería de Software, Departamento de Ciencias de Computación, Universidad de Helsinki, 5 de octubre de 2004. Disponible vía web en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.126.6108&rep=rep1&type=pdf. Revisada por última vez el 7 de noviembre de 2010.

15. Tanja Vos, Arthur Baars y Beatriz Marín, *Pruebas funcionales evolutivas en Industria*, Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos, Vol. 4, No. 5, 2010, páginas 59-66.

16. ETF, Manual de Usuario. Disponible vía web en http://www.evotest.eu. Revisado por última vez el 11 de septiembre de 2010.

17. Abran, J. Moore, P. Bourque, R. Dupuis, L. Tripp, *Guide to Software Engineering Body of Knowledge*, IEEE, 2004.

18. Agustin Yagüe, Juan Garbajosa, *Las Pruebas en Metodologías Ágiles y Convencionales: Papeles Diferentes, Acta de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos*, Vol. 3, No. 4, 2009.