UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

"HERRAMIENTA DE SOPORTE A LA VALORACIÓN RÁPIDA DE PROCESOS SOFTWARE UTILIZANDO EL MODELO MOPROSOFT BAJO UN ENFOQUE RIA"

TESIS

PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN

PRESENTA

DAGOBERTO CRUZ SANDOVAL

DIRECTOR DE TESIS
DR. IVÁN ANTONIO GARCÍA PACHECO

HUAJUAPAN DE LEÓN, OAX.; 18 DE MARZO DE 2010

Tesis presentada el 18 de Marzo de 2010.
Ante los siguientes sinodales:
Dr. Carlos Fernández y Fernández.
M. C. Rodolfo Maximiliano Valdés Dorado
M.C. David Martínez Torres

Director de Tesis:

Dr. Iván A. García Pacheco

Dedicatoria

El presente trabajo se lo dedico especialmente a mi padre: Dagoberto L. Cruz Alavez, aunque físicamente ya no se encuentra conmigo, sus consejos y enseñanzas me acompañan a cada momento de mi vida. Su ejemplo de superación y sabiduría marcaron las bases de mi educación y el deseo de superarme día con día.

A mi madre, Yolanda G. Sandoval Ángeles por el amor incondicional que me ha brindado a lo largo de todos estos años, por el apoyo en los momentos más difíciles de mi vida y a esa fuerza descomunal para superar los problemas, que me ha servido para admirarla y amarla cada día mucho más.

A ambos por su motivación y apoyo incondicional durante mis estudios, y por legarme la herencia más grande de todas: una educación sólida que me ayude a discernir y competir en lo que me resta de vida.

Agradecimientos

Especialmente a mi director de tesis Dr. Iván A. García Pacheco, por sus instrucciones, vasto conocimiento y paciencia para llevar por buen camino el desarrollo de este trabajo, pero sobre todo por el placer de contar con su excelente amistad. Muchas gracias Doc.

A mi hermano Jahir, por sus palabras de ánimo en el momento más difícil de mi vida, por el apoyo que me ha brindado durante todos mis estudios, y por convertirse en uno de mis mejores amigos. ¡Te quiero mucho carnalito!

A mis hermanas, Nury y Mayrena por ser parte del círculo de personas que más quiero en este mundo, por su motivación a superarme y por todas las cosas que he aprendido directa o indirectamente de ellas.

A mis sobrinas, Alexia y Michelle por completar el circulo de personas especiales en mi vida y para que llegado el momento, este trabajo les sirva de ejemplo para la consecución de sus propias metas.

A mis amigos, Román, Memo, Gus, Oma y Paco que me acompañaron en numerosas aventuras durante el transcurso de la carrera y por su amistad que hizo de esta etapa de mi vida algo mucho más agradable. ¡Gracias banda!

A Josué por la ayuda brindada en la realización y presentación de este trabajo.

A todos mis amigos, en especial a Jacobo, Ángel, Rafa, Addair, Carlos, Andrea y Lulú que fuera del aspecto académico, he compartido con ellos muchos momentos gratos y me han enseñado innumerables cosas en diversos aspectos de la vida.

A toda aquella persona que haya incidido de manera positiva en cualquier aspecto de mi vida.

Dago.

Índice

Lista de tablas

Lista de figuras

Resumen

En la actualidad existen modelos y estándares de calidad que brindan ayuda a las empresas software para mejorar sus procesos de desarrollo, lo cual se ve reflejado en la calidad de los productos que generan. El estándar NMX-I-059/02-NYCE-2005 (también conocido como Moprosoft) está enfocado principalmente a las Micro, Pequeña y Medianas Empresas mexicanas y les ofrece mecanismos para la mejora de sus procesos software. El interés de estas empresas por adquirir esta certificación ha ido creciendo, ya que mejorar sus procesos de desarrollo software es una estrategia para asegurar la calidad de sus productos. Sin embargo, los métodos de evaluación para que la empresa determine el nivel de capacidad de los procesos implementados y el nivel de madurez de la organización están ligados a agentes expertos en el estándar, comúnmente externos a la organización, lo que se traduce en un alto costo para la Micro, Pequeña y Mediana Empresa. Este es el principal factor que trunca la posibilidad de darle un seguimiento periódico a la capacidad de los procesos de la organización. Este trabajo de tesis se enfoca al desarrollo de una herramienta basada en el modelo Moprosoft que brinde a la organización una evaluación fidedigna y fiable, que le ayude a monitorear periódicamente el estado de sus procesos, así como determinar la madurez de la empresa. Asimismo la herramienta ofrece una serie de pasos y sugerencias para subsanar los procesos deficientes encontrados.

Abstract

Nowadays there are models and standards which attempt to introduce quality in the enterprises' software development process with the objective to introduce high quality levels in the produced software. The NMX-I-059/02-NYCE-2005 standard (also known as MoProSoft) is focused on small and medium software enterprises, or small groups of software development within a larger organization, with the aim of promoting the standardization of an effective process in the software industry. Mexican enterprises now have a software standard that enables them to achieve a high level of quality in the software that they produce. However, the adoption of any standard is not an easy task. This paper aims to show that the development and implementation of a Rich Internet Application-based tool that could support improvement initiatives, therefore strengthening the standard adoption. SelfVation is a software tool we have developed to facilitate the implementation of SPI initiatives under the NMX-I-059/02-NYCE-2005 standard and reduce the cost of certification investment.

1. Introducción

De acuerdo con el estándar IEEE-729 [IEEE 729-1983] el software se define como un "*conjunto de programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación*". A partir de esta definición, el software va más allá de los programas de computadora, forman también parte del software los procedimientos, documentación y todos los productos que se relacionen con el desarrollo de un programa computarizado.

El software se ha convertido en un pilar fundamental en la evolución de los sistemas y productos informáticos. En las últimas décadas, el software ha pasado de ser una solución a problemas específicos y una herramienta para el manejo de información, a ser una industria autónoma. Sin embargo, a la par de este crecimiento ha surgido un conjunto de problemas que persisten hasta el día de hoy. La demanda en cantidad y calidad de software ha ido creciendo de forma muy rápida en las últimas décadas, lo que ha desatado un gran auge en la industria software, sin embargo, esta industria aún se encuentra evolucionando de una forma artesanal hacia una disciplina de ingeniería. Esto ha generado problemas tales como: el software es (casi) siempre entregado más tarde de lo previsto, es más caro de lo planeado, y tiene una funcionalidad distinta a la esperada [Standish08]; estos son los principales factores que dieron pie a la llamada "crisis del software" y que por lo visto, siguen presentándose.

De acuerdo con [Pressman02] muchas de las causas de la "crisis del software" se deben a mitos y teorías que surgieron durante los primeros años del desarrollo software. Los mitos del software propagaron información errónea y confusa. La razón que hace peligrosos estos mitos es su origen, ya que la mayoría de estos surgieron debido a las experiencias frente a ciertos eventos, tuvieron un sentido intuitivo y frecuentemente fueron promulgados por expertos de aquel tiempo. Los mitos se encuentran en las principales entidades implicadas en el desarrollo de software tal y como se resume a continuación:

- *Mitos de gestión:* Los gestores en la mayoría de las disciplinas están normalmente bajo la presión de cumplir los presupuestos, hacer que no se retrase el proyecto. Al observar desde una perspectiva totalmente ajena a la operativa, tienden a creer que el ¿qué hacer?, es igual o equivalente al ¿cómo hacer? (p.e. Se tiene un libro lleno de estándares que le proporciona a la gente todo lo que necesita saber, la gente cuenta con las herramientas más avanzadas para el desarrollo de software y con las computadoras más modernas para este fin, si se falla en la planificación se pueden añadir más programadores y recuperar el tiempo perdido).

- *Mitos del cliente:* Un cliente que solicita algún software puede desenvolverse en cualquier rubro que sienta la necesidad de una solución informática para su organización. Los mitos conducen a que el cliente se cree falsas expectativas y, finalmente, quede insatisfecho con el software desarrollado (p.e. Una declaración general de los objetivos es suficiente para empezar a programar los detalles que se puedan dar más adelante, los requisitos del proyecto cambian continuamente pero los cambios pueden acomodarse fácilmente ya que el software es flexible).
- *Mitos de los desarrolladores:* En la actualidad, los desarrolladores siguen rigiéndose por los mitos que surgieron hace alrededor de cincuenta años, enfocándose al desarrollo de un programa, y si bien el programa es parte del software no lo es todo (p.e. Una vez que se escribe y se hace que funcione un programa, el trabajo del desarrollador ha terminado. Hasta que no se tenga el programa ejecutándose no se tiene forma de comprobar su calidad, lo único que se entrega al terminar el proyecto es el programa funcionando).

La Ingeniería de Software (IS) es la medida que surge a raíz de la "crisis del software", la cual establece la premisa de ofrecer métodos y técnicas para desarrollar y mantener software de calidad. La definición formal que utiliza el IEEE se refiere a "*la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación (funcionamiento) y mantenimiento del software: es decir, el estudio de enfoques en la aplicación de Ingeniería al Software*". La meta fundamental de la IS es proporcionar un marco de trabajo para construir software con mayor calidad.

Por lo tanto, un proceso software es *un conjunto de actividades y resultados asociados que producen un producto software, son actividades genéricas que pueden organizarse de diferentes formas y describirse en diferentes niveles de detalle para diferentes tipos de software* [Sommerville05].

El *proceso software* define el enfoque que se adopta mientras el software está en desarrollo, en tanto que la IS, además del proceso software, abarca también las tecnologías que requiere dicho proceso (p.e. métodos, técnicas y herramientas automatizadas). Aunque existen muchos procesos diferentes para el desarrollo software, éste cuenta con fases genéricas que son comunes para todos ellos:

1. *Especificación*. Se debe definir la funcionalidad del software y las restricciones en su operación.
2. *Diseño e implementación*. Se debe producir software que cumpla su especificación.
3. *Validación*. Se debe validar el software para asegurar que hace lo que el cliente desea.
4. *Mantenimiento*. Se centra en el cambio que va asociado con la corrección de errores debido a las mejoras producidas por los requisitos cambiantes del cliente.

Para resolver un problema con un desarrollo software, se debe de contar con una estrategia que acompañe al proceso, métodos, herramientas y las fases genéricas. Estas estrategias son llamadas *modelo del proceso software*. Se selecciona un modelo del proceso según la naturaleza del proyecto. Por otro lado existen diferentes "modelos tradicionales" de desarrollo, entre los más conocidos se encuentran: modelo en cascada, modelo de construcción de prototipos, Desarrollo Rápido de Aplicaciones (DRA), modelo incremental, modelo espiral y modelo espiral WINWIN.

Estos "modelos" son ampliamente utilizados en el área de la IS ya que proporcionan una base para el proceso de desarrollo software, y además son significativamente más efectivos que un enfoque hecho al azar. Sin embargo, representan debilidades; la razón de esto se debe a que algunos

de estos son modificaciones de otros anteriores. Pero estas fallas no dependen solamente del modelo, sino del proyecto al que se apliquen, la elección de un modelo para un proceso software está ligada a la naturaleza del proyecto que se pretende realizar. La Tabla 1 nos indica que los principales problemas de los modelos de desarrollo han creado una gran incertidumbre entre los grupos de desarrolladores ya que se crean interrogantes referentes a: qué tipo de proyecto se intenta realizar, qué modelo es el mejor para este tipo de proyecto, y si se obtendrán los resultados esperados con la implantación del modelo elegido. La implantación de un modelo de desarrollo marca la base para conseguir la meta fundamental, la calidad del producto software. Sin embargo, no la garantiza ya que nuestro proceso puede ser de capacidad y madurez muy baja, y solo con la mejora constante se puede llegar a la consecución de esta meta.

Tabla 1. Principales problemas con los modelos de proceso software.

Modelo de Proceso Software	Principales Problemas
Cascada	• Proyectos reales raras veces se acoplan al modelo secuencial. • Requiere exponer explícitamente todos los requisitos. • Un cambio en los requisitos iníciales, produce confusión y es difícil de tratarse. • Una versión del programa estará disponible hasta que el trabajo se encuentre muy avanzado, un error grave detectado en esta etapa resulta desastroso para el proyecto.
Construcción de Prototipos	• Problemas con sistemas grandes y complejos. • El cliente puede confundirse con la versión final y el prototipo. • La gestión del desarrollo es muy lenta. • El desarrollador elige herramientas para hacer que los prototipos funcionen rápidamente, olvidándose de las características del sistema global.
DRA	• Requiere mayor cantidad de recursos humanos. • Este modelo no es aplicable si no existe un compromiso de realizar rápidamente las actividades entre clientes y desarrolladores. • Inadecuado para proyectos con riesgos técnicos altos.
Incremental	• Si un avance incremental del producto no cumple las características que el cliente requiere, será trabajo perdido ya que se tiene que volver a desarrollar dicho avance. • Ciclo de vida largo, ya que con la presentación de avances al cliente pueden surgir nuevos requerimientos que no estaban contemplados. • Mayor exigencia del cliente, el nuevo avance tiene que ser mucho mejor que el anterior.
Espiral	• Contar con un alto nivel de habilidad para la evaluación de riesgos. • Difícil de comprender y aplicar.

	• En caso de riesgos no descubiertos a tiempo, se convierten en problemas graves.
Espiral WINWIN	• Alto compromiso de parte del cliente para brindar información acerca del sistema. • Nivel alto de negociación entre cliente y desarrolladores, lo que lleva a que el grupo de desarrollo convenza al cliente de funcionalidades que no son las que desea.

En los años 80 y principios de los 90, había una opinión general de que la mejor forma de obtener un buen software era mediante una planificación cuidadosa del proyecto, garantías de calidad formalizadas y procesos de desarrollo controlados y rigurosos. Esta opinión provenía de la comunidad de ingenieros de software implicada en el desarrollo de grandes sistemas de software, los cuales en su gran mayoría se trataban de sistemas críticos. Este tipo de enfoque, involucraba una sobrecarga de trabajo en cuanto a la planificación, diseño y documentación del sistema. Dicho esfuerzo se justifica cuando se tiene que coordinar el trabajo de múltiples equipos de desarrollo, cuando se trata de un sistema crítico y cuando una gran cantidad de personas estarán implicadas en el mantenimiento del software durante su vida útil. Sin embargo, en la adopción de este tipo de enfoques por parte de negocios pequeños, el esfuerzo invertido en la planificación era tan grande que algunas veces dominaba el proceso de desarrollo. El descontento con estos enfoques pesados trajo como consecuencia la propuesta de nuevos métodos conocidos como ágiles.

Los métodos ágiles se centran en el software mismo en vez de su diseño y documentación. Los métodos ágiles dependen de un enfoque iterativo para la especificación, desarrollo y entrega del software, y principalmente fueron diseñados para apoyar al desarrollo de aplicaciones de negocio donde los requerimientos del sistema normalmente cambiaban rápidamente durante el proceso de desarrollo, están pensados para entregar software funcional de forma rápida a los clientes quienes pueden entonces proponer que se incluyan en iteraciones posteriores del sistema nuevos requerimientos o cambios en los mismos [Sommerville05]. Los métodos ágiles se basan en la noción de desarrollo y entregas incrementales, pero proponen procesos diferentes para alcanzar sus objetivos. Sin embargo, tienen principios en común que proporcionan las bases para un método ágil (véase Tabla 2).

Tabla 2. Principios de los métodos ágiles.

Principio	Descripción
Participación del cliente	Los clientes deben de estar fuertemente implicados en todo el proceso de desarrollo. Su papel es proporcionar y priorizar nuevos requerimientos del sistema y evaluar las iteraciones del sistema.
Entrega Incremental	El software se desarrolla en incrementos, donde el cliente especifica los requerimientos a incluir en cada incremento.
Personas, no procesos	Se deben reconocer y explotar las habilidades del equipo de desarrollo. Se les debe dejar desarrollar sus propias formas de trabajar, sin procesos formales.
Aceptar el cambio	Se debe contar con que los requerimientos del sistema cambian, por lo que el sistema debe de dar cabida a estos cambios.

Mantener la simplicidad

Se deben de centrar en la simplicidad tanto en el software a desarrollar como en el proceso de desarrollo. Donde sea posible, se trabaja activamente para eliminar la complejidad del sistema.

La programación extrema (XP) es el método ágil más conocido y ampliamente utilizado. Debe su nombre al uso de prácticas reconocidas, como el desarrollo iterativo, y a la participación del cliente en niveles "extremos". En XP, todos los requerimientos se expresan como escenarios, los cuales se implementan directamente como una serie de tareas. Los programadores trabajan en parejas y desarrollan pruebas para cada tarea antes de escribir código. Todas las pruebas se deben ejecutar satisfactoriamente cuando el nuevo código se integre al sistema. Existe un pequeño espacio de tiempo entre las entregas del sistema. La XP implica varias practicas (véase Figura 1.1), que se ajustan a los principios de los métodos ágiles.

Figura 1.1. Ciclo de entrega de la XP.

Si bien el enfoque de métodos ágiles también posee principios difíciles de realizar entre los que se cuentan:

1. La idea de la participación del cliente en el proceso de desarrollo es atractiva, pero el éxito depende de tener a un cliente que esté dispuesto y pueda pasar tiempo con el equipo de desarrollo, y esto se torna difícil ya que están sometidos a otras presiones o actividades y no pueden participar plenamente en el desarrollo del software.
2. Los miembros individuales del equipo pueden no tener la personalidad o capacidad apropiada para la participación intensa que se requiere en los métodos ágiles, por lo cual es posible que no se relacionen adecuadamente con los otros miembros del equipo.
3. Priorizar los cambios puede ser extremadamente difícil, especialmente en sistemas en los que existen gran cantidad de *stakeholders*[1].
4. Mantener la simplicidad requiere un trabajo extra. Los miembros del equipo pueden no tener tiempo de llevar a cabo las simplificaciones deseables si trabajan bajo presión.

En consecuencia, los métodos ágiles son sólo apropiados para algunos tipos de desarrollo. Son los más idóneos para el desarrollo de sistemas pequeños o de tamaño medio, pero no son adecuados

[1] De acuerdo a [Kotonya00] los *stakeholders* son la gente u organizaciones que tienen influencia directa o indirecta en los requisitos del sistema o quiénes pueden afectar dicho sistema.

para sistemas a gran escala. Tampoco deben de ser utilizados para el desarrollo de sistemas críticos en los que es necesario un análisis detallado de todos los requerimientos del sistema para comprender sus características de seguridad o protección.

Sin embargo, ninguna de las soluciones mencionadas anteriormente ha sido capaz por sí sola de proporcionar alivio a la "crisis del software", ya que si bien el foco de interés de los esfuerzos se ha situado en diversas áreas, con frecuencia se ha dirigido a la búsqueda de soluciones óptimas a problemas particulares. Muestra de lo anterior se refleja en los estudios realizados por el Standish Group [Standish04], el informe *Chaos Extreme*[2], establece que del año 2000 al 2006 el abandono de proyectos software y los proyectos software que no cumplían con los requisitos del usuario siguen siendo problemas latentes y redundantes dentro de la industria software, también indica el aumento paulatino del porcentaje de proyectos software terminados (véase Figura 1.2).

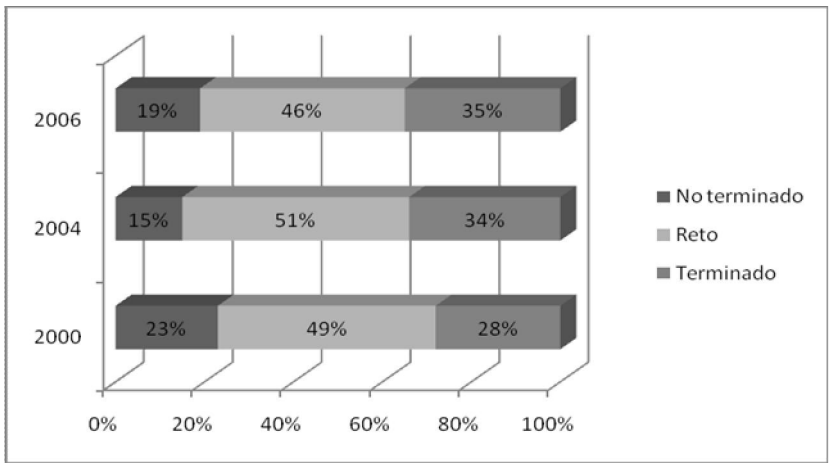


Figura 1.2. Resultados del estudios CHAOS (2000-2006)

El mismo reporte indica que el 35% de los proyectos software fueron terminados, el 19% no fueron terminados y un 46% no cumplió con los requisitos establecidos por el cliente (véase Figura 1.3).

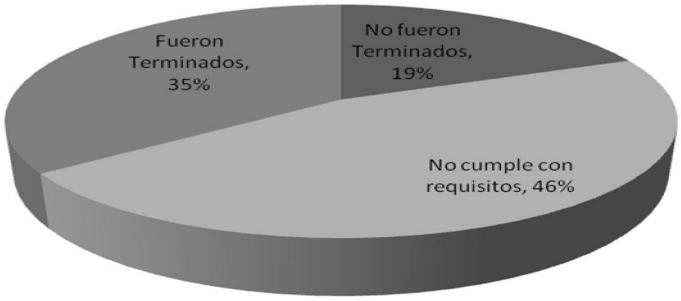


Figura 1.3. Porcentaje de proyectos software terminados, no terminados y que no cumplieron con los requisitos del cliente en el año 2006.

[2] Desde 1994 el reporte del Standish Group, el famoso Chaos Report, se ha vuelto en el patrón dorado sobre la calidad de los proyecto software. Estos estudios demuestran la situación de Europa y Estados Unidos en relación al éxito y/o fracaso de los proyectos informáticos. Para más información consultar www.standishgroup.com

De acuerdo a la Figura 1.4, otro de los puntos que el Standish Group refleja en su informe como causas principales del por qué los proyectos software no cumplen con los requisitos establecidos por el cliente: en primer lugar el 45% de los proyectos software superan los costos establecidos para su desarrollo, el 63% los supera en materia de tiempo y en promedio sólo se entrega al cliente un 67% de la funcionalidad total del sistema.

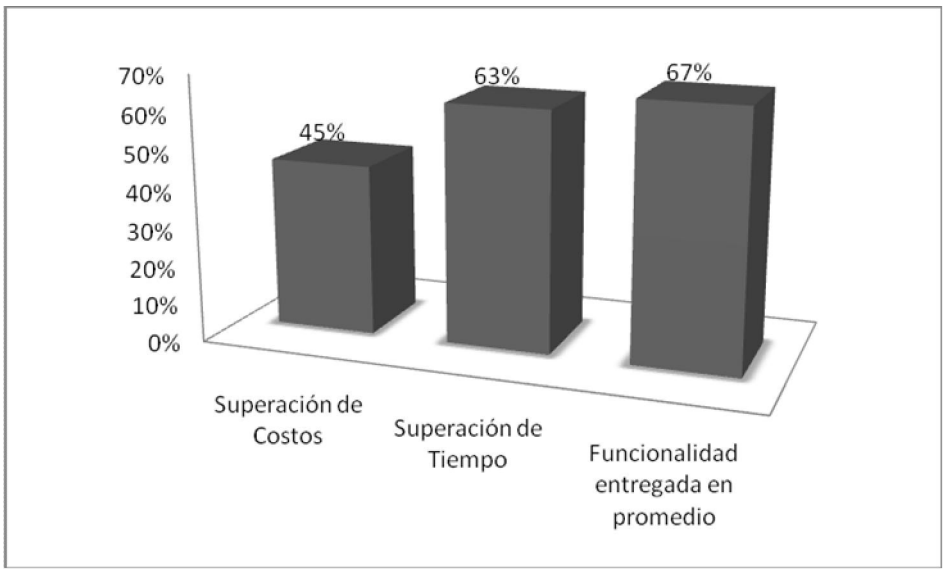


Figura 1.4. Principales causas de no cumplir con los requisitos del cliente en proyectos software.

Los resultados obtenidos por el Standish Group reflejan los problemas que existen en la actualidad: la industria software ha intentado incrementar la productividad y la calidad con la aplicación de nuevas metodologías y tecnologías, pero se ha reconocido que el problema fundamental es la incapacidad de gestionar el proceso software. De esta manera, el interés de las empresas ha estado cambiando de soluciones basadas en tecnología a soluciones basadas en el proceso.

En los últimos años, la Mejora del Proceso Software (SPI, *Software Process Improvement*) se ha convertido en un tema importante para la IS y para las empresas software que buscan ser competitivas en un mercado mundial cada vez más difícil. Uno de los principales objetivos de SPI es producir software de calidad a tiempo, dentro del presupuesto y con el funcionamiento deseado. La calidad en SPI está basada en la premisa de que procesos maduros y capaces generan productos de calidad [Serrano06]. Además, SPI debe de ser una actividad continua durante la duración de un proyecto. Los enfoques de mejora pueden darse a nivel organizacional y/o a nivel técnico. A nivel organizacional son más económicas pero las mejoras no se ven a corto plazo, mientras que a nivel técnico son más costosas pero las mejoras se ven a corto plazo [Pino06]. Enseguida, Pino muestra las propuestas de mejora (véase Tabla 3).

Tabla 3. Propuestas de mejora.

%	Propuestas de Mejora
18%	Establecimiento de procesos software. Usar una guía electrónica de proceso – ERP y repositorio de experiencias – ER. Adaptar y utilizar practicas de RUP, XP, SCRUM, entre otras. Autovaloración de procesos por los empleados (herramienta WEB).
9%	Priorizar los esfuerzos de SPI (a través del método DAIIPS, Software Process Matrix, Express Process Appraisal ó Framework para tomar decisiones de negocio y producto).
4%	Evaluación de un programa SPI (poco riguroso, ó definir y usar un programa de métricas).
13%	Guiar los esfuerzos de SPI (método Pr2imer, Framework IMPACT, método/modelo MESOPYME, usando patrones de mejora, siguiendo un proceso de valoración, redes neuronales).
45%	Adaptación y utilización de estándares SPI (PSP, TSP, CMM, CMMI, IDEAL, ISO 15504:2004, SPICE, ISO 9001).
11%	Definición y utilización de Framework para pruebas. Conducción de un experimento SPI. Usar gestión de conocimiento para SPI. Adquirir infraestructura técnica lista para usar. Mejorar la relación y cooperación con el cliente.

En la tabla anterior se puede observar que la mayor concentración de mejora (9%+4%+13%+45%=71%) radica en guiar un proyecto de mejora, priorizar la implementación de las mejoras, utilizar modelos de mejora existentes ajustándolos a las necesidades y evaluar las mejoras introducidas por el programa SPI. Además hay otras propuestas de mejora (18%) centradas en la definición, evaluación y soporte de los proceso software [Pino06].

El ciclo de Deming, propuesto en [Deming86], muestra que el seleccionar un enfoque bien definido es un buen punto de partida para conducir un esfuerzo de SPI, y se requiere de la medición de los procesos para conectar la evaluación y mejora de manera efectiva (véase Figura 1.5).

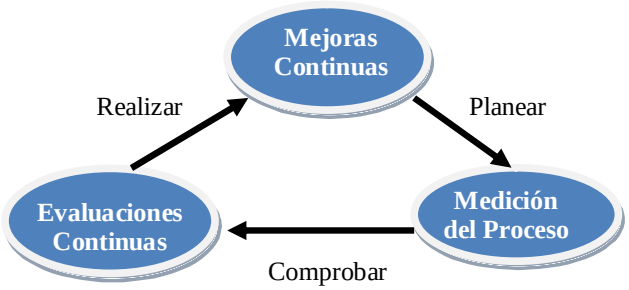


Figura 1.5 Ciclo de mejora continúa de procesos

El programa SPI involucra diferentes tipos de modelos y métodos [Pino06]:

- El modelo que conduce a la mejora el cual describe la infraestructura, actividades, ciclo de vida y consideraciones prácticas para la evolución de los procesos.
- El método de evaluación de procesos especifica la ejecución de la evaluación para producir un resultado cuantitativo que caracterice la capacidad del proceso o la madurez de la organización.

- El modelo de procesos de referencia describe cuáles son reconocidas como las *mejores prácticas*[3] que una organización debe de implementar para el desarrollo de software.

Durante los últimos años, han sido desarrollados muchos métodos para establecer una iniciativa de SPI como una alternativa para lograr el crecimiento de la calidad de los productos y servicios que una empresa software brinda. Con la adopción de las mejores prácticas del software se amplía la gama de ventajas, se hace mejor uso de los métodos y tecnologías actualmente disponibles en el desarrollo software. Los modelos de mejora buscan reconocer las oportunidades de mejora y establecer un programa de mejora mediante la compresión y el conocimiento de los costos y beneficios que conlleva su implementación. El creciente número de artículos que tratan el tema de acuerdo al análisis de la tendencia de publicaciones de SPI presentado en [Hall02], así como la aparición de un gran número de iniciativas internacionales relacionadas con SPI, entre las que se encuentran CMMI-DEV [CMMI06], Moprosoft [NYCE05a], COBIT 4.0 [COBIT05], ISO/IEC 15504 [ISO04]. Además la norma ISO 9001:2000 [ISO00] que está siendo utilizada en este campo.

Los modelos de proceso de mejores prácticas ayudan a definir los planes de acción dentro de la organización, los cuales incluyen la definición de los procesos a mejorar. Mediante estos modelos las organizaciones han empezado a afrontar el reto de SPI. Los modelos de proceso brindan a las empresas una serie de actividades y guías de qué hacer para realizar una actividad, pero no indican cómo implementar las mejores prácticas dentro de la organización. Sin embargo, toda iniciativa de mejora necesita de un Modelo de Evaluación del Proceso Software (SPA, *Software Process Assessment*) que determine cuál es el estado actual del proceso software, sus fortalezas y debilidades mediante la comparación de las prácticas existentes en la empresa con las de un modelo de proceso de mejores prácticas. Modelos para SPA como SCAMPI [SCAMPI06], ISO/IEC 15504, Evalprosoft [NYCE05b] los cuales son compatibles con un modelo de proceso de mejores prácticas.

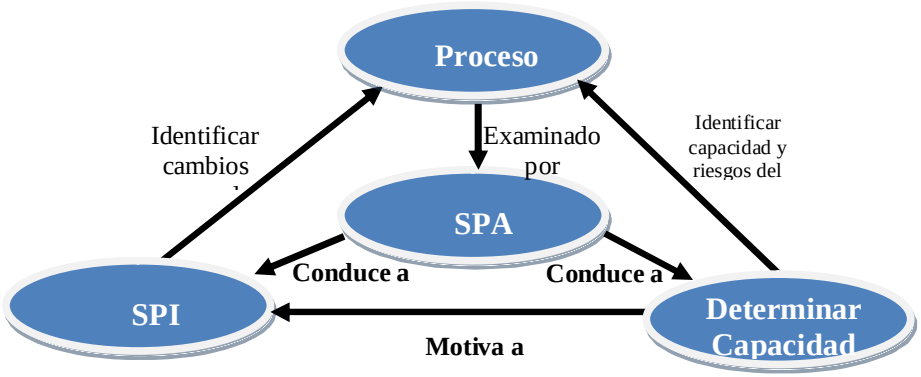


Figura 1.6. Contexto de trabajo de SPA**.**

La Figura 1.6 muestra que SPA permite identificar la capacidad de los procesos y en base a los resultados de la evaluación se puede implementar la iniciativa de SPI mediante la identificación de fortalezas, debilidades, riesgos y la prevención de estos. De esta manera el SPA es una etapa clave para el desarrollo eficiente de un esfuerzo destinado a la SPI [Hunter01]. Cuando las empresas

[3] De acuerdo a [Walker03] *"las mejores prácticas son todas aquellas metodologías y herramientas que mejoran consistentemente la productividad y calidad de los productos software cuando son implementadas, y al aplicarlas se tiene un beneficio directo en el proyecto"* Por lo tanto, las mejores prácticas de software son aquellas prácticas que las personas con experiencia reconocida en el área específica tienen identificadas a través de la experiencia y esa contribución se hace significativa para el buen término del proyecto software [Adams04].

llevan a cabo una evaluación, aumentan la comprensión de su estado actual en su gestión de actividades, lo que les muestra un potencial de mejora, en pocas palabras, SPA establece una organización basada en medios importantes para evaluar y mejorar el comportamiento de la empresa [Loon04].

1.1. Importancia del problema y necesidad de la solución

Las Pequeñas y Medianas Empresas (Pymes) son una pieza importante en el desarrollo de la economía mundial, por ejemplo, las microempresas (nueve o menos empleados en su plantilla) representan el 93% de empresas en Europa, un 56% en EUA y un 66% a nivel mundial [OECD02]. La industria software en la mayoría de los países está compuesta en gran parte por Micro, Pequeñas y Medianas Empresas desarrolladoras de software (MPyMEs) que favorecen a las economías nacionales [Pino06].

Aun cuando las MPyMEs representan un alto porcentaje en la industria software, la mayoría de programas SPI sólo han funcionado en escenarios concebidos para las grandes compañías, las cuales cuentan con los recursos suficientes para trabajar con este tipo de prácticas. El mayor argumento se debe a la infraestructura de las MPyMEs, al costo que los programas de mejora implican y a que los estándares de mejora propuestos internacionalmente por organismos como el *Software Engineering Institute* (SEI) y la *International Organization for Standardization* (ISO) no han sido creados para éste tipo de empresas sino para empresas grandes [Hareton01]. A pesar de esto la mejora del software se ha extendido a las pequeñas organizaciones, ante la necesidad de alcanzar la mejora en la calidad de los servicios y productos que como empresa proporcionan, con el fin de posicionarse en un mercado cada vez más competitivo. De aquí el interés de la comunidad científica en abordar el SPI en las MPyMEs, así lo refleja el creciente número de publicaciones acerca del tema en los últimos años mostrados en el estudio [Pino06] (véase Figura 1.7) y la creación de [URL-1]. De acuerdo al mismo estudio, se reporta en forma ascendente los ocho primeros países que han destinado más esfuerzo en materia de mejora a las MPyMEs son México, Suecia, EUA, Brasil, Dinamarca, Finlandia, Irlanda y Australia (véase Figura 1.8).

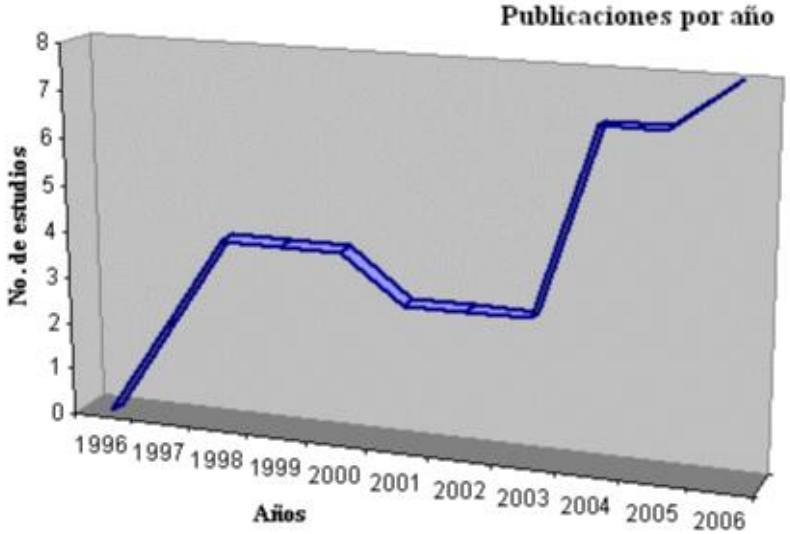


Figura 1.7. Tendencia de publicaciones de SPI en MPyMEs.

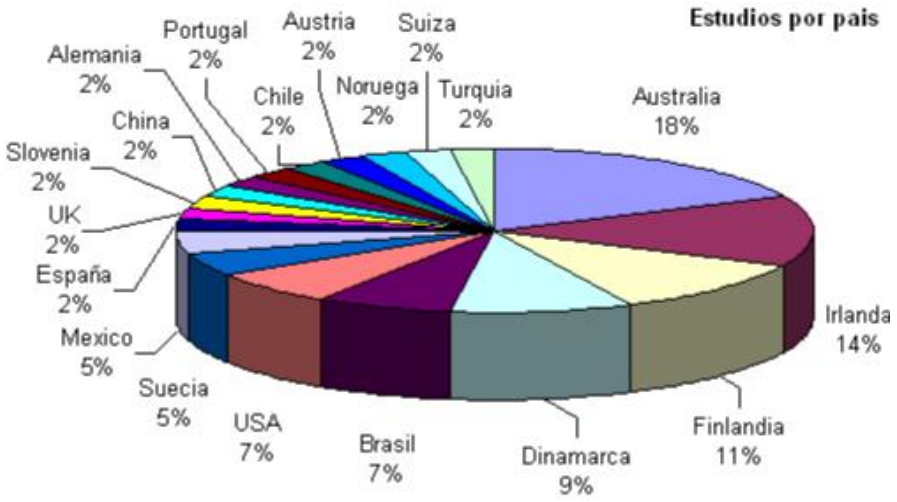


Figura 1.8. Distribución de los reportes de mejora por país.

México cuenta con una posición favorable para convertirse en un competidor de talla mundial en materia de desarrollo de software, esto principalmente gracias a su ubicación geográfica, perfil demográfico y estado de desarrollo tecnológico [González06]. De acuerdo a una investigación realizada sobre el nivel de madurez y capacidad de procesos de la industria TI de México, realizada por la Secretaria de Economía de México (SE) a través del Programa para el Desarrollo de la Industria del Software (PROSOFT) en el año 2004, aplicada a 123 empresas, el perfil fue el que se resume en la Tabla 4 [SE04]:

Tabla 4. Perfil de la industria software en México.

Tamaño	Número de Empleados	Promedio de Empleados	Número de Empresas	%
Micro	1 a 10	5	48	39
Pequeña	11 a 50	25	53	43
Mediana	51 a 100	75	12	9.8
Grande	101 o mas	101	10	8.2
	Total		123	100

El mismo estudio revela que el 91.8% de la industria software son MPyMEs. Otro estudio de la SE en conjunto con ESANE Consultores, muestra que el 75% de la industria software y servicios relacionados tiene 20 empleados o menos, y cerca del 46% tienen 1 a 10 empleados. Tal y como se observa en la Figura 1.9, las empresas grandes que cuentan con más de 100 empleados representan sólo el 6.6% [ESANE04].

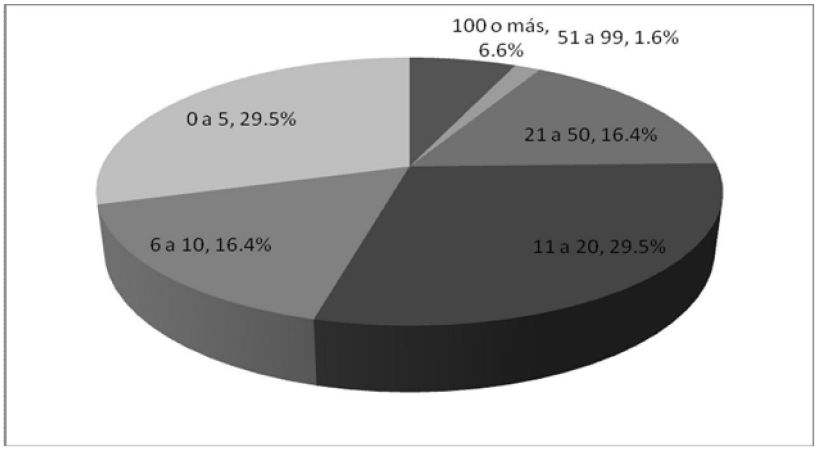


Figura 1.9. Número de empleados en las empresas de la industria software.

La industria software en México se caracteriza por ser joven, según el estudio realizado en [González06] el 47% de las empresas fueron creadas hace menos de 7 años. La antigüedad media de las empresas que participaron en el estudio es cercana a los 9 años. Las empresas más antiguas se encuentran en el mercado desde hace 25 años y las más jóvenes son menores a un año. Tomando en cuenta el tiempo de permanencia en el mercado se identifican tres bloques: emergentes, maduras y consolidadas (véase Tabla 5).

Tabla 5. Antigüedad de las empresas software en México.

Antigüedad	Porcentaje
Emergentes (entre 0 y 7 años)	47.1
Maduras (entre 8 y 15 años)	42.6
Consolidadas (más de 16 años)	10.3

Sobre la ubicación geográfica de las empresas software de México, el estudio refleja que en tan sólo 4 estados se concentra más del 70% de las empresas ligadas a este sector: Jalisco, D.F., Sinaloa y Nuevo León (véase Figura 1.10) [ESANE04]. Sobre el número total de empresas software en México no existen cifras recientes ni precisas. Sin embargo, en [ESANE04] se estima que a nivel nacional pueden existir cerca de 1,500 empresas.

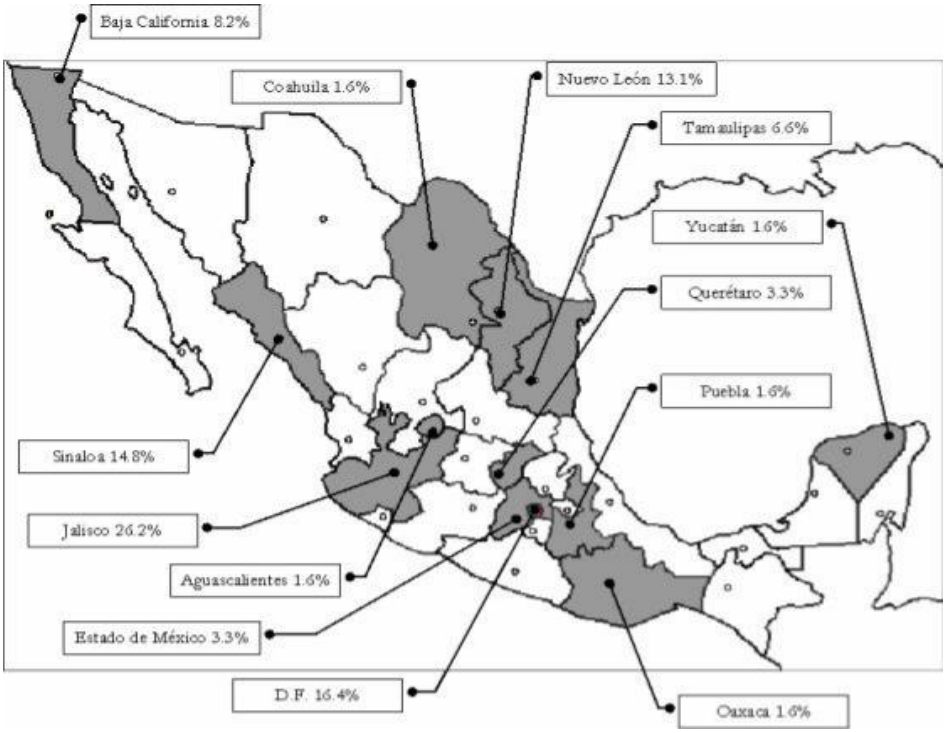


Figura 1.10. Distribución geográfica de empresas software en México.

La industria software mexicana cuenta con un gran potencial de crecimiento, sin embargo, se encuentra en una etapa aún emergente. Es una industria aún joven, más del 60% de las empresas del sector tiene menos de 10 años, y es dominada como MPyME con un 83%. Aunque existe una práctica generalizada de adopción de modelos de proceso, los niveles actuales de certificación de la industria son muy bajos. Según un estudio del Sistema Nacional de Indicadores de la Industria de Tecnologías de Información (SNIITI) para el año 2007 [URL-2], existían 57 empresas con una certificación en modelos de proceso como CMM, CMMI y Moprosoft, gran parte del porcentaje de estas empresas radica en los cuatro estados que cuentan con más empresas software dentro de su territorio: D.F., Jalisco, Sinaloa y Nuevo León (véase Figura 1.11).

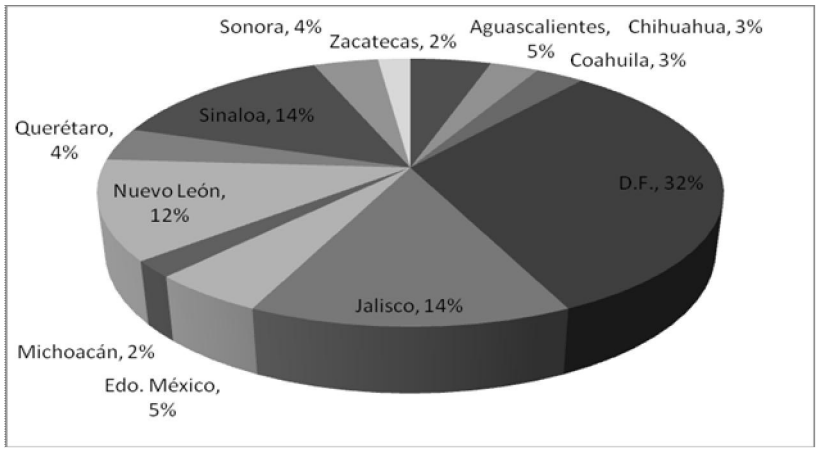


Figura 1.11. Participación por entidad en modelos de proceso.

El D.F. es la entidad que cuenta con la mayor cantidad de empresas con un modelo de proceso establecido, formada con un total de 18 certificaciones, de las cuales cinco son de CMM, diez en CMMI y tres en Moprosoft, seguido de Jalisco y Sinaloa con 8 certificaciones entre CMMI y Moprosoft (véase Figura 1.12).

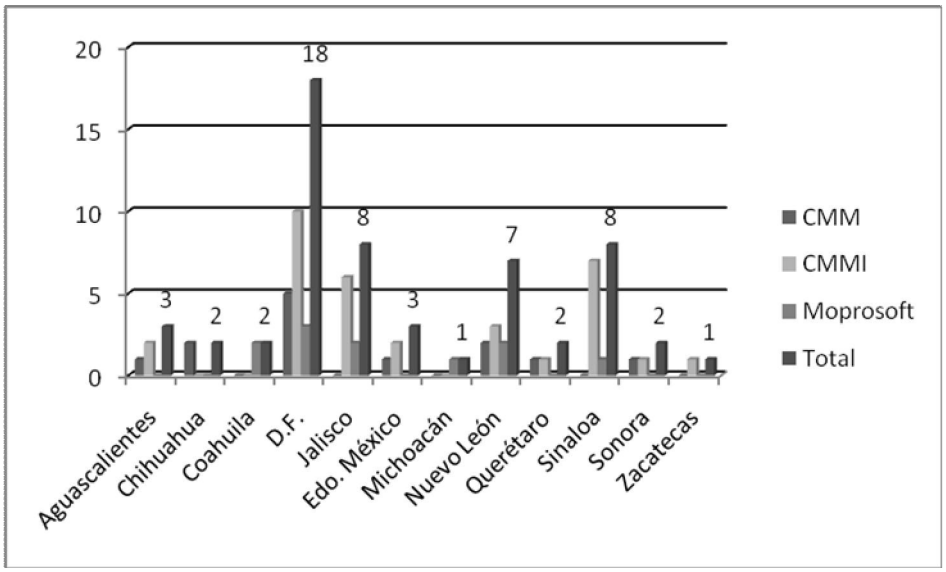


Figura 1.12. Comparativo de modelos de proceso en México por entidad.

El estudio muestra que los modelos de proceso de mayor implementación en el país son CMM y CMMI. Sin embargo, los elevados costos de adopción de estos modelos se ven lejanos para las MPyMEs, ya que las empresas que cuentan con certificaciones en estos modelos son empresas de gran envergadura y edad madura como Hildebrando Software Factory, IBM, y Softeck, por dar algunos ejemplos. Cabe mencionar que algunas de estas empresas figuran en el ámbito internacional. Esto refleja el gran problema que representa para las MPyMEs la implementación de modelos de proceso internacionales, ya que al estar desarrollados para empresas grandes y tener un alto costo, hace que la implementación se convierta en una tarea incosteable para estas empresas.

Dada esta problemática, en el año 2002 el gobierno de México comienza un programa para fomentar el crecimiento de la industria software. El nivel de su capacidad de procesos promedio de las empresas desarrolladoras de software es 0.9 (en una escala del 0 a 5 del modelo ISO/IEC 15504). Para incrementar la competitividad de la industria se necesitaba adoptar un programa masivo de SPI, pero los recursos, el capital, y los expertos en los modelos internacionales era limitada, proporcionar un modelo SPI de bajo costo era la estrategia del gobierno para conseguir esta meta [Oktaba06]. La selección de un modelo de proceso software y un método de evaluación, accesibles para la industria local, fue el primer problema que se intento resolver. El gobierno y la industria definieron un criterio de selección para evaluar la conveniencia de adoptar alguno de los más populares estándares y modelos. La conclusión de este análisis fue que ninguno de los modelos cubría totalmente los criterios de selección. A consecuencia de esto el gobierno de México decidió desarrollar el Modelo de Procesos para la Industria de Software (Moprosoft) y Evalprosoft [NYCE05] un método de SPA.

Mientras que la adopción de Moprosoft ha ido aumentando entre las MPyMEs poco a poco, mediante la adquisición de la norma NMX-059-NYCE-2005, la evaluación del modelo sigue estando en manos de un organismo rector, el cual a través de un costo monetario acude a las empresas a realizar el diagnostico y evaluación de la empresa, forzando así a las empresas a utilizar

esta única forma de evaluación para este modelo. Las evaluaciones en Moprosoft se componen por una serie de actividades y un proceso de éxito compuesto de varios pasos (véase Figura 1.13) [Flores08]. Ante la exhortación de que SPI debe ser una actividad continua, lo anterior se convierte en un obstáculo, ya que para determinar la capacidad de forma fiable de Moprosoft se depende de un organismo externo a la empresa lo que representa un costo monetario.

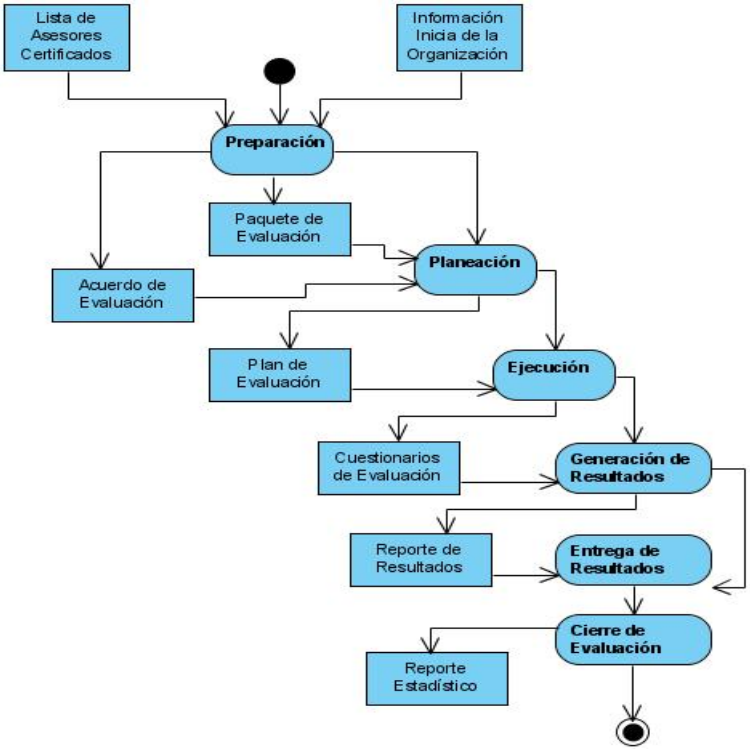


Figura 1.13. Directrices para actividades de evaluación.

En la actualidad existen herramientas de SPA basadas en modelos internacionales, las cuales pueden convertirse en la solución para empresas que desean conocer la capacidad de sus procesos de manera interna, y con esto tomar las medidas necesarias para una implementación satisfactoria. Aunque estas herramientas están disponibles en el mercado, ninguna de ellas está basada en el modelo Moprosoft, por lo que para las MPyMEs mexicanas interesadas en la adopción del modelo nacional, dichas herramientas no proporcionaran resultados completamente apegados a los procesos y practicas propuestos por el modelo, ya que estas herramientas están desarrolladas en base a prácticas, procesos, categorías y definiciones de otro modelo, lo cual además de no brindar un resultado fiable puede presentar confusión en cuanto a términos durante el proceso de evaluación.

De lo anterior surge la necesidad de realizar una herramienta gratuita completamente basada en Moprosoft, apegada a las prácticas y procesos del modelo, y que brinde el componente SPA a la empresa, la cual proporciona una parte esencial para un flujo continuo del SPI, ya que por medio de la herramienta la empresa puede evaluarse de manera continua y de esta manera conocer el nivel de madurez/capacidad con el que cuenta, así como también conocer las prácticas en las cuales debe mejorar para alcanzar un nivel deseado. De esta manera la empresa podrá recurrir al organismo rector para obtener la certificación, con la seguridad de que cumple con las prácticas necesarias para alcanzar un cierto nivel de madurez del modelo.

1.2. Delimitaciones de la tesis

Este trabajo de tesis se enfoca en brindar un mecanismo de evaluación a las MPyMEs desarrolladoras de software relacionadas o interesadas en trabajar con el modelo Moprosoft, con el propósito de conocer la madurez/capacidad actual de sus procesos de desarrollo y así propiciar un ciclo de mejora en los mismos.

La implementación del mecanismo de evaluación será a través del desarrollo de una herramienta automatizada (SelfVation), cuyos resultados estarán delimitados a:

- Las empresas que serán evaluadas deben ser MPyMEs desarrolladoras de software.
- SelfVation, se centra únicamente en evaluar empresas y generar planes de acción para la mejora.
- La solución propuesta se delimita a proporcionar un marco de evaluación basado en las prácticas de la categoría Operación compuesta por los procesos Administración de Proyectos Específicos y Mantenimiento y Desarrollo de Software, del modelo Moprosoft.
- El producto de esta tesis tiene el fin de demostrar la aplicabilidad del mecanismo de evaluación desarrollado. No se realizarán pruebas de usabilidad a usuario, puesto que se considera más importante la aprobación del contenido sistemático.

1.3. Limitaciones

- La investigación de esta tesis se limita a diseñar y construir un mecanismo para la obtención y análisis de datos.
- La herramienta desarrollada será probada en MPyMEs desarrolladoras de software de los estados de Oaxaca y Tlaxcala, las cuales cuentan con la implementación del modelo Moprosoft y conocen su estructura.
- Por otra parte será evaluada una empresa desarrolladora de software del estado de Oaxaca la cual no tiene implantado ningún modelo de procesos y desconoce la estructura del modelo Moprosoft.

1.4. Objetivos del trabajo

Los objetivos del trabajo están compuestos de un objetivo general y varios objetivos secundarios, los cuales son descritos a continuación:

1.4.1. Objetivo general

Diseñar y construir una herramienta bajo el enfoque SPA, basada en el modelo Moprosoft, la cual permita conocer la madurez/capacidad de los procesos en las MPyMEs desarrolladoras de software. Esta herramienta le permitirá conocer los procesos inmaduros en el desarrollo de software, y proporcionarle un plan de acción en base a las mejoras prácticas que permitirá mejorar su proceso software.

1.4.2. Objetivos específicos

Para alcanzar el objetivo general será necesario conseguir ciertos objetivos secundarios. Estos establecerán las aportaciones esperadas al final de la tesis:

1. Desarrollar un estudio exploratorio acerca de la estructura del modelo de proceso Moprosoft y el modelo de evaluación de procesos Evalprosoft.
2. Desarrollar un estudio de herramientas de evaluación similares basadas en el modelo Moprosoft, si es que existiesen.
3. Desarrollar un mecanismo de evaluación que permita identificar adecuadamente las fortalezas y debilidades de los procesos de desarrollo software de las MPyMEs desarrolladoras de software.
4. Desarrollar un mecanismo de guía para las MPyMEs desarrolladoras de software, la cual brinde información para la mejora de procesos y la obtención de procesos maduros de acuerdo a las necesidades especificas de este tipo de empresas.

1.5. Solución propuesta

La propuesta de esta tesis pretende auxiliar a las MPyMEs desarrolladoras de software a identificar sus fortalezas y debilidades por medio de un sistema de evaluación interno, confiable y seguro que se apoya en el modelo Moprosoft y que permita identificar la situación actual del proceso de desarrollo de software en la organización. El mecanismo también permitirá identificar la madurez de los procesos actuales, reconociendo actividades frágiles, y mediante el análisis de estos datos generar un plan de acción. El análisis hará uso de factores como:

- Áreas débiles y fuertes,
- Actividades del modelo del proceso utilizado como referencia,
- Madurez del proceso de desarrollo de acuerdo al modelo de procesos usado como referencia, entre otros.

Los planes de acción consisten en un conjunto de acciones para mejorar el proceso software, algunas de estas acciones de mejora deben interactuar o apoyarse para alcanzar los objetivos propuestos en un corto o largo plazo [SPICE07]. La generación de acciones está ligada a las actividades propuestas del modelo de proceso de referencia, analizando cuales de estas actividades se adaptan a las características de la empresa evaluada. El conjunto de acciones encontradas serán presentadas en forma de reporte, indicando cuáles de estas acciones se necesitan implementar en un lapso de tiempo corto, mediano y largo.

Igualmente se pretende que este mecanismo sirva de apoyo para que las MPyMEs mejoren la calidad de sus procesos y como consecuencia la calidad de sus productos, a través de la recomendación de prácticas propuestas por un modelo de procesos enfocado principalmente a la estructura de este tipo de empresas. La propuesta se limita a proporcionar un marco teórico de mejora de procesos basados en las prácticas de la categoría de Operación del modelo Moprosoft, ya que esta categoría se caracteriza por enfocarse al proceso de desarrollo de las empresas desarrolladas de software, cómo es planeado, ejecutado y probado. La categoría Operación está compuesta a su vez por las áreas de proceso Administración de Proyectos Específicos y Desarrollo y Mantenimiento de Software.

El modelo de proceso de referencia fue seleccionado en base a que se ha convertido en una norma y estándar nacional, y está enfocado a la estructura de MPyMEs desarrolladoras de software

mexicanas y a su alta demanda en los últimos años en este tipo de organizaciones. Además, la falta de herramientas de autoevaluación interna que permitan a las MPyMEs desarrolladoras de software mejorar sus procesos software sin incrementar los costos para llevarlo a cabo, hace pertinente el desarrollo de este mecanismo. La experimentación se llevara a cabo mediante la construcción de una herramienta computarizada que implemente el mecanismo de evaluación propuesto y la generación de planes de acción que proporcionen una guía a las empresas desarrolladoras de software sobre las actividades a realizar para implementar un proceso de SPI. Conjuntamente, la herramienta debe brindar una estimación del nivel de madurez de la empresa adoptando la escala propuesta por el modelo de proceso de referencia, en base al análisis de su estado actual (véase Figura 1.14).

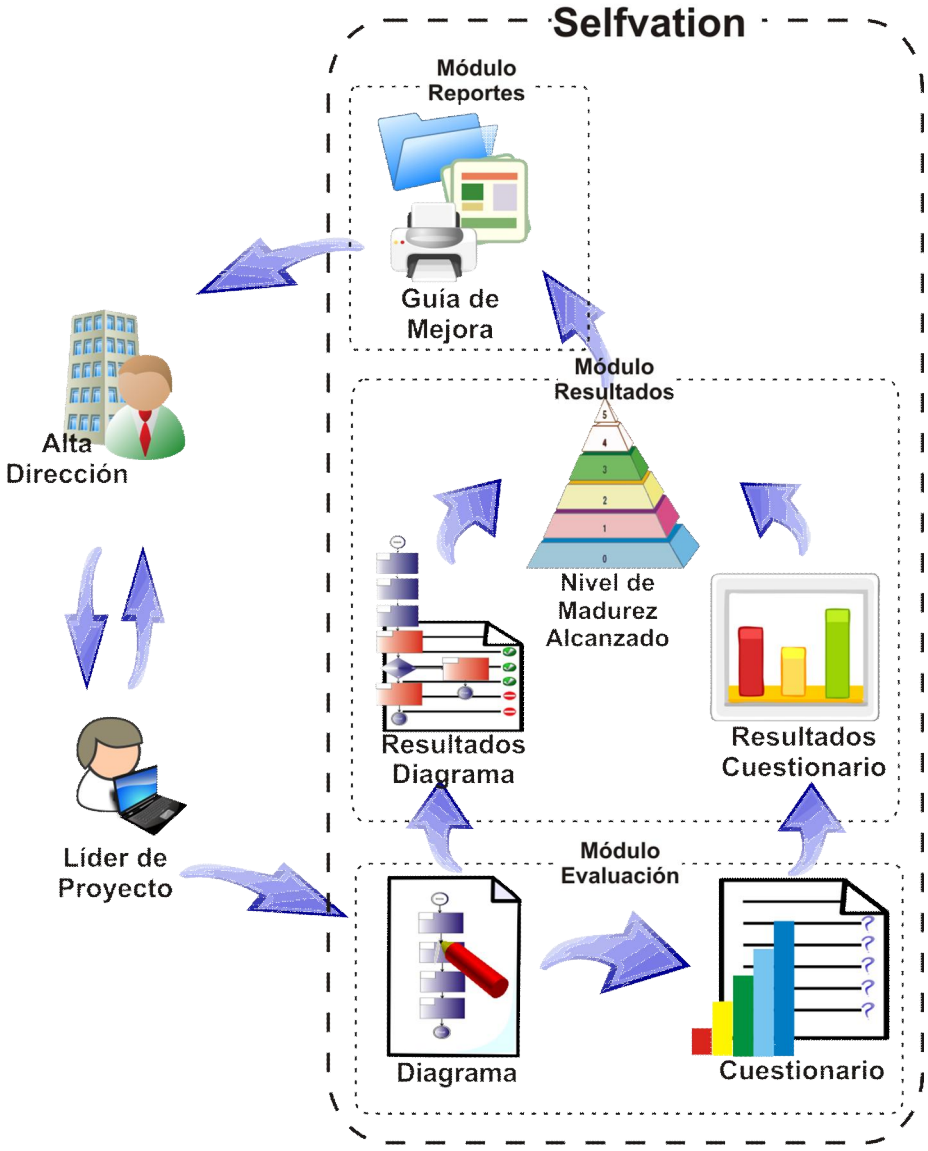


Figura 1.14. Descripción general de objetivos específicos a alcanzar.

La propuesta tiene como principal factor de innovación el trabajo conjunto de dos tipos de evaluaciones, la primera basada en un cuestionario, el cual nos proporcionara el saber qué prácticas y de qué manera se llevan a cabo estas prácticas que propone el modelo dentro de la organización, y

la segunda basada en conocer el proceso actual de desarrollo en la empresa, mediante la esquematización de un diagrama de proceso que nos brinde la situación actual de la empresa y posteriormente se realice un mapeo con el proceso sugerido por el modelo y brinde una aproximación de que tan apegado esta el proceso actual en la empresa con el propuesto en el modelo. A partir del resultado de las dos evaluaciones se proporcionará una estimación del nivel de madurez de sus procesos de acuerdo a los niveles del modelo de referencia (véase Figura 1.14). El primer tipo de evaluación es una forma utilizada con anterioridad basada en un cuestionario y posibles respuestas, agregándole un marco más interactivo diferente al entorno estático utilizado en otras herramientas similares. La segunda propuesta de evaluación integra más elementos gráficos lo cual se traduce en una forma nueva e interactiva de evaluación, utilizando componentes que no han sido explorados en este tipo de herramientas.

Ante la necesidad de establecer un marco interactivo de mayor nivel, las características de las herramientas de desarrollo disponibles para Web tradicional no cumplen con los requerimientos necesarios para este tipo de aplicaciones. De lo anterior, surge la necesidad de utilizar tecnologías bajo un enfoque RIA (*Rich Internet Applications*), el cual tiene como principales características a utilizar:

- No producen recargas de página Web, ya que desde el principio se carga toda la aplicación, y sólo se produce comunicación con el servidor cuando se necesitan datos externos como datos de una Base de Datos o de otros ficheros externos.
- Mejora importante en la experiencia visual, que hacen del uso de la aplicación algo muy sencillo, ofrece mejoras en la conectividad y despliegue instantáneo de la aplicación, agilizando su acceso.
- Garantizan la desvinculación de la capa de presentación, es decir acceso a la aplicación desde cualquier computadora en cualquier lugar del mundo.
- Ofrecen aplicaciones interactivas que no se pueden obtener utilizando solo HTML.
- Más capacidad de respuesta, ya que el usuario interactúa directamente con el servidor, sin necesidad de recargar la página.
- No necesitan instalación (solo es necesario mantener actualizado el navegador Web).

La idea plantea desarrollar una herramienta bajo un enfoque RIA, que brinde a las MPyMEs desarrolladoras de software un método de SPA que las ayude a darle fluidez al ciclo de SPI [Hunter01], sin la necesidad de depender para esto de un organismo externo sino de la empresa por si sola y a partir de las autoevaluaciones periódicas tener un proceso de mejora continuo; ya que esto se traduce en el desarrollo de productos de calidad, además de la consecución de una certificación en un nivel de madurez determinado en el modelo.

1.6. Estructura de la tesis

La estructura del documento de tesis se detalla a continuación:

El capitulo 2 presenta el marco teórico sobre el modelo de proceso y método de evaluación, el cual sirve de base para desarrollar el mecanismo de evaluación y la definición de planes de acción. Se realiza la exploración de literatura acerca de mecanismos del mismo tipo y basadas en el modelo de procesos a utilizar (si es que existiesen), y mediante un estudio comparativo se presenta por qué la solución propuesta es factible.

El capitulo 3 expone el diseño y construcción de SelfVation, utilizando los lenguajes y plataformas que se adapten a los requisitos del sistema.

El capitulo 4 presenta los resultados obtenidos al experimentar la herramienta SelfVation en MPyMEs desarrolladoras de software.

El capitulo 5 presenta un resumen de las conclusiones finales obtenidas durante el desarrollo del presente trabajo y el trabajo futuro que corresponde a mejorar el uso y funcionalidad de la herramienta.

Los anexos B, C, D proporcionan información detallada sobre el diseño e implementación de los componentes de la herramienta SelfVation. Mientras que el Anexo A resume una lista de acrónimos y el Anexo E ofrece una copia del acta de publicación del artículo generado con este trabajo de tesis.

Por último se presentan las referencias bibliográficas utilizadas en el desarrollo de esta tesis.

1.7. Publicaciones generadas

A continuación se enlistan algunas de las publicaciones que se generaron durante el desarrollo del presente trabajo.

Autores:	Garcia, Ivan & **Cruz, D.**
Título:	Supporting the Management Process of Software Process Improvement Initiatives based on NMX-I-059/02-NYCE-2005
Congreso:	8th ACIS Conference on Software Engineering Research, Management and Applications (SERA 2010)
Publicación:	2010 IEEE Computer Society Conference Proceedings IEEE Catalog number: 07EX1720C ISBN: 1-4244-1080-0 ISSN: 1091-5281 Library of Congress: 2007921695
Lugar:	Montreal, Canadá.
Año:	24, 25 y 26 de Mayo 2010

2. Marco Conceptual

2.1. Antecedentes

El gobierno de México a través de los años se ha dado cuenta de la importancia del uso de la tecnología y la informática para el desarrollo del país, así lo expresa el Plan Nacional de Desarrollo (PND) 2001-2006 [SE01]: *"cada día convergen nuevas tecnologías, servicios y contenidos, que ofrecen oportunidades hasta hace poco inimaginables. Éste es el cuarto motor de la globalización, este nuevo entorno en el que convergen tecnologías de gran capacidad y cobertura de diversos servicios sirve como punto de referencia para lograr el salto cualitativo y cuantitativo como nación, también permitirá aprovechar las oportunidades del avance tecnológico y la convergencia para superar los problemas del país"*. El PND plantea como uno de sus objetivos elevar y extender la competitividad del país, mediante la estrategia de promover el uso y aprovechamiento de la tecnología y de la información. Dicho objetivo se basa en la promoción de acciones para el uso y aprovechamiento de las tecnologías como recursos estratégicos que contribuyan a la satisfacción de las necesidades de la sociedad mexicana y adoptar los mejores estándares tecnológicos [URL-3]. De la amplia gama de puntos a tratar para la aplicación de una estrategia coherente, se tiene uno fuertemente ligado a la industria software mexicana: *Impulso al desarrollo de la industria de TI.*

Es así, como la SE en coordinación con organismos empresariales y empresas del sector diseñó PROSOFT como uno de los medios para lograr el punto del PND. El objetivo de PROSOFT es impulsar a la industria de software y extender el mercado de TI en México. Para alcanzar estos objetivos, la SE en consenso con la industria y con los organismos gubernamentales relacionados con el sector, acordaron desarrollar siete estrategias [URL-4] (véase Tabla 6).

Tabla 6. Descripción de estrategias PROSOFT.

No.	Estrategia	Descripción
1	Promover las exportaciones y la atracción de inversiones.	Aprovechando las ventajas del país por su cercanía y el mismo uso horario, procurando que las empresas incursionen en nichos de alto valor agregado.
2	Educación y formación de personal competente en el desarrollo software, en cantidad y calidad convenientes.	Ofreciendo capacitación a los ingenieros y técnicos que se encuentran en el mercado y la adecuación de los planes de estudio para que sean acordes con las necesidades de la industria.
3	Contar con un marco legal	Un marco legal que fomente el uso de TI y el desarrollo de la industria con reglas como la norma de conservación de

	promotor de la industria.	mensajes de datos, factura electrónica y firma digital.
4	Desarrollar el mercado interno.	Apoyando a las empresas para que usen hardware y software en sus operaciones (Inventarios, Normas, Contabilidad) y en su relación con proveedores y clientes (Digitalización de Cadenas de Valor).
5	Fortalecer la industria local.	Mediante programas de financiamiento adecuado para sus necesidades de capital de trabajo y capacitación, la disponibilidad de capital de riesgo, el uso de las compras de gobierno para desarrollar una industria de calidad y la incubación de nuevas empresas software.
6	Alcanzar niveles internacionales en capacidad de procesos.	A efecto de que las empresas cuenten con las mejores prácticas internacionales en la producción de sus sistemas. Para ello se impulsará la normalización, la creación de una entidad local de certificación, se apoyará la investigación y desarrollo con el fondo sectorial de apoyo creado por la SE y CONACYT.
7	Promover la construcción de infraestructura básica y de telecomunicaciones.	Apoyando al desarrollo de parques de alta tecnología vinculados a centros de investigación.

De acuerdo a [URL-5], a partir de la estrategia número seis se desprenden siete puntos para llevar a cabo la estrategia:

1. **Definición de un modelo de procesos y de evaluación apropiado para la industria software mexicana**.
2. Formación de instituciones de capacitación y asesoría en mejora de procesos.
3. Apoyo financiero para la capacitación y la evaluación de la capacidad de los procesos.
4. Premio nacional de calidad en TI.
5. Estímulos fiscales al desarrollo tecnológico en las empresas.
6. Formación de un cajón de financiamiento para actividades de investigación y desarrollo.
7. Otros apoyos para actividades de investigación y desarrollo.

La selección de un modelo de procesos de referencia y un método de evaluación, accesibles para la industria local, fue el primer problema a resolver [Oktaba06]. El gobierno y la industria software definieron un criterio de selección, además de un consenso con algunas empresas sobre sus necesidades respecto a un modelo de procesos y su evaluación, se obtuvo lo que se traducía en: algo fácil de entender, práctico y barato, asimismo la SE requería que se tratara de un modelo que se pudiera adoptar como norma mexicana [Oktaba05]. A partir de esta especificación de requerimientos, la cual se aplicaría para la evaluación de la conveniencia de adopción de los más populares estándares y modelos de referencia: SW-CMM [Paulk95], CMMI, ISO/IEC 12207 [ISO02], ISO 9000:2000 e ISO/IEC 15504.

2.2. Evaluación de los Estándares y Modelos de Procesos Seleccionados

De acuerdo a los requerimientos planteados por el gobierno y la industria software, el criterio de selección para la deliberación de un modelo de procesos y un método de evaluación para la industria software mexicana, se baso en cinco criterios [Oktaba06]:

C1. Adecuado para las MPyMEs con nivel de madurez bajo.

C2. No elevado en costo para adopción y evaluación.

C3. Admisible como norma nacional.

C4. Especifico para empresas de desarrollo y mantenimiento software.

C5. Definido como un conjunto de procesos basados en prácticas reconocidas internacionalmente.

En el mismo análisis, Oktaba muestra los resultados de la evaluación (véase Tabla 7). Un *Sí* o *No*, significa que el estándar o el modelo cumple o no cumple con los criterios de selección. El signo de interrogación (?) significa que no existen pruebas suficientes para tomar una decisión [Oktaba06].

Tabla 7. Comparación de Modelos.

Modelo	C1	C2	C3	C4	C5
ISO 9000:2000	Sí	Sí	Sí	No	No
CMM/CMMI	Sí	No	No	Sí	Sí
ISO/IEC 12207	?	?	Sí	Sí	Sí
ISO/IEC 15504	?	?	Sí	Sí	No

2.2.1. Resultados de la Evaluación a Modelos de Proceso

El estándar ISO 9000:2000 no es una norma específica para las organizaciones desarrolladoras de software. Puede ser utilizada en cualquier industria y entorno: hardware, materiales procesados y servicios [De la Villa04]. Además no se define como un conjunto de procesos, ya que es un estándar que dice qué hacer pero no cómo hacerlo, describe lo que un proceso debe dirigir en lugar de cómo un proceso debe ser implementado [Paulk02], [Mutafelija03]. A razón de estas dos características el estándar ISO 9000:2000 no cumple con los criterios C4 y C5.

Por otra parte, el principal inconveniente de los modelos de proceso SW-CMM y CMMI son los costos de adopción y evaluación, además de llegar a ser demasiado detallados y difícil de entender para algunas organizaciones [De la Villa04]. Otro problema que se presenta, es que, debido a la ley mexicana, estos modelos no pueden ser aceptados como norma nacional [Oktaba06]. Debido a estos problemas, estos modelos no cumplen con los criterios C2 y C4.

El problema con ISO/IEC 12207 y ISO/IEC 15504 es que ambos han sufrido modificaciones significativas durante los últimos tiempos, por lo que su maduración ha sido lenta [Oktaba06], [De la Villa04]. En el análisis también se muestra que, debido a las pocas experiencias de adopción y evaluación, la eficacia en MPyMEs y los costos de adopción eran desconocidos [El Eman97], [Oktaba06]. Estas interrogantes fueron inconvenientes para decidir si cumplían o no con los criterios C1 y C2. Aunado a esto, ISO/IEC 15504 no cuenta con un conjunto de procesos, es solo un método de evaluación, por lo que no cumple con el criterio C5.

2.2.2. Conclusiones y Consecuencias del Análisis a Modelos de Proceso

La principal conclusión del análisis fue: *"Ninguno de los modelos propuestos cumplía con todo el criterio de selección. Los altos costos de adopción de SW-CMM y CMMI y la necesidad de una norma nacional, fueron las razones principales para el desarrollo de un nuevo modelo de proceso software"* [Oktaba06]. El nuevo modelo de proceso, Moprosoft, fue desarrollado a partir de las mejores prácticas del SW-CMM, ISO 9000:2000, PMBoK [PMI04], y otros. En él se ofrecen una nueva estructura de procesos, algunos nuevos elementos de documentación de procesos, una relación más precisa entre procesos, y un mecanismo explicito para la mejora de procesos [Oktaba07], [Oktaba06]. Moprosoft es completado por el método de Evaluación de Procesos para Industria Software (Evalprosoft), el cual fue creado basado en las recomendaciones de ISO/IEC 15504 (Parte 2). Los ensayos con Moprosoft y Evalprosoft en cuatro empresas mexicanas confirmaron lo apto del modelo para las organizaciones pequeñas con niveles bajos de madurez, confirmada por las mejoras logradas y el bajo costo del proceso de adopción [Oktaba07].

En agosto del 2005, en México se aprobó Moprosoft y Evalprosoft como parte de una norma nacional, el nombre oficial de la norma fue *NMX-I-059-NYCE-2005: Tecnología de la Información-Software – Modelos de procesos y evaluación para desarrollo y mantenimiento de software* [NYCE05]. La nueva norma consiste en cuatro fascículos (véase Tabla 8) [Flores08]:

Tabla 8. Estructura de la norma NMX-I-059-NYCE-2005.

Parte 01	NMX-I-059/01-NYCE-2005
Descripción	Definición de conceptos y productos
Parte 02	NMX-I-059/02-NYCE-2005
Descripción	Requisitos de Procesos (Moprosoft)
Parte 03	NMX-I-059/03-NYCE-2005
Descripción	Guía de implementación de procesos
Parte 04	NMX-I-059/04-NYCE-2005
Descripción	Directrices para la evaluación de procesos (Evalprosoft)

Los fascículos 2 y 4 contienen el modelo de procesos y el método de evaluación, respectivamente, las partes restantes se concentran en definiciones, conceptos y guías para entender e implantar los procesos o llevar a cabo las evaluaciones.

2.3. Modelo de Procesos para la Industria de Software (Moprosoft)

2.3.1. Estructura de Procesos

Para definir la estructura del modelo de proceso, en primera instancia se tuvo que analizar la estructura de las empresas desarrolladoras de software locales. En la mayoría de las empresas, incluso las llamadas microempresas (con menos de 10 empleados) cuentan con una alta dirección la cual es la encargada de tomar las decisiones que marquen la dirección del negocio, los mandos intermedios que son los responsables de los proyectos y la obtención de los recursos y el control de los mismos y por último un grupo operativo encargado del desarrollo de los proyectos utilizando los recursos asignados. Los miembros de esos grupos conocen las responsabilidades a través de la

asignación de roles, las líneas de autoridad son de orden vertical y las relaciones de colaboración se dan de manera horizontal [Oktaba07]. Debido a este análisis las categorías que marcan la estructura de Moprosoft son: Alta Dirección, Gestión y Operación (véase Figura 2.1).

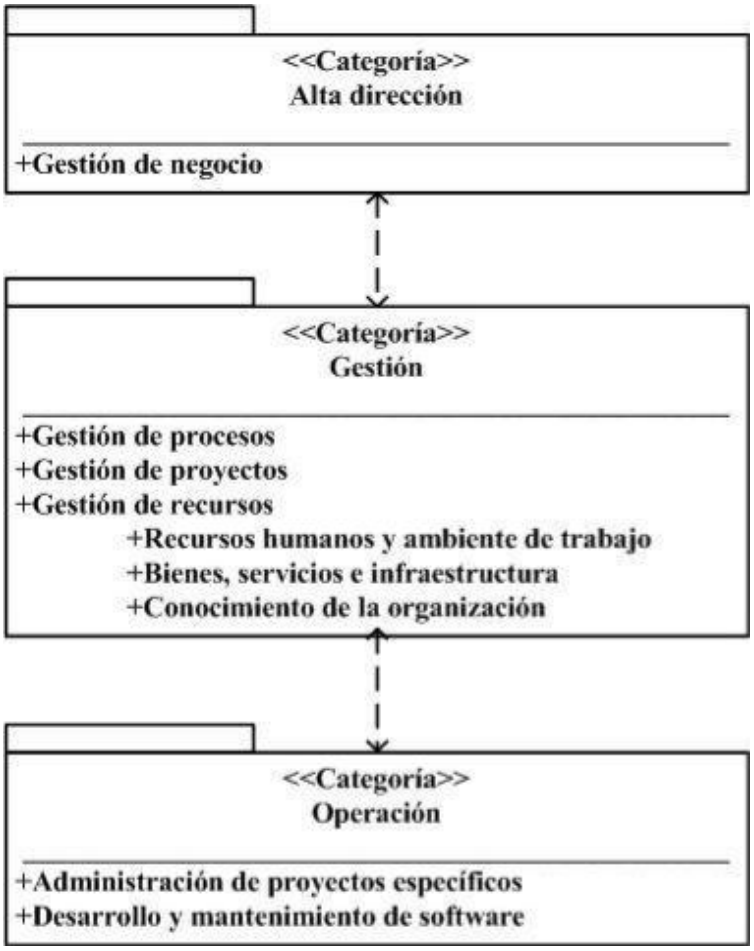


Figura 2.1. Estructura de Moprosoft

2.3.1.1. Categorías de Proceso

La categoría de Alta Dirección (DIR) contiene el proceso de Gestión de Negocio. Esta categoría aborda las prácticas de Alta Dirección relacionadas con la gestión de negocios. Proporciona los lineamientos a los procesos de la categoría de Gestión y se retroalimenta de la información generada por ellos [NYCE05].

La categoría de Gestión (GES) está integrada por los procesos Gestión de Procesos, Gestión de Proyectos y Gestión de Recursos. Éste último está constituido por los subprocesos de Recursos Humanos y Ambiente de Trabajo, Bienes, Servicios e Infraestructura y Conocimiento de la Organización. Esta categoría de procesos aborda las prácticas de gestión de procesos, proyectos y recursos en función de los lineamientos establecidos por la categoría de Alta Dirección. Su labor principal es proporcionar los elementos para el funcionamiento de los procesos de la Categoría de Operación, recibir y evaluar la información generada por éstos y comunicar los resultados a la categoría de Alta Dirección [NYCE05].

La categoría de Operación (OPE) está integrada por los procesos de Administración de Proyectos Específicos y de Desarrollo y Mantenimiento de Software. Esta categoría de procesos aborda las prácticas de los proyectos de desarrollo y mantenimiento de software. Esta categoría realiza las actividades de acuerdo a los elementos proporcionados por la categoría de Gestión y entrega a ésta la información y productos generados [NYCE05].

2.3.1.2. Procesos

DIR.1 Gestión de Negocio	El propósito de la Gestión de Negocio es establecer la razón de ser de la organización, sus objetivos y las condiciones para lograrlo y las condiciones para lograrlo, para lo cual es necesario considerar las necesidades de los clientes, así como evaluar los resultados para poder proponer cambios que permitan la mejora continua.
GES.1 Gestión de Procesos	El propósito de la Gestión de Procesos es establecer los procesos de la organización, en función de los procesos identificados en el plan estratégico. Así como definir, planificar, e implementar las actividades de mejora en los mismos.
GES.2 Gestión de Proyectos	El propósito de la Gestión de Proyectos es asegurar que los proyectos contribuyan al cumplimiento de los objetivos y estrategias de la organización.
GES.3 Gestión de Recursos	El propósito de la Gestión de Recursos es conseguir y dotar a la organización de los recursos humanos, infraestructura, ambiente de trabajo y proveedores, así como crear y mantener la base de conocimiento de la organización. La finalidad es apoyar el cumplimiento de los objetivos del plan estratégico de la organización.
GES.3.1 Recursos Humanos y Ambiente de Trabajo	El propósito de Recursos Humanos y Ambiente de Trabajo es proporcionar los recursos humanos adecuados para cumplir las responsabilidades asignadas a los roles dentro de la organización, así como la evaluación del ambiente de trabajo.
GES.3.2 Bienes Servicios e Infraestructura	El propósito de Bienes, Servicios e Infraestructura es proporcionar proveedores de bienes, servicios e infraestructura que satisfagan los requisitos de adquisición de los procesos y proyectos.
GES.3.3 Conocimiento de la Organización	El propósito del Conocimiento de la Organización es mantener disponible y administrar la base de conocimiento que contiene la información y los productos generados por la organización.
OPE.1Administración de Proyectos Específicos	El propósito de la Administración de Proyectos Específicos es establecer y llevar a cabo sistemáticamente las actividades que permitan cumplir con los objetivos de un proyecto en tiempo y costos esperados.

OPE.2 Desarrollo y Mantenimiento de Software	El propósito de Desarrollo y Mantenimiento de Software es la realización sistemática de las actividades de análisis, diseño, construcción, integración y pruebas de productos de software (nuevos o modificados) cumpliendo con los requerimientos especificados.

La relación entre los procesos está basada en el intercambio de productos y las participación de los roles en cada uno de los procesos (véase Figura 2.2). Cada uno de los productos de salida generados por algún proceso es identificado explícitamente como la entrada de uno o más procesos. Los productos internos son utilizados por el mismo proceso que lo ha generado. La relación de procesos basada en la participación de roles significa que algunos de los roles de un proceso participa en las actividades de otros procesos [Oktaba07].

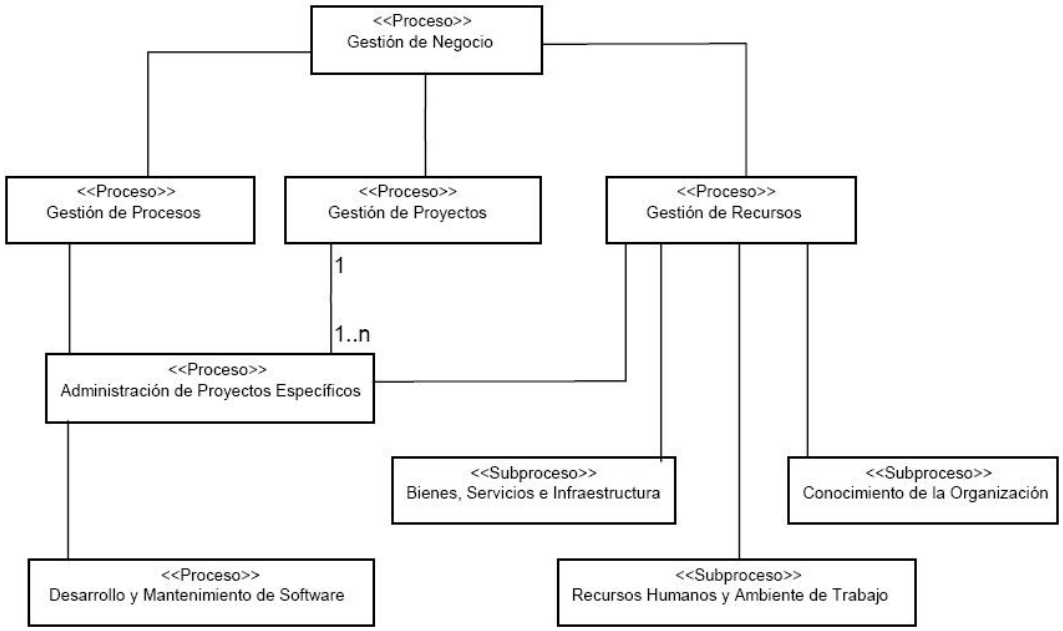


Figura 2.2. Diagrama de relación entre procesos.

2.3.2. Patrón de Procesos

El patrón de procesos es un esquema de elementos que son necesarios para la documentación de los procesos. Está constituido por tres secciones: Definición general del proceso. Prácticas y Guías de ajuste [Oktaba06].

En la Definición genera del proceso se identifica su nombre, categoría a la que pertenece, propósito, descripción general de sus actividades, objetivos, indicadores, metas cuantitativas, responsabilidad y autoridad, subprocesos en caso de tenerlos, procesos relacionados, entradas, salidas, productos internos, y referencias bibliográficas.

En las Prácticas se identifican los roles involucrados en el proceso y la capacitación requerida, se presenta un diagrama de flujo de trabajo, se describen las verificaciones y validaciones requeridas, se listan los productos que se incorporan a la base de conocimiento, se identifican los

recursos de infraestructura necesarios para apoyar las actividades, se establecen las mediciones del proceso, así como las prácticas para la capacitación, manejo de situaciones excepcionales y uso de lecciones aprendidas.

En las Guías de ajuste se sugieren modificaciones al proceso que no deben afectar los objetivos del mismo.

Las organizaciones utilizan este patrón para documentar todos los procesos Moprosoft.

2.4. Herramientas para la valoración rápida de procesos software basadas en Moprosoft

Es bien sabido que el número de empresas que se certifican en Moprosoft está creciendo considerablemente en México. Sin embargo, aquellas empresas que no son capaces de adoptar el modelo, regularmente no lo hacen por no estar preparadas para formular toda la documentación requerida para obtener la subvención del fondo patrocinador. Este motivo ha generado preocupación en distintos entornos universitarios y se ha propuesto la tarea de reducir el tiempo de adopción e implantación del modelo a través de herramientas automatizadas que ahorren tiempo y dinero a las pequeñas empresas que intentan obtener la certificación en el modelo.

Una empresa para obtener información relevante sobre la ejecución de sus procesos puede llevar a cabo evaluaciones internas de estos; las cuales consumen poco tiempo, pocos recursos y que tienen poca rigurosidad; son conocidas como *valoraciones rápidas de procesos software* [Cater-Steel04]. De acuerdo a Pino [Pino07], la valoración de procesos tiene dos objetivos:

- Generar datos de alta calidad que identifiquen los problemas de los procesos software.
- Brindar la base para tomar decisiones al interior de la empresa.

Con este tipo de valoraciones se pueden obtener datos sobre el impacto de las acciones de mejora, llevadas a cabo a través del esfuerzo SPI, en los procesos de la organización.

Considerando la importancia de la calidad de la información a obtener mediante esta actividad, es importante proporcionar a las empresas herramientas software que den soporte y ayuden a la ejecución de valoraciones rápidas de procesos. Este tipo de herramientas permiten soportar acciones repetitivas, reduciendo la carga cognitiva de las personas involucradas en la actividad de valoración y reducir cualquier carga administrativa asociada con la aplicación manual de esta actividad [Pino07].

Para la valoración de procesos software existen diferentes herramientas comerciales tales como CMM-Quest [URL-6], Appraisal Wizard [URL-7], SPiCE 1-2-1 [URL-8] e IME Toolkit [URL-9]. Sin embargo, ninguna de estas herramientas basa su método de evaluación en el modelo Moprosoft, lo que representa que las empresas mexicanas apegadas a esta norma no pueden llevar a cabo valoraciones rápidas de sus procesos, debido a la confusión que puede presentarse al evaluarse en base a otro modelo de proceso, además los resultados estarán basados en el método de evaluación propio de ese modelo y no del que se está intentando adoptar. A pesar de no existir herramientas comerciales basadas en Moprosoft, existen esfuerzos académicos por crear una herramienta con esta característica.

Las herramientas que a continuación se exponen han sido creadas a partir de esfuerzos e investigaciones académicas o esfuerzos impulsados por el gobierno. También se describen las consideraciones a tener en cuenta para cada una de ellas.

2.4.1. Kuali

Con el propósito de acelerar la adopción de Moprosoft, se pensó en una herramienta que fuera auxiliar a la implementación de este modelo y que permitiera que un número mayor de empresas emprendieran exitosamente iniciativas de mejoras de proceso [Strevel05].

Kuali es una herramienta derivada del proyecto AceleraProsoft[4] que ofrece la posibilidad de administrar proyectos basados en Moprosoft, la cual provee varias funcionalidades:

- **Navegación del proceso:** Permite la consulta integrada en la herramienta de todos los elementos que contempla Moprosoft para cada uno de los seis procesos: descripción de actividades, roles, métricas. Asimismo, la herramienta permite la navegación por medio de hipervínculos lo cual facilita la consulta de la información.
- **Base de Conocimiento:** Uno de los elementos más importantes contemplado por Moprosoft es el de la Base de Conocimiento, el cual es un repositorio donde se depositan todos los productos de los procesos. Kuali ofrece la funcionalidad de proveer este repositorio en el cual se puede almacenar de manera segura todos los productos generados de acuerdo a cada uno de los procesos. Se ofrece un repositorio jerárquico en el cual se soporta un control de versiones, visor de documentos, soporte a plantillas y definición de permisos de acuerdo a roles.
- **Elementos de Trabajo:** Este componente de la herramienta permite la captura y control de los principales elementos de trabajo relacionados con el proceso de construcción de software entre los cuales se encuentra: defectos, requerimientos, tareas y riesgos. La herramienta soporta ciclos de trabajo sobre los diferentes estados que puede tomar un elemento de trabajo (Nuevo, Activo, Cerrado).

2.4.1.1. Ventajas

- Desarrollada en base al modelo Moprosoft.
- Acelera adopción de Moprosoft.
- Auxilia a la implementación de Moprosoft en la empresa software, ayudando a desempeñar las tareas más demandantes del modelo, con el fin de normalizar sus procesos y ser más competitivos.
- Describe los procesos de Moprosoft, lo que facilita al personal consultar los procedimientos que debe de llevar a cabo para dar solución a problemas específicos.
- Pone énfasis en que los procesos sean documentados y realizados tal y como fueron planteados.
- Promueve el SPI en la empresa software.

2.4.1.2. Desventajas

- No es una herramienta de valoración de procesos software.
- Se trata de una versión Beta[5], la cual está en constante modificación y actualización (véase Figura 2.3) [Valenzuela07].

[4] AceleraProsoft es una iniciativa desarrollada por la SE, en conjunto con Microsoft, Visionaria y Amiti; la cual consiste en ayudar a los participantes a ejecutar un plan de negocios para incrementar sus ventas a mediado y corto plazo, además de fortalecer sus procesos de planeación, operación, gestión de ventas y mercadotecnia.

- Está orientada a la estética del patrón de procesos y no cubre por completo el patrón de procesos definido por Moprosoft [Valenzuela07].
- Después de la liberación de la versión Beta de la herramienta, no ha habido muestras de actualizaciones o de una versión final, incluso los sitios de Internet donde se podía disponer de dicha herramienta son inválidos.
- Nunca fue probada en entornos reales de trabajo, al menos no existe evidencia de esto.

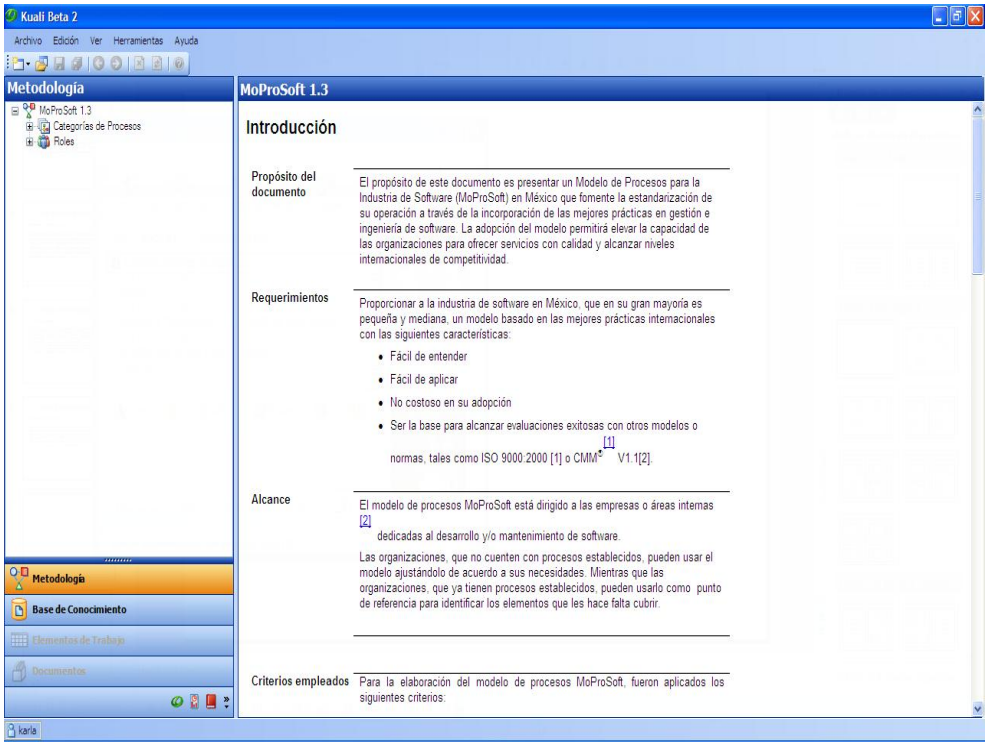


Figura 2.3. Pantalla de navegación de procesos de Kuali.

2.4.2. Manejador de Documentos Moprosoft (MDM)

El Manejador de Documentos de Moprosoft (MDM) [Caballero05] es un trabajo de tesis desarrollado en la Universidad de las Américas Puebla (UDLAP) en el año 2005. El objetivo general de la aplicación es ayudar a las empresas mexicanas desarrolladoras de software a documentar los procesos de Gestión de Negocios, Gestión de Procesos y Administración de Proyectos Específicos que forman parte de Moprosoft. Según [Caballero05] esta herramienta está disponible vía Internet y permite la creación de plantillas, modificación, administración y almacenamiento de los documentos sobre los procesos de algunos proyectos específicos de una etapa de Moprosoft.

En [Caballero05] el autor menciona las dos características más importantes de MDM:

[5] El término *versión Beta* se utiliza en la industria software para diferenciar aquellas versiones de un programa que no están completamente depuradas y, por lo tanto, no pueden ser comercializadas. Podría decirse que las versiones beta son versiones de prueba de un software determinado, desarrolladas antes de la versión definitiva.

1. La herramienta maneja los documentos de Moprosoft de tal forma que si este modelo de procesos sufre alguna alteración o se liberan nuevas versiones, solo se tenga que aumentar un componente en el sistema.
2. La herramienta manejadora de documentos permite a sus usuarios crear documentos, modificarlos, compartirlos, crear reportes, y administrarlos. Esto incluye controlar accesos, modificaciones y autorizaciones de impresión.

La herramienta ayuda a las empresas software mexicanas a implementar la norma mexicana de modelo de proceso software como su modelo de mejora de procesos, esto en base a documentar continuamente cada una de las actividades que la norma requiere y que se encuentran explicitas en el sistema MDM. De acuerdo al análisis realizado, esta herramienta fue diseñada como una opción para facilitar la adopción de Moprosoft por parte de las empresas mexicanas, de una manera sencilla y fácil de aplicar.

2.4.2.1. Ventajas

- Desarrollada en base a los procesos de Moprosoft.
- Apoya la adopción de Moprosoft, de manera sencilla y fácil de aplicar (sin comprobar).
- Sirve como auxiliar para la documentación continua de los procesos de la empresa.
- Promueve el SPI dentro de la empresa.
- Disponible vía Internet (sin comprobar).

2.4.2.2. Desventajas

- No es una herramienta de valoración de procesos.
- Al igual que Kuali, está orientada a la estética del patrón de procesos y no cubre por completo el patrón de procesos definido por Moprosoft [Valenzuela07].
- Solo incorpora tres procesos limitando a la empresa en el seguimiento, interrelación e implantación de los procesos restantes [Valenzuela07].
- No existe evidencia de su utilización en entornos reales.

2.4.3. Herramienta Integral para Moprosoft (HIM) y Herramienta de Guía y Supervisión para el Uso Automatizado del Modelo de Procesos Moprosoft (AsistenteHIM)

La Herramienta Integral para Moprosoft (HIM), es un proyecto a cargo de la Maestría en Ciencia e Ingeniería de la Computación de la Universidad Nacional Autónoma de México (UNAM) [Zurita05]. HIM permite adaptar y dar seguimiento al modelo Moprosoft a través de una aplicación Web al facilitar un entorno de trabajo adaptado al modelo, esto es, muestra los procesos, actividades y productos a realizar de un rol (asignado a los usuarios) especificados en el modelo [Zurita05].

La herramienta es un sistema Web que facilita el entorno de trabajo para Moprosoft y una de sus características principales es que la base de conocimiento fue desarrollada utilizando el marco de trabajo para la descripción de recursos (RDF, *Resource Description Framework*), que permite generar la información que ofrece Moprosoft por medio de archivos RDF (archivos de texto con sintaxis XML). La herramienta HIM utiliza parte de esa información para su funcionamiento, sin embargo se podían tener más funcionalidades que aún no habían sido implementadas. Por otro lado, se tenían requerimientos pendientes como el de brindar ayuda al usuario en el manejo del sistema [Cárdenas06].

De acuerdo a lo anterior se necesitaba una adecuación a la HIM, que utilizara la base de conocimiento para brindar más funcionalidades y que fungiera como guía del seguimiento de los procesos de Moprosoft, así como ayudar al usuario en el manejo de la herramienta y en la realización de sus actividades. El sistema se llama herramienta de guía y supervisión para el uso automatizado del modelo de procesos Moprosoft (AsistenteHIM), el cual representa un trabajo de tesis de maestría en la UNAM [Cárdenas06a].

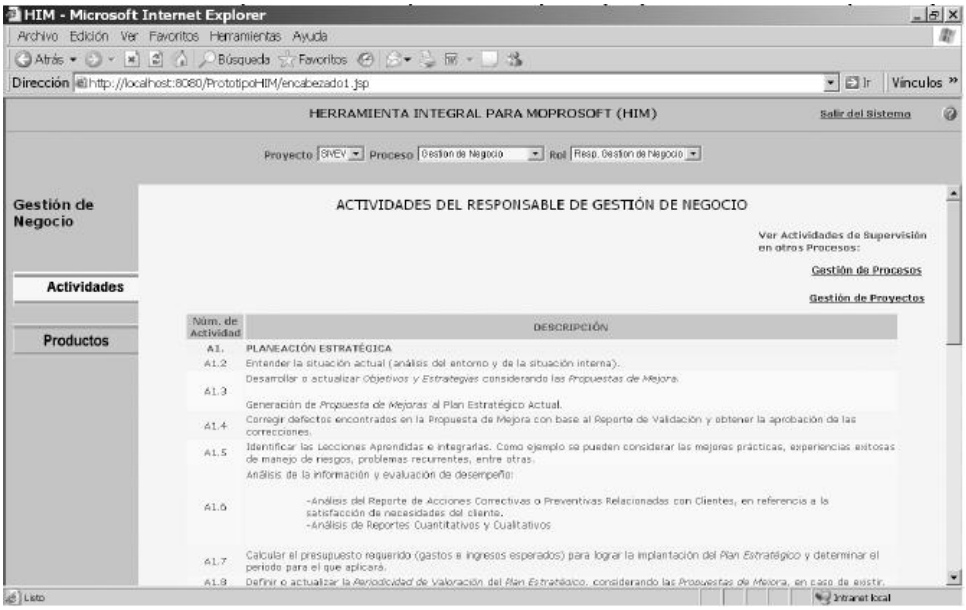


Figura 2.4. AsistenteHIM muestra las actividades del responsable de Gestión de Negocios, así como los vínculos de los procesos en los que también participa.

Los objetivos del AsistenteHIM se resumen como la provisión de un asistente que ayuda a los usuarios con el trabajo realizado con la herramienta, brindándoles información de sus responsabilidades y posibilidades (véase Figura 2.4), tareas a realizar y responsables de ellas, sugerencias sobre la manera de llevarlas a cabo, recordatorios de tareas pendientes y coordinación del trabajo con otros usuarios; todo de acuerdo a Moprosoft. Para cumplir con lo especificado, la herramienta implementa los enfoques de agentes de software y Razonamiento Basado en Casos, áreas de Inteligencia Artificial [Vargas07].

2.4.3.1. Ventajas

- Promueve el SPI dentro de la organización.
- Proporciona seguimiento y supervisión para el uso del modelo de procesos Moprosoft.
- Brinda información de las responsabilidades a cada uno de los involucrados en la implantación del modelo.
- Promueve la automatización de la implantación del modelo, auxiliando a los usuarios a realizar sus tareas, sugerencias de cómo llevarlas a cabo, recordatorio de tareas pendientes y coordinación de trabajo con otros usuarios.
- Promueve el trabajo en conjunto de dos áreas: Ingeniería de Software e Inteligencia Artificial, dejando abiertas muchas opciones para trabajos futuros [Cárdenas06a].

2.4.3.2. Desventajas

- Se centra en la ejecución de procesos y coordinación de actividades [Valenzuela07], pero no realiza una valoración de procesos.
- Como lo expresa el autor en [Cárdenas06a]: la herramienta tiene algunas limitaciones, por lo que no alcanza a cubrir la totalidad de los objetivos propuestos.
- Los agentes y el sistema basado en casos, quedaron inconclusos y como trabajo futuro se propuso implementar la totalidad de estas características [Valenzuela07].
- No existe evidencia de su uso en entornos reales.

2.4.4. Instrumento de Auto-evaluación para Diagnosticar el Estatus de las Organizaciones en México con Respecto a Moprosoft: Proceso de Gestión de Procesos de la Categoría de Gestión

Esta herramienta fue desarrollada como parte de un proyecto de investigación por parte de la Universidad Politécnica de Aguascalientes (UPA) y la Universidad Autónoma de Aguascalientes (UAA) [Reyes08]. El trabajo propone una metodología de desarrollo de un instrumento de medición basado en Moprosoft, que ayude a las organizaciones en México con áreas generadoras de software, a realizar una autoevaluación de su proceso de Gestión de Procesos para encontrar vías de mejora en sus procesos evaluados [Reyes09]. Se evalúa solo la categoría de Gestión, y en particular solo el proceso Gestión de Procesos por tratarse del encargado de proporcionar la guía para el cumplimiento de todos los demás procesos de la organización, proporcionando el modelo de procesos necesario para alcanzar los objetivos estratégicos, así como la manera que cómo estos deben ser medidos [Reyes08] (véase Figura 2.5). La información que se recolecta es de tipo cualitativo y nominal, por lo que la herramienta utiliza una escala Likert[6]. El resultado lo proporciona el promedio de los reactivos que aseguren obtener el nivel de capacidad en cuestión, de acuerdo a los valores asignados a la escala de Likert.

Figura 2.5. Pantalla principal del instrumento de auto-evaluación automatizado para el diagnóstico de las organizaciones en México con respecto a Moprosoft.

[6] La escala de Likert mide actitudes o predisposiciones individuales en contextos sociales particulares. Se le conoce como escala sumada, debido a que la puntuación de cada unidad de análisis se obtiene mediante la sumatoria de las respuestas obtenidas en cada ítem. La escala se construye en función de una serie de ítems que reflejan una actitud positiva o negativa acerca de un estímulo o referente. Cada ítem está estructurado con cinco alternativas de respuesta.

2.4.4.1. Ventajas

- Es un instrumento dedicado a la valoración de los procesos software.
- Al ser validada por un panel de expertos, el instrumento cuenta con un alto grado de confiabilidad, validez y practicidad.
- Promueve el SPI dentro de la empresa.
- Auxilia a las empresas que desean adoptar el modelo Moprosoft, ya que mediante una evaluación continua existen más posibilidades de mejoras a sus procesos.

2.4.4.2. Desventajas

- Solo se evalúa el proceso Gestión de Procesos del modelo Moprosoft.
- Se presentan resultados satisfactorios ante la evaluación de un panel de expertos, pero no se muestran experimentaciones y resultados con empresas software.
- No proponen un guía de mejora para las empresas, mostrando qué actividades realizar para implantar un mejor mecanismo de SPI dentro de la empresa.
- No proporciona a la empresa una estimación de cual es nivel de madurez de sus procesos de acuerdo a los niveles propuestos por Moprosoft.
- Solo presenta resultados, no se asegura de marcar el rumbo de la empresa para mejorar las actividades débiles que fueron halladas durante el proceso de evaluación.

2.5. Análisis empírico sobre las herramientas para la valoración rápida de procesos software basadas en Moprosoft

Como se ha podido observar, las herramientas presentadas en el apartado anterior intentan agilizar la adopción del modelo pero solo una de ellas se enfoca en evaluar la situación actual de la empresa en relación al mismo.

Consideramos que una herramienta de evaluación y generación de planes apoyaría en gran medida la implantación de iniciativas de SPI en las pequeñas empresas mexicanas y representaría una ventaja para aquellas empresas que deseen implantar oficialmente el modelo, pero que requieran de un apoyo automatizado antes de dedicar tiempo y esfuerzo sin realizar pruebas reales sobre su proceso actual. Con el objetivo de proponer una herramienta alternativa, a continuación se presenta un benchmarking[7] sobre las herramientas analizadas. El símbolo ✘ denota que la herramienta no cumple con el criterio propuesto, tanto que el símbolo ✔ significa que la herramienta cumple con el criterio, y por último el símbolo ? representa que de acuerdo al análisis de la literatura no se comprueba si cumple o no con el criterio.

[7] El benchmarking es un anglicismo que, en las ciencias de la administración de empresas, puede definirse como un proceso sistemático y continuo para evaluar comparativamente los productos, servicios y procesos de trabajo en organizaciones. Consiste en tomar "comparadores" o *benchmarks* a aquellos productos, servicios y procesos de trabajo que pertenezcan a organizaciones que evidencien las mejores prácticas sobre el área de interés, con el propósito de transferir el conocimiento de las mejores prácticas y su aplicación; es "copiar al mejor".

Tabla 9. Benchmarking sobre las herramientas analizadas.

Criterios	Herramientas			
	Kuali	MDM	AsistenteHIM	Autoevaluación Moprosoft
Realiza evaluaciones cuantitativas del proceso actual	✗	✗	✗	✓
Genera Planes concretos de mejora	✗	✗	✗	✗
Incluye mecanismos de diagramación para evaluar procesos	✗	✗	✗	✗
Establece una “fotografía” del nivel actual, de acuerdo al modelo	✗	✗	✗	✓
Proporciona tutoría sobre la realización de las prácticas	✓	✓	✓	✓
Incluye todas las áreas del modelo	✗	✗	✗	✗
Facilita su uso por mejora visual	✗	?	?	?
Funciona bajo el enfoque Web	✗	✓	✓	✓
Apoya a iniciativas SPI	✗	✗	✗	✓

En base a esto, la solución que se plantea en el siguiente capítulo deberá cumplir las deficiencias que han sido identificadas en la Tabla 9, pero además deberá incluir las ventajas que ofrecen las herramientas ya existentes.

3. Desarrollo de una Herramienta de Valoración Rápida bajo el enfoque RIA (Rich Internet Applications)

3.1. Rich Internet Applications (RIA)

En la actualidad, las aplicaciones Web no solo deben de ofrecer variedad de funciones, sino que además deben de ser lo más óptimas posibles y sus interfaces gráficas deben ser mucho más impactantes y cómodas para los usuarios [Farré05]. En los últimos años, la demanda de aplicaciones basadas en Internet continúa creciendo y, en general, es bastante diferente a la demanda que se presentaba en la década de los noventa. Los usuarios finales y las empresas ahora exigen más de sus inversiones en tecnología Internet. La capacidad para distribuir valores reales a los usuarios está obligando a muchas empresas a buscar modelos de aplicaciones de Internet más ricas; modelos que combinan la potencia de las computadoras de sobremesa tradicionales, ricas en dispositivos, con la utilización y la naturaleza rica en contenido de las aplicaciones Web [Tapper09], [Davis08].

En Marzo de 2002 Macromedia acuño el término *Aplicaciones Ricas de Internet* (RIA, *Rich Internet Applications*) las cuales son un nuevo tipo de aplicación Web cuyo objetivo es el incrementar y mejorar las opciones y capacidades de las aplicaciones Web tradicionales [Tapper09], [Farré05]. De acuerdo a [URL-10], las RIA ofrecen una experiencia sofisticada y atractiva que mejora la satisfacción del usuario y aumenta su productividad. Gracias al amplio alcance de Internet, las RIA pueden implementarse en navegadores, escritorios y dispositivos.

Las limitaciones en la capa de presentación de los actuales navegadores Web y del lenguaje HTML han motivado a los desarrolladores a utilizar las aplicaciones tipo RIA que permiten, entre otras cosas, mejorar la experiencia entre el usuario y la aplicación, la ejecución de contenido multimedia y la carga de aplicaciones online/offline, dependiendo de la tecnología que se utilice [Farré05]. La idea de las RIA es poder realizar programas con las funcionalidades de una aplicación de escritorio común, con la gran ventaja de que no es necesario que el usuario tenga la aplicación instalada en su PC, sino que sean accesibles desde cualquier navegador Web, con lo cual se obtiene una aplicación multiplataforma que funciona en cualquier computadora con un navegador Web y conexión a Internet (véase Figura 3.1).

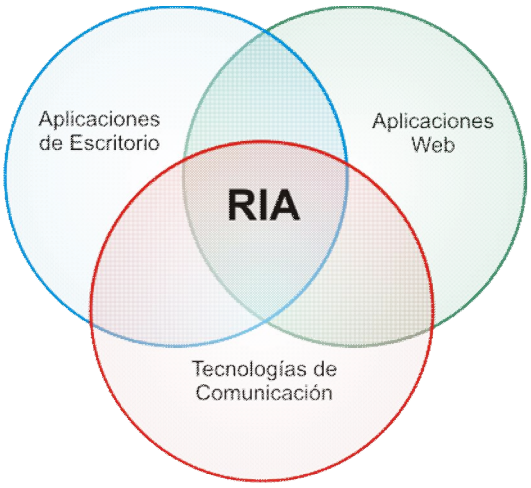


Figura 3.1. Idea principal de un enfoque RIA.

Las RIA son aplicaciones Web que transfieren la mayor parte de la carga de procesamiento de la interfaz de usuario para el cliente Web, mientras que la parte predominante de los datos (desde el control y el mantenimiento a los datos) permanece en el servidor de aplicaciones [Martinez-Ruiz06]. La Figura 3.2 muestra una arquitectura RIA estándar.

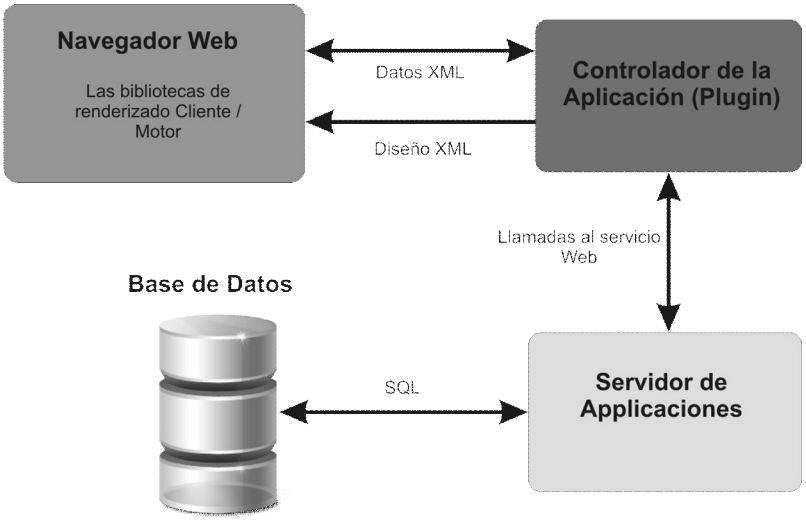


Figura 3.2. Arquitectura típica de RIA.

Una de las principales ideas de este tipo de aplicaciones es el mejorar el método en que se envía y recibe el flujo de información entre el cliente y el servidor, las aplicaciones Web tradicionales cargan todo el procesamiento de datos en el servidor, recargando toda la página con cualquier acción que haga el usuario, aún si el cambio realizado es mínimo; mientras que bajo un enfoque RIA sólo es necesario cargar la interfaz solo una vez, y para cada acción del usuario se realiza una petición especifica de la información solicitada al servidor sin la necesidad de recargar toda la página. De esta manera, se consigue reducir el tiempo de espera del usuario, el ancho de banda consumido y la carga de trabajo para el servidor [Tapper09], [Pérez08], [Davis08], [Brown08]. En la actualidad existen muchas opciones tecnológicas para construir una aplicación tipo RIA. Las opciones más populares son las iniciativas basadas en HTML como AJAX

(*Asynchronous Javascript and XML*, *Javascript* y XML Asíncronos) [URL-11], y las opciones basadas en *plugin* como Adobe Flash [URL-12], Adobe Flex [URL-13] y otros que se ejecutan en Flash Player. También existen nuevas tecnologías de Microsoft, como Windows Presentation Foundation (WPF) [URL-14] y Silverlight [URL-15].

3.1.1. Ventajas

Las RIA ofrecen a las empresas un modo económico y de probada eficiencia para proporcionar modernas aplicaciones con las ventajas de los negocios reales. Entre algunas de sus ventajas se encuentran las siguientes:

- Una RIA bien diseñada reduce el nivel de frustración del usuario porque ya no es necesario navegar por varias páginas para encontrar lo que necesita o esperar a que una página nueva se cargue antes de continuar con la actividad [Tapper09].
- El tiempo que el usuario necesita para aprender cómo utilizar la aplicación puede reducirse mucho, con lo que el usuario está más capacitado [Tapper09].
- Se aleja de la arquitectura basada en la página lo que reduce la carga de los servidores Web y reduce el tráfico total de la red, lo que deriva en una navegación Web mucho más ágil [Tapper09, Pérez08].
- En lugar de transmitir páginas enteras una y otra vez, la aplicación completa se descarga una vez y la única comunicación desde y hacia el servidor es la de los datos que aparecen en la página [Tapper09, Davis08, Brown08].
- Se pueden desarrollar herramientas más potentes. Las tecnologías que implementan el concepto RIA suelen facilitar la generación de efectos que hacen la navegación mucho más cómoda [Pérez08, Farré05].
- Mejora de la experiencia visual, gracias a la aportación de nuevos componentes (mucho más avanzados) [Goralski08].
- Permite crear aplicaciones más atractivas mediante la utilización de audio, video y gráficos [Goralski08].
- Permiten desarrollar aplicaciones interactivas que no se pueden obtener utilizando solo HTML [Goralski08, Brown08].

3.2. Metodología de desarrollo

Partiendo del objetivo general de este trabajo, el cual establece el diseño y la construcción de una herramienta de tipo SPA, basada en el modelo Moprosoft. Por lo tanto la metodología de desarrollo (véase Figura 3.3) para la construcción de esta herramienta se divide en las siguientes etapas:

1. Establecer la categoría y por consecuencia los procesos del modelo Moprosoft que la herramienta evaluará.
2. Realizar un estudio acerca de las características de los procesos delimitados.
3. Definir un contexto de trabajo general de la herramienta.
4. Definir los componentes principales que se implementarán en la herramienta.

   4.1. Definición de un mecanismo de evaluación.

   - Definición de un método de evaluación basado en cuestionarios.

- Definición de un método de evaluación basado en el modelado del proceso de desarrollo actual de la empresa.

4.2. Definición de la fase de resultados

4.3. Definición de un mecanismo para la generación de planes de acción.

5. Capturar requisitos formales para la implementación de la herramienta.
6. Diseño conceptual de la herramienta en base a la metodología UML-based Web Engineering (UWE).
7. Diseño de interfaz de usuario RIA utilizando el método RUX.
8. Implementar los módulos diseñados, bajo el lenguaje RIA: Adobe Flex.

   8.1. Corregir los defectos encontrados hasta lograr pruebas unitarias sin defectos.
9. Integrar los módulos construidos para la obtención de la herramienta completa.

   9.1. Corregir los defectos encontrados, hasta lograr pruebas del sistema sin defectos.
10. Publicar la herramienta en un sitio Web para su utilización y acceso remoto.

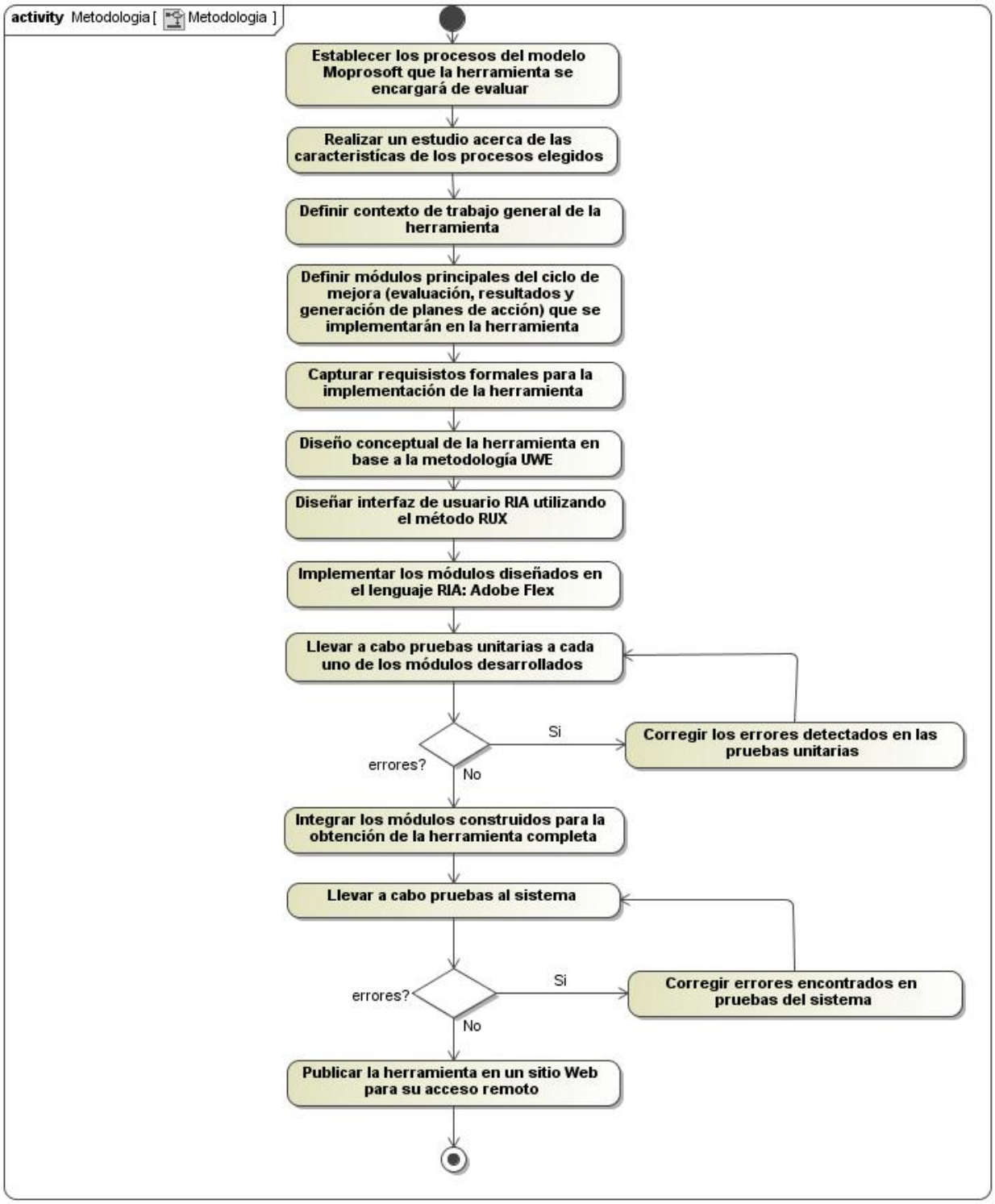


Figura 3.3. Metodología de desarrollo para la construcción de SelfVation.

3.3. Desarrollo de la Herramienta de Evaluación

El análisis empírico que se describió al final del capítulo anterior, en el cual se analizaron las ventajas y desventajas de las herramientas desarrolladas para apoyo a la adopción del modelo Moprosoft, nos proporciona la información suficiente para sustentar y exponer la propuesta de solución para el desarrollo de una herramienta de evaluación del proceso software en base al modelo Moprosoft. Este análisis refleja que solo existe un trabajo con el enfoque de esta tesis, presentado en [Reyes08], el cual a pesar de fomentar la mejora al proceso software dentro de la empresa, no cumple con criterios importantes que la hagan una herramienta que promueva SPI completamente dentro en la organización.

En este sentido, este trabajo de tesis propone una herramienta de evaluación práctica, que permita crear planes de acción en base a las debilidades y fortalezas de las MPyMEs desarrolladoras de software en base al análisis de las evaluaciones de sus procesos actuales. La Figura 3.4 muestra que la herramienta está compuesta por tres módulos básicos: Evaluación, Resultados y Generación de Planes de Acción. La solución está dirigida a dos tipos de roles dentro de la empresa: la Alta Dirección y los mandos Operativos. Los primeros porque son los encargados de establecer la evaluación, elegir los procesos a evaluar y asignar personal a cada evaluación, así como también de implantar los planes de acción generados por la herramienta, además de tener acceso a los resultados de las evaluaciones. Los mandos Operativos serán los encargados de involucrarse en la evaluación, ya que al encontrase más cercanos al proceso de desarrollo en la empresa por consecuencia conocen los procesos y productos actuales dentro de la organización.

Una de las innovaciones de este trabajo es el proponer una nueva forma de evaluación, la cual consta de dos fases: la primera, y ya utilizada con anterioridad, que consiste en el desarrollo de un cuestionario que sea utilizado como un instrumento para la obtención de datos basado en las prácticas propuestas por el modelo de referencia, la segunda fase basada en el proceso de desarrollo actual de la MPyMEs desarrolladoras de software. Esta fase, que será explicada más profundamente en los siguientes apartados de este capítulo, a grandes rasgos consiste en modelar el proceso de desarrollo de la empresa por medio de la herramienta, posteriormente el mecanismo de evaluación realizará un mapeo del proceso de la empresa con el proceso ideal que plantea el modelo Moprosoft, y como resultado se obtendrá un diagrama de proceso nuevo, que estará compuesto por las actividades del proceso actual que cumplan con el modelo y será enriquecido con actividades que el modelo propone y que no figuran en el proceso original de la empresa, así como también mostrará los posibles errores que se llevan a cabo dentro del proceso inicial de la empresa.

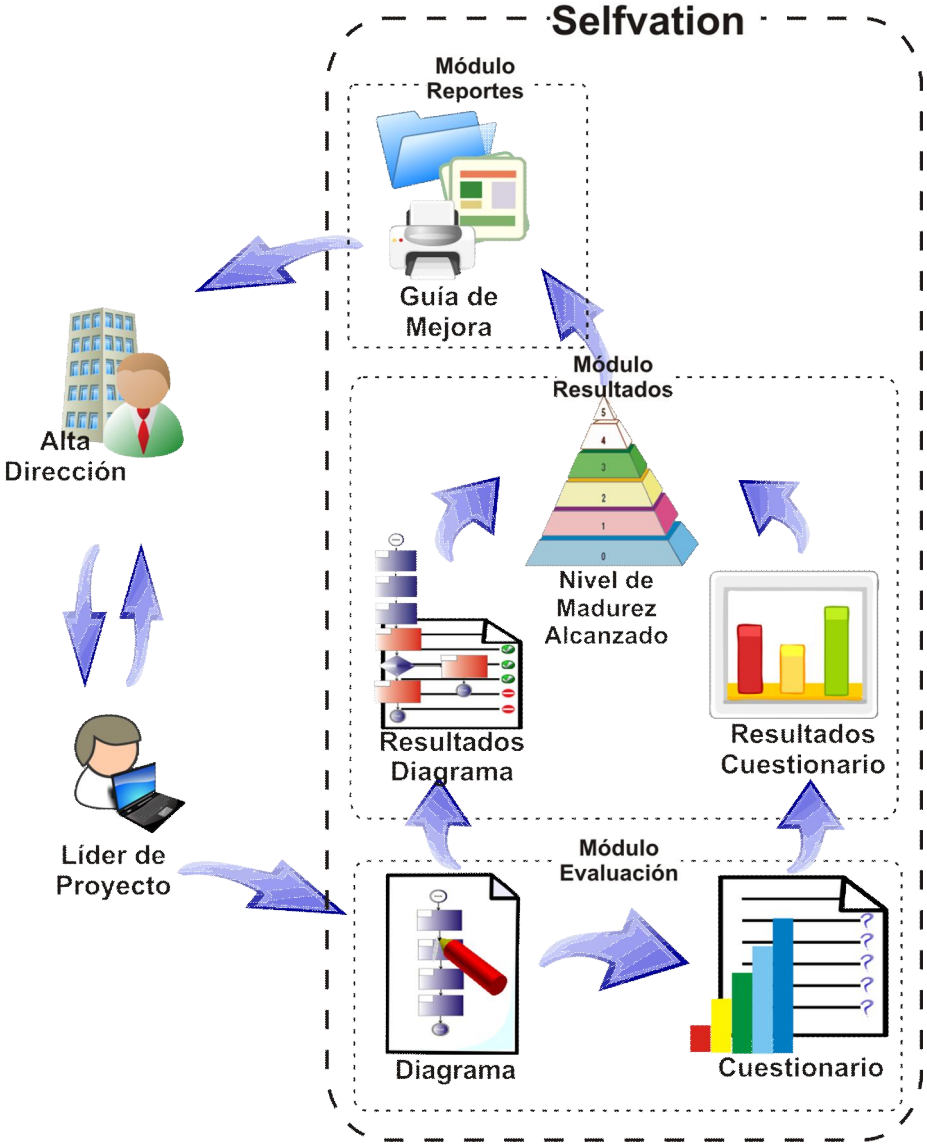


Figura 3.4. Solución propuesta.

3.3.1. Definición de un mecanismo de evaluación

Tal y como se ha mencionado, el mecanismo de evaluación a implementar consta de dos partes, en primera instancia se encuentra el desarrollo de un cuestionario basado en el modelo Moprosoft, el cual revele la situación actual de la empresa con relación a las prácticas propuestas por el modelo. Este tipo de evaluaciones han sido utilizadas con anterioridad por otras herramientas de valoración de procesos, tales como CMM-Quest y Appraisal Wizard, aunque desarrolladas en base a un distinto modelo de procesos.

La segunda parte del mecanismo de evaluación incorpora un elemento no explorado hasta ahora, ya que no existen herramientas dedicadas a la evaluación de procesos software que modelen el proceso actual de la empresa y lo mapeen con el proceso ideal que propone el modelo de proceso; y de esta forma obtener un diagrama de proceso enriquecido, formado con las actividades actuales de la empresa y complementado con las actividades propuestas por el modelo en cuestión. Un punto

importante al intentar relacionar estos dos tipos de evaluaciones consiste en la obtención de valoraciones más veraces, ya que al contar con dos tipos de mecanismos de evaluación, con estructura y presentación diferente pero basadas en los mismos principios, se pueden detectar con mayor facilidad inconsistencias en las respuestas de los usuarios sobre los procesos actuales en sus empresas, debido a que si en un determinado proceso en la evaluación por modelado se carece de una cierta actividad y en la evaluación por cuestionario refleja que las actividades de ese tipo se encuentran implantadas dentro de la empresa, se puede inferir que el evaluado está respondiendo al azar el cuestionario o mintiendo en sus respuestas.

3.3.1.1. Definición y Diseño de un Cuestionario para evaluar la categoría Operación del modelo Moprosoft

Se indicó en el apartado 1.2 que el cuestionario propuesto sería utilizado como un instrumento de obtención de datos en la evaluación de los procesos correspondientes a la categoría de Operación del modelo Moprosoft. Este mecanismo de evaluación fue elegido porque brinda una rápida solución en una investigación metodológica y debido a que la investigación puede determinar las preguntas a realizar y el rango de respuestas que pueden ser concedidas. Esto lo hace más preciso y fácil de analizar desde el punto de vista de la investigación [Garcia07]. Además que la aplicación de cuestionarios consume menos tiempo, esfuerzo y recursos financieros que otros métodos de obtención de datos como las entrevistas y revisión de documentos [Brodman99]. La categoría Operación del modelo Moprosoft fue seleccionada ya que es la que se encuentra más relacionada con el proceso de desarrollo dentro de MPyMEs, basándose en las fases genéricas de un ciclo de desarrollo.

Es verdad que existen un gran número de instrumentos para la obtención de datos y que pueden ser utilizados en las evaluaciones: cuestionarios, estudios, entrevistas, y revisión de documentación, sin embargo cada uno de estos tiene sus propias ventajas y desventajas. Una de las técnicas más utilizadas es el *cuestionario*. Esto se debe principalmente a que puede ser aplicado a mucha gente, es efectivo en costo, no invasivo, proporciona datos cuantitativos y los resultados pueden ser analizados de inmediato [Gillham00].

Los cuestionarios pueden clasificarse de preguntas abiertas y cerradas. Una pregunta abierta proporciona más información que una pregunta cerrada. Sin embargo, de acuerdo a [Yamanishi02] la complejidad de analizar los datos proporcionados por las preguntas abiertas es mayor que en las preguntas cerradas. Por otro lado, una pregunta cerrada proporciona menos información pero sus resultados pueden ser analizados más fácilmente y se obtienen más rápido que con las preguntas abiertas [Garcia07]. Por lo anterior, la propuesta de solución utiliza un cuestionario cerrado, ya que al tratarse del desarrollo de una herramienta automatizada, las características de este tipo de cuestionarios se acoplan a las particularidades que se pretenden para la solución.

El cuestionario utiliza una versión redefinida de la escala Likert [Russell92] y establece un nivel de desempeño similar al cuestionario de dos fases propuesto en [Garcia07]. La Tabla 10 muestra como cada respuesta de la escala de Likert corresponde con un nivel establecido de funcionamiento para determinar el porcentaje en el cual se realiza cada práctica

Tabla 10. Clasificación del Nivel de Desempeño.

Posibles Respuestas	Nivel de Desempeño	Descripción
Siempre	4	La actividad es documentada y establecida en la organización (entre el 86% y 100% de las veces).
Usualmente	3	La actividad es establecida en la organización pero rara vez documentada (entre el 60% y 85% de las veces).
A veces	2	La actividad es débilmente establecida en la organización (entre el 26% y 59% de las veces).
Rara vez	1	La actividad es rara vez realizada en la organización (entre el 1% y 25% de las veces).
Nunca	0	Esta actividad no es realizada en la organización.

Dando un peso específico a cada respuesta podemos analizar fácilmente los resultados de la evaluación e identificar qué prácticas son comunes dentro de la organización y cuáles no se realizan en absoluto. El cuestionario obtenido está dividido para las áreas de proceso Administración de Proyectos Específicos y Desarrollo y Mantenimiento de Software de la categoría Operación del modelo Moprosoft y las preguntas se presentan en base a las fases genéricas de un ciclo de vida cualquiera para el desarrollo de software. En la Tabla 11 se presenta un ejemplo del cuestionario obtenido para la fase de requisitos. Para consultar el cuestionario completo obtenido en este apartado, examinar el Anexo B de este documento.

Tabla 11. Ejemplo de cuestionario obtenido para la fase de requisitos.

1. ¿Distribuyen las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Obtienen requisitos y los documentan en una Especificación de Requisitos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Verifican y Validan la Especificación de Requisitos, generando Reporte Verificación y Reporte de Validación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Elaboran un Plan de Pruebas del Sistema?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Verifican el Plan de Pruebas del Sistema, generando un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Incorporan la Especificación de Requisitos y Plan de Pruebas del Sistema como líneas base a la Configuración de Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Elaboran un Reporte de Actividades correspondiente a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

Existen muchas herramientas de valoración, tanto comerciales como trabajos de investigación, que han utilizado los cuestionarios como mecanismos de evaluación. Sin embargo, la mayor parte de estas herramientas presentan los cuestionarios de una forma tediosa y cansada para el usuario, debido a que presentan una gran cantidad de preguntas en una sola pantalla lo que puede traducirse en desidia del usuario para contestar el cuestionario ya que independientemente del número de preguntas el observar un gran número de ellas en la pantalla puede dar inicio a este problema (véase Figura 3.5).

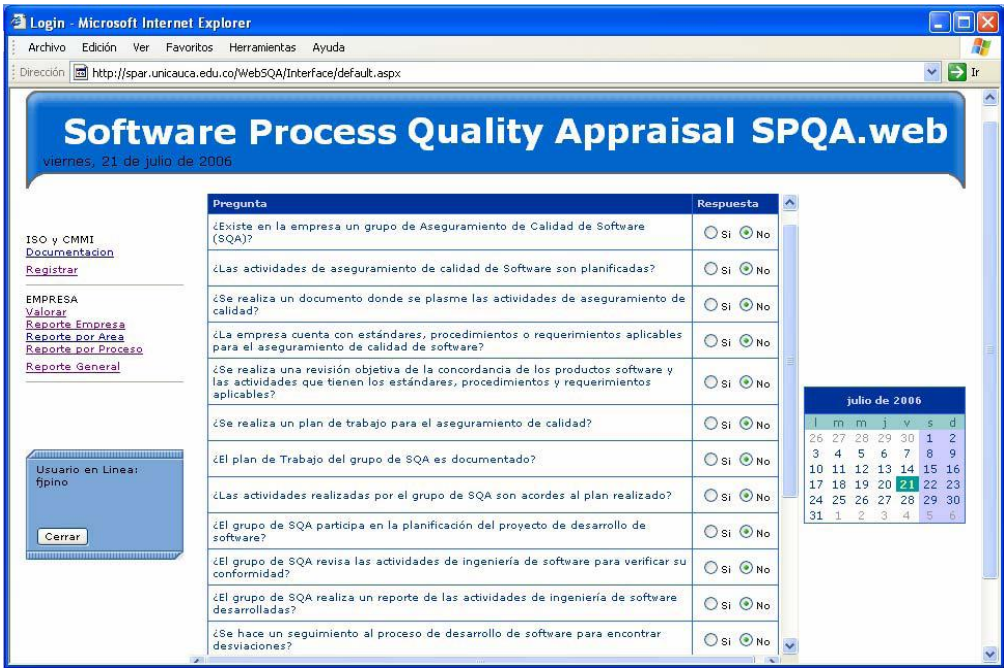


Figura 3.5. Presentación tradicional de un cuestionario de valoración de procesos.

Una de las propuestas de este trabajo es darle un enfoque más ágil a la evaluación. Para conseguir este objetivo, la herramienta presenta un marco más interactivo para la evaluación por cuestionario en comparación con las soluciones anteriores.

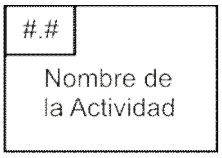
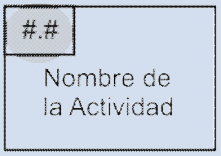
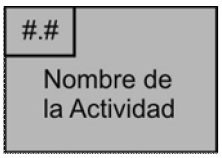
3.3.1.2. Definición de un método de evaluación basado en el Modelado del Proceso de desarrollo actual

La segunda fase de la evaluación está relacionada con el modelado del proceso de desarrollo de software actual en la empresa. Esta propuesta hace uso de la *Process Change Methodology* propuesta en [Fowler99] para describir el proceso actual basado en Moprosoft, el cual en una siguiente etapa será mapeado con el proceso ideal que marca el modelo. Para llevar a cabo esta tarea la herramienta introduce el uso de una notación gráfica para evaluar o modificar el actual proceso de la empresa.

En este sentido, el SEI en colaboración con el SEPG (*Orgname's Software Process Engineering Groups*) propuso una notación gráfica para construir diagramas que mostraran las fases del desarrollo de software. En particular se utilizan diagramas de flujo para ilustrar la perspectiva del comportamiento del ciclo de vida del desarrollo software. Cada fase del desarrollo software está compuesta por actividades y los diagramas de flujo pueden ser usados para describir las relaciones

entre estas actividades (como la secuencia, colocación y decisiones). La Tabla 12 presenta la notación gráfica utilizada para construir los diagramas de procesos en la herramienta.

Tabla 12. Notación gráfica para construir diagramas de proceso.

Componente	Símbolo	Descripción
Flecha		Las flechas indican la secuencia de las actividades. En otros casos representa la asignación de una actividad para un determinado rol.
Caja		Una caja representa una actividad dentro del ciclo de vida del proceso software. Cada caja contiene un el nombre de la actividad y número de actividad dentro del proceso.
Números de las Cajas		• El número antes del punto decimal representa la fase a la pertenece la actividad. • El número después del punto decimal indica el número de actividad dentro de la fase.
Caja Sombreada		El sombreado se utiliza para mostrar un énfasis particular a ciertas actividades dentro de la fase de desarrollo.
Diamante		El diamante representa una decisión binaria (si/no): Si la respuesta para la pregunta es **si**, entonces la secuencia de la siguiente actividad parte de la etiqueta “SI”. Si la respuesta a la pregunta es **no**, entonces la secuencia de la siguiente actividad se genera a partir de la etiqueta “NO”.
Persona		Este símbolo representa un determinado rol dentro de la empresa el cual es responsable de una cierta actividad.
Inicio		Representa el inicio del proceso de desarrollo.
Final		El símbolo final representa la culminación del proceso de desarrollo software.

En base a la Tabla anterior, la herramienta proporciona al usuario los símbolos adecuados para el modelado del proceso software actual de la empresa. La interfaz consiste en un panel de símbolos donde el usuario puede elegir los elementos necesarios para describir su proceso de desarrollo actual de manera gráfica en un área de dibujo.

Cada símbolo posee características que ayudan a la mejor descripción del proceso de la organización, estas características también sirven de ayuda para el mecanismo, ya que se obtienen datos a ser utilizados posteriormente para mapear el proceso de la empresa con el proceso ideal propuesto por Moprosoft. Los componentes cuyas características serán parte del mapeo con el proceso ideal son *caja*, *diamante* y *persona*, debido a que con ciertas características de estos componentes se conseguirá ligar estos símbolos con los elementos del diagrama ideal (véase Figura 3.6).

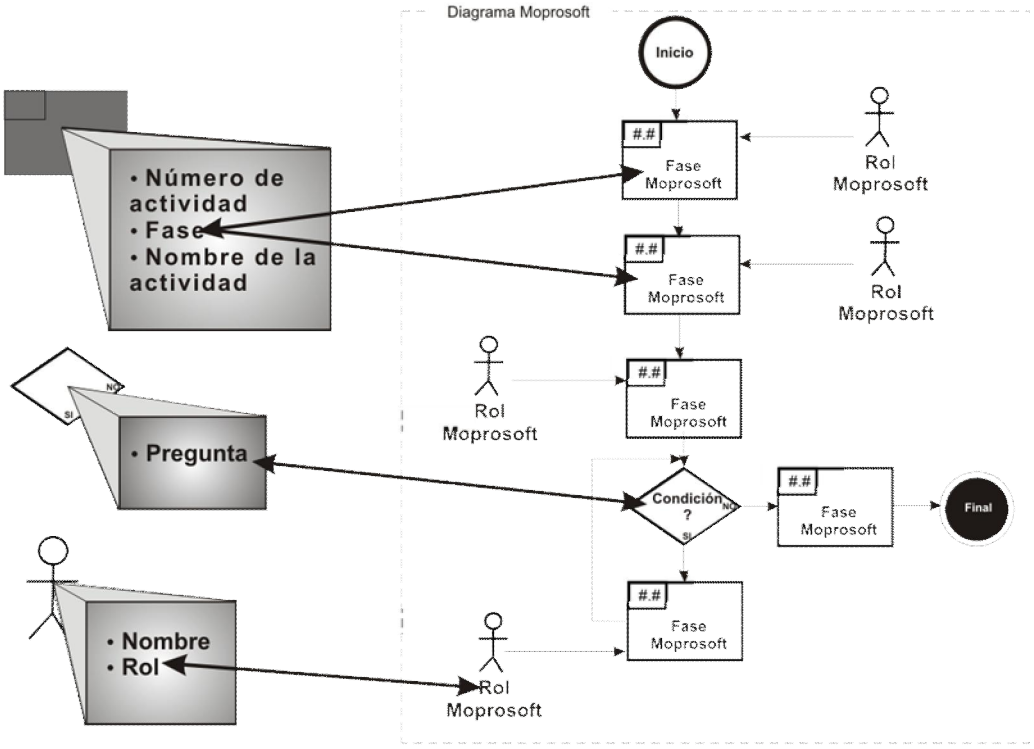


Figura 3.6. Principio de la forma en que se relacionan los elementos del diagrama actual con los elementos del diagrama Moprosoft.

En el caso del *componente caja* el número y nombre de la actividad son propios del diagrama del proceso actual en la empresa, en el caso de fase es la característica que nos brinda el poder ligar este componente con el proceso propuesto por el modelo, debido a que el nombre de la actividad puede ser cualquiera que la empresa le asigne, pero debe de pertenecer a alguna fase la cual es compatible con una fase que el modelo contiene. En el caso del *componente persona* el principio es el mismo, ya que cada empresa puede conocer el puesto que desarrolla una cierta actividad con un determinado nombre, pero este debe pertenecer a un rol propuesto por el modelo. En el caso de los *componentes diamantes* la correspondencia se hace por medio de las actividades que los anteceden y los suceden, así como de qué tipo de pregunta se trata. Este tipo de evaluación se basa en el cumplimiento de las fases genéricas del desarrollo de software y algunas actividades críticas para llevar por buen camino el desarrollo software. El sistema posee restricciones propias de un editor de diagrama de flujos y otras para llevar a cabo la evaluación de forma correcta como:

- Todo diagrama debe contar con un símbolo de inicio.
- El diagrama debe concluir con un componente final.
- Los números de actividades deben ser secuenciales impidiendo el salto de números.

- La secuencia de actividades debe estar ordenada.
- No pueden existir componentes o un bloque de estos sin conexión con el diagrama general.

Algunas de estas restricciones se muestran al realizar el diagrama y otras en la etapa de resultados.

Al encontrarse relacionadas las actividades y demás componentes del diagrama del proceso de desarrollo de la empresa con el proceso propuesto por el modelo de referencia se tiene la base para poder comparar y hallar inconsistencias en el proceso actual de la organización. De esta manera se comparan los dos procesos mediante el mecanismo de evaluación interno de la herramienta.

El mecanismo trabaja de forma secuencial buscando las actividades que propone el modelo ideal, además de comprobar que estas actividades sean realizadas en las fases que les corresponden. Tomando la secuencia del proceso idóneo se puede verificar qué actividades se encuentran fuera de secuencia y las actividades que se realizan en fases distintas a las que pertenecen. Otro punto a analizar es que las actividades tengan una secuencia coherente. Así también, se verifica que los responsables de determinadas actividades cumplan con el rol que se especifica en Moprosoft, independientemente del nombre que se le dé en una empresa a un determinado puesto de trabajo, éste debe de pertenecer a un rol propuesto por el modelo.

Los componentes de decisión son evaluados de forma diferente, ya que en este caso se toman en cuenta las actividades antecesoras y sucesoras debido a que este patrón se debe de repetir en ambos diagramas para ser correcto el símbolo, en el caso de encontrarse un cierto componente de decisión en el diagrama de la empresa y no así en el diagrama ideal, el método de evaluación respetará este símbolo como una buena práctica mientras cumpla con una secuencia y no traslape actividades.

Como resultado del mapeo de los dos diagramas se obtiene un nuevo diagrama, compuesto por las actividades del proceso de desarrollo de la organización que hayan cumplido con los criterios de evaluación del mecanismo, así como también de las actividades del proceso ideal que no sean halladas durante el proceso de mapeo (véase Figura 3.7).

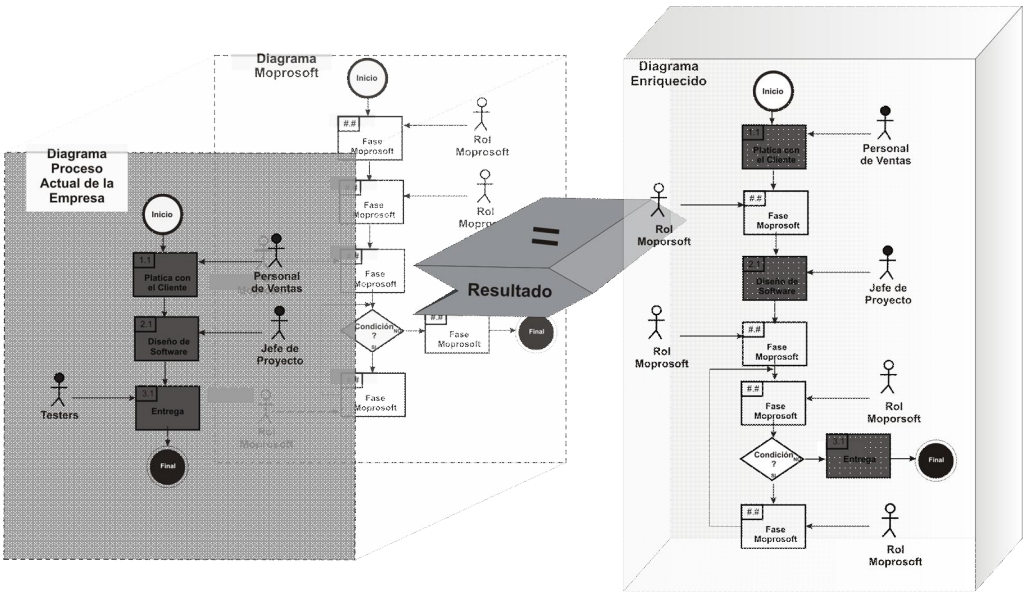


Figura 3.7. Mapeo entre el proceso actual y proceso Moprosoft.

El nuevo modelo brinda una perspectiva más cercana a lo que debería ser un proceso ideal en la empresa en base a las actividades que se llevan a cabo actualmente. Además de enriquecer el diagrama, el mecanismo marca las actividades que puedan estar mal ubicadas dentro del ciclo de vida del desarrollo software o los roles que no deben de ser responsables de una determinada actividad.

3.3.2. Definición de la fase de resultados

La implementación de esta fase consiste en calificar los mecanismos de evaluación y mostrar un resultado cuantitativo acerca de la situación de los procesos y actividades que se llevan a cabo dentro de la organización. Esta fase, permite a los miembros de la alta dirección de la empresa obtener la información acerca del ciclo SPI en cualquier momento, con esto puede controlar el rendimiento de los jefes de proyecto por medio de la fase de evaluación, obteniendo resultados finales y las gráficas procedentes de todo el proceso de evaluación. Esta fase sólo proporciona resultados a nivel de rendimiento a los jefes de proyecto, los resultados de toda la organización sólo pueden revisarse por la alta dirección a través del generador de reportes en la fase de mejora al proceso software.

Además de las gráficas de rendimiento generadas a partir de la fase de evaluación, se presenta una estimación del nivel de capacidad del proceso software alcanzado en base a los niveles propuestos por el método Evalprosoft presentados en [NYCE05b] (véase Tabla 13).

Tabla 13. Niveles de capacidad de Moprosoft

Nivel	Descripción
Nivel 0 Proceso Incompleto	El proceso no está implantado o falla en el alcanzar el propósito del proceso.
Nivel 1 Proceso Realizado	El proceso implantado logra su propósito y obtiene los resultados definidos.
Nivel 2 Proceso Administrado	El proceso Realizado se implanta de manera administrada y sus productos de trabajo están apropiadamente establecidos, controlados y mantenidos.
Nivel 3 Proceso Establecido	El proceso Administrado es implantado mediante el proceso definido, el cual es capaz de de lograr los resultados del proceso.
Nivel 4 Proceso Predecible	El proceso Establecido opera dentro de los límites para lograr sus resultados.
Nivel 5 Optimizando el Proceso	El proceso Predecible es continuamente mejorado para lograr las metas de negocios actuales y futuras relevantes.

Al contar con los datos obtenidos en la fase de evaluación y mediante la escala de respuestas usada y propuesta en [Garcia07] se procede a la etapa de calificación de los atributos del proceso. Para esta tarea se realiza un vaciado de los datos de la evaluación y basados en la escala ordinal

propuesta en [NYCE05b], se califica el grado de cumplimiento del atributo del proceso (véase Tabla 14). El conjunto de las calificaciones de los atributos de un proceso forman su perfil.

Tabla 14. Escala ordinal para calificar el grado del cumplimiento del atributo del proceso.

Escala	Descripción	Valor
N	No alcanzado	0 -25% del alcance
P	Parcialmente alcanzado	> 25% hasta el 60% del alcance
A	Ampliamente alcanzado	> 60% hasta el 85% del alcance
C	Completamente alcanzado	> 85% hasta el 100% del alcance

El nivel de capacidad alcanzado por el proceso se deriva de la calificación de los atributos correspondientes tomando como referencia la Tabla 15 (NYCE05b). En la Tabla la letra **A** significa Ampliamente Alcanzado, mientras que la letra **C** expresa completamente Alcanzado.

Tabla 15. Calificación del nivel de capacidad del proceso.

Nivel / Calificación mínima Atributo	1	2	3	4	5
Realización del proceso	A	C	C	C	C
Administración de la realización	-	A	C	C	C
Administración del producto de trabajo	-	A	C	C	C
Definición del proceso	-	-	A	C	C
Implantación del proceso	-	-	A	C	C
Medición del proceso	-	-	-	A	C
Control del proceso	-	-	-	A	C
Innovación del proceso	-	-	-	-	A
Optimización del proceso	-	-	-	-	A

Las calificaciones de los atributos de un proceso conforman su perfil de calificaciones de atributos. El nivel de capacidad del proceso es *"el nivel cuyo cumplimiento de los atributos es, al menos, ampliamente alcanzado y el cumplimiento de los atributos de los niveles inferiores es completamente alcanzado"*.

Con este mecanismo, la herramienta además de proporcionar gráficas referentes al nivel de desempeño de los jefes de proyecto provee a la alta dirección una aproximación del nivel de capacidad de los procesos de la empresa. El contexto de trabajo de la fase de resultados consiste en

dos partes, un mecanismo para generar los resultados del cuestionario y otro para generar los resultados del modelado. El primero consiste en vaciar la base de datos obtenida con las respuestas de los usuarios a los cuestionarios desarrollados en la fase de evaluación; una vez que estos datos son obtenidos del mecanismo en base a las directrices de calificación propuestas por Evalprosoft, el resultado se muestra en forma de gráficas donde se muestra el nivel de desempeño por cada proceso evaluado, así como también las calificaciones obtenidas para cada actividad.

El segundo mecanismo es el encargado de generar los resultados para la evaluación por modelado de proceso, el mecanismo hace uso del modelo generado por el usuario en la fase anterior junto con el proceso ideal guardado previamente en una base de datos de activos de Moprosoft, los resultados a mostrar son un nuevo diagrama enriquecido con actividades no halladas en el proceso actual y contrastando las actividades erróneas, además se proporciona en forma de lista los errores y carencias detectados en el proceso actual de desarrollo software de la organización. Con esto se pretende que el usuario tenga dos formas de analizar el resultado de la evaluación por modelado.

Ambos mecanismos trabajan en conjunto para detectar incoherencias en la evaluación, que puedan presentarse por la falta de compromiso y veracidad del personal evaluado, dependiendo del nivel de inconsistencias entre las dos evaluaciones, la calificación del proceso o actividad penalizará o en su defecto se anulará. Por último, en esta fase se proporciona una estimación del nivel de capacidad de los procesos de acuerdo a los lineamientos marcados por Evalprosoft para adquirir un cierto nivel de madurez (véase Figura 3.8).

Los resultados de nivel de desempeño serán proporcionados a los jefes de proyecto. En el caso de la alta dirección, se proporciona un análisis más completo sobre el estado de la empresa en la fase de mejora al proceso.

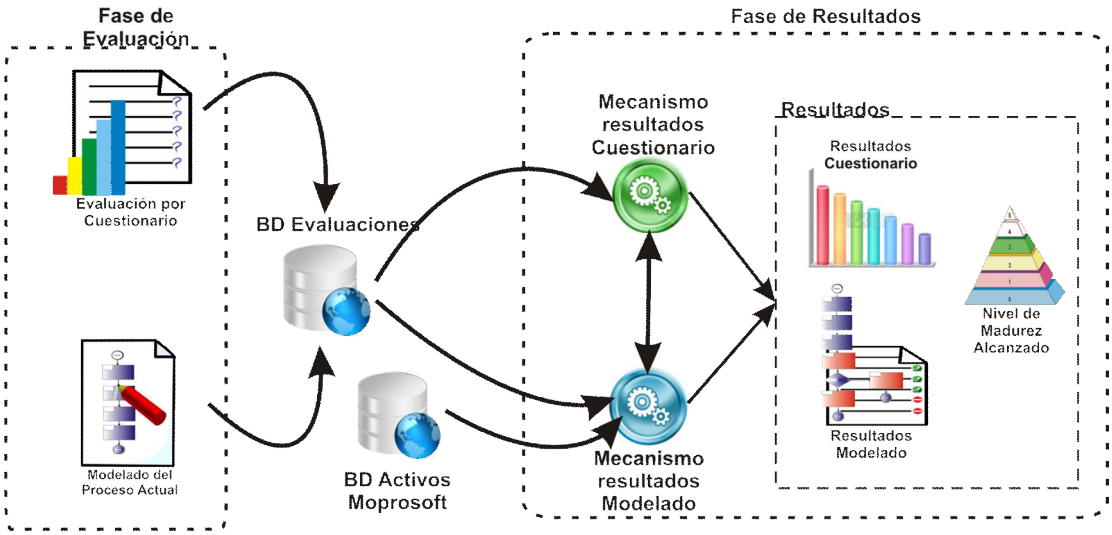


Figura 3.8. Contexto de trabajo de la fase de Resultados.

3.3.3. Definición de un mecanismo de generación de planes de acción para la fase de mejora al proceso dentro de la empresa

Para recomendar acciones de mejora al proceso software de una MPyME desarrolladora de software se propone un mecanismo de análisis y una plantilla de plan de acción, en base a los resultados de los dos tipos de evaluaciones realizados. El análisis de las respuestas obtenidas en el cuestionario y los resultados del mapeo del proceso permiten determinar qué prácticas no son cubiertas por el equipo de desarrollo software de la empresa y las que han sido implantadas en la empresa. Este análisis también nos brinda la oportunidad de hallar los productos que no se generan

en la empresa a causa de la poca capacidad de ciertas actividades. De esta manera este análisis ayuda a identificar prácticas y productos que no han sido implantados en la empresa, así como las prácticas que necesitan mejorarse para difundirlas dentro de la organización. Este mecanismo se centra en proporcionar recomendaciones que se basan en las prácticas y productos del modelo Moprosoft, para implantar ciertas prácticas y productos inexistentes en la empresa o mejorar prácticas débilmente implantadas en cada una de las áreas de proceso evaluadas.

Los planes de acción generados por el mecanismo se basan en las prácticas débiles, debido a que la calificación obtenida en las prácticas define el nivel de capacidad de un área de proceso. Esto permitirá detectar las prácticas que no han sido implantadas o que necesiten mejorarse, y por consecuencia de esto la madurez de la organización y la capacidad del proceso se verán mejoradas. Con efecto de facilitar el análisis del desempeño de las fases de cada área de proceso, se propone relacionar colores (rojo, amarillo y verde) para las prácticas que necesiten mejorar en base a los propuesto por el método de evaluación Evalprosoft (véase Tabla 16), en caso de no contar con un grado de cumplimiento mínimo, la práctica se enlistara como una práctica que no existe en la organización.

Tabla 16. Caracterización de los niveles de desempeño por fase.

Color	Nivel de Desempeño (%)	Descripción
-	0 -15%	La fase no alcanzo el grado de cumplimiento necesario o no existe dentro de la empresa.
Rojo	16% - 50%	La fase tiene un grado de cumplimiento pobremente alcanzado.
Amarillo	51% - 85%	La fase tiene un grado de cumplimiento medianamente alcanzado.
Verde	86% -100%	La fase tiene un grado de cumplimiento completamente alcanzado.

Conociendo el nivel de desempeño de las fases se pone especial énfasis en las que se encuentran dentro de un porcentaje 0 – 50%, ya que son las que impactan negativamente en la consecución de un determinado nivel de capacidad del modelo Moprosoft y afectan el logro de una meta, por lo tanto en estas se centran las recomendaciones de mejora. Para identificar aquellas prácticas que estén afectando negativamente al resultado obtenido para la fase de desarrollo, y por lo tanto al proceso, se revisarán las respuestas de las actividades de tipo *"A veces"*, *"Rara Vez"* y *"Nunca"*, ya que significan que inciden en el bajo desempeño de una fase de desarrollo o proceso y por lo tanto esa práctica es candidata para considerarla como un objetivo de mejora. Otro punto es que algunas actividades tienen asociado un producto, por lo tanto el producto ligado a una actividad de desempeño bajo será candidato también a un proceso de mejora. Para llevar a cabo el proceso de mejora en la práctica se propone una actividad a realizar la cual puede o no generar un producto (véase Tabla 17).

Tabla 17. Ejemplo de organización de los elementos de una fase.

Fase	Práctica	Actividad	Productos
Fase de Inicio	1	A1	Producto 1
	2	A2	Producto 2
	3	A3	Producto3
Fase de Requisitos	1	A1	Producto 1
	2		
	3	A2	
	4	A3	Producto2

El listado de productos se realiza mediante la extracción de la base de datos de evaluaciones aquellos productos que se deben generar por cada práctica evaluada entre el 0% y 50%. Los productos que se deben de obtener al realizar las actividades se obtienen del modelo Moprosoft (en los apartados Salidas y Productos Internos de la norma). La importancia de especificar un listado de productos en el plan es porque estos representan el resultado de ejecutar las tareas y porque permitirán la validación de la ejecución del plan [Garcia07].

Se dejan de lado las recomendaciones de tiempo, costo y recursos materiales necesarios para ejecutar el plan, centrándose el plan de acción en actividades a realizar y los productos esperados. La Figura 3.9 muestra la plantilla a utilizar para la presentación de la guía de mejora.

Los datos del encabezado son proporcionados por la alta dirección en el momento de dar el alta en la herramienta de evaluación; en esta parte también se asigna el personal que participará en las fases de evaluación, así como también las áreas de proceso que se requiere evaluar.

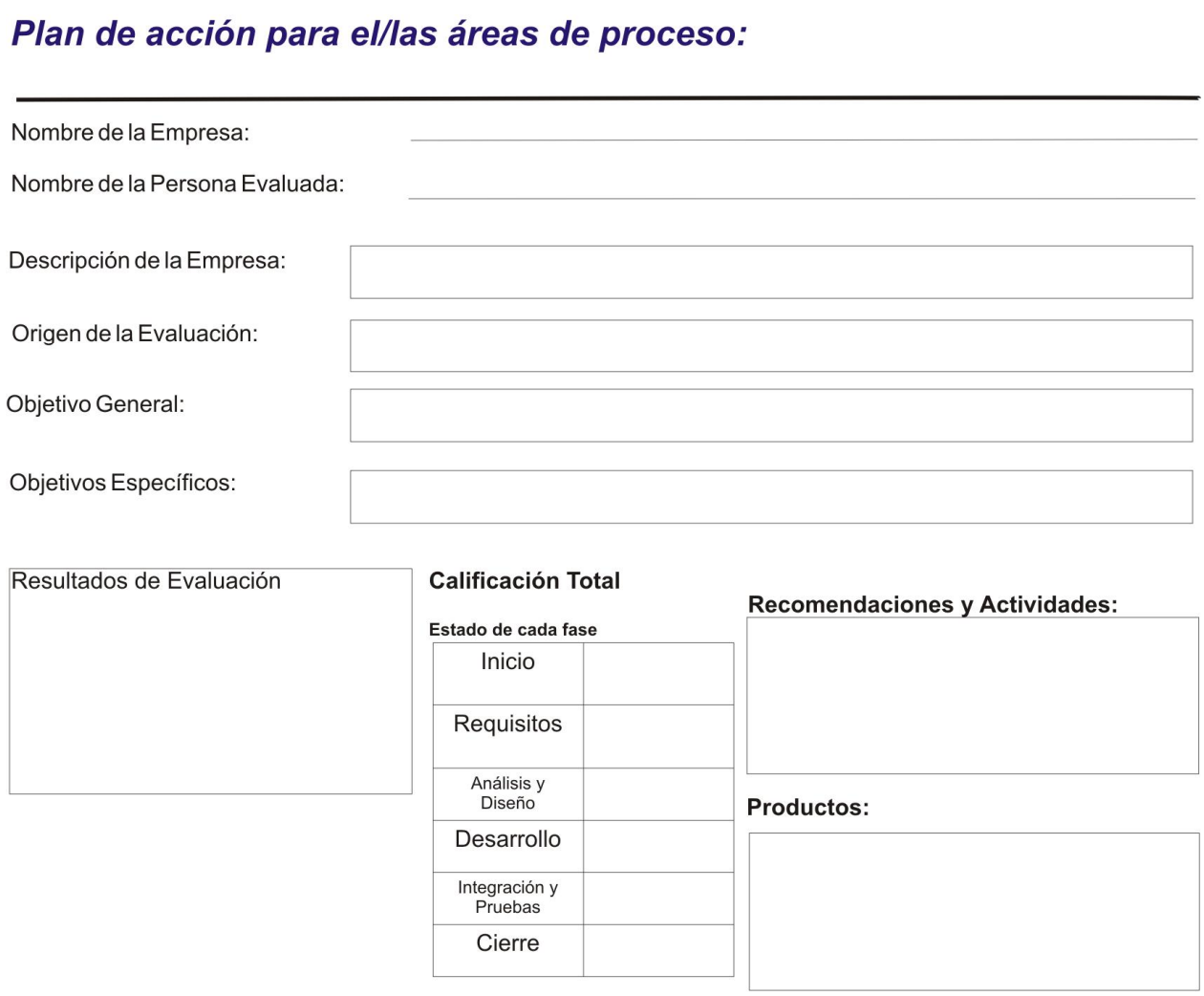


Figura 3.9. Plantilla de plan de acción.

3.3.4. Diseño de SelfVation combinando UWE y el método RUX

El desarrollo RIA ha venido incrementándose en los últimos años, sin embargo existen pocas metodologías para el desarrollo de este tipo de aplicaciones. Un estudio presentado en [Preciado05] justifica el "por qué" las metodologías de desarrollo Web tradicional no se acoplan al desarrollo de las RIA. Las RIA permiten construir aplicaciones ricas en datos y contenidos multimedia, con alta interactividad, lo que aumenta las capacidades que ofrecen las aplicaciones Web tradicionales. Es por esta razón que muchas de las metodologías existentes para el desarrollo Web no cumplen con los requisitos o nuevas funcionalidades que caracterizan a las RIA. Por esto es que una RIA sigue siendo desarrollada de forma 'ad-hoc', lo cual a menudo da lugar a errores y dificulta el mantenimiento [Preciado08].

Para el desarrollo de este trabajo se utilizó la metodología propuesta en [Preciado08]. Está metodología consiste en combinar el enfoque de ingeniería Web basado en UML (UWE) [Koch07] con el método RUX [Linaje07] para el desarrollo de la herramienta RIA. UWE ofrece una notación específica para la representación gráfica de las aplicaciones Web y un método para conducir y modelar el desarrollo de sistemas Web. Por su parte el método RUX es un enfoque para modelar y conducir el modelado de interfaces de usuario (UI) de las RIA. En esta metodología, UWE es

utilizado para especificar el contenido, navegación y procesos de negocio de la aplicación Web, y el método RUX es usado en la última capa del modelo UWE para agregar las capacidades típicas de una UI enriquecida, por ejemplo el comportamiento temporal e interacciones enriquecidas para el usuario.

3.3.4.1. Diseño conceptual en base a la metodología UWE

Para el diseño conceptual de la herramienta se utilizó la metodología para Web tradicionales ya que ofrece los conceptos básicos para obtener una idea general de cómo deberá de comportarse la herramienta. El método de creación elegido se presenta en [Koch07a], el cual está basado en un enfoque UWE. Comenzando con un análisis de requisitos realizado bajo la técnica de casos de uso, que se centra en la fase de diseño, el modelo conceptual de la aplicación es usado como la pauta para modelar el modelo de navegación. El modelo de navegación es el siguiente en realizarse, el cual muestra cómo navegar por el espacio de navegación de la herramienta utilizando elementos de acceso como índices, consultas y menús. Por último, un modelo de presentación es construido basado en el modelo de navegación [Koch07a].

El método proporciona orientación para la construcción sistemática y gradual de los modelos mencionados. La construcción se realiza en un proceso de diseño iterativo e incremental. Las actividades que se modelan son el análisis de requisitos, conceptual, de navegación y el diseño de la presentación. Los cuales producen:

- Modelo de Casos de Uso.
- Modelo Conceptual.
- Modelo de Navegación.
- Modelo de Presentación.

La Figura 3.10 muestra los modelos representados como paquetes UML y mostrando la relación que existe entre cada uno de ellos.

El objetivo del análisis de requisitos es encontrar los requerimientos funcionales de la aplicación Web y representar estos requisitos mediante un diagrama de casos de uso.

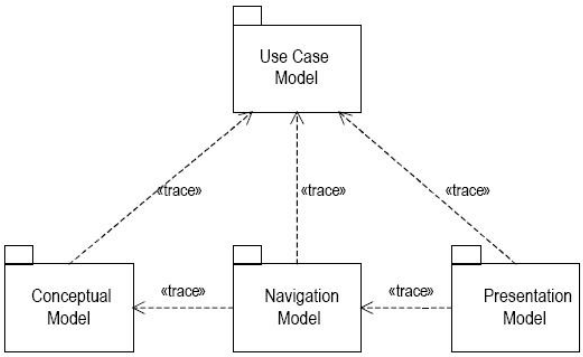


Figura 3.10. Modelos utilizados en el método de creación UWE.

El objetivo del diseño conceptual es construir un modelo conceptual del dominio de la aplicación, teniendo en cuenta los requisitos capturados con los casos de uso. Las técnicas orientadas a objetos tradicionales se utilizan para construir el modelo conceptual, como la creación de clases, asociaciones y la definición de estructuras de herencia. El modelo se representa mediante un diagrama UML ordinario de clases.

Basado en el modelo conceptual el método de navegación propone un conjunto de directrices para construir un modelo de navegación que representa el espacio de navegación y la estructura de navegación mediante la adición de elementos de acceso que pueden ser utilizados para la navegación. El método incluye un conjunto de elementos de modelado UML estereotipados para el diseño de navegación, como índices, visitas guiadas, consultas a la base de datos y menús. Estos estereotipos se utilizan para representar el modelo de navegación de una forma más clara y sencilla.

El modelo de presentación tiene como objetivo el diseño de interfaces de usuario abstractas y el diseño de la interacción del usuario con la aplicación Web.

3.3.4.1.1. **Modelo de casos de uso**

El método de creación para UWE propone los casos de uso para la captura de requisitos del sistema. Es una técnica centrada en el usuario que obliga a definir quienes son los usuarios (actores) de la aplicación y ofrece una forma intuitiva para representar las funcionalidades que tiene que cumplir una aplicación para cada actor.

El método de creación aplica los pasos sugeridos por muchos autores para la construcción del modelo de casos de uso de una aplicación Web. Estos pasos son:

1. Establecer los actores.
2. Para cada actor describa las actividades que tendrá que realizar.
3. Agrupar las actividades por casos de uso.
4. Establecer relaciones entre actores y casos de uso.
5. Establecer relaciones “include” y “extends” entre casos de uso.
6. Simplificar el modelo de casos de uso mediante la definición de relaciones de herencia entre los actores y/o entre casos de uso.

En el diagrama mostrado en la Figura 3.11 se proponen los casos de uso necesarios para capturar los requisitos del sistema. El diagrama describe una parte del comportamiento de la aplicación sin revelar la estructura interna. En el diagrama se definen los dos actores principales que harán uso de la herramienta como lo son *un miembro de la alta dirección* y *un jefe de proyecto*. Como se observa en el diagrama el miembro de los altos mandos de la empresa es el encargado de iniciar una evaluación, así como dar de alta a los usuarios a los que estará dirigida.

Además, este usuario es el encargado de definir qué procesos se evaluarán en la organización y asignar cada uno de estos procesos a un usuario dado de alta previamente. Este usuario cuenta con los mayores privilegios dentro del sistema, por lo que puede visualizar los resultados de todos los usuarios que se hayan evaluado dentro de la organización. El plan de acción generado en base a las evaluaciones será proporcionado a este usuario, ya que al encontrarse en los altos mandos de la empresa será el encargado de poner en marcha una iniciativa SPI dentro de la organización en base a las actividades y productos propuestos en el plan de acción generado por SelfVation.

El segundo actor que se representa en el diagrama es un líder de proyecto o personal altamente inmiscuido en todo el proceso de desarrollo software y que conoce todas las fases y ciclo de vida del software. Este tipo de usuarios serán los encargados de llevar a cabo las etapas de las evaluaciones propuestas: por una parte contestar el cuestionario y por otra el modelado del diagrama de procesos. Este usuario solo tendrá acceso a los resultados de su evaluación, pero quedará restringido el acceso a los resultados de otros usuarios y al plan de acción generado en base a los resultados. En ambos casos, se tendrá que iniciar una sesión para hacer uso de la herramienta, en caso contrario el sistema no proporcionará ningún tipo de funcionalidad.

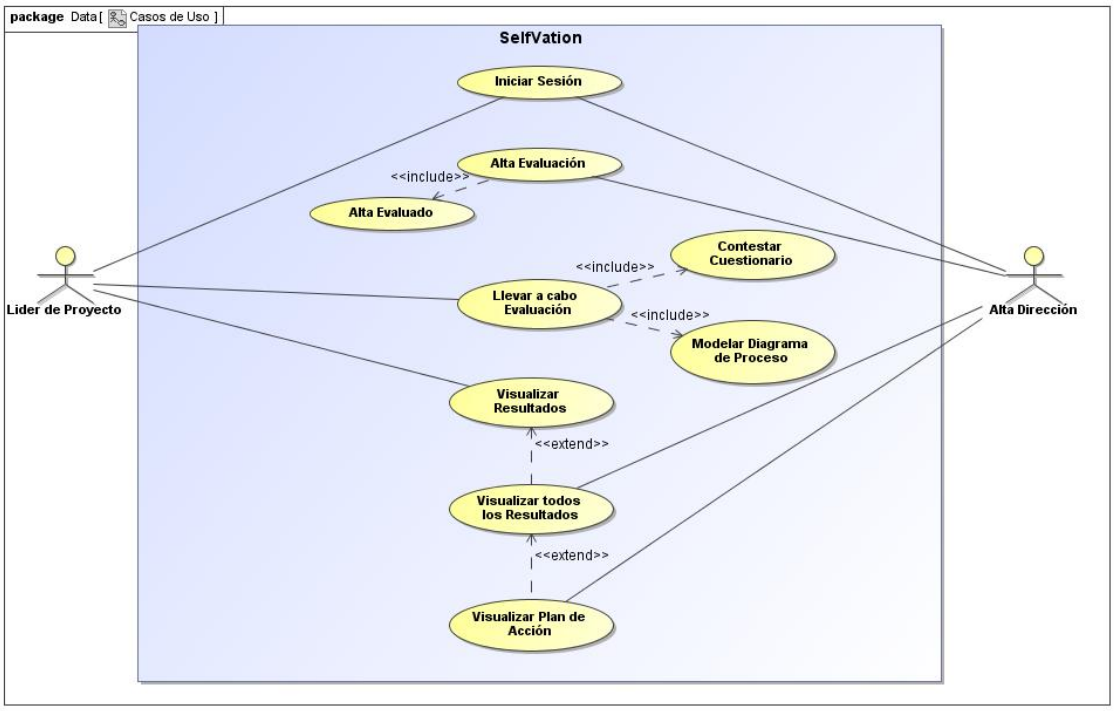


Figura 3.11. Diagrama de casos de uso de SelfVation

3.3.4.1.2. **Modelo Conceptual**

El diseño conceptual se basa en el análisis de requisitos de la etapa anterior. Incluye los objetos implicados en la interacción entre el usuario y la aplicación. El diseño conceptual tiene como objetivo construir el modelo de clases con estos objetos, ignorando las rutas de navegación, presentación y aspectos de la interacción en lo posible. Estos aspectos se posponen a los pasos de navegación y presentación del proceso de creación UWE [Koch07a].

El diagrama de contenido se refiere al modelo conceptual de la herramienta. Un diagrama de clases en UML se utiliza para representar gráficamente un modelo conceptual como visión estática que demuestre una colección de los elementos estáticos del dominio [Ocaña04]. Un diagrama de contenido proporciona una especificación del dominio de la información relevante para el software Web [Preciado08]. El modelo conceptual incluye los objetos implicados en las actividades típicas que los usuarios realizarán en la herramienta, es decir, los objetos que son relevantes para la realización de una actividad o que son el resultado de una de ellas.

El desarrollador puede seguir técnicas reconocidas de modelado orientado a objetos para construir el modelo de clases UML que representen el diagrama de contenido, como:

1. Búsqueda de clases del sistema.
2. Especificación de los atributos más importantes.
3. Determinación de las asociaciones entre clases.
4. Identificar agregación y composición de clases.
5. Definir herencia entre clases.
6. Definir condiciones.

En la Figura 3.12 se muestra el diagrama de contenido vinculado con el modelo conceptual de la herramienta SelfVation. El diagrama muestra los elementos principales que componen el modelo conceptual de la herramienta. El componente importante de la herramienta estará dado por las empresas que se evalúen en la herramienta, para cada empresa estará asociada una o varias evaluaciones, además la empresa está asociado con uno o varios usuarios, el número se define por la cantidad de empleados que la empresa quiera evaluar. Los elementos de importancia en el modelo conceptual de la herramienta serán los métodos de evaluación, por una parte las preguntas que componen al cuestionario de evaluación, estas preguntas pueden o no estar ligadas a un producto, y a su vez generarán una respuesta por parte del usuario. El modelado del proceso es otra parte esencial del modelo conceptual, este segmento está constituido esencialmente de un diagrama de procesos que el usuario generará, el mismo estará compuesto por los símbolos de los que haga uso el usuario. Cabe resaltar que los elementos mostrados en el diagrama son los elementos estáticos de los que hará uso la herramienta, ya que estos componentes son esenciales para su buen funcionamiento, ya que elementos como los resultados son llevados a cabo en tiempo real por los mecanismos de evaluación correspondientes.

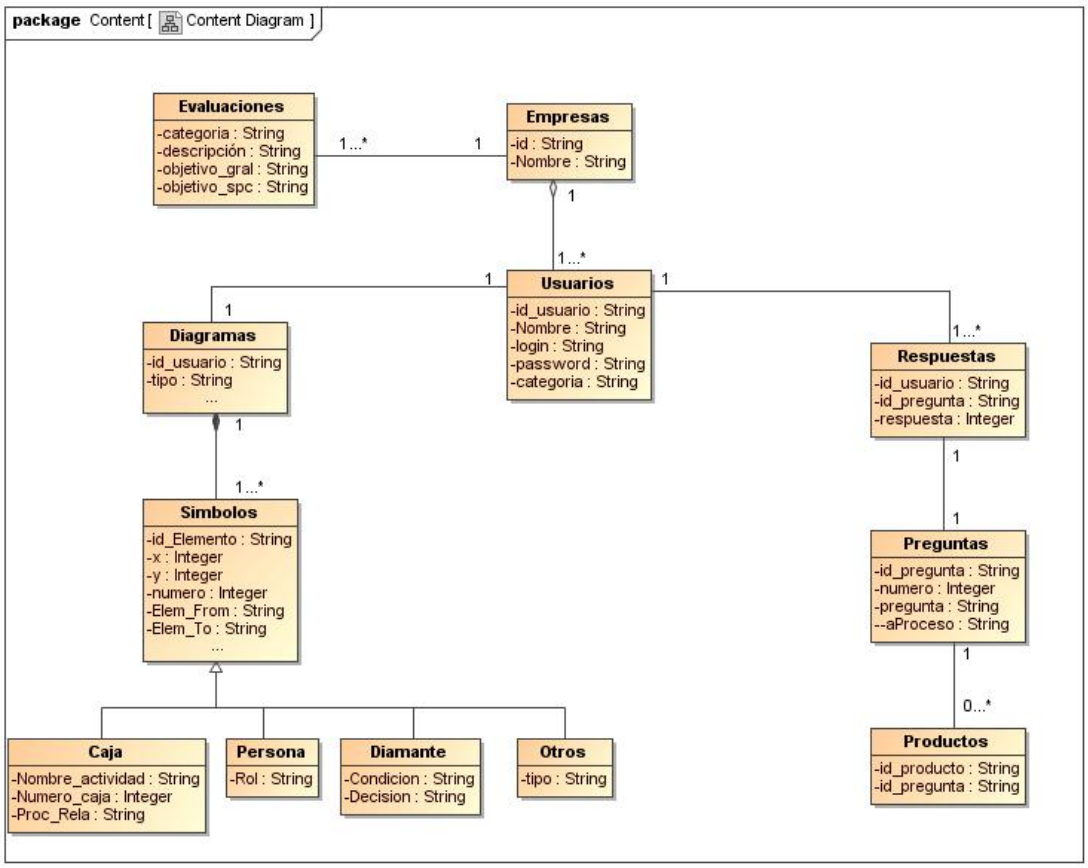


Figura 3.12. Diagrama de contenido.

3.3.4.1.3. **Modelo de navegación**

Basados en los requisitos y en el modelo de contenido, el modelo de navegación de la aplicación Web se construye para especificar la estructura del hipertexto del sistema, que se da por nodos y enlaces [Preciado08]. En este modelo se describe cómo es la navegación con el apoyo de elementos de acceso, como índices, visitas guiadas, consultas a la base de datos y menús [Koch02].

Este modelo comprende la especificación de qué objetos pueden ser visitados mediante la navegación a través de la aplicación Web y las asociaciones entre ellos. En el marco de UWE se destaca como el más importante, pues con él se pueden representar elementos estáticos, a la vez que se pueden incorporar lineamientos semánticos de referencia para las funcionalidades dinámicas de una aplicación Web [Ocaña04].

Aunque no hay una forma automatizada para la construcción del modelo de navegación, hay varias pautas que puede seguir el desarrollador:

1. Incluir las clases del modelo conceptual que son relevantes para la navegación como clases de navegación en este modelo (es decir, las clases de navegación se pueden asignar a las clases conceptuales). Si una clase conceptual no es un objeto de visita en el modelo de casos de uso, es irrelevante en el proceso de navegación y por lo tanto se omite en el modelo de navegación.
2. Mantener la información de la clases que se omiten (si es necesario) como atributos de otras clases en el modelo de navegación. Todos los demás atributos que corresponden a las clases del modo conceptual se insertan en las clases del modelo de navegación correspondiente. Excluir a los atributos de las clases conceptuales que resulten irrelevantes para la presentación del modelo de navegación.
3. Las asociaciones del modelo conceptual se mantienen en el modelo de navegación. Las asociaciones adicionales se pueden agregar para la navegación directa para evitar las rutas de navegación de longitud mayor que uno.
4. Añadir las asociaciones adicionales basadas en la descripción de los requisitos o los escenarios descritos por el modelo de casos de uso.
5. Considerar aquellas asociaciones, que tengan multiplicidad mayor que uno al final de una asociación dirigida.
6. Para cada asociación de este tipo, elegir uno o más elementos de acceso (consultas, índices) para realizar la navegación.
7. Considerar aquellas asociaciones, que tienen como inicio una clase de navegación.
8. Asociar a cada clase de navegación, que tiene (paso 6) por lo menos una asociación saliente, su correspondiente clase menú. La asociación entre una clase de navegación y su correspondiente clase menú es una composición.
9. Reorganizar un menú en un menú con submenús.

En la Figura 3.13 se muestra el diagrama de navegación correspondiente a la aplicación SelfVation. El diagrama muestra las diferentes tareas que se pueden realizar en la aplicación, así como la interacción de los elementos dinámicos con los estáticos. En el diagrama se describe la navegación con la que debe de contar la aplicación, partiendo de una página principal y un inicio de sesión que será necesario para acceder a las funcionalidades totales del sistema, el sistema restringirá o proveerá funcionalidades propias del tipo de usuario que acceda al sistema. Así mismo el diagrama muestra procesos internos del sistema para llevar a cabo ciertas tareas, este tipo de procesos no son visibles para el usuario pero son necesarias para el buen desempeño de la herramienta. En el diagrama también se puede observar los procesos y la forma en que se hace una consulta a la base de datos ya sea para consultar información, que servirá para mostrar ciertos datos o para generar resultados, así como también para agregar nuevos datos que el usuario generé durante el transcurso de las evaluaciones.

En el sistema la navegación comienza por medio de un menú principal, en primera instancia el usuario debe de iniciar sesión para acceder a todas las opciones del menú principal. Iniciada la

sesión, el sistema mostrará las opciones que correspondan al tipo de usuario que ha iniciado sesión. Como se ha mencionado con anterioridad los usuarios pertenecientes a los altos mandos de la empresa les corresponden las opciones de la implantación de la mejora al proceso de desarrollo software, mientras que a los usuarios pertenecientes a los cargos operativos les corresponden las etapas de evaluaciones. El diagrama muestra la forma en que los diferentes usuarios podrán navegar en la herramienta para acceder a las funcionalidades que esta ofrece.

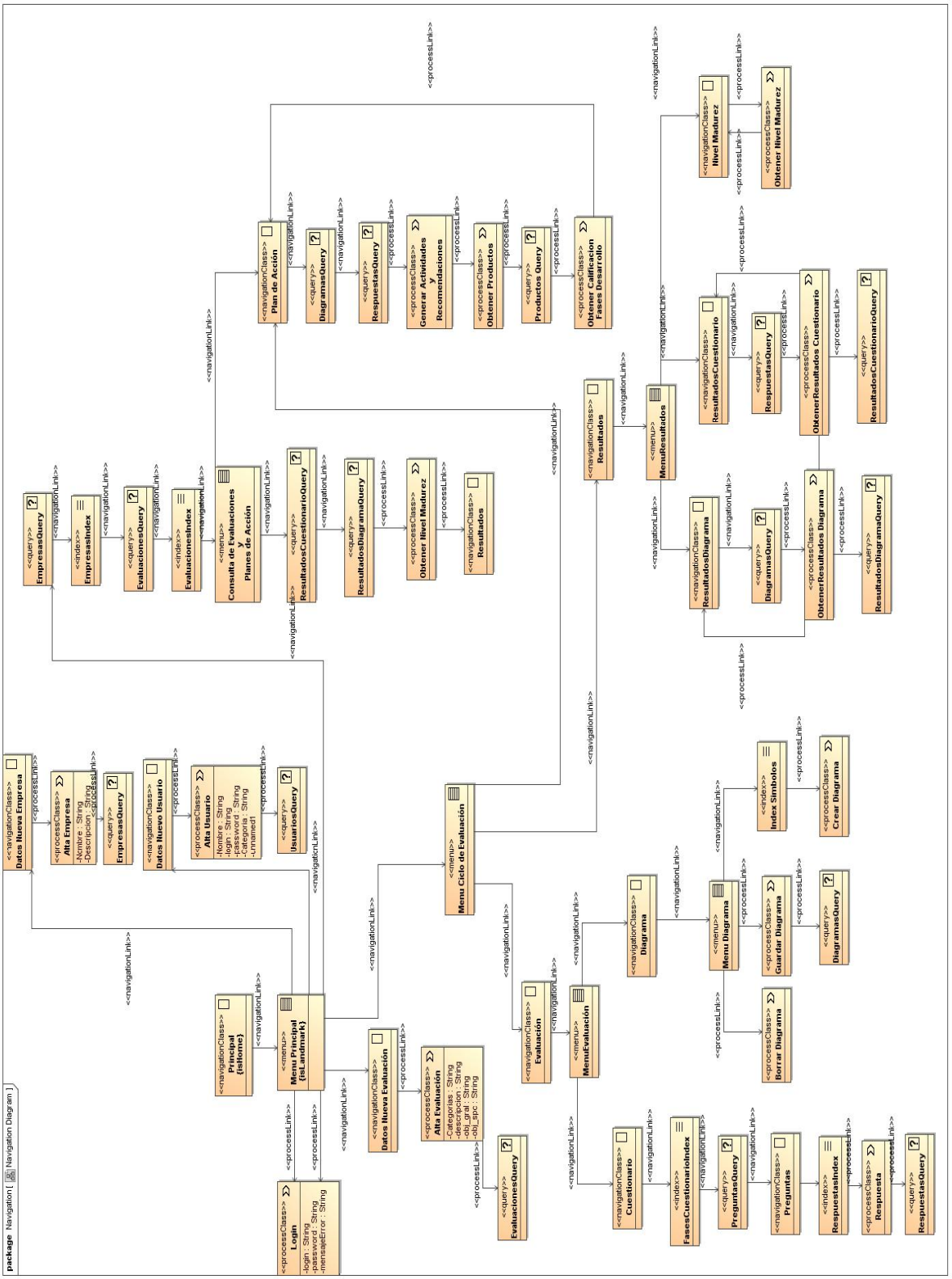


Figura 3.13. Diagrama de navegación.

3.3.4.1.3.1 Capa de estructura de procesos

Con el fin de mostrar las relaciones entre las diferentes clases de procesos, las cuales no se encuentran a la vista del usuario como las clases navegables pero son parte fundamental del sistema ya que son las encargadas de llevar a cabo internamente los procesos necesarios para que el sistema funcione de manera correcta y los resultados sean los esperados.

En la Figura 3.14 se muestra el diagrama de estructura de procesos correspondiente a la herramienta SelfVation, en el se muestran los procesos que se llevan a cabo internamente dentro de la empresa, así como también se muestran las relaciones que existen entre cada uno de ellos. Algunos de estos procesos se pueden encontrar en el diagrama de navegación de la Figura 3.6, pero en este diagrama se muestran nuevos procesos que componen o que complementan a los mostrados en el diagrama anterior, así como las relaciones existentes en cada uno de estos procesos.

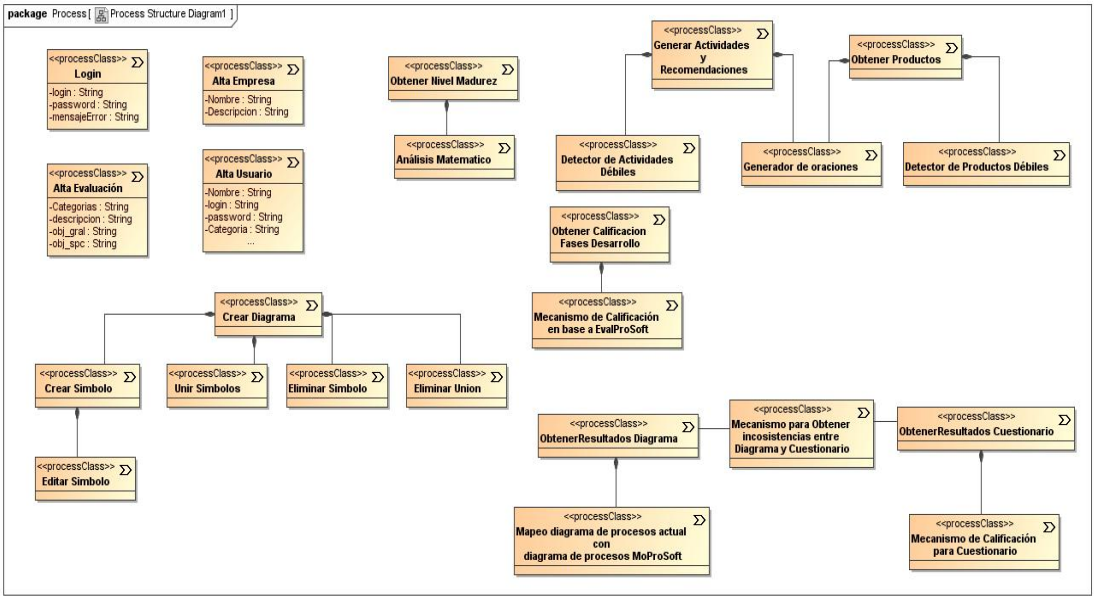


Figura 3.14. Diagrama de estructura de procesos.

3.3.4.1.4. **Modelo de Presentación**

El modelo de presentación pretende proporcionar una representación abstracta de la interfaz de usuario final y definir la interacción de las clases navegables con el usuario. Esta capa está basada en el modelo de navegación. El objetivo principal de esta capa es conseguir un modelo abstracto de la interfaz de usuario Web 1.0, que permita mostrar la forma en que la estructura de navegación es presentada al usuario [Preciado08]. El modelo de interfaz es un diseño a grandes rasgos de lo que será la interfaz final, decisiones acerca de detalles como tamaños, colores, fuentes u otros objetos de interfaz específicos no se consideran por ser parte de la implementación. El objetivo es visualizar la organización de la estructura de la aplicación Web de una manera más intuitiva que como lo hace el modelo de navegación [Koch02].

Para diseñar un modelo de presentación, se parte del modelo de navegación de la aplicación Web. Cada interfaz de usuario abstracta se representa como una composición de clases. Las siguientes reglas se pueden utilizar como directrices para la construcción del modelo de presentación:

1. Construir una clase de presentación para cada clase de navegación que se encuentra en el modelo de navegación. La clase define la presentación de una platilla adecuada para presentar las instancias de la clase, teniendo en cuenta los atributos dados. Elementos de interfaz estereotipados, como <<text>>,<<image>>, <<audio>>, <<video>> se utilizan para los tipos primitivos de presentación y <<collections>> son utilizados para un conjunto de elementos.
2. Construir una clase de presentación para cada menú e índice que se encuentran en el modelo de navegación. La presentación de un menú o un índice por lo general consiste en una lista de elementos “anchor”. Utilizar los estereotipos <<anchor>> o <<anchored collection>> para este propósito.
3. Construir una clase de presentación para cada consulta y visita guiada. Para las consultas utilizar el estereotipo <<form>> y para las visitas guiadas utilizar un menú con los elementos <<next>> y <<previous>>.
4. Determinar qué elementos de presentación deberán presentarse juntos al usuario (en una sola ventana). La clase de presentación correspondiente debe estar compuesta en una vista de interfaz de usuario (estereotipada por <<UI view>> o <<presentationPage>>).

En la Figura 3.15 se muestra la representación abstracta para la interfaz de la herramienta SelfVation. Esta interfaz está basada en el modelo de navegación presentado en la Figura 3.13, el modelo marca las bases para la implementación y navegación de la aplicación. En el modelo se proponen los menús, hipervínculos, textos, formularios, imágenes y otros componentes, con los que la aplicación debe de contar para llevar a cabo las tareas que se requieren para su correcto funcionamiento. El modelo brinda una amplia idea de cómo debe estructurarse la interfaz de usuario final de la herramienta, pero esté modelo es susceptible a cambios en la fase de implementación, ya que debido al enfoque RIA que se utiliza para el desarrollo de este sistema se pueden agregar elementos que ayuden a explotar de mejor forma las características de este tipo de aplicaciones. Este modelo será la base para desarrollar una UI bajo enfoque RIA.

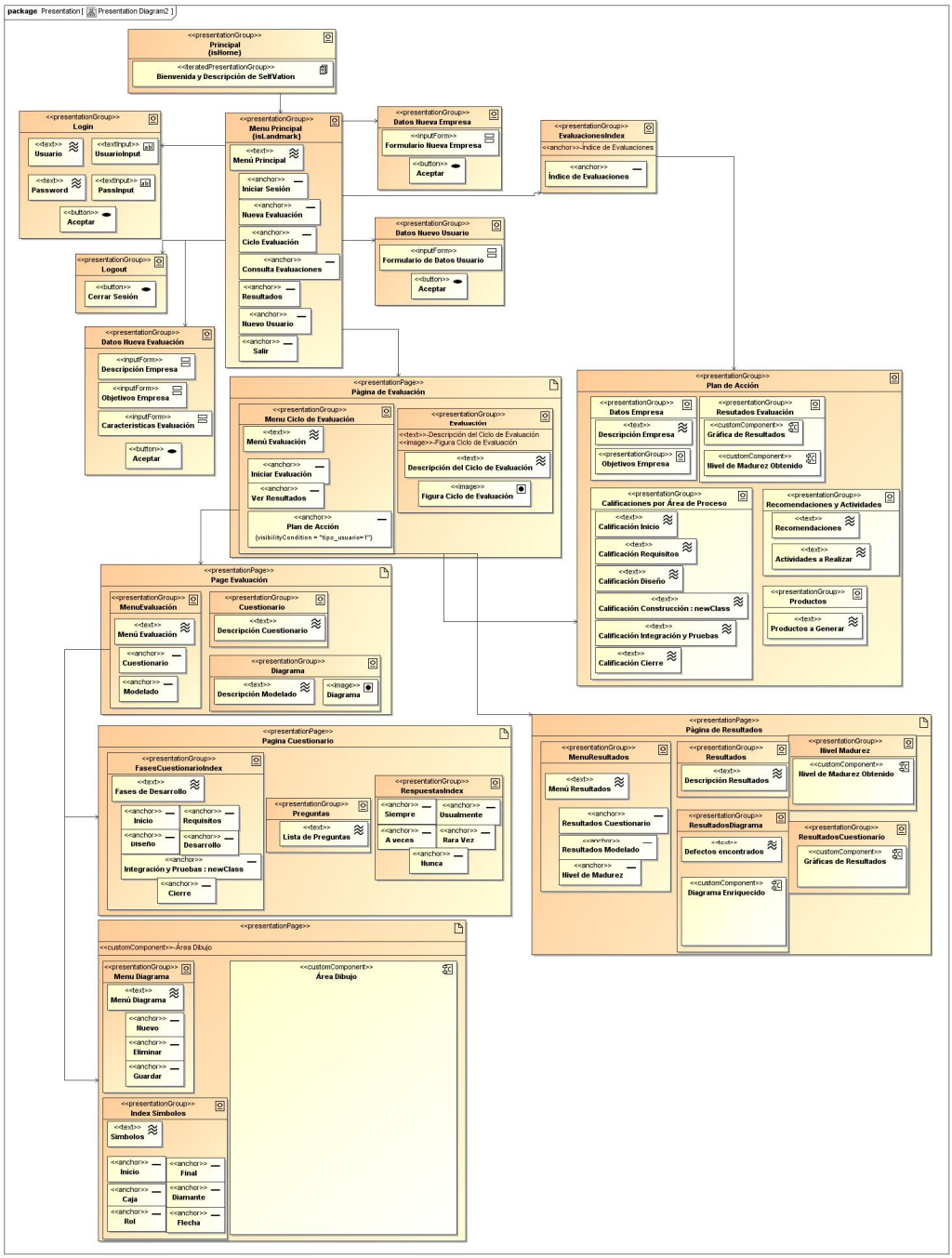


Figura 3.15. Modelo abstracto de interfaz de usuario para SelfVation.

3.3.4.2. Método RUX para el modelado de interfaz de usuario de RIAs.

El método RUX (véase Figura 3.16) es un modelo para guiar la especificación sistemática de UIs multimedia e interactivas. El método se puede combinar con otras metodologías Web para modelar los datos y la lógica de negocios de la aplicación Web [Linaje08].

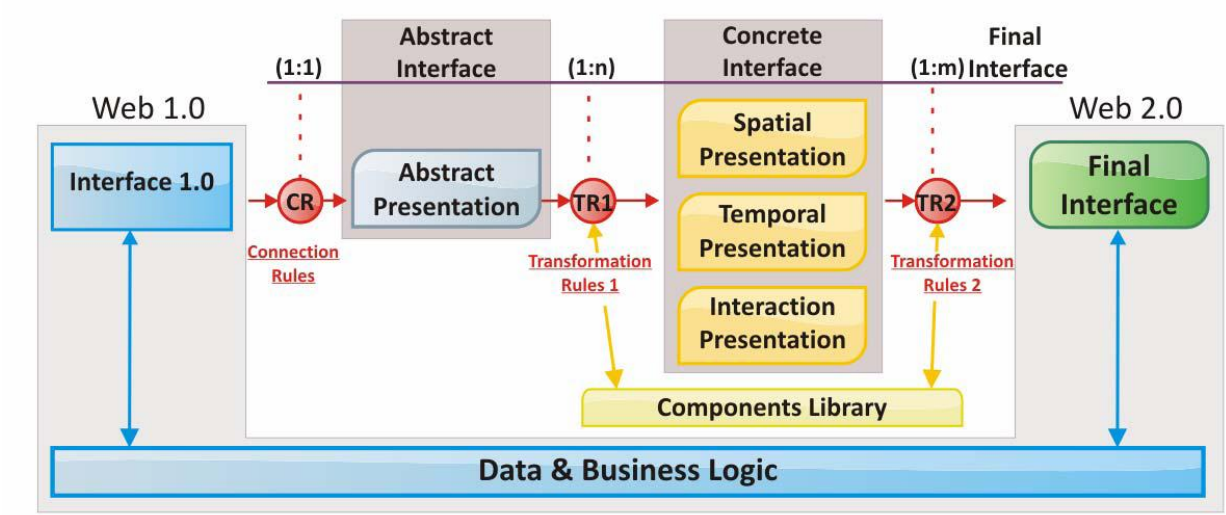


Figura 3.16. Descripción de la arquitectura del método RUX.

El método RUX distingue tres niveles diferentes de interfaz los cuales proporcionan una cadena conceptual de refinamiento: Interfaz Abstracta, Interfaz Concreta y la Interfaz Final. En primera instancia el método tiene como entrada la interfaz Web 1.0, en nuestro caso es la interfaz desarrollada con la metodología UWE en la capa de presentación. La interfaz abstracta proporciona una representación de UI común para todas las plataformas del desarrollo RIA, sin ninguna clase de presentación espacial, apariencia y comportamiento de las dependencias. La interfaz concreta es independiente de la plataforma de desarrollo pero específica para las características del tipo de aplicación que se quiere llevar a cabo. Se divide en tres niveles de presentación: presentación espacial, temporal y de interacción. Puesto que la capa abstracta brinda una primera vista de la distribución espacial de la interfaz, en la presentación espacial simplemente se necesita perfeccionar esta distribución, especificar la disposición espacial de los componentes, y definir dimensiones y apariencias. La presentación temporal permite la especificación de la conducta de tareas que requieren una sincronización temporal (p.e. animaciones). La presentación de interacción permite la especificación de la conducta del usuario con la interfaz de usuario RIA, en las aplicaciones RIA la interacción del usuario con la UI es capturada generalmente por los componentes de la aplicación que son capaces de capturar cierta clase de eventos. La interfaz final contiene la información para la generación de la UI de la aplicación a desarrollar de acuerdo a una determinada plataforma de desarrollo RIA tales como Flex, AJAX o Laszlo [Preciado08].

De acuerdo a los tres niveles de interfaz, también hay tres fases de transformación en el método RUX. En la primera fase de transformación se capturan y adaptan los datos y la capa lógica especificados en el modelado de la herramienta Web (en nuestro caso el diseño mediante la metodología UWE) a la interfaz abstracta del método RUX, y se le llaman reglas de conexión (marcada como CR en la Figura 3.16). La interfaz abstracta se adapta en la segunda fase de la transformación a uno o más dispositivos particulares y concede el acceso a la lógica de negocios de la aplicación. A esta fase se le llama reglas de transformación 1 (marcadas como TR1 en la Figura 3.16). Finalmente, en las reglas de transformación 2 (marcadas como TR2 en la Figura 3.16) el ciclo de vida de la Arquitectura Dirigida por Modelos (MDA, *Modelo Driven Architecture*) del método RUX es completada mediante la generación de código.

Una vez que la interfaz abstracta fue obtenida aplicando las CRs al modelo Web 1.0, pueden hacerse nuevos refinamientos hacia la interfaz concreta y la interfaz final, para finalmente generar

una interfaz RIA que proporcione las mismas funcionalidades, pero que brinde una interfaz de usuario más amigable y de uso mucho más fácil [Preciado08].

3.3.4.2.1. **Diseño de interfaz abstracta**

La interfaz abstracta del método RUX esta compuesta por tres diferentes tipos de elementos: *Connectors, Media y Views*.

- ***Connectors*** se utilizan para establecer la relación entre componentes de la UI y los datos representados en el modelo conceptual;
- ***Media*** son los elementos de información atómica que son independientes de la tecnología de representación del cliente. Los elementos *media* se clasifican en discretos (textos e imágenes) y continuos (video, audio y animación), cada uno con sus propios atributos. Cada elemento *media* también apoya procesos de entrada-salida ya que las RIAs ayudan estas clases de operaciones.
- ***Views*** son utilizados para agrupar información que el usuario observará al mismo tiempo. Para agrupar información, el modelo RUX ofrece cuatro tipos distintos de *Views* (simple, alternativa, duplicada y jerárquica).

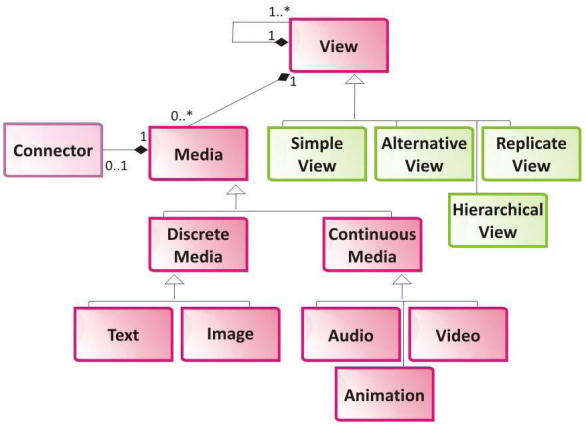


Figura 3.17. Metamodelo de la interfaz abstracta del método RUX.

En el método RUX, el elemento raíz de la interfaz abstracta es siempre un *View*. Cuando el tipo de *View* es simple puede contener otros *Views* y/o elementos *Media*. Las *Views* alternativa, duplicada y jerárquica no pueden contener elementos *Media* por si solos, deben de estar contenidas en una *View* simple. Una *View* alternativa indica que solo una de las *View* que contiene se muestra al mismo tiempo al usuario (por ejemplo, los paneles por pestaña). Las *View* duplicadas determinan *Views* que se van a repetir en toda la *View* que las contiene (por ejemplo, una lista de elementos). Las jerárquicas representan elementos en una vista de árbol (por ejemplo, un conjunto de categorías y de subcategorías). El metamodelo de la interfaz abstracta del método RUX se muestra en la Figura 3.17.

La metodología presentada en [Preciado08] presenta un algoritmo para generar la interfaz abstracta (véase Figura 3.18). Este algoritmo esta organizado en dos pasos. El primer paso crea una interfaz abstracta vacía. En el segundo, cada <<presentation class>> del modelo de presentación UWE y los elementos UI que contiene cada una de estas clases, son agregados como nuevos *Views* a la *SimpleView* raíz.

Start
Create a SimpleView as the root element.
For each p, p is of type «presentation class», «text» or «image» contained in the presentation model
    Create a **SimpleView V** inside the last view we have created from the **p.parent**, or inside the **root** if p has no parents.
    If **p** is a «presentation» node:
        Create a **Connector** that references to **p** inside **V**
        If **p** has relationships with a multiplicity of..* (unbounded) at the children role
            Create a **Replicate View R** inside **V**, in order to use the **R** as the parent for the results of applying CR to **p**'s children
    ElseIf **p** is a «text», «image», «anchor», etc. node:
        Create a **Media** inside **V**.
        Connect the **Media** with the **Connector** created from **p.parent**.
        Select the type of **Media** conveniently (direct since UWE and the RUX-Method support the same types of media).
        Specifically indicate which attribute of the **Connector** use the **Media**. This can not be inferred so it is required because there is only *one* Connector for every UWE <<presentation>> node, and it has usually more than one attribute.
    EndIf
End

Figura 3.18. Pseudocódigo del proceso de creación de la interfaz abstracta a partir del modelo de presentación UWE.

3.3.4.2.2. **Diseño de Interfaz Concreta**

Las CRs además de construir la interfaz abstracta también recuperan y distribuyen a la interfaz concreta la información requerida para crear una Lista de Enlaces (LUL, *Lis of Useful Links*) de acuerdo a los diferentes tipos de acciones definidos por el modelo RUX (por ejemplo, UIActios o CallActios)[Linaje07]. Debido a que UWE modela aplicaciones Web 1.0, los elementos contenidos en el LUL son todos CallActios, los cuales modelan un tipo de interacción simple, como un click del ratón para seguir un link de hipertexto. Una LUL consiste en los <<navigation link>> y <<process link>> del modelo de navegación UWE.

Para aplicar las TR1 y conseguir la interfaz concreta a partir de la interfaz abstracta, el método RUX debe de tener en cuenta la información ofrecida por los modelos UWE en cuanto a las cadenas de operación. El algoritmo para llevar a cabo esta tarea se presenta en la Figura 3.19.

For each LINK ("L") listed in **LUL**
    Create a handler with name "L"
    Add a CallAction to reference "L"
For each Media "M" in the abstract interface:
    Add a listener for every output "M" (RUX-Method Media elements can be for input purposes e.g. combobox or for output e.g. label), called **O**
    For each listener **O**
        Set the handler descriptor to be the same as created before.
        Connect with the first event defined in the component library (default event) for the component (usually, **click**)

Figura 3.19. Algoritmo para representar las cadenas de navegación de los modelos UWE en la interfaz concreta de método RUX.

En [Preciado08a] se dan las bases para las TR1, en el se describen los componentes básicos que deben de ser utilizadas para el mapeo de elementos entre la interfaz abstracta y la interfaz concreta del método RUX. En la Figura 3.20 se muestra el origen: los elementos de la interfaz abstracta (lado izquierdo) y el destino: los elementos de la interfaz concreta (lado derecho); y la relación de mapeo que existe entre ellos.

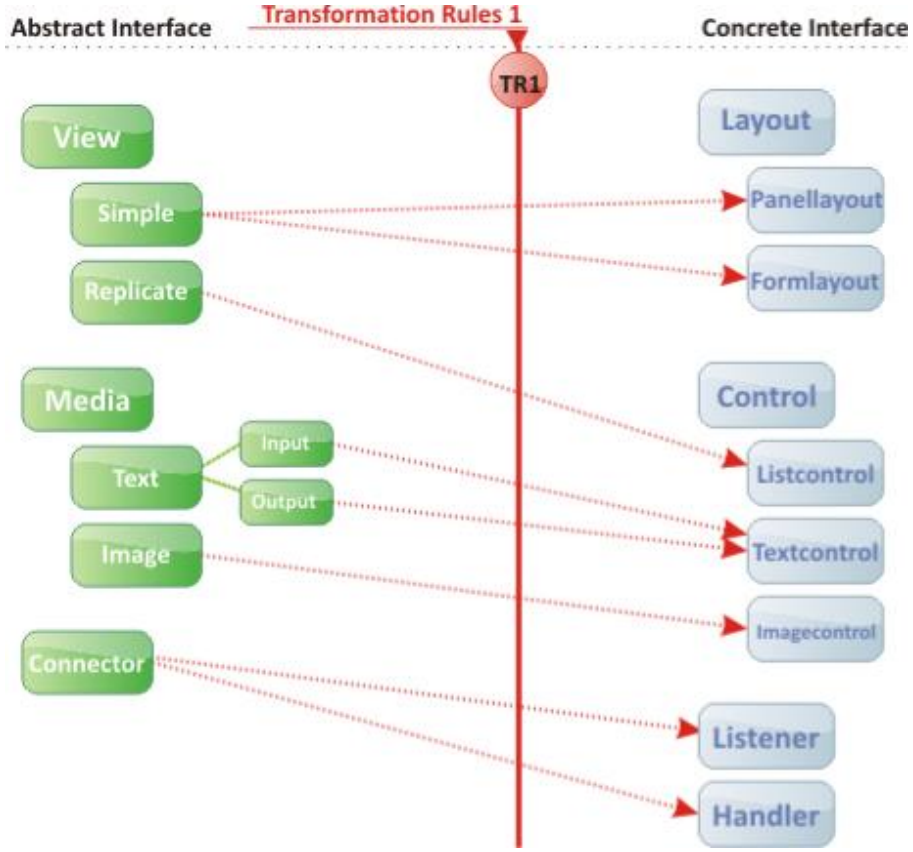


Figura 3.20. Mapeo entre los elementos de la interfaz abstracta y la interfaz concreta.

A continuación se presenta el ciclo de diseño de la interfaz RIA mediante el método RUX (véase Figura 3.21), el caso de diseño que se presenta es para la etapa de evaluación por cuestionario. En primera instancia se presenta el modelo de presentación UWE para el cuestionario (izquierda), en segunda instancia se presenta la interfaz abstracta (centro), y por ultimo la interfaz concreta para la evaluación por cuestionario (derecha). Con el uso de las reglas propuestas por la metodología se obtuvo la interfaz concreta, y después de unos ajustes manuales se obtuvo el resultado presentado en la Figura 3.21.

La última etapa del método RUX se realizo manualmente ya que la herramienta propuesta por [Preciado08] solo genera código para los lenguajes RIA AJAX y Laszlo, por lo que el diseño de la interfaz final se tuvo que realizar de forma manual, ya que el lenguaje RIA a utilizarse en la implementación no corresponde a ninguno de los propuestos por la herramienta.

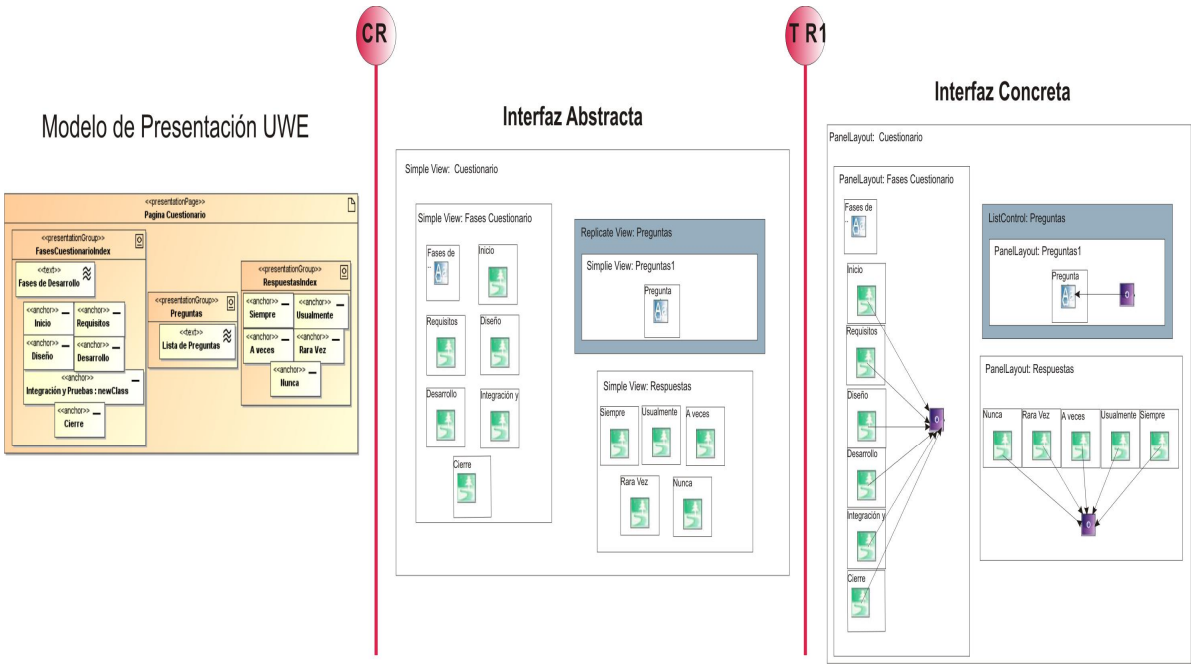


Figura 3.21. Proceso de Diseño de Interfaz de Usuario para la evaluación por cuestionario mediante el método RUX.

3.3.5. Implementación de SelfVation

La implementación es el flujo de trabajo donde se convierte el resultado de diseño en código fuente y en una aplicación funcional.

Terminada la etapa de diseño, se tiene todo listo para la creación de un sistema software funcional. Basándose en la arquitectura del diseño obtenido en la etapa anterior se logrará la implementación de la herramienta SelfVation en archivos de código fuente ActionScript y MXML (*Multimedia eXtensible Markup Language*) correspondientes al lenguaje de desarrollo Adobe Flex, y archivos PHP (*PHP Hypertext Pre-processor*) para generar el puente entre SelfVation y las consultas a la base de datos, y archivos HTML y SWF que permiten la publicación y posterior uso en un sitio Web.

En esta fase se obtienen los componentes que constituyen a SelfVation. Un componente es la parte física de los elementos del diseño que son las clases, páginas, y/o interfaces. Un componente puede ser un archivo de código fuente, un archivo ejecutable, una tabla de base de datos u otro documento.

La primera fase de la implementación se llevará a cabo mediante el desarrollo de la herramienta en forma de aplicación de escritorio, con el fin de realizar el desarrollo de forma más ágil, ya que en este tipo de aplicaciones no se necesitan subir o actualizar archivos en un servidor Web para probar la herramienta, sino que se trabaja en forma local, de esta manera la base de datos a ocupar se implementará de forma local. La segunda parte de la implementación se llevará a cabo cuando la herramienta se encuentre terminada en forma de aplicación de escritorio, ya que se migrará dicha aplicación a una aplicación Web debido a que la herramienta de desarrollo lo permite de una forma fácil y confiable, creación de la base de datos en el servidor remoto, y por último la publicación de SelfVation en un sitio Web.

3.3.5.1. Implementación de la arquitectura

La primera parte de la implementación es construir el modelo conceptual de la herramienta, en primera instancia algunos de los elementos que comprenden a este modelo debieron construirse en base a tablas de una base de datos (BD) que sirvan para almacenar los datos recabados por la herramienta y a su vez almacenar los datos necesarios para llevar a cabo funcionalidades de SelfVation. Es por estas razones que la primera parte de la implementación es la creación de la BD que estará ligada a SelfVation. Esta BD fue creada a partir del sistema de administración de base de datos MySQL, debido a facilidad en el uso de este gestor, y a la compatibilidad de trabajo con el lenguaje Flex mediante la generación de conexiones y consultas a partir de archivos PHP.

La siguiente parte de la implementación consistió en identificar la complejidad de los componentes (mostrados en la Figura 3.13) más significativos de la herramienta, para comenzar su implementación, ya que la implementación de algunos componentes es menos complicada que otras, como en el caso de "Resultados Diagrama" que sugiere ser más complicado que el resto de los componentes. En base a la identificación de complejidad y dependencia entre componentes el orden de implementación de los componentes fue el siguiente:

1. Login.
2. Alta Empresa.
3. Alta Usuario.
4. Cuestionario.
5. Crear Diagrama.
6. Borrar Diagrama.
7. Guardar Diagrama.
8. Resultados Diagrama.
9. Resultados Cuestionario.
10. Obtener Nivel Madurez.
11. Generar Actividades y Recomendaciones.
12. Obtener Productos.
13. Obtener Calificación Fases Desarrollo.
14. Plan de Acción.

Los componentes enlistados son aquellos que llevan a cabo algún tipo de proceso como consultas a la BD, implementación de algoritmos o ambos, por parte del programador. Se hace énfasis en estos componentes, ya que son en los que se necesita llevar un proceso de implementación más complicado por parte del desarrollador. Todos estos componentes están relacionados con componentes de UI como se puede observar en el modelo de navegación mostrado en la Figura 3.13, debido a que son llamados por medio de eventos en la UI o sus resultados son mostrados en componentes UI. La implementación de este tipo de componentes es omitida en este apartado, ya que la herramienta de desarrollo cuenta con la característica de generación de código en base al diseño de la UI.

A continuación se aborda la implementación de los componentes más complejos y con mayor grado de programación. En la descripción de la implementación se presenta un contexto de trabajo del componente, así como un pseudocódigo de cómo realiza sus funciones (El código fuente detallado de estas implementaciones se encuentra en el Anexo C de este documento). Se omiten aquellos componentes que basan su complejidad en consultas y actualizaciones a la BD.

3.3.5.2. Componentes fundamentales de SelfVation

3.3.5.2.1. **Componente "*Crear Diagrama*"**

El componente "Crear Diagrama" es el encargado de llevar a cabo la creación del modelado del proceso de desarrollo actual de la empresa. Este componente esta asociado a la UI que contiene los elementos y el área de dibujo para crear este tipo de diagrama. El contexto de trabajo de este componente es dependiente de los "eventos" que se realicen en el área de dibujo contenido en la UI. Así pues, cuando un elemento es arrastrado al área de dibujo el componente "Crear Diagrama" crea y visualiza este nuevo elemento, cuando se intenta realizar una conexión entre dos elementos, "Crear Diagrama" es el encargado de visualizar y crear la flecha que indica el vínculo entre dos elementos. En otras palabras, el componente "Crear Diagrama" es el encargado de crear, destruir, vincular y editar los elementos que componen un diagrama de proceso encargado de modelar el actual estado del proceso de desarrollo de la empresa. La detección de eventos con los que cuenta Flex hace la implementación un poco más sencilla.

Puesto que la creación de nuevos elementos solo tiene como parte compleja el crear un nuevo objeto del tipo de elemento que se elige, posicionando el nuevo símbolo en las coordenadas que el usuario deseé; no se explicará la implementación de este tipo de tarea. El método más complejo en este componente es el encargado de crear las líneas de conexión entre elementos. A continuación se presenta el pseudocódigo del algoritmo que se implementó para llevar a cabo esta tarea (el código fuente de la implementación se encuentra detallado en el Anexo C).

```
Start
If el evento "mouseDown" se realiza dentro del área de un elemento E (inicio, caja, diamante, persona, final) y está seleccionada la opción de crear línea:
        Crear línea temporal LT con las coordenadas de inicio y final  x, y correspondientes al instante del evento "mouseDown"
        While no se produzca el evento "mouseUp":
                If se detecta un evento "mouseMove"
                        Actualizar coordenadas x, y finales de LT con la posición del mouse.
                EndIf
        EndWhile
EndIf
If  el evento "mouseUp" se realiza dentro del área de un elemento E2 (inicio, caja, diamante, persona, final) y E es diferente de E2:
        If E es diferente de "final" && E2 es diferente de "inicio"
                Crear Línea L
                Calcular coordenadas de inicio y final para L según la posición y características de E y E2
                L.elementoOrigen=E
                L.elementoDestino=E2
      EndIf
EndIf
End
```

3.3.5.2.2. **Componente *"Resultados Diagrama"***

Este componente es el encargado de realizar el mapeo entre el diagrama actual de la empresa con el diagrama ideal propuesto por el modelo Moprosoft (definido en la sección 3.3.1.2). El flujo de trabajo de este componente es iniciado por medio del evento "click" en el botón de la UI asignado para comenzar a analizar el diagrama del proceso de desarrollo actual de la empresa. El algoritmo para realizar el mapeo y la creación de un diagrama enriquecido se muestra a continuación. Como precondiciones para llevar a cabo el algoritmo se tienen que realizar consultas a la BD para obtener los elementos del diagrama del proceso actual y los elementos del diagrama ideal Moprosoft, ambos elementos serán almacenados en **tablaDiagrama** y **tablaDiagramaMoprosoft** respectivamente, ambas tablas serán utilizadas en el siguiente algoritmo, además de una nueva tabla que almacenará los elementos del diagrama enriquecido **tablaDiagramaNuevo**. El algoritmo principal se auxilia de un método secundario llamado insertaProcesosMoprosoft que se describe enseguida del algoritmo principal. El algoritmo presenta la idea en general para su desarrollo para mayor detalles consultar el código fuente en el Anexo C.

```
Start
For each E, E es un elemento caja, diamante, persona, inico, final contenido en  tablaDiagrama
        Emopro es el elemento que corresponde a E.procesoRelacionado en tablaDiagramaMoprosoft
        If E.procesoOrigen.procesoRelacionado= Emopro.procesoOrigen.proceso ó E.procesoRelacionado =
        E.procesoOrigen.procesoRelacionado
                Se agrega el elemento E a tablaDiagramaNuevo denotando que es una actividad correcta.
                Se actualiza procesoOrigen y procesoDestino de E con los elementos contenidos en tablaDiagramaNuevo.
      ElseIf
                Se agrega el elemento E a tablaDiagramaNuevo denotando que es una actividad incorrrecta.
                Se agregan los procesos Moprososft faltantes por medio del método insertaProcesosMoprosoft.
                Se actualizan procesoOrigen y procesoDestino del primer elemento insertado  con el elemento E contenido
                en tablaDiagramaNuevo.
      EndIf
EndFor
End
Función insertaProcesosMoprosoft (E.procesoRelacionado, E.procesoOrigen.procesoRelacionado)
Start
//E.procesoRelacionado es el proceso Moprosoft del elemento actual  y E.procesoOrigen.procesoRelacionado es el //proceso
Moprosoft del proceso Origen del elemento actual, en el diagrama del proceso actual de la empresa
If  E.procesoRelacionado<E.procesoOrigen.procesoRelacionado:
        //El proceso actual debe de ir antes que el proceso Origen
        Generar mensaje de error debido a traslapación de procesos.
ElseIf
        //Se insertarán nuevos procesos al diagrama enriquecido
        Generar mensaje de error debido a falta de procesos.
        Emopro es el elemento que corresponde a E.procesoOrigen.procesoRelacionado en tablaDiagramaMoprosoft
        Do
                Se agrega el elemento Emopro a tablaDiagramaNuevo denotando que es una actividad nueva.
                Se actualizan procesoOrigen y procesoDestino del elemento Emopro.
        While Emopro.proceso /= E.procesoRelacionado;
EndIf
End
```

3.3.5.3. Pruebas Unitarias

El objetivo de esta actividad fue verificar el buen funcionamiento de cada componente desarrollado. Para revisar la funcionalidad se desarrollaron casos de pruebas para cada uno de los componentes desarrollados. La Tabla 18 muestra las pruebas funcionales realizadas a cada componente. En un principio, para llevar a cabo estas pruebas se tuvo que dar de alta un usuario en la BD de forma manual con el fin de probar el componente "Login".

Tabla 18. Pruebas funcionales realizadas a cada componente de SelfVation.

Componente	Pruebas	Resultado Esperado
Login	Intentar iniciar sesión con un nombre de usuario y password correctos.	Mostrar mensaje de que el nombre de usuario y password son correctos, y que se puede iniciar sesión.
	Intentar iniciar sesión con el nombre de usuario y password incorrectos.	Mostrar mensaje de que no se puede iniciar sesión.
AltaEmpresa	Intentar dar de alta una empresa de nombre inexistente en la BD.	Inserción correcta de la nueva empresa en la BD.
	Intentar dar de alta una empresa de nombre existente en la BD.	Mostrar mensaje de que la empresa ya existe en la BD y no realizar ningún tipo de inserción en la BD.
AltaUsuario	Intentar dar de alta un nuevo usuario con un nombre de usuario inexistente en la BD.	Inserción correcta del nuevo usuario en la BD.
	Intentar dar de alta un nuevo usuario con un nombre de usuario que existe en la BD.	Mostrar mensaje que el nombre de usuario ya ha sido utilizado y no realizar ningún tipo de inserción en la BD.
Cuestionario	Responder una pregunta con cualquiera de las respuestas posibles.	Inserción a la BD de la respuesta a la pregunta.
	Elegir una nueva fase del cuestionario.	Mostrar pregunta correspondiente a la nueva fase elegida.
	Responder todo el cuestionario correspondiente a una fase.	Mostrar mensaje de que a evaluación para esa fase ha terminado.
	Elegir una nueva fase del cuestionario.	Mostrar pregunta correspondiente a la nueva fase elegida.
Crear Diagrama	Inserta un nuevo elemento en el área de dibujo.	Mostrar el nuevo elemento en el área de dibujo.
	Seleccionar un elemento determinado.	Marcar el elemento seleccionado de una forma diferente de los elementos no seleccionados.
	Seleccionar una conexión determinada.	Marcar la conexión seleccionado de una forma diferente de los elementos no seleccionados.
	Realizar una conexión entre dos elementos.	Mostrar la línea que denota la conexión entre dos elementos
	Mover un determinado símbolo a una nueva posición en e área de dibujo.	Mover el elemento a una nueva posición dentro del área de dibujo, en caso a que este elemento esté asociado a una o más conexiones desplazar estas conexiones a la nueva posición del elemento.
	Eliminar un elemento.	Borrar el elemento del área de dibujo, en caso de tener asociada alguna conexión eliminar la línea que denota esta conexión.
	Eliminar una conexión.	Borrar la conexión del área de dibujo.
	Realizar una conexión de cualquier elemento hacia un elemento "Inicio"	Invalidar la conexión, de forma que esta conexión no se muestre en el área de dibujo.
	Realizar una conexión de un elemento "Final" hacia cualquier elemento.	Invalidar la conexión, de forma que esta conexión no se muestre en el área de dibujo.

	Realizar una conexión de un elemento "Persona" hacia un elemento que no sea tipo "Caja".	Invalidar la conexión, de forma que esta conexión no se muestre en el área de dibujo.
	Realizar una conexión de un elemento "Diamante" hacia un elemento que no sea tipo "Caja".	Invalidar la conexión, de forma que esta conexión no se muestre en el área de dibujo.
BorrarDiagrama	Borrar el diagrama editado mediante el botón destinado para este fin.	Eliminar todos los elementos y conexiones mostrados en el área de dibujo.
Guardar Diagrama	Guardar diagrama en el que no existen elementos desconectados o elementos fuera de la secuencia del diagrama.	Inserción a la BD del diagrama editado.
	Guardar diagrama en el que existen elementos desconectados o elementos fuera de la secuencia del diagrama.	Mostrar mensaje de que existen elementos desconectados de la secuencia del diagrama.
ResultadosDiagrama	Iniciar proceso de mapeo del diagrama editado con el diagrama ideal Moprosoft.	Mostrar en el área de dibujo el diagrama enriquecido obtenido a partir del proceso de mapeo, además de obtener los errores en forma de texto encontrados durante este proceso.
ResultadosCuestionario	Consultar las fase de resultados de la evaluación	Mostrar gráficas que muestren los resultados del cuestionario en cada una de las fases evaluadas.
Obtener Nivel Madurez	Consultar las fase de resultados de la evaluación	Mostrar imagen que denote el nivel de madurez obtenido por la empresa.
Generar Actividades y Recomendaciones	Consultar el plan de acción.	Mostrar lista de actividades y recomendaciones, en las cuales la empresa debe poner énfasis en realizar.
Obtener Productos	Consultar el plan de acción.	Mostrar lista de productos, que la empresa no esta generando o se están generando de forma débil.
Obtener Calificación Fases de Desarrollo	Consultar el plan de acción.	Mostrar calificación obtenida en cada una de las fases de Desarrollo.

Cabe mencionar que para los componentes más complejos fue necesaria más de una iteración de estas pruebas, debido a que los resultados no fueron satisfactorios en la primera iteración. Debido a esto se dio por culminada la construcción de un componente hasta que cumpliera satisfactoriamente con las pruebas funcionales desarrolladas.

3.3.6. Integración y Pruebas de SelfVation

Al término del desarrollo de cada componente y cumpliendo con las funcionalidades requeridas se procedieron a la integración de estos en un solo sistema, que cumpliera en conjunto con todas la funcionalidades propuestas para la herramienta SelfVation. Con el fin de probar la funcionalidad del sistema SelfVation se desarrolló un sistema de prueba. Como sistema de prueba se consideró crear simulaciones de evaluación que abarcaran todas las funcionalidades de la herramienta. A continuación se muestra uno de los sistemas de prueba desarrollados, el sistema se basa en los pasos a seguir en una evaluación real. Los resultados de este sistema de prueba se presentan en el Anexo D de este documento.

El sistema consiste en lo siguiente:

- Iniciar sesión como administrador del sistema.
- Dar de alta una nueva empresa.
- Dar de alta usuario, que pertenezca a la empresa dada de alta en el paso anterior.
- Iniciar sesión como el nuevo usuario del paso anterior.

- Modelar un diagrama de procesos que carezca de actividades relacionados con los procesos Moprosoft “Análisis y Diseño” y “Cierre
- Contestar la fase de evaluación por cuestionario de la siguiente manera:
  - La fase de “Inicio” contestarla con las respuestas “Rara Vez” y “Algunas Veces”.
  - La fase de “Requisitos” contestarla con las respuestas “Usualmente” y “Siempre”
  - La fase de “Análisis y Diseño” contestarla con las respuestas “Algunas Veces”
  - La fase de “Construcción” contestarla con las respuestas “Siempre”.
  - La fase de “Integración y Pruebas” contestarla con las respuestas “Usualmente”.
  - La fase de “Cierre” contestarla con las respuestas “Nunca”.
- Generar el diagrama enriquecido propuesta por SelfVation.
- Consultar la fase de resultados.
- Consultar el plan de acción generado por SelfVation.

4. Resultados Experimentales

Dentro de un marco de trabajo que tenía como principal objetivo obtener datos que contribuyan a una aplicación exitosa de la herramienta SelfVation, se desarrolló un proyecto de investigación. Las tareas principales de la metodología para este proyecto fueron:

1. Identificar MPyMEs desarrolladoras de software que deseen asumir un compromiso de mejora.
2. Utilizar la herramienta SelfVation para evaluar la situación actual de la organización.
3. Utilizar a herramienta SelfVation para mejorar las debilidades encontradas durante el paso 2.
4. Implementar el conocimiento adquirido a través de la herramienta en los procesos claves.
5. Ejecución y evaluación oficial de cumplimiento con el nivel de capacidad Moprosoft.

Con el fin de elegir las MPyMEs desarrolladoras de software correctas, se decidió experimentar con tres tipos de empresas: el primer tipo, aquellas que conozcan y utilicen el estándar, el segundo tipo, aquellas que solamente conozcan el estándar y por último aquellas que no conozcan el estándar. Dado que nuestro principal objetivo es reducir el tiempo de esfuerzo para adoptar el modelo Moprosoft, uno de los principales factores de SelfVation es que proporciona la misma guía de acción tanto para aquellas empresas que tienen implementado el modelo dentro de su organización como en las que no tienen conocimiento previo del modelo. La Tabla 19 muestra las características de las MPyMEs desarrolladoras de software participantes en el estudio.

Tabla 19. Perfil de la MPyMEs participantes.

MPyME	Rubro	Tamaño (No de Empleados)	Experiencia
1	Mantenimiento y Desarrollo de Software	10	El modelo esta implementado
2	Mantenimiento y Desarrollo de Software	20	El modelo es conocido, pero no está implementado
3	Desarrollo de soluciones software para pequeñas empresas	15	Desconoce el modelo
4	Servicios de Hardware & Software y soluciones integrales	8	El modelo es conocido, pero no está implementado

Un caso de estudio se utiliza habitualmente como una estrategia de investigación empírica en el campo de los sistemas de información, a menudo es utilizado para describir las relaciones dentro de la organización [Galliers92]. En el contexto de este trabajo, se utilizó un caso de estudio como un método para proporcionar un contexto organizacional para la aplicación de SelfVation en la identificación de factores SPI y posteriormente su aplicación y evolución dentro de cuatro pequeñas empresas software. Debe mencionarse que para este caso de estudio se tenían consideradas otras empresas, pero ante la falta de compromiso para la aplicación de las evaluaciones, se optó por dejarlas fuera del caso de estudio mostrado a continuación.

4.1. Caso de Estudio: Adopción de iniciativas SPI en cuatro pequeñas empresas software mexicanas

Para llevar a cabo el caso de estudio para este trabajo se diseñó un experimento controlado con cuatro pequeñas empresas en el que los esfuerzos se enfocaron en una versión semiformal del enfoque Evalprosoft. Un equipo evaluador y cuatro jefes de proyectos fueron elegidos, además se dividieron las cuatro organizaciones dentro de tres categorías de acuerdo a su experiencia con el modelo. Los jefes de proyecto elegidos son profesionales que conocen la cultura de la empresa y la forma en que se gestionan los proyectos de desarrollo. Estos jefes de proyecto recibieron una charla introductoria de Moprosoft. De manera similar recibieron una plática acerca de las categorías a evaluar por la herramienta.

Los mecanismos de evaluación desarrollados se pusieron a disposición de los jefes de proyecto por medio de la herramienta SelfVation mediante un portal Web, esto con la finalidad de que los evaluados pudiesen acceder independientemente del lugar en que se encontraran, además de reducir la complicación que significa realizar evaluaciones por medios escritos; y guardar el historial de las evaluaciones que se fueron realizando para determinar las mejoras que se fueron presentando durante el transcurso de la evaluación.

A continuación, se presentan los resultados que la herramienta SelfVation generó a partir de la evaluación de las cuatro empresas que colaboraron en este caso de estudio. Los resultados que se presentan son los referentes a la evaluación por modelado, en primera instancia, en donde se muestra el diagrama que la empresa editó para describir el proceso actual de desarrollo que llevan a cabo y el diagrama enriquecido que la herramienta SelfVation generó a partir del mecanismo implementado para esta tarea, la segunda parte de los resultados presentados se refiere al plan de mejora qué obtiene la herramienta a partir del trabajo conjunto de las evaluaciones por modelado y por cuestionario.

Las imágenes que muestran ambos resultados, han sido editadas manualmente para su mejor presentación, ya que de esta manera se muestra el diagrama enriquecido completo en el caso de los resultados de evaluación por modelado; y la totalidad de las actividades y productos generados en el plan de acción, así como el nivel de madurez, resultados y calificación por fase, para cada empresa evaluada.

Los resultados por modelado muestran el diagrama del proceso actual de la empresa y el diagrama enriquecido propuesto por SelfVation, lado izquierdo y derecho de las figuras respectivamente. En el caso de los planes de acción se presentan la totalidad de las actividades recomendadas para llevar a cabo una iniciativa de mejora, además de los productos que la empresa no está generando de manera correcta, en esta figuras se presentan además, el nivel de madurez obtenido por cada empresa, los resultados en las fases evaluados y las calificaciones obtenidas en cada una de estas fases.

En la Figura 4.1 se presenta el resultado obtenido en la evaluación por modelado, por parte de la MPyME1. En la figura se puede observar como en el proceso actual de esta empresa se llevan a cabo actividades que cubren todas las fases de desarrollo propuestas por Moprosoft, pero se detectaron algunos errores de falta de secuencia y carencia de actividades de verificación y validación en las fases de “Análisis y Diseño”, “Construcción” e “Integración y Pruebas”.

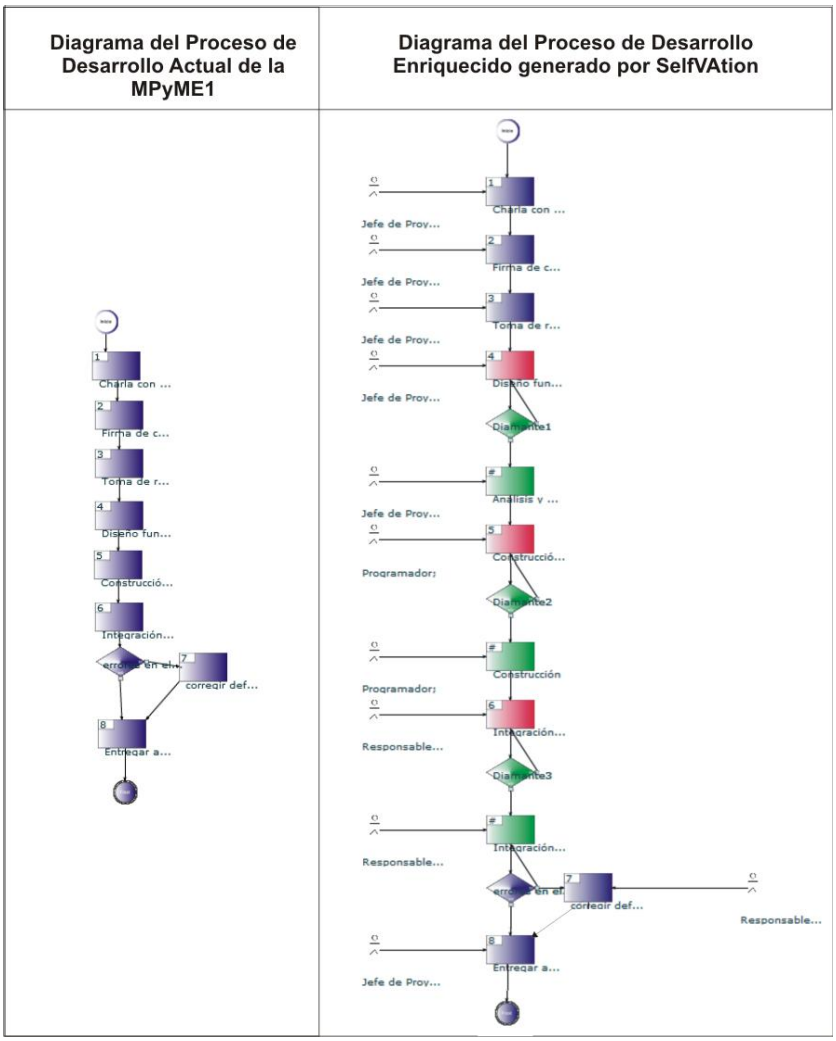
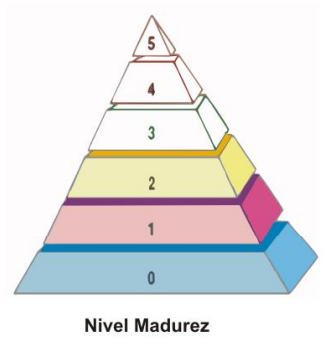


Figura 4.1. Resultados de la evaluación por modelado para la MPyME1.

En la Figura 4.2 se muestra el plan de mejora obtenido por la herramienta SelfVation para la MPyME1, en este plan se confirma las características de la empresa, ya que los resultados obtenidos son satisfactorios en base al nivel Moprosoft que se tiene implementado en la organización. Como se puede observar en las calificaciones por fases (lado izquierdo-superior de la figura), las primeras dos fases de desarrollo obtuvieron una calificación buena, las calificaciones de las fases siguientes son promedio, en parte debido a las penalizaciones en algunas de ellas por los errores detectados en la evaluación por modelado.



Recomendaciones y Actividades:

Inicio

-Definir un Proceso Específico con base en la Descripción del Proyecto y el proceso de Desarrollo y Mantenimiento de Software de la organización o con base en el acuerdo con el Cliente
-Realizar un cálculo del Costo Estimado del proyecto, considerando las Metas Cuantitativas para el Proyecto
-Elaborar un plan de Adquisiciones y Capacitación para llevar a cabo el proyecto

Requisitos

-Dar seguimiento a Plan de Adquisiciones y Capacitación
-Llevar a cabo una administración de los subcontratistas
-Verificar el Plan de Pruebas del Sistema, generando un Reporte de Verificación
-Generar Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente

Análisis y Diseño

-Distribuir las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Elaborar un Plan de Pruebas de Integración
-Elaborar un Registro de Rastreo
-Verificar el Plan de Pruebas de Integración y generar un Reporte de Verificación

Construcción

-Definir y aplicar pruebas unitarias para verificar que el funcionamiento de cada componente esté acorde con el Análisis y Diseño
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Elaborar un Reporte de Actividades correspondientes a esta fase
-Actualizar el Registro de Rastreo, incorporando los componentes construidos o modificados

Integración y Pruebas

-Establecer Acciones Correctivas de acuerdo a la evaluación de los Planes de Proyecto y Desarrollo
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Identificar nuevos riesgos y actualizan el Plan de Manejo de Riesgos
-Actualizar el Registro de Rastreo, incorporando los subsistemas o el sistema Software
-Documentar los resultados de las pruebas en un Reporte de Pruebas de Integración
-Elaborar un Manual de Operación
-Realizar pruebas de sistema siguiendo el Plan de Pruebas del Sistema, documentando los resultados en un Reporte de Pruebas de Sistema
-Verificar el Manual de Usuario y generar un Reporte de Verificación
-Incorporar el Software, Reporte de Pruebas de Integración, Registro de Rastreo, Manual de Operación y Manual de Usuario como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Cierre

-Incorporar el Manual de Mantenimiento como línea base a la Configuración de Software
-Realizar cierre de actividades con subcontratistas de acuerdo al contrato establecido
-Realizar un Documento de Aceptación
-Verificar el Manual de Mantenimiento y generar un Reporte de Verificación
-Identificar y Documentan las Lecciones Aprendidas de este proceso

Productos:

Inicio

-Plan de Adquisiciones y Capacitación

Requisitos

-Reporte de Verificación para Plan de Pruebas del Sistema
-Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente

Análisis y Diseño

-Plan de Pruebas de Integración
-Registro de Rastreo
-Reporte de Verificación de Plan de Pruebas de Integración

Construcción

-Reporte de Verificación de Registro de Rastreo
-Reporte de Actividades correspondientes a fase de Construcción

Integración y Pruebas

-Resultados de las pruebas en un Reporte de Pruebas de Integración
-Manual de Operación
-Reporte de Pruebas de Sistema
-Reporte de Verificación de Manual de Usuario
-Reporte de Actividades correspondiente a fase Integración y Pruebas

Cierre

-Documento de Aceptación
-Reporte de Verificación de Manual de Mantenimiento
-Lecciones Aprendidas

Figura 4.2. Plan de acción generado por SelfVation para la MPyME1.

La Figura 4.3, es la concerniente a los resultados por modelado de la MPyME2, en la imagen se puede observar que la empresa lleva a cabo actividades afines con todas las fases de desarrollo

propuestas por Moprosoft. Sin embargo, se detectaron inconsistencias en las etapas de “Análisis y Diseño”, “Integración y Pruebas” y “Cierre”.

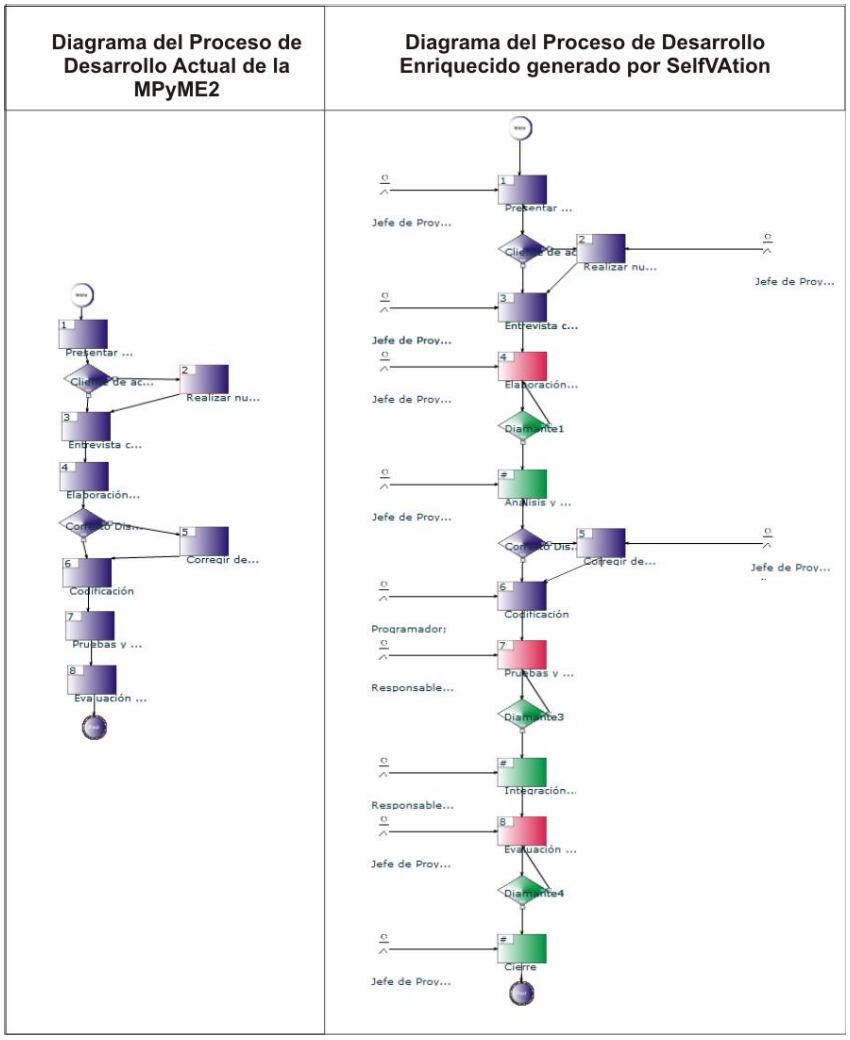


Figura 4.3. Resultados de la evaluación por modelado para la MPyME2.

La Figura 4.4, presenta el plan de acción generado por SelfVation en base a los resultados de la evaluación para la empresa número dos. Al igual que las calificaciones generadas para la MPyME1, las inconsistencias encontradas en la evaluación por modelado inciden en las calificaciones obtenidas en cada fase de desarrollo. A diferencia del plan de mejora de la MPyME1, en el plan generado para la empresa dos, se detectan fases de desarrollo en donde la empresa tiene que hacer mayor énfasis para realizar una mejora, como los son las fases de “Construcción”, “Integración y Pruebas” y “Cierre”. Este plan de acción es más extenso en cuanto a recomendaciones y productos, esto demuestra que la MPyME2 necesita mayores iniciativas de mejora que las propuestas para la MPyME1. El nivel de madurez obtenido muestra la situación de la empresa en cuanto al modelo Moprosoft, que aunque no se tiene implementado dentro de la organización, se conoce su estructura, debido al interés de implementarlo en un futuro.

Recomendaciones y Actividades:

Inicio
-Definir un Proceso Específico con base en la Descripción del Proyecto y el proceso de Desarrollo y Mantenimiento de Software de la organización o con base en el acuerdo con el Cliente
-Definir un Plan de Manejo de Riesgos
-Determinar el Tiempo Estimado para cada actividad, considerando las Metas Cuantitativas para el Proyecto
-Elaborar un plan de Adquisiciones y Capacitación para llevar a cabo el proyecto
-Definir Ciclos y Actividades con base en la Descripción del Proyecto y en el Proceso Específico
-Realizar un cálculo del Costo Estimado del proyecto, considerando las Metas Cuantitativas para el Proyecto
-Integrar o actualizar un Plan de Proyecto por el inicio de un nuevo ciclo
-Verificar y validar el Plan de Proyecto y el Plan de Desarrollo
-Generar Reportes de Verificación y Validación
-Revisar el Plan de Desarrollo actual con los miembros del equipo de trabajo
-Elaborar un Reporte de Actividades correspondientes a la fase de Inicio

Requisitos
-Planificar, revisar y auditar, a los subcontratistas para asegurar la calidad de los servicios y cumplimiento con los estándares y especificaciones
-Distribuir la información al Equipo de Trabajo con base en el Plan de Comunicación e Implantación
-Realizar una recopilación de los Reportes de Actividades, Reportes de Mediciones y Sugerencias de Mejora
-Dar seguimiento a Plan de Adquisiciones y Capacitación
-Revisar la Descripción del Producto, al Equipo de Trabajo y Calendario
-Llevar a cabo una administración de los subcontratistas
-Generar Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente
-Verificar y Validar la Especificación de Requisitos, generando Reporte Verificación y Reporte de Validación
-Verificar el Plan de Pruebas del Sistema, generando un Reporte de Verificación
-Incorporar la Especificación de Requisitos y Plan de Pruebas del Sistema como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Análisis y Diseño
-En base al análisis de Especificación de Requisitos documentar Análisis y Diseño
-Incorporar el Análisis y Diseño, Registro de Rastreo y Plan de Pruebas de Integración como líneas base a la Configuración de Software
-Elaborar un Registro de Rastreo
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Elaborar un Reporte de Actividades correspondiente a esta fase

Construcción
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Actualizar el Registro de Rastreo, incorporando los componentes construidos o modificados
-Incorporar los Componentes de software y el Registro de Rastreo como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondientes a esta fase

Integración y Pruebas
-Evaluar el cumplimiento del Plan de Proyecto y el Plan de Desarrollo
-Realizar un seguimiento y control del Plan de Manejo de Riesgos
-Establecer Acciones Correctivas de acuerdo a la evaluación de los Planes de Proyecto y Desarrollo
-Identificar nuevos riesgos y actualizan el Plan de Manejo de Riesgos
-Generar un Reporte de Seguimiento del proyecto, considerando los Reportes de Actividades
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Integrar los componentes en subsistemas o en el sistema Software y aplicar las pruebas siguiendo el Plan de Pruebas de Integración
-Documentar los resultados de las pruebas en un Reporte de Pruebas de Integración
-Actualizar el Registro de Rastreo, incorporando los subsistemas o el sistema Software
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Elaborar un Manual de Operación
-Verificar el Manual de Operación y generar un Reporte de Verificación
-Corregir los defectos encontrados en base al Reporte de Pruebas del Sistema, hasta lograr una prueba de sistema sin defectos
-Verificar el Manual de Usuario y generar un Reporte de Verificación
-Incorporar el Software, Reporte de Pruebas de Integración, Registro de Rastreo, Manual de Operación y Manual de Usuario como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Cierre
-Realizar un cierre formal del ciclo o del proyecto
-Hacer entrega de la Configuración de Software de acuerdo al Protocolo de Entrega
-Incorporar el Manual de Mantenimiento como línea base a la Configuración de Software
-Realizar cierre de actividades con subcontratistas de acuerdo al contrato establecido
-Verificar el Manual de Mantenimiento y generar un Reporte de Verificación
-Generar Reporte de Mediciones y Sugerencias con base en el Plan de Desarrollo
-Elaborar Reporte de Actividades correspondiente a esta fase

Productos:

Inicio
-Plan de Manejo de Riesgos
-Plan de Adquisiciones y Capacitación
-Reportes de Verificación y Validación
-Reporte de Actividades correspondientes a la fase de Inicio

Requisitos
-Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente
-Reporte Verificación y Reporte de Validación de Requisitos
-Reporte de Verificación para Plan de Pruebas del Sistema
-Reporte de Actividades correspondiente a fase de Requisitos

Análisis y Diseño
-Documentar Análisis y Diseño
-Registro de Rastreo
-Reporte de Verificación de Registro de Rastreo
-Reporte de Actividades correspondiente a fase de Análisis y Diseño

Construcción
-Reporte de Verificación de Registro de Rastreo
-Reporte de Actividades correspondientes a fase de Construcción

Integración y Pruebas
-Reporte de Seguimiento del proyecto
-Resultados de las pruebas en un Reporte de Pruebas de Integración
-Reporte de Verificación de Registro de Rastreo
-Manual de Operación
-Reporte de Verificación de Manual de Operación
-Reporte de Verificación de Manual de Usuario
-Reporte de Actividades correspondiente a fase Integración y Pruebas

Cierre
-Reporte de Verificación de Manual de Mantenimiento
-Reporte de Mediciones y Sugerencias
-Reporte de Actividades correspondiente a fase de Cierre

Figura 4.4. Plan de Acción generado por SelfVation para la MPyME2.

La Figura 4.5 muestra los resultados de la evaluación por modelado para la MPyME3. En el diagrama del proceso actual se puede observar un proceso más pobre en cuanto a actividades. Este proceso a diferencia de los dos anteriores carece de actividades relacionadas con algunas fases Moprosoft, para ser más concretos con las referentes a “Análisis y Diseño” y “Cierre”. Además, el proceso carece de verificaciones y validaciones para cada una de las fases de Desarrollo.

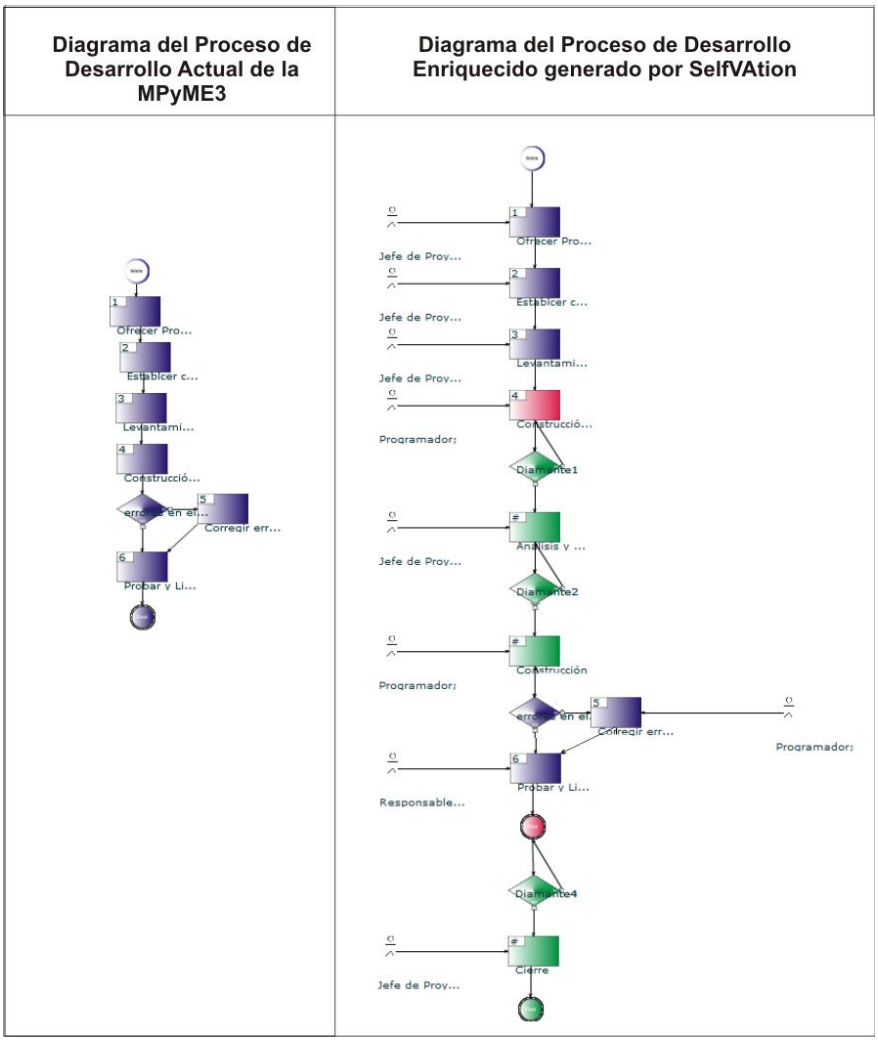
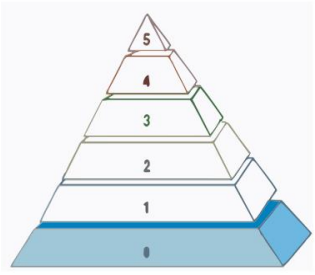


Figura 4.5. Resultados de la evaluación por modelado para la MPyME3.

El plan de acción generado para esta empresa (véase Figuras 4.6 y 4.7) muestra el estado del proceso de desarrollo que se lleva a cabo. La calificación a las fases de desarrollo es baja, el nivel de madurez obtenido es Incompleto debido a la forma “artesanal” en que desarrollan software, lo cual quiere decir, que la empresa carece de un proceso formal para el desarrollo de productos software. El plan de acción obtenido por SelfVation, es mucho más extenso que los generados anteriormente, incluyendo casi la mayoría de recomendaciones que la herramienta puede generar.



Nivel Madurez

Recomendaciones y Actividades:

Inicio:

-Definir Ciclos y Actividades con base en la Descripción del Proyecto y en el Proceso Específico
-Definir con el Cliente un Protocolo de Entrega
-Definir un Proceso Específico con base en la Descripción del Proyecto y el proceso de Desarrollo y Mantenimiento de Software de la organización o con base en el acuerdo con el Cliente
-Elaborar un plan de Adquisiciones y Capacitación para llevar a cabo el proyecto
-Integrar un Equipo de Trabajo, asignando roles y responsabilidades basándose en la Descripción del Proyecto
-Determinar el Tiempo Estimado para cada actividad, considerando las Metas Cuantitativas para el Proyecto
-Establecer un Calendario de Actividades
-Definir un Plan de Manejo de Riesgos
-Integrar o actualizar un Plan de Proyecto por el inicio de un nuevo ciclo
-Generar un Plan de Desarrollo en función del Plan de Proyecto
-Verificar y validar el Plan de Proyecto y el Plan de Desarrollo
-Generar Reportes de Verificación y Validación
-Revisar el Plan de Desarrollo actual con los miembros del equipo de trabajo
-Elaborar un Reporte de Actividades correspondientes a la fase de Inicio

Requisitos:

-Distribuir la información al Equipo de Trabajo con base en el Plan de Comunicación e Implantación
-Llevar a cabo una administración de los subcontratistas
-Revisar la Descripción del Producto, al Equipo de Trabajo y Calendario
-Dar seguimiento a Plan de Adquisiciones y Capacitación
-Realizar una recopilación de los Reportes de Actividades, Reportes de Mediciones y Sugerencias de Mejora
-Planificar, revisar y auditar, a los subcontratistas para asegurar la calidad de los servicios y cumplimiento con los estándares y especificaciones
-Registrar los costos y recursos reales del ciclo
-Revisar los productos generados durante el ciclo
-Realizar reuniones de revisión con el equipo de trabajo y con el Cliente
-Generar Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Obtener requisitos y documentar en una Especificación de Requisitos
-Verificar y Validar la Especificación de Requisitos, generando Reporte Verificación y Reporte de Validación
-Elaborar un Plan de Pruebas del Sistema
-Verificar el Plan de Pruebas del Sistema, generando un Reporte de Verificación
-Incorporar la Especificación de Requisitos y Plan de Pruebas del Sistema como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Análisis y Diseño:

-Elaborar un Registro de Rastreo
-Distribuir las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Elaborar un Plan de Pruebas de Integración
-En base al análisis de Especificación de Requisitos documentar Análisis y Diseño
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Verificar el Plan de Pruebas de Integración y generar un Reporte de Verificación
-Incorporar el Análisis y Diseño, Registro de Rastreo y Plan de Pruebas de Integración como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Construcción:

-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Definir y aplicar pruebas unitarias para verificar que el funcionamiento de cada componente esté acorde con el Análisis y Diseño
-Corregir los defectos encontrados hasta lograr pruebas unitarias sin defectos
-Actualizar el Registro de Rastreo, incorporando los componentes construidos o modificados
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Incorporar los Componentes de software y el Registro de Rastreo como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondientes a esta fase

Integración y Pruebas:

-Evaluar el cumplimiento del Plan de Proyecto y el Plan de Desarrollo
-Establecer Acciones Correctivas de acuerdo a la evaluación de los Planes de Proyecto y Desarrollo
-Realizar un seguimiento y control del Plan de Manejo de Riesgos
-Identificar nuevos riesgos y actualizan el Plan de Manejo de Riesgos
-Generar un Reporte de Seguimiento del proyecto, considerando los Reportes de Actividades
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Integrar los componentes en subsistemas o en el sistema Software y aplicar las pruebas siguiendo el Plan de Pruebas de Integración
-Documentar los resultados de las pruebas en un Reporte de Pruebas de Integración
-Corregir los defectos encontrados en base al Reporte de Pruebas de Integración, hasta lograr una prueba de integración sin defectos
-Actualizar el Registro de Rastreo, incorporando los subsistemas o el sistema Software
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Elaborar un Manual de Operación
-Verificar el Manual de Operación y generar un Reporte de Verificación
-Realizar pruebas de sistema siguiendo el Plan de Pruebas del Sistema, documentando los resultados en un Reporte de Pruebas de Sistema
-Corregir los defectos encontrados en base al Reporte de Pruebas del Sistema, hasta lograr una prueba de sistema sin defectos
-Elaborar un Manual de Usuario
-Verificar el Manual de Usuario y generar un Reporte de Verificación
-Incorporar el Software, Reporte de Pruebas de Integración, Registro de Rastreo, Manual de Operación y Manual de Usuario como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Cierre:

-Realizar un cierre formal del ciclo o del proyecto
-Hacer entrega de la Configuración de Software de acuerdo al Protocolo de Entrega
-Realizar cierre de actividades con subcontratistas de acuerdo al contrato establecido
-¿Elaboran un Manual de Mantenimiento?
-Realizar un Documento de Aceptación
-Verificar el Manual de Mantenimiento y generar un Reporte de Verificación
-Incorporar el Manual de Mantenimiento como línea base a la Configuración de Software
-Generar Reporte de Mediciones y Sugerencias con base en el Plan de Desarrollo
-Elaborar Reporte de Actividades correspondiente a esta fase
-Identificar y Documentan las Lecciones Aprendidas de este proceso

Figura 4.6. Primera parte del Plan de Acción generado por SelfVation para la MPyME3.

Productos:

Inicio:
- -Protocolo de Entrega
- -Plan de Adquisiciones y Capacitación
- -Calendario de Actividades
- -Plan de Manejo de Riesgos
- -Plan de Desarrollo
- -Reportes de Verificación y Validación
- -Reporte de Actividades correspondientes a la fase de Inicio

Requisitos:
- -Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente
- -Especificación de Requisitos
- -Reporte Verificación y Reporte de Validación de Requisitos
- -Plan de Pruebas del Sistema
- -Reporte de Verificación para Plan de Pruebas del Sistema
- -Reporte de Actividades correspondiente a fase de Requisitos

Análisis y Diseño:
- -Registro de Rastreo
- -Plan de Pruebas de Integración
- -Documentar Análisis y Diseño
- -Reporte de Verificación de Registro de Rastreo
- -Reporte de Verificación de Plan de Pruebas de Integración
- -Reporte de Actividades correspondiente a fase de Análisis y Diseño

Construcción:
- -Reporte de Verificación de Registro de Rastreo
- -Reporte de Actividades correspondientes a fase de Construcción

Integración y Pruebas:
- -Reporte de Seguimiento del proyecto
- -Resultados de las pruebas en un Reporte de Pruebas de Integración
- -Reporte de Verificación de Registro de Rastreo
- -Manual de Operación
- -Reporte de Verificación de Manual de Operación
- -Reporte de Pruebas de Sistema
- -Manual de Usuario
- -Reporte de Verificación de Manual de Usuario
- -Reporte de Actividades correspondiente a fase Integración y Pruebas

Cierre:
- -Manual de Mantenimiento
- -Documento de Aceptación
- -Reporte de Verificación de Manual de Mantenimiento
- -Reporte de Mediciones y Sugerencias
- -Reporte de Actividades correspondiente a fase de Cierre
- -Lecciones Aprendidas

Figura 4.7. Segunda parte del Plan de Acción generado por SelfVation para la MPyME3.

La Figura 4.8 presenta el resultado obtenido por la MPyME4 en la etapa de evaluación por modelado. En el caso del diagrama del proceso actual, se puede apreciar que es similar al presentado en la Figura 4.3 perteneciente a la MPyME2, esto puede deberse a las características similares de las dos empresas. El mecanismo de evaluación por modelado arrojó como resultado que la MPyME4 realiza actividades relacionadas con las tres primeras fases de desarrollo Moprosoft de manera correcta, pero, las tres siguientes presentan anomalías, las cuales se penalizarán al momento de calificar estas fases.

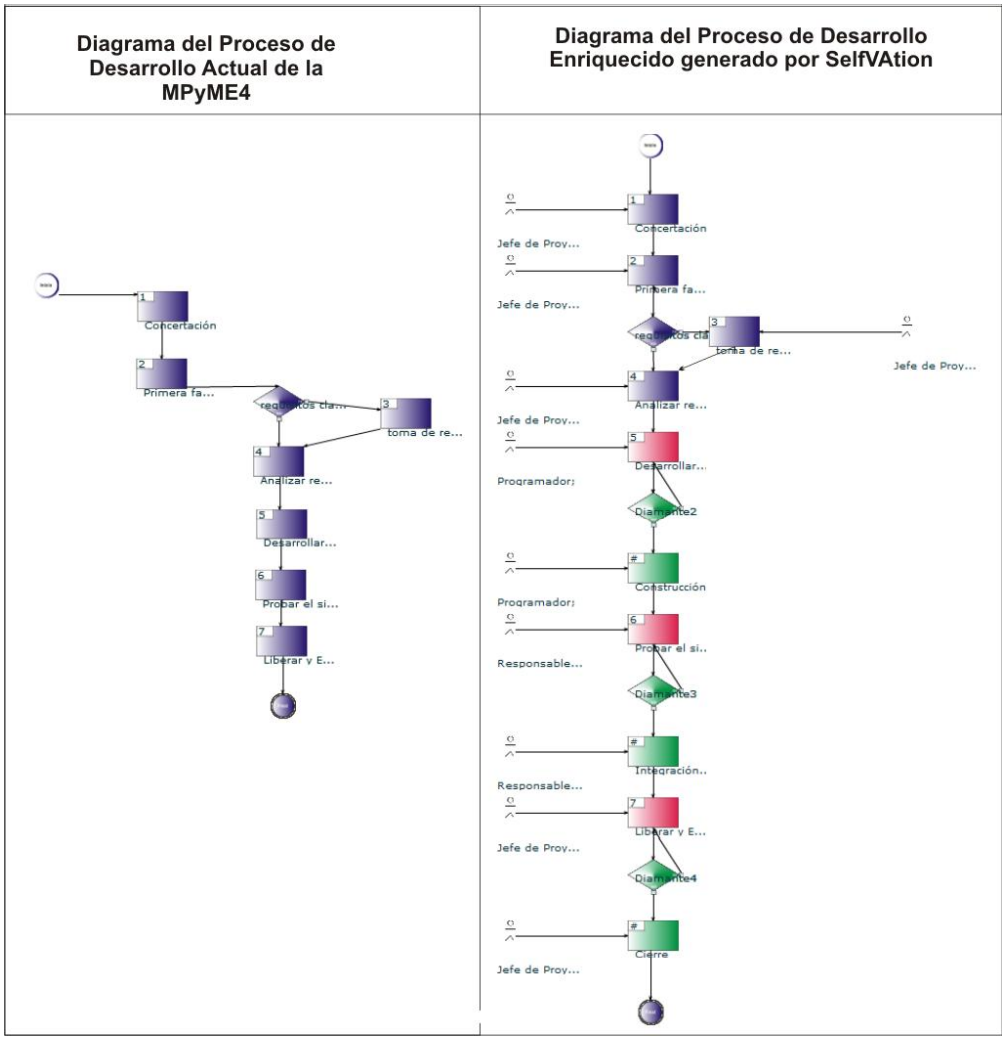


Figura 4.8. Resultados de la evaluación por modelado para la MPyME4.

El plan de acción mostrado en la Figura 4.9, es el generado por SelfVation para la MPyME4, la calificación obtenida en las fases de desarrollo es similar a la obtenida por la MPyME2. Sin embargo, son en mayor número las recomendaciones y productos que la herramienta genera para llevar a cabo una iniciativa de mejora dentro de la empresa.

Recomendaciones y Actividades:

Inicio:
-Definir un Proceso Específico con base en la Descripción del Proyecto y el proceso de Desarrollo y Mantenimiento de Software de la organización o con base en el acuerdo con el Cliente
-Definir con el Cliente un Protocolo de Entrega
-Definir Ciclos y Actividades con base en la Descripción del Proyecto y en el Proceso Específico
-Determinar el Tiempo Estimado para cada actividad, considerando las Metas Cuantitativas para el Proyecto
-Elaborar un plan de Adquisiciones y Capacitación para llevar a cabo el proyecto
-Establecer un Calendario de Actividades
-Definir un Plan de Manejo de Riesgos
-Integrar o actualizar un Plan de Proyecto por el inicio de un nuevo ciclo
-Verificar y validar el Plan de Proyecto y el Plan de Desarrollo
-Generar un Plan de Desarrollo en función del Plan de Proyecto
-Generar Reportes de Verificación y Validación
-Realizar un cálculo del Costo Estimado del proyecto, considerando las Metas Cuantitativas para el Proyecto

Requisitos:
-Asignar las tareas al Equipo de Trabajo, incluyendo a los subcontratistas
-Revisar la Descripción del Producto, al Equipo de Trabajo y Calendario
-Llevar a cabo una administración de los subcontratistas
-Distribuir la información al Equipo de Trabajo con base en el Plan de Comunicación e Implantación
-Planificar, revisar y auditar, a los subcontratistas para asegurar la calidad de los servicios y cumplimiento con los estándares y especificaciones
-Realizar una recopilación de los Reportes de Actividades, Reportes de Mediciones y Sugerencias de Mejora
-Verificar el Plan de Pruebas del Sistema, generando un Reporte de Verificación
-Verificar y Validar la Especificación de Requisitos, generando Reporte Verificación y Reporte de Validación
-Incorporar la Especificación de Requisitos y Plan de Pruebas del Sistema como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase
-Dar seguimiento a Plan de Adquisiciones y Capacitación

Análisis y Diseño:
-Distribuir las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-En base al análisis de Especificación de Requisitos documentar Análisis y Diseño
-Verificar el Plan de Pruebas de Integración y generar un Reporte de Verificación
-Elaborar un Registro de Rastreo
-Incorporar el Análisis y Diseño, Registro de Rastreo y Plan de Pruebas de Integración como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Construcción:
-Actualizar el Registro de Rastreo, incorporando los componentes construidos o modificados
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Elaborar un Reporte de Actividades correspondientes a esta fase
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Incorporar los Componentes de software y el Registro de Rastreo como líneas base a la Configuración de Software

Integración y Pruebas:
-Evaluar el cumplimiento del Plan de Proyecto y el Plan de Desarrollo
-Realizar un seguimiento y control del Plan de Manejo de Riesgos
-Establecer Acciones Correctivas de acuerdo a la evaluación de los Planes de Proyecto y Desarrollo
-Generar un Reporte de Seguimiento del proyecto, considerando los Reportes de Actividades
-Identificar nuevos riesgos y actualizan el Plan de Manejo de Riesgos
-Distribuir las a tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo
-Integrar los componentes en subsistemas o en el sistema Software y aplicar las pruebas siguiendo el Plan de Pruebas de Integración
-Documentar los resultados de las pruebas en un Reporte de Pruebas de Integración
-Corregir los defectos encontrados en base al Reporte de Pruebas de Integración, hasta lograr una prueba de integración sin defectos
-Actualizar el Registro de Rastreo, incorporando los subsistemas o el sistema Software
-Verificar el Registro de Rastreo y generar un Reporte de Verificación
-Elaborar un Manual de Operación
-Verificar el Manual de Operación y generar un Reporte de Verificación
-Realizar pruebas de sistema siguiendo el Plan de Pruebas del Sistema, documentando los resultados en un Reporte de Pruebas de Sistema
-Corregir los defectos encontrados en base al Reporte de Pruebas del Sistema, hasta lograr una prueba de sistema sin defectos
-Verificar el Manual de Usuario y generar un Reporte de Verificación
-Incorporar el Software, Reporte de Pruebas de Integración, Registro de Rastreo, Manual de Operación y Manual de Usuario como líneas base a la Configuración de Software
-Elaborar un Reporte de Actividades correspondiente a esta fase

Cierre:
-Realizar un cierre formal del ciclo o del proyecto
-Realizar cierre de actividades con subcontratistas de acuerdo al contrato establecido
-Verificar el Manual de Mantenimiento y generar un Reporte de Verificación
-Generar Reporte de Mediciones y Sugerencias con base en el Plan de Desarrollo
-Incorporar el Manual de Mantenimiento como línea base a la Configuración de Software
-Realizar un Documento de Aceptación
-Identificar y Documentan las Lecciones Aprendidas de este proceso
-Elaborar Reporte de Actividades correspondiente a esta fase

Productos:

Inicio:
-Protocolo de Entrega
-Plan de Adquisiciones y Capacitación
-Calendario de Actividades
-Plan de Manejo de Riesgos
-Plan de Desarrollo
-Reportes de Verificación y Validación

Requisitos:
-Reporte de Verificación para Plan de Pruebas del Sistema
-Reporte Verificación y Reporte de Validación de Requisitos
-Reporte de Actividades correspondiente a fase de Requisitos

Análisis y Diseño:
-Reporte de Verificación de Registro de Rastreo
-Documentar Análisis y Diseño
-Reporte de Verificación de Plan de Pruebas de Integración
-Registro de Rastreo
-Reporte de Actividades correspondiente a fase de Análisis y Diseño

Construcción:
-Reporte de Actividades correspondientes a fase de Construcción
-Reporte de Verificación de Registro de Rastreo

Integración y Pruebas:
-Reporte de Seguimiento del proyecto
-Resultados de las pruebas en un Reporte de Pruebas de Integración
-Reporte de Verificación de Registro de Rastreo
-Manual de Operación
-Reporte de Verificación de Manual de Operación
-Reporte de Pruebas de Sistema
-Reporte de Verificación de Manual de Usuario
-Reporte de Actividades correspondiente a fase Integración y Pruebas

Cierre:
-Reporte de Verificación de Manual de Mantenimiento
-Reporte de Mediciones y Sugerencias
-Documento de Aceptación
-Lecciones Aprendidas
-Reporte de Actividades correspondiente a fase de Cierre

Figura 4.9. Plan de Acción generado por SelfVation para la MPyME4.

4.2. Resultados Experimentales y Lecciones Aprendidas

Después de la implementación de la fase de evaluación y mejora se obtuvo información acerca de las prácticas y el esfuerzo realizado por las cuatro pequeñas empresas. En base a los datos obtenidos, se observo que la MPyME1 la cual cuenta con la implementación oficial de la norma NMX-I-059/02-NYCE-2005 nos proporciona una medida contrastante con las otras tres empresas. Todas las empresas fueron evaluadas en el Nivel 1 de Moprosoft (nivel Proceso Realizado), la Figura 4.10 muestra los resultados obtenidos en la sesión de evaluación.

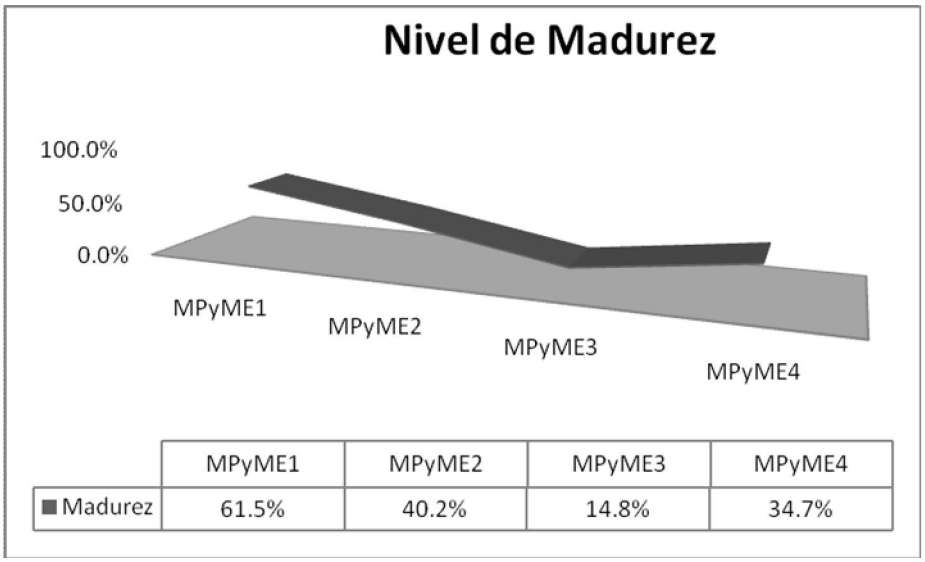


Figura 4.10. Nivel de Madurez obtenido por empresa.

Se puede creer que existe una relación entre el tamaño de la empresa (No. de empleados) y la eficacia para la adopción de la norma. No se puede afirmar que exista una relación proporcional entre ambos factores, pero es posible que los pequeños equipos de trabajo puedan ser más organizados y realicen más rápidamente la adopción de la norma. La Figura 4.10 muestra que la MPyME1 obtuvo el 61.5% de cobertura en el Nivel 1 de la norma, lo cual corresponde al nivel real con el que cuenta la empresa en la actualidad. Las MPyME2 y MPyME4 obtuvieron resultados de cobertura similares, aproximadamente el 37%, estos resultados son insuficientes para alcanzar el Nivel 1 (de acuerdo al mecanismo de evaluación debe superarse el 60%). Por último la MPyME3 obtuvo un índice de cobertura muy bajo del 14.8% lo que la sitúa en una situación *ad-hoc* o caótica. Estos resultados fueron obtenidos al llevar a cabo una iniciativa de mejora usando la herramienta SelfVation.

Por otra parte, la cobertura por cada fase propuesta por la norma NMX-I-059/02-NYCE-2005 fue obtenida también. La Figura 4.11 muestra que la mejor cobertura fue la obtenida en la fase de Requisitos, con un promedio del 60% de cobertura para las cuatro empresas y una desviación estándar del 17%, lo que indica que las empresas ponen un mayor énfasis en esta etapa ya que los requerimientos se utilizan como base para desarrollar un plan de trabajo y construir un buen producto software. La Figura 4.11 también muestra que la peor cobertura fue la obtenida para la fase de Integración y Pruebas con un 22% de promedio y una desviación estándar del 17% esto se debe a que las pequeñas compañías desarrolladoras de software no utilizan métodos formales para validar y verificar sus productos software.

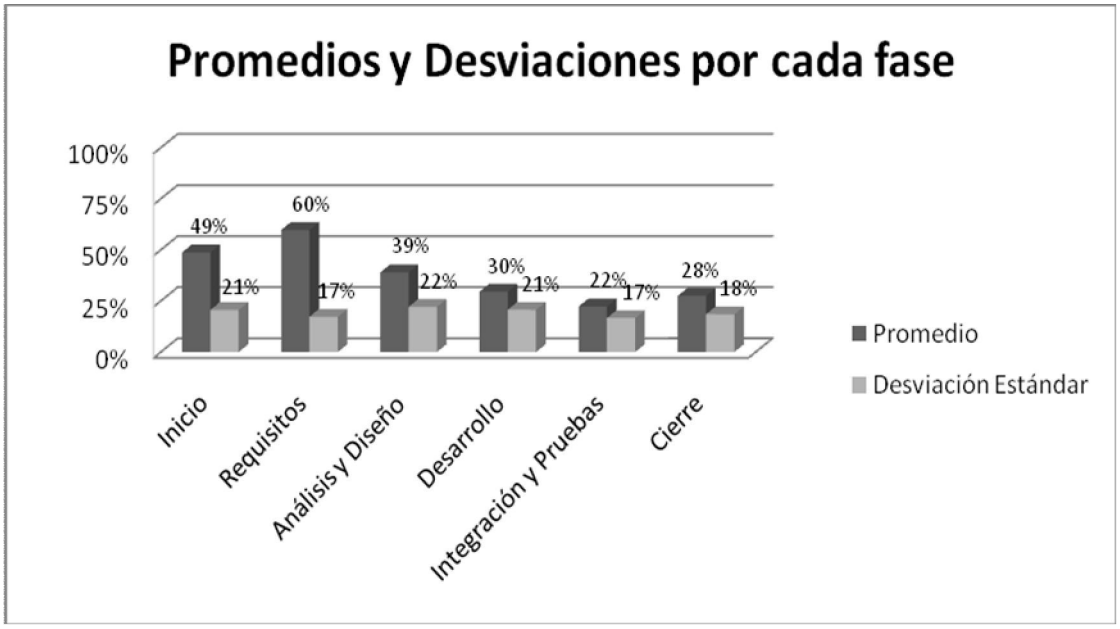


Figura 4.11. Cobertura por fase.

El propósito de este caso de estudio fue demostrar la facilidad y utilidad de SelfVation como una herramienta de SPI para MPyMEs desarrolladoras de software, la cual reduce los costos de llevar a cabo una iniciativa SPI dentro de la empresa, teniendo en cuenta las características particulares de este tipo de organizaciones. Las evaluaciones iniciales con SelfVation dieron la pauta para establecer la línea base de la capacidad de las empresas de acuerdo al estándar NMX-I-059/02-NYCE-2005. El resultado fue “pobre” entre 0 y 1 (según los niveles propuestos por Moprosoft). Durante los siguientes 3 meses la herramienta SelfVation dirigió a las empresas en la adaptación y adopción de iniciativas SPI a través de los planes de acción generados por la herramienta.

Finalmente, se aplicó una segunda evaluación con SelfVation a cada empresa; todas las empresas alcanzaron un aumento promedio de 1.00 en el nivel de madurez de dos categorías: Administración de Proyectos Específicos y Desarrollo y Mantenimiento de Software. El aumento de nivel se debe a que el proceso inicial tiene una cobertura de más del 100%. Por ejemplo, la MPyME1 tuvo una cobertura inicial del 61.5% cuando se realizó la primera evaluación. En la nueva evaluación, después de utilizar SelfVation, se mostró un incremento en los procesos de un 0.28 (en la primera iteración). Por lo tanto, el proceso de la empresa tiene un nivel de mejora de 61.5%+61.5%*0.28. La Tabla 20 muestra los niveles de mejora en todas las empresas.

Tabla 20. Datos de mejora y esfuerzo por empresa.

	Empresas			
	MPyME1	**MPyME2**	**MPyME3**	**MPyME4**
Esfuerzo total (horas)	**287**	**484**	**554**	**220**
Esfuerzo total Selfvation	**243**	**400**	**495**	**185**
Esfuerzo por persona (horas)	**28.7**	**24.2**	**36.9**	**27.5**
Esfuerzo con SelfVation	**14.2**	**11.3**	**17.9**	**20.2**
Promedio de mejora	**0.28**	**0.86**	**1.43**	**1.00**

La evaluación oficial en la MPyME1 demuestra que los resultados de SelfVation coinciden con los datos obtenidos por dicha evaluación; sin embargo con la herramienta se obtuvo esta información con menos esfuerzo, esto significa que SelfVation es capaz de reducir el tiempo de adopción. Estos resultados muestran la factibilidad de la aplicación de la herramienta como apoyo para implantar iniciativas SPI dentro de una organización pequeña; sin embargo se tiene que experimentar con más de cuatro empresas. Como trabajo futuro se está diseñando un experimento que envuelva a 15 pequeñas compañías de diferentes ciudades en la Republica Mexicana.

5. Conclusiones

El éxito en la implementación de una iniciativa de mejora al proceso software en las empresas depende del compromiso que se establezca en la alta dirección de la organización. Las pequeñas compañías no son la excepción, este trabajo de investigación permite identificar que las pequeñas empresas software no entienden los beneficios que el proceso de mejora tendría en la empresa. Aunado a la iniciativa de un programa de mejora, están la participación y experiencia del personal clave de la empresa los cuales pueden ser factores muy importantes que contribuyan a fortalecer la iniciativa de mejora. Pero, la norma NMX-059/02-NYCE-2005 es relativamente nueva en México y la experiencia en SPI por parte de las pequeñas compañías es muy pequeña o nula.

Este trabajo de investigación, por lo tanto, ha desarrollado un instrumento para definir y poner en práctica iniciativas SPI para mejorar la situación actual de las prácticas que se llevan a cabo durante el ciclo de desarrollo software dentro de las MPyMEs utilizando la norma NMX-059/02-NYCE-2005 como modelo de referencia. El objetivo es estudiar la viabilidad del uso de este instrumento en las pequeñas empresas software mexicanas y establecer las bases de una investigación futura.

Uno de los aspectos importantes durante el proyecto fue verificar la viabilidad del nuevo mecanismo de evaluación implementado para la herramienta SelfVation. El cuestionario era una técnica utilizada con anterioridad y que brindo los resultados esperados, ya que al tratarse de una forma de evaluación conocida y fácil de implementar se adopto perfectamente al objetivo de este proyecto. Sin embargo, el nuevo concepto de evaluación implementado en este trabajo se enfocó a modelar el proceso actual de la empresa, este tipo de evaluación tenía dos objetivos principales, ofrecer un "*snapshot*" de la situación actual de la empresa y brindar un apoyo a la evaluación por cuestionario para fortalecer la veracidad de las respuestas de los evaluados. Ambos objetivos fueron alcanzados y con resultados favorables, ya que mediante este mecanismo se aumenta la veracidad de las evaluaciones y nos proporciona un marco para generar planes de mejora más apegados a las necesidades de la empresa relacionadas con la mejora de sus procesos. Este es el primer acercamiento para evitar las evaluaciones no veraces que se presentan en este tipo de herramientas y marca la pauta para utilizar este tipo de mecanismos de evaluación en sistemas de esta índole.

Cabe mencionar que esto fue comprobado en la experimentación con una MPyME alterna la cual no fue incluida en el estudio presentado en el Capítulo 4 por cuestiones ajenas a nuestro control. Se detectó la falta de compromiso para establecer una iniciativa de SPI y se comprobó la efectividad del mecanismo creado para SelfVation cuando fue posible detectar la intención de "mentir" en la evaluación. La empresa en cuestión presentó serias inconsistencias de información entre los jefes de proyectos evaluados lo que nos permitió comprobar que no existe un proceso definido que esté institucionalizado a nivel organizacional. Sin el mecanismo de evaluación por modelado esto hubiera sido mucho más difícil de demostrar con evidencia real.

Una de las limitaciones de este estudio es la generalización en base a la cantidad limitada de de datos recogidos y analizados concerniente en las pequeñas organizaciones que colaboraron en este estudio. Esto sugiere que cuando este estudio cualitativo sea aumentado y reforzado por estudios cuantitativos para consolidar los datos se fortalecerá la necesidad y aplicabilidad de SelfVation dentro de la comunidad de empresas pequeñas mexicanas.

En el momento de esta investigación, la MPyME1 que colaboró para este estudio había decidido iniciar un proceso de mejora en sus procesos para alcanzar el Nivel 2 de la certificación. En este sentido, se colaboró mediante el establecimiento de un ciclo iterativo de SPI utilizando SelfVation para alcanzar la meta propuesta.

Como trabajo futuro se plantea el desarrollo de una herramienta automatizada que cumpla con la implantación de un ciclo SPI completo, cubriendo las fases de evaluación, planificación, control y seguimiento de una iniciativa de mejora dentro de la pequeña empresa y brindarles apoyo a en la implementación, control y seguimiento de iniciativas de SPI bajo el modelo Moprosoft, es decir establecer procesos maduros y capaces que generen productos de calidad.

6. Anexo A.- Acrónimos

IEEE	Instituto de Ingenieros en Electricidad y Electrónica
DRA	Desarrollo Rápido de Aplicaciones
IS	Ingeniería de Software
XP	Programación Extrema
SPI	Mejora al Proceso Software
CMMI	Modelo de Madurez y Capacidad Integrado
Moprosoft	Modelo de Procesos para la Industria de Software
SPA	Modelo de Evaluación del Proceso Software
Pymes	Pequeñas y medianas empresas
MPyMEs	Micro, Pequeñas y Medianas Empresas
SEI	Instituto de Ingeniería de Software
ISO	Organización Internacional para la Estandarización
TI	Tecnologías de Información
SE	Secretaría de Economía
PROSOFT	Programa para el Desarrollo de la Industria del Software
SNIITI	Sistema Nacional de Indicadores de la Industria de Tecnologías de Información
RIA	Aplicaciones Ricas en Internet
PND	Plan Nacional de Desarrollo
Evalprosoft	Método de Evaluación de procesos para la Industria Software
MDM	Manejador de Documentos de Moprosoft
HIM	Herramienta Integral para Moprosoft

7. Anexo B.- Cuestionario para la evaluación de la Categoría Operación del modelo Moprosoft

El siguiente cuestionario fue elaborado en base a las prácticas propuestas por los procesos Administración de Proyectos Específicos y Desarrollo y Mantenimiento pertenecientes a la categoría Operación del modelo Moprosoft. El cuestionario se divide en base a las fases propuestas por las áreas de proceso.

7.1. Administración de Proyectos Específicos

El propósito de la Administración de Proyectos Específicos es establecer y llevar a cabo sistemáticamente las actividades que permitan cumplir con los objetivos de un proyecto en tiempo y costos esperados.

I. Planificación					
1. ¿Define un Proceso Específico con base en la Descripción del Proyecto y el Proceso de Desarrollo y Mantenimiento de Software de la organización o con base en el acuerdo con el Cliente?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Definen con el Cliente un Protocolo de Entrega?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Definen Ciclos y Actividades con base en la Descripción del Proyecto y en el Proceso Específico?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Determinan el Tiempo Estimado para cada actividad, considerando las Metas Cuantitativas para el Proyecto?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Elaboran un plan de Adquisiciones y Capacitación para llevar a cabo el proyecto?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Integran un Equipo de Trabajo, asignando roles y responsabilidades basándose en la Descripción del Proyecto?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Establecen un Calendario de Actividades?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

8. ¿Realizan un cálculo del Costo Estimado del proyecto, considerando las Metas Cuantitativas para el Proyecto?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
9. ¿Definen un Plan de Manejo de Riesgos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
10. ¿Integran o actualizan un Plan de Proyecto por el inicio de un nuevo ciclo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
11. ¿Generan un Plan de Desarrollo en función del Plan de Proyecto?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
12. ¿Verifican y validan el Plan de Proyecto y el Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
13. ¿Generan Reportes de Verificación y Validación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

II. Realización					
1. ¿Asignan las tareas al Equipo de Trabajo, incluyendo a los subcontratistas?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Distribuyen la información al Equipo de Trabajo con base en el Plan de Comunicación e Implantación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Revisan la Descripción del Producto, al Equipo de Trabajo y Calendario?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Dan seguimiento a Plan de Adquisiciones y Capacitación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Llevan a cabo una administración de los subcontratistas?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Planifican, revisan y auditan, a los subcontratistas para asegurar la calidad de los servicios y cumplimiento con los estándares y especificaciones?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Realizan una recopilación de los Reportes de Actividades, Reportes de Mediciones y Sugerencias de Mejora?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
8. ¿Registran los costos y recursos reales del ciclo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
9. ¿Revisan los productos generados durante el ciclo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
10. ¿Realizan reuniones de revisión con el equipo de trabajo y con el Cliente?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

11. ¿Generan Minutas con puntos tratados y acuerdos tomados en las reuniones con el Cliente?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
III. Evaluación y Control					
1. ¿Evalúan el cumplimiento del Plan de Proyecto y el Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Establecen Acciones Correctivas de acuerdo a la evaluación de los Planes de Proyecto y Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Realizan un seguimiento y control del Plan de Manejo de Riesgos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Identifican nuevos riesgos y actualizan el Plan de Manejo de Riesgos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Generan un Reporte de Seguimiento del proyecto, considerando los Reportes de Actividades?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
IV. Cierre					
1. ¿Realizan un cierre formal del ciclo o del proyecto?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Hacen entrega de la Configuración de Software de acuerdo al Protocolo de Entrega?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Realizan un Documento de Aceptación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Realizan cierre de actividades con subcontratistas de acuerdo al contrato establecido?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Generan Reporte de Mediciones y Sugerencias de Mejora del proceso en base al Plan de Mediciones de Procesos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Identifican y Documentan las Lecciones Aprendidas de este proceso?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

7.2. Desarrollo y Mantenimiento de Software

El propósito de Desarrollo y Mantenimiento de Software es la realización sistemática de las actividades de análisis, diseño, construcción, integración y pruebas de productos de software, nuevos o modificados, cumpliendo con los requerimientos especificados.

I. Realización de la fase de Inicio					
1. ¿Revisan el Plan de Desarrollo actual con los miembros del equipo de trabajo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Elaboran un Reporte de Actividades correspondientes a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
II. Realización de la fase de Requisitos					
1. ¿Distribuyen las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Obtienen requisitos y los documentan en una Especificación de Requisitos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Verifican y Validan la Especificación de Requisitos, generando Reporte Verificación y Reporte de Validación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Elaboran un Plan de Pruebas del Sistema?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Verifican el Plan de Pruebas del Sistema, generando un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Incorporan la Especificación de Requisitos y Plan de Pruebas del Sistema como líneas base a la Configuración de Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Elaboran un Reporte de Actividades correspondiente a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
III. Realización de la fase de Análisis y Requisitos					
1. ¿Distribuyen las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿En base al análisis de Especificación de Requisitos documentan Análisis y Diseño?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Elaboran un Registro de Rastreo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

4. ¿Verifican el Registro de Rastreo y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Elaboran un Plan de Pruebas de Integración?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Verifican el Plan de Pruebas de Integración y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Incorporan el Análisis y Diseño, Registro de Rastreo y Plan de Pruebas de Integración como líneas base a la Configuración de Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
8. ¿Elaboran un Reporte de Actividades correspondiente a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
IV. Realización de la fase de Construcción					
1. ¿Distribuyen las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Construyen los Componentes de Software con base en el Análisis y Diseño?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Definen y aplican pruebas unitarias para verificar que el funcionamiento de cada componente esté acorde con el Análisis y Diseño?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Corrigen los defectos encontrados hasta lograr pruebas unitarias sin defectos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Actualizan el Registro de Rastreo, incorporando los componentes construidos o modificados?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Verifican el Registro de Rastreo y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Incorporan los Componentes de software y el Registro de Rastreo como líneas base a la Configuración de Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
8. ¿Elaboran un Reporte de Actividades correspondiente a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
V. Realización de la fase de Integración y Pruebas					
1. ¿Distribuyen las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Integran los componentes en subsistemas o en el sistema Software y aplican las pruebas siguiendo el Plan de Pruebas de Integración?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

3. ¿Documentan los resultados de las pruebas en un Reporte de Pruebas de Integración?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Corrigen los defectos encontrados en base al Reporte de Pruebas de Integración, hasta lograr una prueba de integración sin defectos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
5. ¿Actualizan el Registro de Rastreo, incorporando los subsistemas o el sistema Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Verifican el Registro de Rastreo y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Elaboran un Manual de Operación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
8. ¿Verifican el Manual de Operación y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
9. ¿Realizan pruebas de sistema siguiendo el Plan de Pruebas del Sistema, documentando los resultados en un Reporte de Pruebas de Sistema?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
10. ¿Corrigen los defectos encontrados en base al Reporte de Pruebas del Sistema, hasta lograr una prueba de sistema sin defectos?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
11. ¿Elaboran un Manual de Usuario?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
12. ¿Verifican el Manual de Usuario y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
13. ¿Incorporan el Software, Reporte de Pruebas de Integración, Registro de Rastreo, Manual de Operación y Manual de Usuario como líneas base a la Configuración de Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
14. ¿Elaboran un Reporte de Actividades correspondiente a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
VI. Realización de la fase de Cierre					
1. ¿Distribuyen las tareas a los miembros del equipo de trabajo según sus roles, de acuerdo al Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
2. ¿Elaboran un Manual de Mantenimiento?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
3. ¿Verifican el Manual de Mantenimiento y generan un Reporte de Verificación?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
4. ¿Incorporan el Manual de Mantenimiento como línea base a la Configuración de Software?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

5. ¿Generan Reporte de Mediciones y Sugerencias con base en el Plan de Desarrollo?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
6. ¿Identifican y Documentan las Lecciones Aprendidas de este proceso?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒
7. ¿Elaboración del Reporte de Actividades correspondiente a esta fase?	Siempre ❒	Usualmente ❒	A veces ❒	Rara vez ❒	Nunca ❒

8. Anexo C.- Código Fuente comentado de los componentes más importantes del sistema

8.1. Código fuente de métodos de mayor importancia en Componente "CrearDiagrama"

```
/*Cuando se detecta el evento mouseDown se empieza a dibujar una línea o a
arrastrar un elemento dentro del área de dibujo.
Si el botón de línea esta seleccionado, las coordenadas del mouse son el
origen y final de la línea temporal, si no esta seleccionado el botón para
dibujar una línea quiere decir que se va a mover un elemento en el área de
dibujo*/
    private function mouseDown(event:MouseEvent):void{
      //Se checa si esta seleccionado el botón de dibujar una línea
      if (!designer.getIsDrawEnable()){
//Si no esta seleccionado se inica el proceso de mover el elemento
//seleccionado
        this.startDrag();
      }
      else{
       // Si esta seleccionado el botón de dibujar línea se procede a crear
      //una línea temporal.
       //Si se trata de un elemento de tipo diamante se hacen las
restricciones para que
       //el origen de las líneas sea solamente en las áreas marcadas para las
decisiones
       // Si o No.
        if(tipoSimbolo=="diamante"){
            if(isNo(event.localX,event.localY))
                  {
                  desicionDiamante="No";
                  designer.setDesicionDiamante(desicionDiamante);
                  designer.prepareDrawing();
                  designer.setCurrentFromBox(this);
                  }
            else if(isYes(event.localX,event.localY))
                  {
                  desicionDiamante="Si";
                  designer.setDesicionDiamante(desicionDiamante);
                  designer.prepareDrawing();
                  designer.setCurrentFromBox(this);
                  }
        }
        else{
        //En caso de tratarse de un diamante se preparan para dibujar una
linea temporal
        designer.prepareDrawing();
        //Se asigna el elemento en el que se detecto el evento mouseDown como
el elemento
        //Origen de la línea.
        designer.setCurrentFromBox(this);
        }
      diamanteSi=false;
      diamanteNo=false;
      }
    }
```

```
//Este método trabaja cuando se detecta un mouseDown en un elemento
   // y pone los coordenadas de origen en la línea temporal
    public function prepareDrawing():void{
       //Si se esta dibujando una linea temporal asigna las coordenadas
       //del mouse como origen de la linea temporal
      if (isDrawEnable){
        templateLine.graphics.clear();
        templateLine.setX1(designArea.mouseX);
        templateLine.setY1(designArea.mouseY);
        templateLine.visible = true;
        isDrawing = true;
      }
    }
//Este método detecta el evento mouseMove y mueve la línea temporal
//según las nuevas coordenadas del mouse si isDrawing es verdadero
    public function mouseMove(event:MouseEvent):void{
      if (isDrawing){
        //Se llama al método que actualiza las coordenadas destino
        //de la línea temporal
        drawLine();
      }
    }
// Este método actualiza las coordenadas destino de la línea temporal
// de acuerdo a la nueva posición del mouse.
    public function drawLine():void{
      if (isDrawing){
        // se actualizan las coordenadas destino de la línea temporal
        templateLine.setX2(designArea.mouseX);
        templateLine.setY2(designArea.mouseY);
      // se dibuja de nuevo la linea
        templateLine.draw();
      }
    }
// El evento mouseUp define si se para de arrastrar el elemento o se finaliza
// de dibujar una línea.
//Si se finaliza de dibujar una línea se actualizan las coordenadas finales
// de la línea
//Si se paro de arrastrar el elemento se actualizan sus corneadas
    private function mouseUp(event:MouseEvent):void{
      if (!designer.getIsDrawEnable()){
       this.coorX=event.localX;
       this.coorY=event.localY;
       this.stopDrag();
       coordenadas.x=this.getX();
       coordenadas.y=this.getY();
       this.drawLabel(this.tipoSimbolo);
      }
      else{
      //Se asigna el elemento en el que se detecto el evento mouseUp como el
      //elemento destino de la línea.
      designer.setCurrentToBox(this);
      //se agrega una conexión al área de dibujo
      designer.addLine();
      }
    }
```

```
// Este método es el encargado de crear una conexión
//ya que se cumplieron las condiciones de una conexión.
public function addLine():void{
      var bandera:Boolean=false;
      if (isDrawing){
        // Se verifica existan las condiciones necesarias para
        // crear una conexión por ejemplo que un elemento "Inicio"
        // No puede ser el destino de una conexión

if((currentFromBox.getTipoSimbolo()=="diamante"&&currentToBox.getTipoSimbolo()
=="diamante")||

(currentFromBox.getTipoSimbolo()=="diamante"&&currentToBox.getTipoSimbolo()=="
final"))
            bandera=true;
        if(currentFromBox.getTipoSimbolo()!="final" &&
        currentToBox.getTipoSimbolo()!="inicio" && rolValidacion() &&
!bandera)
        {
        //Se crea un nuevo objeto Línea
        var newLine:Line = new Line();
        newLine.setId(getId("Line"));
        //se asigna el elemento Origen a la línea
        newLine.setFromBox(currentFromBox);
        if(currentFromBox.getTipoSimbolo()=="diamante")
            newLine.setDesicionDiamante(desicionD);
        //se asigna el elemento Destino a la línea
        newLine.setToBox(currentToBox);
        // se dibuja la nueva línea con las características de los elementos
        //Origen y Destino
        newLine.draw();
        currentFromBox.addFromLine(newLine);
        currentToBox.addToLine(newLine);
        //Se agrega la nueva a un arreglo que contiene todas las conexiones
existentes
        lines.addItem(newLine);
        designArea.addChild(newLine);
        newLine.setDesigner(this);
        cancelDrawing();
        }
      }
      else{
        //En caso de no cumplirse los condiciones se cancela el dibujo de la
        //conexión
        cancelDrawing();
      }
         }
```

8.2. Código Fuente de métodos de mayor Importancia en Componente "ResultadosDiagrama"

```
//Método principal del componente "ResultadosDiagrama", en este método se lleva cabo
//la construcción del diagrama enriquecido.
//En este método se utilizan variables obtenidas anteriormente por ejemplo:
// tablaDiagrama contiene el diagrama creado en el editor
//elementoConsulta es la consulta que se realizo a tabla que contiene el diagrama editado en
//Selfvation
// diagramaMoprosoft contiene el diagrama ideal Moprosoft
// diagramaRich contiene el diagrama enriquecido que se va generando
private function crearRichDiagrama():void{
var i:int=0;
var indice:int=0;
var procesoActual:String;
var cadenaAux:String;
var indiceRich:int=0;
var elementoRich:Object;
var estado:int;
//Se realiza un ciclo para checar todos los elementos del diagrama actual!!
for(i=0;i<elementosConsulta.length();i++){
tablaDiagrama.selectedIndex=i;
//No se toman en cuenta los elementos tipo persona
if(tablaDiagrama.selectedItem.tipo=="persona")
       continue;
tablaDiagrama.selectedIndex=i;
//En este apartado de if-else se busca el índice del elemento Moprososft que corresponde
//al elemento analizado
if(tablaDiagrama.selectedItem.tipo=="caja"){
indice=buscaElementoMoprosoft(tablaDiagrama.selectedItem.procRela);
procesoActual=tablaDiagrama.selectedItem.procRela;
}
else{
indice=buscaElementoMoprosoft(tablaDiagrama.selectedItem.tipo);
procesoActual=tablaDiagrama.selectedItem.tipo;
}
diagramaMoprosoft.selectedIndex=indice;
//Se busca el elemento Origen del elemento Actual
tablaDiagrama.selectedIndex=buscaElemento(tablaDiagrama.selectedItem.procFrom);
//Se busca el elemento Moprosonft que corresponda al elemento Origen del Elemento Actual
diagramaMoprosoft.selectedIndex=diagramaMoprosoft.selectedItem.procesoFrom;
cadenaAux=diagramaMoprosoft.selectedItem.proceso;
//Se verifica si el elemento Origen en el diagrama actual es el mismo elemento Origen
// que en el diagrama Moprosoft ó es el mismo elemento
if(tablaDiagrama.selectedItem.procRela==cadenaAux ||
procesoActual==tablaDiagrama.selectedItem.procRela){
       //Si es verdad se agrega el elemento actual al diagrama enriquecido denotando que es
       // una actividad correcta
       tablaDiagrama.selectedIndex=i;
      elementoRich=new Object();
       elementoRich.id_Elemento=tablaDiagrama.selectedItem.id_Elemento;
      elementoRich.tipo=tablaDiagrama.selectedItem.tipo;
       elementoRich.procFrom=tablaDiagrama.selectedItem.procFrom;
       elementoRich.procRela=tablaDiagrama.selectedItem.procRela;
       elementoRich.procTo=tablaDiagrama.selectedItem.procTo;
       elementoRich.Numero=tablaDiagrama.selectedItem.Numero;
       elementoRich.Nombre=tablaDiagrama.selectedItem.Nombre;
       elementoRich.tipodesicion=tablaDiagrama.selectedItem.tipodesicion;
       elementoRich.estado=1;
      diagramaRich.addItem(elementoRich);
      if(diagramaRich[diagramaRich.length-2]!="diamante")
          diagramaRich[diagramaRich.length-2].procTo=elementoRich.id_Elemento;
          indiceRich++;
           }
}
else{
//Si los elementos Origen no son iguales se Insertar el proceso al diagrama enriquecido denotando
//que esta mal!!!
tablaDiagrama.selectedIndex=i;
```

```
elementoRich=new Object();
elementoRich.id_Elemento=tablaDiagrama.selectedItem.id_Elemento;
elementoRich.estado=estado;
elementoRich.procFrom=diagramaRich[diagramaRich.length-1].id_Elemento;
elementoRich.procTo=tablaDiagrama.selectedItem.procTo;
elementoRich.Numero=tablaDiagrama.selectedItem.Numero;
elementoRich.procRela=tablaDiagrama.selectedItem.procRela;
elementoRich.Nombre=tablaDiagrama.selectedItem.Nombre;
elementoRich.tipo=tablaDiagrama.selectedItem.tipo;
elementoRich.tipodesicion=tablaDiagrama.selectedItem.tipodesicion;
diagramaRich.addItem(elementoRich);
if(diagramaRich[diagramaRich.length-2]!="diamante")
       diagramaRich[diagramaRich.length-2].procTo=elementoRich.id_Elemento;
//Se agregan los procesos Moprosoft que falten en el diagrama
agregarElementosMoprosoft(tablaDiagrama.selectedIndex);
//Se añade al textArea la cadena que describa el error
textoDiagrama.text=cadenaDiagrama;
}//Fin Else
}//Fin For
calcularCoordenadas();//SE calculan las coordenadas para el nuevo Diagrama
drawDiagrama2();//Se dibuja el diagrama enriquecido
drawRoles();//Se dibujan los roles correctos para el diagrama enriquecido
estadoProceso();//Se hace una lista del estado de las fases de desarrollo
Resultados.setEstadoFases(estadoFases); //Se envía el estado de las fases a el componente
//Resultados para penalizar o anular la evaluación de una determinada área de proceso que
//presente inconsistencias
}
//Este método es el encargado de generar el mensaje de error y de agregar los procesos Moprosoft
// que faltasen en el diagrama actual de la empresa
private function agregarElementosMoprosoft(indiceActual:int):void{
var elementoFinal:String;
var cadenaAux:String;
var elementoActual:String;
var indice:int=0;
var elementoRich:Object;
var bandera:Boolean=true;
var indiceMopro:int;
var insertaProcesos:Boolean=true;
var nombreAnterior:String;
var procesoAnterior:String;
elementoActual=tablaDiagrama.selectedItem.id_Elemento;
//En este apartado de if-else se busca el índice del elemento Moprososft que corresponde
//al elemento analizado
if(tablaDiagrama.selectedItem.tipo=="caja")
{
elementoFinal=tablaDiagrama.selectedItem.procRela;
diagramaMoprosoft.selectedIndex=buscaElementoMoprosoft(tablaDiagrama.selectedItem.procRela);
indiceMopro=diagramaMoprosoft.selectedIndex;
tablaDiagrama.selectedIndex=buscaElemento(tablaDiagrama.selectedItem.procFrom);
if(tablaDiagrama.selectedItem.tipo=="inicio" )
       cadenaAux="Inicio";
else
       cadenaAux=tablaDiagrama.selectedItem.procRela;
nombreAnterior=tablaDiagrama.selectedItem.Nombre;
procesoAnterior=tablaDiagrama.selectedItem.procRela;
}
else{
elementoFinal=tablaDiagrama.selectedItem.tipo;
diagramaMoprosoft.selectedIndex=buscaElementoMoprosoft(tablaDiagrama.selectedItem.tipo);
indiceMopro=diagramaMoprosoft.selectedIndex;
tablaDiagrama.selectedIndex=buscaElemento(tablaDiagrama.selectedItem.procFrom);
///Aquí vamos a buscar el elemento anterior Moprosoft para insertar desde ese elemento buscado
//hasta el elemento Moprosoft que coincida con el elemento actual
diagramaMoprosoft.selectedIndex=diagramaMoprosoft.selectedItem.procesoFrom;
var elementoAnterior:String=diagramaMoprosoft.selectedItem.proceso;
diagramaMoprosoft.selectedIndex=indiceMopro;
while(true){
         if(diagramaMoprosoft.selectedItem.proceso==cadenaAux){
           indice=diagramaMoprosoft.selectedItem.procesoTo;
            break;
```

```
            }
        else{
        diagramaMoprosoft.selectedIndex=diagramaMoprosoft.selectedItem.procesoFrom;
        }
        if(diagramaMoprosoft.selectedItem.proceso=="inicio")
            break;
}

tablaDiagrama.selectedIndex=indiceActual;
var conta:int=0;
//Si el proceso Destino es menor que el Proceso Origen, no se insertan procesos y se genera
//mensaje de error de traslape de procesos
if(getNumeroProceso(elementoFinal)<getNumeroProceso(cadenaAux))
{
cadenaDiagrama+="\nLa actividad \" "+tablaDiagrama.selectedItem.Nombre+"\" relacionada con el
proceso Moprosoft \""+tablaDiagrama.selectedItem.Rela+"\" debe realizarse antes de la actividad
\" "+nombreAnterior+"\" relacionada con el proceso Moprosoft \" "+procesoAnterior+"\" \n";
insertaProcesos=false;
}
//Se genera mensaje de error que no existen actividades relacionados con uno o más procesos
//Moprosoft
cadenaDiagrama+="\nEntre las actividades: \" "+tablaDiagrama.selectedItem.Nombre+"\" relacionada
con el proceso Moprosoft \""+tablaDiagrama.selectedItem.procRela+"\" y  \""+nombreAnterior+"\"
relacionada con el proceso Moprosoft \" "+procesoAnterior+"\", no  existen actividades
relacionadas con los siguientes procesos Moprosoft: \n";
//Inserta los procesos Moprosoft que hacen falta en el diagrama actual denotando que son
//actividades nuevos.
if(insertaProcesos)
do{
        diagramaMoprosoft.selectedIndex=indice;
        elementoRich=new Object();
        elementoRich.id_Elemento=diagramaMoprosoft.selectedItem.proceso;
        elementoRich.tipodesicion=diagramaMoprosoft.selectedItem.decision;
        elementoRich.procRela=diagramaMoprosoft.selectedItem.proceso;
        elementoRich.estado=3;
        diagramaRich.addItem(elementoRich);
        diagramaMoprosoft.selectedIndex=indice;
        cadenaDiagrama+="\""+diagramaMoprosoft.selectedItem.proceso+"\"\n";
   }while(diagramaMoprosoft.selectedItem.proceso!=elementoFinal);
}
```

9. Anexo D.- Pruebas del Sistema mediante la evaluación de una empresa ficticia.

El sistema de pruebas se llevo a cabo mediante la realización de las actividades para el establecimiento de una iniciativa SPI. La Figura 9.1 muestra la realización de las actividades generales, es decir actividades que tienen que ver con dar de alta una empresa, dar de alta un nuevo usuario, iniciar sesión y acceder a las funcionalidades de la herramienta SelfVation.

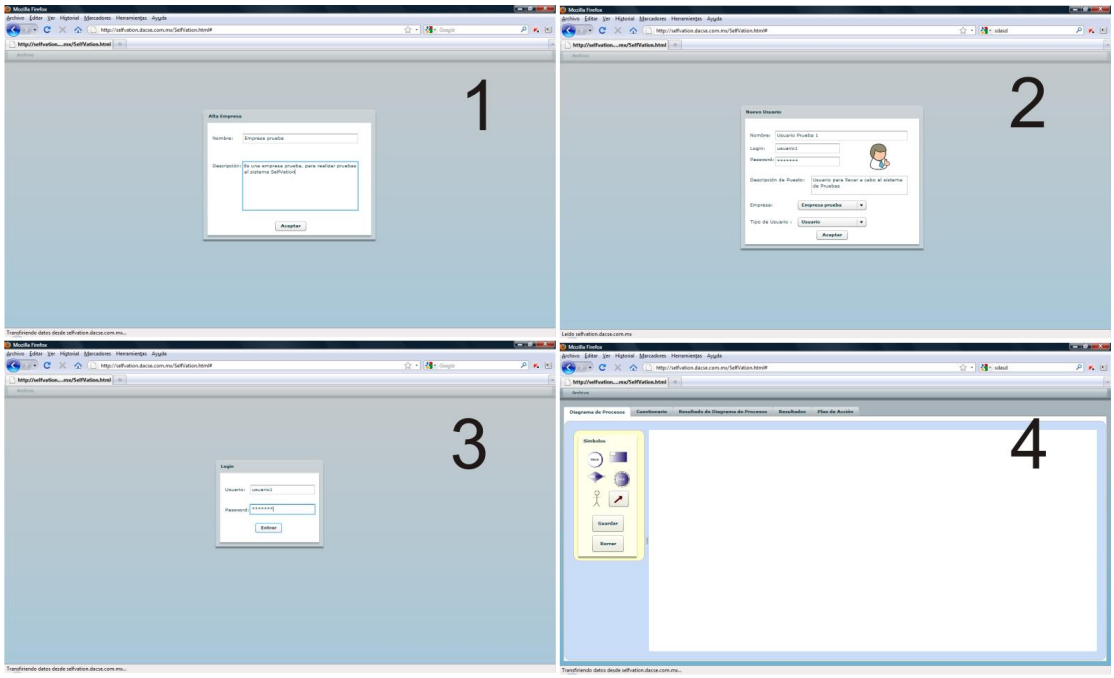


Figura 9.1. Desarrollo de tareas relacionadas con el acceso a la herramienta.

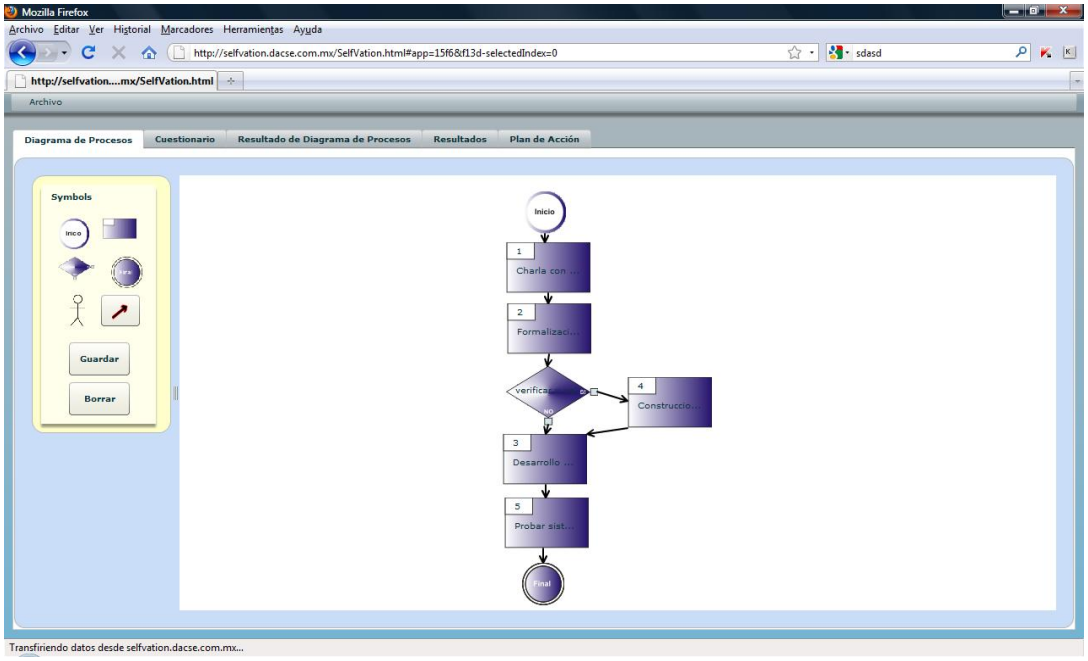


Figura 9.2. Creación de un diagrama con las características descritas en el sistema de prueba.

La Figura 9.2 muestra un diagrama prueba el cual cumple con las características propuestas por el sistema de prueba presentado en la sección 3.3.6.

La siguiente actividad en el sistema de pruebas, fue contestar el cuestionario propuesto por la herramienta de una manera que se pudiese verificar que los resultados generados eran correctos. El cuestionario se contestó de la manera propuesta en el sistema de pruebas, dicha tarea se muestra en la Figura 9.3.

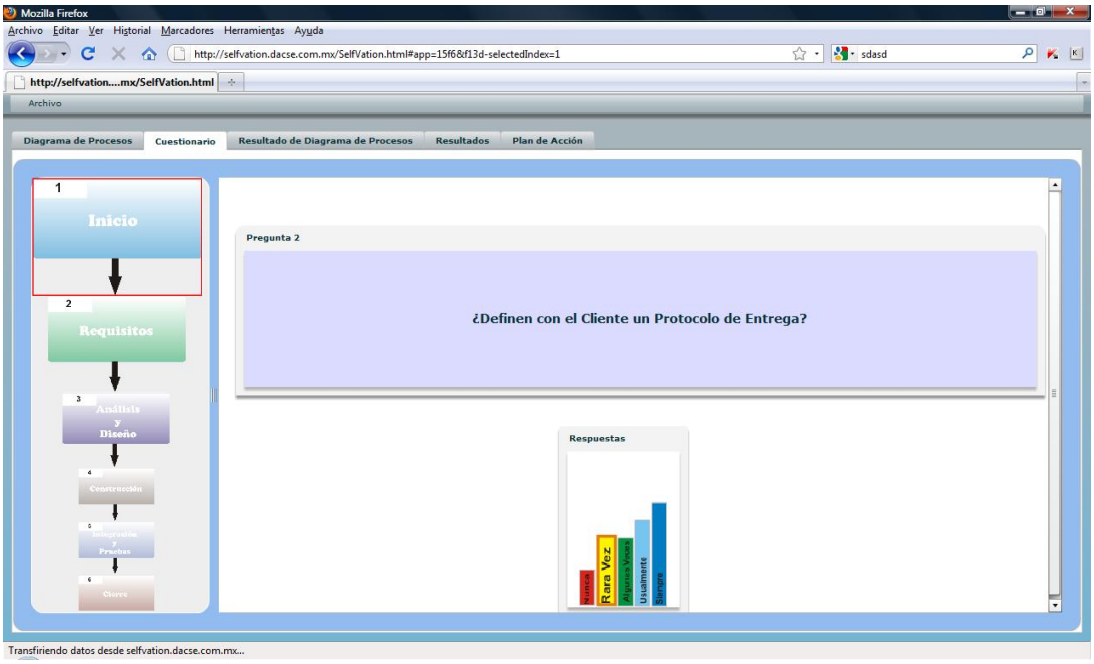


Figura 9.3. Prueba de la evaluación por cuestionario.

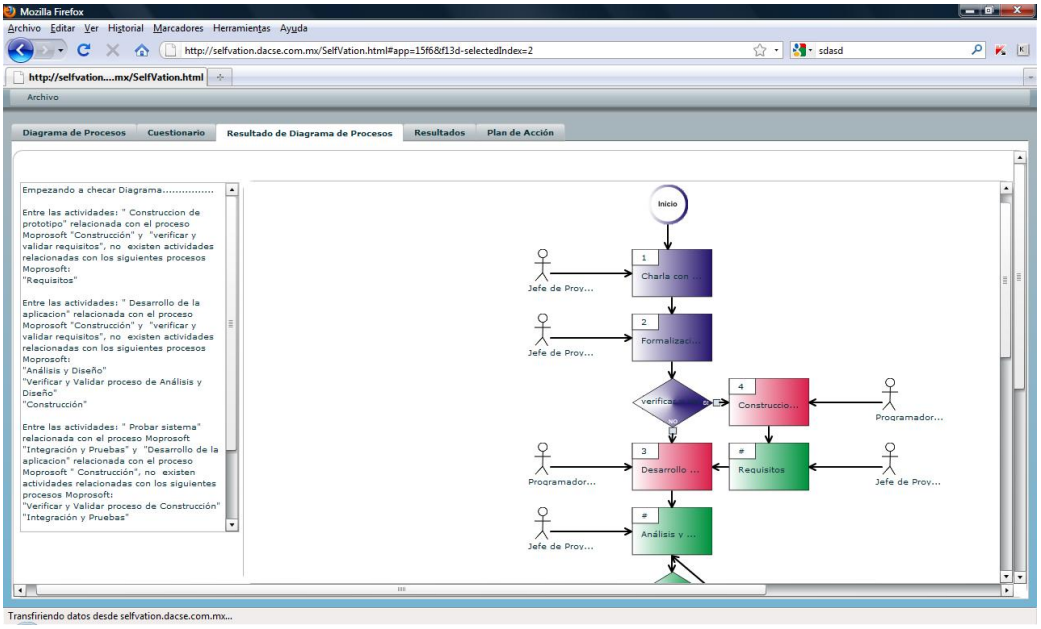


Figura 9.4. Prueba de la fase resultados Diagrama.

La Figura 9.4 muestra el diagrama enriquecido obtenido por SelfVation a partir del diagrama actual del proceso de desarrollo modelado por la empresa ficticia. En la Figura 9.4 se puede observar un diagrama más robusto que el editado en la Figura 9.2, esto se debe a que se siguieron las definiciones marcadas en la sección 3.3.1.2. En el diagrama enriquecido se pueden observar nuevas actividades propias del modelo Moprosoft que no se están llevando a cabo en el proceso actual de desarrollo de la empresa ficticia, estas actividades se marcan con un color verde, también se marcan de manera diferente, en este caso color rojo las actividades que no cumplen con los características propuestas por el diagrama ideal Moprosoft. Además, se añaden los roles dentro de la empresa que deben participar dentro del desarrollo de la actividad.

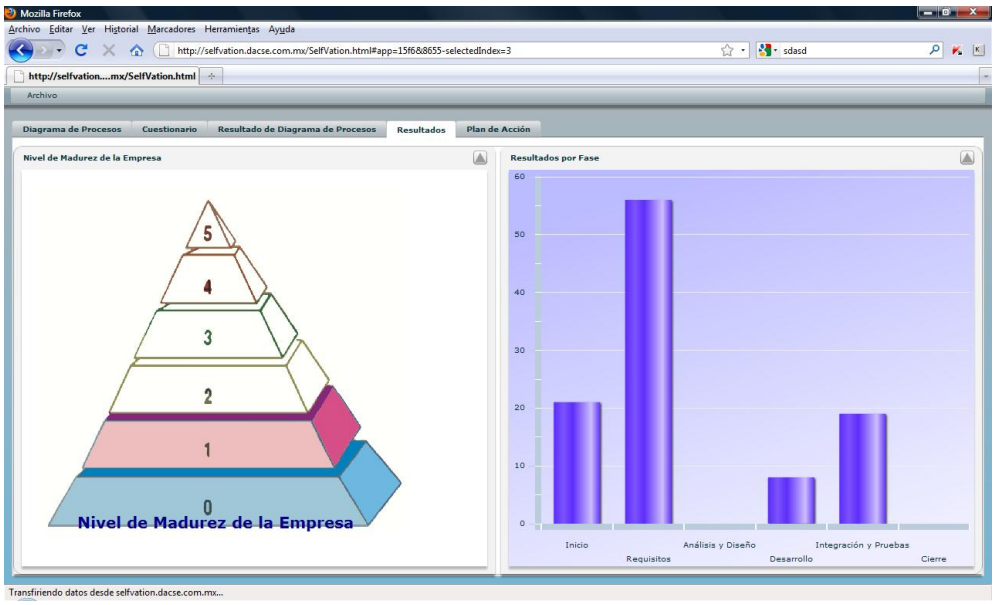


Figura 9.5. Resultados obtenido para el sistema de prueba.

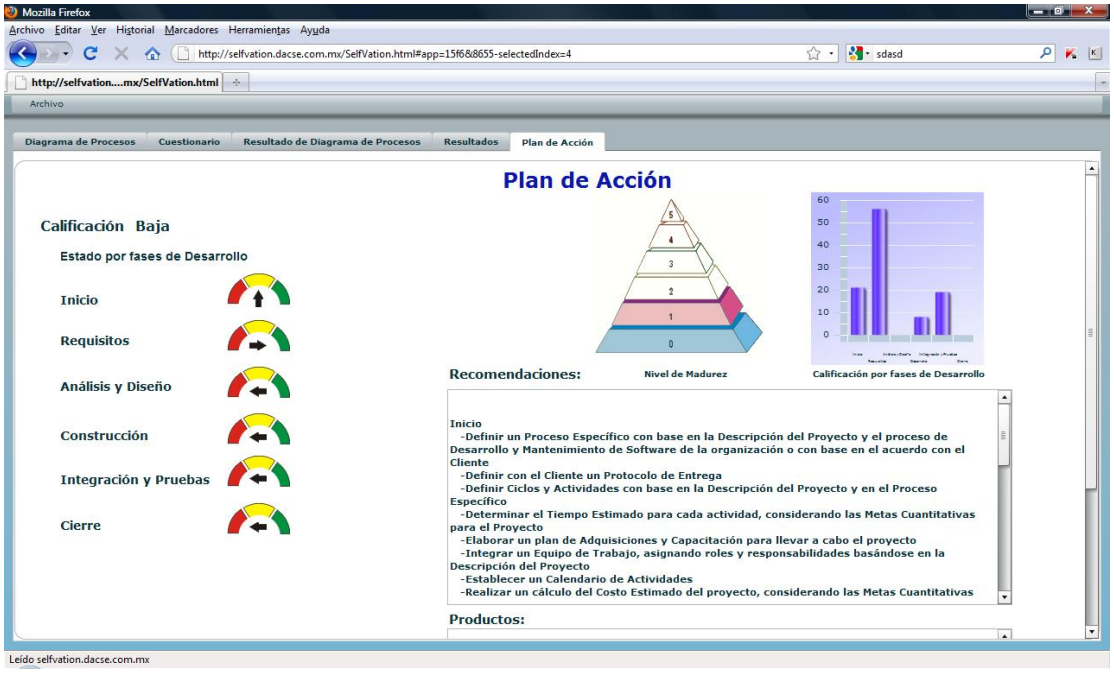


Figura 9.6. Plan de acción generado para el sistema de prueba

Los resultados y el plan de acción generados en base a la etapa de evaluación se muestran en las Figuras 9.5 y 9.6 respectivamente. En ellos se puede observar el trabajo conjunto de los dos métodos de evaluación. Lo anterior se demuestra de manera más clara en la gráfica presentada en ambas figuras y la calificación por fases mostrada en la Figura 9.6 (lado izquierdo). Por ejemplo, las calificaciones de las etapas de “Inicio” y “Requisitos” fueron respetadas, ya que en la evaluación por modelado se detectó que las actividades relacionadas con estas fases se llevan a cabo de manera correcta en la empresa. En el caso de la actividad “Análisis y Diseño” los resultados del cuestionario se anularon, debido a que en la evaluación por modelado se detectó que la empresa carece de actividades relacionadas con este proceso de Moprosoft. Las calificaciones en la fase de “Construcción” fueron penalizadas, ya que aunque en el proceso de modelado se detectaron actividades relacionadas con esta fase Moprosoft, estas actividades no se llevaban a cabo de manera correcta.

Estos ajustes en los resultados de las fases de desarrollo afectan de manera significativa la obtención del nivel de madurez de la empresa, también inciden en la producción de las actividades y productos en las cuales la empresa debe poner énfasis para llevar a cabo un ciclo de mejora.

10. Anexo E. – Actas de Publicaciones

Supporting the Management Process of Software Process Improvement Initiatives based on NMX-I-059/02-NYCE-2005

Garcia, I. & Cruz, D.

Postgraduate Department
Technological University of the Mixtec Region. Mexico
{ivan@mixteco.utm.mx, dago@mixteco.utm.mx}

ABSTRACT. Nowadays there are models and standards which attempt to introduce quality in the enterprises' software development process with the objective to introduce high quality levels in the produced software. The NMX-I-059/02-NYCE-2005 standard (also known as MoProSoft) is focused on small and medium software enterprises, or small groups of software development within a larger organization, with the aim of promoting the standardization of an effective process in the software industry. Mexican enterprises now have a software standard that enables them to achieve a high level of quality in the software that they produce. However, the adoption of any standard is not an easy task. This paper aims to show that the development and implementation of a RIA-based tool that could support improvement initiatives, therefore strengthening the standard adoption.

KEYWORDS: Software process improvement, effort and adoption time, MoProSoft, small software enterprises, process assessment and improvement.

1 Introduction

Software has arisen as a fundamental pillar in the evolution of computational products and services. In the last two decades, software has changed from a "specific problem solution" to an autonomous industry [11]. However, this change keeps the old same problems since the "software crisis" in 1969. Nowadays, the quantity and quality demands dominate the market. According to the last report of Standish Group Inc. [38]:

- Software is (almost) always delivered out of the initial planning,
- Software is more expensive than the original cost,
- Software has a different functionality.

Pressman says that: *"the majority of software crisis causes have their origins in myths and theories that arose in the early years of Software Engineering"* This origin makes the myths more dangerous; but the truth is that they do not look like myths any more. Figure 1 shows that a recent study [39] establishes that software does not accomplish the original requirements because: 45% of software exceeds the cost, 63% of software exceeds the planned schedule, and software fulfills the 67% of the required functionality.

11. Referencias Bibliográficas

[**Adams04**] Adams, R., Eslinger, S.., Owens, K. & Rich, M. A. "Software Acquisition Best Practices 2004 edition". *Proc. of the 3° OSD Conference on the Acquisition of Software-Intensive Systems*, The Aerospace Corporation, January, 2004.

[**Brodman99**] Brodman, J. & Johnson, D. "*Project Planning: Disaster Insurance for Small Software Projects*". LOGOS International, Inc. Proceedings from SEPG 2000: Ways to Make Better Software. Seattle, Washington. March, 1999.

[**Brown08**] Brown, C. *The Essential Guide to Flex 3*. Ed. Friends of ED. 2008.

[**Caballero05**] Caballero-De la Villa, D. "*Manejador de Documentos de MoProSoft*". Tesis Licenciatura: Universidad de las Américas Puebla. Mayo, 2005.

[**Cárdenas06**] Cárdenas, E. "*Herramienta de Guía y Supervisión para el uso automatizado del Modelo de Procesos Moprosoft*". Memorias del III Encuentro: Participación de la Mujer en la Ciencia. 2006.

[**Cárdenas06a**] Cárdenas, E. "*Herramienta de Guía y Supervisión para el uso automatizado del Modelo de Procesos Moprosoft*". Tesis de Maestría: Universidad Autónoma de México.2006.

[**Cater-Steel04**] Cater-Steel, A.P. "Low-rigour, Rapid Software Process Assessment for Small Software Development Firms" *Proc. of the 2004 Australian Software Engineering Conference* (ASWEC'04), IEEE Computer Society, pp. 368-377, 2004.

[**CMMI06**] CMMI Product Team. *CMMI® for Development, Version 1.2. Improving processes for better products*, August 2006. CMU/SEI-2006-TR-008, ESC-TR-2006-008.

[**COBIT05**] IT Governance Institute. *COBIT 4.0*. Released by the COBIT Steering Committee and the IT Governance Institute. ISBN 1-933284-37-4.

[**Davis08**] Davis, M. & Phillips, J. *Flex 3: A Beginner´s Guide*. McGraw- Hill Companies. 2008.

[De la Villa04] De la Villa, M., Ruiz, M. & Ramos, I. "*Modelos de evaluación y mejora de procesos: análisis comparativo"* Ceur Workshop Proceedings (Online). Vol. 120. 2004.

[**Deming86**] Deming, W. E. *Out of the Crisis: Quality, Productivity and Competitive Position*. Cambridge, MA: Cambridge University Press, 1986.

[**El Emam97**] El Emam, K. & Briand, L. "*Costs and Benefits of Software Process Improvement*" International Software Engineering Network, Technical Report 047.97/E, Fraunhofer, 1997.

[**ESANE04**] ESANE, Consultores S.C & Secretaría de Economía. "*Perfil de la Industria Mexicana del Software y Servicios Relacionados*". Secretaría de Economía del Gobierno Mexicano, Fase 1/Criterio 2, 2004.

[**Farré05**] Farré, X. & Messeguer, R. *Rich Internet Applications*. Universidad Politécnica de Cataluña. Julio, 2005.

[**Flores08**] Flores, B., Astorga, M., Olguín, J. & Andrade, M. "Experiences on the Implementation of MoProSoft and Assessment Processes under the NMX-I-059/02-NYCE-2005 Standard in a Small Software Development Enterprise" *Proc. of the 2008 Mexican International Conference on Computer Science*, IEEE Computer Society, pp. 323-328, 2008.

[**Fowler99**] Fowler, P., Middlecoat, B. & Yo, S. "*Lessons Learned Collaborating on a Process for SPI at Xerox* (CMU/SEI-99-TR-006, ADA373332)". Pittsburg, PA: Software Engineering Institute, Carnegie Mellon University. 1999.

[**Galliers92**] Galliers, R. *Information Systems Research: Issues, Methods and Practical Guideline*. Alfred Waller Ltd. Chippenham, Wiltshire. England, 1992.

[**Garcia07**] Garcia, I., Calvo-Manzano, J., Cuevas, G. & San Feliu, T. "Determining Practice Achievement in Project Management Using a Two-Phase Questionnaire on Small and Medium Enterprises" *Proc. of the 2007 European Systems and Software Process Improvement and Innovation Conference* (EUROSPI 2007), Springer-Verlag Berlin Heidelberg, LNCS 4764, pp. 46-58, 2007.

[**Gillham00**] Gillham, B. *Developing a Questionnaire*. London; New York: Continuum. 2000.

[**González06**] González, D. "*Estudio Exploratorio de los Factores Críticos de Éxito de la Industria Mexicana del Software y su Relación con la Orientación Estratégica del Negocio*". Departamento de Organización de Empresas/ITIO/Universidad Politécnica de Valencia, España. Febrero, 2006.

[**Goralski08**] Goralski, G. & Leon, L. *Flex for Designers*. Friends of ED. 2008.

[**Hareton01**] Hareton, L. & Terence, Y. "A process framework for small projects" *Software Process: Improvement and Practice*, 6(2): 67-83. 2001.

[**Hall02**] Hall, T., Rainer, A. & Badoo, N. "Implementing Software Process Improvement: An Empirical Study". *Software Process: Improvement and Practice*, 7(1): 3-15. 2002.

[**Hunter01**] Hunter, R. & Thayer, R. *Software Process Improvement*. Institute of Electrical and Electronic Engineers, IEEE Computer Society, 2001.

[**IEEE 729-1983**] Glossary of Software Engineering Terminology - Redesignated as IEEE 610.12, Institute of Electrical and Electronics Engineers.

[**ISO00**] International Organization for Standardization. ISO 9001:2000. *Quality management systems – Requirements:* Geneva, 2000.

[**ISO02**] International Organization for Standardization. ISO/IEC 12207:1995. *Information Technology Software Life Cycle Processes*. Amd.1, 2002.

[**ISO04**] International Organization for Standardization.ISO/IEC 15504-2:2003/Cor.1:2004(E): *Information Technology – Process Assessment – Part 2: Performing an Assessment*. Geneva 2004.

[**Koch07**] Koch, N., Knapp, A., Zhang, G. & Baumeister, H. "*UML-Based Web Engineering: An Approach Based on Standards"*. Web Engineering: Modeling and Implementing Web Applications. HCI Series, Springer-Verlag, 2007.

[**Koch07a**] Koch, N., Kraus, A., & Hennicker, R. "*The Authoring Process of the UML-based Web Engineering Approach*". Institute of Computer Science Ludwing-Maximilians University of Munich. 2002.

[**Kotonya00**] Kotonya, G. & Sommerville, I. *Requirements Engineering: process and techniques*. John Wiley and Sons, 2000.

[**Linaje07**] Linaje, M., Preciado, J. C. & Sánchez-Figueroa, F. "Engineering Rich Internet Application User Interfaces over Legacy Web Models". *Internet Computing Magazine IEEE*, 11(6): 53-59. 2007.

[**Loon04**] Loon, H. *Process Assessment and ISO/IEC15504: A Reference Book*. The Kluwer International Series in Engineering and Computer Science, Springer. May, 2004.

[**Martinez-Ruiz06**] Martinez-Ruiz, F., Muñoz, J., Vanderdonckt, J., González-Calleros, J & Mendoza, R. "A first draft of a Model-driven Method for Designing Graphical User Interfaces of Rich Internet Applications" *Proc. of the Fourth Latin American Web Congress* (LA WEB'06), IEEE Computer Society, pp. 32-38, 2006.

[**Mutafelija03**] Mutafelija, B. & Stromberg, H. *Exploring CMMI-ISO 9001:2000 synergy when developing a process improvement strategy*. Boston, MA. Artech House Computing Library. 2003.

[**NYCE05**] Normalización y Certificación Electrónica, A. C. Tecnología de la Información – Software – Modelos de Proceso y Evaluación para Desarrollo y Mantenimiento de Software – Parte 01: Definición de conceptos y productos. NMX-I-059/01-NYCE-2005.

[**NYCE05a**] Normalización y Certificación Electrónica, A. C. Tecnología de la Información – Software – Modelos de Proceso y Evaluación para Desarrollo y Mantenimiento de Software – Parte 02: Requisitos de Procesos (MOPROSOFT). NMX-I-059/02-NYCE-2005.

[**NYCE05b**] Normalización y Certificación Electrónica, A. C. Tecnología de la Información – Software – Modelos de Proceso y Evaluación para Desarrollo y Mantenimiento de Software – Parte 04: Guía para la evaluación de procesos (EVALPROSOFT). NMX-I-059/04-NYCE-2005.

[**Ocaña04**] Ocaña, J. & Rossainz, M. "*Introducción a la Ingeniería Web Basada en UML*". Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación. 2004.

[**OECD02**] Organization for Economic Co-Operation and Development. "*Small and Medium Enterprise Outlook*". OECD, 2002.

[**Oktaba05**] Oktaba, H. "Historia de una Norma: MoProSoft y sus Primeros Pasos" *Revista Software Gurú*, 1(8): 6. May-Jun, 2005.

[**Oktaba06**] Oktaba, H. "MoProSoft: A Software Process Model for Small Enterprises" *Proc. of the First International Research Workshop for Process Improvement in Small Settings*, Software Engineering Institute, Carnegie Mellon University, pp. 93-101, 2006.

[**Oktaba07**] Oktaba, H., Garcia, F., Piattini, M., Ruiz, F., Pino, F. & Alquicira, C. "Software Process Improvement: The Competisoft Project" *Computer*, 40(10): 21-28. 2007.

[**Paulk95**] Paulk, M. C. *The Capability Maturity Model Guidelines for Improving the Software Process*. MA: Addison-Wesley. 1995.

[**Paulk02**] Paulk, M. C. "*Comparing ISO 9001:2000 and software CMM v1.1*". September, 2002.

[**Pérez08**] Pérez, M. & Messeguer, R. "*Evaluación y prueba de aplicaciones RIA con AJAX*". Universidad Politécnica de Cataluña. Abril, 2008.

[**Pino06**] Pino, Francisco J. "Revisión sistemática de mejora de procesos software en micro, pequeñas y medianas empresas". *Revista Española de Innovación, Calidad e Ingeniería del Software*, 2(1): 5-13.2006.

[**Pino07**] Pino, Francisco J., García, F. & Piattini, M. "Herramienta de Soporte a la Valoración Rápida de Procesos Software". *IEEE Latin America Transactions*. Vol. 5(4). July, 2007.

[**PMI04**] Project Management Institute. *A Guide to the Project Management Body of Knowledge (PMBoK Guide)*. Project Management Institute. 2004.

[**Preciado05**] Preciado, J. C, Linaje, M., Sanchez, F. & Comai, S. "Necessity of methodologies to model Rich Internet Applications". *Proc. of the Seventh IEEE International Symposium on Web Site Evolution* (WSE'05), IEEE Computer Society, pp. 7-13. 2005.

[**Preciado08**] Preciado, J. C, Linaje, M., Morales-Chaparro, R., Sanchez-Figueroa, F., Zhang, G., Kroiβ, C. & Koch, N. "Designing Rich Internet Applications Combining UWE and RUX-Method". *Proc. of the Eighth International Conference on Web Engineering* (ICWE'08), IEEE Computer Society, pp. 148-154, 2008.

[**Preciado08a**] Preciado, J.C, Linaje, M. & Sanchez-Figueroa, F. "*Adapting Web 1.0 User Interfaces to Web 2.0 Multidevice User Interfaces using RUX-Method*". Journal of Universal Computer Science, vol. 14, no. 13. 2008.

[**Pressman02**] Pressman, R. S. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, pp. 8-10. 2002.

[**Reyes08**] Reyes, P. Y., Margarin, M. L., Álvarez, F. & Muñoz, J. "*Diseño de un Instrumento de Auto-evaluación para Diagnosticar el Estatus de las Organizaciones en México con Respecto al Modelo ProSoft: Proceso de Gestión de Procesos de la Categoría Gestión*". Universidad Politécnica de Aguascalientes. 2008.

[**Reyes09**] Reyes, P. Y., Margarin, M. L., Álvarez, F. & Muñoz, J. "Aplicación de instrumento de diagnóstico en proceso "gestión de procesos" con base en MoProSoft". *Revista Investigación y Ciencia de la Universidad Autónoma de Aguascalientes*, 1(43). Enero-Abril, 2009.

[**Russell92**] Russell, C. & Bobko, P. "Moderated Regression Analysis and Likert Scales too Corse Comfort". *Journal of Applied Psychology*, 77(3): 336-342. 1992.

[**SCAMPI06**] Members of the Assessment Method Integrated Team. Standard CMMI Appraisal Method for Process Improvement (SCAMPI), Version 1.1, CMU/SEI-2001-HB-001. Software Engineering Institute, Carnegie Mellon University. 2006.

[**SE01**] Secretaria de Economía. "*Plan de Desarrollo Nacional 2001-2006*". Gobierno de los Estados Unidos Mexicanos. 2001.

[**SE04**] Secretaría de Economía. "*Estudio del nivel de madurez y capacidad de procesos de la industria de tecnologías de información en el área metropolitana de Monterrey, Nuevo León y el Distrito Federal y su área metropolitana*". Secretaría de Economía del Gobierno Mexicano, 2004.

[**Serrano06**] Serrano, M., Montes de Oca, C. & Cedillo, K. "An experience on Implementing the CMMI in a Small Organization Using the Team Software Process" *Proc. of the First International Research Workshop for Process Improvement in Small Settings*, Software Engineering Institute, Carnegie Mellon University, pp. 81-92, 2006.

[**Sommerville05**] Sommerville, I. *Software Engineering*. Seventh Ed. Addison Wesley. 2005.

[**SPICE07**] SPICE. Software Process Assessment Part 1 and Part 2 V1.00. ISO/IEC Software Process Assessment. http://www.sqi.gu.edu.au/spice/

[**Standish04**] The Standish Group International. "Chaos Extreme". 2004.

[**Standish06**] The Standish Group International. "Chaos Extreme". 2006.

[**Standish08**] The Standish Group International. "My Life is Failure & CHAOS Summary 2008". Copyright 2006.

[**Strevel05**] Strevel, C. "*Kuali: Herramienta Auxiliar para implementación de Moprosoft*". DevDays, Intellekt. 2005.

[**Tapper09**] Tapper, J., Labriola, M. Boles, M. & Talbot, J. *Adobe Flex 3*. Edición en Español por Ediciones Anaya Multimedia. 2009.

[**Valenzuela07**] Valenzuela, L. & Flores, B. "*Especificación Formal de Elementos MoProSoft a partir del Modelo de Referencia de Flujos de Trabajo*". Universidad Autónoma de Baja California. 2007.

[**Vargas07**] Vargas, E., Oktaba, H., Guardati, S. & Laureano, A. "Agents, Case-Based Reasoning and their relation to the Mexican Software Process Model (MoProSoft)" *Proc. of the 31st Annual International Computer Software and Applications Conference* (COMPSAC), IEEE Computer Society, pp. 326-334, 2007.

[**Walker03**] Walker, E: "*Implementing Best Practices in the Join Battlespace Infosphere (JBI) program at AFRL"*. Conference on the Acquisition of Software-Intensive System, Carnegie Mellon University, January 28-30, 2003.

[**Yamanishi02**] Yamanishi, K. & Li, H. "Mining Open Answers in Questionnaire Data". *IEEE Intelligent Systems*. 17(5): 58-65. 2002.

[**Zurita05**] Zurita-Rendón, H. "*Arquitectura de la Herramienta Integral para MoProSoft*". Tesis de Maestría: Universidad Nacional Autónoma de México. 2005.

11.1. Sitios de Internet

[**URL-1**] http://www.esi.es/

Instituto Europeo de Software (Último acceso: Septiembre 2009).

[**URL-2**] http://www.fidsoftware.org/

Organismo rector de la estrategia de la industria de TI en el estado de Sinaloa (Último acceso: Octubre 2009).

[**URL-3**] http://pnd.fox.presidencia.gob.mx/

Plan Nacional de Desarrollo 2001-2006 (Último acceso: Octubre 2009).

[**URL-4**] http://www.economia.gob.mx/

Secretaria de Economía (Último acceso: Octubre 2009).

[**URL-5**] http://www.prosoft.economia.gob.mx/

Programa para el Desarrollo de la Industria Software (Último acceso: Octubre 2009).

[**URL-6**] http://www.cmm-quest.com/

Self Assessment Tool CMM-Quest v1.2 (Último acceso: Octubre 2009).

[**URL-7**] http://www.isd-inc.com/

Appraisal Wizard, Formal or informal appraisal tool (Último acceso: Octubre 2009).

[**URL-8**] http://www.spice121.com/

SPiCE 1-2-1 (Último acceso: Octubre 2009).

[**URL-9**] http://www.man-info-systems.com/

IME Toolkit (Último acceso: Octubre 2009).

[**URL-10**] http://www.adobe.com/es/resources/business/rich_internet_apps/

Adobe: Rich Internet Applications (Último acceso: Noviembre 2009).

[**URL-11**] http://www.ajax.org/

Asynchronous Javascript and XML (Último acceso: Noviembre 2009).

[**URL-12**] http://www.adobe.com/es/products/flash/

Adobe Flash (Último acceso: Noviembre 2009).

[**URL-13**] http://www.adobe.com/es/products/flex/
Adobe Flex (Último acceso: Noviembre 2009).
[**URL-14**] http://windowsclient.net/
Windows Presentation Foundation (Último acceso: Noviembre 2009).
[**URL-15**] http://silverlight.net/
Microsoft Silverlight (Último acceso: Noviembre 2009).