



UNIVERSIDAD POPULAR AUTÓNOMA DEL ESTADO DE PUEBLA

CENTRO INTERDISCIPLINARIO DE POSGRADOS
INVESTIGACIÓN Y CONSULTORÍA
DEPARTAMENTO DE INGENIERÍA

DOCTORADO EN TECNOLOGÍAS DE INFORMACIÓN
Y ANÁLISIS DE DECISIONES

METODOLOGÍA MDSIC-M PARA EL DESARROLLO DE APLICACIONES MÓVILES EN EMPRESAS PYMES

T E S I S

QUE PARA OBTENER EL GRADO DE:

DOCTOR EN TECNOLOGÍAS DE
INFORMACIÓN Y ANÁLISIS DE DECISIONES

P R E S E N T A :
HEBERTO FERREIRA MEDINA

DIRECTOR:
DR. CARLOS ARTURO VEGA LEBRÚN

PUEBLA, MÉXICO

JULIO 2015



UPAEP – Secretaría General

Dirección General de Apoyos Académicos

Dirección del Centro de Recursos para el Aprendizaje y la Investigación.

Biblioteca Central - **Karol Wojtyla**

Tesis Digitales Restricciones de uso:

DERECHOS RESERVADOS ©

PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de textos, imágenes, gráficas, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente de donde la obtuvo mencionando el autor o autores involucrados en el documento.

Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.



UNIVERSIDAD POPULAR AUTÓNOMA DEL ESTADO DE PUEBLA

CENTRO INTERDISCIPLINARIO DE POSGRADOS
INVESTIGACIÓN Y CONSULTORÍA
DEPARTAMENTO DE INGENIERÍA

DOCTORADO EN TECNOLOGÍAS DE INFORMACIÓN
Y ANÁLISIS DE DECISIONES

METODOLOGÍA MDSIC-M PARA EL DESARROLLO DE APLICACIONES MÓVILES EN EMPRESAS PYMES

T E S I S

QUE PARA OBTENER EL GRADO DE:

DOCTOR EN TECNOLOGÍAS DE
INFORMACIÓN Y ANÁLISIS DE DECISIONES

P R E S E N T A :

HEBERTO FERREIRA MEDINA

DIRECTOR:

DR. CARLOS ARTURO VEGA LEBRÚN

COASESORES:

DRA. MARÍA DEL ROCÍO GUADALUPE MORALES SALGADO

DR. ERICK DE LA BARRERA MONTPELLIER

DR. JOSÉ LUIS CENDEJAS VALDEZ

**CUERPO ACADÉMICO DE TRANSFERENCIA TECNOLÓGICA PARA LA
CONSTRUCCIÓN DE SOFTWARE**

PUEBLA, MÉXICO

JULIO 2015



Universidad Popular Autónoma del Estado de Puebla
Centro Interdisciplinario de Posgrados
Investigación y Consultoría
Departamento de Ingeniería
Doctorado en Tecnologías de Información
y Análisis de Decisiones

Se aprueba la Tesis:

Nombre de la tesis

Metodología MDSIC-M para el desarrollo de aplicaciones móviles
en empresas PYMES

Nombre del Alumno

MTI. Heberto Ferreira Medina

Dr. Carlos Arturo Vega Lebrún



Director(a) de Tesis

Ma. del Rocío G. Morales Salgado



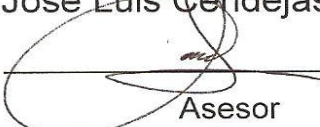
Asesor

Dr. Erick de la Barrera M.



Asesor

José Luis Gondejas Valdez



Asesor

Puebla, México.

26 de Junio de 2015

Agradecimientos

A mi familia por su paciencia y apoyo incondicional; a mi esposa Gaby por su cariño y comprensión, a mis hijas Tsuni e Itsi por ser la alegría de nuestro hogar. Las quiero.

A mis padres y hermanos por su gran ánimo y compañía; a mi mamá Marisela por sus enseñanzas y cariño, a mi papá Ismael por sus conocimientos y cariño, a mis hermanos Elizabeth, Joel e Ismael por su apoyo y confianza.

A la familia Zavala Acosta con quienes comparto experiencias de vida; a la señora Ana, al señor Macario, a mis grandes amigos y cuñados; Miriam, Rocío, Juan Carlos, Francisco y Mario. Gracias por su compañía y apoyo.

A mis asesores que me dirigieron durante mis estudios y en el desarrollo de este proyecto:

Al Dr. Carlos Arturo Vega Lebrún, por su gran apoyo, confianza y experiencia.

A la Dra. Ma. del Rocío Guadalupe Morales Salgado, por sus atinados comentarios, ayuda y apoyo incondicional.

Al Dr. José Luis Cendejas Valdéz, por su gran ayuda, guía, compañerismo y sobretodo su invaluable amistad.

Al Dr. Erick de la Barrera Montppellier por su gran apoyo, confianza y amistad.

A mis amigos y colegas del Instituto Tecnológico de Morelia, por sus conocimientos y ánimo de trabajar; G. Lúa, M. Zulma, Y. Vega, K. Ayala, R. Vargas, C. Martínez, R. Contreras, L. Alvarado, B. González, M. Blancas, R. Ferreira, A. Antolino, H. Fernández, J. Mora, J. García, F. Morales, R. Guzmán, J. Sánchez, J. Ruiz, F. Villaseñor, I. Aguilar, O. Ortiz, C. Torres, E. Pérez y D. Acosta. Gracias.

A mis grandes amigos de la UNAM, Campus Morelia, por compartir su experiencia y conocimientos; A. López, A. Valencia, H. Zavala, M. Espejel, R. Ahedo, A. Ginori, G. Zavala, F. Hernández, P. García, M. Magaña, E. Espinosa y J. Martínez.

Al consejo Interno del CIEco ahora IIES, por su confianza y apoyo. A las(os) doctoras(es); E. Andresen, M. Gavito, P. Balvanera, E. del Val, A. Castillo, J. Benítez, P. Ávila, A. Casas, M. Martínez, F. Espinosa, V. Jaramillo, M. Maass, H. Arita, K. Oyama, M. Quesada, V. Toledo, E. de la Barrera, A. González, F. García, J. Larsen, G. Ibarra, R. Lindig, O. Masera, H. Paz, D. Pérez, V. Arroyo, J. Schondube, E. Vega y G. Bocco. Gracias.

A todas(os) mis compañeras(os) y amigas(os) que me apoyaron durante mis estudios del Posgrado en la UPAEP, por todas las largas horas de discusión al compartir ideas y en el desarrollo de proyectos.

*A los clústeres de tecnología por su apoyo; AIETIC, ClusterTIM, Cluster ITESI, InoovaTiA e IJALTI.
Por difundir este proyecto.*

*A los Institutos Tecnológicos de: Irapuato, Morelia y Nayarit, a la Universidad Tecnológica de Morelia
y a la Universidad de Morelia. Por su ayuda.*

*A los Institutos y Centros de Investigación de la UNAM, por su gran apoyo; ENES-Morelia,
CIEco-IIES, CIGA y CRyA.*

*A las PyMEs de la región por su gran ayuda. En especial al Ing. Juan Márquez e Ing. David Jáuregui.
Gracias por su confianza.*

*A los maestros Atzimba G. López M., Alberto Valencia G., Hugo A. Zavala V. , Miguel Espejel C. y Raúl
Ahedo H. por su ayuda técnica y calidad en el desarrollo de este trabajo. Muchas gracias.*

*A mis maestros y asesores del Posgrado de la UPAEP por sus recomendaciones y comentarios
atinados: Dra. Anayasi Careta, M.C. Enrique Sánchez, Dr. Héctor Meneses, Dra. Ma. Guadalupe López,
Dr. Manuel González y Dr. José P. Nuño.*

*A mis alumnos del Instituto Tecnológico de Morelia, de la Universidad de Morelia y de la
ENES-UNAM, por ayudarme a mejorar día a día y mantenerme siempre con los pies en la tierra.
Muchas gracias*

Resumen

La demanda en el mercado para el desarrollo de aplicaciones que se ejecutan en dispositivos móviles (APP), ha fomentado un auge en el desarrollo de software. Se observa que las personas utilizan cada vez más estos dispositivos. Por ello, existe la necesidad de mejorar e innovar en la construcción de una APP, las cuales deben ser más rápidas y fáciles de elaborar.

Este mercado sigue creciendo y se está convirtiendo en una prioridad de los usuarios. Se puede observar en la literatura la ausencia de entornos de desarrollo completos para los programadores de esta industria, los cuales demandan mejores prácticas y métodos. En México se reconoce la labor del gobierno federal por impulsar clústeres de tecnologías de información en diferentes regiones del país, así como programas de apoyo para incentivar la industria.

La necesidad de los consumidores de adquirir aplicaciones móviles de calidad complican a las empresas que compiten en esta industria, los estándares y la Norma Oficial Mexicana (NOM) se vuelven un referente importante para la medición de la calidad.

En esta tesis se propone el diseño de la metodología MDSIC-M (Modelo de Desarrollo de Software Integral Colaborativo para dispositivos Móviles) la cual está orientada al desarrollo rápido de aplicaciones móviles y que atiende a las necesidades de las pequeñas y medianas empresas (PyMEs). Finalmente como ventaja de la metodología propone almacenar historias de proyectos, en donde el tamaño el tiempo, los costos y los riesgos que ocurrieron durante su desarrollo son almacenados y documentados, permitiendo a una empresa el consultar y no repetir errores. Ayuda a mejorar el proceso de construcción, el reúso de componentes y promueve la calidad. Ofreciendo con estas características ser una herramienta que contribuye a la competitividad en las PyMEs y al crecimiento económico de las regiones.

Abstract

The increasing demand for development of applications running on mobile devices (APP) in the market has stimulated a boom in software development, people are using this type of devices more commonly, therefore improvement and innovation has become a top priority nowadays.

Literature does not contain explicitly full development environment accordingly to programmers demanding the best practices and methods. In Mexico, federal government is recognized to push clusters of technologies in several country regions and making support programs to stimulate software industry. Quality request for mobile applications drive companies into a competition for the market, having as reference different standards and the Mexican Official Norm (NOM).

This thesis proposes a design methodology MDSIC-M which is oriented to rapid application development matching the needs of small and medium enterprises (SMEs). Improves those projects that needed to be deliver fast and having a beta prototypes allows an organized development, contemplating quality standards by having processes and quality evaluations.

There have been carried out experimental tests with regional enterprises to observe the methodology utility and gather information whether or not is a good alternative for software factories in Mexico. Finally as an advantage the methodology proposes store project histories, documenting size, time, costs and risks that took place during the project, allowing the Company to consult this database and do not repeat the mistakes, improving its development process. Components reuse is the best alternative to be competitive in mobile development market. The methodology enables quality mobile applications development, contributing to SMEs competitiveness.

ÍNDICE GENERAL

Resumen	v
Abstract.....	vi
Índice de figuras.....	xi
Índice de Tablas.	xv
Introducción	1
Capítulo 1. Propósito y organización de la tesis.	3
1.1 Descripción del problema.	3
1.2 Propósito de la investigación.	4
1.3 Antecedentes.....	5
1.4 Justificación de la investigación.	7
1.5 Hipótesis.	8
1.6 Objetivos de la Investigación.....	8
Objetivo General.....	8
Objetivos específicos.	8
1.7 Preguntas de la Investigación.	9
1.8 Alcances y limitaciones.....	10
1.9 Organización de la tesis.....	10
Capítulo 2. Caracterización del desarrollo de aplicaciones móviles (APPs).	13
2.1 Retos en el desarrollo de aplicaciones móviles.	13
2.1 El mercado de las APP en Mundo.	13
2.2 El mercado de las APP en México.	15
2.3 Estado del arte en el desarrollo de APPs.....	20
2.3.1 Metodología DIMP.....	24
2.3.2 Metodología MASAM.	26
2.3.3 Metodología Mobile-D.....	29
2.3.4. Metodología Scrum para el desarrollo móvil.	32
2.4. Modelos de desarrollo de software ágiles para el SPM.....	35
2.4.1 CMMI/CMM.	36
2.4.2 Kanban.	39

2.4.3 Moprosoft.....	41
2.4.4 ISO/IEC.....	46
2.4.5 MDSIC.	51
2.2. Síntesis de las ventajas y desventajas de metodología analizadas	56
2.5 Metodología ágil y adelgazada como medio de competitividad.	58
2.5.1 Filosofía Lean en el SPM.....	58
2.5.2 Desarrollo ágil (Agile).....	62
2.5.3 Aplicación del Lean y Agile en el desarrollo.	66
Capítulo 3. Clúster de Tecnologías de Información (CTI) y Fábricas de software (FS).....	69
3.1 Fábricas de Software.	69
3.2 Clúster de TI.....	70
3.2.1 La formación de clústeres de software en México.	71
3.2.2 Clústeres de TI en México.....	76
3.2.3. Organización de un Clúster TI para la competitividad.....	80
3.3 La TICs como competitividad en PyMEs.	86
3.4 Experiencias y desafíos en la apropiación de las TICs en PyMEs.	91
3.5 Competitividad de las PyMEs.	92
3.5.1 Alineación de las TI y ventaja competitiva.....	94
3.6 Tecnologías de la información en la inteligencia organizacional.	95
3.7 Innovación tecnológica en la industria TI.	95
3.8 Fábricas de software.....	100
3.8.1 Buenas prácticas aplicables a las FS MiPyMEs.	103
3.9 Administración de proyectos en fábricas de software.....	105
3.10 Modelos y metodologías para el desarrollo de software.....	108
3.10.1 Desarrollo de software dirigido por modelos (MDD) versus Fábricas de Software (FS).....	109
3.11 Tendencias en fábricas de software sobre el desarrollo móvil.....	111
Capítulo 4. Proceso metodológico para el diseño de la metodología MDSIC-M.....	113
4.1 Identificación de variables independientes.....	113
4.2 Análisis de la población.	116
4.3 Diseño de la encuesta.....	119
4.4 Resultados de la encuesta.....	125
4.5 Análisis de correlaciones.....	127

4.5.1 Tipos de análisis aplicados a las encuestas.....	127
4.6 Interpretación de las respuestas de la encuesta.....	131
4.7 Indicadores de Calidad.....	141
4.8 Estándares y normas de calidad.....	144
4.9 Alcance de la encuesta.....	146
Capítulo 5. Metodología MDSIC-M.....	147
5.1 Calidad en el desarrollo de una APP.....	147
5.2 Mejores prácticas PMI Agile & Lean.....	148
5.3 Elementos de MDSIC-M.....	150
5.2.1 Modelos de calidad en metodologías ágiles y en MDSIC-M.....	152
5.3 Modelado del planificador (<i>scheduler</i>) de MDSIC-M.....	154
5.3.1 Modelado del scheduler con Redes de Petri Coloreadas (CPN).....	156
5.4 Etapas y procesos de MDSIC-M.....	161
5.4.1 Fase de planeación.....	164
5.4.2 Fase de prototipos.....	164
5.4.3 Fase de desarrollo.....	165
5.4.4 Fase de refactorización.....	166
5.4.5 Fase de Aseguramiento de la calidad.....	167
5.4.6 Fase de cierre.....	168
5.5 Procesos para verificación de la calidad mediante los KPI.....	169
5.5.1 Establecimiento de Métricas y KPIs.....	170
5.5.2 KPIs en MDSIC-M.....	173
5.6 Medición de la calidad.....	175
5.6.1 Otros modelos para medir la calidad.....	178
5.6.2 El modelo de calidad de desarrollo para metodologías ágiles.....	179
5.6.3 Control de defectos para garantizar la calidad.....	181
5.7 Estándares y la NOM para la calidad en el desarrollo.....	183
5.8 Modelo de Madurez para MDSIC-M.....	187
5.9 Calidad en los equipos y roles.....	191
Capítulo 6. Pruebas de la metodología MDSIC-M.....	192
6.1 Requerimientos para la herramienta MDSIC 2.0.....	192
6.1.2 Requerimientos funcionales y no funcionales para el diseño de MDSIC 2.0.....	196

6.2 Establecimiento del Plan para el desarrollo de MDSIC 2.0.....	197
6.3 Diseño del plan de pruebas.....	199
6.4 Diseño del prototipo.....	201
6.4.1 Casos de uso.....	201
6.4.2 Diseño de la base de datos (BD).	204
6.5 Desarrollo de la herramienta MDSIC 2.0.....	206
6.5.1 Herramientas y Diseño Orientado a Objetos (DOO).....	207
6.5.2 Construcción del scheduler.	209
6.5.3 Componentes de desarrollo.....	210
6.6 Pruebas y Cierre del proyecto.....	213
6.6.1 Indicadores de calidad.....	213
Capítulo 7. Análisis e interpretación de los resultados y trabajos futuros.	215
7.1 Discusión.....	215
7.2 Validación de un cuasi experimento.....	216
7.3 Comparativa con otras metodologías.....	219
7.4 Comparativas de MDSIC 2.0 con otras herramientas.	224
7.5 Experiencias de desarrollo con las PyMEs.....	227
7.6 Trabajos futuros.....	229
7.7 Aportación de MDSIC-M como metodología ágil para FS.....	232
Bibliografía	243
ANEXO A. Diccionario de datos de la Herramienta MDSIC 2.0.....	253
ANEXO B. Descripción de procesos y Subprocesos de MDSIC-M.	259
Fase de planeación.	261
Fase de prototipos.	261
Fase de desarrollo.....	262
Fase de refactorización.....	262
Fase de Aseguramiento de la calidad.....	263
Fase de cierre.....	264
ANEXO C. Manual de usuario y técnico de MDSIC2.0.....	265
MANUAL DE USUARIO.....	265
MANUAL TÉCNICO.....	271
ANEXO D. Código ejemplo del scheduler.	274

Índice de figuras.

Figura 2.1 A), B), D), E). Resultados de la encuesta a desarrolladores de APP a nivel mundial.	14
Figura 2.2 A), B). Metodologías que conocen y que siguen las empresas encuestadas.	15
Figura 2.3 A) y B). Análisis de Esfuerzo-Beneficio para las mejores prácticas de difusión y venta.	17
Figura 2.4. Cadena de valor en la industria del desarrollo móvil.	18
Figura 2.5. Análisis de las fuerzas de Porter para esta industria.	18
Figura 2.6. Perfil de la demanda por sector.	19
Figura 2.7. El mercado de dispositivos móviles en México, población de 118.4 millones de habitantes.	20
Figura 2.8. Diagrama que representa a la arquitectura de alto nivel de <i>DIMP</i>	25
Figura 2.9. Diagrama que representa la estructura de <i>DIMP</i>	26
Figura 2.10. Etapas de la metodología MASAM híbrida con XP.	29
Figura 2.11. Fases y estados de la metodología Mobile-D.	30
Figura 2.12. Metodología de desarrollo basado en Scrum.	34
Figura 2.13. Niveles de madurez en el desarrollo de software CMM/CMMI.	39
Figura 2.14. Pasos en la metodología ágil Kanban para el desarrollo de software.	40
Figura 2.15. Niveles de madurez propuestos en Moprosoft.	44
Figura 2.16. Modelo de evaluación del nivel de madurez sugerido por COMPETISOFT.	45
Figura 2.17. Estándares ISO/IEC que promueven la calidad en el desarrollo de software.	48
Figura 2.18. Metodología de diseño del MDSIC.	51
Figura 2.19. Modelo de desarrollo de software integral colaborativo.	55
Figura 2.20. Número de publicaciones del desarrollo ágil desde 2001 al 2011.	64
Figura 2.21. Principales revistas científicas que trataron el tema de desarrollo ágil.	65
Figura 3.1. Regiones definidas por la CANIETI.	77
Figura 3.2. Diamante de Porter para los clúster de TI.	81
3.3. Proceso evolutivo de la gestación y maduración de un Clúster.	82
Figura 3.4 Características de los Clúster de TI.	83
Figura 3.5. Matriz comparativa de los principales CTI en México de acuerdo a su madurez.	85

Figura 3.6. Ventajas competitivas del sector TIC en México.	88
Figura 3.7. Factores que estimulan o cohiben la decisión de una persona para iniciar un negocio.	89
Figura 3.8. Creación de PyMEs en el sector de TI para el año de 2012.	90
Figura 3.9. Principales programas para el emprendimiento e innovación en México.	90
Figura 3.10. Modelo de competitividad de las PyMEs.	93
Figura 3.11. Análisis de la capacidad de absorción de una PyME.	94
Figura 3.12. Crecimiento del PIB con respecto de la TI y las TIC, miles de millones de pesos Mexicanos.	96
Figura 3.13. Tasa de crecimiento por sector en el empleo.	97
Figura 3.14. Crecimiento de profesionistas por sector, proyección 2025.	98
Figura 3.15. Necesidades de capacitación para fomentar la innovación en TI.	99
Figura 3.16. Procesos del área de TI que requieren certificaciones para promover la calidad.	100
Figura 3.17 A) y B). Evolución de FS basadas en modelos a FS basada en buenas prácticas.	102
Figura 3.18. Modelo estandarizado de servicio en una FS.	105
Figura 3.19. Principales sectores dentro de la Tics que más se desarrollan en PyMEs.	111
Figura 4.1. Proceso metodológico para el diseño de MDSIC-M.	114
Figura 4.2. Cálculo de la muestra n con la mayor precisión, un intervalo de confianza y un error.	118
Figura 4.3. Encuesta realizada para censar la industria del desarrollo móvil en la región	120
Figura 4.4. Formula para calcular alfa de Cronbach a partir de las varianzas.	125
Figura 4.5. Cálculo del α de las respuestas de $k = 38$ encuestas de empresas que desarrollan APPs.	126
Figura 4.6. Resultado del análisis de correlaciones, bivariada de Pearson (r) con significancia ≥ 0.50	130
Figura 4.7 A), B), C), D), E) y F). Principales correlaciones con $r \geq 0.5$, preguntas con escala de Likert.	134
Figura 4.8. Resultado de las preguntas que tuvieron mejor aprobación entre las 38 de 70 empresas	135
Figura 4.9. PyMEs que contestaron la encuesta por estado 46 de 70 contestaron.	137
Figura 4.10. Tamaño de las empresas, 42 de 70 contestaron (60%).	137
Figura 4.11. Puestos de trabajo de los entrevistados 43 de 70 contestaron (61%).	137
Figura 4.12. Tipos de servicios para los que desarrollan APPS, 38 de 70 contestaron (54%).	138
Figura 4.13. Métodos que conoce para garantizar la calidad, 25 de 70 contestaron.	138

Figura 4.14. Metodologías de desarrollo para APPs que conoce, 38 de 70 contestaron.	139
Figura 4.15. Estrategia de desarrollo que siguen las empresas para desarrollar una APP	139
Figura 4.16. Métodos de seguridad que aplican en las APPs, 16 de 70 contestaron.	140
Figura 4.17. A) y B). Pertenece a un clúster de TI, solo 11 de los encuestados si pertenece a un clúster.	140
Figura 4.18. Tiempo que llevan desarrollando APPs, 30 de 70 contestaron.	141
Figura 4.19. Tipo de aplicaciones que más desarrollan, 38 de 70 empresas contestó.....	142
Figura 4.20. Tendencia hacia los métodos, metodologías o formas propias para garantizar la calidad	146
Figura 5.1. Áreas del conocimiento y de aplicación del PMI-ACP.....	148
Figura 5.2. Elementos de MDSIC-M para garantizar la calidad en el desarrollo de una APP.	150
Figura 5.3. Elementos de la metodología que se ejecutan como procesos para garantizar la calidad	154
Figura 5.4. Elementos del WBS para establecer las tareas y entregables de un proyecto.	155
Figura 5.5. Elementos de una CPN.	158
Figura 5.6. CPN que define los elementos del scheduler diseñado para MDSIC-M.	160
Figura 5.8. Frecuencia de uso de una APP y su categoría.	173
Figura 5.9. Estructura de la evaluación de un proceso.....	177
Figura 5.10. Estructura del grado de certificación de la calidad de un proceso.	177
Figura 5.11. Nivel de madurez de un proceso basado en KPAs.....	178
Figura 5.12. Ontología que describe el proceso de detección de defectos de calidad.	182
Figura 5.13. Ejemplo de los atributos de una proceso para el nivel de capacidad 1 y 2.	185
Figura 5.14 Especificación del proceso y de su capacidad NMX.	185
Figura 5.15. Evaluación de la capacidad de un proceso y sus atributos.	186
Figura 5.16 Establecimiento del nivel de madurez del proceso Planear y sus KPIs.	188
Figura 5.17 Establecimiento del nivel de madurez del proceso Prototipo y sus KPIs.....	188
Figura 5.18 Establecimiento del nivel de madurez del proceso Desarrollo y sus KPIs.	189
Figura 5.19 Establecimiento del nivel de madurez del proceso Refactorizar y sus KPIs.	189
Figura 5.20 Establecimiento del nivel de madurez del proceso Asegurar la calidad y sus KPIs.	190
Figura 5.21 Establecimiento del nivel de madurez del proceso Cerrar proyecto y sus KPIs.	190

Figura 5.22 Establecimiento de equipos y roles de trabajo propuesto por TSP.....	191
Figura 6.1 Técnicas, herramientas y conocimientos, habilidades para la administración de proyectos.....	194
Figura 6.2 A), B) y C). Diseño de la herramienta MDSIC 2.0 que utiliza la metodología MDSIC-M.	196
Figura 6.3 WBS construido para el desarrollo de la herramienta MDSIC 2.0.	198
Figura 6.4 Casos de uso para el desarrollo con POO de MDSIC 2.0.	203
Figura 6.5. Diagrama entidad-relación de la BD de MDSIC 2.0.....	205
Figura 6.5. DOO de los elementos y componentes de la herramienta MDSIC 2.0.	209
Figura 6.6. Atributos y métodos de las clases scheduler, Requerimiento, Tareas y requerimientos.....	210
Figura 6.7. Interfaces del prototipo de MDSIC 2.0 para su desarrollo.....	211
Figura 6.8 A), B) y C) Interfaces que se diseñaron para validar la calidad	214
Figura 7.1. Contexto de desarrollo de la metodología MDSIC-M.....	215
Figura 7.2. Evaluación de la madurez de procesos MDSIC-M en dos proyectos reales.....	229
Figura 7.3. Comparativa del uso de la metodología en entornos ágiles.....	233
Figura 7.4 A), B) y C). Capacidad de agilidad de las metodologías analizadas.	234
Figura 7.5. Aplicabilidad de las metodologías analizadas.	234
Figura 7.6. Comparativa con respecto a procesos y productos, nivel de abstracción de normas	235
Figura 7.7. Comparativa con respecto a procesos y productos, las actividades cubiertas por el método.	236
Figura 7.8. Comparativa con respecto a procesos y productos, productos de las actividades del método	236
Figura 7.9. Totales de las características censadas de las metodologías ágiles.....	237
Figura 7.10. Fases o etapas que considera la metodología para desarrollo de APPs.	238
Figura 7.11. Número de miembros en los equipos de trabajo y roles de las metodologías.	238
Figura 7.12. Número de KPIs que censan las metodologías en la evaluación de calidad.	239
Figura 7.13 A), B), C), D) y E). Madurez de los procesos de MDSIC-M de acuerdo a la calidad censada	242

Índice de Tablas.

Tabla 2.1. Descripción de las fases y tareas en la metodología Mobile-D.	30
Tabla 2.2 Estructura de la NMX-I-059-NYCE-2005.....	43
Tabla 2.3. Elementos de la norma ISO/IEC 15504 SPICE.	47
Tabla 2.4 Grupos de trabajo y áreas de la norma ISO/IEC JTC 1/SC 7.	50
Tabla 2.5. Funcionamiento de los niveles del MDSIC.	52
Tabla 2.6 Ventajas y desventajas de las metodologías para desarrollo móvil actuales.	56
Tabla 2.7 Principios de la filosofía adelgazada (<i>Lean</i>).	58
Tabla 2.8 Problemas al aplicar la filosofía Lean en el SPM.	60
Tabla 2.9. Principales aplicaciones del Lean al desarrollo de Software en las empresas.	66
Tabla 3.1. Políticas nacionales que han apoyado al software.....	71
Tabla 3.2. Asociaciones y organismos que promueven las TI en México.	72
Tabla 3.3. Clústeres que fomentan la industria de TI en México.	78
Tabla 3.4. Buenas prácticas utilizadas en FS.....	103
Tabla 3.5. Ventajas y desventajas del MDA vs FS.....	109
Tabla 4.1. Variables independientes que se necesitan censar para MDSIC-M.....	115
Tabla 4.2. Población económicamente activa en la región centro-occidente de México.....	117
Tabla 4.3. Análisis de la población para aplicar la encuesta.....	117
Tabla 4.4. Cálculo tamaño de la muestra con un intervalo de confianza del 95% y 97% y un error del 5%....	119
Tabla 4.5. Preguntas realizadas en la encuesta, 37 preguntas; 15 preguntas abiertas y 22 en Likert.....	122
Tabla 5.1 Elementos que definen la metodología MDSIC-M.	150
Tabla 5.2. Propiedades de una CPN.	158
Tabla 5.3. Procesos y subprocesos de la metodología MDSCI-M.	169
Tabla 5.4. KPIs para establecer la calidad mediante factores de éxito.	173
Tabla 5.5 Modelos para validar la calidad en el desarrollo de software.	178
Tabla 5.6 Técnicas para garantizar la calidad en métodos ágiles.	180
Tabla 5.7 Defectos de calidad por nivel.	181

Tabla 5.8. KPAs y sus correspondencias con el modelo de calidad de Chappel.....	183
Tabla 5.9. Mapeo de los procesos y KPIs de MDSIC-M hacia el estándar NMX-I-15504	186
Tabla 6.1 Diferentes tipos de requerimientos para el software MDSIC 2.0.....	196
Tabla 6.2. Formato de pruebas para el desarrollo de MDSIC 2.0.	199
Tabla 6.3 Pruebas que se aplicarán a las interfaces de la herramienta MDSIC 2.0.	200
Tabla 6.4 Herramientas y Framework utilizados para el desarrollo de MDSIC 2.0.	206
Tabla 6.5. Componentes desarrollados para creación y modificación de proyectos.....	211
Tabla 6.6. Componentes agrupados para creación y modificación de tareas y fases	212
Tabla 7.1. KPAs y su correspondencia con el modelo de calidad de Chappel (2014).	217
Tabla 7.2. Validación de la calidad en 2 proyectos en PyMEs de la región.	217
Tabla 7.3. Comparativa entre metodologías para el desarrollo de APPs.	220
Tabla 7.4. A), B), C) y D). Comparativa de MDSIC-M con otras metodologías ágiles.	221
Tabla 7.5. Comparativa de herramientas para la administración de proyectos ágiles.....	226

Introducción

La demanda del mercado para el desarrollo de aplicaciones que se ejecutarán en sistemas operativos móviles (reducidos en capacidades), ha generado un gran auge en el desarrollo de aplicaciones móviles (APPs). Recientemente se observa que las personas pasan más tiempo en los dispositivos móviles que en su computadora de escritorio. Por ello, existe la necesidad de mejorar e innovar en el desarrollo de una APP, las cuales deben ser cada vez más rápidas en desempeño y fáciles de construir.

En el estudio de (Piattini & Garzas, 2010) mencionan que este mercado seguirá creciendo y es una prioridad. Existen en la actualidad “marcos de desarrollo que no cumplen necesariamente con esta demanda”.

En México se reconoce una gran labor del gobierno federal por impulsar clústeres locales de Tecnologías de la Información (TI) en diferentes estados del país, así como programas de apoyo para incentivar la industria del software (PROSOFT 3.0) (Secretaría de Economía, 2014). Sobresalen los clústeres de TI de la región Centro-Sur y del Norte de México quienes son los que impulsan a esta industria a nivel nacional. Uno de estos es el clúster de alta tecnología de la región Centro-Occidente de México que está en pleno desarrollo y consolidación - que aglutina los estados de Aguascalientes, Colima, Guanajuato, Jalisco, Michoacán y Nayarit - (definido por la Cámara Nacional de la Industria Electrónica, de Telecomunicaciones y Tecnologías de la Información, CANIETI). El reto es construir fábricas de software que compitan a nivel nacional e internacional, (Escutia Núñez, 2010).

El desarrollo de aplicaciones móviles de calidad es una necesidad en este mercado por lo que los estándares y la NOM son el referente a seguir para los desarrollos hechos a la medida. En (Cendejas V., Vega L., Careta I., Gutierrez S., & Ferreira M., 2014) se propone el Modelo de Desarrollo de Software Integral Colaborativo (MDISC) el cual está orientado al desarrollo rápido

de software y que es una necesidad de las Pequeñas y Medianas Empresas (PyMEs) de la región Centro-Occidente de México. Este modelo propone una forma fácil y sencilla del desarrollo de software sin dejar de lado los estándares de calidad. Sin embargo este modelo no está diseñado para el desarrollo de una APP (no fue considerado como una prioridad), por lo que es necesario hacer las mejoras para que una metodología basada en MDSIC permita el desarrollo móvil con calidad y que pueda ser una realidad para empresas PyMEs en México. El objetivo es que esta se convierta en un referente de desarrollo y permita la competitividad para las fábricas de software (FS) de la región.

Esta tesis presenta el diseño de una metodología fundamentada en la experiencia de la industria del desarrollo móvil, que permitirá la construcción de APPs apegadas a normas y a las necesidades de las empresas que estén inmersas en este mercado. Además se presentan los resultados y experiencias de aplicarla en FS de la región, observando el comportamiento y necesidades al utilizar MDSIC-M y su herramienta MDSIC 2.0 (software guía para aplicar la metodología en el desarrollo de una APP de calidad).

Capítulo 1. Propósito y organización de la tesis.

1.1 Descripción del problema.

El desarrollo de aplicaciones móviles requiere de nuevas técnicas y nuevos retos. Recientemente las metodologías ágiles han inundado el mercado del desarrollo de aplicaciones, metodologías como XP (*eXtream Programming*) (XP, 2015), Scrum (marco de trabajo para la gestión y desarrollo de software y entornos basados en el desarrollo ágil de software) (Scrum, 2015), entre otras son conocidas como metodologías AUP (*Agile Process Unified*) (Jeong, Lee, & Shin, 2008). Se reconoce que las metodologías más acordes para el software en dispositivos móviles se centran en el desarrollo ágil y la generación rápida de código (Agile, 2012). Por lo general las aplicaciones móviles son pequeñas (satisfacen un mercado especializado) y son desarrolladas por empresas pequeñas de software, el desarrollo sistemático es quizá el problema más difícil ya que se deben de entregar resultados rápidamente Jeong et. al. (2008), existen muchas metodologías que pueden servir para este propósito, el problema es que las empresas gastan muchos recursos para adoptarlas y dar un seguimiento de calidad - se termina trabajando para la metodología -. Esto afecta en gran medida a las fábricas de software que son PyMEs, Cendejas et. al. (2014).

Los métodos que utilizan las empresas generadoras de software en la actualidad deben obtener aplicaciones en forma rápida y de acuerdo a las necesidades de los usuarios; quienes utilizan cada vez más los dispositivos móviles en todos los ámbitos del quehacer humano. Los métodos tradicionales de desarrollo aunado a las metodologías ágiles no ayudan a generar software de calidad. En Piattini et. al. (2010), se reconoce una falta de estándares y normas que permitan validar su calidad y fiabilidad. Los investigadores hicieron análisis y revisiones sobre las empresas que desarrollan software en países como España, Argentina, México y Venezuela. Se obtuvieron datos interesantes sobre la industria del software y su impacto regional. El desarrollo de aplicaciones en fábricas de Software lleva varios años y han enfrentado varios retos, desde la

crisis del software hasta los estándares de calidad que en la actualidad se están requiriendo. En la última década se han publicado diversos estudios y análisis que describen como se deben seguir diferentes etapas para garantizar la calidad en los productos de software, estándares como: ANSI, AIAA R-013-1992, EIA/IS-632, IEC 61508, IEEE 1228-1994, ISO/IEC 2382-7:1989, 9126, 25000, entre otros (Bednar & Robertson, 2005). Y modelos como: CMM-CMMI, Moprosoft, PMI Agile & Lean, ISO 9000, UML, entre otros (Kan, 2003). Son propuestas de un desarrollo dirigido por modelos que permite la generación de software de calidad, beneficiando al consumidor con productos estables y funcionales como:

- líneas de software,
- los servicios Web,
- el desarrollo software global,
- nuevas tecnologías de desarrollo y
- el surgimiento de nuevo software para dispositivos.

Todo estas necesidades hace que la industria del software tenga nuevos retos para lograr satisfacer este nuevo mercado en donde la gestión de requisitos, las pruebas, la configuración, aspectos organizativos y de gestión de la organización, son algunos aspectos que las FS deben considerar para ser un éxito.

1.2 Propósito de la investigación.

El propósito principal que se busca en este proyecto de investigación es diseñar e implementar una metodología basada en el MDSIC, que permita el desarrollo ágil en aplicaciones móviles y apegadas a estándares de calidad (NOM). Se propone la construcción de la metodología orientada a FS que son PyMEs, quiénes al utilizar MDSIC-M puedan competir en el mercado global de las APPs.

La utilización de un marco de referencia y una metodología basada en estándares podrá generar un crecimiento económico de la industria de TI de las regiones, fomentando así la

innovación, la competitividad de las PyMEs de la región y en consecuencia un crecimiento económico.

1.3 Antecedentes.

El desarrollo de software para plataformas móviles viene con características únicas y las metodologías que se utilizan corresponden a las etapas del ciclo de vida del desarrollo de software (desarrollo ágil). Los entornos y las tecnologías que apoyan al software en dispositivos móviles tiene diferentes parámetros de construcción en comparación a los valores "tradicionales" del desarrollo de software. Un análisis de las restricciones asociadas a las APPS se presenta en (Abowd, et al., 2005). Los autores menciona dos tipos de restricciones, en constante evolución e inherentes:

- 1) Restricciones en evolución; tales como ancho de banda, cobertura y seguridad.
- 2) Las limitaciones inherentes; tales como el espacio en pantalla limitada, la capacidad de entrada de datos reducida (debido a un teclado limitado, por ejemplo), la capacidad de memoria, potencia de procesamiento, el arranque y la ejecución limitada -son permanentes, por lo menos en relación con los entornos de escritorio -.

Los desarrolladores han superado estas limitantes mediante el enfoque del desarrollo ágil de acuerdo a los estudios realizados por (Yang-Jae, Ji-Hyeon, & Gyu-Sang, 2008) y (Xiaofeng, Kieran, & Oisin, 2012). El uso de los métodos ágiles ha recibido tanto apoyo y en ocasiones oposición; el principal argumento en contra de los métodos ágiles es la presunta falta de la validación científica en sus actividades y prácticas asociadas, así como la dificultad de integrarlas en un plan -falta de planeación-. De hecho, algunos proyectos presentan una mezcla de características tradicionales con métodos ágiles, en cuyo caso debe lograrse un equilibrio en el uso de ambos tipos de métodos (Nerur, Mahapatra, & Mangalaraj, 2005).

Hay también incertidumbre en la distinción de los métodos ágiles de programación ad-hoc a las necesidades del desarrollo. Sin embargo, como se indica en (Salo & Abrahamsson, 2006), los

métodos ágiles proporcionan un enfoque de desarrollo organizado que ayuda a un desarrollo con ciertos estándares de calidad.

En el artículo del grupo de investigación (Agile, 2012), analizan las publicaciones y los avances en desarrollo ágil que se han realizado en los últimos 10 años, de este sobresalen las metodologías como: XP (Wells, 2013), Scrum (Scharff & Verma, 2010) (Abrahamsson, et al., 2004), Lean Six Sigma (goleansixsigma, 2014), Crystal (Cockburn, 2004), entre otras.

El rápido crecimiento del desarrollo móvil hacia nuevas aplicaciones ha creado una nueva industria en el software y es quizá el reto más grande para los desarrolladores tradicionales. En el artículo de (Salvaneschi, Ghezzi, & Pradella, 2012) mencionan que la demanda de aplicaciones en diversos entornos y el uso del cómputo ubicuo (con movilidad) se está acelerando, uno de los grandes retos en esta industria es crear aplicaciones basadas en el contexto (interacción entre humanos, aplicaciones y su entorno), estas aplicaciones deberán facilitar la interacción entre humanos y los dispositivos ubicuos. Desafortunadamente el desarrollo basado en el contexto todavía está por desarrollarse ya que los cambios y las mejoras en el cómputo ubicuo deben adaptarse a las nuevas tecnologías que surgen día a día (Fortier, Rossi, Gordillo, & Challiol, 2009).

Si bien la perspectiva propuesta por Salvaneschi et al. (2012) es útil para definir el contexto de una aplicación basada en la ubicuidad, en la actualidad existen nuevas tendencias que permiten el desarrollo de aplicaciones móviles en ambientes distribuidos (en el contexto de una red). En (De Silva, Darshana, Silva, Basnayake, & Wickramaratne, 2013) explican que las futuras tecnologías relacionadas con dispositivos inteligentes (móviles) harán la vida más fácil, el ahorro en el tiempo de una persona que utiliza esta tecnología en forma ubicua es el impacto mayor.

Los dispositivos móviles ya han añadido valor a nuestras tareas cotidianas. Sin embargo, los programadores de este tipo de desarrollo, hasta el momento, parece que se les ha complicado el uso de estas tecnologías y sobre todo el desarrollo rápido con este tipo de necesidades hacen que el desarrollo tradicional basado en modelos cambie.

De las metodologías o modelos más documentados y utilizados en la literaturas sobresalen; *Development Environment for Mobile Platforms* (DIMP), *Mobile Application Software Based Agile Methodology* (MASAM), *methodology for agile software development* (Mobile-D), *agile approach integrates Scrum and Lean Six Sigma* (SLeSS) (Harlem, Swati, & Xiaofeng, 2014) y MDSIC (Cendejas V., Vega L., Careta I., Gutierrez S., & Ferreira M., 2014), este último en la región centro occidente de México.

Se reconoce que los desarrollos de aplicaciones móviles son alentados por fábricas de software (SF, por sus sigla en inglés) a nivel mundial (Piatini V. & Garzas P., 2010). Una FS implica una forma de organizar el trabajo con una considerable especialización, así como una formalización y estandarización de procesos. También se define como una instalación (facility) que ensambla (no necesariamente codifica) aplicaciones de software conforme a una especificación siguiendo una metodología estricta.

1.4 Justificación de la investigación.

El desarrollo de aplicaciones móviles requiere de nuevas técnicas y retos que no se tenían contemplados en el desarrollo estándar de software (hecho a la medida), para este tipo de dispositivos y cuya finalidad es la solución de problemas de ubicuidad de los usuarios.

Se reconoce en la literatura que las metodologías más acordes para estas nuevas formas de construir software deben centrarse en el de desarrollo ágil y la generación rápida de código, es decir en métodos que ayuden a las FS-PyMEs (en México) ha obtener aplicaciones en tiempos más cortos, con calidad y de acuerdo a las necesidades de los usuarios - los cuales utilizan cada vez más los dispositivos móviles en todos los ámbitos del quehacer humano -. La movilidad y las restricciones de estos dispositivos es un nuevo reto en la ingeniería de software.

Se ha observado que los métodos tradicionales no ayudan a generar software de calidad con estos requerimientos y que existen nuevos retos para obtener soluciones en un tiempo corto una APP con calidad.

1.5 Hipótesis.

Las FS-PyMEs que utilicen la metodología MDSIC-M lograrán generar software de calidad y competir en la industria, al utilizar una herramienta que apegada a las NOM les ayude a organizar sus procesos de desarrollo, mejorando con esto la satisfacción de los usuarios al utilizar una APP.

1.6 Objetivos de la Investigación.

Objetivo General.

Diseñar una metodología para el desarrollo de software en dispositivos móviles que contribuya a la calidad de los productos generados por las fábricas de software, fomentando el desarrollo de aplicaciones en la región Centro Occidente del país, para fortalecer la competitividad y el desarrollo económico. Además de incrementar la satisfacción del usuario de una APP.

Objetivos específicos.

- Realizar una caracterización del desarrollo de aplicaciones móviles para conocer sus componentes y construir una metodología que ayude al desarrollo de una APP de calidad.
- Investigar las fábricas de software que desarrollan aplicaciones móviles en los clústeres de la región centro-occidente de México, para conocer las buenas prácticas utilizadas en el desarrollo.
- Investigar los estándares de calidad utilizados en la industria del desarrollo del software en la región centro-occidente de México, para aplicarlos en la metodología.

- Desarrollar las fases complementarias al modelo MDSIC para el desarrollo Móvil, para un desarrollo con calidad apegado a las necesidades de las PyMEs de la región.
- Desarrollar un esquema para validar la calidad en el desarrollo móvil en la región centro-occidente de México, para promover los estándares y ayudar a la competitividad en la industria.
- Analizar los resultados y conclusiones de aplicar la metodología en el desarrollo de una APP, para mejorar los procesos y la madures en su construcción. Aumentando con esto la satisfacción del usuario.

1.7 Preguntas de la Investigación.

¿Cómo el desarrollo rápido de aplicaciones para dispositivos móviles ayudará garantizar la calidad del Software? Se observa en las diferentes plataformas de descarga de aplicaciones móviles un auge por la entrega rápida de programas y aplicaciones, pero estos desarrollos por lo general no cumple el 100% de lo que se estableció como requerimientos, además se ha detectado la falta de calidad en el desarrollo de software, ya que al descargarse y ejecutarse genera fallos y no cumplen con los objetivos para los cual debería de servir la APP.

¿Cuál será la forma de construir una metodología que ayude al desarrollo de aplicaciones móviles de calidad y que permita un desarrollo rápido y apegado a los requerimientos del usuario?

En Cendejas V. et al. (2014), se menciona que el principal problema del desarrollo de software es construir productos con calidad que satisfagan las necesidades y objetivos de las organizaciones. Además que el software no se encuentra alineado con los objetivos y metas de la organización, es decir el software solo es desarrollado por expertos de TI que se dedican a realizar un análisis, diseño y desarrollo del mismo, pero nunca se hacen acompañar por expertos de los procesos organizacionales que beneficien el desarrollo del producto de una manera formal. MDSIC es un Modelo que sirve como base para la construcción de aplicaciones y que siguen las mejores

prácticas. El modelo define 5 diferentes niveles que brindan un apoyo para el desarrollo de software, se utilizan las mejores prácticas que propone el *Project Management Institute* (PMI, 2014) en la planeación de proyectos, lo que permite generar un software de calidad - alineado con la estrategia de la organización -. MDSIC-M es la metodología que ayudará al desarrollo de APPs de calidad y que basada en el Modelo permitirá promover las mejores prácticas en la industria.

1.8 Alcances y limitaciones.

- Se propone la construcción de la metodología basada en el MDSIC para un desarrollo colaborativo y ágil.
- Se investigará la experiencia de los desarrolladores de esta industria y se observará las mejores prácticas para proponer las adecuaciones necesarias en la metodología propuesta.
- Se analizará los estándares de calidad vigentes y la NOM para garantizar un desarrollo rápido basado en un estándar de calidad.
- Se revisarán las nuevas tecnologías de desarrollo ágiles y nuevas tendencias.
- Se hará una propuesta para validar la calidad y dar seguimiento durante el desarrollo de aplicaciones móviles.

1.9 Organización de la tesis.

La tesis se organiza en siete capítulos que describen el propósito y las preguntas de investigación, la caracterización del desarrollo móvil, la competitividad de los clústeres de TI y las fábricas de software, el diseño de la metodología, las características de la metodología, las pruebas de la propuesta y los trabajos futuros en la utilización y aprendizaje de MDSIC-M. Los capítulos se describen a continuación.

En el capítulo 1 se establece el problema a resolver, el propósito de la investigación, los antecedentes, la justificación, la hipótesis, los objetivos, las preguntas de investigación y los alcances y limitaciones de esta tesis.

En el capítulo 2 se realiza una caracterización del desarrollo móvil, cuales son los retos, el mercado mundial y nacional, se revisa el estado del arte del desarrollo de una APP, que metodologías son las más utilizadas, cuales son los modelos de desarrollo ágiles y las metodologías ágiles y adelgazadas que se utilizan en el desarrollo de software.

En el capítulo 3 se describe la importancia de conocer la competitividad de los clústeres de tecnologías y las fábricas de software, se investiga el papel del TICs como medio de competitividad, el impacto de las TI en las PyMEs, la innovación tecnológica en las TI, se estudia también la innovación en las fábricas de software y sus tendencias en el desarrollo móvil.

En el capítulo 4 se describe el proceso metodológico para desarrollar la metodología MDSIC-M, se identifican las variables a censar, se determina la población que se desea estudiar, se diseña y aplica una encuesta a esta población, se determinan que variables son las de mayor importancia. Se hace también un análisis de la consistencia de la encuesta y sus correlaciones, se revisa cuales son los indicadores de calidad que más impactan a la empresas encuestadas y finalmente se concluye cuales son los estándares y normas de calidad que más utilizan la empresas desarrolladoras de APPs.

En el capítulo 5 se diseña la metodología MDSIC-M para el desarrollo de APPs con calidad, se estudian los modelos de calidad y mejores prácticas, también se definen los elementos de la metodología y se diseñan sus principales fases y procesos, se establece la necesidad de construir un planificador de tareas que permita el desarrollo con calidad, se describen todas la etapas de la metodología, se establecen los principales procesos y se define la forma de medir la calidad de una APP de acuerdo a su proceso de desarrollo y apegada a estándares de la NOM y un modelo de madurez.

En el capítulo 6 se realizan pruebas sobre la metodología, se definen los requerimientos de la herramienta MDSIC 2.0 que ayudará a utilizar la metodología MDSIC-M, se elabora un plan del proyecto para construir la herramienta así como un plan de pruebas. Se diseña el prototipo

utilizando el diseño y la programación orientado a objetos. Se establecen los casos de uso y se elaboran las interfaces y componentes de la herramienta. Finalmente se prueban los indicadores de calidad y cómo es posible censarlos en un proyecto real.

En el capítulo 7 se concluye y se describen los trabajos futuros de la metodología MDSIC-M, se establece como una necesidad de la industria del desarrollo ágil en México. Se valida la calidad en dos desarrollos pilotos, se compara la herramienta con otras que existen el mercado. También se compara a la metodología con otras utilizando el tamaño de un proyecto real y finalmente se describe la pertinencia de la metodología y los trabajos futuros que pueden desarrollarse.

Los anexos que se incluyen en esta tesis son los siguientes: A. Diccionario de datos de la Herramienta MDSIC 2.0, B. Descripción de los procesos y subprocesos de MDSIC-M, C. Manual de usuario y técnico de MDSIC 2.0 y D. Código ejemplo del scheduler.

Capítulo 2. Caracterización del desarrollo de aplicaciones móviles (APPs).

2.1 Retos en el desarrollo de aplicaciones móviles.

El desarrollo de aplicaciones móviles trae nuevos retos a la industria del software ya que la movilidad, la interconexión y la demanda de aplicaciones simples es cada vez más solicitado por los usuarios, se puede observar que en muchas tiendas en línea *Global-Delivery* (entrega global de una APP compatible a través de tienda) la tendencia es hacia la compra-venta de software conocido como “*freemium*” (licencia gratuita sencilla que crea una necesidad en el usuario y que a cambio de una licencia pagada se puede acceder a todas las características de software) y al mercado de la actualizaciones o mejoras de las aplicaciones. En todo el mundo y en México este mercado es estudiado por su relevancia y posibilidades de obtener ganancias para empresas de esta industria.

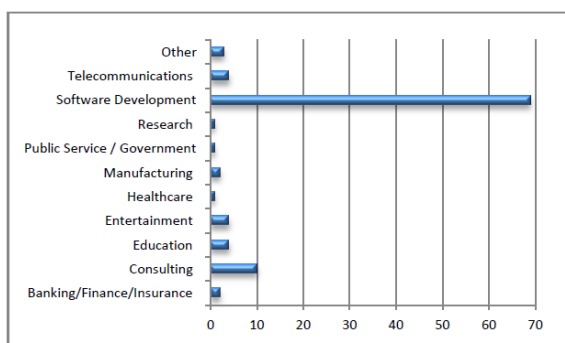
2.1 El mercado de las APP en Mundo.

En (Harlem, Swati, & Xiaofeng, 2014) se realizó un encuesta para estudiar qué métodos, las tendencias en el desarrollo y las metodologías que se utilizan en el desarrollo de una APP, además de obtener una mejor comprensión de las diferentes formas utilizadas por las empresas de telefonía para la construcción de una APP. Esta encuesta se diseño utilizando la literatura disponible para formular las preguntas. También participaron los miembros del equipo de desarrollo, expertos en el uso de metodologías ágiles, investigadores y otros interesados. Se obtuvieron los siguientes resultados:

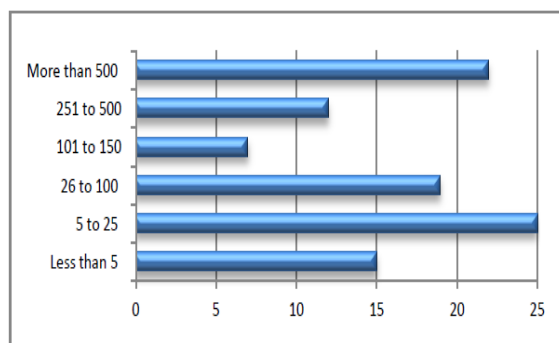
1. El 69% de los encuestados procedían de la industria móvil y tenía experiencia real en el desarrollo de aplicaciones móviles.

2. La mayoría de los encuestados tenían experiencia desarrollando una APP para la educación (14%), los servicios públicos (14%) y las comunicaciones (13%).
3. Los participantes eran del equipo de desarrollo (47%), consultores/formadores (33%) y equipo de gestión de nivel superior (20%).
4. Los encuestados provenían de distintos lugares del mundo y el (38%) eran procedentes de América del Norte.

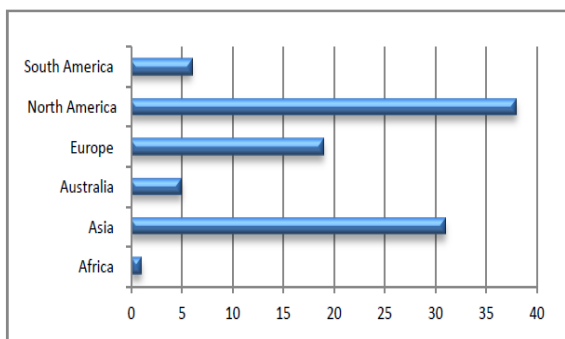
En la figura 2.1 A), B), C) y D), muestra los resultados más significativos de esta encuesta Harlem et. al (2014).



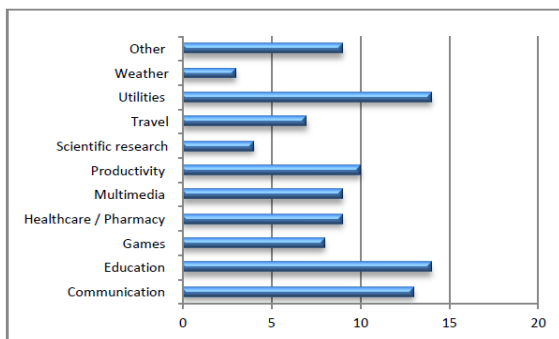
A) Empresas por tipo de organización.



B) Por tamaño y número de empleados.



D) Localización de la empresa.

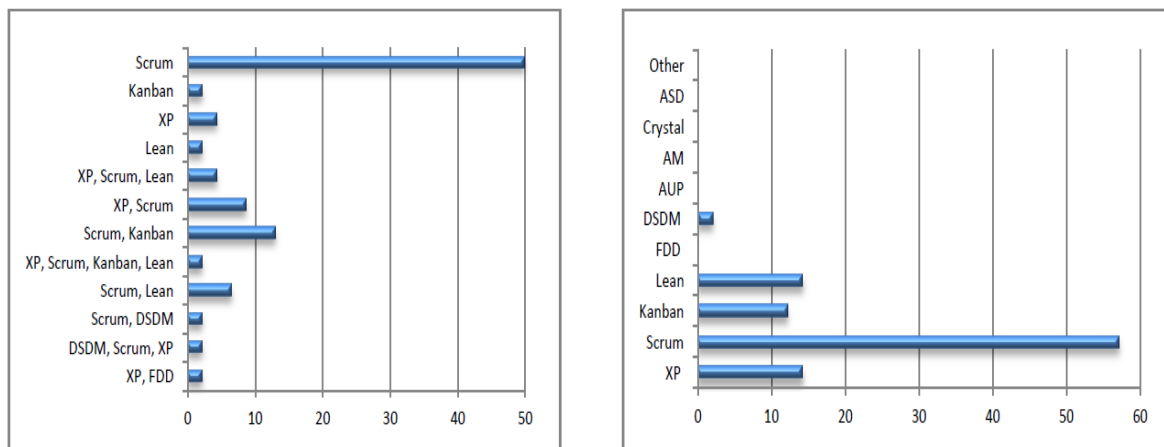


E) Tipos de APP que desarrollan los encuestados.

Figura 2.1 A), B), D) y E). Resultados de la encuesta a desarrolladores de APPs a nivel mundial.

Fuente: Harlem et. al. (2014).

Se les preguntó también a las empresas que tipo de metodologías utilizaban y cuál era la más apropiada para el desarrollo de una APP, en la figura 2.2 A) y B) se muestra el resultado.



A) Metodología ágil que cree que es más apropiada. B) Metodología de desarrollo que siguen las empresas.

Figura 2.2 A) y B). Metodologías que conocen y que siguen las empresas encuestadas.

Fuente: Harlem et. al (2014).

Como se observa, los resultados de la encuesta arrojaron que las metodologías más conocidas son: *Scrum*, *Scrum-Kanban*, *XP-Scrum*, *XP-Scrum-Lean*, *Scrum-Lean*, entre otras. Las metodologías que más utilizaron en los proceso de desarrollo fueron: *Scrum*, *Lean*, *Kanban*, *XP* y *DSDM*. En los temas siguientes se describe con más detalle cada una de estas.

2.2 El mercado de las APP en México.

En México se tiene poco conocimiento sobre como las empresas desarrollan aplicaciones móviles, y que metodologías o herramientas utilizan. La AMITI desarrolló un análisis de esta industria en el año 2013 llamado “Estudio de perspectivas y estrategias de desarrollo y difusión de aplicaciones móviles” (AMITI, INFOTEC, 2013). Se menciona que en México hasta mayo de 2012, se tenía una penetración del 20% de teléfonos inteligentes entre la población, se contaba con 102 millones de dispositivos, de los cuales:

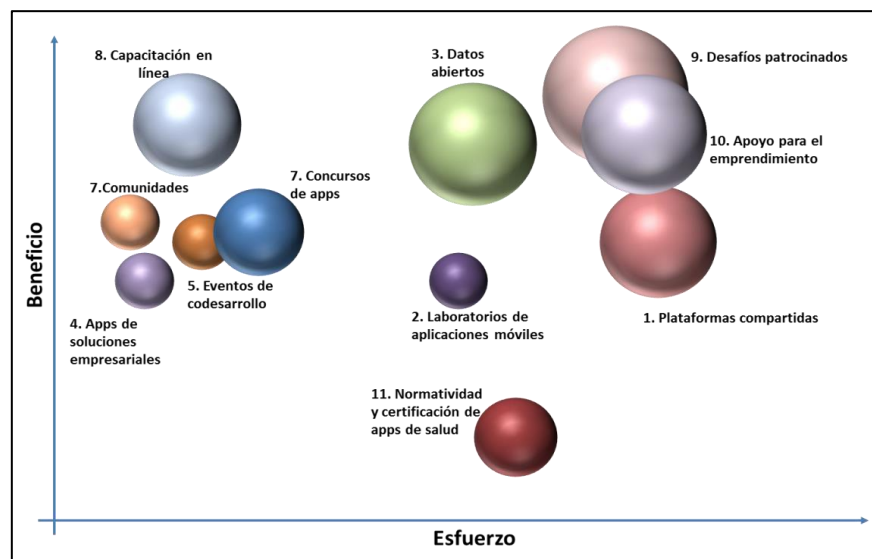
- El 25% utilizan el sistema operativo (OS) (BlackBerry, 2015).
- El 42% de las descargas de aplicaciones se realizaron con el OS de *Android* (Google Play, 2015) y móviles Samsung.
- Otro 40% utilizó el *App Store* de *Apple* (Apple IOS, 2015).

- Y el resto utilizó sistemas como *Windows Phone* (Microsoft, 2015), *Symbian* (proyecto cerrado), *Bada* (proyecto cerrado), entre otros.

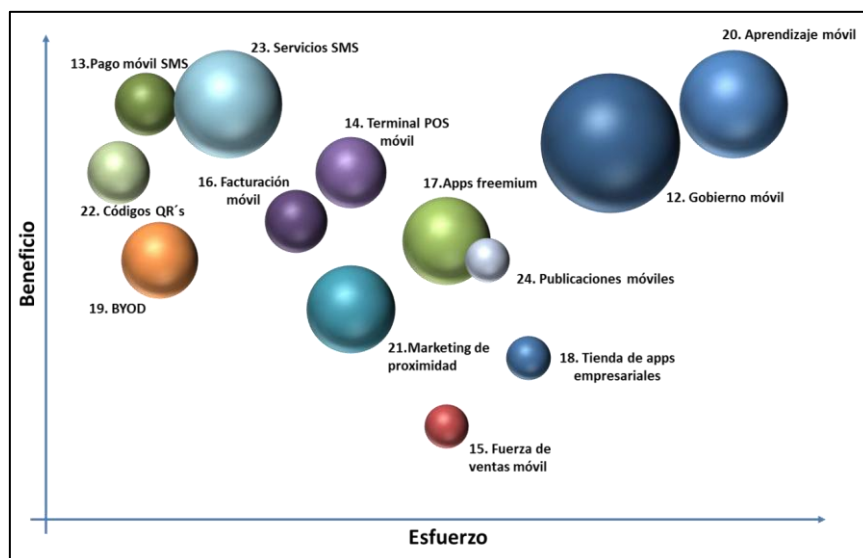
México se ubicó como el segundo mercado más grande de la región en términos de descarga de aplicaciones, donde las más populares son gratuitas (65%) de este porcentaje los juegos tienen el 30% y las herramientas (donde se incluyen APP para de redes sociales) 35%, mientras que las APPs de productividad tiene un porcentaje de 4%.

El entorno de desarrollo que rodea a los dispositivos móviles, ha permitido que miles de desarrolladores utilicen diferentes plataformas para la realización de aplicaciones que aprovechan la movilidad y los beneficios tecnológicos como la pantalla capacitiva, grabación de video y servicios de localización. Sin embargo, para ello se requiere también potenciar la cobertura y eficiencia de las redes móviles que soporten el tráfico de datos a través de tecnologías 3G, 4G y las mejores prácticas en el desarrollo de una APP. Se tienen estimaciones que para el año 2016 el mercado de telefonía móvil podría llegar a 173 millones de dispositivos. La masificación de los dispositivos y el incremento en sus capacidades ha llevado a que el 64% de los usuarios los utilicen en prácticamente cualquier situación, principalmente en casa y trabajo.

En México uno de los grandes problemas del desarrollo móvil es la utilización de las mejores prácticas para la difusión y venta en el uso de esta tecnología, en la figura 2.3 A) y B) se muestra los esfuerzos de las empresas de esta industria para difusión y venta de tecnología móvil.



A) Mejores prácticas para la difusión.



B) Mejores prácticas para la venta.

Figura 2.3 A) y B). Análisis de Esfuerzo-Beneficio para las mejores prácticas de difusión y venta.
Fuente: AMITI (2013).

El estudio de la AMITI se realizó también un análisis de la cadena de valor y los procesos que la integran así como un análisis de las fuerzas de Porter (Porter, 2009) para esta industria. En la figuras 2.5 y 2.4 se muestran los resultados.

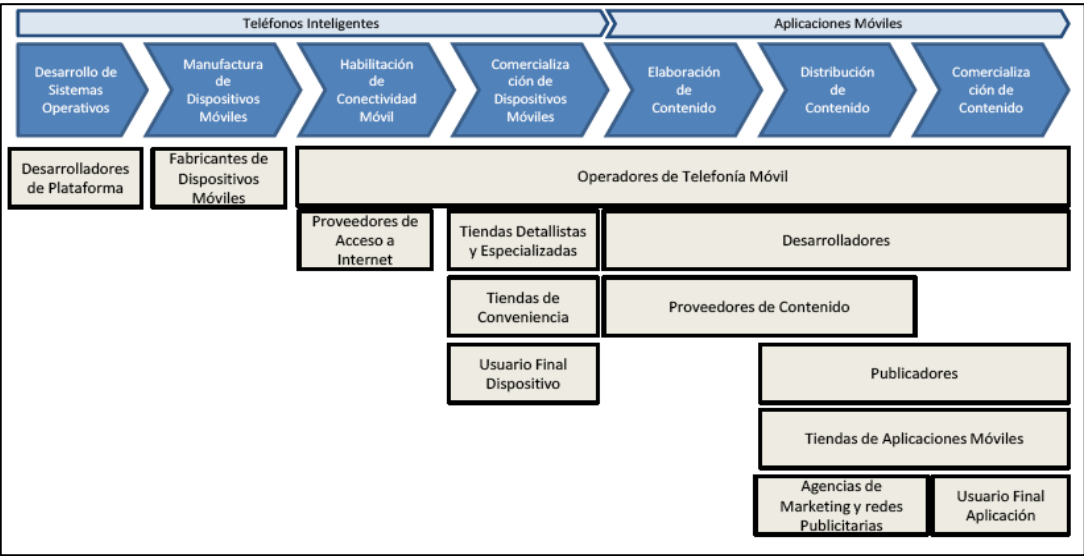


Figura 2.4. Cadena de valor en la industria del desarrollo móvil.
Fuente: AMITI (2013).

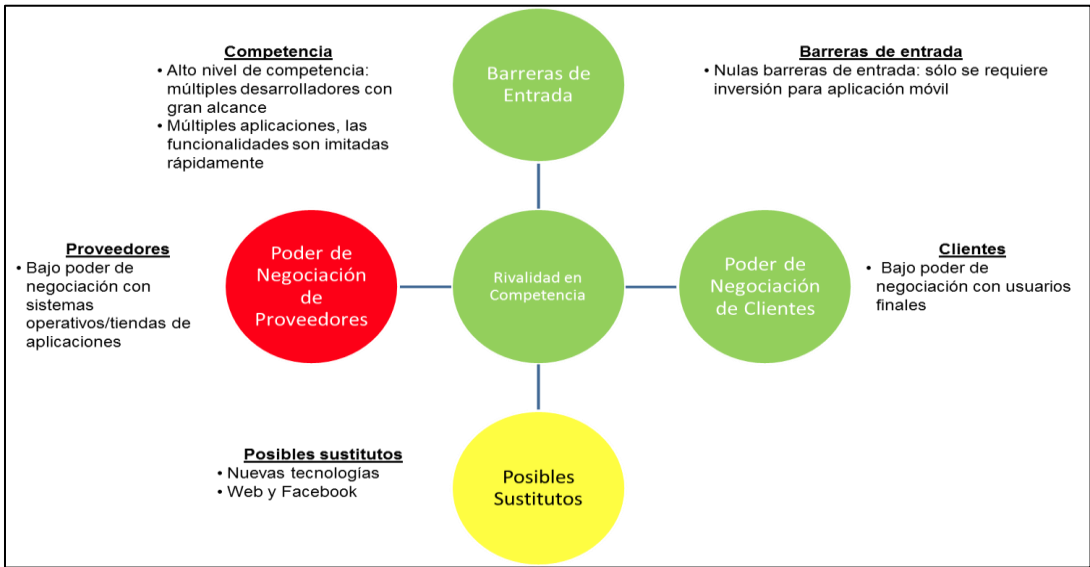


Figura 2.5. Análisis de las fuerzas de Porter para esta industria.
Fuente: AMITI (2013).

Como se observa en la figura anterior 2.5 esta industria es global y existen muchos competidores que están desarrollando aplicaciones, se observa que en México los clientes tienen un bajo poder de negociación y todavía falta desarrollar más APPs que solucionen sus necesidades (mercado regional), no se tiene un control sobre las tiendas y el desarrollador se debe apegar a sus políticas de calidad y uso.

Finalmente se construyó el perfil de demanda de aplicaciones móviles, que fue enfocado a las 17 actividades económicas más relevantes para el país, e identificaron los tipos de aplicaciones y funcionalidad global disponible para cada una de ellas, de acuerdo a las actividades primarias y de soporte de cada sector. En la figura 2.6 se muestra este perfil.

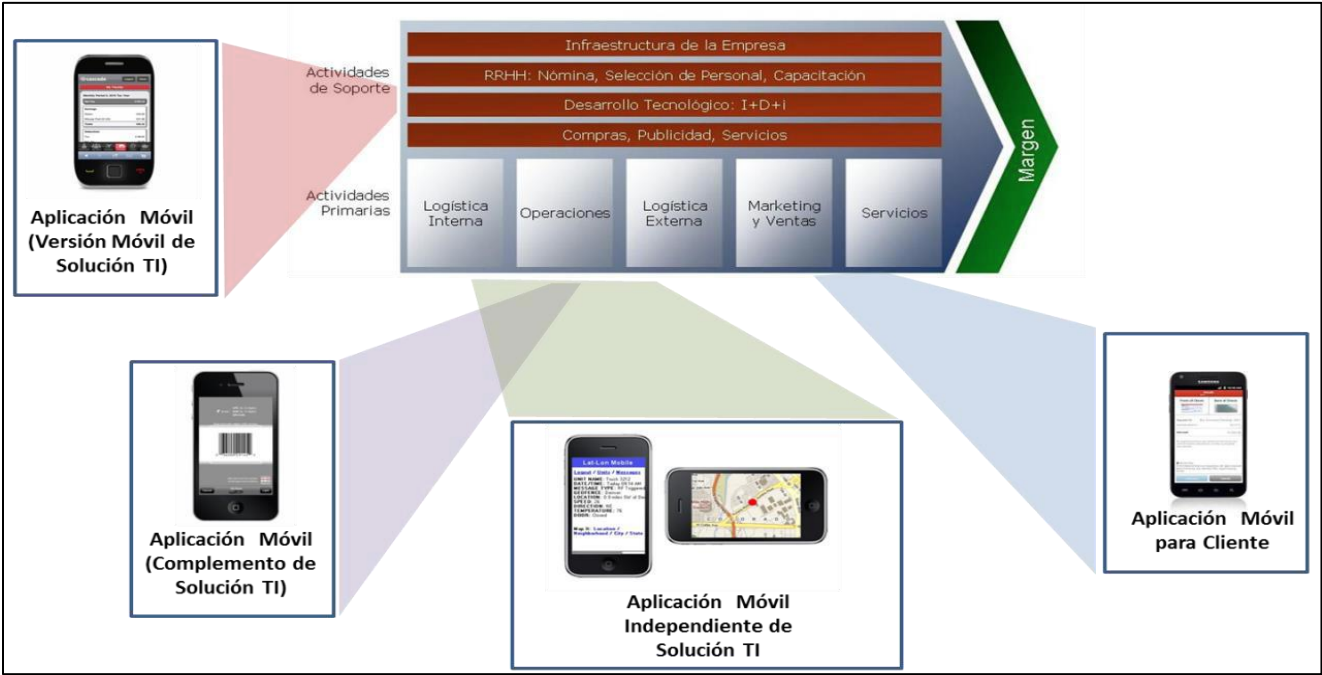


Figura 2.6. Perfil de la demanda por sector.
Fuente: AMITI (2013).

Como parte final del estudio se concluye que: “Con base en la evaluación de las estrategias contra factores internos y externos del entorno y las PyMEs mexicanas, se recomienda iniciar con acciones de estrategia de liderazgo en costos con algunas acciones de planeación estratégica”.

En (Fonseca, 2014) se hace un análisis del mercado móvil en México, se obtuvieron las siguientes estadísticas, ver figura 2.7.

Tipo	Población y porcentaje Año 2014.
Usuarios de Internet	60.1 millones (55%).
Teléfonos Celulares	101.3 millones (85%).
Smart Phones	23.5 millones (20%).
Tablets	3.8 millones (3%).
Usuarios de Facebook	52.8 millones (45%).

Figura 2.7. El mercado de dispositivos móviles en México, basado en una población de 118.4 millones de habitantes.
Fuente: Fonseca (2014).

El uso masivo de este tipo de dispositivos y la oportunidad que se observa en este mercado hace que muchas PyMEs se constituyan como desarrolladoras de APPs. No se tiene estudios consistentes de los métodos de desarrollo que utilizan estas empresas para competir.

2.3 Estado del arte en el desarrollo de APPs.

El rápido crecimiento del desarrollo móvil hacia nuevas aplicaciones ha creado una nueva industria en el software y quizá es el reto más grande para el desarrollo de software. En su artículo (K. Dey, D. Abowd, & Salber, 2001), mencionaban que el siglo veinte la demanda de aplicaciones en diversos entornos y el uso del cómputo ubicuo será una prioridad, uno de los grandes retos en esta industria es la facilidad de crear aplicaciones basadas en el contexto (interacción entre humanos, aplicaciones y su entorno), estas aplicaciones deberán de facilitar la interacción entre humanos y los dispositivos ubicuos. Desafortunadamente el desarrollo basado en el contexto todavía está en pleno desarrollo ya que los cambios y las mejoras en el cómputo ubicuo deben adaptarse a las nuevas tecnologías que surgen día a día Salvaneschi et. al. (2012). Se definen tres grandes objetivos para el desarrollo de aplicaciones rápidas basadas en el contexto:

1. Analizar el contexto sobre el cual se desarrollará la aplicación,

2. utilizar un marco de desarrollo estandarizado para el diseño de aplicaciones basadas en el contexto y
3. construir herramientas que permitan el desarrollo de este tipo de aplicaciones.

La metodología que proponen K. Dey et. al. (2001) definía los pasos siguientes:

1. Identificar Entidades. Utilizando el AOO (análisis orientado a objetos) encontrar las entidades que interactúan en el desarrollo que se requiere, además de su contexto.
2. Identificar atributos contexto. Que propiedades/atributos tienen las entidades de contexto y su interacción entre ellas.
3. Identificar la calidad de los requerimientos de servicio. Restricciones de cada uno de los atributos contexto y su dinamismo en las interacciones.
4. Seleccionar sensores. Se identifican las piezas del contexto que se necesita adquirir y la calidad del requerimiento del servicio para cada uno de ellos.
5. Derivar un diseño de software. Se requiere derivar en un diseño de software basado en el análisis de las entidades, sus atributos y el contexto. Se mapea entidades a componentes, los atributos contexto se derivan a *Widgets* (pequeños programas) y finalmente identificar elementos para abstraer o inferir sobre la información del contexto.

En el desarrollo basado en el contexto era necesario contar con un middleware (software intermediario para acceder a un servicio) que permitiera la interconexión e interacción entre entidades – estos ya son una realidad en nuestros días -, los más comunes son:

1. Abstracciones. *Widgets* para la creación de sensores, que permiten la interacción entre entidades remotas.
2. Agentes. Software que soporta aplicaciones multimodales y ambientes de ubicuidad (servicios en red).
3. Espacios de tuplas. Necesarios para almacenar, buscar e investigar elementos o servicios basadas en la red.

Actualmente existen marcos de desarrollo basados en el contexto y que han ayudado al desarrollo de una APP, estas propuestas ya son parte de los entornos de desarrollo integrados (IDE, por sus siglas en inglés) y se han agregado a las nuevas metodologías del desarrollo móvil. Una APP es una representación y adquisición del contexto, que se hace a través de la definición de una estructura taxonómica de general a lo particular. En su diseño y construcción debe tomarse en cuenta los atributos y entidades del contexto sobre las que funcionará, debe considerar propiedades como: privacidad, definición sin ambigüedad de los datos del contexto, programación de alto nivel de las necesidades de los usuarios, entre otras características apegadas a las necesidades de los usuarios.

Si bien la metodología propuesta por K. Dey et. al. (2001) es útil para definir el contexto de una aplicación basada en la ubicuidad, en la actualidad existen nuevas tendencias que permiten el desarrollo de aplicaciones móviles en ambientes distribuidos (en el contexto de una red). En De Silva et. al. (2013) explican que las futuras tecnologías relacionadas con dispositivos inteligentes harán la vida más fácil, el ahorro en el tiempo de una persona que utiliza esta tecnología en forma ubicua es el impacto mayor. Los dispositivos móviles ya han añadido valor a nuestras tareas cotidianas. Sin embargo, los programadores de este tipo de desarrollo, hasta el momento, parece que se les ha complicado el uso de estas tecnologías y sobre todo el desarrollo rápido con este tipo de dispositivos inteligentes, el contexto, sus propiedades y la ubicuidad es quizá el principal problema.

En la tesis de Spataru (2010), se hace un análisis de los métodos de desarrollo ágil para el desarrollo de APPs. Las plataformas móviles en los últimos cinco años han experimentado una rápida expansión y cada vez van mejorado en su rendimiento – que está cercano a una máquina PC- , por lo tanto el dispositivo móvil ya ofrece una sólida alternativa para los usuarios; que junto con los recursos de red son una prioridad de uso.

El desarrollo de software para plataformas móviles viene con características únicas y las metodologías que más utilizan se centran principalmente en las etapas de “planeación y desarrollo”, del ciclo de vida del desarrollo de software. El entorno de desarrollo y las tecnologías

que apoyan al software en dispositivos móviles son diferentes en comparación a los valores "tradicionales" del desarrollo describe Spataru (2010). Las características distintivas más importantes se identifican en (Abrahamsson, et al., 2004), las particularidades del entorno de desarrollo móvil son: un alto nivel de competitividad, necesariamente un corto tiempo para la entrega y dificultades para identificar a los interesados y sus necesidades. Los equipos de desarrollo deben hacer frente al desafío de un entorno dinámico, con modificaciones frecuentes en las necesidades y expectativas del cliente.

Las limitaciones técnicas se aplican a las plataformas móviles en forma de recursos físicos limitados y especificaciones que cambian rápidamente. Otro problema en el desarrollo es también que hay una gran variedad de dispositivos, cada uno con características particulares de hardware, firmware y sistemas operativos.

Otra vista de las restricciones asociadas a las aplicaciones móviles se presenta en (Hayes, 2003). El autor menciona dos tipos de restricciones, en constante evolución e inherentes. 1) Restricciones en evolución; tales como ancho de banda, cobertura y seguridad. Por otra parte 2) las limitaciones inherentes, tales como el espacio en pantalla limitada, la capacidad de entrada de datos reducida (debido a un teclado limitado, por ejemplo), la capacidad de memoria, potencia de procesamiento, el arranque y la ejecución limitada son permanentes, por lo menos en relación con los entornos de escritorio.

De lo anterior surge la necesidad de que diversos "enfoques" se deben de utilizar en el desarrollo con el fin de reducir el impacto de las restricciones inherentes del desarrollo móvil. Los desarrolladores han atacado estas limitantes mediante el uso del desarrollo ágil, de los métodos adelgazados (Lean) de desarrollo, estos dos enfoques han recibido tanto apoyo y en ocasiones oposición. El principal argumento "en contra" de los métodos ágiles es la presunta falta de la validación científica de las actividades y prácticas asociadas, así como la dificultad de integrar las prácticas basadas en un plan o falta de planeación. De hecho, algunos proyectos presentan una mezcla de características tradicionales con métodos ágiles, en cuyo caso debe lograrse un equilibrio en el uso de ambos tipos de métodos (Bohem, 2002).

Hay también incertidumbre en la distinción de los métodos ágiles de programación ad-hoc a las necesidades del desarrollo de APPs. Sin embargo, como se indica en (Salo & Abrahamsson, 2006), los métodos ágiles proporcionan un enfoque de desarrollo organizado.

En (Rahimian & Rasmin, 2008) se realiza una comparación directa entre las características del método ágil y características de las APPs, centrado en la volatilidad del entorno, la cantidad de documentación producida, la cantidad de planificación de los involucrados, el tamaño del equipo de desarrollo, la ampliación de la aplicación en el desarrollo, la identificación del cliente y la orientación a objetos. Excepto la identificación del cliente, todas las demás características hacen que los métodos ágiles sean una forma adecuada para el desarrollo de aplicaciones móviles. El cliente puede ser identificado como el distribuidor de software. Sin embargo, especialmente en el caso de las aplicaciones móviles, el problema de la identificación del cliente es mucho más complejo debido a las características cambiantes de las plataformas móviles y al contexto de la ubicuidad.

2.3.1 Metodología DIMP.

El Entorno de Desarrollo Integrado Distribuido para plataformas Móviles (*DIMP*, por sus siglas en inglés) descrito por De Silva et. al. (2013), se orienta hacia una innovadora forma de escribir programas de software. Para ayudar a la utilización de un dispositivo móvil, o un teléfono móvil, una tableta o incluso una máquina personal, *DIMP* permite escribir códigos fuente y de compilar en un entorno distribuido. Consiste en una aplicación móvil, un servidor central y un conjunto de servidores de compilación, una consola administrativa basada en una interface web que ayuda con las funciones administrativas. Ver figura 2.8.

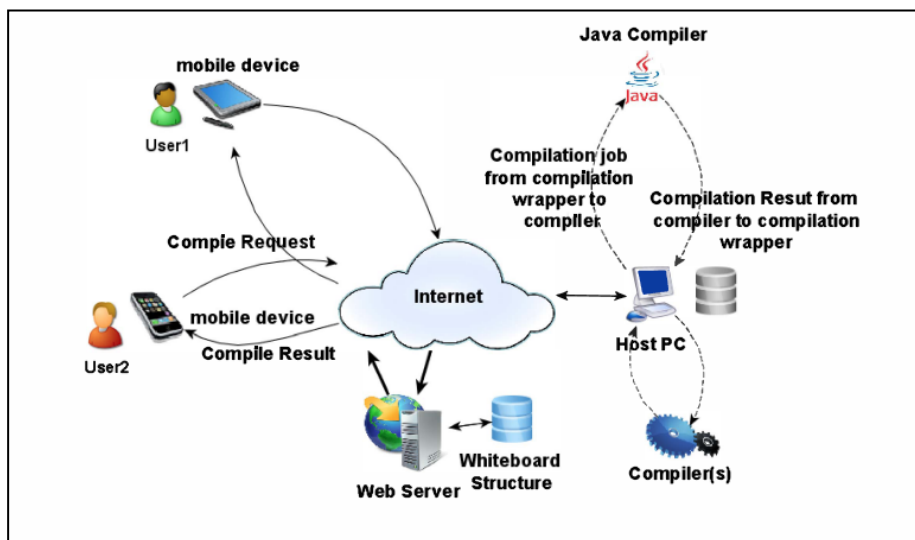


Figura 2.8. Diagrama que representa a la arquitectura de alto nivel de un entorno de programación móvil con *DIMP*.
Fuente: De Silva et. al. (2013).

La aplicación móvil es un software desarrollado para el sistema “*Android*”, que proporciona un editor completo de código fuente integrado. Permite compilar y ejecutar los códigos desde el origen donde los usuarios pueden escribir programas en el idioma seleccionado. Si el código fuente está libre de errores, un usuario puede esperar una salida válida, mientras que un código fuente con errores genera un mensaje de error, o un mensaje con sugerencias útiles para la depuración. Un beneficio de *DIMP* es que permite a un usuario mantener el espacio de trabajo en línea así como un espacio de trabajo fuera de línea (si estar conectado en la red). Los códigos fuente se pueden compartir con otros usuarios interesados. Es una propuesta de desarrollo de nueva generación en donde las APPs se generan utilizando una herramienta en línea, de rápido desarrollo y con código reutilizable basado en componentes, que al integrarlos proveen una solución. Una propuesta similar a *DIMP* es el “*APP Inventor*” del *Massachusetts Institute of Technology* (MIT) que provee a través de una plataforma el desarrollo rápido de aplicaciones y el prototipado (MIT, 2013).

El entorno de trabajo de DIMP puede observarse en figura 2.9.

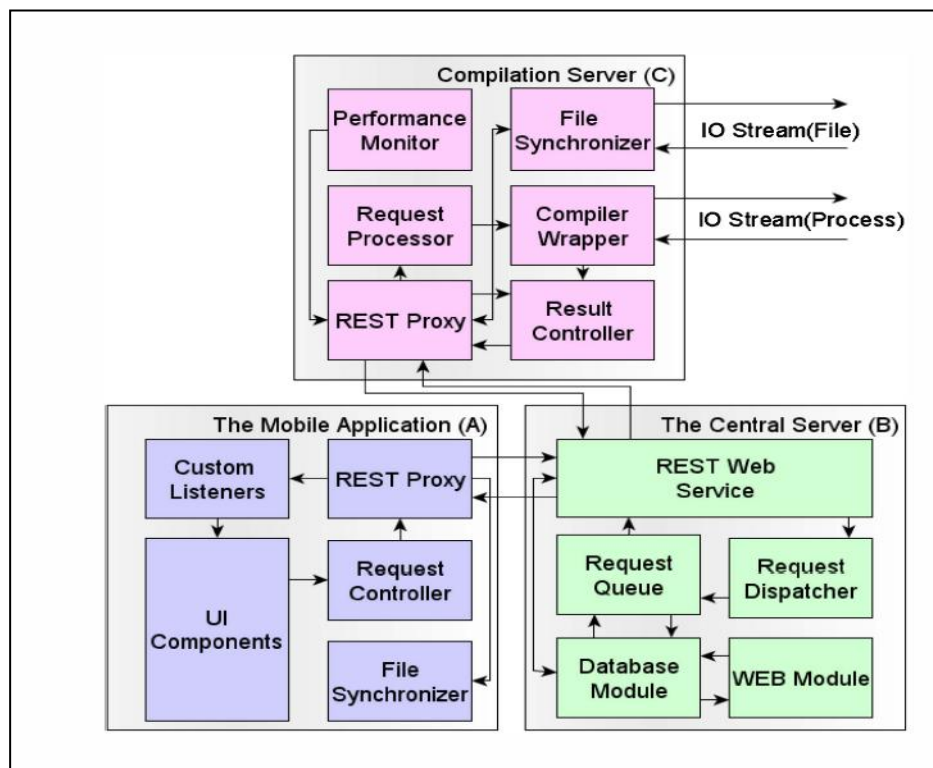


Figura 2.9. Diagrama que representa la estructura de DIMP.
Fuente: De Silva et. al. (2013).

Como se observa *DIMP* ofrece una excelente alternativa para el desarrollo rápido de aplicaciones, pero sigue careciendo de una metodología que permita agregar mejoras en el desarrollo y ofrecer un software de calidad.

2.3.2 Metodología MASAM.

En (Jeong, Lee, & Shin, 2008) proponen una metodología llamada MASAM (*Mobile Application Software based on Agile Methodology*) que permite el desarrollo de una APP con calidad utilizando metodologías ágiles. Además de MASAM existen otras utilizadas para este propósito como XP y RUP (solo algunas etapas para hacer ligero el desarrollo) (Ambler, 2014) que han ido en aumento. El enfoque de estas se centra en el grupo de programadores y en la etapa de desarrollo del software (comúnmente incluida en todas las metodologías). Las acciones del equipo de desarrollo están orientadas a una elección rápida del tipo de producto que desea, se construye

y se lanza al mercado lo más rápido posible. Esta característica encaja perfectamente con la necesidad en la actualidad de desarrollar aplicaciones en forma ágil y que el usuario pueda utilizarlo en un corto tiempo.

El problema del desarrollo móvil es que basa en dispositivo que aún tiene restricciones de capacidades y recursos limitados, por lo que la construcción de una APP compleja es limitada. Se ha observado también que el desarrollo móvil es realizado por empresas pequeñas que se ven limitadas en personal y en infraestructura, por lo que las metodologías clásicas no les resultan útiles. Existe una tendencia a nivel internacional en el desarrollo de meta-modelos como marco conceptual para ofrecer una metodología este es conocido como SPEM (*Software & Systems Process Engineering Metamodel Specification*) (OMG, 2015), la metodología MASAM se basa en un meta-modelo que ofrece las ventajas siguientes:

1. Separación clara del método de definición y del contenido del proceso para el desarrollo de una APP.
2. Mantenimiento consistente de los diferentes procesos para el desarrollo.
3. Diferentes modelos del ciclo de vida de un proyecto.
4. Variabilidad, flexibilidad de procesos y extensibilidad de mecanismos (plug-in).
5. Patrones de procesos reusables, para mantener las mejores prácticas, además del uso de procesos de ensamblado rápido.
6. Componentes de procesos reemplazables y reutilizables, utilización de los principios de la encapsulación y el DOO.

En MASAM se definen tres procesos que son la base de la metodología:

1. Asignación del rol de los participantes. El que elabora el plan, el administrador, el diseñador, el desarrollador, definir un equipo de desarrollo, un equipo de arranque, un probador, un usuario, etc.
2. Asignación y seguimiento de tareas. Que incluye actividades como; producto resumen, planeación inicial, definición de usuario, análisis inicial, recurso seleccionado, selección de proceso, establecimiento del entorno, escritura de storycard, diseño de interfaces, etc.

3. Producto trabajo. Resumen del producto, planeación del proyecto, ejemplo de usuario, ejemplo de interface, modelo de interface, patrón de arquitectura, patrón de aplicación, storycard (historias de usuario), taskcard (tarea a desarrollar), modelo de arquitectura, modelo de componente, caso de prueba, entre otras actividades.

Además de la definición de procesos se proponen 4 fases para aplicar la metodología, que se describen a continuación:

1. Fase de preparación del desarrollo. Esta fase se desarrolla en conjunto con el cliente, el consumidor y el equipo de desarrollo. Se definen tres grandes actividades: i) Entender el producto; resumen y pre-planeación, ii) Compartir el concepto del producto; definición de usuario, análisis inicial del producto y iii) Iniciar el proyecto; coordinar procesos de desarrollo, coordinar recursos del proyecto y estudio previo.
2. Fase de materialización (encarnación). El consumidor del producto debe estar representado en forma concreta. Los requerimiento del usuario deben ser gradualmente implementados en código, se recomienda en esta fase la construcción de un prototipo que simule la interface de usuario. Las actividades que se desarrollan en esta fase son; i) Entender las necesidades del usuario; trabajar con el storycard, proponer el diseño de la interface de usuario. ii) Construcción de la arquitectura; análisis de requerimientos no funcionales, definición de la arquitectura y administración del patrón de producto.
3. Fase de desarrollo del producto. En esta fase el producto debe ser construido gradualmente e irse liberando al cliente; quién debe verificar el resultado y revisar y es ad-hoc a lo que se requirió. Las actividades en esta fase son: i) preparación de la implementación; iniciar el entorno, planeación del desarrollo. ii) Ciclo de liberación; ciclo de interacciones, planeación de iteración, ciclo de implementación, retroalimentación, control de entregas y pruebas de aceptación.
4. Fase de comercialización. Las aplicaciones móviles operan en diversas plataformas, por lo que debe de estudiarse en que entorno operará, las políticas de entrega y el entorno o contexto del país en donde se hará la liberación. Las actividades de esta fase son; i)

Prueba del sistema, prueba de aceptación y prueba de usuario. ii) Venta del producto; prueba de lanzamiento y lanzamiento del producto.

Es importante señalar que MASAM es una propuesta para desarrollar aplicaciones basada en el ciclo de vida del desarrollo de un producto y además en un marco de desarrollo de software. Existen varias desventajas que no se contemplan en diseño como el control de la calidad y la falta de herramientas para seguir el proceso de desarrollo. Ver figura 2.10.

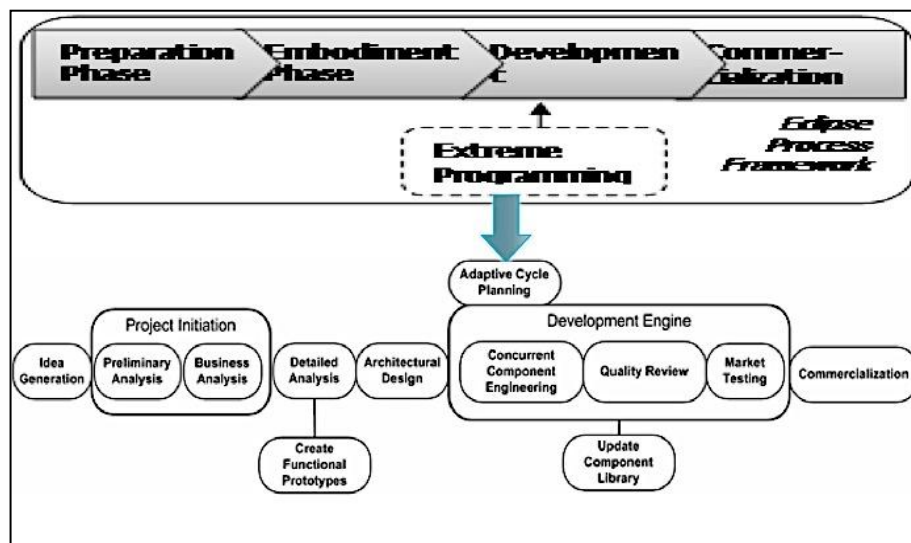


Figura 2.10. Etapas de la metodología MASAM híbrida con XP.
Fuente: Jeong et. al. (2008).

2.3.3 Metodología Mobile-D.

Una alternativa para el desarrollo de aplicaciones móviles es la metodología Mobile-D propuesta por Abrahamsson (2004), que se basa principalmente en los métodos de desarrollo XP y Crystal, además de utilizar algunos elementos de RUP. La metodología Mobile-D comprende cinco fases y se muestran en la figura 2.11.

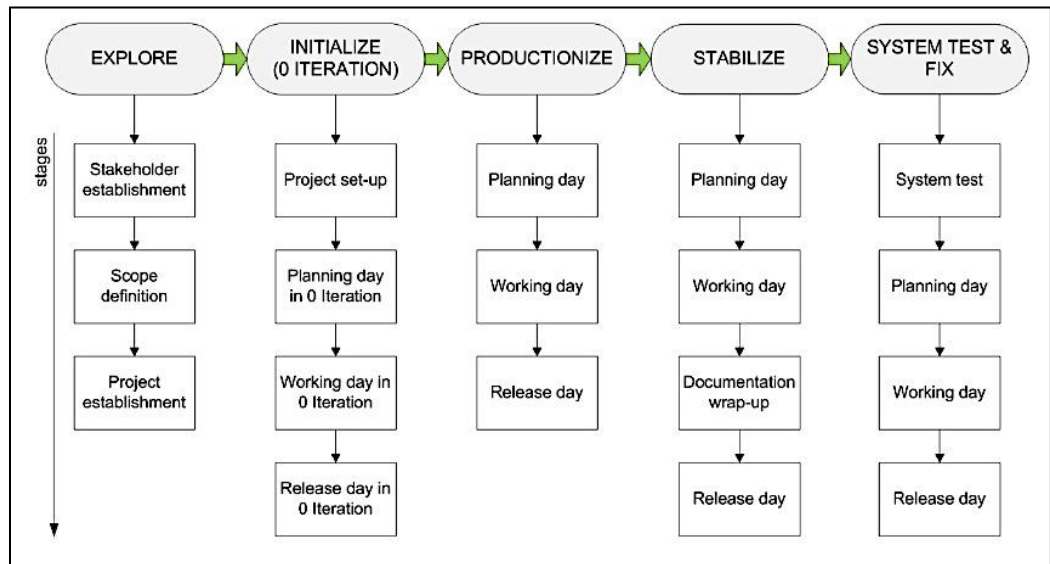


Figura 2.11. Fases y estados de la metodología Mobile-D.

Fuente: (VTT Electronics, 2008)

Como se observa la metodología ofrece una serie de fases y estados para el desarrollo de aplicaciones en forma ágil. En la tabla 2.1 se describen los documentos que apoyan a la metodología en cada una de sus fases.

Tabla 2.1. Descripción de las fases y tareas en la metodología Mobile-D.

Fuente: elaboración propia.

Fase	Tareas a realizar	Descripción
Exploración	Establecimiento de los interesados y del cliente.	Se define quiénes serán los que participarán en el proyecto, los roles del grupo de dirección, grupo del cliente, estructura del grupo, se establece el propósito del producto, el contrato, personas responsables, se define el plan del proyecto, se define el documento de requerimientos, estándares relevantes, se define el tamaño del proyecto, se definen y aseguran los recursos; humanos, técnicos, asociados al entorno de trabajo. Se define el plan de entrenamiento, se define la línea base a seguir, se definen prácticas, métricas y herramientas.
	Definición del alcance del proyecto	
	Establecimiento del proyecto.	
Iniciación (por iteración)	Configuración del proyecto.	En esta fase se configura los recursos físicos necesarios, los recursos técnicos, se familiariza al equipo del proyecto con las herramientas y los procesos, se establece un acuerdo con los clientes. Se hace una compresión global del producto, se prepara el proyecto y la
	Planeación inicial (día de planeación iteración)	

	0).	estructura de planeación, se preparan los planes para resolver los puntos críticos en el desarrollo. En el estado trabajo se garantiza la funcionalidad del entorno de desarrollo técnico, se implementa el núcleo de funcionalidad del sistema. Se libera el producto en su versión de iteración para iniciar la producción o programación.
	Día de trabajo (iteración 0).	
	Día de lanzamiento (iteración 0).	
Producción	Día de planeación.	El propósito de la fase de producción es implementar la funcionalidad requerida en el producto mediante la aplicación del ciclo de desarrollo iterativo e incremental. En el día de planeación se selecciona el plan de trabajo para cada iteración y se definen las tareas a realizar, se crean formatos o plantillas para registro del avance. En el día de trabajo se implementa lo planeado en el día, debe apegarse el desarrollo a lo proyectado con el cliente, se registran las tareas y el código generado debe estar bien documentado. En el día de lanzamiento debe tenerse una versión completa desarrollada de las funcionalidades planeadas.
	Día de trabajo.	
	Día de lanzamiento.	
Estabilización	Día de planeación.	El propósito de esta fase es asegurar la calidad de la implementación del proyecto. En el día de planeación se define el contenido (historias y tareas) para la ejecución de las funciones del producto y para la mejora de la calidad externa e interna del producto (refactorizar). En el día de trabajo se finaliza la implementación de acuerdo a lo planeado. En la recapitulación se documentan la arquitectura del software, el diseño y la interfaces de usuario. En el día de lanzamiento se verifica y valida la funcionalidad y la calidad del software y su documentación. Se tienen formatos de documentos donde se captura todas las actividades.
	Día de trabajo.	
	Recapitulación de documentación.	
	Día de lanzamiento.	
Prueba del sistema y arreglos	Pruebas del sistema.	Finalmente se realiza la fase de prueba del sistema y arreglos, es para ver si el sistema que se produjo esta acorde a las funcionalidades definidas por el cliente, además de la retroalimentación del equipo de proyecto que verificó la funcionalidad y corregir los defectos encontrados. El día de planeación se define el contenido de los arreglos y se planean. En el día de trabajo se realizan todas las correcciones planeadas y se recapitula la documentación para cada una de estas. Finalmente se hace una verificación de funcionalidad implementada y la calidad del
	Día de planeación.	
	Día de trabajo.	
	Día de lanzamiento.	

		software. Si todo las funcionalidades ya fueron implementadas se hace la liberación final y el cierre del proyecto.
--	--	---

Mobile-D es una metodología completa, sin embargo tiene varias desventajas; aunque se basa en control de calidad y en estándares, el seguimiento por tarea y proceso no está claro lo cual puede llevar a costos extraordinarios a la hora de la ejecución del proyecto, el control de riesgos tampoco es considerado. Otra desventaja es que no se acompaña de un software que permita generar y documentar en forma fácil el desarrollo de una aplicación en forma completa, (Spataru, 2010).

2.3.4. Metodología Scrum para el desarrollo móvil.

En Scharff et. al. (2010), mencionan que el desarrollo de aplicaciones móviles es una tarea difícil, donde la tecnología y la creatividad son esenciales. Las metodologías ágiles son las mejores adaptadas para el desarrollo de aplicaciones en la telefonía móvil, que tienen como características; un alcance y un usuario limitado, la interacción y el tiempo de comercialización son particularmente importantes para lograr una competitividad. Este trabajo consiste en el estudio que evalúa el uso de Scrum como una metodología para asegurar el éxito en el desarrollo de aplicaciones móviles, se define un modelo que utiliza Scrum en un ambiente de desarrollo real. Scrum es un marco iterativo e incremental de uso común con metodologías ágiles. Se utiliza iteraciones de duración fija (de una a cuatro semanas) llamadas “Sprints”. Hay tres roles principales:

- 1) El propietario del producto; es responsable de la obtención de valor del negocio y está a cargo de la visión del producto representado por una lista priorizada de nuevas necesidades, tareas, limitaciones y errores organizados en una “Listado o Pila del Producto”. Los requisitos se describen de manera general por el usuario a manera de historias.
- 2) El equipo de Scrum/entrega (equipo de tres a quince personas); quiénes desarrollan el software.

3) El Scrum Master; que es el encargado de guiar al equipo a través del proceso de desarrollo y ayuda al equipo a resolver los problemas.

Al comienzo de cada Sprint y durante la planificación, el equipo se compromete a completar un determinado número de tareas que se establezcan desde la “pila de producto” y los documentos en un “Sprint Backlog”. Las tareas a las que se han comprometido a los miembros del equipo son realizadas. Al final de cada Sprint hay una demostración/opinión (de las tareas realizadas) y una retrospectiva a manera de aprendizaje. Durante la retrospectiva del Sprint, la velocidad de ejecución, la moral de los equipos y la calidad de los trabajos son evaluadas. El equipo se reúne en forma diaria con el Scrum Master, en donde cada miembro del equipo responde a tres preguntas:

- 1) lo que él/ella han hecho desde la última reunión,
- 2) lo que él/ella espera hacer hasta la próxima reunión y
- 3) cuáles son los obstáculos para llegar a la meta.

La cantidad de tiempo necesario para completar cada una de las tareas se actualiza en la Pila del Sprint. El Scrum Master mantiene una tabla del Sprint realizado “*Burn down*”, como una métrica para medir el progreso del equipo, que muestra para cada tarea la cantidad de horas o días utilizados hasta que se completa el compromiso del equipo. Ver figura 2.12.

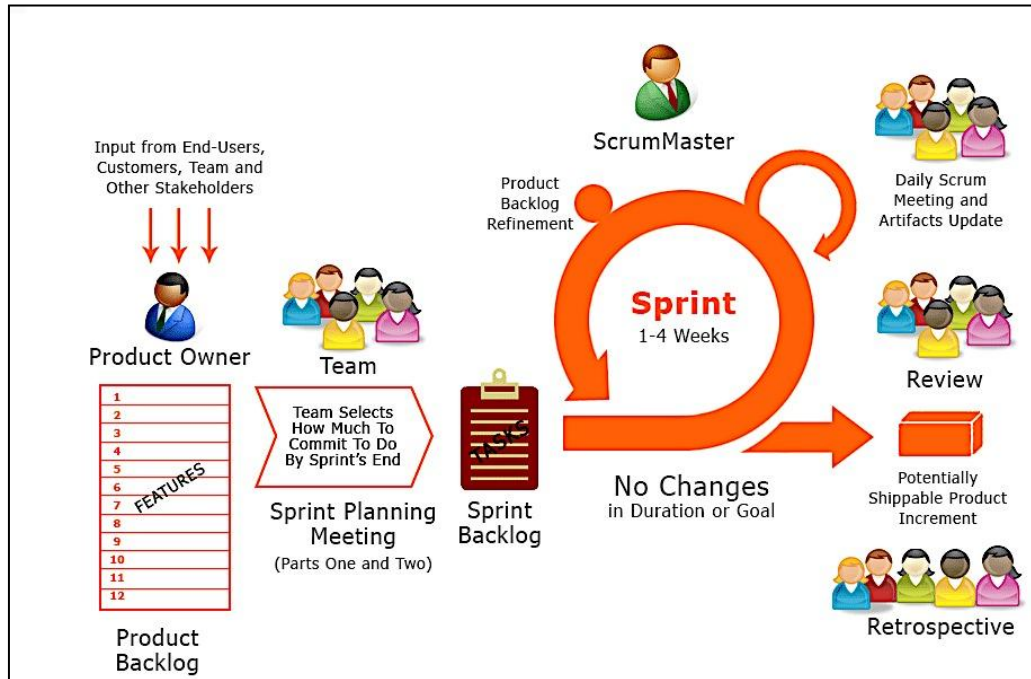


Figura 2.12. Metodología de desarrollo basado en Scrum.
Fuente: Scharff et. al (2010).

Una de las desventajas claras de Scrum es la consolidación del equipo de trabajo y el liderazgo que debe ejercer el *Scrum master*. Los problemas conocidos de la metodología son:

- No se aplica para grandes proyectos, en caso de aplicaciones móviles de gran tamaño esto puede ser una desventaja.
- Se requiere de un experto que revise el cumplimiento de la metodología, lo cual conlleva a una capacitación previa de los encargados del proyecto.
- Las empresas deben adaptarse a la aplicación de la metodología “tal cual” y no viceversa, ya que podría generarse atascos en la aplicación de la metodología.
- Se presupone que el equipo está bien formado y motivado
- Se presupone que el cliente está totalmente involucrado en el desarrollo, participa en forma activa y continua, revisa en forma frecuente el avance y la funcionalidad del desarrollo después de cada Sprint. Se ha observado que muchos clientes se involucran al inicio del proyecto en el establecimiento de los requerimientos y después ya no dedica tiempo en revisar avances.

- Se presupone que el cliente no exige o no necesita de toda la documentación del proyecto y que los diferentes estándares o normativas internacionales comúnmente exigen como prueba de calidad.
- Existen herramientas que permiten llevar el control en la metodología pero no son totalmente colaborativas ni integrales, lo que arroja inconsistencias entre el costo proyectado del proyecto, el tiempo total del proyecto (scope) y los riesgos o costos en que se incurre si un sprint no se realiza en tiempo y forma.

2.4. Modelos de desarrollo de software ágiles para el SPM.

El concepto de gestión de producto fue introducido desde los años 30 en empresas como Procter and Gamble quienes contrataron a una persona específica para asumir la responsabilidad de la creación de un producto, a este proceso se le conoce como Gestión de Productos de Software (SPM, por sus siglas en inglés) es "la disciplina, que regula un producto (o solución o servicio) desde su inicio hasta el mercado de la entrega (cliente) con el fin de generar el mayor valor posible para el negocio" (Maglyas, Nikula, & Smolander, 2012).

El SPM permite generar productos de software de calidad, en donde la rentabilidad y la previsibilidad desempeñan un papel fundamental en la gestión y el logro de los objetivos de negocio, proporcionando las mejores prácticas para estrategias ganadoras.

Actualmente existe muchos modelos que tratan el desarrollo de software con la filosofía SPM, a este conjunto de métodos es conocido como MDD (*Model Driven Development*) otro conjunto de metodologías son las conocidas como SDLC (*Software Development Life Cycle*, basadas en el ciclo de vida) y el desarrollo de software dirigido por procesos (*Model Driven Software Development*, MDSD), en general todos los enfoques tratan de mejorar el proceso de desarrollo de software (Piatini V. & Garzas P., 2010).

2.4.1 CMMI/CMM.

En (Jones, 2008) se menciona que debido a que grandes proyectos de software a menudo salen fuera de control (crisis del software), se crearon métodos eficaces de desarrollo para ayudar a reducir los problemas. Un método que ha tenido mucho éxito es el modelo de madurez (CMM, *Capability Maturity Model*) y el CMMI (CCM Integration) propuestos por el *Software Engineering Institute* (SEI) (CMMI, 2015). Cuando estos métodos se aplican a proyectos pequeños, por lo general es necesario personalizarlos, ya que, en su forma original, son muy complejos. Recientemente, métodos alternativos como el desarrollo de software ágil ayudan a este tipo de proyectos. Sin embargo es posible fusionar conceptos ágiles con conceptos de CMM y CMMI. En años recientes CMMI, planteó el *Team Software Process* (TSP) y el *Personal Software Process* (PSP), que han ayudado a equipos de desarrollo para proyectos pequeños, estas metodologías apegadas al manifiesto del software ágil (ver más adelante) hacen de CMM/CMMI una posibilidad para aplicarse en proyectos pequeños. Tanto los métodos ágiles como el CMM/CMMI están preocupados por tres problemas fundamentales en el desarrollo:

1. Los requisitos de software siempre cambian.
2. La corrección de errores de software es la actividad que más recursos absorbe.
3. Una alta calidad conduce a una alta productividad y horarios cortos.

A pesar de las similitudes en los métodos Ágiles y el enfoque CMM/CMMI, se contraponen en otros dos problemas fundamentales:

1. Los trámites (convenios, protocolos, firmas, etc.) es la segunda actividad que conlleva costos.
2. Sin mediciones cuidadosas, es poco probable que exista un progreso continuo.

Un método ágil toma una posición firme de que los requisitos en documentos de papel y especificaciones rigurosas son demasiado lentos y engorrosos para ser eficaces. En la vista de una metodología ágil, las reuniones diarias con los clientes son más eficaces que las especificaciones escritas. En un método ágil, las reuniones de equipo todos los días o sesiones (Scrum) son la mejor manera de seguir un progreso, a diferencia de los informes de estado escritos. El CMM/CMMI no

apoya esta opinión. Toman una posición firme de que las mediciones de la calidad, la productividad, horarios, costos, etc., son un complemento necesario para la mejora de procesos y debe hacerse lo mejor posible. En opinión del SEI, es difícil probar que una metodología es exitosa o no, sin datos que demuestran su eficacia, Jones (2008). CMM/CMMI definen 20 actividades claves para el desarrollo de software:

1. Requisitos.
2. Prototipos.
3. Arquitectura.
4. La planificación y estimación de proyectos.
5. El diseño inicial.
6. El diseño detallado.
7. Inspecciones de diseño.
8. Codificación.
9. Adquisición y reutilización.
10. Inspecciones de código.
11. Cambio y control de configuración.
12. Aseguramiento de calidad del software.
13. Integración.
14. Los planes de prueba.
15. Prueba de la unidad.
16. Nueva prueba de la función.
17. Las pruebas de regresión.
18. Las pruebas de integración.
19. Las pruebas de aceptación.
20. Administración del proyecto.

Pueden parecer complejas las actividades que define el CMM/CMMI pero en proyectos reales se ha aplicado en forma extensa, se tiene documentación de más de 13,000 proyectos hasta el año 2010 en donde CMM/CMMI fueron utilizados para la construcción de software con calidad. Sin embargo a pesar de este éxito, se tiene referencias que en empresas pequeñas y medianas esta

metodología no fue del todo exitosa y puede causar muchos retrasos y costos extras. En un análisis de los métodos utilizados en este tipo de proyectos se encontró que existe la utilización de muchos métodos híbridos que demuestran la tendencia de utilizar CMM/CMMI con métodos ágiles:

- Métodos ágiles unidos con desarrollo orientada a objetos (Agile+OOD).
- Métodos ágiles unidos con Lean Six-Sigma.
- Métodos ágiles unidos con TSP y PSP.
- Métodos ágiles unidos con CMM/CMMI.
- CMM/CMMI unidos con Six Sigma.
- CMM/CMMI unidos con certificaciones ISO.
- CMM/CMMI unidos con buenas prácticas de ITIL.
- CMM/CMMI unidos con QFD (*Quality Function Deployment*).
- CMM/CMMI unidos con TSP y PSP.
- XP unido con Lean Six-Sigma.
- ITIL unido con Six-Sigma.
- ISO unido con TickIT.
- OOD unido con el SOA (Service-Oriented Architecture).
- Six-Sigma unido con QFD.
- Six-Sigma unido con ITIL.
- Six-Sigma unido con TSP y PSP.
- SOA unido con ITIL.
- TickIT unido con Six-Sigma.
- Agile unido con OOD, unidos con Six-Sigma e ITIL.
- CMM unido con Agile, unidos con TSP/PSP, unidos con Six-Sigma y DFD.

La metodología propuesta por el CMM/CMMI sigue siendo un referente en la industria del software y principalmente el modelo de madurez es un estándar a seguir, se reconoce que en México varias empresas PyMEs están buscando un nivel de madurez 3, 4 y 5 que define una capacidad instalada para desarrollar software con calidad, ver figura 2.13.

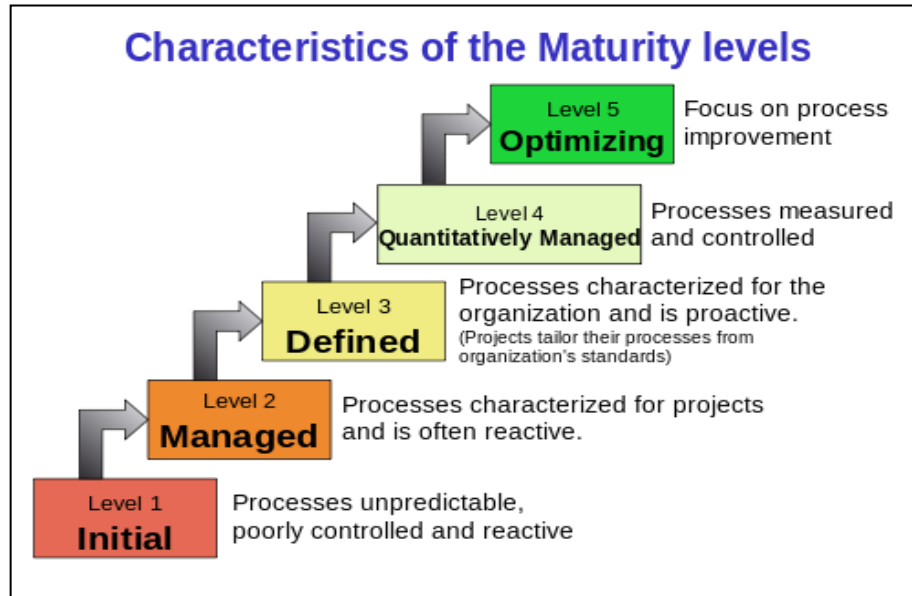


Figura 2.13. Niveles de madurez en el desarrollo de software CMM/CMMI.
Fuente: (CMMI, 2015).

2.4.2 Kanban.

En (Ovais A., Markkula, & Ovio, 2013) se menciona que el uso de Kanban en el desarrollo de software es un tema novedoso, se han reportado beneficios al utilizar este método, se mejoraron tiempo de espera para la entrega de software, mejora la calidad del software, mejora de la comunicación y la coordinación, una mayor coherencia de la entrega y la disminución de defectos a la entrega y revisión del cliente. Los desafíos reportados en el uso de esta metodología se incluyen la falta de conocimiento y la formación especializada, así como diversas cuestiones de organización (integración de equipos, asignación de roles y establecimiento de metas). En la figura 2.14 se muestra el flujo de trabajo.

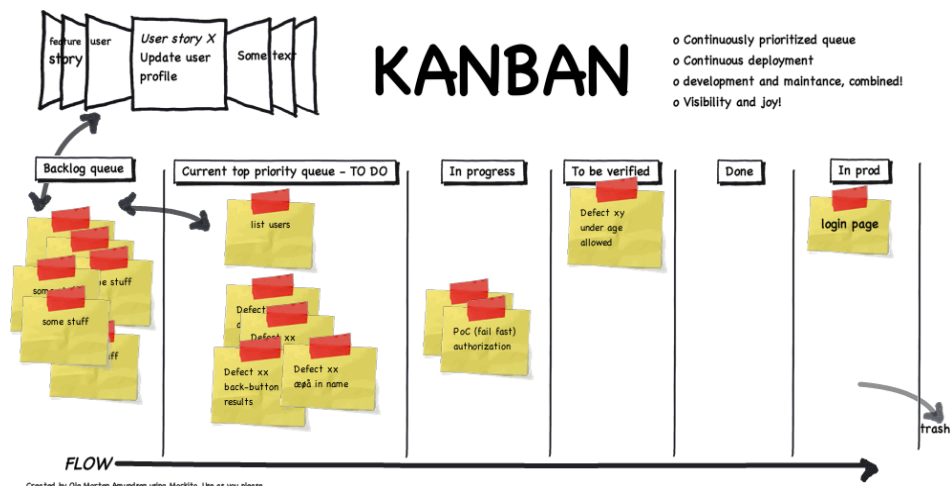


Figura 2.14. Pasos en la metodología ágil Kanban para el desarrollo de software.

Fuente: (Ovais A., Markkula, & Ovio, 2013)

Como se observa Kanban se basa en un sistema visual en el que las diversas tareas que componen un proyecto se muestran como tarjetas en un tablero. El flujo se separa en diferentes etapas, representadas como "carriles" en el tablero Kanban, y las tarjetas se mueven de un carril a otro a medida que progresan a través de diferentes estados, el administrador del proyecto es el responsable junto con el equipo de desarrollo de realizar los avances de la tareas (Cohen & Elalouf, 2014). Es un sistema de tickets que tiene como finalidad:

- Visualizar - Hacer las diferentes tareas más visibles para los gerentes y equipos.
- Administrar el flujo - Gestionar todo el ciclo de vida. Hacer frente a los problemas y mitigarlos a nivel equipo.
- Establecer un sistema de procesamiento de tareas.
- Hacer más claras las políticas de proceso.
- Mejorar la colaboración entre los diferentes equipos.
- Centrarse en los resultados finales.

Este modelo se ha utilizado como una herramienta para la Gestión del Valor Agregado (EVM, por sus siglas en inglés) que permite planificar y supervisar un proyecto ágil, las tareas están orientadas a terminar en tiempo y forma un proyecto, logrando con esto establecer costos y calidad.

En la tesis de (Bergström & Engvall, 2011) se documentó la utilización de Kanban como metodología ágil para un proyecto real del desarrollo de una aplicación móvil para el sector público. La metodología propone el uso de mejoras desde la fase de ejecución y se divide en fases más pequeñas con metas secundarias y evaluaciones recurrentes de procesos. Se elabora un tablero Kanban que ayuda a visualizar las tareas y el flujo de trabajo entre los miembros del equipo y los propietarios de los proyectos para obtener una buena visión de todo el proyecto y la situación actualizada de avance.

Varios ajustes se realizaron durante todo el proyecto y los llamados tiempos de espera; el tiempo que tarda en completar una tarea, se acortó en promedio desde el inicio del proyecto en comparación con el final del proyecto. Cada fase en la parte de la implementación del proyecto dio lugar a diferentes funcionalidades en la App. Las fases duraron en promedio dos semanas y terminaban con una demostración de una nueva funcionalidad; donde participan todos los miembros del proyecto para dar su opinión ofreciendo con esto calidad en las aplicaciones. La metodología resultó ser una herramienta adecuada para la implementación del proyecto; mediante la medición de los tiempos de entrega y otras actividades que daban certidumbre del desarrollo y además los clientes siempre estuvieron conformes con el desarrollo de la App.

2.4.3 Moprosoft.

En el año de 2002 el gobierno mexicano lanzó una iniciativa conocida como el Programa para el Desarrollo de la Industria del software (Prosoft) (Secretaría de Economía, 2014) cuya finalidad es la de promover competitividad de la industria del software a nivel nacional. Las estrategias de esta iniciativa son las siguientes:

1. Promover exportaciones y atraer inversiones,
2. Crear programas de educación y formación de personal competente,
3. Contar con un marco legal promotor de la industria,
4. Desarrollar el mercado interno,
5. Fortalecer la industria local,

6. Alcanzar niveles internacionales en capacidad de procesos y
7. Promover la construcción de infraestructura física y telecomunicaciones.

El origen del estándar Moprosoft (NYCE, 2011) es la necesidad de cumplir con la estrategia número 6 del Prosoft relativa a "alcanzar niveles internacionales de capacidad de procesos" por parte de las PyMEs mexicanas desarrolladoras de software. El esquema Moprosoft permite desarrollar software en las PyMEs; al desarrollar su capacidad de utilizar procesos y con esto hacerlas más competitivas, a fin de que tengan mayores probabilidades de permanecer en el mercado.

En el ámbito de las TI, NYCE contribuyó a la elaboración y posterior evaluación del estándar o norma NMX-I-059/02-NYCE-2011 "Tecnología de la información - Ingeniería de Software - Calidad de producto" (Moprosoft). La creación de este estándar no fue casual, ya que con esto se logró dar legitimidad y certeza jurídica al modelo de evaluación de madurez de la capacidad de procesos, para así elevarlo a la categoría de norma, hoy estándar Moprosoft.

Como Unidad de Verificación de Tecnologías de Información (UVTI) acreditada desde noviembre de 2005 por la Entidad Mexicana de Acreditación en los términos de la Ley Federal sobre Metrología y Normalización (LFMN), NYCE evalúa el cumplimiento de la norma NMX-I-059/02-NYCE-2011 (Moprosoft), que estandariza los niveles de madurez de la capacidad del proceso de las empresas, a las cuales se les otorga el correspondiente Dictamen.

¿Qué es la Verificación de la NMX-I-059/02-NYCE-2011 (Moprosoft)?. La verificación conforme a la norma mexicana NMX-I-059/02-NYCE-2011 consiste en determinar el nivel de madurez de los 9 procesos en las organizaciones que tienen como referencia el modelo Moprosoft.

Estos 9 procesos están contenidos en tres categorías: Alta Dirección (DIR), Gerencia (GER) y Operación (OPE), lo que asegura una cobertura total en la organización. Se determina el nivel de madurez de capacidades para cada proceso verificado y, con base en ello, el nivel de madurez de

capacidades de la organización, que es el máximo nivel de capacidad alcanzado por todos los procesos de Moprosoft. En la tabla 2.2 se muestra la estructura.

Tabla 2.2 Estructura de la NMX-I-059-NYCE-2005.
Fuente: Flores Rios et. al. (2008).

No. de parte	Descripción
01. NMX-I-059/01-NYCE-2005	Definición de conceptos y productos.
02. NMX-I-059/02-NYCE-2005	Requerimientos del Proceso (Moprosoft).
03. NMX-I-059/03-NYCE-2005	Lineamientos para los procesos de implementación.
04. NMX-I-059/04-NYCE-2005	Lineamientos para los Procesos de Evaluación (EvalProSoft).

En Flores Rios et. al. (2008) se describe que Moprosoft esta basado en los estándares de CMMI y los ISO/IEC 9001:2000 e ISO/IEC 12207. EvalProSoft se basa en el estándar ISO/IEC 15504-2. Se reporta que MoProSoft se está utilizando como referente en la industria de TI en México y su principales características son (NYCE, 2011):

- Modelo mexicano basado.
- Cubre el ámbito completo de una organización.
- Orientado a empresas dedicadas al desarrollo y mantenimiento del software.
- Sencillo y entendible.
- Fácil de implantar.
- Penetración en las PyMEs en México.
- Bajos costos de verificación en comparación con otros estándares.
- Modelo derivado del proyecto Prosoft del Gobierno Mexicano.
- Obtención de un dictamen de conformidad de los procesos verificados en términos de la LFMN.
- Se implementan y verifican 9 procesos sin importar el tamaño de la organización.

El Nivel de Madurez de Capacidades de la Organización está definido como el máximo nivel de capacidad alcanzado en una organización por todos los procesos de Moprosoft. En la figura 2.15 se muestran estos niveles.

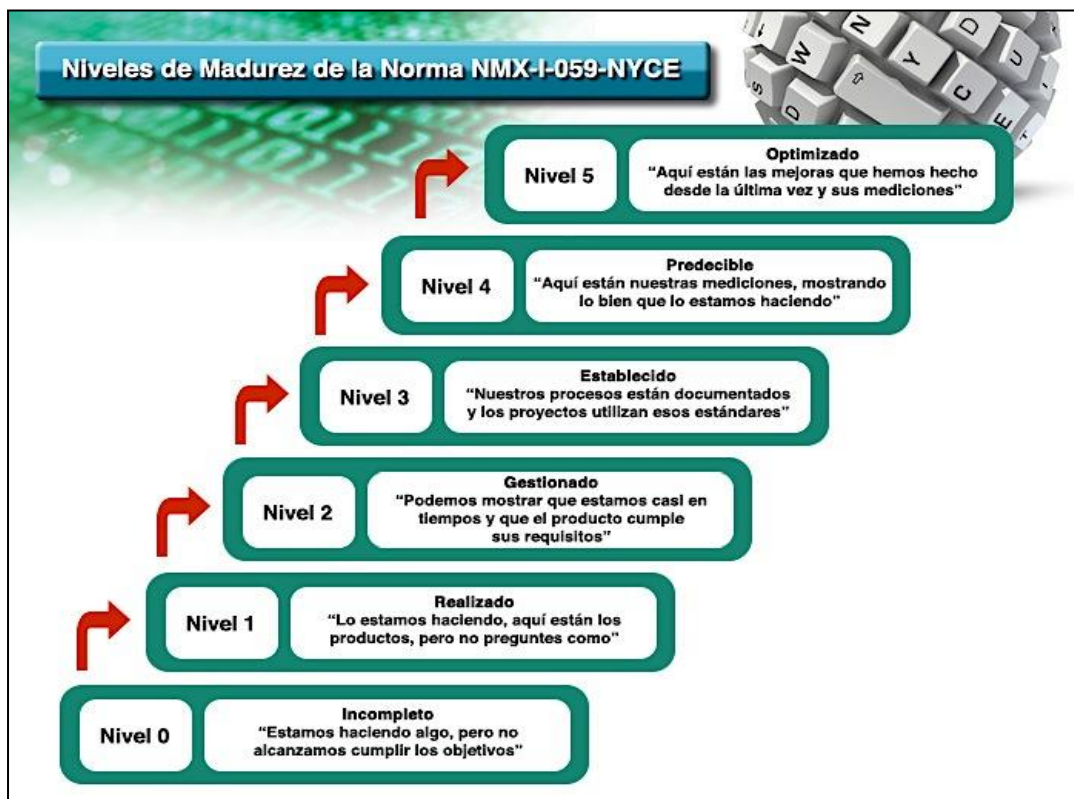


Figura 2.15. Niveles de madurez propuestos en Moprosoft.
Fuente: NYCE (2015).

A pesar de los esfuerzos de muchas PyMEs por estandarizar procesos en el desarrollo de software se ha encontrado problemas al alinear las estrategias de competitividad de las empresas con la estandarización y la búsqueda de calidad. Se desarrolló un marco de implementación como una técnica SPI (*Software Process Improvement*, mejora de procesos) llamado COMPETISOFT (Oktaba, García, Piatinni, Pino, Alquicira, & Ruiz, 2007). La mejora de procesos es una iniciativa para fomentar la competitividad de la pequeña y mediana industria del software de Iberoamérica, es financiado por el Programa Iberoamericano de Ciencia y Tecnología para el Desarrollo (CYTED, 2015). Para evaluar el nivel de madurez de una PyME se propone seguir el modelo descrito en la figura 2.16.

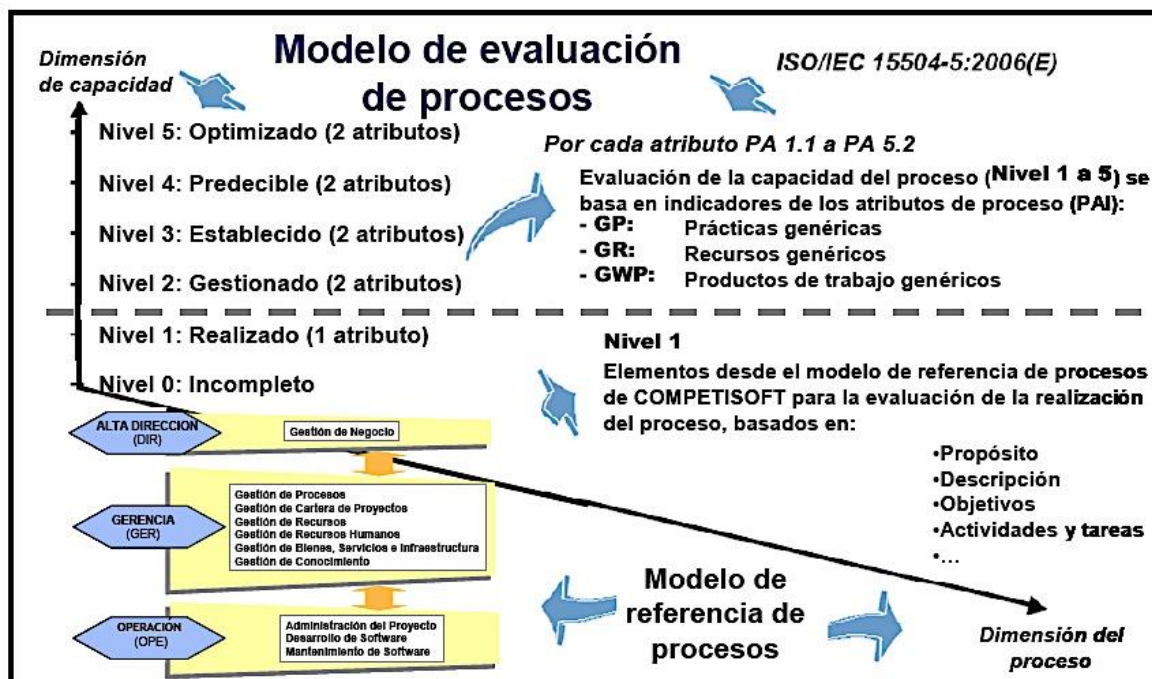


Figura 2.16. Modelo de evaluación del nivel de madurez sugerido por COMPETISOFT.
 Fuente: Oktaba H. et. al. (2007).

A pesar de los esfuerzos de estandarización en PyMEs de Moprosoft se tiene documentado varios problemas en la implementación del modelo; se describen costos muy altos y un esfuerzo muy grande de parte del equipo de desarrollo para lograr la certificación, que en empresas pequeñas menores a 10 empleados se complica la adopción del estándar. En (Lozada Carrillo, 2013) se describe el problema de certificación que muchas empresas tiene al intentar adoptar el estándar. En medianas empresas los costos y tiempos invertidos para seguir la NOM no requiere tanto esfuerzo y muchas la han logrado. El problema grave es en las empresas PyMEs, el costo, el tiempo y la documentación es el principal factor de fracaso.

Finalmente se han reportado proyectos en los que Moprosoft se utilizó como estándar para el desarrollo de software en PyMEs en aplicaciones Web (Schaffer L. & Oktaba, 2008), donde el nivel de madurez inicial es nivel 0 y se trata de mejorar el proceso de desarrollo utilizando el estándar, logrando niveles de madurez 5 en algunos casos.

2.4.4 ISO/IEC.

ISO (Organización Internacional de Normalización) es una organización no gubernamental independiente. Las Directivas ISO/IEC definen los procedimientos básicos que deben seguirse en la elaboración de normas internacionales y publicaciones. Normas que permiten que las cosas funcionen a través de especificaciones de productos, servicios y sistemas, para garantizar la calidad, seguridad y eficiencia. Son fundamentales para facilitar el comercio internacional. ISO ha publicado más de 19,500 normas internacionales cubriendo casi todas las industrias, desde la tecnología, la seguridad alimentaria, la agricultura y la salud. ISO tiene un impacto en todos los ámbitos (ISO.org, 2015). Tiene 163 países miembros, que son los organismos nacionales de normalización de todo el mundo. Los estándares más populares son:

- ISO 9000. Gestión de la calidad.
- ISO 14000. Gestión ambiental.
- ISO 3166. Los códigos de país.
- ISO 22000. Gestión de la seguridad alimentaria.
- ISO 26000. Responsabilidad social.
- ISO 50001. Gestión de la energía.
- ISO 31000. Gestión del riesgo.
- ISO 4217. Códigos de las monedas.
- ISO 639. Códigos de idioma.
- ISO 20121. Eventos sostenibles.
- ISO 27001. Seguridad de la información.

Una de las normas más utilizadas en el desarrollo del software son las normas ISO/IEC 15504: 2004, ISO/IEC 15504: 1998, ISO/IEC 12207: 2004 e ISO 9001: 2000 (Pino, García, & Piatinni, 2008). La ISO/IEC 15504, también conocida como determinación de la capacidad de la mejora de procesos de Software (SPICE, *Software Process Improvement Capability Determination*). Promueve un modelo para la mejora en el desarrollo de software, evaluación de los procesos de

desarrollo, mantenimiento de sistemas de información y productos de software. La norma se divide en varias revisiones y partes, ver tabla 2.3.

Tabla 2.3. Elementos de la norma ISO/IEC 15504 SPICE.
Fuente (ISO.org, 2015).

Norma	Descripción	Año
ISO/IEC 15504-2:2003/Cor. 1	Revisión y correcciones de la norma <i>Software Process Improvement Capability Determination</i> (SPICE).	2004
ISO/IEC 15504-1	Tecnología de la información - Evaluación de proceso - Parte 1: Conceptos y vocabulario.	2004
ISO/IEC 15504-2	Tecnología de la información - Evaluación de proceso - Parte 2: Realización y evaluación.	2004
ISO/IEC 15504-3	Tecnología de la información - Evaluación de proceso - Parte 3: Guía para la realización de una evaluación.	2004
ISO/IEC 15504-4	Tecnología de la información - Evaluación de proceso - Parte 4: Orientaciones sobre el uso para la mejora de procesos y la determinación de capacidad de proceso.	2004
ISO/IEC 15504-5	Tecnología de la información - Evaluación de proceso - Parte 5: Un modelo de evaluación del proceso del ciclo de vida del software ejemplar.	2012
ISO / IEC 15504-6	Tecnología de la información - Evaluación de proceso - Parte 6: Un modelo de evaluación del proceso del ciclo de vida del sistema ejemplar.	2013
ISO / IEC 15504-8 TS	Tecnología de la información - Evaluación de proceso - Parte 8: Un modelo de evaluación del proceso de ejemplo para la gestión de servicios de TI.	2012
ISO / IEC 15504-9 TS	Tecnología de la información - Evaluación de proceso - Parte 9: Perfiles del Proceso objetivo.	2011

La norma ISO/IEC 9001:2000 (y su modificación ISO/IEC 9001:2008 y 2009), se centran en los sistemas de gestión de la calidad y en los requerimientos (*quality management systems – requirements*). El ISO/IEC 12207:2008 estandariza los sistemas y la ingeniería de software, en su ciclo de vida (*Systems and software engineering - Software life cycle processes*). Finalmente la ISO

31000 que estandariza la administración del riesgo (*Risk management*). En la figura 2.17 siguiente se muestran los principales estándares que ayuda al desarrollo de software y su calidad.

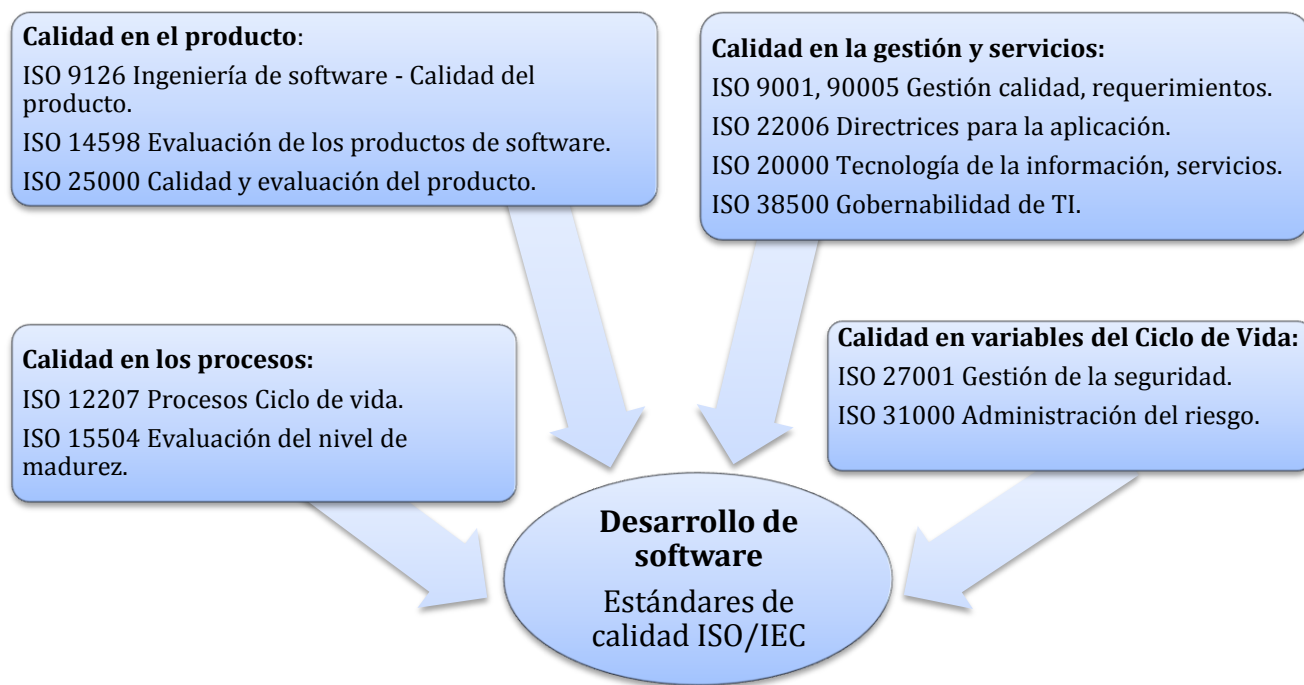


Figura 2.17. Estándares ISO/IEC que promueven la calidad en el desarrollo de software.
Fuente: elaboración propia.

En Pino et. al (2008) se menciona que las PyMEs son una pieza muy importante en los engranajes de la economía mundial. La industria del software en la mayoría de los países se compone principalmente de PyMEs que desarrollan software. De aquí la importancia de fortalecer a este tipo de organizaciones, mediante el uso de estándares que mejoren la ingeniería de software. En las últimas dos décadas, la comunidad de Ingeniería del Software ha mostrado un especial interés en la mejora de procesos de software (SPI) en un esfuerzo por aumentar la calidad del producto de software, así como la productividad del desarrollo de software. Sin embargo, hay una tendencia generalizada, el éxito del SPI es posible sólo para las grandes empresas debido al esfuerzo del grupo de calidad y a los costos al implementar estándares de calidad.

En (Laporte, Simon, Renault, & Crowder, 2008) se reconoce que hay empresas muy pequeñas que desarrollan partes o componentes de software. Estas empresas son muy

importantes para la economía mundial, y las partes que desarrollan a menudo se integran en los productos fabricados por las empresas más grandes. Se han documentado muchos fracasos de estas empresas al entregar un producto de calidad a tiempo y dentro del presupuesto proyectado, esto pone en peligro la competitividad de las organizaciones. Una forma de mitigar estos riesgos es hacer que participen todos los proveedores en la cadena de producción, que utilicen prácticas de ingeniería reconocidas y estándares.

Muchas normas internacionales han sido desarrolladas para capturar prácticas de ingeniería probadas. Sin embargo, estas normas no fueron escritas por PyMEs, aquellas que tienen menos de 25 empleados, y que son, en consecuencia difícil de aplicar en estos entornos. El ISO/IEC JTC 1/SC7 es la estandarización de los procesos en PyMEs, propone herramientas de apoyo, y el apoyo a las tecnologías de la ingeniería de sistemas y productos de software. La norma incluye:

1. Los procesos de ingeniería de software y sistemas.
2. Gestión de Servicios de TI.
3. Permite Servicios y *Outsourcing* para procesos de negocios.
4. Productos de sistemas de software.
5. Arquitectura empresarial.
6. Entorno de ingeniería de software.
7. Cuerpo de ingeniería de software de conocimientos.
8. Gestión de activos de TI.

El objetivo de esta norma es ayudar a las PyMEs a:

1. Proporcionar los estándares de calidad que cubren todo el ciclo de vida de los sistemas de información.
2. Proporcionar los estándares de calidad que satisfagan las necesidades de los usuarios en los mercados generales.
3. Administrar el conjunto de normas de manera efectiva a través de marco documentado
4. Promover el uso de normas, proporcionando materiales de apoyo.

Proporcionar liderazgo en la normalización a través de:

1. Un proceso de vigilancia tecnológica continúa utilizando grupos de estudio para explorar nuevas áreas y mercados.
2. El desarrollo de un amplio conjunto de normas integradas con amplio consenso internacional y profesional.
3. Inicio de actividades de cooperación con las organizaciones profesionales internacionales y estándares producir.

Un marco que:

1. Facilita la integración y la subcontratación de las normas elaboradas en otras normas que producen las organizaciones.
2. Facilita el desarrollo cooperativo de normas conjuntas con otras normas internacionales de producción de las organizaciones.
3. Minimiza las inconsistencias entre la norma ISO/IEC JTC 1/SC 7, incluyendo las desarrolladas por otras normas que producen las organizaciones.

En la tabla 2.4 se muestran las partes de la norma ISO/IEC JTC 1/SC 7.

Tabla 2.4 Grupos de trabajo y áreas de la norma ISO/IEC JTC 1/SC 7.
Fuente: (ISO.org, 2015).

Grupos de trabajo	Área De Trabajo
ISO / IEC JTC 1 / SC 7 / WG 2	Sistema de Documentación de software.
ISO / IEC JTC 1 / SC 7 / WG 4	Herramientas y Medio Ambiente.
ISO / IEC JTC 1 / SC 7 / WG 6	Evaluación y Métrica.
ISO / IEC JTC 1 / SC 7 / WG 7	Gestión del Ciclo de Vida.
ISO / IEC JTC 1 / SC 7 / WG 10	Evaluación de Procesos.
ISO / IEC JTC 1 / SC 7 / WG 19	Procesamiento distribuido abierto (ODP) y Lenguajes de modelado.
ISO / IEC JTC 1 / SC 7 / WG 20	Software y Sistemas Cuerpos de Conocimiento y Profesionalización.
ISO / IEC JTC 1 / SC 7 / WG 21	Gestión de Tecnología de la Información de Activos.
ISO / IEC JTC 1 / SC 7 / WG 23	Sistema de Gestión de Calidad.
ISO / IEC JTC 1 / SC 7 / WG 24	Ciclo de Vida del Software (SLC) Perfiles de entidades muy pequeñas

	(microempresas).
ISO / IEC JTC 1 / SC 7 WG 25	Gestión de Servicios de TI.
ISO / IEC JTC 1 / SC 7 / WG 26	Pruebas de Software.
ISO / IEC JTC 1 / SC 7 / WG 27	IT Servicios Habilitados .
ISO / IEC JTC 1 / SC 7 / WG 28	Formatos comunes de la industria para los informes de usabilidad (conjuntas entre ISO/IEC JTC 1/SC 7 e ISO/TC 159/SC 4).
ISO / IEC JTC 1 / SC 7 / WG 42	Arquitectura.

2.4.5 MDSIC.

En Cendejas et. al (2014) se propone un modelo que permite desarrollar software en PyMEs con calidad basándose en la alineación de las TI con los propósitos de la organización además de ser una modelo ágil y adelgazado. En la figura 2.18 se muestra la metodología aplicada para el desarrollo de este modelo basado en la experiencia de los desarrolladores de la región centro-occidente de México.

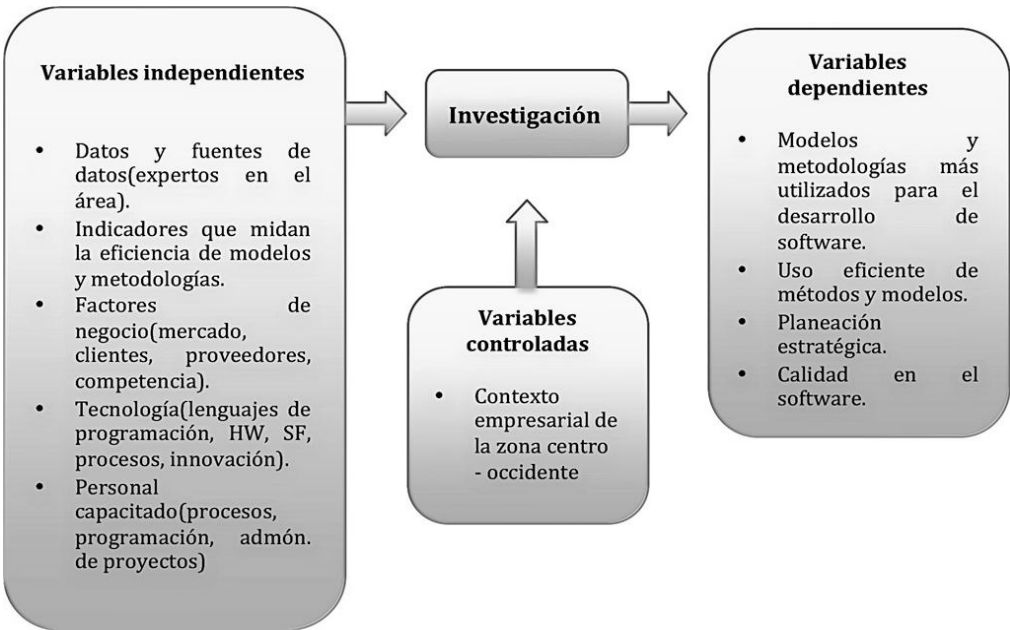


Figura 2.18. Metodología de diseño del MDSIC.
Fuente: (Cendejas Valdez, 2013).

Para censar el contexto de desarrollo de software en la región se elaboró una encuesta para censar las variables de esta industria. Con los resultados del análisis de esta y el objetivo de diseñar un modelo para la construcción de software ad-hoc, se propone el MDSIC que acompaña a los desarrolladores expertos a través de cinco diferentes niveles que brindan las mejores prácticas para el desarrollo de software, además de ser respaldados por los procesos que propone el *Project Management Institute* (PMI, 2014), lo que permite generar software de calidad alineado con la estrategia de la organización. Los niveles del modelo son los siguientes:

Nivel 0: Detección del Problema.

Nivel 1: Análisis y Diseño.

Nivel 2: Desarrollo.

Nivel 3: Implementación.

Nivel 4: Indicadores de calidad.

También considera los primeros cinco niveles de los grupos de procesos que contempla el (PMI, 2014): 1) Integración de la administración del proyecto, 2) Alcance, 3) Tiempo, 4) Costo y 5) Calidad. Los cuales permiten producir un software de calidad. A continuación se describe el proceso general de los pasos que se deben de seguir para utilizar el modelo en el desarrollo de software integral colaborativo, además se indica los elementos de calidad que se evalúan en cada nivel. Se muestran en la tabla 2.5.

Tabla 2.5. Funcionamiento de los niveles del MDSIC.
Fuente: elaboración propia.

Nivel	Nombre	Descripción	Elementos de calidad
0	Identificación del problema.	Consiste en reuniones (de 3 a 10) en donde se busca conocer las necesidades específicas del proyecto y el alcance que se pretende. Se genera un análisis de viabilidad y fiabilidad para que el cliente conozca las desventajas y ventajas de la generación del software.	Requerimientos. Viabilidad y presupuesto.

		Estudiar la factibilidad de disminuir tiempos, costos y generar beneficios en el producto. El equipo se conforma por el cliente y los <i>stakeholders</i> (administrador del proyecto) de cada una de las áreas que dominan los procesos del modelo, el objetivo es identificar las necesidades del proyecto. Se genera la “memoria de actividad” la cual almacenara todos los datos de la información del proyecto a desarrollar.	Costos y tiempo de desarrollo. Integración de equipos e identificación de necesidades. Evidencia del proyecto.
1	Análisis y diseño.	Se unen las etapas de análisis y diseño (comúnmente definidos en otras modelos) como un enfoque de desarrollo ágil y adelgazado. Se utiliza los principios de la planeación estratégica para alinear las TI con los objetivos de la organización; se analizan los requerimientos y se planea en un árbol de actividades (Work Breakdown Structure, WBS), las tareas y los entregables del proyecto. Se desarrolla el diseño de las GUI y se selecciona el IDE para el desarrollo del producto. Se miden y comparan los resultados generados mediante un “plan de pruebas”, para revisar avances y resultados generados del equipo de desarrollo. Se desarrolla un Prototipo (Beta testing), para que el cliente lo revise y lo apruebe, debe contener los requerimientos y especificaciones del nivel 0. Se revisa la satisfacción del cliente y se definen las fortalezas, oportunidades, debilidades y amenazas que puede tener el sistema planeado.	Adelgazamiento de procesos. Alineación de la TI con la organización y el WBS. Generación del diseño. Plan de pruebas y revisión de avances en el diseño. Generación de prototipos. Alineación de los requerimientos con el prototipo.
2	Desarrollo.	Desarrollo del sistema a través de módulos o por partes. Evaluación de los módulos desarrollados y realizar pruebas de funcionamiento.	Elementos de desarrollo Verificar módulos

		Aprobación de los módulos para su implementación.	Módulos validados
3	Implementación.	Se inicia la implementación del sistema. Se realizan pruebas del funcionamiento del sistema. Pre-entrega del sistema al usuario para validación. Se corrigen los errores reportados. Entrega de la documentación y se inician los procesos post-desarrollo.	Entorno. Pruebas. Pre-entrega. Corrección. Documentación e inicio de cierre.
4	Indicadores de calidad.	Se revisa cada uno de los indicadores por el equipo de calidad en donde se censan los indicadores por grupos: A) Corrección, fiabilidad, eficiencia, integridad y facilidad de uso. B) Facilidad de mantenimiento, flexibilidad y facilidad de prueba. C) Portabilidad, reusabilidad e interoperabilidad. Finalmente se realiza el proceso de cierre.	Revisión de KPIs Entrega del proyecto y lecciones aprendidas.

El objetivo principal del MDSIC es diseñar e implementar un modelo que sirva como base para desarrollar software de calidad que contribuya a la realización de proyectos de corte tecnológico en empresas de México. Las conclusiones generales de los objetivos específicos del modelo son las siguientes:

- Realizar un análisis de los modelos y metodologías actuales para el desarrollo de software, que permita determinar las principales etapas de nuestro modelo.
- Analizar cuáles empresas del sector de la Industria del software son las que presentan mejores resultados, teniendo en consideración la influencia de la innovación tecnológica, el uso de alguna metodología y la orientación de mercado sobre los resultados.
- Determinar las fases mínimas para la planeación, ejecución y cierre del desarrollo de software.
- Desarrollar las fases del modelo que permita fortalecer los puntos críticos de los modelos y metodologías que se encuentran en el mercado, bajo un enfoque orientado a procesos

El MDSIC se asocia con un software que permite la interacción entre los miembros del proyecto quienes obtienen un rol al pertenecer a este. La herramienta permite la integración, colaboración y el control de la calidad de un proyecto ver figura 2.19.

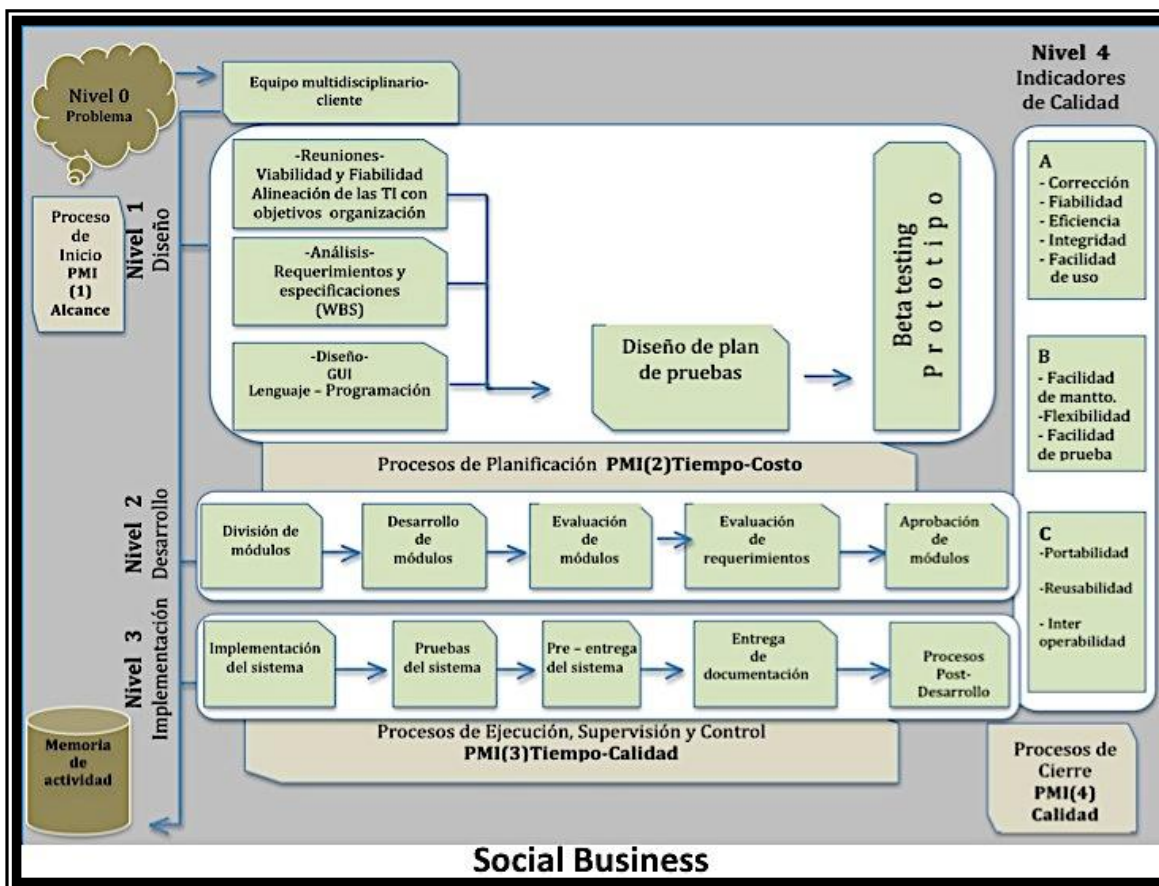


Figura 2.19. Modelo de desarrollo de software integral colaborativo.
Fuente: Cendejas V. et. al. (2014).

MDSIC está en pleno desarrollo y validación, todavía faltan muchas pruebas sobre el modelo, sin embargo presenta algunas deficiencias que es necesario revisar para mejorar el desarrollo:

- No se tiene un control sobre las tareas y procesos del proyecto, se tiene un WBS como base para definir las tareas pero no se le da seguimiento.
- No se definen entregables en la metodología por lo que el trabajo en un Sprint no está bien definido, esto es una parte muy importante para el desarrollo ágil.

- No se le da seguimiento al cumplimiento de una tarea en tiempo y forma, por lo que el costo del proyecto tiende a elevarse si los promotores del proyecto no están atentos al mismo.
- Carece de una metodología para definir los elementos del proyecto, desde procesos, tareas, seguimiento de tareas, costos, métricas y control del Scope del proyecto.
- No se contemplan los costos de los riesgos al no desarrollarse una tarea en tiempo.
- Debe agregarse un elemento en la metodología que defina el contexto del producto y sus características para aplicarle las mejores prácticas y desarrollar el proyecto en un tiempo justo y basado en los costos.

2.2. Síntesis de las ventajas y desventajas de metodología analizadas

En la tabla 2.6 se sintetizan las características más importantes de cada metodología como comparativa.

Tabla 2.6 Ventajas y desventajas de las metodologías para desarrollo móvil actuales.
Fuente: elaboración propia.

Metodología	Ventajas	Desventajas
DIMP (De Silva et. al., 2013): <i>Distributed Integrated Development Environment for Mobile Platforms.</i>	Desarrollar y ejecutar códigos fuentes desde el origen y de compilar en un entorno distribuido. Ofrece una excelente alternativa para el desarrollo rápido de aplicaciones.	No ofrece una metodología que permita agregar mejoras en el desarrollo. No tiene un control sobre el proceso de desarrollo y censado de la calidad.
MASAM (Yang-Jae et. al., 2008): <i>Mobile Application Software based on Agile Methodology.</i>	Esta metodología se centra en el equipo de desarrollo y en la etapa de desarrollo del software (RUP). Se orienta a una elección rápida del tipo de producto que se desea, se construye y se lanza al mercado lo más rápido posible. Orientada a empresas pequeñas que se ven limitadas en personal y en infraestructura.	No se contempla el control de la calidad en la etapa de diseño. No hay herramientas para un seguimiento basado en un plan. No se reporta alguna herramienta integral colaborativa.

	Se basa en tres fases: 1) rol de los participantes, 2) tareas y procesos, y 3) planeación y entrega del producto. Propone 4 etapas por cada fase: 1) preparación, 2) materialización, 3) desarrollo y 4) comercialización.	
Mobile-D (Abrahamsson, et al., 2004): <i>Mobile application Development</i>.	Se basa en el desarrollo XP y Crystal (Cockburn, 2004). Ofrece un desarrollo ágil e incremental, fases que considera son: 1) Exploración, 2) Inicio de Interacciones, 3) Producción, 4) Estabilización y 5) Pruebas del sistema y correcciones. Se ha aplicado en varias empresas y proyectos con éxito.	El seguimiento por tarea y procesos no está bien definido lo cual puede incurrir en costos a la hora de la ejecución del proyecto (Spataru, 2010). El control de riesgos no es considerado. No se acompaña de un software para el control.
Scrum (Scharff & Verma, 2010), (Abrahamsson, et al., 2004): Marco de trabajo para la gestión y desarrollo de software y entornos basados en el desarrollo ágil de software.	Se evaluó el uso de Scrum como una metodología para tener éxito en el desarrollo de aplicaciones móviles. Se ofrece un marco iterativo e incremental, se hacen iteraciones fijas en tiempo conocidas como <i>sprints</i>. Se definen roles para el desarrollo de los <i>sprints</i>: 1) propietario, 2) equipo de Scrum 3) Scrum Master; encargado de guiar al equipo a través del proceso de desarrollo y a resolver problemas.	La consolidación del equipo de trabajo y el liderazgo la debe ejercer el Scrum master (con un alto costo si este falla) Se requiere de un experto que revise el cumplimiento de la metodología, lo cual conlleva a una capacitación previa. Las empresas deben adaptarse a la metodología y no viceversa.
SleSS (Cunha, Dantas, & Andrade, 2011): Método basado en Scrum y Lean Six Sigma.	Es una metodología ágil y adelgazada. Se ha probado en el desarrollo de proyectos. Se establecen elementos de calidad, y utiliza el enfoque de un desarrollo progresivo.	Está en proceso de implementación. No se menciona si existe algún software de seguimiento.
MDSIC (Cendejas V., et. al., 2014): Modelo de Desarrollo de software Integral Colaborativo.	Desarrollo de productos de calidad que satisfacen las necesidades y los objetivos de las PyMEs. Se considera al equipo de desarrollo como colaborativo que ayuda en diferentes	No se tiene un control sobre las tareas y procesos del proyecto. No se definen los <i>sprints</i> del proyecto en forma clara.

	departamentos. Define 5 diferentes niveles que brindan las mejores prácticas para el desarrollo. Se utilizan las mejores prácticas de planeación y seguimiento de tareas (PMI, 2014).	No se le da seguimiento al cumplimiento de una tarea en tiempo y forma. Carece de una metodología para definir los elementos como; procesos, seguimiento de tareas, costos, métricas y el control del tiempo.
--	---	--

2.5 Metodología ágil y adelgazada como medio de competitividad.

Se ha observado que en muchas organizaciones, están todavía en etapa de consolidación y el enfoque SPM puede complicarse (se generan muchos pasos entre procesos), de aquí la necesidad de los principios de *Lean*. La Filosofía *Lean Management* se centra en el aumento de valor mediante la eliminación de los residuos

2.5.1 Filosofía Lean en el SPM.

Muchas empresas han adaptado estos principios de desarrollo de software, ya que el SPM no se limita al desarrollo solamente sino que también abarca disciplinas como la gestión de productos, desarrollo de negocios, marketing, ventas y soporte. Los cinco principios de la filosofía adelgazada se muestran en la tabla 2.7.

Tabla 2.7 Principios de la filosofía adelgazada (*Lean*).
Fuente: elaboración propia.

No.	Principio	Descripción
1	Valor	El objetivo principal de cada producto o servicio es ofrecer un valor adicional a un cliente. Identificar el valor del producto para un cliente específico se acopla perfectamente con la visión y la estrategia de desarrollo de productos. Es el primer paso para entender por qué existe una empresa en el mercado. La falta de comprensión de su propio negocio y el valor que ofrece a los clientes de una empresa, lleva a que se centra en cuestiones de corto plazo y las acciones de

		reducción de costos menores. A medida que la empresa crece, estas acciones tomarán más y más tiempo sin aportar un valor adicional para los clientes.
2	Cadena de valor	Se establece un mapa de flujo de valor que incluye todos los pasos, desde el concepto hasta la entrega. El mapeo de la cadena de valor tiene como objetivo identificar y eliminar los pasos que no crean valor. Permite a la empresa eliminar pasos innecesarios y reducir el tiempo de ciclo de entrega. Con el tiempo, la eficiencia de la empresa crece, y se les puede dar más valor a los clientes al utilizar menos recursos.
3	El flujo	Después de eliminar los pasos que no crean valor, la empresa debe estandarizar (suavizar) el proceso de creación de valor. Para lograr esto, <i>Lean</i> sugiere centrarse en todo el proceso de creación de valor en lugar de trabajar en departamentos aislados. La reorganización de la producción de un departamento el uso de un procesamiento (o proceso) por lotes para un flujo continuo aumenta la productividad y reduce los errores y la cantidad de recursos necesarios. El principal problema es que el "pensamiento de flujo es contrario a la intuición", las empresas están acostumbradas a trabajar en departamentos con la separación de funciones. Esto conduce a una situación en la que cada departamento funciona tan eficazmente como sea posible en sus propios procesos, pero toda la eficiencia de la producción tiene enlaces menos eficientes en la cadena, que se manifiesta con pasos intermedios entre los departamentos.
4	Jalar	Jalar significa que el cliente envía una petición de un producto y la empresa ofrece el producto que satisfagan las necesidades del cliente tan pronto como sea posible. Adoptando el "procedimiento de flujo" reduce el tiempo desde la concepción hasta la entrega. Además, debido al tirón del cliente, la empresa se evita la producción y almacenamiento de productos no deseados. Como resultado los productos de la empresa satisfacen las necesidades y deseos del cliente.
5	Perfeccionar	El último paso es analizar los resultados y el plan para cualquier implementación. Los otros cuatro principios se centran en lo que debe hacer, este se centra en cómo hacer realidad las acciones. La filosofía adelgazada sugiere concentrarse en un conjunto de maneras de pensar para aumentar la eficiencia y la rentabilidad poco a poco, en forma progresiva con un enfoque a largo plazo.

En Maglyas et. al. (2012) observaron el comportamiento en el desarrollo de software en varias empresas, quienes tuvieron algunos problemas al aplicar la filosofía *Lean* en conjunto con el SPM, en la tabla 2.8 se sintetizan.

Tabla 2.8 Problemas al aplicar la filosofía Lean en el SPM.
Fuente: (Maglyas, Nikula, & Smolander, 2012).

No.	Problema	Solución posible
1	Ciclo de lanzamiento del producto muy largo. Los administradores tuvieron dificultades al tratar de definir el ciclo de vida del producto, desde el concepto hasta la entrega. Las empresas fueron organizadas como departamentos separados, contradiciendo el principio del flujo.	Uso del flujo para reducir el tiempo de comercialización. El uso del <i>Lean</i> puede hacer el proceso de desarrollo más predecible en términos de características implementadas en la final de la iteración, las características de implementación deben ser conocidas al comienzo de la iteración, y la adición de nuevas características durante la iteración no está permitido. Esto ayuda al departamento de marketing ya que inicia su trabajo sin necesidad de esperar hasta el final de la iteración. Entonces se reduce el tiempo de comercialización. Esto ayuda a un flujo suave para cada producto en el que todas las unidades trabajan en conjunto. El principal problema de dicha adopción es la necesidad de sincronización entre todas las unidades; el gerente de producto puede resolver esto actuando como el especialista del plan.
2	No hay métricas para evaluar el trabajo. En las organizaciones observadas, el gerente responsable del producto no tenía indicadores clave de rendimiento (KPIs). Algunos utilizaron una práctica relacionada; la dirección por objetivos, pero el procedimiento de identificación de estos era muy especulativa.	Utilizar el valor para identificar los KPI. <i>Lean</i> aboga por el uso de métricas para evaluar y mejorar las actividades y toda la cadena de valor. Los KPI del gerente de producto deben ser representados en una forma numérica y deben mostrar el rendimiento. Los KPI para la gestión de un producto deben estar relacionados con sus características. Por lo tanto, los pasos en la identificación de los KPI deben incluir la comprensión de los objetivos de la empresa y del objetivo del producto, actuando como un todo, para tener éxito.
3	La colaboración entre las organizaciones y los clientes.	Utilizar el principio de jalar para desarrollar productos más rápido y con menos recursos apegados a las

	Se observó que algunas de las organizaciones están muy orientadas al cliente, pero no todas. Incluso se observó especulaciones acerca de las necesidades de los clientes y deseos cuando no había colaboración con ellos.	necesidades del cliente. El cliente es que inicia el proceso de la necesidad de un producto y solo con esta se puede tener éxito y no al revés como muchas empresas comúnmente hacen.
4	Pensamiento a corto plazo. Casi todas las organizaciones estudiadas se concentraron en las acciones a corto plazo, tales como la implementación de nuevos procesos de desarrollo basados en la moda, modernas formas de desarrollo, todos sin un profundo análisis de sus resultados. El desarrollo de muchos productos en paralelo sin entender el negocio principal de la compañía. Incluso los altos directivos no estaban listos para hablar de sus planes en el horizonte de más de un año.	Utilizando el principio de perfección para la adopción de pensamiento a largo plazo. Realizar acciones de corto plazo para obtener ganancias rápidas a largo plazo es ampliamente discutido en la literatura <i>Lean</i> . En la planeación estratégica se ha discutido las ventajas de estrategias a largo plazo. Acciones a corto plazo a menudo se producen como reacción a las condiciones externas, influencia de los competidores. Cuando una empresa sigue sus competidores de cerca, le resulta difícil lograr su propia excelencia en cualquier área. Una estrategia a largo plazo permite una empresa entregar un valor único para el cliente, reducir sus propios costos y aumentar la eficiencia. La empresa podría actuar de forma independiente de acuerdo con sus objetivos internos, en lugar de simplemente reaccionar a sus competidores.
5	Tratar de cambiar de forma instantánea. El SPM es una disciplina compleja que consta de muchos problemas y actividades. Se observó que este comportamiento del cambio instantáneo es una característica de las PyMEs y en organizaciones grandes y extra-grandes y grandes no, estás ya saben por experiencias anteriores que los cambios radicales son riesgosos.	Utilizar el principio de perfeccionamiento para cambios incrementales. Se sugiere cambios pequeños y simples e incrementales que permiten a la organización identificar los problemas existentes y cortar futuros problemas de raíz. Las grandes organizaciones pueden tener los recursos necesarios para hacer esto, pero podrían carecer de la capacidad de reaccionar de manera efectiva a los cambios externos. Por otra parte, para las grandes organizaciones, los cambios en los modelos de negocio, valores y sobre todo en la cultura es difícil o incluso imposible.

En el mercado de software competitivo, se ha vuelto importante la entrega de productos a tiempo (*just in time*), reduciendo las actividades a lo más mínimo desde de la petición del cliente hasta la entrega. SPM sugiere prácticas, métodos y herramientas para el logro de este objetivo, pero su adopción incluye desafíos que son comunes para la mayoría de las organizaciones.

El uso de prácticas *Lean* para la SPM permite a las empresas concentrarse en lo más importante y fácil de implementar, el uso de prácticas de gestión de productos con mejoras incrementales constantes. La gestión de productos y *Lean* tienen características similares, como el papel central de valor y la atención a las necesidades del cliente. Por lo tanto, la unificación de estos dos enfoques podría beneficiar a la industria del software. Si una empresa ya practica el desarrollo de software con enfoque *Lean*, es recomendable que amplíe este enfoque en el contexto de la SPM, incluyendo actividades como la estrategia, el marketing, las ventas y las actividades de apoyo. Alcanzando con esto la excelencia en el mercado y competir.

2.5.2 Desarrollo ágil (*Agile*).

En (Agile, 2012) se describe cuales han sido los grandes cambios en desarrollo de software a partir de la introducción de los conceptos o manifiesto del desarrollo ágil, que enfatiza en mejorar el desarrollo de software primero valorando (agile manifesto, 2001):

- Individuos e interacciones sobre procesos y herramientas.
- Software funcionando sobre documentación extensiva.
- Colaboración con el cliente sobre negociación contractual.
- Respuesta ante el cambio sobre seguir un plan.

Y segundo siguiendo los doce principios:

1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.

3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
7. El software funcionando es la medida principal de progreso.
8. Los procesos ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Al igual que con cualquier disciplina naciente, los primeros años de desarrollo ágil estuvieron marcados por la exuberancia de unos pocos y por el escepticismo entre muchos. Surgieron una gran cantidad de métodos, y se desarrollaron XP, Scrum, desarrollo de software *Lean*, metodologías impulsadas por el equipo de desarrollo (FDD), Crystal, entre otros.

En términos generales, todos estos métodos se esforzaron para hacer frente a los principios básicos del manifiesto y ocurrieron cuatro grandes cambios. En primer lugar, hubo un énfasis en el desarrollo colaborativo. En segundo lugar, la mentalidad dominante "adelgazada" se defendió con el fin de ir minimizando el trabajo innecesario, particularmente con respecto a documentación exhaustiva. Esto fue malinterpretado por muchos como "sin documentación". En tercer lugar, los clientes y las partes interesadas participan ahora en forma activa en el desarrollo y ayudan a obtener el producto final. En cuarto lugar, se aceptó el hecho de que la incertidumbre era una

parte integral del desarrollo de software, y que la tendencia inherente a controlar variaciones por medios estadísticos y otros era inútil, Agile (2012).

En la figura 2.20 se muestra como el desarrollo ágil es un tema muy investigado y tiene un gran auge desde el año 2001 a la fecha.

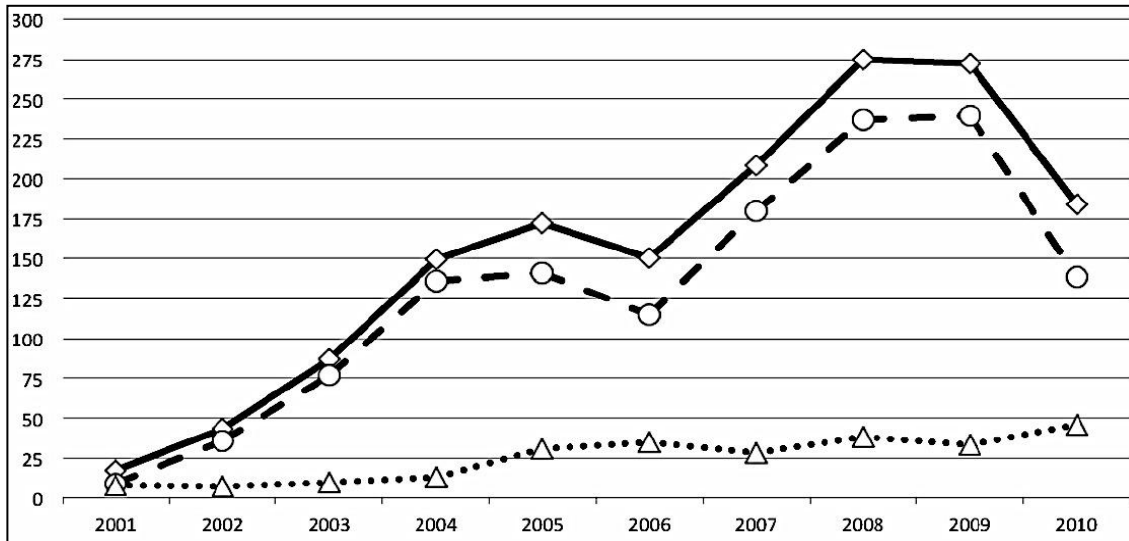


Figura 2.20. Número de publicaciones del desarrollo ágil desde 2001 al 2011. Número máximo◆, artículo en conferencia ○ y artículo en revista Δ.
Fuente: Agile (2012).

En la figura 2.21 se muestran las principales revistas científicas que difundieron esta disciplina.

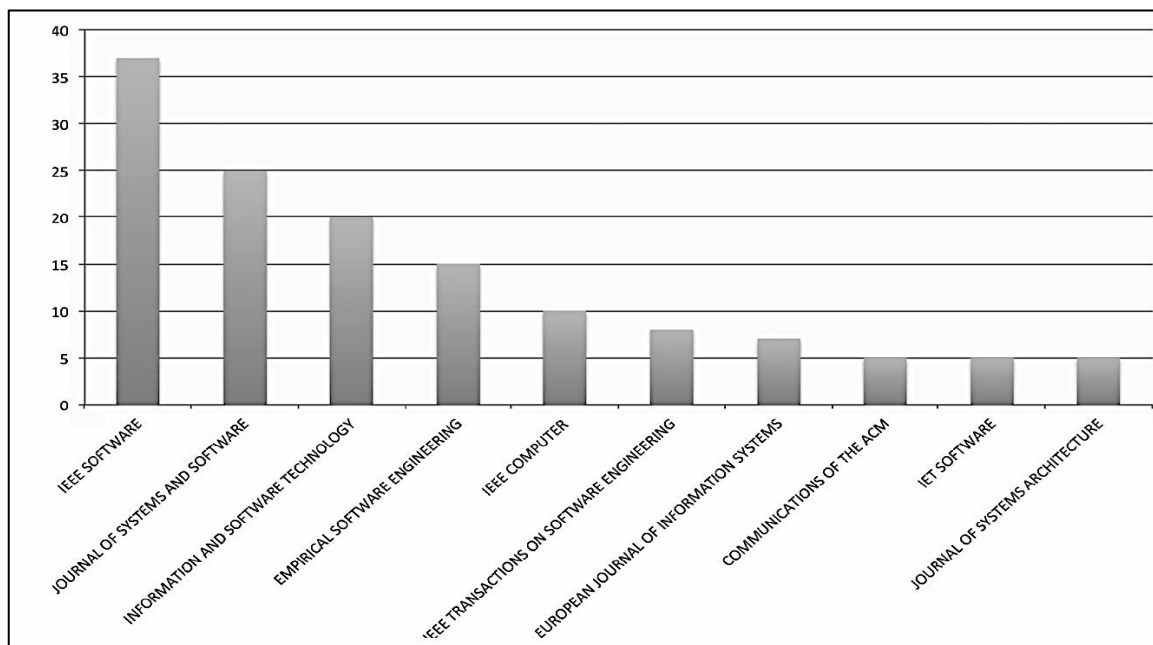


Figura 2.21. Principales revistas científicas que trataron el tema de desarrollo ágil.
Fuente: Agile (2012).

En una revisión sistemática de los estudios empíricos publicados hasta 2008 (Dyba° & Dingsøyr, 2008) observaron un aumento en el número y la calidad de los estudios. La revisión también encontró que la mayoría de los estudios se centran en XP y muy pocos en el Scrum y su proceso de desarrollo, que va ganando una fuerza significativa en la industria. Además, la revisión mostró la necesidad urgente de más estudios con equipos de desarrollo ágiles que ya han madurado en su proceso. Se conoce que falta mucho que hacer y sobre todo la falta de un marco unificado de desarrollo.

Una tendencia clara del desarrollo es la que está basada en flujos, así como en el desarrollo de software *Lean*, que ha sido popular entre los profesionales de la industria, pero no tiene todavía una amplia investigación. Dyba° et. al (2008) propusieron una ruta para la investigación en el desarrollo de software ágil, centrándose en los proveedores de ingeniería más investigación empírica, principalmente en equipos ágiles con experiencia. Se reconoce que se requiere más atención en los enfoques de la SPM. Se debe también atacar el desarrollo ágil en sistemas distribuidos que aportan ubicuidad. Las nuevas áreas de investigación que figuran son:

- Desarrollo ágil aplicado al contexto del desarrollo de software de código abierto.

- El software como servicio (SaaS).
- Estudio de los factores que afectan la adaptación organizativa de métodos ágiles que incluyen la adaptación a proyectos específicos.
- Estudio de las diferentes formas de desarrollo y los factores que facilitan la flexibilidad distribuida, la eficiencia y la eficacia del desarrollo.
- Nuevas formas de extender las prácticas del desarrollo ágil más allá de los equipos de software en el ámbito de la organización
- Identificar los límites del desarrollo ágil mediante la aplicación de agilidad para proyectos tradicionalmente considerados como no ágil.

Se reconoce que todavía falta mucho por hacer en las metodologías ágiles para que se conviertan en un estándar sin embargo en estos últimos años se está trabajando en la estandarización de procesos, en el desarrollo con calidad y en la planeación estratégica como factores de cambio en fábricas de software.

2.5.3 Aplicación del *Lean* y *Agile* en el desarrollo.

En el artículo de (Xiaofeng, Kieran, & Oisin, 2012) se hace una análisis de los métodos *Leagile (Lean y agile)* donde se estudia el uso de los principios *Lean* aplicados al desarrollo ágil de software. Se sintetizó las propuestas de varios autores que han utilizado estos conceptos así como a las empresas que ha adoptados ambos conceptos como ventaja competitiva. En la tabla 2.9 se sintetizan esta prácticas *Lean* en diferentes empresas.

Tabla 2.9. Principales aplicaciones del *Lean* al desarrollo de Software en las empresas.
Fuente: Xiaofeng et. al (2012).

Prácticas	Descripción
Combinación no intencionada de agilidad y eficiencia en los procesos de desarrollo de software.	Empresas pequeñas y medianas que están en proceso de aprendizaje del <i>Lean</i> y la utilización de un método ágil.
Aplicación intencional de los principios <i>Lean</i> en el	Empresas medianas y grandes que han adoptado en

desarrollo de software ágil. El enfoque <i>Lean</i> se aplican a las áreas de negocio relacionadas con el desarrollo de software. Procesos ágiles, <i>Lean</i> externo-alcance: el uso de los principio <i>Lean</i> permite interactuar con la unidad de negocios, manteniendo los procesos de desarrollo ágiles internamente.	sus procesos en adelgazamiento sin dejar de lado el desarrollo ágil. Son empresas que son competitivas en el mercado.
Los principios <i>Lean</i> se aplican a los procesos de desarrollo de software directamente. <i>Lean</i> es facilitador de la adopción ágil: antes o durante el proceso de transición.	Empresas que ya inician un proceso de madurez y que sus desarrollos son ágiles y con procesos eficientes. La calidad empieza a ser un factor diferenciador.
<i>Lean</i>, dentro del desarrollo ágil: el uso de diversos principios <i>Lean</i> para mejorar los procesos ágiles.	Empresas que ya tienen un nivel de madurez y mejoran sus procesos de desarrollo cada vez más.
De ágiles a <i>Lean</i>: aplicación amplia de enfoques <i>Lean</i> para transformar los procesos ágiles.	Empresas que utilizan métodos ágiles para competir y e inicia su transformación de procesos mediante el enfoque <i>Lean</i>.
Sincronización de procesos ágiles y delgados: el equipo de desarrollo ágil y el equipo <i>Lean</i> trabajan en paralelo de manera sincronizada.	Empresas con una amplia experiencia en desarrollo y cuyos procesos ya han madurado hasta tener una alta calidad en el desarrollo de software.

Es una tendencia actual que las metodologías ágiles como Scrum y XP empiecen a aplicar principios *Lean* en el desarrollo de software como una manera de competir en la industria del desarrollo de software. Los enfoques *Lean* se están aplicado en el desarrollo de software ágil. Se tienen experiencias de aplicación del *Lean* en desarrollos reales y como han logrado mejorar el proceso. Aún se está investigando como mejorar el proceso de adopción *Lean* en combinación de una metodología ágil, ya que no siempre resulta del todo buena, hay problemas en la calidad. Un área importante de mejora es el establecer un marco operacional para ayudar a los desarrolladores ha:

- 1) asignar los diversos potenciales de los "procesos *leagile*" a su propio contexto e
- 2) implementar la combinación de ambos en un proceso seleccionado.

Un reto para las empresas es cómo adoptar eficazmente estas estrategias para adaptarse la situación y las necesidades específicas de la organización sin dejar de lado sus objetivos. Se

observó un problema en varias empresas, los conceptos "ágil" y "ajustada" son términos mal definidos y muy amplios y a menudo, sus interpretaciones son vagas y pueden significar diferentes cosas en diferentes entornos. Finalmente se observó una potencial vía de investigación futura, realizar estudios de casos que permitan diseñar nuevas formas de aplicar ambas filosofías, para lograr que empresas puedan competir en esta industria, Xiaofeng et. al (2012).

Capítulo 3. Clúster de Tecnologías de Información (CTI) y Fábricas de software (FS).

3.1 Fábricas de Software.

Una Fábrica de software (SF, por sus siglas en inglés) se asemeja a una fábrica industrial (que usa componentes estandarizados, conjuntos de habilidades especializadas, procesos paralelos y una consistencia de calidad predecible y escalable). Tiene como finalidad el ensamblaje de componentes para crear aplicaciones nuevas de acuerdo a la demanda del mercado, (Piattini & Garzas, 2010). El concepto de FS responde a tres necesidades:

1. La escasez de mano de obra calificada (avances en el desarrollo de aplicaciones y nuevas tecnologías que impactan en la industria del software).
2. Los fracasos de la industria del software, debido a la madurez de la industria.
3. La evolución incesante de la tecnología que obliga a las empresas a cambiar constantemente.

Sus orígenes fué en la década de los años 70, donde las empresas se caracterizaban por ser consorcios desarrolladoras de software y que se apegaban a un ciclo de vida en el desarrollo de software. En las décadas de los años 80 y 90 (debido a la crisis del software) la FS se orientaron hacia la madurez de procesos basados en modelos como CMM y CMMI. También en la década de los años 90 se desarrollan FS basadas en la reutilización de software, cuya tecnología estaba basada en código y bibliotecas reutilizables, esto permitió el desarrollo utilizando procesos. El tamaño de las FS es considerablemente menor en empleados (pequeña y mediana empresa). Para los años 2000 las FS empiezan a construir software basado en técnicas de gestión de calidad, en modelos y metodologías que promueven líneas de productos, convirtiéndose en empresas que desarrollan aplicaciones ad-hoc a una necesidad específica. En el año 2003 con el avance de la tecnología en los dispositivos nace el mercado de las APPs. Para finales del 2010 esta industria

crece de tal forma que se vuelve una prioridad para las pequeñas y medianas empresas a nivel mundial. En México la PyMEs son la que impulsan esta industria.

3.2 Clúster de TI.

Los CTI son agrupamientos de empresas e instituciones afines que poseen capacidades similares y complementarias. Por lo común sus socios están concentrados geográficamente y compiten dentro de un mismo sector de la industria (Matus R., Ramírez A., & Buenrostro M., 2013). Los clúster han surgido como un elemento activo y relevante para el desarrollo local y como un espacio para la innovación, promoviendo con esto una ventaja competitiva en un sector de la industria. Esta iniciativa es promovida desde organismos internacionales como la Organización para la Cooperación y el Desarrollo Económico (OCDE) (OCDE, 2009) y después adoptado por México y sus estados. Este tipo de programas ha permitido insertar a las regiones dentro de cadenas de valor internacionales y aumentar la competitividad local. En los últimos años el impulso a la formación de clústeres se ha convertido en una de las principales estrategias de los países para ingresar a la economía del conocimiento como parte de una política global para impulsar el desarrollo regional. Los elementos de competitividad comunes son:

- Empresas relacionadas en un territorio con cierta homogeneidad de elementos económicos.
- Coordinación institucional, de cooperación y de intercambio entre empresas; asociación de mercados y relaciones, flujos de conocimiento, emprendimiento (también conocido como emprendedurismo), entre otros.
- Formación de vínculos de cooperación con agentes externos como estrategia de innovación.
- Factores intangibles de producción que consolida al grupo de empresas (*know how*).
- Vinculación con instituciones de educación superior como fuente de innovación.
- Agente central que gestiona redes, promoción, consolidación y circulación del conocimiento en la región.

Es importante mencionar que la construcción de un clúster es iniciada por asociaciones industriales que representan y coordinan a empresas del sector privado. Estas potencializan la transmisión del conocimiento entre sus miembros, también funcionan como diseminadoras (brokers) de información y generadoras de un lenguaje de cooperación y estándares entre las empresas que la conforman.

Un clúster surge de las alianzas multi-actorales (diferentes modelos de asociación en relación con diversos procesos, velocidades y grados de desarrollo), esto debido a que las asociaciones que conforman el clúster se enmarcan en contextos históricos, culturales, sociales, políticos y económicos muy particulares (Gandlgruber, 2010).

3.2.1 La formación de clústeres de software en México.

En año 2002 el gobierno federal, por medio de la Secretaría de Economía, formula la primera política para aumentar y ampliar la capacidad del país a través fomentar el uso de las TIC, a fin de insertar México en un mercado mundial con altas tasas de crecimiento. De forma paralela a la política nacional de fomento al sector, se establecieron un conjunto de instrumentos para contribuir a la producción y a la exportación de las TI en los mercados globales, para consolidar a México como un país atractivo para realizar actividades del desarrollo de software. Las políticas nacionales que han incentivado a la industria del software se muestran en la tabla 3.1.

Tabla 3.1. Políticas nacionales que han apoyado al software.
Fuente: Matus R. Et. al. (2013).

Año	Programa	Objetivo
2002	PROSOFT	Consolidar la competitividad de las TIC a nivel nacional en el mediano plazo y convertirse en el largo plazo en uno de los líderes de este ámbito a partir del apoyo a las empresas que desarrollan software y ofrecen servicios de TI y relacionados.
2004	Fondo PROSOFT	Proporcionar financiamiento a proyectos tecnológicos, en el sector del software.

2005	Moprosoft.	Norma para regular el desarrollo de software en las empresas, basada en ISO 9001:200 y CMM/CMMI. Para subsanar las deficiencias de calidad en la empresas de software en México.
2008	PROSOFT 2.0	Crear las condiciones para que México cuente con un sector de servicios TIC más competitivo a nivel internacional y asegurar su crecimiento a largo plazo.
2008	Programa para el desarrollo de la industria de Medios Interactivos (ProMedia).	Crear las condiciones necesarias para asegurar el crecimiento y la consolidación de la industria de medios interactivos en México, así como elevar su competitividad internacional. Buscará insertar a México en el mercado de medios interactivos aprovechando la creatividad del talento mexicano en la producción de videojuegos y animaciones.
2014	PROSOFT 3.0	Posicionar al país como jugador global de servicios de TI con iniciativas y acciones que son prioritarias en la agenda nacional. Los servicios de TI deben actuar como catalizadores de innovación y productividad para el resto de los sectores económicos nacionales.

Vinculados con las políticas nacionales, algunos gobiernos estatales instrumentaron políticas para fomentar el desarrollo de clústeres que agruparan a las empresas, universidades y/o centros de investigación e instituciones públicas y privadas de apoyo y vinculación entre los distintos actores del sector.

Se reconoce a varias asociaciones que promueven el uso de la TI en diferentes sectores de la industria, en la SEDECO se tienen registradas las siguientes asociaciones que han apoyado el desarrollo de los clústeres en diferentes regiones, ver tabla 3.2.

Tabla 3.2. Asociaciones y organismos que promueven las TI en México.
Fuente: elaboración propia.

Siglas	Nombre	Descripción
AMIPCI	Asociación Mexicana de Internet (AMIPCI, 2014).	Asociación líder en materia de internet en México (cuenta con más de 220 empresas), que fomenta el uso responsable y productivo de internet por parte de organizaciones, empresas, instituciones y usuarios.

		Realiza estudios y eventos anuales que permiten conocer más sobre las tendencias en línea y la percepción de los usuarios en temas fundamentales alrededor de internet. Colabora directamente con organizaciones aliadas, autoridades, reguladores, funcionarios, legisladores, miembros de la academia, la industria y de la sociedad civil por un debido aprovechamiento y uso responsable y productivo de internet y de las TIC.
AMITI	Asociación Mexicana de la Industria de Tecnologías de Información (AMITI, 2015).	Cuenta con más de 339 socios en 20 estados del país, además de convenios de colaboración y relaciones con universidades, embajadas, dependencias gubernamentales, asociaciones y cámaras para promover el desarrollo de la industria de tecnologías de información. El principal objetivo de la AMITI es promover el crecimiento de la industria de TI en beneficio del país, la industria y los miembros de AMITI. Las principales iniciativas que impulsa son: • Incrementar la inversión de TIC en el sector público y privado. • Fomentar la exportación de servicios de TI. • Formación de Capital Humano en las TI. • Gobierno como un serio promotor de las TI. AMITI se encarga de operar el programa IT-LINK, que es un esfuerzo conjunto con la SEDECO, con el apoyo del Banco Mundial, con el cual se busca generar diferentes tipos de enlace entre los actores de la industria de TI (clústeres y PyMEs de TI, academia, organismos empresariales, etc.) con multinacionales del sector.
AMESOL	Asociación Mexicana Empresarial de Software Libre (AMESOL, 2015).	Organización que reúne principalmente a empresas cuyo modelo de negocio involucra o está basado en el <i>open source</i>. Su función es apoyar los planes estratégicos, operativos y de marketing de sus socios. Se identifica como parte de una comunidad más grande de interesados en el software libre. Está consciente de que al apoyar a dicha comunidad, se beneficia todo el ecosistema que incluye a las empresas. Las acciones de AMESOL están divididas en los siguientes temas: • Conocer el mercado mexicano y encontrar nichos de oportunidad para las soluciones y productos de nuestros socios. • Actividades de divulgación acerca de los beneficios del paradigma FLOSS (Free Libre Open Source Software) y específicamente de los

		productos de nuestros socios. • Cursos, seminarios y pláticas a los sectores interesados en el tema: periodistas, estudiantes, académicos, empresarios, investigadores y sector gobierno. • Mayor acercamiento con los principales actores de la industria de TI en el país: cámaras, otras asociaciones, secretarías, poder legislativo y medios de comunicación.
Anadic y AnadicSoft	Asociación Nacional de Distribuidores de Tecnología Informática y Comunicaciones (ANADIC, 2015)	Es un grupo de Empresas y Socios reunidos con el fin de apoyar la relación comercial de sus productos y servicios conectándolos entre sí para la óptima ganancia de todos los miembros que la forman. Es una asociación que promueva el liderazgo tecnológico, empresarial y de conocimiento, generando oportunidades de negocio con Gobierno e iniciativa privada, con representatividad nacional e internacional: • Liderazgo emprendedor. • Trabajo en equipo. • Innovación y Creatividad. • Actitud de servicio al asociado y a la comunidad. • Apertura. • Lealtad. • Honestidad. • Profesionalismo. • Responsabilidad. Promueve también el programa Anadicsoft que ofrece varias soluciones de software (desde la instalación hasta la administración) enfocadas a satisfacer las necesidades de administración y control en los diversos sectores productivos y de servicios tanto para la micro, pequeña y mediana empresa, como para corporativos y gobierno.
ANIEI	Asociación Nacional de Instituciones de Educación en Tecnologías de la Información (ANIEI, 2014).	Su objetivo es contribuir a la formación de profesionales en informática y computación, además busca: • Orientar, proponer y difundir las actividades que en materia de docencia, investigación y extensión educativa se realizan en el área de informática. • Contribuir a la integración, actualización y superación de la educación en informática, en todos sus niveles. • Analizar los problemas relacionados con la enseñanza de la

		informática, proponer soluciones y colaborar en su implantación. Con este fin, entre sus actividades destacan: • Continuar con los trabajos de alineación de los planes y programas de estudio a las necesidades actuales de las tecnologías de la información, estas acciones se implementarán a través del Modelo Curricular y “Paracurricular”. • Vincular a las instituciones educativas con el proyecto “Talento en TI” y la Asociación Academia Industria Gobierno en Tecnologías de la Información (IMPULSA-TI). • Fortalecer el intercambio de proyectos con la comunidad de Latinoamérica. ANIEI organiza anualmente la Reunión de Directores de Escuelas y Facultades de Informática y Computación. Asimismo, organiza un congreso anual llamado “Congreso de Informática y Computación”.
CANIETI	Cámara Nacional de la Industria Electrónica, de Telecomunicaciones y Tecnologías de la Información (CANIETI, 2015).	Su propósito principal es lograr el desarrollo competitivo de la Industria con sentido gremial y responsabilidad social. Mantiene una relación estrecha con en el Gobierno Federal, el Poder Ejecutivo y Legislativo; los diversos Gobiernos Estatales y la amplia gama de organismos empresariales que existen en el mundo. Es un organismo operador del programa MexicoIT (MexicoIT, 2014), respaldado por el Gobierno Federal y los Estatales, así como por diversos organismos empresariales; desplegando una campaña de posicionamiento de nuestra marca país en el mercado global de tecnologías de la información y en particular en los Estados Unidos de Norteamérica, lo cual ha permitido concientizar a las audiencias meta acerca de la existencia, las capacidades, beneficios y oferta diferenciada que existe en México. De igual forma, también administra el programa México-FIRST (MexicoFirst, 2015), apoyado por el Gobierno Federal y los Estatales, además del Banco Mundial, cuyo objetivo principal es la generación de capital humano tanto en cantidad como en calidad, de manera tal que nos permita estar a la altura de países competidores en el ámbito global. La Cámara es reconocida –para beneficio de sus empresas miembro-, como el organismo empresarial que más recursos obtiene de los diversos programas de fomento que ofrece el

		Gobierno Federal y los Estatales.
IMPULSA-TI	Sociedad Academia Industria Gobierno en Tecnologías de Información (ImpulsaTI, 2014).	Es una instancia que profesionaliza el proceso de vinculación entre la academia, la industria y el gobierno con el propósito de formar el capital humano en cantidad y calidad conveniente que requiere la industria de TI. Esto la hace a través del desarrollo de modelos de formación y certificación basados en competencias laborales del capital humano los cuales son creados a partir de la vinculación entre la academia y la industria y el apoyo y reconocimiento del gobierno. IMPULSA-TI desarrolló el proyecto rector denominado “Talento en TI”, cuyo objetivo es desarrollar un modelo que permita alinear las capacidades de capital humano, entre los requerimientos de la industria y la oferta académica. Actualmente se están desarrollando los cincuenta contenidos de los cinco perfiles del Modelo Paracurricular en línea, los cuales son diseñados y contruidos por la Universidad Autónoma de Aguascalientes, la Universidad Veracruzana y la Universidad Politécnica de Aguascalientes. Estos contenidos están disponibles al público en general a través de la plataforma Capacinet. También se ha iniciado la definición de requerimientos para la construcción del Sistema Inteligente de Capital Humano en Tecnologías de la Información (SICAPH-TI), el cual servirá como base de datos para proveer inteligencia acerca de los estudiantes y graduados en carreras de Tecnologías de Información de forma que se puedan conocer las capacidades reales del país en términos de capital humano.

3.2.2 Clústeres de TI en México.

En año de 2009 la CANIETI crea la vicepresidencia Nacional de clústeres, que tiene como objetivo o coordinar el desarrollo integral de los Clústeres de TI en México, a través de la organización de una Coordinación Nacional que permita comprometer la participación de los diferentes clústeres, asociaciones y organizaciones de TI existentes en México (CANIETI, 2015), esta fomentará las sinergias necesarias para:

- Asegurar la participación de la Industria, Academia, Gobierno, Asociaciones y Organizaciones en general involucradas en el desarrollo del Sector de Tecnologías de Información en México.
- Integrar el perfil regional de clústeres que permita complementar y explotar las fortalezas regionales de acuerdo a sus características naturales, potencializando el crecimiento de la Industria.
- Compartir experiencias del desarrollo regional de los clústeres.
- Articular esfuerzos conjuntos que permitan optimizar la inversión de recursos y despliegue de estrategias.
- Apoyar el desarrollo de una estrategia nacional de la Industria de Software y Tecnologías de Información.

Para lograr este objetivo la CANIETI define las regiones para fomentar los clústeres de acuerdo al campo de acción de la industria de TI, ver figura 3.1.

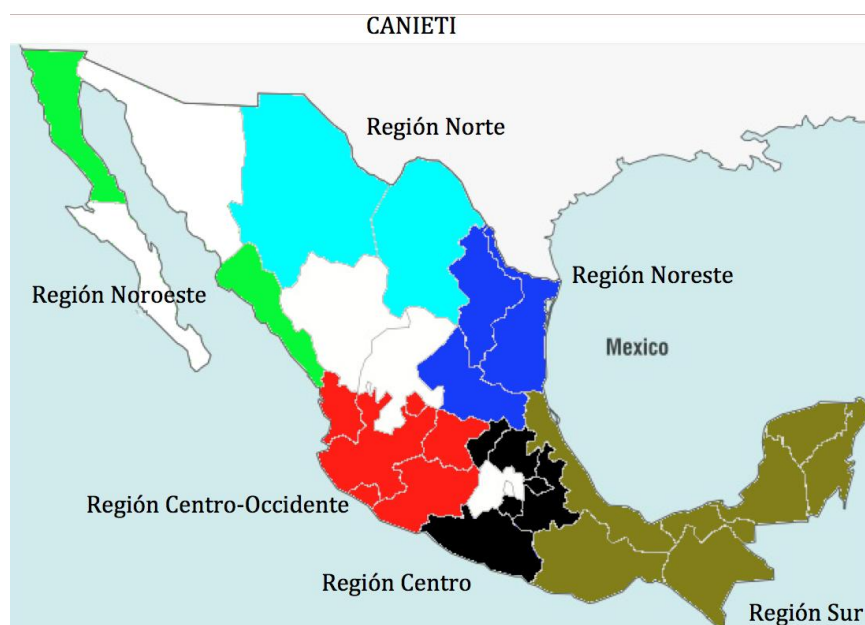


Figura 3.1. Regiones definidas por la CANIETI.
Fuente: (CANIETI, 2015).

La CANIETI, AMITI y la AMIPCI reconocen a 38 clústeres de alta tecnología ubicados en la diferentes regiones del país, que están conformados por más de 2,100 empresas y que

representan a 600,000 profesionales de las tecnologías de información (González, 2012). En la tabla 3.3 se describen.

Tabla 3.3. Clústeres que fomentan la industria de TI en México.
Fuente: (CANIETI, 2015), (Prosoft, SEDECO, 2012)

No.	Nombre	Descripción
1	Clúster de Tecnologías de la Información de Aguascalientes A.C (INNOVATIA).	Industria automotriz y robótica.
2	Clúster de Tecnología de la Información de Baja California A.C. (IT@Baja).	Outsourcing, desarrollo de software y telecomunicaciones.
3	Centro de Tecnologías de la Información de Baja California A.C (CTI).	Investigación, desarrollo de software.
4	Consejo de la Tecnología de la Información de la Laguna A. C. (Coahuila).	Capacitación, fomento a las TI.
5	Clúster de Tecnologías de la Información de la Región Sureste de Coahuila.	Desarrollo de las TI en la industria automotriz, agricultura.
6	Coahuila IT Clúster.	Capacitación, investigación, software.
7	Asociación de la Industria Electrónica y Tecnologías de la Información (Colima).	Electrónica, investigación y desarrollo de software.
8	Clúster de Tecnologías de Información de Chiapas.	En formación, investigación, promoción de TI y capacitación.
9	Clúster de Información y Telecomunicaciones de Chihuahua.	Promoción de TI en minería, aeroespacial y autopartes.
10	Dinformática 21 (Chihuahua).	Soluciones integrales de TI.
11	Corporativo MiPYME de Alta Tecnología (Durango).	Soluciones de TI, aplicaciones móviles, agricultura.
12	Dsoftware (Distrito Federal).	Aplicaciones para los sectores turísticos, alimentos, farmacéutico, servicios y comunicaciones.
13	Clúster de Software ITESI (Guanajuato).	Servicios de TI, software, web, venta y soporte.
14	Clúster Informática y Telecomunicaciones del Estado de Guerrero.	Desarrollo, investigación, innovación, promoción de las TI.
15	Consorcio de Exportación de Tecnologías de Información CTI	Outsourcing de TI, promoción, difusión,

	Jalisco.	video vigilancia, infraestructura de TI.
16	Instituto Jalisciense de TI (IJALTI).	Software, soluciones de TI, multimedia, operación de redes y de seguridad. Electrónica, consumo, público y académico.
17	Prosoftware A.C (Estado de México).	Software, integradoras de sistemas, computación en nube, aplicaciones móviles. Industria, manufactura, público y financiero.
18	Clúster de Tecnologías de la Información y Comunicaciones de Michoacán.	Software, web, animación, seguridad y administración.
19	Clúster para desarrollo de Software en ANADIC Michoacán.	Venta, soporte, integración de TI en la industria.
20	Asociación de la Industria del Software A.C (Morelos).	Desarrollo, capacitación y consultoría, centro de datos.
21	Instituto Nayarita de las Tecnologías de la Información.	Investigación, difusión y promoción de las TI. En conformación.
22	Monterrey IT cluster (Nuevo León).	Aplicaciones a la medida, servicios TIC e infraestructura. manufactura, finanzas y de ventas al por menor.
23	Consejo para el Desarrollo de la Industria de Software de Nuevo León. Csoftmtty.	Inteligencia de negocios y computación de nube. Salud, telecomunicaciones y finanzas.
24	Consejo para el Impulso de la Industria de Medios Interactivos (Nuevo León). MIMEC.	Inteligencia de negocios, movilidad, computación en la nube, audio, video, cine y animación.
25	Clúster TI Oaxaca.	Investigación , promoción, desarrollo.
26	Clúster para la Innovación en tecnologías de la Información (Puebla).	Investigación, desarrollo y promoción, automotriz e industria.
27	Arquitectos de Software (Puebla).	Desarrollo, investigación, automotriz e industrial.
28	InteQSoft Clúster de Tecnologías de Información de Querétaro.	Software, sistemas incrustados, medios interactivos, finanzas, logística, automotriz y servicios públicos.
29	Parque Tecnológico SonoraSoft.	Software a la medida, promoción y difusión. Agricultura e industria.

30	Código TI (Sinaloa).	Aplicaciones móviles y soluciones en la nube. Academia, de finanzas, salud, público y comercio.
31	Clúster Mochis (Sinaloa).	Software, investigación. Naval e industrial
32	CODESIN Zona Norte (Sinaloa).	Software, investigación. Gobierno e industria.
33	CITI Tabasco.	Software a la medida, pruebas, modelación, desarrollo de RIA y multimedia. Petrolera e industria.
34	Clúster de Tecnologías de la Información Tamaulipas. Tit@m.	Promoción, difusión, desarrollo. Aeroespacial, metalmecánica, agricultura y energías renovables.
35	Clúster de Tecnología de Información Tlaxcala. Clustec.	Software, consultoría y redes. Industria, banca y servicios públicos.
36	Clúster Veracruz. Ver@cluster.	Promoción, difusión, investigación. Naval, industria y agricultura.
37	Consejo de la Industria de la Tecnología de la Información de Yucatán.	Desarrollo a la medida, empaquetado y servicios. Industria, educación y gobierno.
38	Clúster de Zacatecas.	Sistemas incrustados, aplicaciones a la medida y automatización de procesos. Se enfoca principalmente a la industria e investigación.

3.2.3. Organización de un Clúster TI para la competitividad.

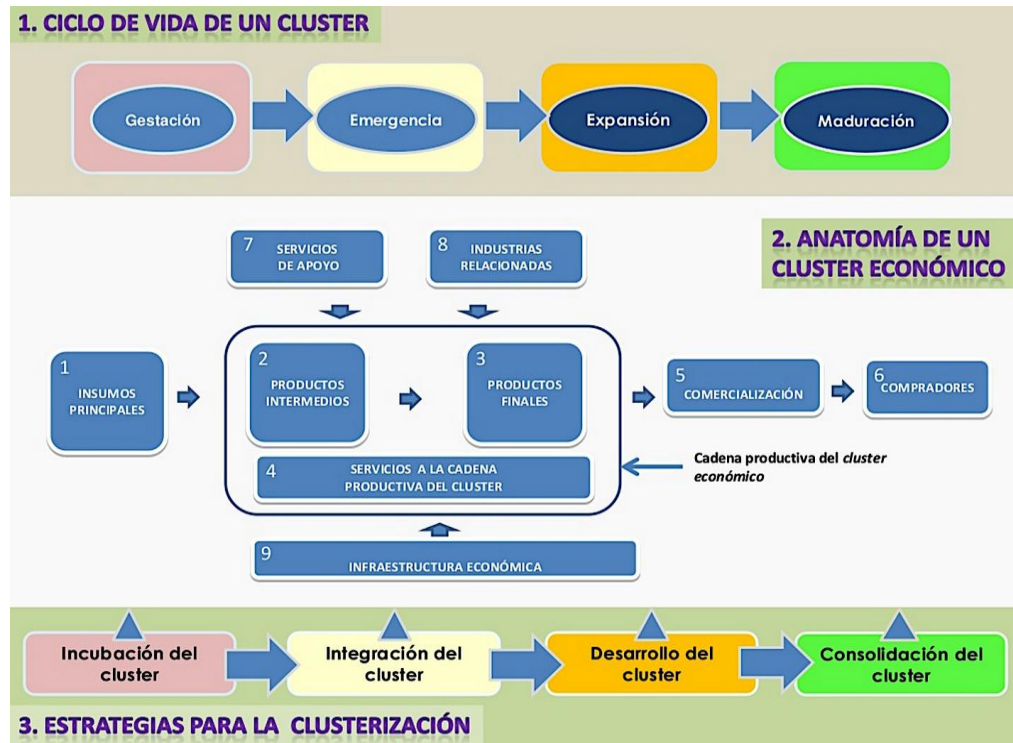
Como describe (Guerra C., 2013) los clústeres proporcionan una estructura apropiada para visualizar el compromiso y la interacción entre los actores relevantes (capital social) que desarrollan e implementan los tres impulsores estratégicos para la innovación y el desarrollo (I+D), de tal forma que las estrategias y políticas públicas para impulsar la prosperidad regional se logren exitosamente. En la figura 3.2 se muestra el diamante de Porter para los Clúster.



Figura 3.2. Diamante de Porter para los clúster de TI.
Fuente: Guerra C. (2013).

Los CTI tiene un proceso evolutivo (ciclo de vida) que va desde la gestación (aglutinamiento de empresas), la emergencia (empresas con vínculos incipientes), la expansión (empresas con vínculos fuertes para complementarse) y finalmente la maduración (redes de colaboración, innovación y desarrollo) pasan por las etapas de incubación, integración, desarrollo y consolidación.

En la figura 3.3 se muestra el ciclo de vida de un clúster, su anatomía o pertinencia económica y sus estrategias para su construcción.



3.3. Proceso evolutivo de la gestación y maduración de un Clúster.

Fuente: Guerra C. (2013).

En su reporte de la estrategia de implementación del programa Prosoft 2.0 para el año 2013 la SEDECO describe el impacto de este programa en los CTI y los beneficios que este ha ofrecido a la empresas que han participado (SEDECO, 2012).

En México sobresalen las características de CTI que se muestran en la figura 3.4 que los hacen muy atractivos para las empresas PyMEs.



Figura 3.4 Características de los Clúster de TI.
Fuente: (SEDECO, 2012).

En México hasta el año 2015 se tenían 38 clúster de TI conformados que aglutinaban a más de 1,000 empresas. Se ha observado que este tipo de agrupamientos han generado una ventaja competitiva para las compañías que; compiten en un entorno global, impulsan la actividad económica de otros sectores económicos de la región y generan beneficios para la población.

Los CTI surgen de una serie de iniciativas de las empresas de una región y tienen como origen unas iniciativas gubernamentales, empresariales o ambas. SEDECO en su estudio encontró que el 45% de los CTI conformados en México tienen una iniciativa mixta, otro 45% fueron por una iniciativa empresarial y el 9% por una iniciativa gubernamental. Los objetivos que han logrado los Clústeres son los siguientes:

- Aumentan el empleo en la región.
- Aumentan las exportaciones.
- Atraen empresas nuevas para inversión.
- Aumentan el valor agregado.
- Desarrollan cadenas de valor.
- Fomentan la innovación.

- Apoyo para fondos.
- Mejoran el ambiente de negocios.
- Cuentan con asesoría profesional-consultoría.
- Reducen costos de producción.
- Fomentan la investigación en la región.

Los CTI buscan como principal objetivo satisfacer las necesidades de sus agremiados en las siguientes:

- ✓ Vincular la academia-empresa-gobierno,
- ✓ Certificar procesos de calidad en sus empresas miembro,
- ✓ Comercializar productos o servicios de sus empresas,
- ✓ Desarrollar e implementar estrategias de crecimiento y
- ✓ Promover la investigación, estudios de mercadotecnia y la demanda en TI.

De acuerdo al estudio desarrollado por la SEDECO (2012) se observó que los CTI tratan de fomentar la participación interna de sus empresas miembros y muchas de sus actividades se centran en esta estrategia.

De acuerdo a sus logros e impacto a nivel regional, nacional e internacional se elaboró un matriz comparativa de los principales CTI de México donde se clasifican de acuerdo a su madurez (inicio, en formación, en expansión y consolidado) y competitividad (desde no competitivo, competitividad mínima, competitivo a nivel regional, competitivo a nivel nacional y competitivo a nivel internacional), ver figura 3.5.

Madurez Competitividad	Iniciación	Formación	Expansión	Consolidación
Nivel I. No competitivo				
Nivel II. Competitividad mínima		CLUSTER TI OAXACA		
Nivel III. Competitivo a nivel regional	CONCYTEG COAHUILA IT CLUSTER	AIETIC AISAC CLUSTER TI PUEBLA CLUSTER TIM INNOVATIA CTI LAGUNA CLUSTER TI TLAXCALA TIT@M, A.C. VER@CLUSTER, A.C. INTEQSOFT	CITYUCATAN FIDSOFTWARE TI SONORA, A.C.	
Nivel IV. Competitivo a nivel nacional	NEW MEDIA	CSOFTMTY PROSOFTWARE MIT CLUSTER	IJALTI IT@BAJA	
Nivel V. Competitivo a nivel internacional				

Figura 3.5. Matriz comparativa de los principales CTI en México de acuerdo a su madurez y competitividad.
Fuente: SEDECO (2012).

Una de las principales formas de promover la calidad en los CTI es la certificación de las empresas desarrolladoras al interior, de acuerdo a los resultados del Prosoft 2.0 (2013) se logró la certificación de 7 empresas en CMM (2%), 48 en CMMI (13%) y 48 en Moprosoft (13%). De aquí se desprenden las siguientes conclusiones:

1. Los clústeres son relativamente jóvenes y se han posicionado en un nivel de competitividad regional y nacional.
2. Los factores externos influyen en gran medida en la competitividad de los clústeres (vulnerabilidad ante las condiciones económicas y políticas en que se desenvuelven).
3. Las condiciones de la demanda (tamaño de mercado local y sofisticación) es el principal factor que afecta la competitividad.
4. En relación con la madurez de los CTI se concluye que:
5. Se requiere fortalecer los niveles de comunicación y cooperación entre sus empresas.
6. Necesidad de promover la certificación tanto nacional (MOPROSOFT) como internacional (CMMI).

7. Promover el crecimiento y diversificación de mercados, orientados a incrementar la participación en el mercado nacional e internacional.
8. Establecer programas de vinculación permanente con instituciones académicas y de investigación.

3.3 La TICs como competitividad en PyMEs.

En el estudio realizado por el Instituto Mexicano de la Competitividad (IMCO, 2014), se muestran los primeros resultados del plan nacional de desarrollo 2013-2018 (Gobierno de la Republica, 2013) en cuestiones de la promoción, el emprendimiento y el aprovechamiento de las TICs en México.

En el año 2012 México destinó 0.37 del PIB en I+D, se pretende que esta inversión crezca en los siguientes 6 años al 1% del PIB. Esta inversión permitirá promover en todos los ámbitos del país en investigación y desarrollo, fomentando así la innovación. Uno de los pilares de la innovación regional son los clústeres regionales, que ayudan al emprendimiento en TICs y la democratización de la productividad. Las empresas PyMEs que fomentan el uso y conocimiento en TICs se han visto rezagadas en apoyos para la I+D, se reconoce como principales barreras:

1. Innovación. Existe una baja inversión en I+D en comparación con los países miembros de la OCDE, en México el 30% de la inversión proviene del sector privado, el 80% es del sector público. Debe promoverse más la inversión del capital privado en TICs.
2. Entorno para hacer negocios. En México se reconoce un alto costo y dificultad para la realización de trámites, legalización y pago de impuestos al crear una empresa PyME.
3. Infraestructura de apoyo. Aunque en años recientes se ha apoyado la creación de PyMEs a través de incubadoras, el modelo ha fallado; 1) las incubadoras no agregan valor y 2) no se cuentan con vínculos fuertes con el sector privado.
4. Capital humano. En México existe graves problemas de educación donde el capital humano ha demostrado ser incapaz de competir con los mercados globales, la falta de acceso a infraestructura de TI y el dominio del idioma inglés son unas de la principales

causas. Los empleadores mencionan que los trabajadores de la PyMEs se encuentran en niveles bajos e incluso nulos en estos dos puntos (TI + Inglés).

5. **Financiamiento.** A pesar de los avances en programas nacionales para incentivar el sector de la TI, todavía hay una brecha muy grande para acceder a estos incentivos. El porcentaje de crédito a empresas PyMEs representó 21% del PIB en México, mientras que el promedio en Latinoamérica fue del 36%. Falta más promoción para facilitar el acceso a recursos o fuentes de financiamiento.
6. **Perfil de la demanda.** A pesar de que las PyMEs son las principales promotoras del uso y explotación de las TICs en México, faltan planes concisos para mejorar el uso entre las empresas y los clientes. De acuerdo con la OCDE, México es de los países más caro para contratar un servicio de banda ancha - el promedio es de 5.3 Mbps de velocidad a un costo de 16.6 dólares por 1 Mbps de descarga (en países de Latinoamérica el promedio es de 19 Mbps a un costo de 7.59 dólares por 1 Mbps) -. Otro factor que impacta es la piratería que representó en el año de 2012 un 57% de las ventas totales en software en México - según la Business Software Alliance (BSA) -.
7. **Integración de la industria TIC.** Aunque en México se reconocen a 28 estados con Clústeres de TI (38), solo el 23% de las nuevas empresas pertenece a alguna asociación y 28% a un clúster. Pese a los logros de PROSOFT todavía hay muchos problemas para vincular a las empresas con estas entidades. Faltan planes e incentivos para que nuevas empresas se inserten a la cadena de valor (en la mayoría de los clústeres a nivel nacional existen listas de espera para anexar nuevas empresas).
8. **Cultura emprendedora.** En México a pesar de los incentivos esta cultura no se ha consolidado, falta una cultura de mentoría por parte de los empresarios exitosos, esto se debe a la falta de interés y de mecanismos que la fomenten.
9. **Estrategia gubernamental.** A pesar de la gran variedad de programas de apoyos el 93% de las PyMEs reconoce que hay muchos programas de apoyo y que son complicados de acceder; existen gestores que cobran por conseguir recursos. Es necesario mejorar el proceso de gestión y facilitar el acceso a estos programas.

A pesar de los avances del sector TIC en México, se ha quedado corto ante otros competidores. En 2012 el sector TIC de la India representó ingresos de 100 mil millones de dólares, más del doble que el sector TIC mexicano. Mientras en México la industria de software representó 0.7% del PIB, en Estados Unidos representó el 4%, casi 6 veces más. En la figura 3.6 se muestran las ventajas competitivas del sector de las TIC en México.



Figura 3.6. Ventajas competitivas del sector TIC en México.
Fuente: IMCO (2014).

En los últimos años, consultores internacionales de empresas especializadas en TI sitúan a México como uno de los jugadores más importantes en el sector. La han colocado en el tercer lugar a nivel mundial en proveeduría de servicios de TI —sólo después de la India y Filipinas—, lo que lo vuelve líder en América Latina.

El IMCO propone una serie de factores que las PyMEs deben de conocer para ser competitivas en el sector de las TI, en la figura 3.7 se muestran estos factores.

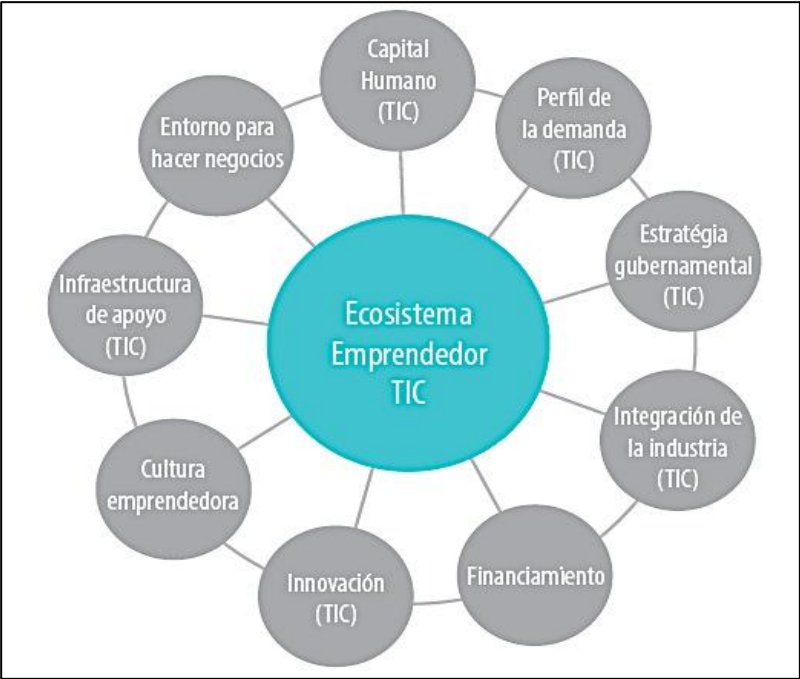


Figura 3.7. Factores que estimulan o cohiben la decisión de una persona para iniciar un negocio.
Fuente: IMCO (2014).

Las empresas emprendedoras de PyMEs en México se centran en 16 estados de la república donde sobresalen los estados de Baja California Sur, Nayarit, Campeche y Colima, en la figura 3.8 se muestra los estados de la república por densidad de creación de PyMEs por cada 1000 habitantes.

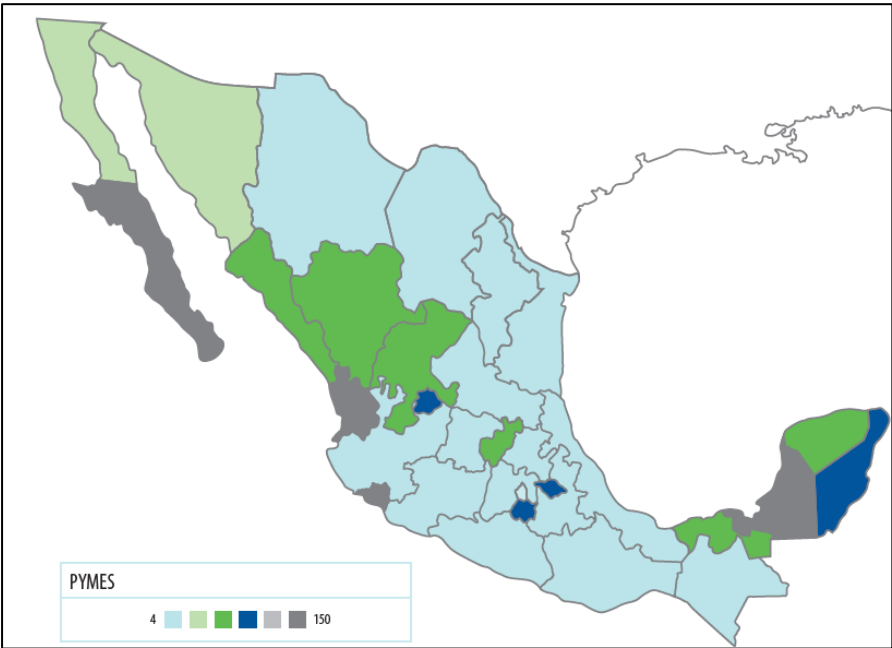


Figura 3.8. Creación de PyMEs en el sector de TI para el año de 2012.
Fuente: (IMCO, 2014).

Las principales fuentes de financiamiento para la innovación y la creación de PyMEs se muestran en la figura 3.9.

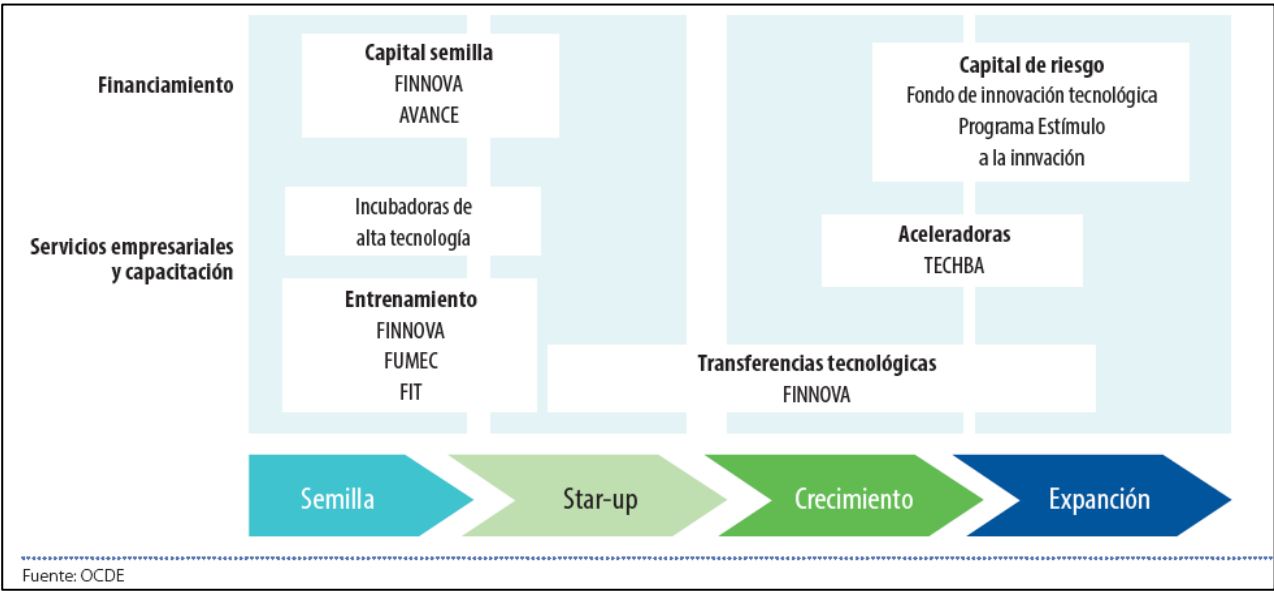


Figura 3.9. Principales programas para el emprendimiento e innovación en México.
Fuente: (OCDE, 2009).

Se reconoce como principales barreras para hacer negocios en México las siguientes:

1. Dificultad para pagar impuestos. Este es un problema muy complejo para las PyMEs ya que el tiempo y costo que deben de invertir en sus declaraciones y la regulación hacendaria.
2. Poca capacidad para ejecutar contratos. Una empresa PyME generalmente carece de una infraestructura legal para formalizar y ejecutar contratos atractivos. Esto se vuelve una barrera importante para el crecimiento de la PyME.
3. Alta presencia de corrupción. Esto se convierte en una barrera ya que las empresas PyMEs que van iniciando en el negocio de las TI tienen que invertir importantes cantidades de capital para competir.

3.4 Experiencias y desafíos en la apropiación de las TICs en PyMEs.

Las empresas compiten y colaboran para generar riqueza. En México para el año 2010, las PyMEs generaron el 24.4% de los empleos del país y el 19.2 de la producción bruta total (INEGI, 2010). De acuerdo con un estudio (AMITI, INFOTEC, 2013) la contribución de las PyMEs cambió del 52% al 34% en los últimos 10 años; donde el 90% de estas organizaciones cierra antes de cumplir la década, en parte, debido a los siguientes problemas (Infotec, 2014):

- a) la falta de dinero o capital (99% de los casos),
- b) la falta de experiencia (79%),
- c) el carecer de buenos sistemas de información (59%) y
- d) las excesivas inversiones en activos fijos (58%).

En estudios recientes se reconoce que el 45% de las PyMEs que permanece en el mercado por 12 años o más se caracteriza por su estabilidad y experiencia en el mercado, son empresas que compiten, innovan y se apropian de la TICs como un factor preponderante (Velasco E., 2013).

Uno de los más grandes retos parece ser la inexistencia de una cultura de calidad en las PyMEs, pues el 50% no emplean técnicas y 80% no obtienen certificaciones relacionadas con su ámbito de competitividad. Al respecto, se reconoce la falta de visión a largo plazo de los

empresarios de las PyMEs como una de las barreras más importantes para generar competitividad. En el estudio sobre los “Efectos del TLCAN en la Competitividad de la Pequeña Empresa en México”, los autores describen cómo muchas PyMEs no estaban preparadas para un contexto competitivo que fomentaba el TLCAN (Haar, Leroy-Beltrán, & Beltrán, 2004).

3.5 Competitividad de las PyMEs.

La competitividad es un tema que define (Porter, 2009), como a la innovación y la diversificación de las exportaciones e inversión en países extranjeros como fuentes de competitividad y productividad de las industrias y por ende de las naciones. El autor menciona que las estructuras de gobernanza deben catalizar la competencia sin intervenir directamente en ella; igualmente describe cuatro características que afectan la capacidad nacional para innovar:

1. Condiciones de los factores,
2. Condiciones de la demanda,
3. Industrias relacionadas y de apoyo y
4. Estrategia, estructura y rivalidad de las firmas.

Nuevos enfoques toman lugar ante el estudio de la competitividad de las naciones y de las empresas. Uno de ellos es el modelo de la competitividad sistémica concebida por el Instituto Alemán de desarrollo en los años 90. De acuerdo con Haar et. al (2004), la competitividad sistémica se nutre de distintas perspectivas –tales como economía de la innovación, redes, neo-estructuralismo y neo-institucionalismo, conglomerados– así como de los hallazgos de la Comisión Económica para América Latina y el Caribe (CEPAL) y de la OCDE y se describen a continuación.

En la competitividad sistémica se analizan cuatro niveles interdependientes de interacción entre el Estado y el mercado (meta, macro, meso y micro) a fin de equilibrar ambos poderes, reducir los costos de transacción y facilitar la formación de sistemas de producción integrados y de redes empresariales horizontales integradas por agentes públicos y privados.

En Haar et. al (2004) se describe un modelo de competitividad de las PyMEs. Ver figura 3.10.



Figura 3.10. Modelo de competitividad de las PyMEs.
Fuente: (Haar, Leroy-Beltrán, & Beltrán, 2004).

En la práctica, se puede observar que las empresas líderes en el mercado se distinguen por tener un desempeño eficiente en el uso de recursos, la permanencia a través del tiempo, la lealtad de sus clientes y la generación continua de propuestas de valor sostenibles, rentables y atractivas en el mercado que atienden. Además desarrollan y adoptan las TIC para gestionar los procesos clave de la cadena de valor y agregar valor a los productos y servicios que son entregados al cliente final. De la misma forma, se generalizan prácticas como el *outsourcing* y la colaboración inter-organizacional a través de alianzas, *joint ventures* (acuerdo comercial de inversión conjunta a largo plazo entre dos o más empresas), redes y asociaciones (Pallares, 2004), que ayudan completar sus capacidades y ofrecer un mejor servicio o producto final. Una variedad de estudios analizan los factores de la competitividad, en (Cabrera, López y Ramírez, 2011) describen los siguientes detonadores:

1. Estrategias de negocio,
2. Cultura organizacional,
3. Tecnología,
4. Cooperación inter-organizacional,

5. Conocimiento del mercado,
6. Investigación, desarrollo e innovación (I + D + i),
7. Capital intelectual,
8. Calidad,
9. Productividad,
10. Gestión financiera,
11. Precios,
12. Diferenciación y
13. Servicio post-venta

3.5.1 Alineación de las TI y ventaja competitiva.

La literatura en teoría organizacional reconoce que la capacidad de absorción es clave para la competitividad a mediano y largo plazo de las empresas, ya que se relaciona de manera directa con la innovación y la generación de ventajas competitivas (Infotec, 2014). Esta capacidad sobresale de otras debido a su relación directa con la innovación (Expósito-Langa, Molina-Morales, & Capó-Vicedo, 2009), además de que no ha sido tratada tan ampliamente en organizaciones PyME. En la figura 3.11 se muestra los factores que influyen en la capacidad de absorción.

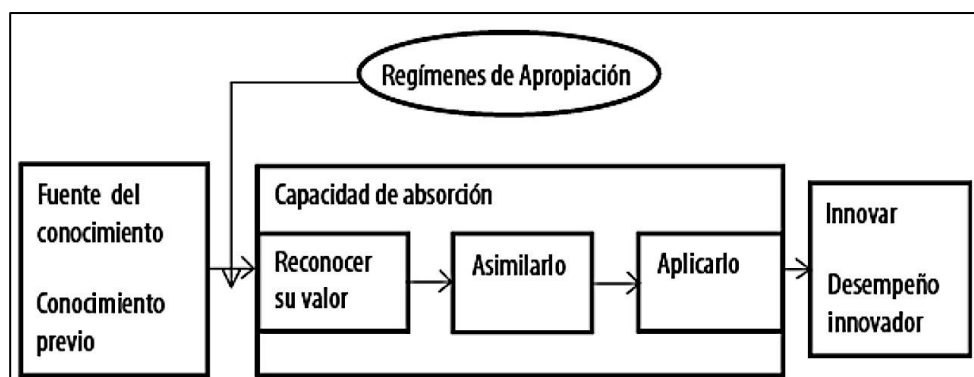


Figura 3.11. Análisis de la capacidad de absorción de una PyME.
Fuente: (Infotec, 2014).

3.6 Tecnologías de la información en la inteligencia organizacional.

Un elemento crucial de la administración del conocimiento es el sistema de gestión del conocimiento, que es su aspecto tecnológico, está compuesto de instrumentos dirigidos al individuo y a la organización, para mejorar la productividad del trabajo de conocimiento (Maier y Hädrich, 2007). Las características del sistema de gestión del conocimiento son:

1. Metas: la principal meta del sistema de gestión de conocimiento es tener a disposición del personal conocimiento del pasado para las actividades del presente, con la finalidad de incrementar la efectividad en los diferentes niveles de la organización.
2. Procesos: el sistema de gestión de conocimiento es desarrollado para apoyar las herramientas, procesos o proyectos del ciclo de vida del conocimiento (creación, organización, almacenamiento, recuperación, transferencia, refinamiento, empaquetamiento, reúso, revisión y retroalimentación).
3. Plataforma: el enfoque en el proceso debe ser desde visión centrada en el usuario para ofrecer un sistema base que permita capturar y distribuir el conocimiento.
4. Servicios avanzados de conocimiento: los servicios que se ofrecen en el sistema continuamente se renovados, de acuerdo a los avances tecnológicos.
5. Instrumentos: son los componentes técnicos de los servicios.
6. Especificidad: el sistema de gestión del conocimiento debe permitir la personalización, de manera que cada individuo debe contar con un espacio de trabajo privado que se adecue a sus requerimientos.
7. Participantes: los usuarios del sistema tienen un rol protagónico en el sistema porque ellos son los que lo alimenta y lo consultan.

3.7 Innovación tecnológica en la industria TI.

La industria TI se caracteriza por ser dinámica y tener una tasa de crecimiento, que por lo general, es mayor a la tasa de crecimiento del total de la economía. Hay diversos estudios que sugieren que la industria TI crece a cierta proporción del crecimiento de la economía visto por el PIB. El desempeño económico constituye la capacidad de una región o país para generar riqueza,

condición necesaria para el crecimiento y desarrollo de cualquier economía (Select Estrategia, 2014). La estructura productiva y sus posibilidades para generar satisfactores para la población, entre ellos acceso a TI y sobre todo al impacto que tiene en la productividad, es parte fundamental del enfoque que se tiene del modelo de oferta y demanda de recursos humanos TI.

En un país como México, con grandes disparidades, el crecimiento de las TI y las TIC (Comunicaciones) en referencia al PIB, expresa amplias divergencias por región, sin embargo, el PIB y el crecimiento de la industria son dos variables de referencia para el modelo de la oferta y demanda de las TI, y en gran medida se debe al indudable grado de correlación entre el desempeño de la industria TI y el PIB.

En la figura 3.12 se muestra el valor del PIB, de TIC y de TI a partir del año 2000 hasta 2025. En valores de mercado, el PIB tiene una trayectoria más ascendente a diferencia de las TIC. La perspectiva para la industria es alentadora e implica en términos de capital humano, que debe haber una amplia disponibilidad para compensar las necesidades de una industria TI y economía en crecimiento y con mayor diversidad de industrias dinámicas.

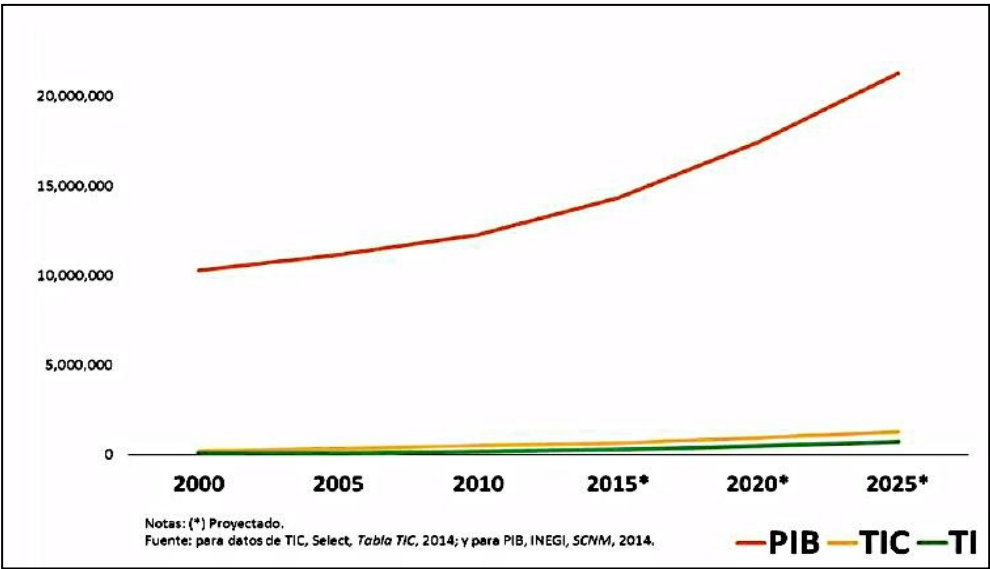


Figura 3.12. Crecimiento del PIB con respecto de la TI y las TIC, miles de millones de pesos Mexicanos.
Fuente: (Select Estrategia, 2014).

Sin embargo pese al crecimiento sostenido proyectado para las TIC y TI es necesario incentivar la innovación en este sector para incrementar las ganancias. Se prevé que las TI y las TICS se vuelvan un referente para el empleo en los próximos 10 años, en la figura 3.13 se muestra la tasa de crecimiento del empleo por sector donde las TI y TICS se vuelven un referente.

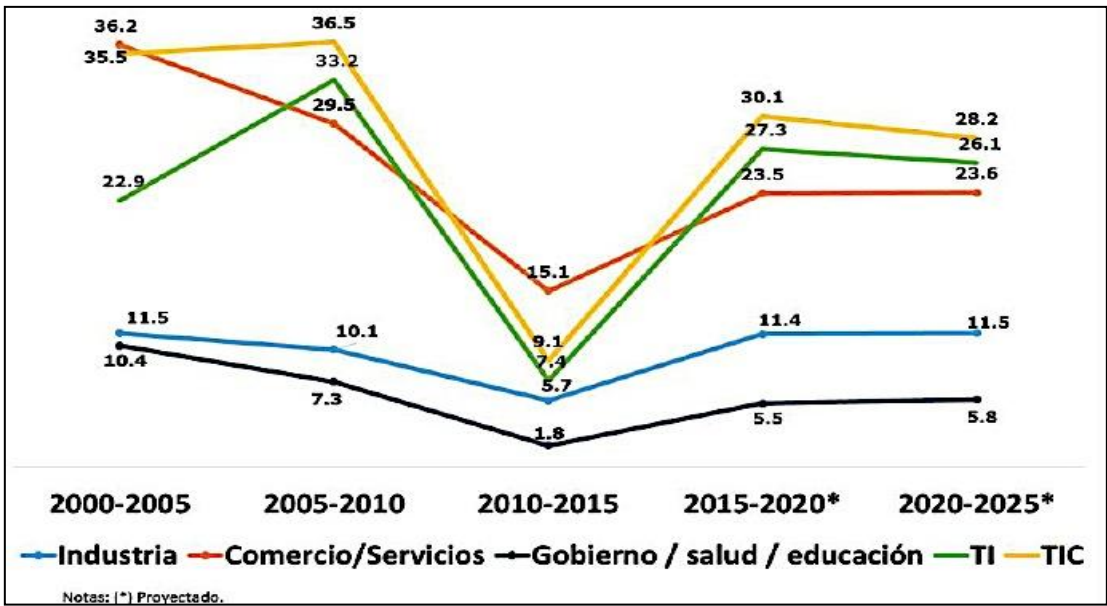


Figura 3.13. Tasa de crecimiento por sector en el empleo.
Fuente: Select Estrategia (2014).

Para promover la innovación el acervo de profesionistas TI por sector se prevé que incremente, siendo aún insuficiente para incentivar a la industria. Ver figura 3.14.

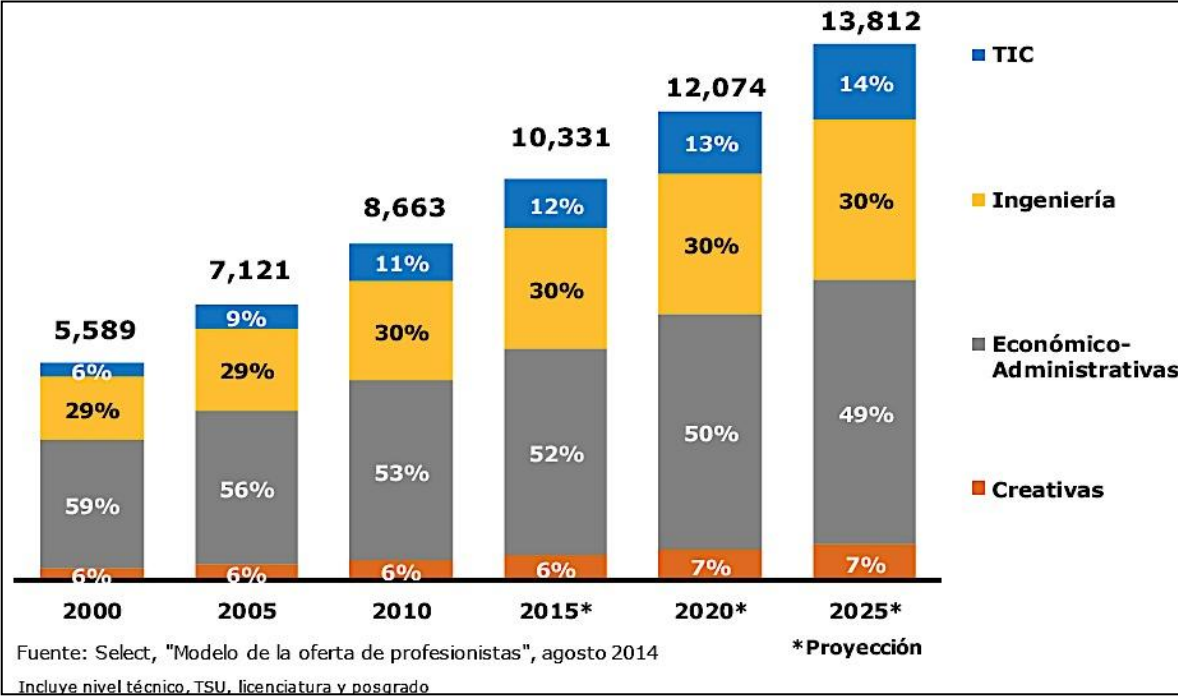


Figura 3.14. Crecimiento de profesionistas por sector, proyección 2025.
Fuente: Select Estrategia (2014).

Para mejorar la innovación se encuestó a profesionistas del sector para saber qué áreas estratégicas de las TI deberían de tener capacitación, en la figura 3.15 se muestra el resultado con base en 801 encuestas aplicadas.



Figura 3.15. Necesidades de capacitación para fomentar la innovación en TI.
Fuente: (Select Estrategia, 2014).

Otra forma de buscar innovación en identificar los procesos que requieren “personal con certificaciones” son; finanzas, TI, desarrollo, ventas, administración, telecomunicaciones, soporte y producción.

En la figura 3.16 se muestran los principales procesos que requieren certificaciones como método para garantizar la calidad e innovar.

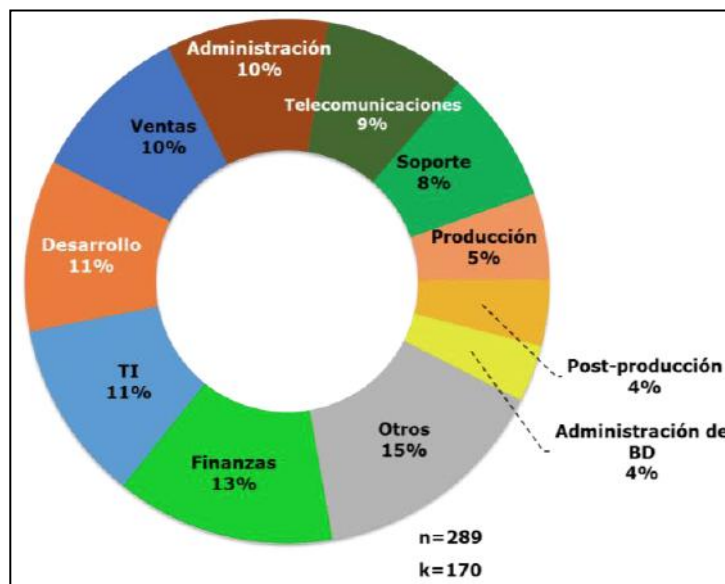


Figura 3.16. Procesos del área de TI que requieren certificaciones para promover la calidad e Innovación.
Fuente: Select Estrategia (2014).

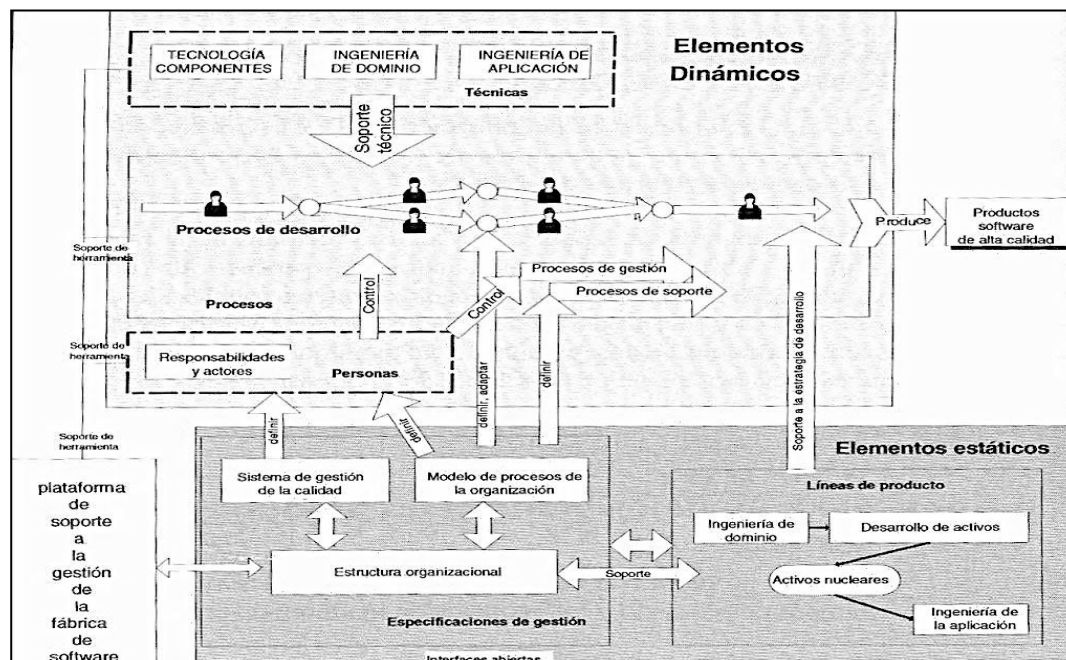
3.8 Fábricas de software.

El termino de fábrica de software surge desde los años 70 donde se define como la manera de generar software a manera de un proceso industrial, la metodologías y métodos de construcción se centraron en esta forma de agilizar el desarrollo. Sin embrago la crisis del software revolucionó este concepto, en los años 80s y 90s el énfasis fue la reutilización y la definición de procesos para mejorar el desarrollo y mejora la aproximación hacia una FS. En (Piatini V. & Garzas P., 2010) se describe las diferentes formas de estas fábricas:

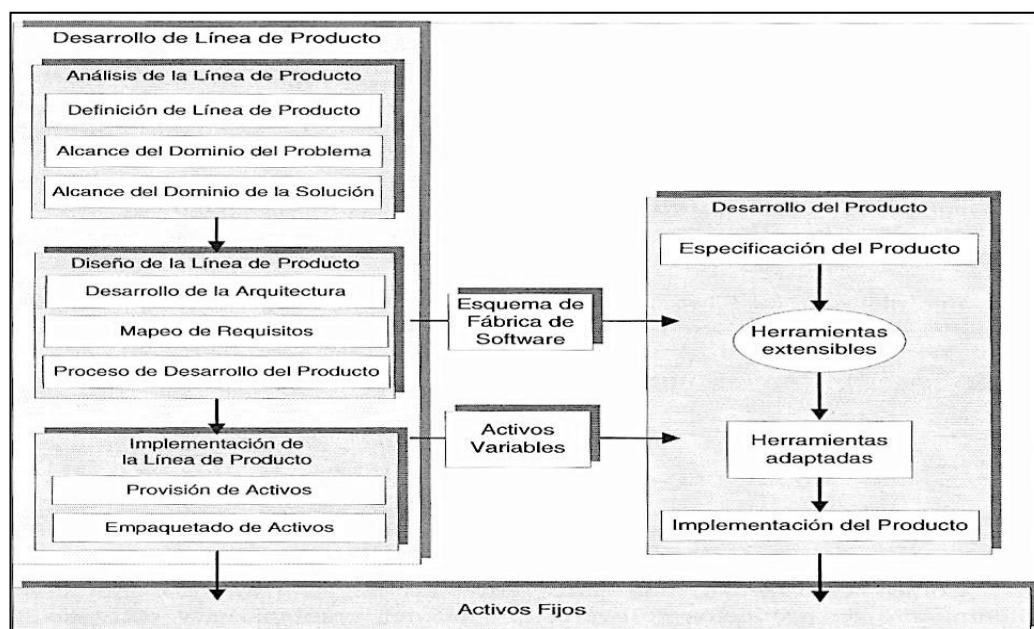
1. Fábricas basadas en entornos de desarrollo. Surgen a finales de los 80s y se basa en herramientas CASE y Entornos de Desarrollo Integrados (IDE). Donde el desarrollo está dirigido por una herramienta CASE/IDE.
2. Fábricas basadas en la experiencia. También conocidas como laboratorios de software o SEL (*Software Engineering Laboratory*). El desarrollo se basa en la experiencia, el aprendizaje y la transferencia tecnológica en un laboratorio. Se utiliza un “empaquetado” de experiencias para la reutilización y generación de código con calidad. Se utiliza un enfoque incremental y la planificación de procesos.

3. Fábricas basadas en la madurez. Basadas en el modelo CMM del SEI, cuyo objetivo es crear un marco para la mejora de procesos que permita construir procesos predecibles, fiables y auto mejorables. La estrategia es seguir mejoras escalonadas, el entendimiento común en toda la organización de las actividades, roles y responsabilidades del proceso. Los niveles de capacidad definidos para todos los procesos son: Inicial, Repetible, Definido, Gestionado y Optimizado.
4. Fábricas basadas en la reutilización. Se propone combinar el concepto de FS con sistemas de fabricación flexibles; donde se construyen partes para ser integradas, se optimiza la producción de componentes y el ensamblado de productos. La certificación, estándares de calidad y las pruebas, son forma de garantizar un desarrollo con calidad.
5. Fábricas para la renovación de software. En los años 2000 se encontraron muchos problemas para dar mantenimiento al software, sobre todo conversión y actualización de software hacia las nuevas tecnologías. El software es tratado como una línea de ensamblado, donde una serie de herramientas de transformación van agregando componentes nuevos al software.
6. Fábricas enfocadas a técnicas de calidad. Es la aplicación de los conceptos de calidad total o gestión de la calidad y reutilización, para la generación con flexibilidad de software. Se combinan la planificación y los sistemas de producción.
7. Fábricas de software basadas en buenas prácticas. Para los años 2010 los FS se centran en diferentes estrategias para la elaboración o mantenimiento de productos de software, destacan la estrategias:
 - Desarrollo tradicional. Con métodos clásicos y buenas prácticas.
 - Generación de código automático. Se basa en un desarrollo dirigido por modelos o programación automatizada.
 - Reutilización. Desarrollo basado en componentes o líneas de producción.

En la figura 3.17 A) y B) se muestra la evolución de una FS basada en modelos de calidad y una FS basada en las buenas prácticas.



A) Fábrica de software basada en modelo o línea de producto, propuesta por Li.



B) Fábricas de software basada en buenas prácticas, propuestas por Greenfield y Short.

Figura 3.17 A) y B). Evolución de FS basadas en modelos a FS basada en buenas prácticas.
Fuente: (Piatini V. & Garzas P., 2010).

En la actualidad las FS deben seleccionar un modelo de calidad de referencia para satisfacer las necesidades de los clientes. De acuerdo a estudios se ha observado que los modelos más utilizados son el CMM/CMMI y el ISO 15504 SPICE. Como ya se revisó el CMM/CMMI no es aplicado en PyMEs por el alcance y costo de este modelo. Sin embargo la norma ISO/IEC 15504 con revisión 7 si ofrece la oportunidad de certificar a FS que son MiPyMEs (Micros y PyMEs) además de ser compatible con el estándar ISO/IEC 9001.

3.8.1 Buenas prácticas aplicables a las FS MiPyMEs.

A continuación se sintetizan una serie de buenas prácticas aplicables a una gran mayoría de las FS, en la tabla 3.4 se describen.

Tabla 3.4. Buenas prácticas utilizadas en FS.
Fuente: (Piattini & Garzas, 2010).

Buena Práctica	Descripción	Aplicada en
Integración continua y gestión de la configuración.	Proceso de integración para un producto de software, donde se desarrolla, se programa, se prueba y se somete a pruebas de calidad. Se tiene un control exhaustivo de la evolución del proyecto, detección temprana de errores, versiones actualizadas de software, se reducen tiempos de integración y hay control de calidad continua por entregable o elemento de desarrollo.	MiPyMES. Énfasis en el talento. Construcción incremental. Modelo de cascada. Reutilización de todo Lo que no mide no se controla.
Control de calidad exhaustivo, periódico y automático.	Se enfatiza en fijar objetivos claros y medibles, realización de mediciones de manera periódica, mediciones automáticas, definir diferentes niveles de abstracción. Las mediciones de calidad deben hacerse durante el desarrollo, y a la vez durante la construcción del producto, repuesta rápida antes fallas retrasos y errores. Se hace un seguimiento periódico o frecuente.	PyMEs. En productos complejos. Entornos donde se usan herramientas automáticas.
Diseño basado en el conocimiento.	Las decisiones se centran en el diseño del proyecto, se trata de reducir tiempo de desarrollo y minimizar riesgos, se requiere de diseñadores expertos que utilizan los patrones, heurísticas, principios, ontologías para dar	PyMEs. Calidad en el diseño. Uso de patrones, heurísticas, principios y ontologías.

	calidad al producto. Exige la integración de un arquitecto de software que aplique cada patrón y regla de diseño en función de los objetivos de negocio.	Desarrollos con alta calidad.
Casos de uso como pieza esencial del desarrollo.	Los casos de uso se utilizan como herramienta para dar guías de desarrollo desde los requisitos hasta el código. Los elementos que se definen con casos de uso son; arquitectura, estimación, seguimiento del proyecto, pruebas funcionales, trazabilidad (certidumbre en el desarrollo), manual de usuario e interfaces de usuario.	MiPyMEs. Uso del ciclo de vida como desarrollo. Desarrollos que requieren diagramas o casos de uso. Software que requiere trazabilidad.
Ciclo de vida iterativo y evolutivo.	Uso del ciclo de vida incremental e iterativa, no se utiliza un desarrollo basado en el modelo de cascada, permite realizar aproximaciones que ofrecen una viabilidad al proyecto, se mitigan riesgos asociados a los problemas de definición de riesgos o diseño.	MiPyMEs. Para proyectos que se requiere de tener resultados rápidos. Desarrollos que mitiguen riesgos. Requerimientos cambiantes.
Estimación basada en puntos de función adoptados.	Se utilizan métricas para establecer el tamaño y complejidad del producto, se definen puntos de función por elemento, módulo o interface de desarrollo, el cálculo del tamaño de un producto depende de la documentación, la experiencia y el know-how de los desarrolladores.	PyMEs. En FS donde el tamaño del proyecto es primordial conocerlo. Donde se tienen métricas promedio para el desarrollo Control de costos exhaustivo.
Guiar las acciones por valor.	Se aplica en desarrollos donde deben medirse los productos respecto a los objetivos económicos, las prácticas debe alinearse a estos objetivos. Tener mediciones cada vez más precisas de las técnicas aplicadas, cual práctica tiene mejor retorno de inversión, que partes del código son la más refactorizadas, que partes producen mayor valor.	MiPyMEs. En desarrollos donde se requiere un control de los costos. En proyectos pequeños y controlables. En donde se requiere alinear el producto con objetivos económicos.

3.9 Administración de proyectos en fábricas de software.

De acuerdo a la definición de una FS; “es un centro de desarrollo e implementación de productos de software que ejecuta actividades como un proceso industrial, definible, repetible y medible bajo un entorno de alta productividad” Greenfield et. al. (2004). Se requieren técnicas especiales para administrar los proyectos en una FS, en (Lopez M., 2013) se define un modelo estandarizado de servicio, en la figura 3.18 se muestra el modelo, donde sobresalen los temas de planeación, gestión, ejecución y cierre de proyectos.

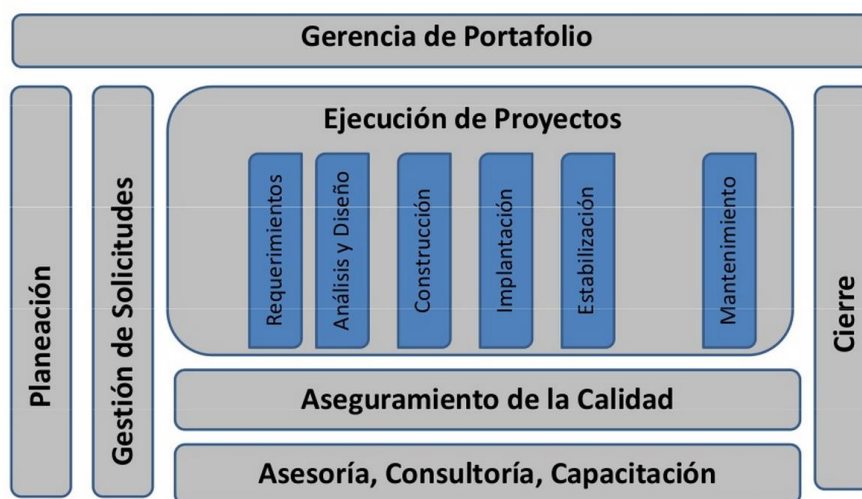


Figura 3.18. Modelo estandarizado de servicio en una FS.
Fuente: (Lopez M., 2013).

Las actividades de la gerencia se centran en las siguientes:

1. Garantizar los resultados esperados.
2. Administración de los recursos.
3. Aplicación de la metodología de administración de proyectos del PMI.
4. Negociación los planes de proyecto y control de cambios.
5. Seguimiento y monitoreo.

La planeación requiere de actividades como; la preparación del modelo operativo para realizar el proyecto, la aprobación de procedimientos y metodologías, la adopción de

herramientas, la consolidación del equipo de trabajo y la personalización de plantillas. Los documentos y artefactos que formalizan la planeación son: las solicitudes del cliente, los entregables, la ejecución del proyecto, los compromisos, los controles de cambios, comunicados, defectos encontrados y finalmente la facturación. El gerente del proyecto debe centrar sus actividades en la gestión del tiempo (tareas, cronograma, hitos), la gestión del alcance (árbol de actividades, cumplimiento de la entregas), la gestión de la calidad (de los artefactos y entregables, establecimiento de métricas, auditoría), la gestión de las comunicaciones (mecanismos, tiempos de respuesta, comités), la gestión de los recursos humanos (responsabilidades, definición de actividades y funciones, reclutamiento, perfiles), la gestión de costos, la gestión de riesgos (identificar, matriz, priorización). Es importante establecer las metodologías necesarias para la estimación de esfuerzo, para el desarrollo y parametrización del software, para la gestión de incidentes y defectos y para la solución de incidentes. Para estas actividades existen generalmente herramientas que permiten simplificar la planeación.

En la gestión de solicitudes en una FS se realizan las actividades en la elaboración de una Plan del proyecto en donde se define el alcance del proyecto, se cuantifican recursos, se define los costos y tiempos estimados de implementación.

En la ejecución de proyectos debe realizarse las actividades siguientes de acuerdo a las fases de:

1. Requerimientos. Conceptualización de los dominios de la solución, especificación de requerimientos funcionales y no funcionales, prototipo y plan de pruebas.
2. Análisis y diseño. Estructura de datos a utilizar, arquitectura de la solución, interfaces y componentes, mecanismos de integración con sistemas externos, guía para la implementación del proyecto.
3. Construcción. Elaboración de la solución establecida, parametrización, adaptación o integración, manuales técnicos y de usuario, pruebas.
4. Implantación. Se coloca en producción la solución implementada, garantizar el funcionamiento correcto, realizar la configuración y parametrización en el ambiente de producción y apoyar por un tiempo definido el correcto funcionamiento de la solución.

5. Estabilización. Revisión continua del correcto funcionamiento de la solución, realizar el monitoreo del desempeño, solucionar incidentes, actualización de la documentación.
6. Mantenimiento. Realizar actividades de actualización de la solución, mantenimiento preventivo y evolutivo, servicio de soporte, aseguramiento de la calidad en el mantenimiento.

El proceso de aseguramiento de la calidad requiere de realizar una serie de tareas durante el proyecto de la cuales sobresalen: revisión transversal de todos los procesos implementados, revisión transversal de todos los entregables; documentos, artefactos y componentes de software, aseguramiento de la calidad en el proyecto entregado; pruebas funcionales, de carga, integridad, interoperabilidad y desempeño, emisión de certificados de calidad del producto/servicio entregado.

Otra actividad importante en la FS, son las actividades de consultoría, asesoría y capacitación que se realizan en los departamentos de; tecnología, capacitación, entrenamiento, uso de TI. Todo esto junto con la solución construida se realizan actividades de diagnóstico y análisis de la madurez tecnológica.

Finalmente se desarrolla el proceso de cierre, en la cual se desarrollan actividades como; cierre del contrato una vez finalizado todos los planes, realizar cesión de derechos de autor, elaborar un informe final de cierre contractual y entrega final del archivo del contrato.

En una FS es importante dejar constancia de las lecciones aprendidas y recomendaciones, dejando con detalle; fechas, plazos, definiciones, técnicas, priorización de proyectos, intervención de entidades externas para supervisión y estandarización y crear una base de información que permita consultarla para posteriores proyectos.

3.10 Modelos y metodologías para el desarrollo de software.

El vertiginoso desarrollo de los programas de cómputo ha provocado que las empresas y sobre todo las PyMEs (que no tienen muchos recursos para invertir), implementen mejoras para el desarrollo de software. Un programa de mejora de procesos en PyMEs que tiene como propósito el desarrollo de software, requieren de prácticas eficientes que tengan en cuenta la flexibilidad, con una estructura plana, un estilo de gestión libre y una administración basada primordialmente en procesos (Pino, 2010). El desarrollo de la mejora se inicia a través de la definición del ciclo de vida de un producto o servicio. La mejora se fundamenta en el trabajo en equipo, en la colaboración y en la comunicación efectiva para lograr la participación dinámica entre los participantes del equipo de mejora y en general de toda la organización.

Un desarrollo de software mejorado se caracteriza por estar compuesto de procesos intensivos en conocimiento, en los cuales se deben considerar de manera explícita las tareas, roles, objetos, relacionados con la gestión del conocimiento. Estos procesos tienen actividades caracterizadas por requerir experiencia y generar conocimiento, que es fundamental para la toma de decisiones. En la implementación de este tipo de herramienta se debe tener en cuenta que en la mejora de los procesos debe existir una transición del aprendizaje individual al aprendizaje organizacional, mediante la disposición y utilización de estos activos generados en su ejecución, por la tanto es importante la gestión de los activos tangibles mediante el uso de la tecnología.

La metodología para la implementación de una mejora de proceso de software tiene los siguientes pasos:

1. Caracterizar el proyecto: descripción de las características del proyecto y su medio ambiente.
2. Establecer metas y medidas cuantificables: alcance de mejoras continuas mediante el establecimiento de procesos y mediciones delgados.
3. Seleccionar los modelos de los procesos y métodos: los modelos de los procesos deben ajustarse al medio ambiente, procesos, herramientas y métodos ya existentes.

4. Ejecutar los procesos, reunir y validar los datos: registrar las medidas definidas en b) cuando se esté ejecutando el proceso definido en c).
5. Analizar los datos reunidos y recomendación de mejoras: es necesario ubicar las causas de los problemas para tomar acciones correctivas que permitan un proceso ágil de software.
6. Empaquetar y almacenar la experiencia: la experiencia es almacenada para ser consultada en futuras mejoras.

3.10.1 Desarrollo de software dirigido por modelos (MDD) versus Fábricas de Software (FS).

En Piatini et. al. (2010), se describe que el desarrollo de software se ha centrado en el Desarrollo Dirigido por Modelos (MDD, Model-Driven Developmnet), se utiliza el MDA (Model Driven-Architechture) como plataforma. El enfoque de FS ha resultado ser más apropiado para aplicaciones que son similares y que van a atacar problemas en el mismo dominio o área de conocimiento (permite crear líneas de producción). Hay muchas desventajas sobre el enfoque MDA ya que solo propone el modelo y no espeífica con detalle el como se realizará tal o cual actividad, siendo el enfoque de FS mejores en este punto ya que se basa en las mejores prácticas. En la tabla 3.5 se muestran la ventajas y desventajas de ambas propuestas.

Tabla 3.5. Ventajas y desventajas del MDA vs FS.
Fuente: (Piatini V. & Garzas P., 2010).

Desarrollo basado en	Ventajas	Desventajas
Dirigido por Modelos.	Los desarrollos basados en MDA en el corto plazo tienen mejores costos. Facilita la tarea de un desarrollo con calidad. Se tiene más conocimiento sobre los modelos. Se adaptan en forma fácil a herramientas IDE. Existen lenguajes como el UML que se adaptan fácilmente a cualquier modelo. Se logra tener un control sobre la madurez de procesos.	Costo inicial para adoptar el modelo Debe tener un conocimiento muy profundo del modelo. Existen modelos que no se acompañan con una metodología lo que complica el panorama general del desarrollo.
Fábricas de Software.	Proporcionan una guía metodológica completa	Se centran en la fase de desarrollo y

	Se tiene certidumbre sobre los componentes y elementos que deben ser desarrollados. Se utilizan las mejores prácticas y permite agilizar el desarrollo. Se ha adoptado por las grandes desarrolladoras de software que están impulsando el conocimiento y uso de este enfoque.	generalmente carecen de documentación. Requiere un tiempo mayor de adopción y generación de calidad Cuesta un poco más de tiempo llegar a un nivel de madurez.
--	---	---

Los aspectos que más se deben de tomar en cuenta al implementar uno u otro enfoque son los siguientes:

1. Tiempo empleado en el desarrollo. MDA ahorra tiempo, sin embargo la FS automatiza el proceso.
2. Mantenimiento de las aplicaciones. MDA yudo a este proceso en forma más sencilla.
3. Prerrequisitos para adoptar la estrategia. MDA requiere de más esfuerzo de adopción que una FS.
4. Generación de software. Ambos enfoques se centran en la generación con calidad por los que en este tema no hay gran diferencia.
5. Metodologías de desarrollo. El MDA y MDD no define ningún ciclo de vida por lo que adaptarlo al desarrollo ágil es más simple. Las FS también ofrecen las mejores prácticas que permiten utilizar metodologías conocidas.

En ambos enfoques existen retos y desafíos que deben considerarse antes de utilizar un MDA o una FS, estos son:

1. Las habilidades y experiencias de las personas que intervienen en el proyecto.
2. Se requiere madurez en los estándares de MDA y en herramientas.
3. La interoperabilidad entre herramientas de desarrollo.
4. Necesidad de mejorar el soporte para el MDD.
5. Existen dificultades al iniciar pero en poco tiempo puede beneficiarse.

3.11 Tendencias en fábricas de software sobre el desarrollo móvil.

En el informe de (Gallegos, Grandet, & Pavel, 2014) se describe que para el año 2013 en México los ramos principales de desarrollo de software fueron los siguientes, ver figura 3.19.

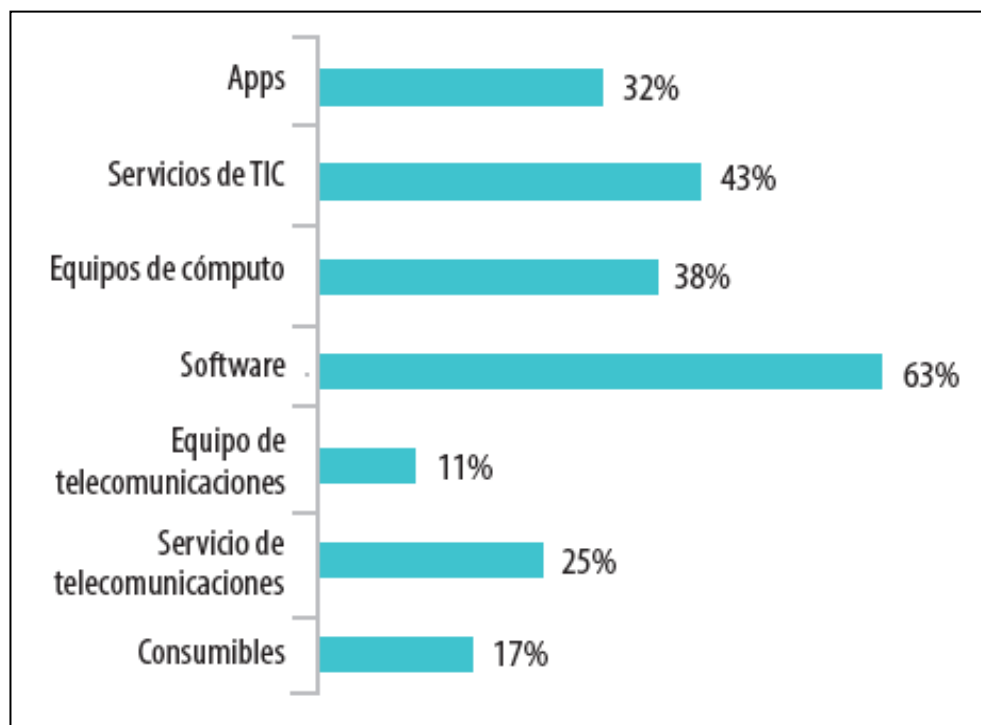


Figura 3.19. Principales sectores dentro de la Tics que más se desarrollan en PyMEs.
Fuente: (Gallegos, Grandet, & Pavel, 2014).

Como se observa las empresas PyMEs tienen como principal sector entre la TICs el desarrollo de software a la medida 63%, le siguen los servicios TIC 43%, el soporte y venta de equipos de cómputo 38% y el desarrollo de APPs con un 32%, entre otros. En el estudio de la AMITI- Infotec (2013) se identificaron más de 200 iniciativas para el uso difusión y desarrollo de APPs. Los factores que fomentan este sector son:

1. Marco legal y normatividad,
2. Plataforma tecnológica,
3. Participación ciudadana,
4. Conocimiento técnico y
5. Recursos económicos.

Los factores que incrementan la demanda y por lo tanto fomentan el uso de aplicaciones móviles son:

1. Solución de necesidades del cliente,
2. Costo,
3. Conveniencia,
4. Capacitación y
5. Comunicación.

El desarrollo de aplicaciones móviles es de las prioridades más importantes en PyMEs, de aquí la necesidad de contar con nuevos modelos y metodologías que aporten nuevas tecnologías y herramientas que permitan competir a las empresas en esta industria.

Capítulo 4. Proceso metodológico para el diseño de la metodología MDSIC-M.

4.1 Identificación de variables independientes.

Como se revisó en los capítulos anteriores las metodologías ágiles y adelgazadas para el desarrollo de aplicaciones móviles todavía está en desarrollo, se han documentado muchas experiencias a nivel internacional, pero falta censar el mercado nacional y saber cuáles son las necesidades de las PyMEs que desarrollan APPs en México.

El objetivo de esta tesis es medir y evaluar indicadores en las PyMEs que desarrollan aplicaciones móviles en la región, para este objetivo se realizó un estudio transversal (métodos y metodologías actuales), cuya naturaleza de la investigación es: cuantitativa, de campo, cuasi-experimental y explicativa (Hernández S., Fernández C., & Baptista L., 2010). Se realizó un análisis síntesis de los diferentes modelos y metodologías encontradas en la literatura, de este análisis se obtuvieron las variables que son necesarias censar y proponer en una metodología ad-hoc.

En Cendejas et al. (2014) se realizó un estudio de las variables que fueron censadas en la industria de TI en México. De acuerdo a este objetivo en la figura 4.1 se muestra el proceso metodológico para el diseño de la metodología MDSIC-M (Modelo de Desarrollo de Software Integral Colaborativo para Móviles).

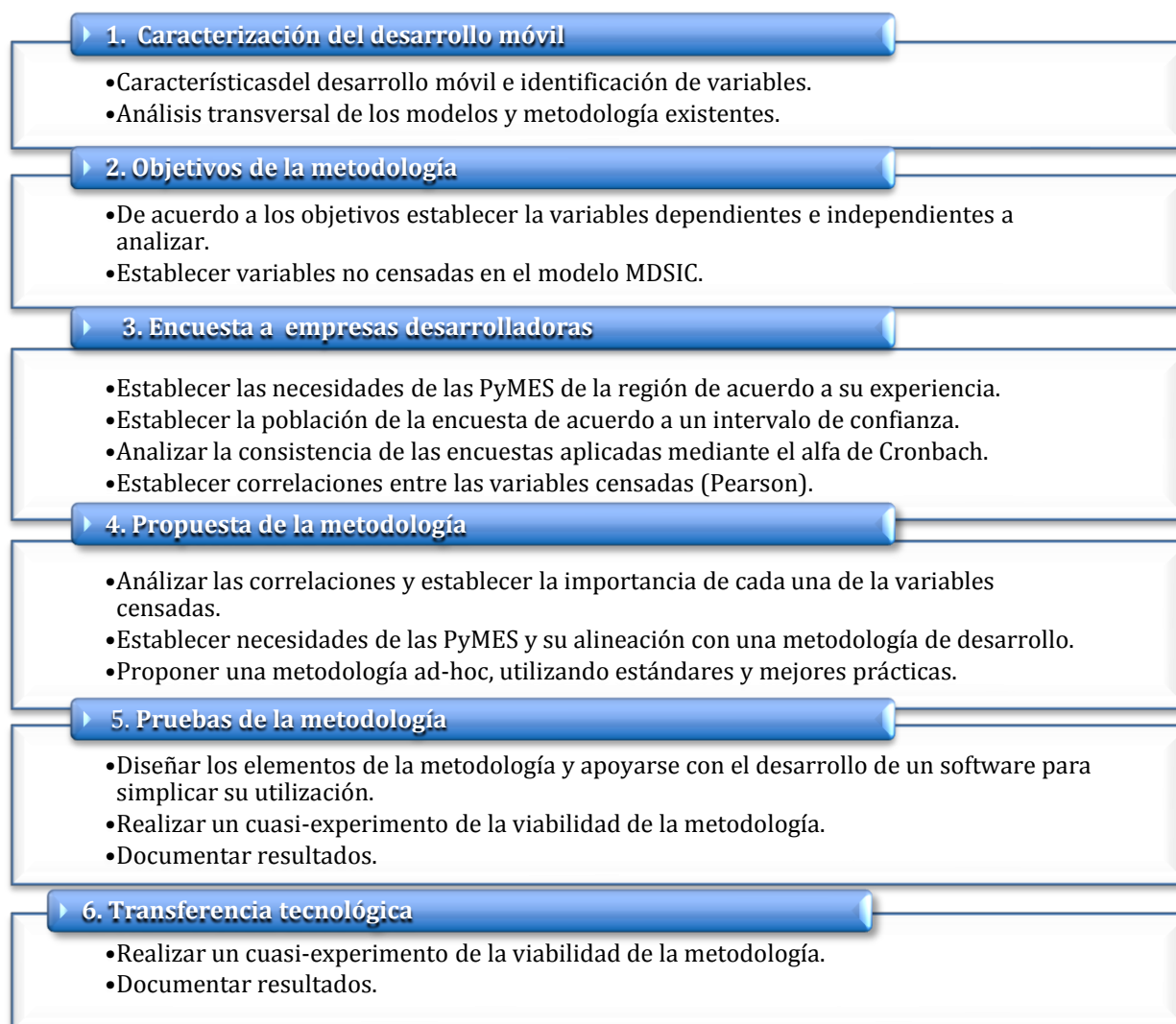


Figura 4.1. Proceso metodológico para el diseño de MDSIC-M.
Fuente: elaboración propia.

Utilizando las características de las metodologías y modelos analizados se obtuvieron las preguntas para censar y determinar la importancia del desarrollo de aplicaciones móviles en PyMES, utilizando los ejes de la planeación, costos, riesgos, desarrollo con calidad (Rai, Song, & Troutt, 1998) y el establecimiento de métricas (Saraiva, de Franca, Soares, Filho, & de Souza, 2015). En la tabla 4.1 se muestran las variables dependientes de cada uno de los objetivos en específico derivados del objetivo principal de la tesis, los elementos no marcados son variables desarrolladas y analizadas en el MDSIC, las variables marcadas son las que se censarán para conocer como impactan el desarrollo de una APP.

Tabla 4.1. Variables independientes que se necesitan censar para ser evaluadas con MDSIC-M.

Fuente: elaboración propia.

Objetivo	Específicos	Variables independientes
Diseñar una metodología para el desarrollo de software en dispositivos móviles que contribuya a la calidad en fábricas de software (PyMEs), fortaleciendo el desarrollo de aplicaciones en la región Centro Occidente del país.	Realizar una caracterización del desarrollo de aplicaciones móviles para conocer sus componentes.	Tipos de tecnologías donde se aplica. ✓ Tipo de conectividad. ✓ Uso de tendencias en el desarrollo; e-Business, e-Commerce, e-education, e-Learning, e-Gamming. - Uso del cómputo distribuido. - Uso del cómputo paralelo. ✓ Uso de un framework de desarrollo. ✓ Tipos de servicios de red utilizados en el desarrollo de una APP.
	Investigar las fábricas de software que desarrollan aplicaciones móviles en los clústeres de la región Centro Occidente.	Tipo de proceso de desarrollo Uso del desarrollo dirigido por modelos (DSDM ó MDA). Uso del desarrollo dirigido por lenguajes, plataformas y componentes (SF/IDE). ✓ Tiempo de desarrollo. - Tipo de mantenimiento en las aplicaciones. ✓ Formas de generación del software. ✓ Metodología empleada. ✓ Tipo de interoperabilidad entre componentes. ✓ Nivel de madurez. - Estándares de calidad. ✓ Modelos de Global-Deliver (tiendas de APPs). ✓ Outsourcing; nearshore (programadores de la región) y offshore (programadores externos).
	Investigar los estándares de calidad utilizados en la industria del desarrollo del software en la región Centro Occidente.	Indicadores de facilidad, usabilidad, confiable, desempeño y soporte (FURPS, McCall para calidad). ✓ Métricas; análisis, diseño, desarrollo y pruebas. - Lenguajes utilizados. ✓ Aplicación de Moprosoft (NOM). ✓ Aplicación de estándares de calidad.
	Desarrollar las fases	Etapas en el desarrollo.

	complementarias al modelo MDSIC para el desarrollo Móvil.	Uso del desarrollo ágil. Planeación del proyecto. ✓ Tiempo del proyecto. ✓ Control de riesgos. ✓ Costos. ✓ Control de tareas. Indicadores de calidad (KPI) y KPA (por áreas).
	Desarrollar un esquema para validar la calidad en el desarrollo móvil en la región centro-occidente.	✓ Métodos para prueba de software. ✓ Modelo de medición de la calidad. Quality Assurance (QA). ✓ Herramientas de gestión.
	Analizar los resultados y conclusiones de aplicar la metodología en desarrollo móvil.	✓ Seguimiento del proyecto. Modelo colaborativo. ✓ Impacto en los costos.

En relación con las variables detectadas y que son importantes para una metodología de desarrollo de software, es necesario consultar a las PyMEs para conocer su importancia entre las FS de la región. Utilizando una encuesta se podrá conocer la percepción de los desarrolladores de aplicaciones móviles, estableciendo con esta la importancia de cada una de las variables independientes, construyendo así una metodología que ayude a las PyMEs a competir en el mercado de las APPs.

4.2 Análisis de la población.

Para determinar el número de empresas probables a censar, se revisaron varias bases de datos de empresas o instituciones dedicadas a la promoción y difusión de las PyMEs en México, de estas se obtuvo la población de la encuesta. Se revisaron las bases de la AMITI, Cendejas et al. (2013), la Sección Amarilla (Sección Amarilla, 2014), el Directorio de empresas en México (infoinfo, 2015) y en el Sistema de Información Empresarial de México (SIEM, 2014), de la Secretaría de Desarrollo Económico (SEDECO). Se utilizaron las etiquetas de búsqueda (Tag), siguientes:

1. “Desarrollo de software”,
2. “Desarrollo móvil”,
3. “Empresa de TI para desarrollo” ,
4. “Desarrollo en radiocomunicación”,
5. “Desarrollo de aplicaciones” e
6. “Informática”.

De acuerdo a la región definida por la CANIETI se determinó la población de la encuesta (económicamente activa), que potencialmente pueden ser PyMEs, ver Tabla 4.2.

Tabla 4.2. Población económicamente activa en la región centro-occidente de México.
Fuente: elaboración propia

No.	Siglas	Cede centro-Occidente	Población Económicamente Activa (PEA)*	%
1	Ags	Aguascalientes	510,541	6%
2	Col	Colima	342,702	4%
3	Gto	Guanajuato	2,412,886	27%
4	Jal	Jalisco	3,369,238	37%
5	Mich	Michoacán	1,896,174	21%
6	Nay	Nayarit	529,436	6%
Total Región			9,060,977	100%
Total PEA en México			52,108,400	
Porcentaje Nacional			18.03%	

*Datos (INEGI, 2014).

Los datos arrojados de acuerdo a las etiquetas de búsqueda se observan en la Tabla 4.3 A), B) y C).

Tabla 4.3. Análisis de la población para aplicar la encuesta de acuerdo a las etiquetas de búsqueda, A) Sección Amarilla, B) infoinfo y C) SIEM 2010. Fuente: elaboración propia.
Consulta de Tags Sección Amarilla (2014).

Edo.	Informática	Desarrollo software	TI	Total	%
Ags	156	18	5	179	7%
Col	163	17	1	181	7%
Gto	581	38	10	629	25%
Jal	790	75	46	911	36%
Mich	462	20	3	485	19%
Nay	128	3	3	134	5%
171				2519	

A) Resultado de búsqueda en la Sección Amarilla

Directorio de empresas infoisinfo (2014)

Edo.	Informática	Desarrollo software	TI	Total	%
Ags	410	23	7	440	6%
Col	695	14	6	715	10%
Gto	2362	32	15	2409	34%
Jal	2292	14	19	2325	33%
Mich	1012	6	14	1032	14%
Nay	211	4	1	216	3%
		93		7137	

B) Resultado de búsqueda en el directorio de empresas infoisinfo.

Sistema Información Empresarial Mexicano, SEDECO SIEM (2014)

Edo.	Empresas Registradas	Informática	Desarrollo software	TI	Total	%
Ags	5,242	139	5	64	208	4%
Col	1,622	59	12	38	109	2%
Gto	38,289	698	15	444	1157	20%
Jal	51,702	1,783	59	235	2077	36%
Mich	48,488	1,233	24	791	2048	35%
Nay	8,975	169	5	49	223	4%
	154,318		120		5822	

C) Resultado de búsqueda en empresas registradas en el SIEM-SEDECO actualizado al año 2010.

De los resultados obtenidos en la búsquedas anteriores se cálculo la muestra confiable de la población para realizar la encuesta basada en un intervalo de confianza del 95% y 97% y un margen de error del 5% (Kothari, 2004), de acuerdo a la formula mostrada en la figura 4.2 siguiente se obtuvieron los tamaños de la muestra (n); ver figura 4.2 y tabla 4.4.

$$n = \frac{z^2 \cdot p \cdot q \cdot N}{(e^2 (N - 1)) + (z^2 \cdot p \cdot q)}$$

Donde:

N = Tamaño de la población a censar (resultado de las búsquedas).

z = Área bajo la curva normal con intervalo de confianza: 95% (1.96) o 97% (2.17).

$q = 1 - p = 0.5$ Proporción de la muestra para una n máxima con precisión mayor.

e = margen de error de la muestra calculada (5%).

Figura 4.2. Cálculo de la muestra n con la mayor precisión, un intervalo de confianza y una margen de error.

Fuente: (Kothari, 2004).

Tabla 4.4. Cálculo tamaño de la muestra con un intervalo de confianza del 95% y 97% y un error del 5%.

Fuente: elaboración propia

Poblaciones posibles de la encuesta con un margen de error del 5%.			
Intervalo de confianza	Sección Amarilla (N=171)	infoisinfo(N=93)	SEDECO (N=120)
97%	$n = 126$	$n = 78$	$n = 96$
95%	$n = 119$	$n = 75$	$n = 92$

Utilizando las variables para censar de acuerdo a los objetivos de la tesis (punto 3 de la metodología propuesta), se elaboró una encuesta en línea para conocer la importancia de cada una de estas entre los desarrolladores de la región. Se utilizó como estrategia la participación de los clúster de TI de cada estado para censar el estatus del desarrollo móvil entre las empresas miembros, además se utilizaron las redes sociales para contactar aquellas que no pertenecen a un clúster de TI.

4.3 Diseño de la encuesta.

Los elementos de la encuesta que surgen de la investigación y la caracterización de las aplicaciones móviles, se censaron en 37 preguntas, las cuales 22 se basan en la escala de Likert (Totalmente de acuerdo (100%), medianamente de acuerdo (75%), indiferente (50%), en desacuerdo (25%) y totalmente en desacuerdo (0%)) y 15 preguntas abiertas. Se realizaron entrevistas a las empresas mediante el apoyo de los Clústeres de tecnologías y redes sociales.

De acuerdo a las variables que se requiere censar, se generó la encuesta que ayudó a conocer la importancia de cada una de estas entre los desarrolladores de la región. En la figura 4.3 se muestran el diseño de la encuesta y un ejemplo de la encuesta realizada en línea (e-encuesta, 2014):

Desarrollo de Aplicaciones Móviles en México

Esta encuesta esta dirigida a los desarrolladores de aplicaciones para dispositivos móviles. El objetivo es caracterizar, investigar y analizar este tipo de desarrollo en México. Agradecemos su valioso tiempo para contestarla.

Tiempo aproximado de la encuesta 15 minutos.



Instituciones participantes

Nota: Los datos solicitados son para fines de estudio de la encuesta y de ninguna manera se utilizarán con fines de lucro o marketing. Agradecemos su ayuda. Política de privacidad de e-encuesta.com.

1. ¿Cuál es el nombre de su empresa?

2. ¿Estado de la república Mexicana donde se ubica su empresa?

Seleccione

3. ¿Cuántos empleados tiene tu empresa (aproximadamente)?

Seleccione

4. ¿Cual es su nombre y puesto que ocupa?

Nombre

Puesto

5. ¿Cuál es su correo electrónico? (necesario en caso de que desee información de los resultados de la encuesta)

0% Completado



Siguiente >

Figura 4.3. Encuesta realizada para censar la industria del desarrollo móvil en la región Centro Occidente.
Fuente: elaboración propia.

La encuesta se divide en los elementos siguientes: descripción de la encuesta, datos generales, caracterización de los dispositivos móviles, PyMEs en los clústeres de TI ventajas y desventajas, estándares de calidad en el desarrollo de aplicaciones móviles, fases complementarias al modelo MDSIC para el desarrollo móvil, análisis de resultados y conclusiones de aplicar una metodología en el desarrollo móvil y al final un agradecimiento por la participación.

La encuesta incluye las preguntas siguientes: 1) datos de la empresa, 2) estado donde se ubica, 3) número de empleados, 4) puesto que ocupa, 5) correo electrónico del encuestado, 6) tipo de App que desarrolla, 7) uso de métodos a la vanguardia, 8) reutilización de componentes, 9) seguridad en la interconexión, 10) servicios, 11) desarrollo dirigido por modelos, 12) desarrollo para una plataforma, 13) mantenimiento usando una metodología, 14) interoperabilidad, 15) modelo de entrega global, 16) uso de programadores locales, 17) uso de programadores externos, 18) hace outsourcing, 19) Uso de indicadores FURPS, 20) usa métricas para el análisis, 21) usa métricas para el diseño, 22) usa métricas para el desarrollo, 23) usa métricas para las pruebas, 24)

Metodología MDSIC-M para desarrollo de aplicaciones móviles en PyMEs

120

usa alguna metodología para la calidad, 25) asigna recursos para la calidad, 26) uso de estándares, 27) uso de un método o pasos para la calidad, 28) uso de la planeación, 29) usa el control de riesgos, 30) costo y presupuesto proyectado, 31) uso de procesos, 32) uso de herramientas de gestión, 33) cual es su estrategia de desarrollo, 34) conoce metodologías ágiles, 35) pertenece a un clúster de TI, 36) uso de herramientas colaborativas y 37) tiempo que lleva desarrollando APPs.

En la tabla 4.5 A), B), C), D), E) y F) se muestran la preguntas realizadas de acuerdo a los objetivos del estudio y el conjunto de valores de cada una.

Tabla 4.5. Preguntas realizadas en la encuesta, 37 preguntas; 15 preguntas abiertas y 22 en escala de Likert.
Fuente: elaboración propia.

Nombre de la encuesta: **"Desarrollo de aplicaciones móviles en México"**
Descripción: Esta encuesta esta dirigida a los desarrolladores de aplicaciones para dispositivos móviles. El objetivo es caracterizar, investigar y analizar este tipo de desarrollo en México. Agradecemos su valioso tiempo para contestarla.
Tiempo aproximado para contestar la encuesta 15 minutos
[Los datos solicitados son para fines de estudio de la encuesta y de ninguna manera se utilizará la información con fines de lucro o marketing. Agradecemos su ayuda. Política de privacidad de e-encuesta.com](#)

Objetivo	No. Pregunta	Pregunta	Respuestas Correspondencia			
Datos Generales	1	¿Cuál es el nombre de su empresa? (opcional)	Abierta			
	2	¿En que estado de la republica Mexicana se ubica su empresa?	Estados			
	3	¿cuántos empleados tiene su empresa (aproximadamente) ?	1-20	20-100	100-500	Más de 500
	4	¿Cuál es su nombre y puesto que ocupa ?	Nombre:	Puesto:		
	5	¿cuál es su correo electrónico? (Solo en caso que se desee información del resultado de la encuesta)				

A) Preguntas de datos generales.

Caracterización de dispositivos móviles	6	¿Que tipo de aplicaciones móviles desarrollas?	Para procesos de negocio (e-bussinnes) E-Bussinnes	Comercio electrónico E-Commerce	Educación o aprendizaje en línea E-Learning	Juegos en una aplicación o en línea E-Gaming	Aplicaciones en general para una plataforma especifica Mobile-app	Desarrollo Web adaptable a la resolución Web Responsive
	7	En el desarrollo de aplicaciones móviles, ¿usted siempre sigue las tendencias de vanguardia en el desarrollo?	Totalmente de Acuerdo Tendencias en el desarrollo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo	
	8	En el desarrollo de aplicaciones móviles, ¿usted reutiliza componentes y código?	Totalmente de acuerdo Reutilización de código	De acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo	
	9	En el desarrollo de aplicaciones móviles, ¿usted consideraría la seguridad en la interconexión y el uso de la aplicación como un tema prioritario?	Totalmente de Acuerdo Seguridad	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo	
	10	¿Que tipo de servicios se utilizan más en las aplicaciones que usted desarrolla?	Redes sociales Servicios	Descarga de archivos	Correo electrónico	Navegación (web)	Multimedia (video, audio, animación)	Otros servicios; impresión, localización y búsqueda

B) Preguntas de caracterización de una APP.

PyMES en Clusters ventajas y desventajas	11	En el desarrollo de una aplicación móvil, ¿usted desarrolla dirigido por algún Modelo?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			DSDM (Desarrollo de Software Dirigido por Modelos) o MDA (Model Driven Architecture)				
	12	En el desarrollo de una aplicación móvil, ¿usted desarrolla dirigido por un lenguaje o plataforma o componentes?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	13	¿Realizaría usted el mantenimiento a las aplicaciones desarrolladas, utilizando alguna	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Fábrica de Software				
	14	¿Siempre considera la interoperabilidad entre plataformas al desarrollar un software móvil?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Mantenimiento de Software				
			Interoperabilidad				
	15	¿Utilizaría usted algún modelo de entrega global (tiendas y/o plataformas de compra-venta) para su aplicación móvil?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Global Delivery				
	16	¿Utilizaría la contratación de programadores a nivel local como ventaja competitiva?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Nearshore				
	17	¿Utilizaría usted programadores de otros estados o países como ventaja competitiva?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Offshore				
	18	¿Su empresa hace outsourcing para otras?	Si	No			
			Outsourcing				

C) Preguntas sobre las FS en los clústeres de TI.

Estándares de calidad en el desarrollo de aplicaciones móviles	19	Al desarrollar aplicaciones móviles considera usted la Fiabilidad, Usabilidad, Confiabilidad, Desempeño	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			FURPS				
	20	¿Utiliza usted métricas (medida estandarizada propia) para medir el desempeño en el análisis de una aplicación?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	21	¿Utiliza métricas para medir el desempeño en el diseño de una aplicación?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Métricas de Análisis				
	22	¿Utiliza métricas para medir el desempeño en el desarrollo de una aplicación?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Métricas de Desempeño				
	23	¿Utiliza métricas para medir el desempeño en las pruebas de una aplicación?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Métricas de Desarrollo				
			Métricas para pruebas				
	24	¿En el desarrollo de aplicaciones móviles utiliza alguna metodología para garantizar la calidad del software?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Quality Assurance				
	25	¿Asignaría recursos para un equipo orientado a la calidad del software?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Quality Assurance Team				
	26	¿En su experiencia el uso de estándares de calidad le han permitido obtener una ventaja competitiva?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
			Uso de estándares de calidad				
	27	¿En su caso para garantizar la calidad de un desarrollo, que método, pasos o etapas realiza?	Abierta				
			Metodología de calidad que se realiza en la PyME				

D) Preguntas sobre estándares de calidad en el desarrollo de APPs.

Fases complementarias al modelo MDSCI para el desarrollo Móvil	28	¿Cuándo desarrolla una aplicación móvil siempre toma en cuenta el tiempo proyectado en la	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	29	¿En las diferentes fases del desarrollo de la aplicación, usted hace control de riesgos para	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	30	¿El desarrollo de la aplicación siempre esta apegado al costo proyectado y se finaliza cuando ya no hay presupuesto?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	31	¿Usted realizaría el desarrollo de una aplicación móvil basado en procesos y/o tareas?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	32	¿Usted utilizaría alguna herramienta de gestión para un desarrollo ágil?	Totalmente de Acuerdo	Medianamente de acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
	33	¿Cómo le da seguimiento a un desarrollo móvil?, ¿Cuál es su estrategia?	Abierta				

E) Preguntas sobre las fases en el desarrollo de software.

Análisis de resultados y conclusiones de aplicar una metodología en el desarrollo móvil	34	¿Conoce alguna metodología ágil (desarrollo rápido de aplicaciones) ?, ¿cuál?	Abierta			
	35	¿Su empresa pertenece a un cluster de tecnología ?	Si	Conocimiento de metodologías ágiles	No	
	36	¿Usted considera importante la colaboración entre los miembros del proyecto?, ¿Ha utilizado alguna	Si	Pertenencia a un Cluster	No	
	37	¿Desde hace cuanto tiempo desarrolla su empresa aplicaciones para dispositivos móviles?	Abierta	Uso de herramientas colaborativas	número de años	

Agradecemos su tiempo y en breve le enviaremos información sobre las diferentes tendencias del desarrollo móvil y nuestra propuesta MDSIC-Móvil

*Política de privacidad de e-encuesta.com

F) Preguntas finales de la encuesta.

4.4 Resultados de la encuesta.

Utilizando la encuesta se censó el desarrollo de las APPs en las PyMEs en la región centro occidente de México, se realizaron entrevistas a las empresas mediante el apoyo de los clústeres de tecnologías (CLUSTERTIM, 2014), (CIAT, 2014), (AIMSI, 2014) y las redes sociales (uso de Facebook y Twitter); 70 empresas contestaron y solo 38 de estas empresas desarrollan aplicaciones móviles (54 %).

Como método de validación para los resultados de la encuesta se calculó el alfa de Cronbach (α) de acuerdo a Hernández S. et. al. (2010); es un coeficiente que sirve para medir la fiabilidad de una escala de medida, se basa en la varianza de los elementos censados en la encuesta. Una parte importante del diseño de la encuesta es basarse en preguntas orientadas al positivismo y en la escala Likert que permite censar el porcentaje de afinidad de cada una de los elementos censados. La formulas utilizadas para el cálculo del α se muestra en la figura 4.4.

Alfa de Cronbach (α) a partir de las varianzas:

$$\alpha = \left[\frac{k}{k-1} \right] \left[1 - \frac{\sum_{i=1}^k S_i^2}{S_t^2} \right]$$

Donde:

S_i^2 es la varianza del ítem i ,

S_t^2 es la varianza de los valores totales observados y

k es el número de preguntas o ítems.

Figura 4.4. Formula para calcular alfa de Cronbach a partir de las varianzas.
Fuente: Hernández S. et. al. (2010).

De acuerdo a los datos censados en la encuesta, se realizó el análisis de alfa de Cronbach para determinar la coherencia interna y el coeficiente resultó en **0.862** (para la preguntas con escala de Likert), esto demuestra buena consistencia en las respuestas. En la figura 4.5 se muestran el cálculo de α utilizando las 38 respuestas de los desarrolladores, el rango de las repuestas corresponde a: totalmente en desacuerdo=1, medianamente en desacuerdo=3,

indiferente=5, medianamente de acuerdo=7 y totalmente de acuerdo=9. Se colocan las respuestas de las 22 preguntas con escala de Likert y se calcula varianza para cada ítem i y finalmente se obtiene el valor de α .

Análisis de Alfa de Cronbach de acuerdo a las varianzas de las preguntas con escala de Likert

Pregunta		2.2	2.3	2.5	2.6	2.7	2.8	2.9	2.10	2.11	3.1	3.2	3.3	3.4	3.5	3.6	3.7	3.8	3.10	3.11	3.12	3.13	3.15		
Item	Encuesta	R7	R8	R10	R11	R12	R13	R14	R15	R16	R18	R19	R20	R21	R22	R23	R24	R25	R27	R28	R29	R30	R32	Total	
1	1	7	7	7	7	7	7	7	9	1	9	7	7	7	7	7	7	7	7	7	9	7	7	154	
2	2	1	7	9	1	9	7	7	1	1	7	3	3	3	3	7	7	1	7	3	7	7	7	108	
3	3	7	9	7	1	7	9	7	9	7	9	5	7	7	7	9	7	7	9	1	7	9	9	156	
4	4	9	9	9	1	1	1	5	5	5	9	5	7	7	7	7	9	9	9	3	9	9	9	144	
5	5	7	7	5	7	5	7	5	5	9	5	7	5	5	5	5	5	5	5	3	3	5	3	118	
6	6	7	9	9	9	7	7	5	9	9	7	7	7	7	7	7	5	5	7	5	5	7	5	152	
7	7	9	9	9	9	9	9	5	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	194	
8	9	7	5	7	9	7	7	7	7	9	3	3	3	3	5	7	7	7	7	9	7	7	9	142	
9	11	9	9	5	9	7	9	9	7	3	9	7	9	9	9	7	7	9	9	7	9	9	9	176	
10	12	9	9	9	9	9	7	5	5	5	9	7	7	7	7	9	7	9	9	9	9	9	9	174	
11	13	5	9	7	7	7	7	5	9	7	9	5	7	5	7	5	5	5	9	5	3	9	7	144	
12	14	7	9	9	9	9	9	9	9	9	9	9	9	9	7	9	9	9	9	9	3	9	9	188	
13	15	5	7	5	9	5	9	9	5	5	9	5	5	5	5	9	9	9	9	9	7	7	5	152	
14	16	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	7	9	9	7	9	9	5	190	
15	18	7	9	9	9	9	9	7	7	7	7	5	5	9	5	7	7	9	9	9	7	9	9	170	
16	19	7	9	9	7	9	9	9	7	7	9	9	7	9	7	7	9	7	9	7	5	9	9	176	
17	20	9	9	9	9	9	9	7	9	7	7	5	7	7	7	5	7	7	7	9	9	7	9	170	
18	21	9	7	7	9	7	9	9	9	7	9	7	7	9	9	7	7	7	7	9	7	5	7	9	172
19	24	9	9	5	9	5	9	5	9	9	9	9	9	9	9	7	9	9	9	9	5	9	5	176	
20	26	7	9	9	9	9	7	9	9	9	9	7	7	7	7	7	9	7	9	7	9	9	9	180	
21	28	7	9	7	7	7	7	9	5	5	7	5	5	5	7	5	5	5	5	5	5	7	5	134	
22	30	7	9	7	9	9	9	7	7	7	9	9	9	9	9	9	9	9	7	7	9	9	9	184	
23	31	7	7	7	7	9	7	5	5	5	9	9	7	7	7	9	5	7	7	5	7	7	7	152	
24	32	7	9	9	9	9	9	9	9	9	9	9	9	9	7	5	9	9	9	5	9	9	9	186	
25	33	7	7	7	7	7	9	7	7	7	7	7	7	7	7	7	9	9	9	7	7	7	7	162	
26	34	7	7	7	7	5	5	5	9	9	9	5	5	5	5	7	7	7	7	5	3	5	7	138	
27	35	7	9	7	7	7	9	7	7	5	9	7	9	9	9	9	7	9	9	9	5	7	7	170	
28	36	7	9	7	7	7	9	9	7	9	9	5	7	5	9	9	5	5	9	5	7	7	9	162	
29	37	9	9	7	7	7	7	7	9	9	9	5	5	5	5	7	9	7	7	9	7	9	7	162	
30	38	7	7	7	9	9	7	5	5	5	7	5	7	7	5	7	9	5	7	5	7	7	9	148	
31	40	9	9	9	9	9	9	9	9	5	9	9	9	9	9	9	9	9	9	9	7	9	9	192	
32	41	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	154	
33	43	9	9	5	7	5	9	9	9	9	5	5	5	5	5	5	5	5	5	5	5	5	5	136	
34	45	5	7	7	5	7	9	9	9	7	9	5	9	7	7	9	7	9	9	7	5	9	5	162	
35	46	7	5	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	7	7	7	5	182	
36	51	7	9	7	7	9	9	9	9	5	7	5	7	7	5	7	9	7	9	7	5	9	9	164	
37	55	9	7	7	9	9	7	7	9	7	9	9	7	7	7	7	9	9	9	7	7	9	9	176	
38	60	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	198	

Correlación		0.112476437	0.044475	0	0.177	0.295	0.29	0.062	0.3	0.01	0.2444	0.5507	0.71	0.5032	0.184	0.08878	0.32128	0.371	0.1094	0.0346	0.13976	0.24508		
σ		1.601030962	1.197436807	1.37028	2.21	1.782	1.593	1.65	1.912	2.23	1.44	1.8903	1.7077	1.8	1.6108	1.407	1.5199	1.86912	1.29	2.1067	1.9467	1.29008	1.78184	20.94883
σ^2		2.563300142	1.433854908	1.87767	4.9	3.175	2.538	2.71	3.656	4.96	2.08	3.5733	2.9161	3.23	2.5946	1.98	2.3101	3.4936	1.664	4.4381	3.7895	1.6643	3.17496	438.85349
$\Sigma\sigma^2$		64.7311522																						
Factor 1	1.034482759	Div F1/F2 0.15																						
Factor 2	0.8524994																							
Alfa	α	0.882	$\left[1-\frac{\sum_{i=1}^k S_i^2}{S_i^2}\right], \quad \alpha = \left[\frac{k}{k-1}\right]$																					

>= 0.7 PASA LA PRUEBA

Figura 4.5. Cálculo del α de las respuestas de $k=38$ encuestas de empresas que desarrollan aplicaciones móviles.
Fuente: elaboración propia

4.5 Análisis de correlaciones.

Una de las ventajas de utilizar la escala de Likert es la consistencia de las respuestas entre los entrevistados, además de que permite en forma rápida calcular la correlación que existe entre las variables censadas (Kothari, 2004). Las ventajas del uso puede resumirse en:

- A) Es relativamente fácil de construir la escala de tipo Likert y evaluar sus resultados.
- B) Es considerada más confiable porque debajo de ella los encuestados contestan cada declaración incluida en el instrumento.
- C) Para cada declaración, se da una prueba empírica para discriminar capacidad y, como tal, permite el uso de declaraciones que no son manifiestamente relacionadas (tener una relación directa) a la actitud que se está estudiando.
- D) Se puede utilizar fácilmente en los estudios que responden a un análisis de centrado y de estímulo centrado, es decir, a través de él podemos estudiar cómo las respuestas difieren entre las personas y cómo las respuestas puede diferir entre los estímulos.
- E) Toma mucho menos tiempo para construir, se utiliza con frecuencia por los estudiantes de investigación. Por otra parte, se ha informado en varios estudios de investigación que hay alto grado de correlación entre las respuestas que utilizan escala tipo Likert.

4.5.1 Tipos de análisis aplicados a las encuestas.

En Kothari (2004) se describen los análisis o cálculos de determinados índices o medidas que se pueden aplicar a los resultados una encuesta, junto con la búsqueda de patrones de relación que existen entre los grupos de respuestas. Los análisis consisten en la estimación de los valores de los parámetros desconocidos de la población y la prueba de hipótesis para hacer inferencias. El análisis puede ser categorizado como análisis descriptivo y análisis inferencia (se conoce a menudo como el análisis estadístico).

El análisis descriptivo es el estudio de la distribución de una variable y nos ayuda a: 1) Proporcionar perfiles derivados del análisis, por ejemplo; perfiles de empresas, grupos de trabajo, las personas y otros sujetos, de un múltiplo de características tales como el tamaño, composición,

eficiencia, preferencias, etc. 2) Este tipo de análisis se relaciona con una variable (análisis unidimensional), o en el caso de dos variables (análisis bivariada) o con respecto a más de dos variables (análisis multivariado). En este contexto se estudian diversas medidas que muestran el tamaño y la forma de una distribución junto con el estudio de las relaciones que se miden entre dos o más variables.

Es posible definir entonces el análisis de correlación y el análisis causal. El análisis de correlación estudia la variación conjunta de dos o más variables para determinar la cantidad de correlación entre estas. El análisis causal tiene que ver con el estudio de cómo una o más variables afectan los cambios en otra variable. Por lo tanto, es un estudio de las relaciones funcionales existentes entre dos o más variables. Este análisis puede ser denominado como el análisis de regresión. El análisis causal se considera relativamente más importante en las investigaciones experimentales, mientras que en investigaciones más del tipo social y empresarial radican en la comprensión y el control de las relaciones entre las variables (correlación).

En los tiempos modernos, con la disponibilidad de las TI, se ha producido un rápido desarrollo del análisis multivariado que puede definirse como "todos los métodos estadísticos que analizan simultáneamente más de dos variables en una muestra de observaciones". Por lo general, los siguientes análisis están involucrados cuando se utiliza un análisis multivariado:

- A) El análisis de regresión múltiple: Este análisis se adopta cuando el investigador tiene una variable dependiente que se presume que es una función de dos o más variables independientes. El objetivo es hacer una predicción sobre la variable dependiente en función de su covarianza con todas las variables independientes interesadas.
- B) Análisis discriminante múltiple: Este análisis es apropiado cuando el investigador tiene una sola variable dependiente que no se puede medir, pero se puede clasificar en dos o más grupos sobre la base de algún atributo. El objeto es predecir la posibilidad de que una entidad pertenezca a un grupo determinado en base a varias variables de predicción.
- C) El análisis multivariado de varianza (o multi-ANOVA): Este análisis es una extensión del análisis de dos vías ANOVA, donde la relación de un grupo varianza dentro de una varianza grupo se encuentra en el conjunto de variables.

D) Análisis canónico: Este análisis se puede utilizar en el caso de las variables sean mensurables y no mensurables, el propósito es predecir simultáneamente un conjunto de variables dependientes de su covarianza conjunta con un conjunto de variables independientes.

Unos de los objetivos del diseñar una metodología ad-hoc a los intereses de los desarrolladores de la región es censar el grado de preferencia de las variables independientes, por lo tanto se utilizaron las medidas de relación entre estas variables.

El coeficiente de Karl Pearson de correlación (o correlación simple, r) es el método más utilizado para medir el grado de relación entre dos variables (el grado de afinidad se mide mediante r^2). Este coeficiente supone lo siguiente:

- 1) que no hay relación lineal entre las dos variables;
- 2) que las dos variables son casualmente relacionada lo que significa que una de las variables es independiente y el otro es dependiente; y que
- 3) un gran número de causas independientes están operando en ambas variables a fin de producir una distribución normal.

El análisis de las correlaciones para determinar la afinidad de las variables censadas, se muestran en la figura 4.6.

Correlación Pearson $r \geq 0.5$	7.Vanguardia	8.Reuso	10.Modelo	11.Usar un IDE	12.Metodología	13.Compatibilidad	14.Global Delivery	15.Nearshore	16.Offshore	18.FURPS	19.Métrica de Análisis	20.Métrica de Diseño	21.Métrica de Desarrollo	22.Métrica de Pruebas	23.Método de Calidad	24.Recursos para Calidad	25.Ventaja al usar Método	27.Tiempo proyectado	28.Control de Riesgos	30.Usar Procesos
7.Vanguardia														0.50			0.52			
8.Reuso																				0.50
10.Modelo				0.50																
11.Usar un IDE																			0.69	
12.Metodología						0.54														
13.Compatibilidad							0.54													
14.Global Delivery																				
15.Nearshore									0.55											
16.Offshore																				
18.FURPS											0.63	0.54	0.56					0.63		0.53
19.Métrica de Análisis												0.74	0.77	0.67			0.56			
20.Métrica de Diseño													0.84	0.83			0.64	0.56		0.52
21.Métrica de Desarrollo														0.71			0.69	0.55		0.52
22.Métrica de Pruebas																	0.54			
23.Método de Calidad																		0.51		
24.Recursos para Calidad																	0.52			
25.Ventaja al usar Método																		0.61	0.58	0.51
27.Tiempo proyectado																				0.69
28.Control de Riesgos																				
30.Usar Procesos																				

Figura 4.6. Resultado del análisis de correlaciones, bivariada de Pearson (r) con significancia ≥ 0.50 .
Fuente: elaboración propia.

El coeficiente de Pearson también es conocido como el coeficiente de correlación del momento del producto. El valor de r se encuentra entre ± 1 . Los valores positivos de r indican correlación positiva entre las dos variables (es decir, los cambios en ambas variables tienen lugar en la dirección declaración), mientras que los valores negativos de la r indican correlación negativa, es decir, los cambios en las dos variables que tienen lugar en las direcciones opuestas. Un valor cero de la r indica que no hay asociación entre las dos variables. Cuando $r = (+) 1$, indica una correlación positiva perfecta y cuando es $(-) 1$, indica perfecta correlación negativa, es decir, de ese modo que las variaciones en la variable independiente (X) explican 100% de las variaciones en la variable dependiente (Y).

También se dice que para un cambio unitario en la variable independiente, si ocurre que hay una variación constante en la variable dependiente en la misma dirección, a continuación, la

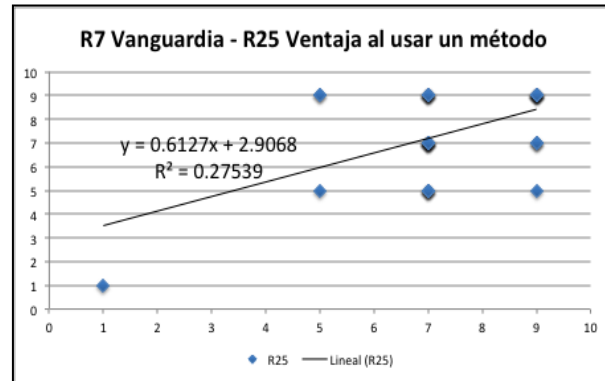
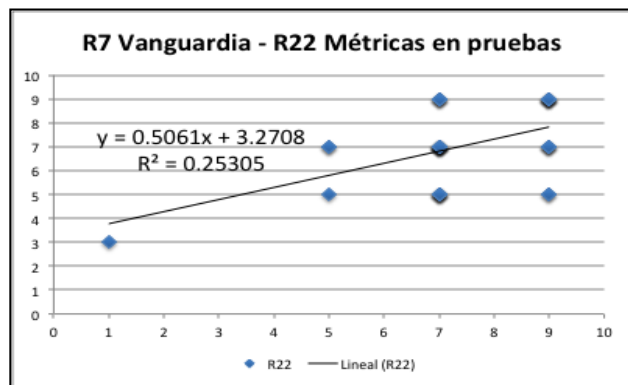
correlación se denomina como la perfecta positiva. Pero si tal cambio se produce en la dirección opuesta, la correlación se denomina como perfecto negativo. El valor de más cerca r a 1 o -1 indica alto grado de correlación entre las dos variables. Para tales efectos en la matriz anterior se colocaron solo aquellas r mayores a 0.5 y de este análisis se puede concluir lo expuesto en el tema siguiente.

4.6 Interpretación de las respuestas de la encuesta.

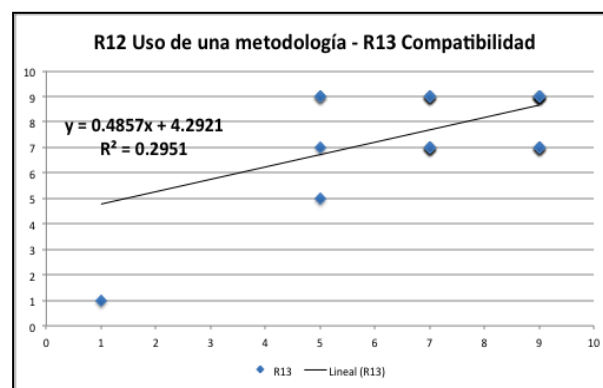
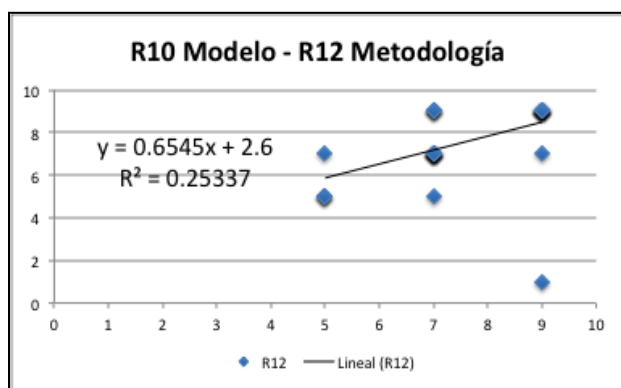
La metodología que se propone basada en el MDSIC debe explotar las características censadas y sus correlaciones ($r \geq 0.5$), se puede resumir que:

1. El reúso de componentes se considera un tema de vanguardia y permite el establecimiento de métricas, procesos y un método para lograr una ventaja competitiva.
2. El uso de un modelo de desarrollo utilizando un IDE, permite el control de riesgos.
3. Una metodología permite un desarrollo con compatibilidad y permite un desarrollo estándar que podrá ser difundido a través del “Global Delivery”.
4. La utilización de métricas, el cálculo del tiempo proyectado de un desarrollo y la utilización de procesos, es un tema importante en las fábricas de software.
5. El uso de un método permite establecer los recursos y costos que ayudan a generar APPs con calidad.
6. El establecimiento de métricas para definir costos en las etapas de análisis, diseño, desarrollo, pruebas y calidad están fuertemente relacionados con una ventaja competitiva.
7. El uso de indicadores de calidad (KPIs) en diferentes etapas del desarrollo de una APP se considera una buena práctica.

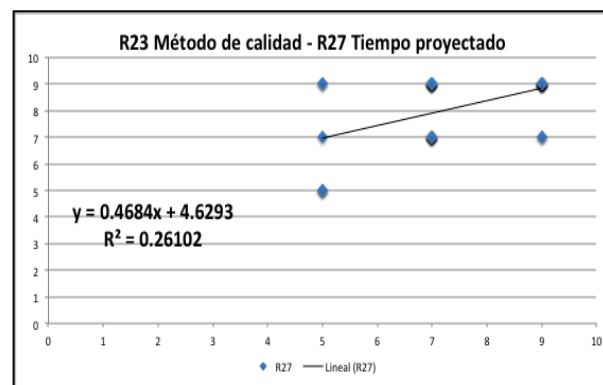
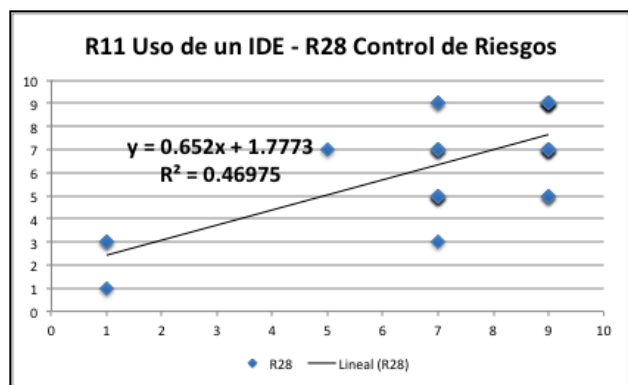
En la figura 4.7 A), B), C), D), E) y F) se muestran las principales correlaciones (variables censadas en la encuesta) y su afinidad r^2 .



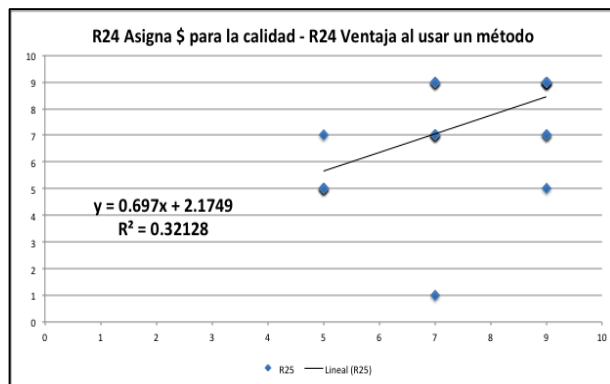
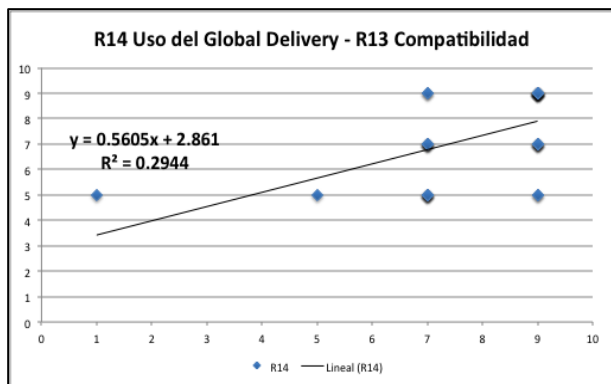
A) Correlación de tema de vanguardia con el uso de métricas y un método en el desarrollo.



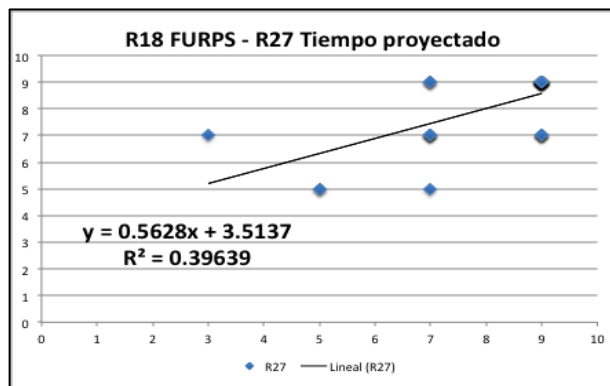
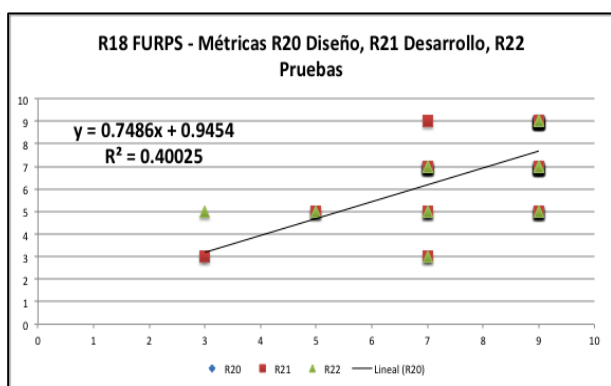
B) Correlación del uso de un modelo, metodología y compatibilidad en el desarrollo.



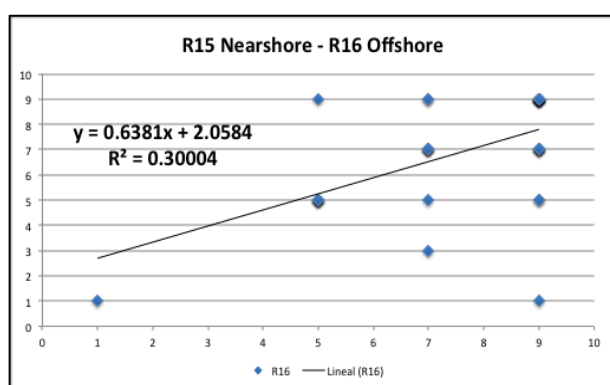
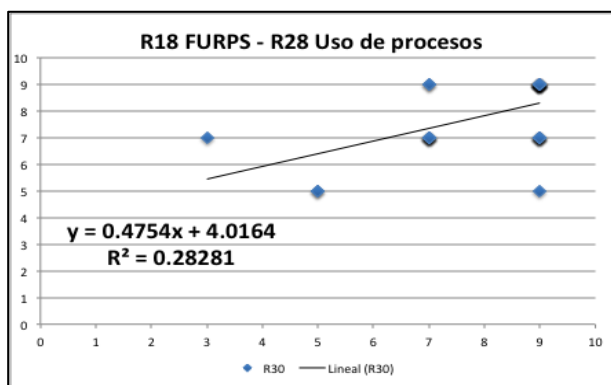
C) Correlación entre el uso de un IDE con control de riesgos y método de calidad con el tiempo proyectado.



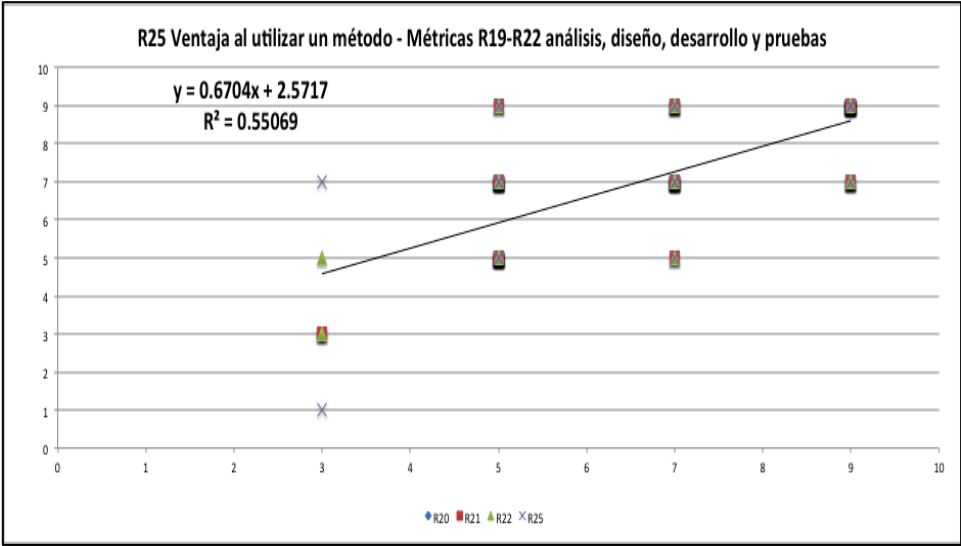
D) Correlación entre Global Delivery con compatibilidad y Asignar fondos para calidad con el uso de un método.



E) Correlación entre FURPS, métricas, desarrollo y pruebas y FURPS con Tiempo proyectado



F) Correlación entre FURPS con procesos y nearshore con offshore.



G) Correlación de uso de un método con métricas.

Figura 4.7 A), B), C), D), E) y F). Principales correlaciones con $r \geq 0.5$, preguntas con escala de Likert.
Fuente: elaboración propia.

De acuerdo a las respuestas de las 22 preguntas en escala de Likert se sintetiza el porcentaje de aprobación para cada una, en la figura 4.8.

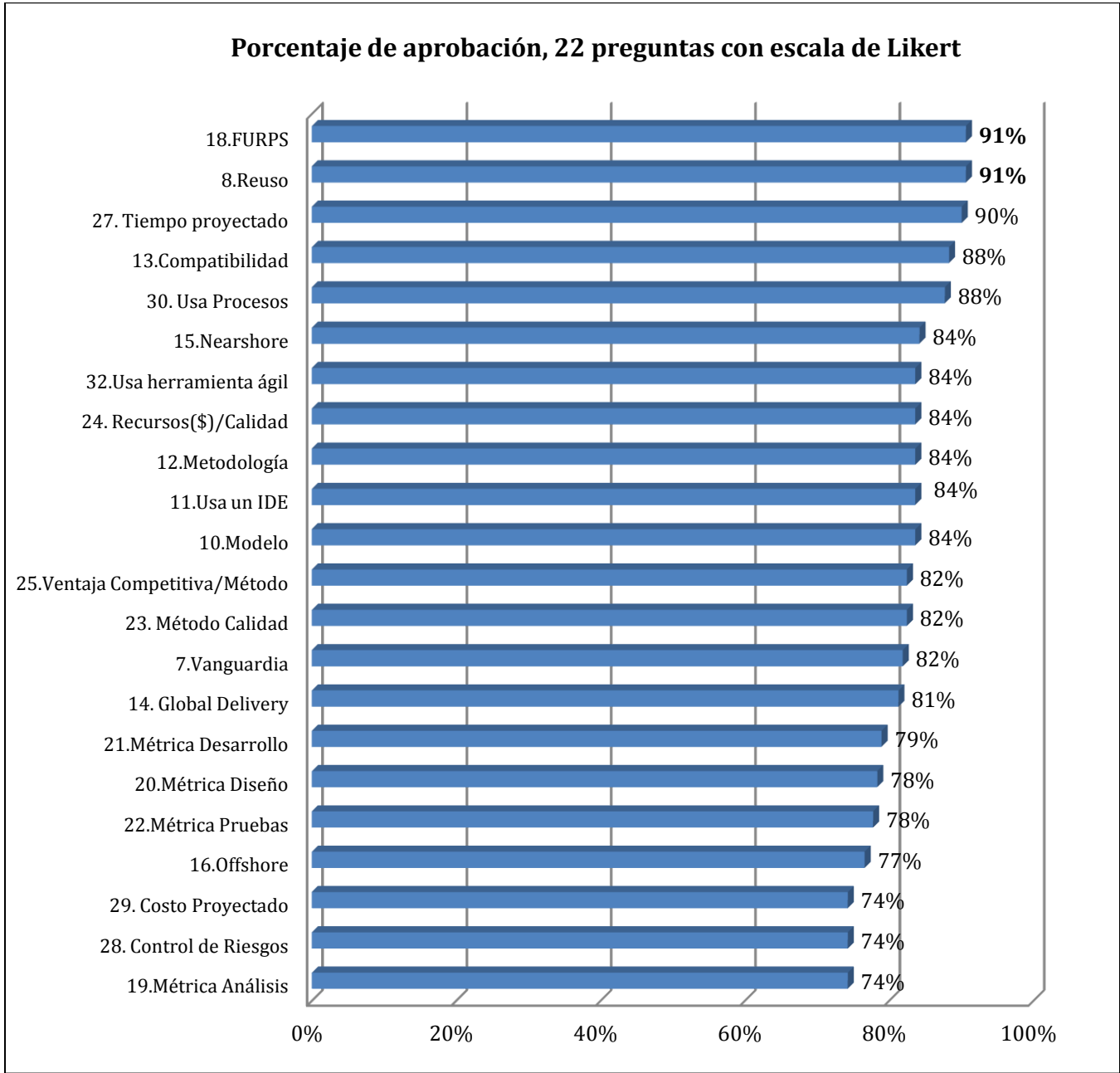


Figura 4.8. Resultado de las preguntas que tuvieron mejor aprobación entre las 38 de 70 empresas que desarrollan aplicaciones móviles.
Fuente: elaboración propia.

Como se observa, los encuestados respondieron a estas preguntas de acuerdo a su percepción (atributo) con la respuesta de “totalmente de acuerdo” hasta “totalmente indiferente”, con esta se obtiene el orden de importancia. Las variables que deben tomarse en cuenta en una

metodología ad-hoc para el desarrollo de aplicaciones móviles, a continuación se describen de acuerdo a su importancia:

1. Los indicadores FURPS se convierten junto con el “reuso” de componente como los temas más importantes que se debe tomar en cuenta en una metodología de desarrollo, 91% de los encuestados están de acuerdo.
2. El tiempo proyectado en el desarrollo es un tema muy importante porque está asociado a la calidad del producto y al costo final, el 90% están de acuerdo.
3. La “compatibilidad” en el desarrollo y la utilización de “procesos” como método de organización es otro tema importante entre los desarrolladores. El 88% están de acuerdo.
4. El uso de una herramienta para desarrollo ágil, de una metodología o un método o un modelo para el desarrollo y la utilización de una herramienta IDE, además de recursos monetarios para garantizar la calidad, son considerados como temas muy importantes en el desarrollo de APPs. El 84 % de los encuestados está de acuerdo.
5. El usar un método, un estándar de calidad y herramientas de vanguardia son considerados como una forma de ventaja competitiva. El 82% están de acuerdo.
6. El uso de una entrega del producto a través de tiendas estandarizadas (*store*) y el uso de métricas en las diferentes etapas del desarrollo es también impórtate. El 80% están de acuerdo.
7. El *offshore* (programadores externos de otros países) y *nearshore* (regionales o locales), formas de *outsourcing* lo consideran como dos temas importante para la ventaja competitiva. El 77% de los encuestados está de acuerdo.
8. Finalmente temas como el costo proyectado, el control de riegos y el uso de métricas en la etapa de análisis son los temas de menor aprobación. El 74% de encuestados están de acuerdo.

Para las preguntas abiertas se elaboraron gráficas que muestran los resultados de las encuestas aplicadas (15). A continuación se muestran las gráficas en las figuras 4.9 a la 4.18.

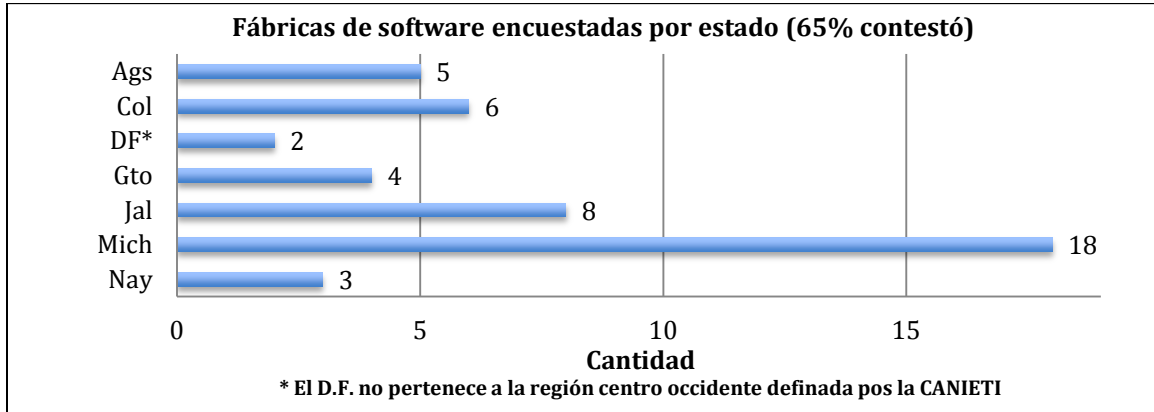


Figura 4.9. PyMEs que contestaron la encuesta por estado 46 de 70 contestaron. Fuente: elaboración propia.

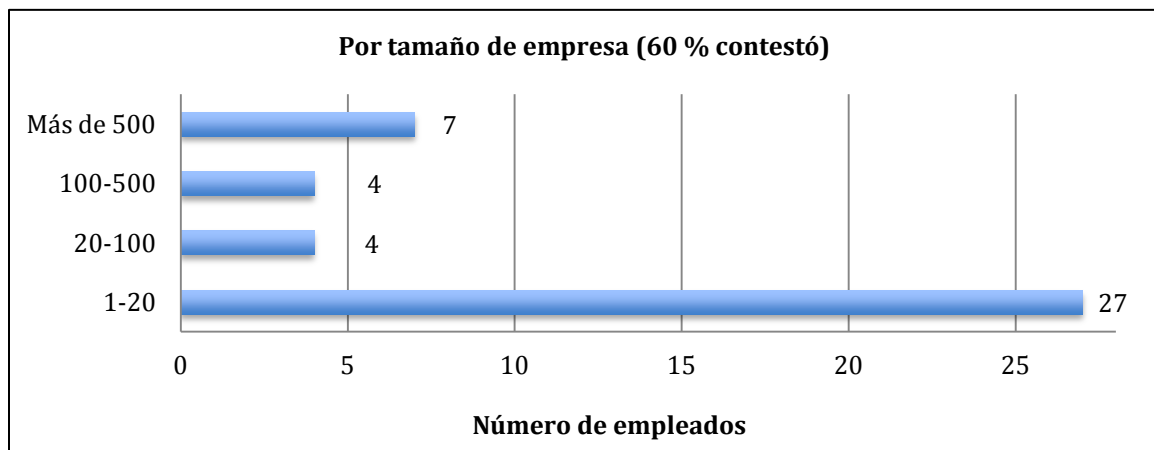


Figura 4.10. Tamaño de las empresas, 42 de 70 contestaron (60%). Fuente: elaboración propia.

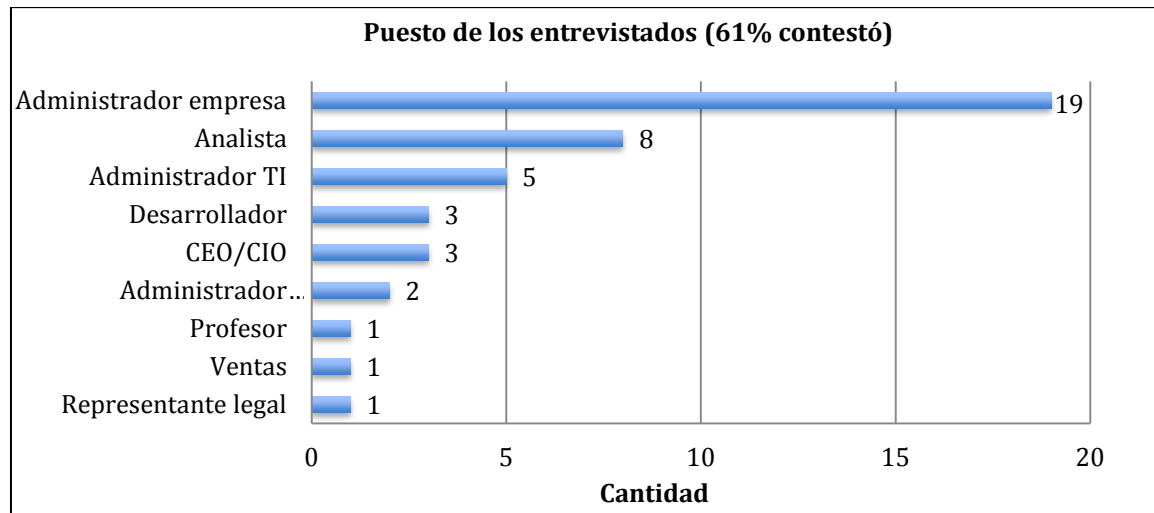


Figura 4.11. Puestos de trabajo de los entrevistados 43 de 70 contestaron (61%). Fuente: elaboración propia.

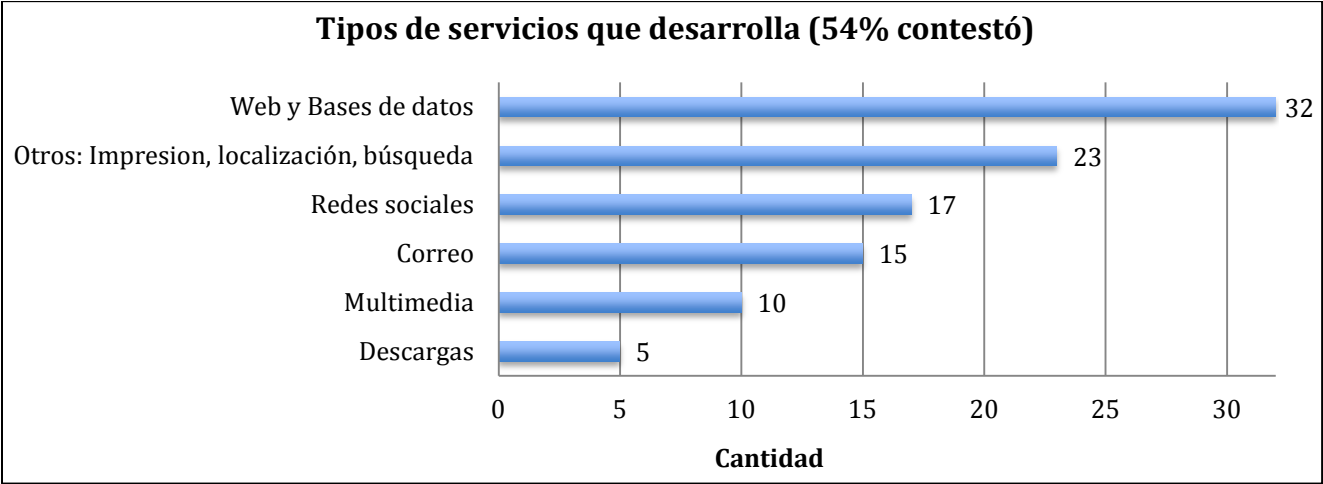


Figura 4.12. Tipos de servicios para los que desarrollan APPS, 38 de 70 contestaron (54%).
Fuente: elaboración propia.

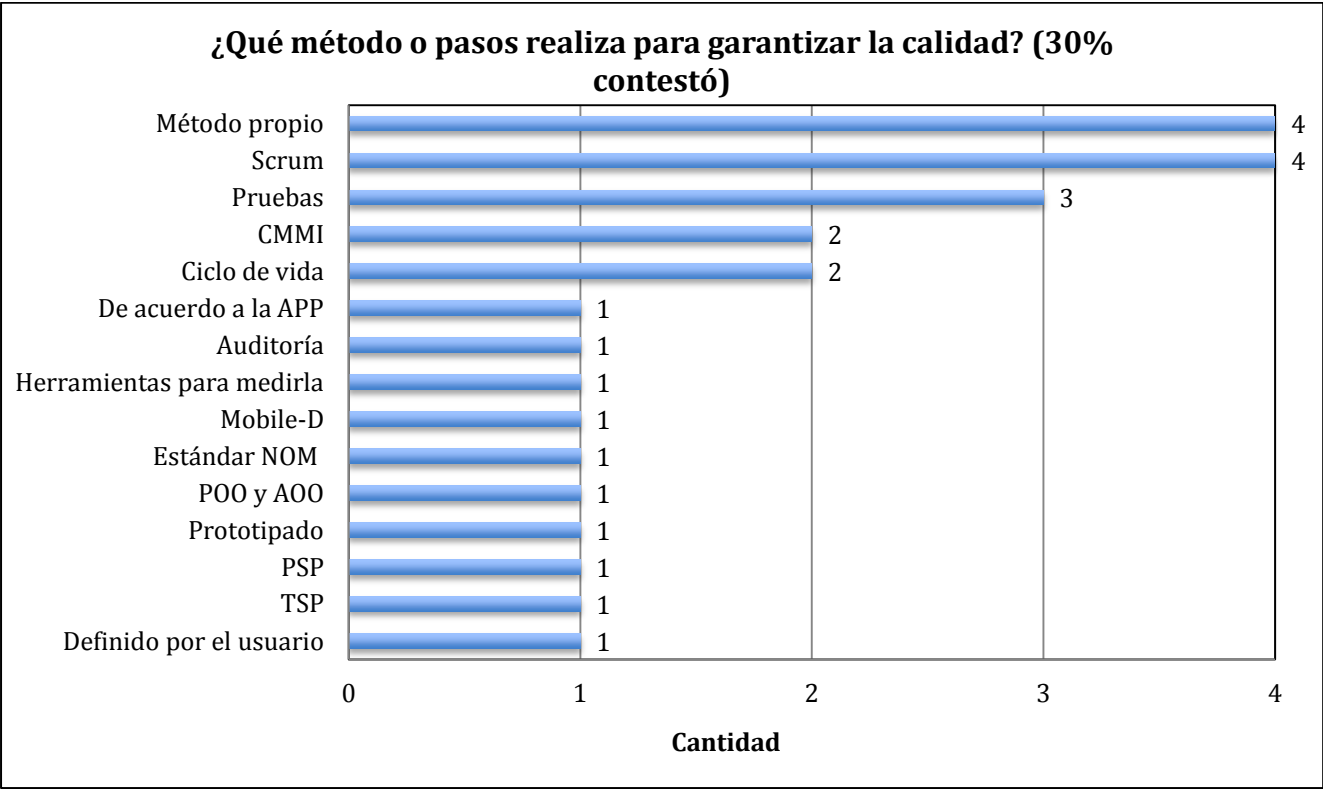


Figura 4.13. Métodos que conoce para garantizar la calidad, 25 de 70 contestaron.
Fuente: elaboración propia.

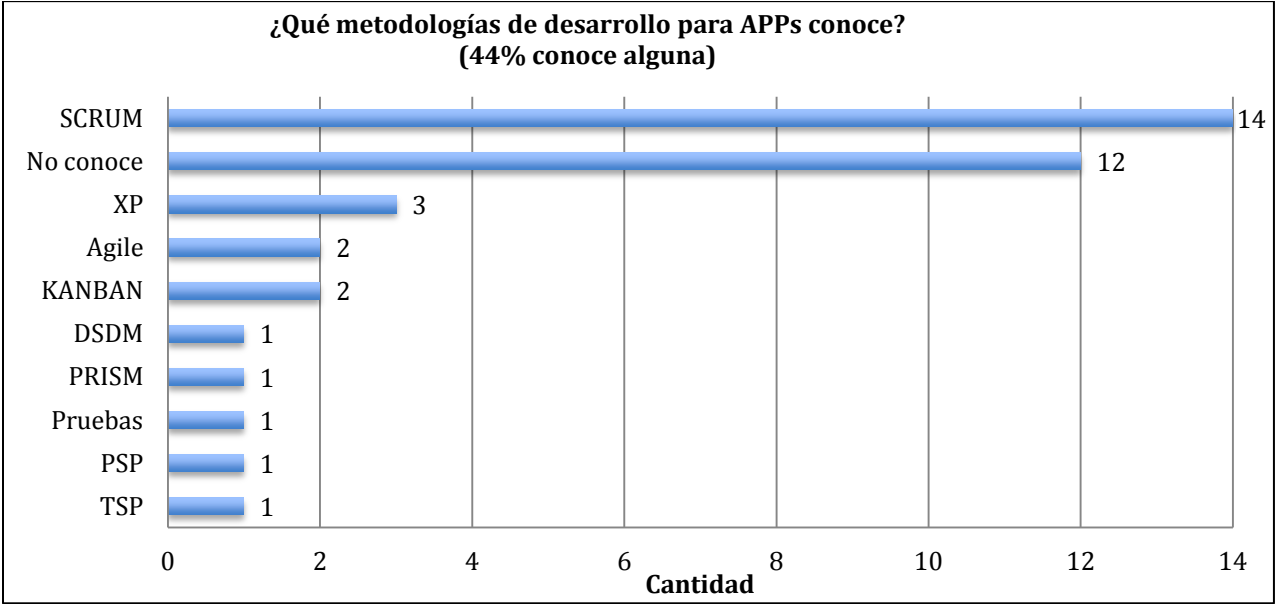


Figura 4.14. Metodologías de desarrollo para APPs que conoce, 38 de 70 contestaron.
Fuente: elaboración propia.

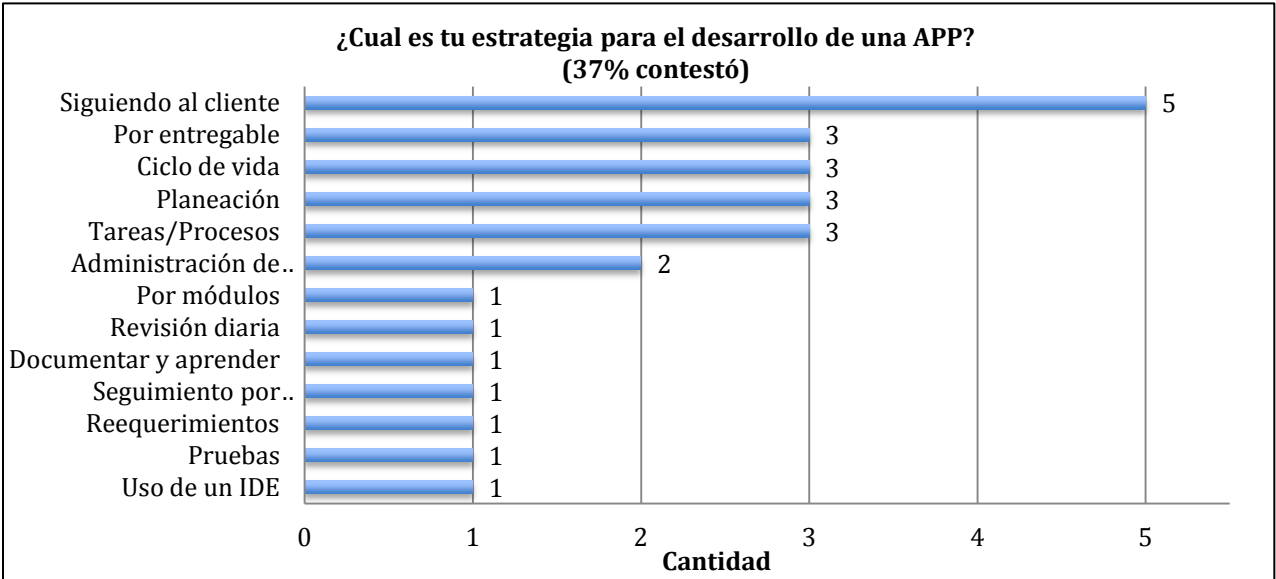


Figura 4.15. Estrategia de desarrollo que siguen las empresas para desarrollar una APP, 26 de 70 contestaron.
Fuente: elaboración propia.

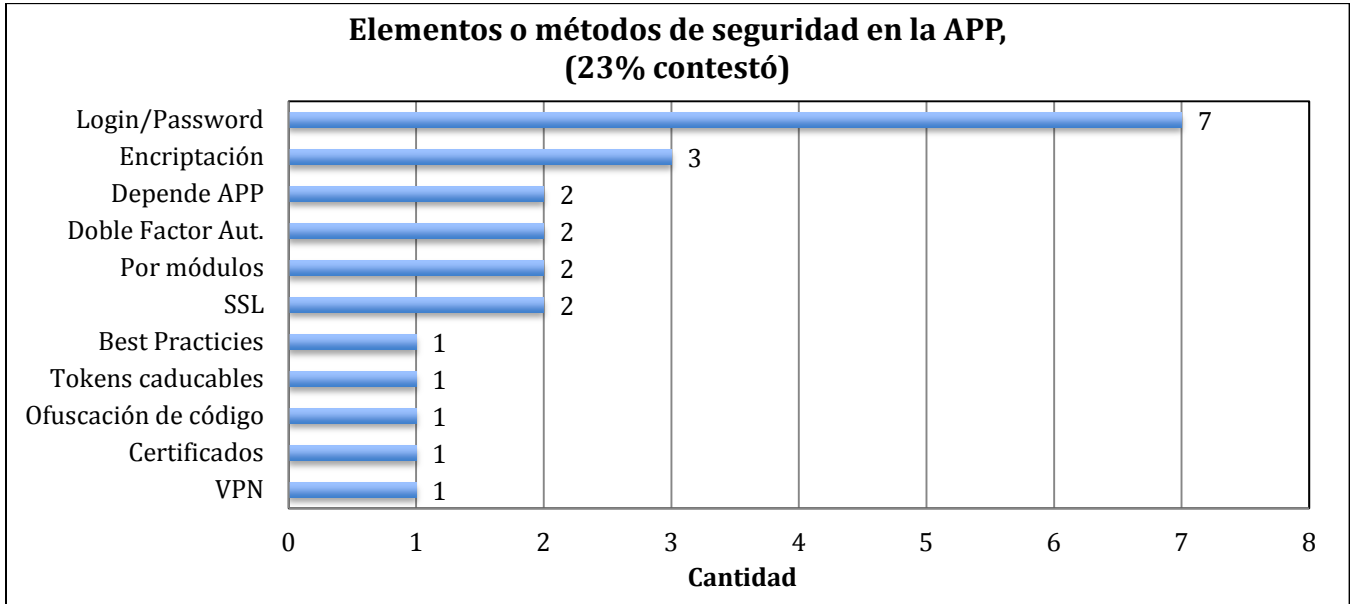
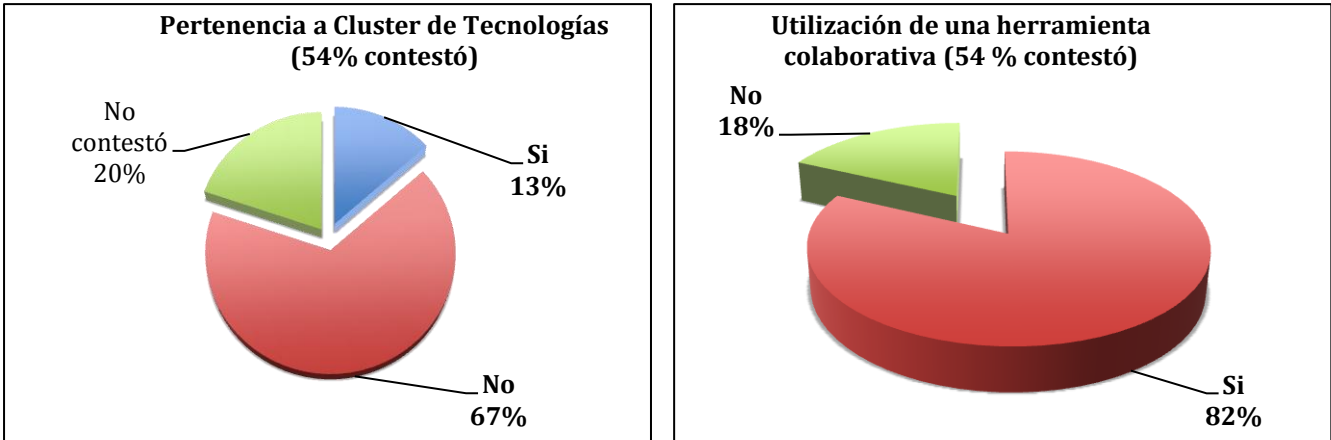


Figura 4.16. Métodos de seguridad que aplican en las APPs, 16 de 70 contestaron.
Fuente: elaboración propia.



A) Pertenece a un Clúster de TI

B) Usa alguna herramienta colaborativa

Figura 4.17. A) y B). Pertenece a un clúster de TI, solo 11 de los encuestados si pertenece a un clúster. B) Uso de herramientas colaborativas para el desarrollo, 20 de los encuestados utilizan alguna herramienta.
Fuente: elaboración propia.

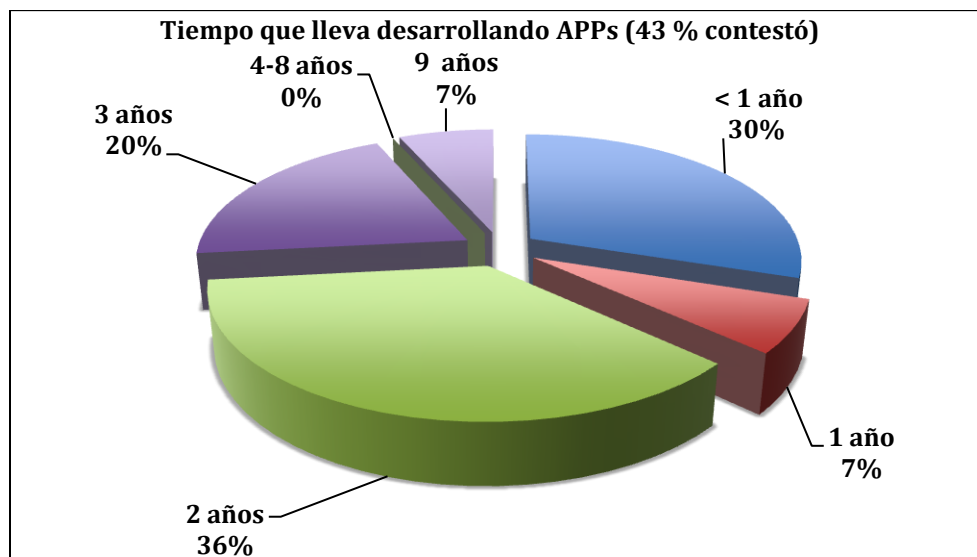


Figura 4.18. Tiempo que llevan desarrollando APPs, 30 de 70 contestaron.
Fuente: elaboración propia.

4.7 Indicadores de Calidad.

De acuerdo a los entrevistados se observó que la mayoría de las empresas lleva al menos dos años desarrollando aplicaciones móviles (36%) y el 37 % lleva un año o menos, esto implica que todavía las empresas están experimentando con alguna metodología, método o entorno de desarrollo en APPs. Siete empresas de más de 500 empleados respondieron la encuesta (10%) y de estas solo el 7% tiene más de 9 años desarrollando APPs. Esto claramente influye en la calidad en el desarrollo que muchas veces tiene que ver con un nivel de madurez.

El 38% de las empresas tienen menos de 20 empleados lo que indica que a falta de personal la calidad puede verse disminuida ya que el principal objetivo es mantener económicamente la empresa. De acuerdo al reporte del IMCO se ha observado una disminución en la competitividad en PyMEs por falta de liquidez y el acceso a programas de capacitación, esto afecta a la calidad en el desarrollo (IMCO, 2014). El 27% de los encuestados fueron los administradores de las empresas y dueños (que a falta de personal se convierte en gestores, desarrolladores y promotores de las APPs), el 30% de los encuestados son desarrolladores, analistas o administradores de TI, esto confirma la falta de personal calificado para promover un desarrollo con calidad. Las aplicaciones

que más desarrollan los encuestados son las páginas web con 29%, le sigue las aplicaciones orientadas a negocios con 22 % y las APPs para una plataforma en específico con un 14%, el resto de las aplicaciones para educación, comercio electrónico, juegos y multimedia suman un 35%. Este comportamiento está muy apegado al 32% para la APPs como prioridad de las PyMEs en la industria de las TIC en México, de acuerdo al estudio de Gallegos et. al (2014). En la figura 4.19 se muestra el tipo de aplicaciones que más desarrollan los encuestados.

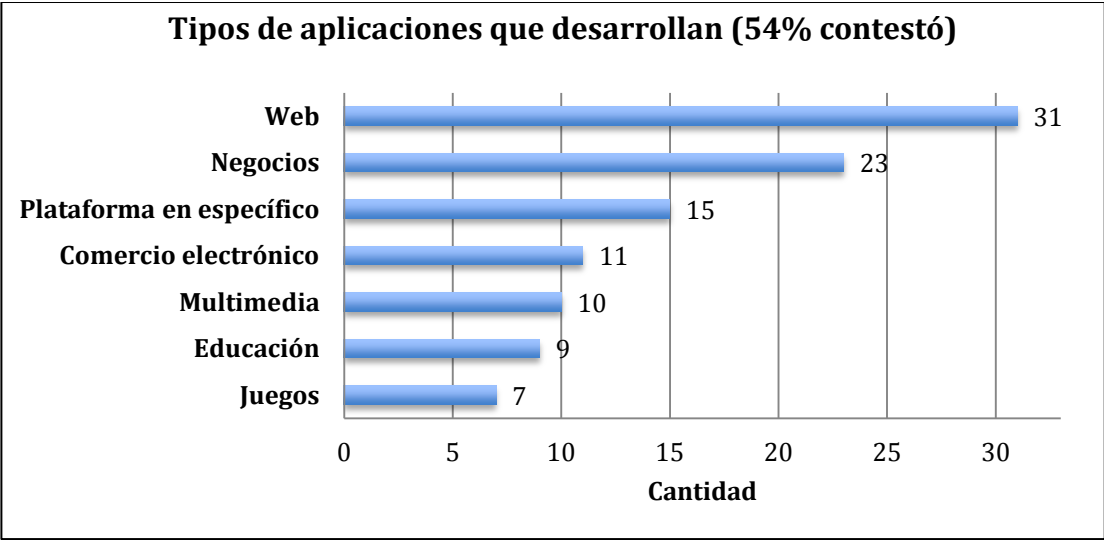


Figura 4.19. Tipo de aplicaciones que más desarrollan, 38 de 70 empresas contestó.
Fuente: elaboración propia.

De 70 empresas encuestadas 38 contestó que los servicios que más desarrollan en estas APPs son del tipo Web y bases de datos con un 31%, le siguen servicios de impresión, localización y búsqueda con un 23% y el desarrollo para redes sociales con un 17%, servicios como el correo electrónico, la multimedia y descargas (a través de las tiendas en línea) con un 46%. Es importante señalar que la mayoría de la empresas PyMEs están teniendo problemas para mantenerse económicamente, el marketing y la difusión a través de las páginas web sigue siendo uno de las principales formas de lucro en sector de TI.

Como parte de la calidad el uso de una metodología de desarrollo puede ser un diferenciador. El 20% de los encuestados declaró que utilizan Scrum como metodología, le sigue

4% con XP y 4% con metodologías como Agile y KANBAN. El resto de las metodologías (DSDM, PRISM, Testing, PSP/TSP) con un 7%.

Para censar el método de competitividad se les preguntó cual era su estrategia para el desarrollo de una APP y solo el 37% contestó (26/70). De estas la principal estrategia fue el desarrollo apegado a las necesidades del cliente con un 7%, le sigue por entregable con un 4%, usando el ciclo de vida 4%, usando la planeación de proyectos 4% y usando Tareas/Procesos con 4%. Estrategias de revisión diaria, documentar y aprender, seguimiento por semana, requerimiento, pruebas y el uso de un IDE tuvieron 10%.

Otro tema importante de calidad es la seguridad de las aplicaciones que desarrollan, los métodos que utilizan son muy variados. El 10% declaró que utilizan como método el login y password, le sigue la encriptación con 4%, dependencia del tipo de APP con un 3%, doble factor de autenticación con un 3%, autenticación por módulos con un 3%, el uso de SSL (capa de seguridad de TCP/IP) con 3% y el uso de las mejores prácticas, tokens caducables en tiempo, ofuscación de código, certificados digitales y las redes privadas virtuales (VPN) con un 7%.

De este análisis se obtiene que muchos de los encuestados relacionan como principales indicadores de calidad los siguientes (ver figura 4.7):

1. Facilidad, usabilidad, confiabilidad y soporte del desarrollo,
2. Reuso de componentes,
3. Garantizar el tiempo proyectado,
4. Desarrollo con compatibilidad,
5. Uso de procesos en el desarrollo,
6. Uso de nearshore (programadores locales) para la competitividad,
7. Uso de un herramienta para el desarrollo ágil,
8. Asignar recursos financieros para garantizarla,
9. Uso de un IDE ,
10. Uso de un modelo, método para la competitividad,
11. Uso de tendencias a la vanguardia en la APP (componentes),

12. Uso del Global Delivery como ventaja competitiva,
13. Uso de métricas en las etapas de desarrollo, diseño, pruebas,
14. Garantizar el costo proyectado,
15. Hacer control de riesgos al asignar recursos,
16. Uso de métricas en el diseño,
17. Uso de una metodología de desarrollo móvil,
18. Utilizar elementos de seguridad,
19. Asignar roles y establecer equipos de trabajo y
20. Utilizar procesos como forma de control del proyecto.

4.8 Estándares y normas de calidad.

De los métodos de calidad con los que desarrollan las PyMEs encuestadas sobresale el echo de que muchas utilizan métodos propios de acuerdo a su experiencia (4 empresas), estas usan una metodología ad-hoc (ver figura 4.13). Las empresas MiPyMEs son las que más innovan con sus propias metodologías.

La metodología Scrum es la más conocida y utilizada entre las PyMEs, se usa como una forma de garantizar un desarrollo con calidad (4 empresas la utilizan como su método principal). La forma de organizar equipos de trabajos y dar seguimiento al proyecto son principalmente las razones de su uso.

Otras metodologías con el testing (pruebas continuas en el desarrollo), son también utilizados como una forma de calidad. CMMI y sus niveles de madurez también es utilizado como una forma de garantizar la calidad, las empresas que siguen este modelo, se capacitan y se certifican para lograr el mayor nivel de madurez posible, es importante señalar que las empresas certificadas en CMMI son generalmente medianas y tienen más de 500 empleados (2 empresas encuestadas) y además invierten recursos para certificarse.

La utilización del ciclo de vida y el modelo de cascada también se utiliza entre las empresas encuestadas. El modelo de análisis, diseño, desarrollo e implementación es el que más utilizan. Esta tendencia fue más notoria entre las PyMEs de pocos empleados (menor a 20), cuyo método es el más utilizado.

Es importante mencionar que solo una empresa manifestó que sigue un estándar de la NOM-MX como método de calidad (Moprosoft). Esto confirma lo que describe el IMCO; seguir a la NOM muchas veces resulta muy costosa y difícil de aplicar en PyMEs. El programa Competisoft y Prosoft todavía no permean en estas empresas. Sin embargo los esfuerzos por lograr estos estándares de calidad si fueron promovidos en aquellas empresas que forman parte de un clúster (el 13% si pertenecen, 6 de las encuestadas).

Otro de los estándares de facto que utilizan las PyMEs es el diseño y la programación orientada a objetos como método de calidad, el uso de UML y casos de usos son la forma de promover un estándar de desarrollo de las APPs (una empresa manifestó este uso).

La utilización de los estándares de PSP y TSP para programadores y equipos de desarrollo también fue expresado por al menos una empresa de las encuestadas.

El uso de prototipos como estándar apegados a las necesidades de los clientes fue expresado por al menos dos empresas de las encuestadas.

La auditoría de datos y el uso de herramientas que miden la calidad fueron expresados por dos empresas. Solo una empresa manifestó el uso de un estándar de desarrollo móvil como Mobile-D, esta empresa se organiza de acuerdo a las etapas que promueve la metodología y su principal ventaja es el desarrollo de una APP compatible y con calidad. En la figura 4.20 se establece la tendencia hacia los métodos de desarrollo propios y Scrum, le siguen las pruebas, el CMMI y el ciclo de vida entre otros.

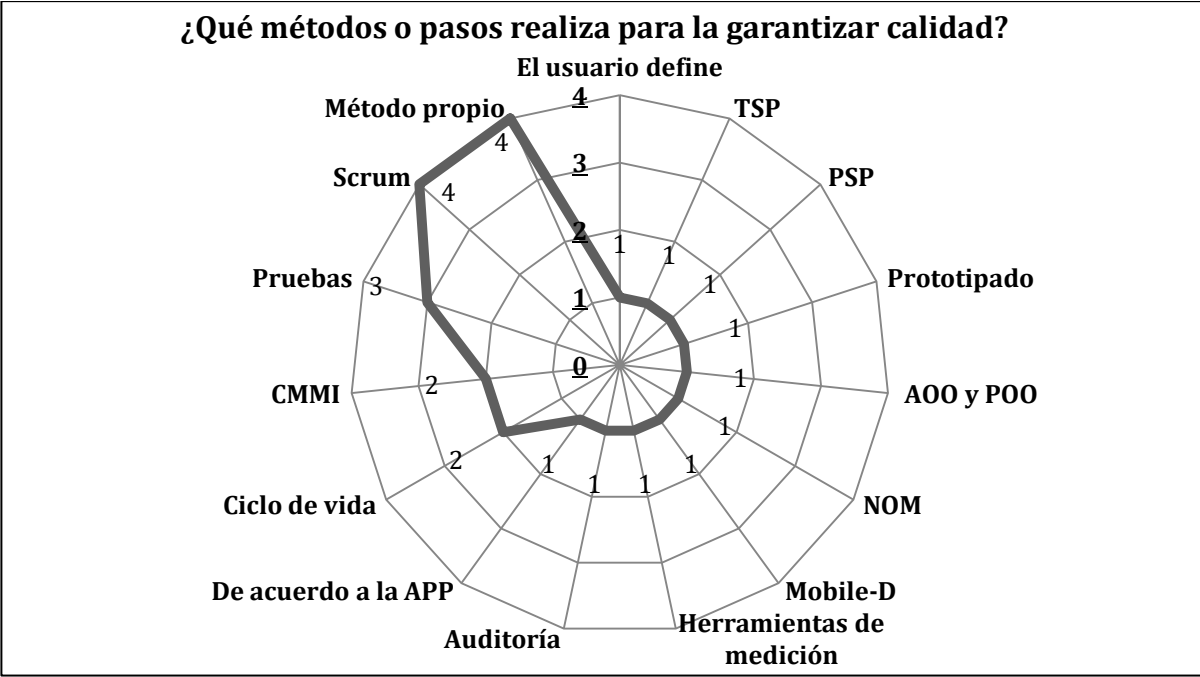


Figura 4.20. Tendencia hacia los métodos, metodologías o formas propias para garantizar la calidad las APPs.
Fuente: elaboración propia.

4.9 Alcance de la encuesta.

Como se observó la tendencia entre las PyMEs es hacia el uso de métodos propios que fomentan el desarrollo de las APPs con calidad “dudosa”. Es importante mencionar que muchas de las tiendas en línea tienen reglas para aceptar y promover una APP de acuerdo a sus criterios de calidad. A pesar de estas, muchas APPs tienen baja calidad y en su uso muchos usuarios se ven decepcionados con su funcionamiento. La forma más rápida de censar la calidad como usuario es durante la consulta de la APP en la tienda, en donde se observa la cantidad de descargas que tiene y los comentarios positivos y negativos expresados - este es el principal criterio de calidad -. Se observa también que las empresas siguen sus propios métodos que suponen un cierto estándar de calidad; algunas empresas PyMEs ya están apegándose a normas y estándares que les permita garantizar un desarrollo bajo normas y de calidad.

Capítulo 5. Metodología MDSIC-M.

5.1 Calidad en el desarrollo de una APP.

MDSIC-M se formula como una alternativa para garantizar la calidad en el desarrollo de APPs, la metodología se basa en procesos y elementos (basados en MDSIC) que deben ser orientados a la colaboración y el mantenimiento del plan de un proyecto de desarrollo móvil (M):

- 1) Establecimiento de requerimientos (tareas, entregables y métricas),
- 2) plan inicial (WBS),
- 3) establecer entregables en el WBS,
- 4) utilización de un planificador para el control de tareas (scheduler),
- 5) control del tiempo del proyecto por entregable,
- 6) control de riesgos al asignar recursos,
- 7) control de costos (establecimiento y métricas),
- 8) garantizar la calidad; KPIs (indicadores de desempeño) y el Scope (tiempo total) del proyecto,
- 9) uso de una herramienta colaborativa basada en el scheduler,
- 10) plan de pruebas y entregas (prototipos) y
- 11) documentar lecciones aprendidas (base de datos con la información del proyecto).

Un tema importante en el diseño de la metodología es basarse en estándares y promover un uso adecuado de estos entre las PyMEs, de aquí la necesidad de integrar estos elementos a la metodología si dejar de lado los preceptos del desarrollo ágil y adelgazado.

5.2 Mejores prácticas PMI Agile & Lean.

El PMI se fundó desde 60's para promover las mejores prácticas en la planeación de proyectos, en los 70's se construye como una institución y en los años 90's promueve la primera versión del *PMBOK (Project Management Body of Knowledge)* el cual se convierte en el pilar para la gestión y dirección de proyectos. En los años 2000's se comienzan a promover certificaciones y estándares en la dirección de proyectos; todos basados en la mejores prácticas de los agremiados (PMI, 2014). PMI Agile es la adopción de métodos ágiles en el mundo empresarial y de los negocios, PMI-Agile Certified Practitioner (PMI-ACP), es un programa completo de certificación en proyectos de software que emplean métodos ágiles. Se promueven los métodos tradicionales, así como los ágiles, se revisan métodos de desarrollo como Scrum, XP, Lean y Kanban. En la figura 5.1 se muestran las áreas de conocimiento (PMI-ACP, 2015).

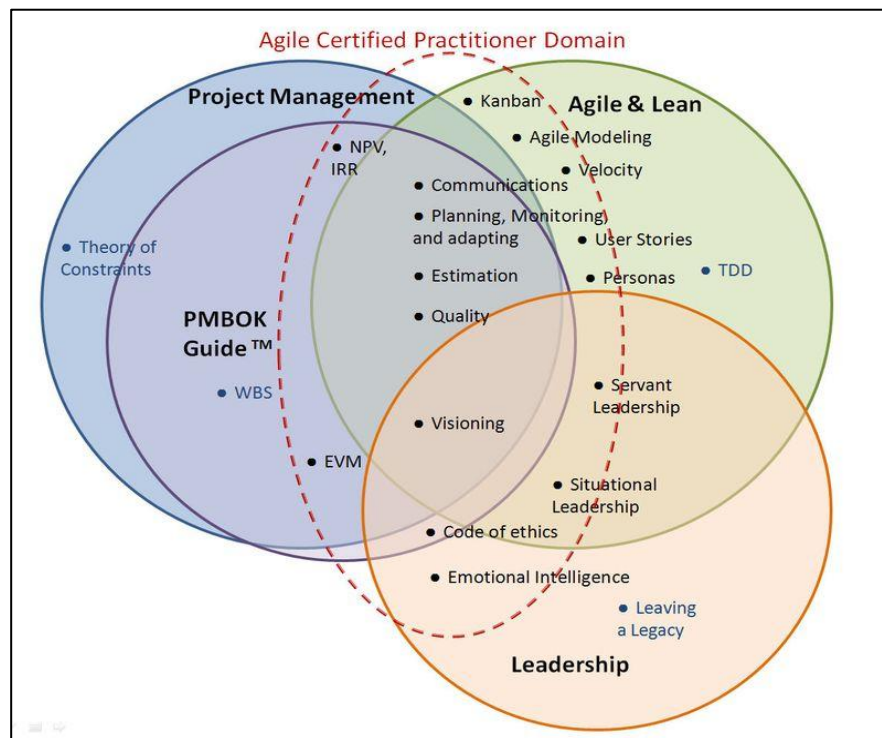


Figura 5.1. Áreas del conocimiento y de aplicación del PMI-ACP.
Fuente: (leadinganswers, 2015).

Como se observa las buenas prácticas en las metodologías ágiles se enfocan en los temas que abarcan las áreas de; administración de proyectos, elementos ágiles y adelgazados y el liderazgo. Estos temas se puede enumerar en:

1. Comunicaciones.
2. Planeación, monitoreo y adopción.
3. Estimación.
4. Calidad.
5. La visión de la administración.
6. Gestión del valor ganado (EVM).
7. Valor presente neto (NPV) y tasa interna de retorno (IRR).
8. Modelado ágil.
9. Metodologías ágiles; Scrum, Kanban, XP, entre otras.
10. Historias de usuario.
11. Liderazgo de servicio.
12. Código de ética.
13. Inteligencia emocional.

Todos estos temas deben ser considerados en la metodología que se propone para un desarrollo basado en las buenas prácticas y con calidad. Una parte importante de la planeación de proyectos, es el WBS que un tema muy revisado en el PMBOK.

El WBS o estructura de desglose especifica un listado de actividades con jerarquía que permite construir un mapa completo de todas la actividades del proyecto; tareas, recursos, tiempos, costos y entregables. Las actividades para ejecutar el proyecto giran en torno al WBS para cada una de las tareas, la administración de la ejecución en forma ordenada y viable es llevada a cabo por el scheduler.

5.3 Elementos de MDSIC-M.

Como se definió el scheduler o planificador es el núcleo de la metodología ya que da seguimiento y controla todas las actividades o tareas definidas en el WBS (buenas prácticas definidas en el *PMBOK*) y en el PMI-ACP. En la figura 5.2 se muestran los elementos que se integran en la metodología propuesta para el desarrollo de APPs.

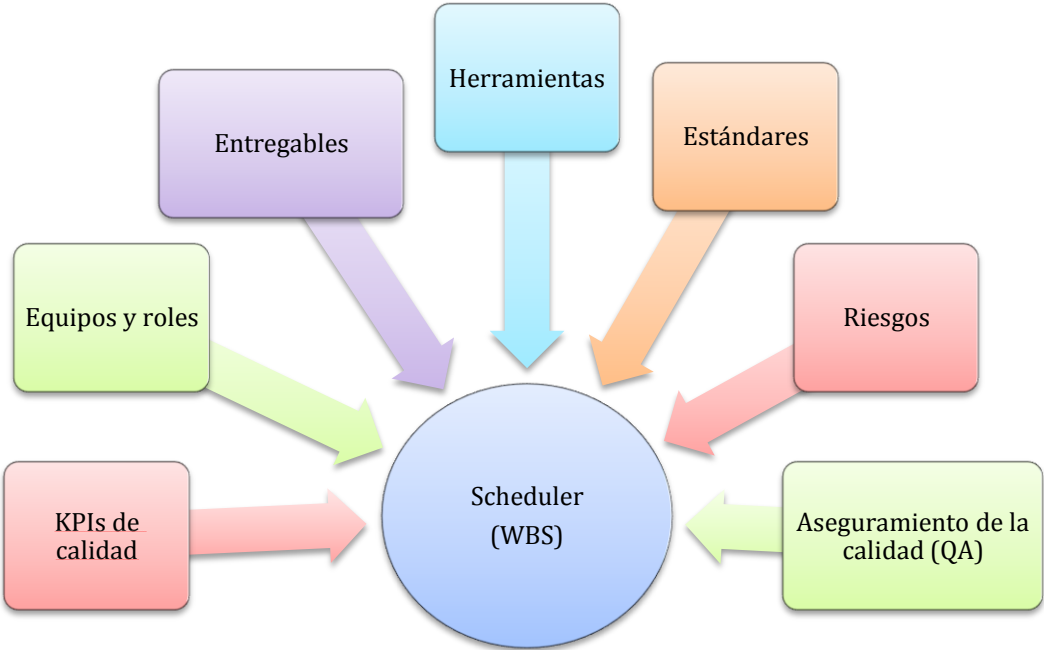


Figura 5.2. Elementos de MDSIC-M para garantizar la calidad en el desarrollo de una APP.
Fuente: elaboración propia.

Los elementos de la metodología se describen a continuación en la tabla 5.1.

Tabla 5.1 Elementos que definen la metodología MDSIC-M.
Fuente: elaboración propia.

Elemento	Descripción	Calidad
Indicador clave de desempeño (KPI)	Los KPIs definen los elementos a evaluar para censar la calidad; es una medida cuantitativa que describe el rendimiento de un atributo (métrica) en un sistema, componente o proceso.	Se deben establecer límites en cada KPI para determinar el grado de calidad que este aporta al medir un atributo (métrica).

Equipos y roles	Se define el rol de cada participante en el proyecto (promotor, administrador, cliente, programador, analista, calidad, entre otros), y los equipos que se conformarán (análisis, prototipo, desarrollo, calidad). Es importante tomar en cuenta la personalidad de cada uno de los integrantes.	El establecimiento de los roles y equipos permite organizar el trabajo en el proyecto y tener certidumbre en cuanto a riesgos, costos y recursos.
Entregables	Utilizando el WBS se definen el conjunto de tareas que construyen a un entregable (metas, objetivos a lograr en un tiempo), el entregable debe definirse con una fecha de inicio y fin que permiten revisar su avance. El establecimiento de hitos (puntos clave de revisión) puede ayudar a censar el avance de un entregable.	Como método para censar la calidad los hitos y entregables se vuelven importantes porque permite saber el grado de avance de un entregable.
Herramientas	Las herramientas se establecen para realizar una actividad en específico en el desarrollo de la APP, se establecen para: - Planeación del proyecto, - Prototipado, - Diseño de historias, - Desarrollo (IDE), - Probar componentes, - Uso de versiones, - Integración de componentes, - Pruebas y - Colocar la APP en una tienda en línea.	Debe definirse la utilidad y el alcance de cada herramienta, debe considerarse la capacitación, instalación y uso de cada una dentro del tiempo proyecto. El uso y conocimiento de las herramientas ayuda a crecer el nivel de madurez de un equipo.
Estándares	Se toman en cuenta los estándares de la NOM NMX-I-059/01-NYCE-2011: 01. Definición de conceptos y productos. 02. Requerimientos del Proceso (Moprosoft). 03. Lineamientos para los procesos de implementación. 04. Lineamientos para los Procesos de Evaluación (EvalProSoft). En el caso de las normas ISO se consideran: • ISO 9001 Gestión de la calidad, requerimientos. • ISO 9126 Calidad del producto. • ISO 12207 Procesos Ciclo de Vida.	Como modelo de calidad se basa en Competisoft y evalprosoft: Evaluación de procesos, procesos de alta dirección, gerencia y operación. Se documentan las métricas y atributos del proceso PAI (genérico) para prácticas genéricas, recursos genéricos y productos de trabajo genéricos. En el modelo MDSIC se definen los formatos necesarios para esta

	• ISO 20000 Servicios de TI. • ISO 25000 Calidad y evaluación del producto. • ISO 27001 Gestión de la seguridad. • ISO 31000 Administración del riesgo. Se considera un nivel de madurez de procesos desde él; 0: incompleto, 1: Realizado, 2: Gestionado, 3: Establecido, 4: Predecible y 5: optimizado	evaluación, además de los procesos de seguimiento y control de tareas definidos para MDSIC-M.
Riesgos	En la etapa de establecimiento de métricas, tareas y entregables en el WBS, se definen los riesgos y el costo en caso de ocurrencia de una tarea (planeación). Se documenta la incidencia de los riesgos para un control y un monitoreo (ejecución). Una práctica recomendada es elaborar una matriz de riesgos y costos.	El control de los riesgos por tarea, permite la gestión y establecer métricas para controlarlos y en su momento aprender de aquellos que comúnmente ocurren.
Aseguramiento de la calidad	El MDSIC hace un seguimiento por etapa del proyecto a implementar; en donde el encargado o el equipo de QA censan y validan para cada componente su avance, adicionalmente en MDSIC-M se agregan componentes como entregables, riesgos, control de tiempo y cálculo de costos. Uno de los procesos más importantes es la planeación donde se establecen junto con los involucrados los objetivos, las actividades y los requerimientos del proyecto. Otro de los procesos importantes que aseguran la calidad es la generación de productos, artefactos o componentes que deben construirse con las herramientas, tecnología y estándares adecuados para un buen funcionamiento.	La gestión, monitoreo y reporte de calidad, permiten el control en todo momento de aquellos componentes, tareas o procesos que no se han cumplido y que conllevan costos y retrasos en el desarrollo de una APP. La evaluación y el aprendizaje son partes vitales en la calidad.

5.2.1 Modelos de calidad en metodologías ágiles y en MDSIC-M.

El aseguramiento de la calidad es uno de los temas más complejos en las metodologías ágiles; el enfoque basado en equipos de trabajo, dinámica de equipos, psicología y calidad en el proceso (Osorio, 2008) son algunos de los factores que la complican. Para controlar estos factores, el desarrollo se centra en cinco prácticas esenciales:

5. Administración ágil de proyectos. Utilizando las prácticas descritas por PMI-ACP
6. Definición de requerimientos (*user stories*). Requerimientos, comunicación y retroalimentación. Elaboración de fichas con objetivos y roles (colaboración).
7. Modelo ágil, diseño y construcción iterativa.
8. Desarrollo basado en pruebas. Donde la pruebas y el control de defectos (*refactoring*) es la base de la calidad.
9. Integración continua. Se utilizan roles participativos y colaborativos.

Los métodos ágiles implementan los principios de la metodología *lean*; el principio básico para garantizar la calidad es detener la línea de producción y corregir problemas (*refactoring*). Es un enfoque diferente a la administración de proyectos. La calidad empieza con la correcta definición de los requerimientos, de allí surgen los procesos que se deben verificar, adaptar, verificar, adaptar, etc.

Cada proyecto debe tratarse “como un todo” y no como partes y se debe desarrollar con integración continua y retroalimentación de los equipos de trabajo.

La metodología MDSIC-M propone mejoras para ofrecer calidad mediante los KPIs; estableciendo roles, equipos de trabajo colaborativos, definiendo de manera clara los entregables, herramientas a utilizar, estándares para generar la información pertinente, controlar los riesgos (costos) y finalmente asegurar la calidad utilizando técnicas y estándares.

En la figura 5.3 se muestra la interacción entre estos componentes de acuerdo a los procesos definidos en el MDSIC y los nuevos elementos para el desarrollo móvil.

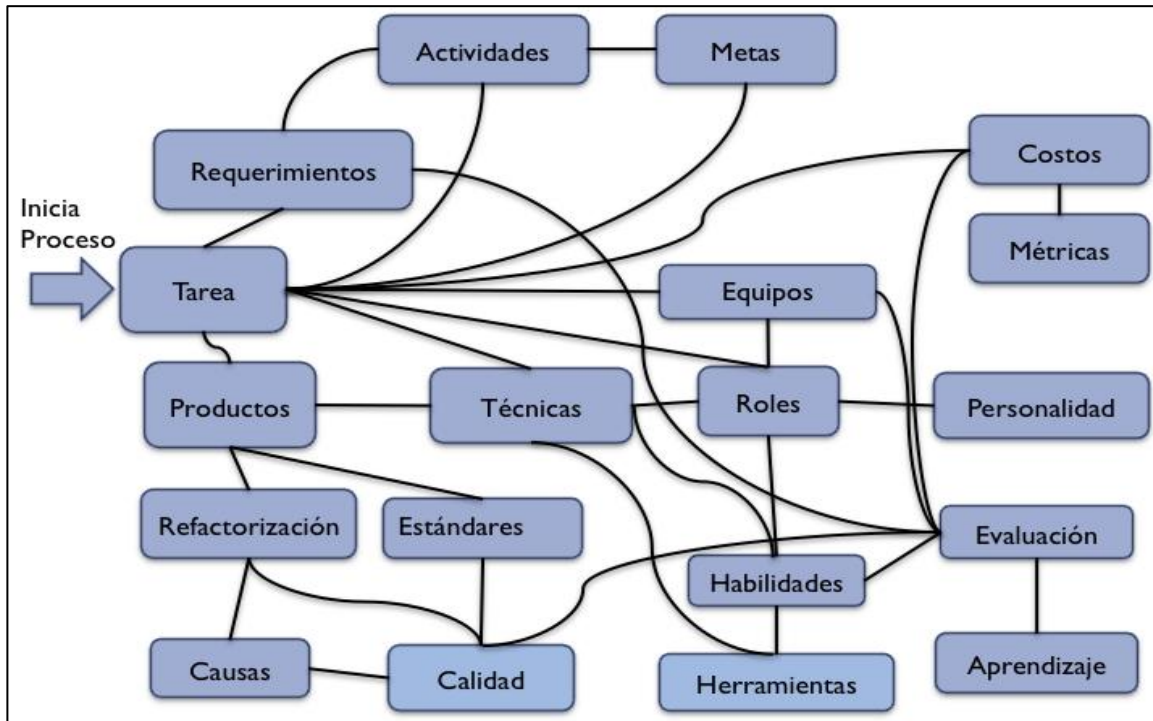


Figura 5.3. Elementos de la metodología que se ejecutan como procesos para garantizar la calidad en el desarrollo.
Fuente: elaboración propia.

La metodología MDSIC-M desarrolla un proyecto utilizando procesos en cada una de sus fases y su principal ventaja es el control de tareas mediante el planificador que es el núcleo. Para garantizar el funcionamiento se utilizó un método de especificación formal que permitió modelar y obtener un modelo confiable del scheduler. En los temas siguientes se describe su diseño.

5.3 Modelado del planificador (*scheduler*) de MDSIC-M.

En la tesis de (Ferreira M., 2010) se analiza la forma de mejorar el diseño y ejecución de tareas definidas en el WBS, que en el *PMBOK* se define como; “el desarrollo de un plan del proyecto usa las salidas de otros procesos de planeación para crear un documento único consistente y coherente que puede ser usado para guiar tanto la ejecución del proyecto como el control de este”. Las técnicas y herramientas para la definición del alcance de un proyecto, son:

- 1) Árbol de actividades WBS. Estructura jerárquica que define las actividades a realizar.

2) Descomposición. La descomposición involucra subdividir las principales entregas del proyecto en componentes más pequeños y manejables hasta que las entregas sean definidas con suficiente detalle como para soportar las actividades futuras del proyecto (planear, ejecutar, controlar y cierre).

El *PMBOK* propone el WBS como un listado jerárquico donde se describen la tareas (T_x) y sus elementos; los recursos que utilizará (R_y), el tiempo que tardará en ejecutarse (Δt), el costo asociado a los recursos utilizados (c_z) y los riesgos que dicha tarea tiene al ejecutarse (Rs_i). El conjunto de tareas se agrupan por entregable definido en el plan del proyecto. Un entregable E_n se define por el tiempo de inicio (t_i) y el tiempo de finalización o terminación (t_f), de cada uno. Tarea que no se describa en el WBS no es parte del proyecto. Ver figura 5.4.

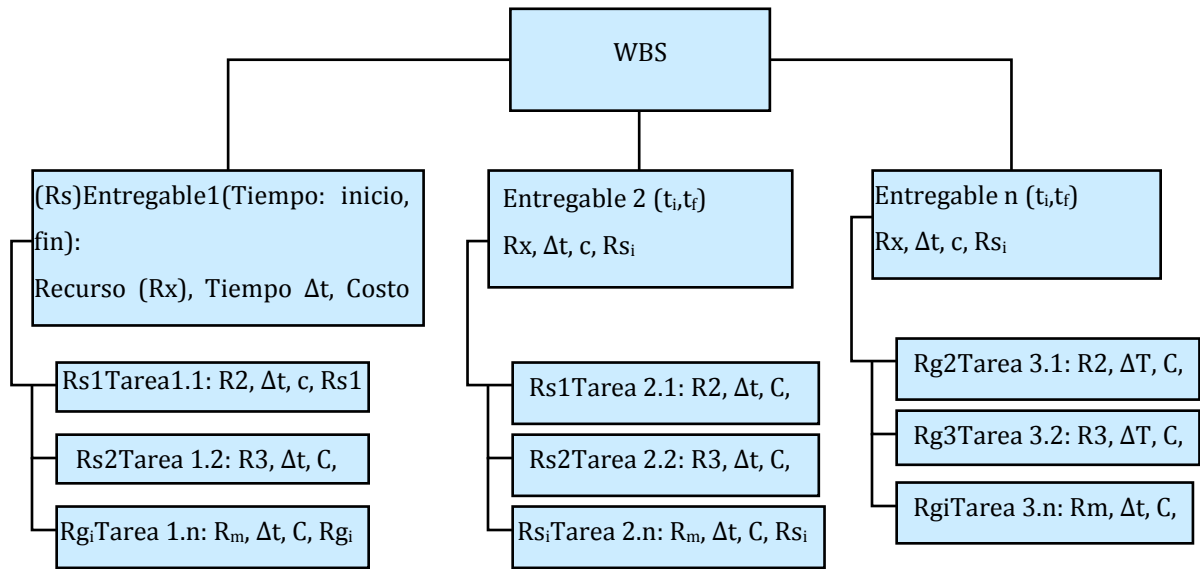


Figura 5.4. Elementos del WBS para establecer las tareas y entregables de un proyecto.
Fuente: Ferreira M. (2010).

La ejecución del plan como se describe en el *PMBOK* depende en gran medida de la experiencia de los promotores del proyecto, de tal situación surge la necesidad de una herramienta que de certidumbre de ejecución y ayude al control; considerando tareas en paralelo, donde los costos y los riesgos sean un factor determinante.

El scheduler que se propone en esta tesis se basa en la correcta descripción de las tareas, los recursos y sus costos definidos en el WBS. Es importante observar que los Δt (intervalo de tiempo que tarda en llevarse a cabo) de cada una de las tareas deben estar alineados con los tiempos definidos para cada uno de los entregables (E_n) del proyecto que se describen en el WBS.

Del WBS de la figura 5.4 se definen los tiempos para los entregables, sus costos y el riesgo de ejecución de cada una de las tareas, las formulas que se utilizan para este cálculo son:

- I). Tiempo total del proyecto ($Scope$)= $\sum_n$ Tiempo de cada Entregable x.
- II). Tiempo del Entregable = $\sum_m$ Tiempo de todas las tarea para el entregable x .
- III). Tiempo y costos por tarea (c_x)= Recursos (R) + Tiempo (Δt) + Costo (métricas (M^*) x Número de horas) + Probabilidad de Riesgo (Costo extra en caso de ocurrencia).
- IV). Riesgo = métrica x costo.

* Nota: Las métricas M, se basan en estimaciones de los costos de las actividades en el desarrollo de un proyecto (buenas prácticas y experiencias de los promotores), se definen como la cantidad de recurso que se requiere para llevar a cabo una tarea.

5.3.1 Modelado del scheduler con Redes de Petri Coloreadas (CPN).

La teoría de las Redes de Petri (PN, por sus siglas en inglés) ha llegado a ser reconocida como una metodología establecida en la literatura de las ciencias exactas para modelar sistemas de diferente índole (Jensen & Kristensen, 2009). Comparada con otros modelos de comportamiento dinámico gráficos, como los diagramas de las máquinas de estados finitos, las PN ofrecen una forma de expresar procesos que requieren diferentes características. Uno de los temas más importante es que pueden ser analizadas de manera formal y obtener información del comportamiento dinámico del sistema modelado.

Para modelar un sistema se usan representaciones matemáticas logrando una abstracción del sistema, esto es logrado con las PN, que además pueden ser estudiadas como autómatas e investigar sus propiedades matemáticas.

Un sistema es un arreglo dinámico que en el transcurso del tiempo tiene variaciones y no permanece estático. Para modelar el estado de un sistema se analiza su comportamiento, es decir de las acciones dadas en un tiempo.

Jensen & Kristensen (2009) definen una CPN como una tupla de 9 elementos, $CPN = (\Sigma, P, T, A, N, C, G, E, I)$ definidas como:

Σ es un conjunto finito de tipos no vacío, *tokens*.

P es un conjunto finito de lugares.

T es un conjunto finito de transiciones.

A es un conjunto finito de arcos, tal que:

$$P \cap T = P \cap A = T \cap A = \emptyset$$

N es un nodo función. Se define de A en $P \times T \cup T \times P$.

C es una función color. Se define de P en Σ .

G es una función guardián (*guard*). Se define desde T en la expresión tal que:

$$\forall t \in T: [\text{Tipo}(G(t)) = \text{Booleano} \wedge \text{Tipo}(G(t)) \subseteq \Sigma]$$

Donde $\text{Tipo} = \{\text{variable}, \text{expresión}, \text{valor}\}$

E es una función expresión de un arco. Se define desde A en la expresión tal que:

$$\forall a \in A: [\text{Tipo}(E(a)) = C(p)_{MS} \wedge \text{Tipo}(E(a)) \subseteq \Sigma]$$

Donde p es un lugar de $N(a)$, $\text{Tipo} = \{\text{variable}, \text{expresión}, \text{valor}\}$

I es una función inicialización. Se define desde P en una expresión cerrada tal que:

$$\forall p \in P: [\text{Tipo}(I(p)) = C(p)_{MS}]$$

Cada uno de estos elementos de la CPN puede verse en la figura 5.5 en donde se representan dos lugares, una transición y un arco.

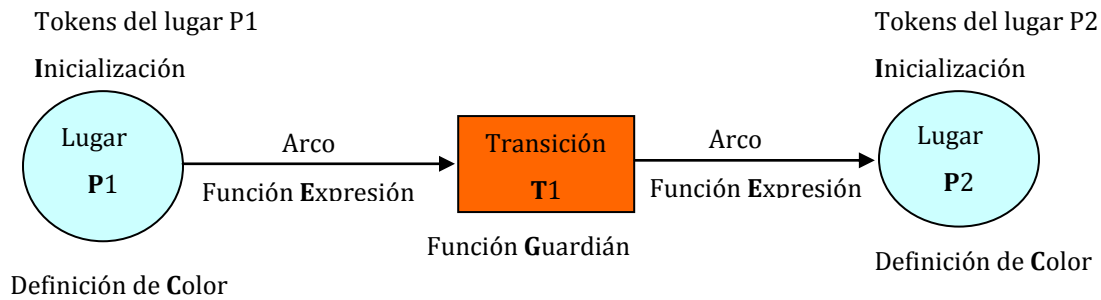


Figura 5.5. Elementos de una CPN.

Fuente: Ferreira M. (2010).

El grupo de desarrollo *CPN Group*, de la Universidad de *Aarhus* en Dinamarca, liderados por *K. Jenssen*, desarrollaron el *CPN-Tools* una herramienta gráfica para el modelado de sistemas utilizando una CPN (Westergaard & Verbeek, 2015).

La herramienta *CPN-Tools* se basa en el lenguaje de programación ML (de propósito general) para la definición de colores y los mensajes que se intercambiarán entre los estados. Esta herramienta además de mostrar la simulación del modelo permite analizar propiedades de un sistema concurrente. En la tabla 5.2 se muestran las propiedades de un modelo con una CPN.

Tabla 5.2. Propiedades de una CPN.

Fuente: Ferreira M. (2010).

PROPIEDAD	DESCRIPCIÓN
Vivacidad (L, <i>Liveness</i>)	Propiedad que permite conocer si una CPN está libre de candados mortales (ciclos inválidos)
Seguridad (S, <i>Safe</i>)	Permite conocer si existe al menos un token en un lugar P que permita iniciar un serie de disparos, la seguridad y vivacidad son dos propiedades complementarias
Alcanzabilidad (R, <i>reachability</i>)	Permite el estudio de las propiedades dinámicas de un sistema, permite conocer si un estado M_n es alcanzado desde M₀ a través de serie de disparos (σ)
Limitada (L, <i>limited</i>)	Permite conocer si el número de tokens que existe en cada lugar no excede de k-tokens
Cobertura	Permite conocer si un transición t_i es potencialmente disparable desde M₀

(C, coverage)	
Reversible (R, reverse)	Una CPN es Reversible si para cada estado M _i en $\sigma = R(\mathbf{M}_0)$, M ₀ es alcanzable desde M _i
Conservativa (C, conservative)	Permite conocer si una PN en todo momento preserva un número de n tokens

De acuerdo a las propiedades de una CPN, es posible construir el scheduler con un nivel de certidumbre en; su ejecución y control de las tareas, asignación de recursos, cálculo de costos y un análisis de riesgos.

En la figura 5.6 A), B), C), D) E) se muestra la construcción del scheduler utilizando CPN-Tools. En donde las áreas que se resalta son:

- A) Definición de tareas; establece recursos, tiempos, riesgos.
- B) Asignación de recursos; necesarios para garantizar la ejecución de la tarea.
- C) Asignación de equipos y roles de trabajo; quiénes participarán en la tarea.
- D) Ejecución de la tarea; se asigna tiempo de ejecución y los recursos son utilizados.
- E) Finalización y liberación de recursos y roles para la tarea.

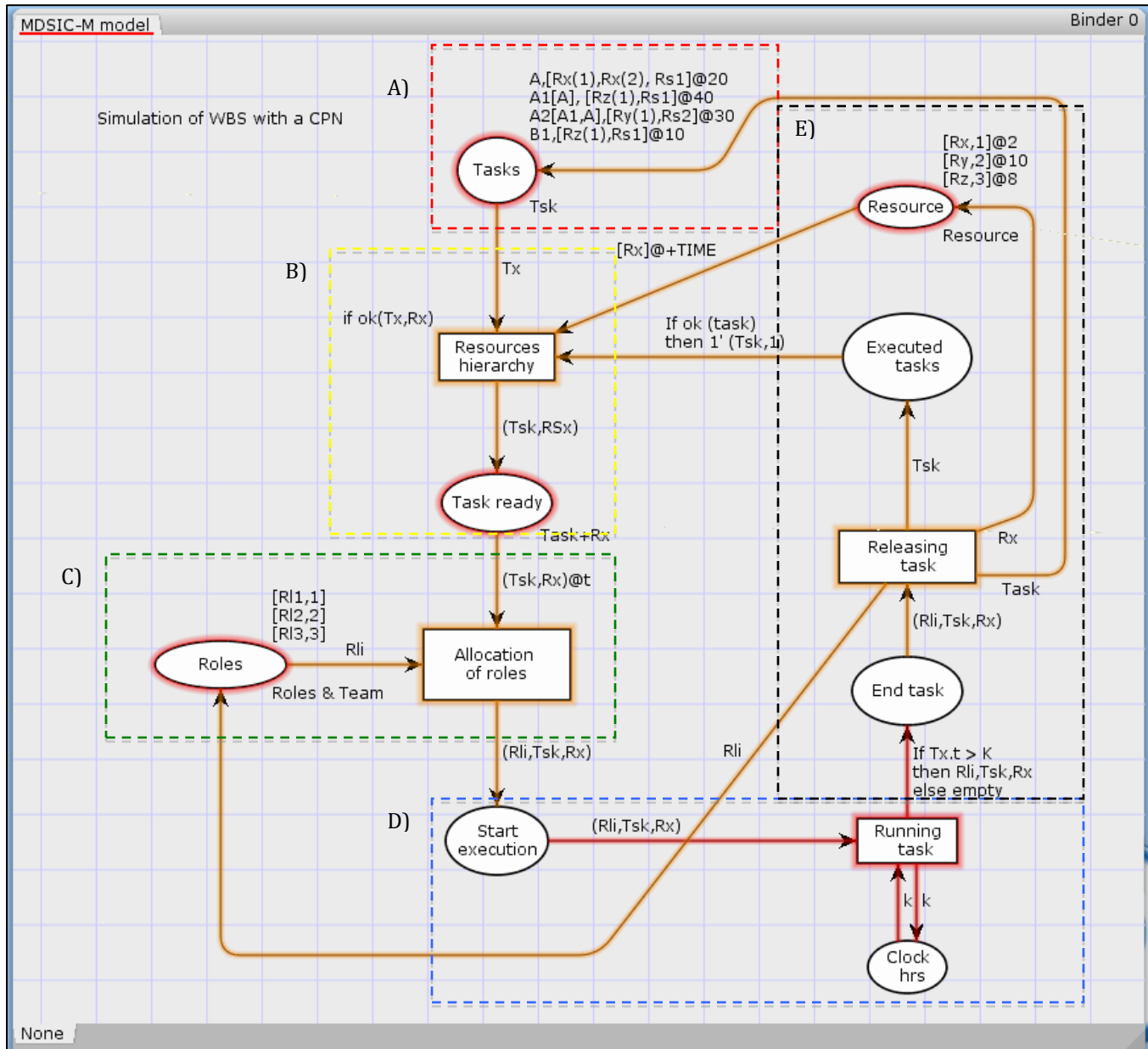


Figura 5.6. CPN que define los elementos del scheduler diseñado para MDSIC-M. A) Definición de tareas, recursos, tiempos, B) Asignación de recursos, C) Asignación de equipos y roles de trabajo, D) Ejecución de la tarea, E) Finalización y liberación de recursos y roles.

Fuente: Elaboración propia.

Como se observa para la ejecución de una tarea se realiza de acuerdo la definición de las jerarquías de tareas del WBS, para cada tarea es necesario:

- A) La definición de la tarea los recursos, tiempo en que se ejecutara (@) y los riesgos de no poder ejecutarse por falta de recursos o tarea postergada.

- B) Para cada tarea es necesario garantizar los recursos (R) necesarios (elementos finitos); son entidades que se requieren para ejecutar la tarea tales como; hora máquina, acceso a internet, accesos a servicios, etc.
- C) Una vez garantizados los recursos ahora se asignan los recursos humanos necesarios para llevar a cabo la tarea (antes de la ejecución), se asignan el personal necesario y/o los equipos de trabajo.
- D) Se inicia la ejecución de la tarea, se monitorea mediante el reloj maestro del scheduler y finaliza cuando el tiempo de la tarea se acaba (@).
- E) Finaliza la tarea y ahora se liberan todos los recursos que necesitó, se contabilizan riesgos que pudieron ocurrir (probabilidad) y finalmente se realiza la suma final de los costos de todas las tareas definen el tiempo y costo de un entregable.

La contabilización de tiempo debe realizarse por tareas y entregables finalizados. Al finalizar cada tarea debe liberar los recursos y las personas que intervinieron en su ejecución. Es importante señalar que la ejecución en tiempo, con los costos proyectados, define un proyecto viable en el modelo.

La ejecución del este modelo en la CPN arrojó las propiedades de vivacidad, seguridad, confiable, alcanzable, limitada y reversible lo que permite establecer que el modelo diseñado es viable de programar en algún lenguaje de programación (Jensen & Kristensen, 2009).

5.4 Etapas y procesos de MDSIC-M.

De acuerdo al diseño del scheduler, ahora se analizará el aseguramiento de la calidad en la metodología, se utilizará de acuerdo el modelo propuesto para metodologías ágiles en (Nathan-Regis & Nasira, 2013). Una metodología ágil que utilice técnicas de calidad debe considerar lo siguiente:

1. Asegurar la calidad en los requerimientos y artefactos, metamorfosis, arquitectura en pico y retroalimentación del cliente en el sitio.

2. En el desarrollo y diseño de componentes se debe garantizar la calidad mediante la refactorización, la programación por pares, reuniones de avances y simplificar problemas.
3. La implementación y aceptación de las pruebas deben hacerse al 100% con retroalimentación de los clientes.

Estas etapas deben generar un producto con calidad. Para controlar en forma adecuada la calidad, se propone el desarrollo de productos (por componentes) utilizando la técnica de refactorización, además se considera como parte importante, la asignación de roles y recursos a quiénes tengan las habilidades adecuadas para su desarrollo; esta asignación se realiza por parte del promotor del proyecto y el analista. Es importante señalar que la técnicas de QA deben ser revisadas y censadas para garantizar una correcta construcción de los componentes del proyecto, algunas de las prácticas propuestas por Nathan-Regis & Nasira (2013) son:

- 1) Refactorización,
- 2) Desarrollo basado en pruebas,
- 3) Integración continua,
- 4) Programación por pares,
- 5) Comunicación cara a cara,
- 6) Participación continua del cliente,
- 7) Retroalimentación frecuente del cliente y
- 8) Metáfora del sistema; que permite abstraer en forma rápida la aplicación e iniciar su desarrollo.

En la tesis de (Oberscheven, 2013), se describen los factores que influyen en la calidad de un software y se censan mediante la calidad funcional (QAF), la calidad estructural (QAs) y la calidad en el proceso (QAp), modelo definido en (Chappel, 2014). Los elementos son:

- 1) QAF; reuniones para requerimientos, desechar defectos o basura, buen desempeño, fácil de aprender y de usar.
- 2) QAs; código fácil de probar, código que sea de fácil mantenimiento, código entendible, código eficiente y código seguro.

3) QAp; reuniones para fechas de entregables, reuniones para garantizar presupuestos y desarrollo de procesos repetibles entregables alcanzables y calidad del software.

La metodología MDSIC-M considera los 4 niveles propuestos en el MDSIC (etapas 1, 3, 4 y 6 en color azul) y agrega controles utilizando el scheduler y el QA en equipos de desarrollo para las (etapa 2 y 5 en color rojo). En la figura 5.7 se muestran las fases para garantizar un desarrollo móvil con calidad.

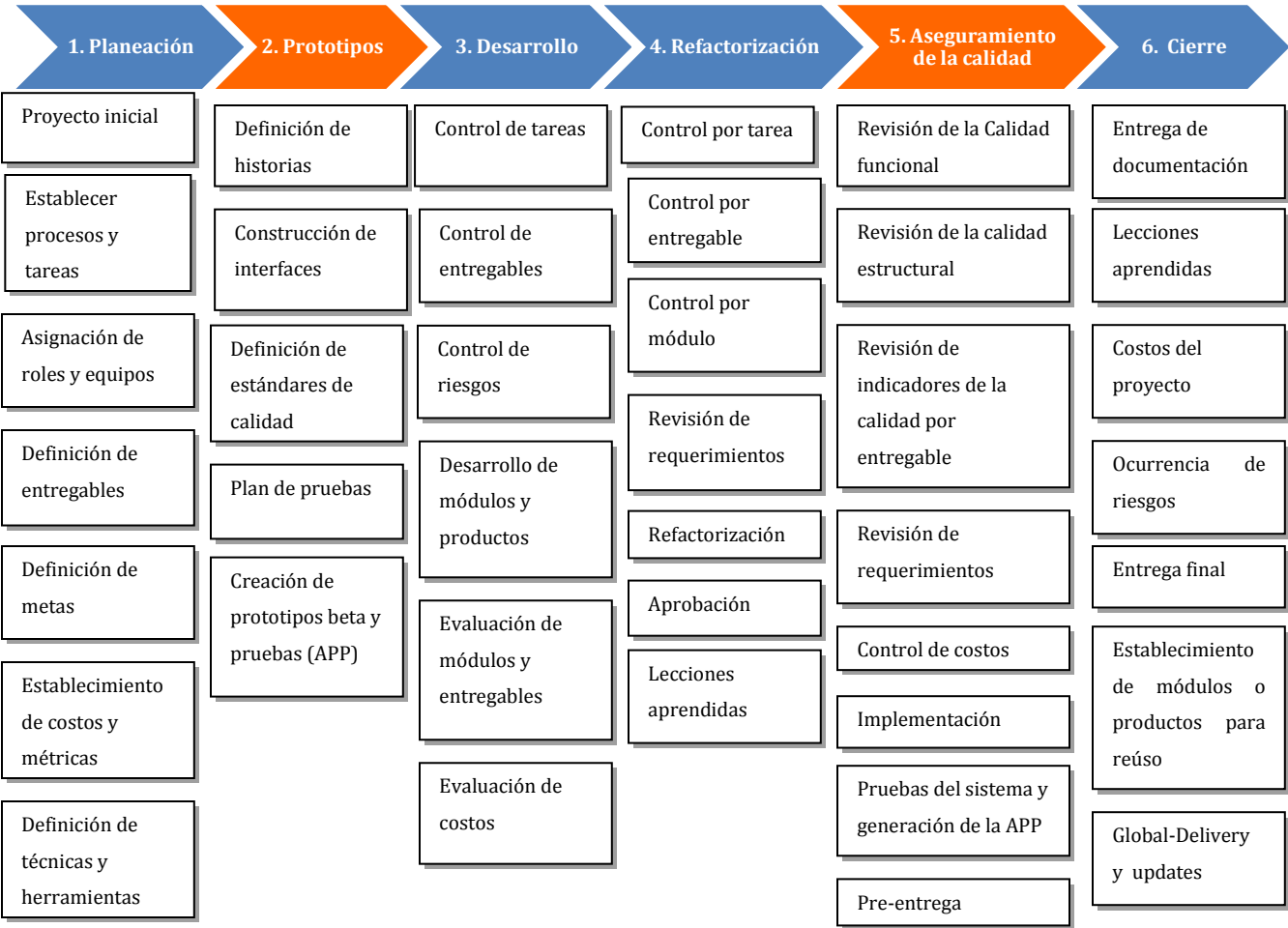


Figura 5.7. Fases y elementos de la metodología MDSIC-M, basada en buenas prácticas de desarrollo.

Fuente: elaboración propia.

A continuación se describe cada uno de los elementos de la metodología por fases.

5.4.1 Fase de planeación.



Los elementos y actividades de este nivel se describen a continuación:

1. Proyecto inicial. Alineación de la TI y de las PyMEs con el proyecto, se establecen objetivos, requerimientos y los participantes en el proyecto.
2. Establecer procesos y tareas. Se construye el WBS en su etapa de desglose por actividades, tratando de ordenarlas por objetivos y entregables.
3. Asignación de roles y equipos de trabajo. Por cada uno de los requerimientos y tareas dos en WBS asignar los encargados de llevar a cabo cada una de estas.
4. Definición de entregables y productos. Establecer los entregables (requerimientos) del proyecto y reordenar las tareas de acuerdo a estos, establecer hitos en caso de ser necesarios.
5. Definición de metas. Describir los logros y productos finales del proyecto.
6. Establecimiento de costos y métricas. Se establecen los costos de cada una de las actividades con la finalidad de establecer un costo aproximado. Se definen las métricas que ayuda a la estimación del costo del proyecto. También se definen las métricas que requiere la APP para ser competitiva (ver tema 5.5).
7. Definición de técnicas y herramientas. De acuerdo a los productos y entregables definidos establecer las herramientas o técnicas necesarias para el desarrollo de la APP.

5.4.2 Fase de prototipos.



Los elementos de esta fase son:

1. Definición de historias. De acuerdo a los requerimientos de los clientes iniciar los bocetos, diagramas o guiones de las posibles interfaces o componentes de la aplicación.
2. Construcción de interfaces. Utilizando las herramientas definidas y las historias, es necesario construir las interfaces de acuerdo a los requerimientos del proyecto.
3. Definición de estándares de calidad. Establecer la definición de conceptos y productos, requerimientos del proceso de desarrollo, lineamientos para los procesos de implementación y los lineamientos para los procesos de evaluación.
4. Plan de pruebas. Es necesario definir las pruebas de acuerdo a las interfaces y componentes o entregables (que se puedan probar) definidos en el WBS.
5. Creación de prototipos beta y pruebas. Siguiendo los requerimientos y las historias de los usuarios, construir los prototipos funcionales que muestren al cliente las interfaces del proyecto.

5.4.3 Fase de desarrollo.



Los elementos de esta fase son:

1. Control de tareas. Ejecución y aseguramiento de los recursos de cada una de las tareas definidas en el WBS, garantizando su ejecución y contabilizando tiempos y costos.
2. Control de entregables. Censar continuamente el avance de los entregables mediante la ejecución de tareas. Establecer fecha de inicio y fin para cada una de las tareas.
3. Control de riesgos. Revisar y resolver problemas en la ejecución de tareas que presentaron riesgos. Contabilizar y agregar los costos para todos los riesgos y establecer nuevos tiempos y costos del proyecto.
4. Desarrollo de módulos y productos. Desarrollar utilizando las técnicas y herramientas establecidas, los módulos o productos que son partes de los entregables del proyecto.
5. Evaluación de módulos y entregables. Revisar los KPI para cada uno de los artefactos, módulos y entregables establecidos en WBS del proyecto.

6. Evaluación de costos. Establecer los costos incurridos por el desarrollo de los artefactos, productos o módulos, revisar la ocurrencia de riesgos y costos extraordinarios.

5.4.4 Fase de refactorización.



Los elementos de esta fase son:

1. Control por tarea. Censar cada una de las tareas y en caso de detectar errores es necesario refactorizar cada una, se notica al administrador y debe agregarse costos extraordinarios.
2. Control por entregables. Censar cada uno de los entregables y en caso de error enviar a refactorizar, revisando los riesgos.
3. Control por módulo. Censar cada uno de los módulos y en caso de error enviar a refactorizar.
4. Revisión de requerimientos. Censar cada uno de los entregables, productos, módulos acordes a los requerimientos establecidos, en caso de detectar errores enviar a refactorizar.
5. Refactorización. Cada uno de los productos, tareas, módulos o entregables deben reprogramarse para entrar en la fase de desarrollo, se contabilizan aquellos elementos que fueron refactorizadas y se calcula los nuevos costos.
6. Aprobación. Cada elemento que es refactorizado debe ser monitoreado por el equipo de QA y establecer si esta correcto o sin errores. Si es aprobado se liberan los recursos.
7. Lecciones aprendidas. Guardar la información de cada uno de los componentes refactorizadas y establecer los motivos de errores para un aprendizaje.

5.4.5 Fase de Aseguramiento de la calidad.



Los elementos de esta fase son:

1. Revisión de la calidad funcional (QAF). Se revisan los requerimientos y el estado de la aplicación para: desechar defectos o basura en su desarrollo, se mide su desempeño con los KPI, se censa si es fácil de aprender, de utilizar y confiable.
2. Revisión de la calidad estructural (QAs). Se revisa para el desarrollo de la aplicación el código desarrollado: si es fácil de probar, que sea de fácil mantenimiento, entendible, eficiente y seguro.
3. Revisión de los indicadores de la calidad por entregable. Se revisa cada uno de los KPI para cada uno de los procesos establecidos en el desarrollo de la APP. Se revisa la calidad por proceso (QAp) que genera entregables.
4. Revisión de requerimientos. Se censa que los entregables estén alineados con las necesidades del cliente y los KPI sean los adecuados.
5. Control de costos. Se verifica el costo actual del proyecto contra el costo estimado en la planeación y en caso de costos extraordinarios documentar las razones.
6. Implementación. Se establece el plan para implementar la aplicación desarrollada una vez que se ha censado su calidad. Se genera la APP para pruebas.
7. Pruebas del sistema. Una generada la App se aplica el plan de pruebas establecido en la etapa de planeación. Se realizan pruebas de integración e implementación.
8. Pre-entrega, se genera la APP revisada y se coloca en una tienda en línea para su posible liberación y censar la compatibilidad, se hacen pruebas de descarga y ejecución (pruebas del sistema).

5.4.6 Fase de cierre.



Los elementos de esta fase son:

1. Entrega de la documentación. Se generan los manuales técnicos, de usuario y la documentación del seguimiento de cada uno de los procesos.
2. Lecciones aprendidas. Se sintetiza todos los elementos del desarrollo de una APP, se establece que componentes se pueden reutilizar y se documenta los riesgos y costos finales.
3. Costos del proyecto. Se revisa el costo proyectado contra el costo real de la aplicación, se almacenan las lecciones aprendidas.
4. Ocurrencia de riesgos. Se contabilizan los riesgos en los que incurrieron las tareas, los errores y los elementos refactorizadas.
5. Entrega final. Se libera la APP en la tienda en línea y se documenta su aceptación entre los clientes.
6. Establecimiento de módulos o productos para reuso. Se documentan aquellos componentes, productos, artefactos o módulos para su reuso.
7. Finalmente se establece un plan de actualizaciones que llevará a desarrollar la APP con nuevas características y métricas.

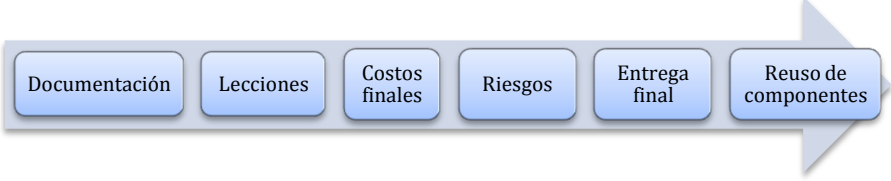
El reúso de componentes se considera un tema de vanguardia entre los desarrolladores y se garantiza en la metodología. En el tema siguiente se definen los procesos genéricos para cada etapa de la MDISC-M, se colocan los KPIs que se deben censar en cada uno como criterio de calidad y nivel de madurez.

5.5 Procesos para verificación de la calidad mediante los KPI.

Es importante para MDSIC-M el establecimiento de procesos que ayudan a clarificar y censar cada una de las KPI en el desarrollo de una APP. En la tabla 5.3 se describen cada uno de los procesos y subprocesos que se ejecutan en la metodología para validar la calidad. Se describen los KPI que ayudan a esta validación.

Tabla 5.3. Procesos y subprocesos de la metodología MDSCI-M.
Fuente: elaboración propia.

Proceso	Subprocesos en MDSIC-M	KPI
Planear		Tiempo. Costo. Riesgos. Métricas.
Prototipado		Tiempo. Desempeño. Pruebas. Satisfacción.
Desarrollar Aplicación		Riesgos. Costo. Tiempo. Equipos. Roles. Refactorización.
Refactorizar		Costo. Riesgos. Tiempo. Aprendizaje. No. Elementos.
Asegurar la Calidad		Nivel de madurez. Revisión. Costo. Tiempo.

		Elementos terminados.
Cerrar el proyecto	 El diagrama muestra una secuencia de procesos de cierre de proyecto: Documentación, Lecciones, Costos finales, Riesgos, Entrega final, y Reuso de componentes. Cada proceso está en un recuadro azul, y una flecha grande azul los conecta en orden de izquierda a derecha.	Reuso. Costo. Riesgos. Tiempo. Aprendizaje.

En el Anexo B, se describen cada unos de los procesos y subprocesos propuestos.

5.5.1 Establecimiento de Métricas y KPIs.

Es importante diferenciar en los KPI y las métricas para el diseño de APPs. La Biblioteca de Infraestructura de Tecnologías de Información (ITIL), es un conjunto de conceptos y prácticas para la gestión de servicios de tecnologías de la información, el desarrollo de tecnologías de la información y las operaciones relacionadas con la misma (ITIL Foudation, 2014).

De acuerdo a la definición de ITIL “No se puede mejorar aquello que no se conoce y no se puede llegar realmente a conocer aquello que no se puede medir”. Es indispensable que la organización de TI defina una serie de métricas que permitan determinar si se han alcanzado los objetivos propuestos así como la calidad y rendimiento de los procesos y tareas involucrados. ITIL recomienda que una organización de TI deba utilizar tres tipos de métricas:

1. Tecnológicas: que miden la capacidad, disponibilidad y rendimiento de las infraestructuras y aplicaciones.
2. De procesos: que miden el rendimiento y calidad de los procesos de gestión de los servicios TI.
3. De servicios: que evalúan los servicios ofrecidos en términos de sus componentes individuales.

Las métricas deben adaptarse a los Factores Críticos de Éxito (CSF) que describen aquello que “debe pasar” para que se cumplan los objetivos preestablecidos. Asociados a cada CSF es necesario definir una serie de Indicadores Críticos de Rendimiento (KPI) que permitan evaluar el

rendimiento y la calidad de los procesos así como su valor y adecuación. De tal forma que los KPIs deben medir aspectos cualitativos y cuantitativos y deben permitir evaluar el cumplimiento de los objetivos.

Es importante que los KPIs no obvien aspectos clave y que su cumplimiento sea una medida objetiva del cumplimiento de los CSF asociados. De aquí surgen una serie de métricas que se pueden convertir en KPI (no todas). Los CSF de MDSIC-M se describen a continuación:

1. Mejorar el proceso de planeación de una APP utilizando el WBS y un scheduler.
2. Mostrar resultados tangibles lo más rápido posible mediante la generación de prototipos.
3. Ayudar al proceso de desarrollo de una aplicación mediante el establecimiento de equipos de trabajos colaborativos y el control de tareas.
4. Controlar los errores de desarrollo mediante el establecimiento y el control de tareas y aquellas que serán enviadas a refactorizar.
5. Medir continuamente el avance de los entregables, productos y artefactos generados en la etapa de desarrollo, mediante el establecimiento de estándares de calidad.
6. Almacenar y sintetizar las experiencias en el desarrollo de una aplicación, de tal forma que sea posible establecer las lecciones aprendidas y los componentes de reuso para nuevos proyectos.

En (Adler, 2014) se mencionan ocho métricas que los desarrolladores de aplicaciones móviles deben tomar en cuenta para competir en el mercado, a continuación se mencionan:

1. Tipo de usuario. Permite conocer a que segmento de audiencia que estará dirigida la aplicación, niños, adultos en general, también se incluye la capacidad económica del usuario, tiene que ver con el comportamiento del usuario.
2. Longitud de sesión. Mide el período de tiempo que la APP es accesada por el usuario; apertura y cierre, puede ser medida por tipo de usuario. Entre más tiempo se utilice será más viable la adquisición por parte del usuario (freemium).

3. Intervalo de sesión. Mide el tiempo entre sesiones del usuario, la frecuencia con que el usuario la abre. Ayudar a conocer el grado de uso de una APP y la preferencia de los usuarios.
4. Tiempo en APP. Mide el tiempo en el que el usuario utiliza la APP, el número de minutos o horas que el usuario permanece en la APP
5. Adquisición. Mide el número de usuarios que descargan e instalan la APP y permite conocer el ROI y el tipo de usuario
6. Pantalla. Rastrea el flujo entre pantallas de la APP; salidas por pantalla, fluir entre pantallas y las ocurrencias del total de visitas a cada pantalla. Permite conocer las interacciones típicas de usuarios en la APP.
7. Retención. Mide el porcentaje de usuarios que regresan a utilizar la aplicación en base a las fechas de descarga y ejecución. Muy útil para medir la tasa de retención de usuarios.
8. Valor de por vida. Representa el valor financiero de la APP, establece la cantidad de usuarios que la consideran una aplicación útil de por vida. Permite establecer ingresos por el número de clientes cautivos.

También se describen otras métricas que definen el mercado de una APP, entre otras son: uso de internet, tipo de sistema operativo, uso de conectividad, plataforma de desarrollo, tipo de dispositivo, costo del dispositivo, número de actualizaciones, entre otras. En (Estorach, 2014) se describe que las métricas ayudan a entender el como la APP está siendo utilizada por los usuarios y su uso varía mucho de acuerdo a la funcionalidad de la APP. En la figura 5.8 se muestran los tipos de APPs agrupadas por funcionalidad.

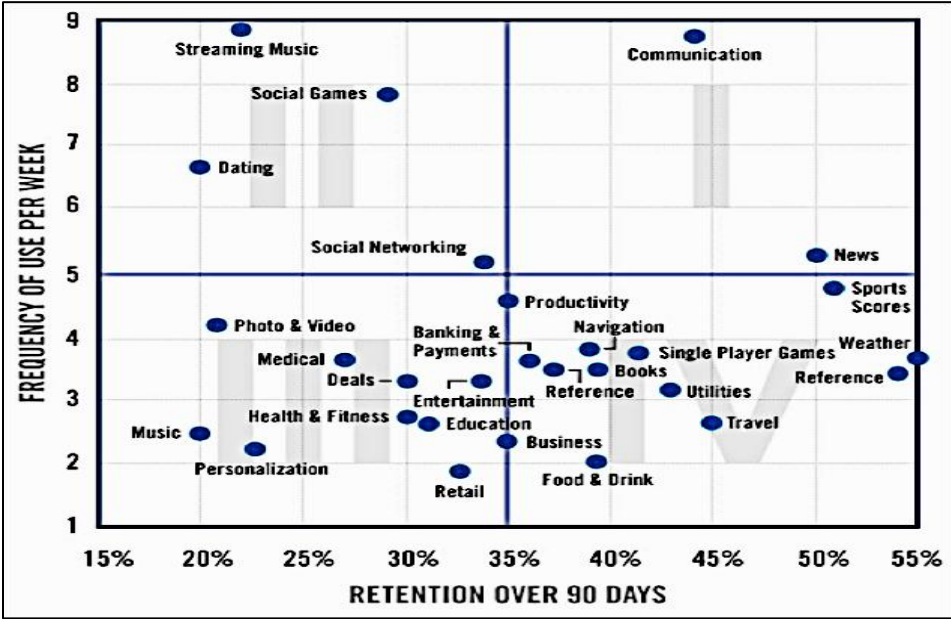


Figura 5.8. Frecuencia de uso de una APP y su categoría.
Fuente: (Estorach, 2014) .

5.5.2 KPIs en MDSIC-M.

Para cada CSF de MDSIC-M se establecen los KPI que permitirán censar la calidad de cada proceso propuesto, tomando en cuenta el desarrollo de una APP. En la tabla 5.4 se describe cada KPI de MDSIC-M asociados a los CSF.

Tabla 5.4. KPIs para establecer la calidad mediante factores de éxito.
Fuente: elaboración propia.

CSF (Proceso)	KPI	Descripción
1. Planear	Tiempo	Representa el tiempo que se tarda en realizar la planeación de un proyecto, debe establecerse con claridad
	Costo	Representa el costo de desarrollo de la aplicación, debe tomar en cuenta la métricas de la APP para el cálculo del ROI
	Riesgo	Representa el porcentaje de ocurrencia de la falta de alguno de los elementos de desarrollo de la APP. El costo final deberá incluir el costo de cada uno de los riesgos ocurridos durante el proyecto
	Métricas	Deben tomarse en cuenta la métricas que definirán el mercado de la APP; usuarios, uso, adquisición, retención, etc.

2. Elaborar Prototipos	Tiempo	Representa el tiempo en que se desarrollarán prototipos para que representen los requerimientos del usuario; debe establecerse claramente para no incurrir en costos extras
	Desempeño	Representa el grado de satisfacción del cliente al generar un versión de un prototipo, el prototipo se apeg a los requerimientos de los clientes y muestra un GUI esperada
	Pruebas	Representa el número de pruebas realizadas a las Pantallas (GUI) de la APP por parte del cliente
3. Controlar el desarrollo	Riesgos	Representa el número de riesgos que ocurrieron durante el desarrollo de un producto, artefacto, módulo o entregable. Se establecen por tarea y se contabilizan para el costo final del proyecto
	Costo	Representa el costo extraordinario del desarrollo que difiere del costo planeado, esto se debe a la refactorización
	Tiempo	Representa el tiempo de desarrollo de cada una de las tareas que generan un entregable, artefacto, producto o módulo
	Equipos	Representa el personal (con diferentes roles) que intervienen en el desarrollo de una artefactos, productos, módulos o entregables
	Roles	Representa los elementos que participan en el desarrollo de un artefacto, producto, módulo o entregable. Estos deben firmar para validar el desarrollo
4. Refactorizar componentes, artefactos	Costo	Representa el costo extraordinario por refactorización de algún componente
	Riesgo	Representa el número de riesgos que ocurrieron durante la refactorización
	Tiempo	Representa el tiempo de refactorización de un componente con errores
	Aprendizaje	Describe el motivo de la refactorización de un componente y ayuda al aprendizaje para evitar errores
5. Asegurar la calidad	Madurez	Representa el grado de madurez de cada uno de los procesos que puede ser (NYCE, 2011), (Myung H., 2009); 0: Incompleto (no se logran los objetivos), 1: Realizado (desarrollo con poco control) 2: Gestionado (se logran tiempos y se cumplen requisitos) 3: Establecido (documentados y logran estándares) 4: Predecible (mediciones y bien planeado) 5: Optimizado (mejoras y mediciones)

	Revisión	Representa el número de veces que censó la calidad de un artefacto, producto, módulo o entregable
	Costo	Representa el costo extraordinario para medir la calidad de un artefacto, producto, módulo o entregable
	Tiempo	Representa el tiempo de realización de las tareas comparándolo con la planeación
6. Almacenar lecciones aprendidas y cierre del proyecto	Reuso	Representa el número de componentes que son factibles de reuso en proyectos subsecuentes
	Costo	Establece la diferencia entre planeado y el costo final del proyecto
	Riesgo	Establece los riesgos que ocurrieron en el desarrollo del productos
	Tiempo	Establece el tiempo de finalización el proyecto
	Aprendizaje	Almacenamiento de la métricas del proyecto, almacena el número de: requerimientos, tareas, componentes, prototipos, entregables, costos, métricas de la APP, equipos, roles, elementos de QA, componentes refactorizadas, etc.

Como se observa la utilización de un desarrollo basado MDSIC-M permite el control de riesgos, ayuda a un desarrollo con compatibilidad; esto permite la entrega de los productos a través de tiendas estándares. La utilización de métricas ofrece a una ventaja competitiva al permitir establecer los recursos y costos de cada componente. Además ayuda a garantizar un desarrollo apegado al presupuesto proyectado y solo los riesgos podrán cambiar el escenario final. El uso de estándares y el almacenamiento de las lecciones aprendidas permiten establecer un nivel de madurez en los procesos. Es una propuesta de desarrollo ágil que ayuda a las PyMEs a adoptar un método simple y eficaz.

5.6 Medición de la calidad.

En el mercado actual, hay modelos de madurez, estándares y pautas que pueden ayudar a una organización a mejorar la forma en que hace negocios. Modelos de evaluación de procesos de software, como ISO/IEC 15504 y CMMI proporcionan una buena estrategia para evaluar la capacidad de desarrollo de software. Sin embargo, estos modelos son demasiado pesados para

aplicarse en PyMEs. La ISO/IEC 15504 para la mejora de procesos de software y la determinación de capacidades CMM/CMMI se consideran como modelos de evaluación de procesos de software; ayudan a los evaluadores a asignar calificaciones de acuerdo a los indicadores y métricas para medir la capacidad de los procesos de software (Myung H., 2009). ISO/IEC 15504 y CMMI definen 5 niveles de capacidad de un proceso:

Nivel 0. Incompleto. Existe un fracaso general para alcanzar el propósito del proceso. Hay productos o resultados del proceso poco o nada identificables.

Nivel 1. Realizado. El propósito del proceso se consigue generalmente. Existe poco control y no está optimizado.

Nivel 2. Gestionado. El proceso entrega los productos de acuerdo con los procedimientos especificado, se planifica y se da seguimiento. Los productos de trabajo se ajustan a las normas y los requisitos especificados.

Nivel 3. Establecido. El proceso se lleva a cabo y es gestionado mediante un proceso definido en base a los principios de la ingeniería de software. Implementaciones individuales del uso proceso aprobado, versiones adaptadas de los procesos estándar, documentados para lograr los resultados del proceso.

Nivel 4. Predecible. El proceso definido se realiza sistemáticamente en la práctica dentro de los límites de control definidos, para lograr sus metas de proceso definidas.

Nivel 5. Optimizado. El rendimiento del proceso está optimizado para satisfacer las necesidades de negocio actuales y futuras, y el proceso consigue ser repetible en el cumplimiento de los objetivos de negocio definidos.

Se describe en Myung (2009) que todo proceso debe tener un guía para la certificación basada en un grupo de calidad quiénes deben regir la evaluación. En la figura 5.9 se muestra la estructura.

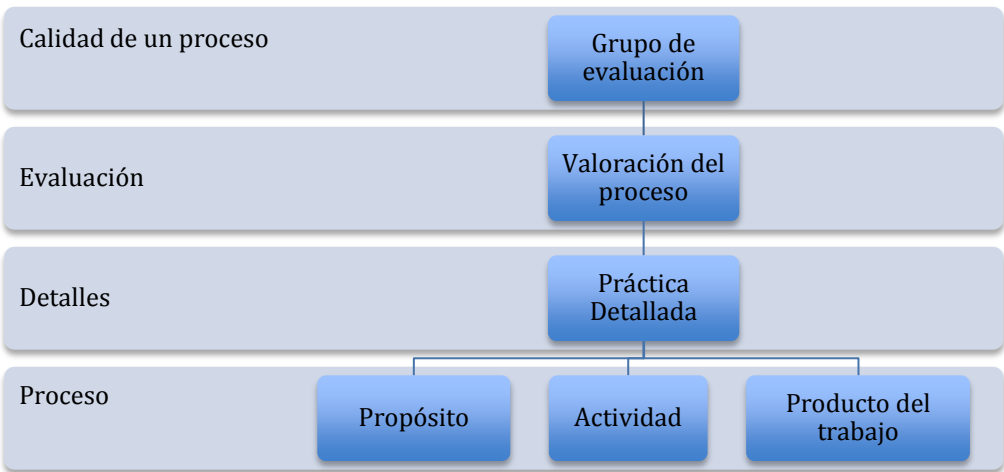


Figura 5.9. Estructura de la evaluación de un proceso.
Fuente: (Myung H., 2009).

Cada proceso puede tener 3 niveles de certificación que permite conocer con exactitud el nivel de madurez, en la figura 5.10 se muestran.

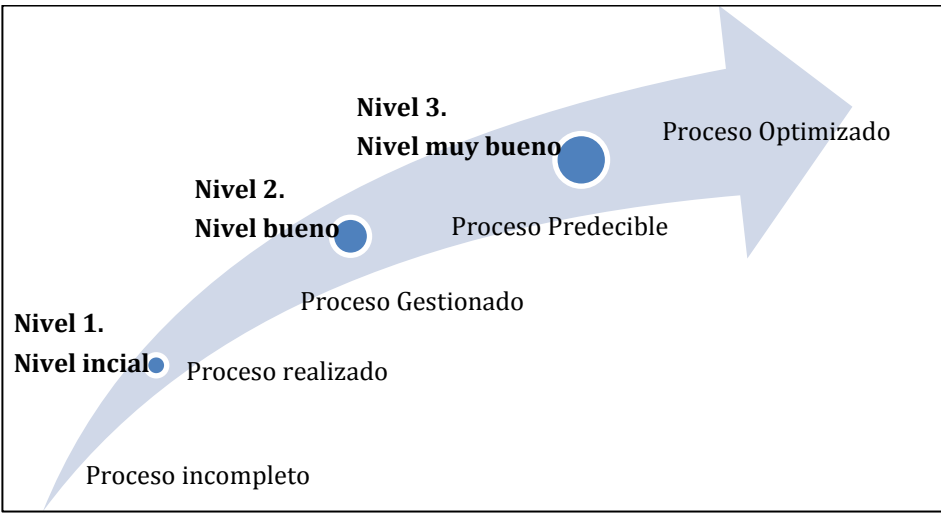


Figura 5.10. Estructura del grado de certificación de la calidad de un proceso.
Fuente: Myung (2009).

CMMI define los niveles de madurez de un proceso para el desarrollo de software, basándose en las áreas de proceso claves (KPAs) y que se asocian con los objetivos de un desarrollo con calidad.

En la figura 5.11 se muestran los KPAs que deben lograrse en cada nivel de madurez de acuerdo al CMMI.

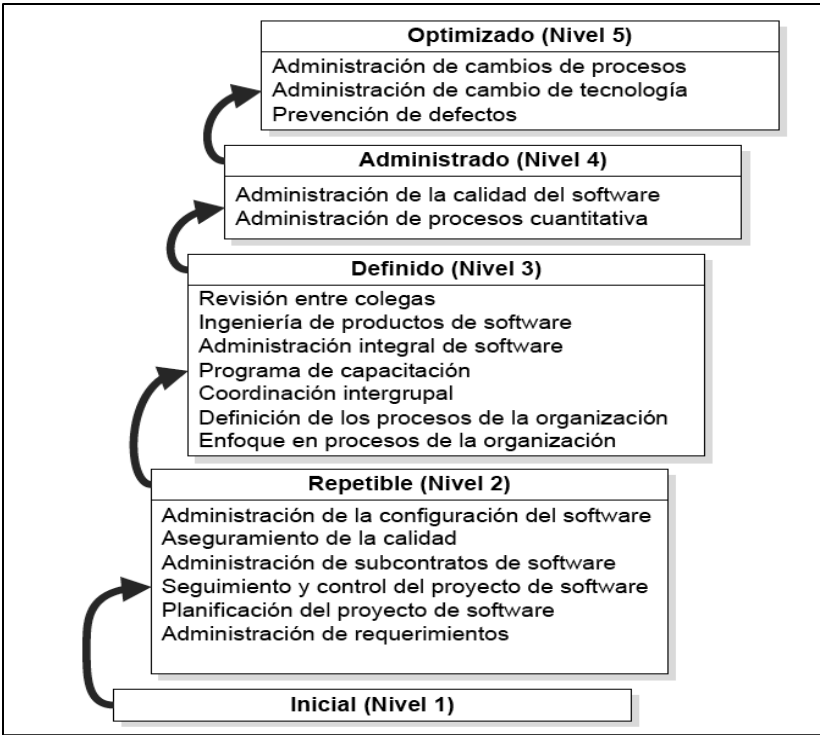


Figura 5.11. Nivel de madurez de un proceso basado en KPAs.
Fuente: (CMMI, 2015).

5.6.1 Otros modelos para medir la calidad.

En (Rueda G., 2010) se describen una serie de modelos para validar la calidad en el desarrollo de software, de los modelos descritos sobresalen los siguientes, ver tabla 5.5.

Tabla 5.5 Modelos para validar la calidad en el desarrollo de software.
Fuente: Rueda G. (2010).

Modelo	Descripción	Aplicado a
IT Mark	Diseñado en el Instituto de Software Europeo (ESI). Evalúa a la empresa en 3 áreas; 1) gestión, 2) seguridad y 3) madurez de procesos. Se basa en la ISO/IEC y CMMI.	Calidad en empresas y sus procesos.

SwTQM <i>Software Total Quality Management</i>	Es una iniciativa de la <i>European Foundation for Quality Management</i> (EFQM). Busca la excelencia de las empresas en calidad, se basa en CMMI y gestión de la calidad europea para empresas de TI y desarrollo.	Excelencia en calidad para empresas europeas.
TickIT	Introducido por el Consejo Nacional de Acreditación de los Organismos de Certificación (NACCB) del reino unido. Se utiliza para el análisis y especificación de los requerimientos, para la planificación, control y supervisión del desarrollo de software. Se basa en ISO 9001.	Especificación de requerimientos y control del desarrollo.
PSP	Propuesto por el SEI. Dirigido a validar las prácticas de trabajo en forma individual y llegar a la producción de un software de calidad. Establecimiento de un proceso de desarrollo bien definido, medible y apegado a metas.	Enseñanza en la planeación y seguimiento de capacidades personales.
TSP	Propuesto por el SEI. Define un conjunto de procesos para integrar equipos de trabajo, que planeen y registren su trabajo. Ayuda a los gerentes a motivar y monitorear a los equipos de trabajo. Tiene 4 fases; requerimientos, diseño, implementación y pruebas.	A equipos de trabajo para medir su calidad en el desarrollo.

Se han documentado otros modelos para censar la calidad pero estos se enfocan más en las buenas prácticas orientadas a la administración de proyecto como son ITIL, COBIT y el PMI.

5.6.2 El modelo de calidad de desarrollo para metodologías ágiles.

En (Nathan-Regis & Nasira, 2013), se mencionan que el desarrollo basado en metodologías ágiles ofrecen varios forma de validar la calidad. La calidad consiste en dos temas:

- 1) la calidad consiste en aquellas características de los productos que satisfacen las necesidades de los clientes y por lo tanto proporcionan satisfacción y
- 2) consiste en generar productos sin deficiencias.

Otra forma de definir la calidad es basarse en los indicadores de McCall como correctitud, compatibilidad, eficiencia, portabilidad, integridad, verificable y de fácil uso. Acorde a esta definición también se agregan los indicadores propuestos por FURPS.

Desde la perspectiva ágil el desarrollo de software con calidad debe responder a un software en constante cambio, donde los elementos para validarla pueden ser cambiantes (por ejemplo los requerimientos). En la metodología XP las técnicas que se utilizan para proveer de calidad un desarrollo se pueden observar en la tabla 5.6.

Tabla 5.6 Técnicas para garantizar la calidad en métodos ágiles.

Fuente: (Nathan-Regis & Nasira, 2013).

Técnica	Descripción
Refactorización	Hacer pequeños cambios en el código, no debe verse afectado el comportamiento del sistema, resultando código de alta calidad.
Desarrollo basado en pruebas	Crear una prueba, ejecutar una prueba, hacer cambios hasta que la prueba pase.
Integración continua	Se hace una prueba de aseguramiento de calidad una vez terminado el sistema, implica la participación de los usuarios, patrocinadores, clientes, desarrolladores, etc.
Programación por pares	Se realiza a diario y después de elaborar una serie de historias del usuario, los requisitos están implementados y probados para ser verificados.
Comunicación cara a cara	Dos desarrolladores se turnan la programación en el mismo equipo, los bugs son identificados y corregidos.
En las instalaciones del cliente	Forma preferida de intercambio de información sobre un proyecto en oposición al uso de otro método de comunicación con el cliente. La implementación se hace en reuniones diarias de no más de veinte minutos que tratan de censar el avance con mutuo acuerdo.
Retroalimentación continua del cliente	Se basa en la comunicación continua del cliente que proporciona información sobre el sistema, y el resultado de uso para mejorarlo y estar más apegado a las necesidades del cliente. La calidad es en los requisitos del cliente.
Metáfora de sistema	Se construye una historia simple de cómo funciona el sistema, simplifica la discusión sobre el sistema entre el cliente/los interesados/usuario y el desarrollador en un formato no técnico. La simplicidad es la clave para la calidad.

5.6.3 Control de defectos para garantizar la calidad.

Una de la técnicas más utilizadas es el descubrimiento de los defectos en desarrollo de una aplicación, es necesario buscar mediante; la inspección de software, inspección de código, pruebas del software, eliminación de bugs (errores ocultos) y pruebas de integración. Muchos de estos procesos para garantizar la calidad son procesos manuales que generalmente tienen muchos problemas y debe ser el equipo de QA quién dirija estos esfuerzos. A continuación en la tabla 5.7 se colocan los principales defectos por nivel.

Tabla 5.7 Defectos de calidad por nivel.
Fuente: Nathan-Regis et. al. (2013).

Nivel	Defecto	Descripción
Código	<i>Code Smells</i>	“Olores en el código”, son indicadores para el refactoring que suelen tener un conjunto de alternativas de solución preestablecidas.
Arquitectura	<i>Architecture Smells</i>	“Olores en la arquitectura”, indicador de errores en el nivel de diseño, define patrones y anti patrones que ofrecen una solución para los errores en el diseño.
Patrones de error	<i>Bug Patterns</i>	Se localizan típicamente en las actividades de depuración y pruebas en los componentes, módulos o artefactos del software.
Diseño	Características de diseño	Generalmente se detectan durante el desarrollo y se detectan errores en el diseño orientado a objetos (DOO) donde más de una clase repiten funciones o tienen problemas de diseño.
Integración	Errores de comunicación	En la integración de componentes es común encontrar este tipos de defectos, la pruebas es la forma más común de detección.

El proceso de detección de defectos de calidad puede observarse en la ontología propuesta por Nathan-Regis et. al (2013). Ver figura 5.12.

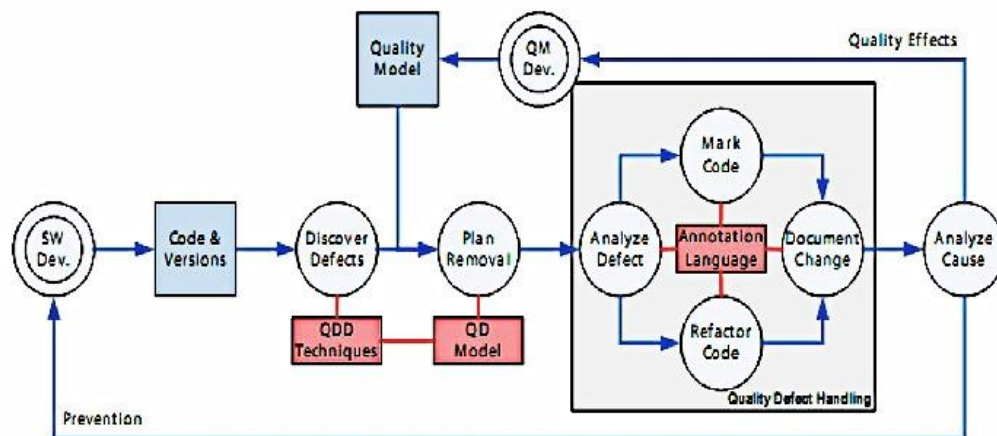


Figura 5.12. Ontología que describe el proceso de detección de defectos de calidad.
Fuente: Nathan-Regis et. al. (2013).

Para validar la calidad se clasificó los KPAs utilizando el modelo definido por Chappel (2014), que define el aseguramiento de calidad en tres elementos:

- 1) Calidad funcional (QAf); reuniones para requerimientos, desechar defectos o basura, buen desempeño, fácil de aprender y de usar.
- 2) Calidad estructural (QAs); código fácil de probar, código que sea de fácil mantenimiento, código entendible, código eficiente y código seguro.
- 3) Calidad en el proceso (QAp); reuniones para fechas de entregables, reuniones para garantizar presupuestos y desarrollo de procesos repetibles entregables alcanzables y calidad del software.

En la tabla 5.8 se clasifican los KPA de calidad propuestos en MDSIC-M y su aseguramiento de acuerdo a los elemento definidos por Chappel (2014).

Tabla 5.8. KPAs y sus correspondencias con el modelo de calidad de Chappel.
Fuente: elaboración propia.

KPAs para calidad	QA Funcional	QA Estructural	QA Proceso
Establecimiento de requerimientos	✓		
Construcción del Plan	✓		
Definición de métricas	✓	✓	
Control de entregables	✓		✓
Prototipo	✓	✓	
Control del tiempo		✓	✓
Revisión continua de costos		✓	✓
Control de riesgos		✓	✓
Control de tareas			✓
Refactorización	✓		
Reuso de componentes		✓	✓

En el capítulo 6 se muestra la aplicación de este modelo de calidad en proyectos elaborados con MDSIC-M.

5.7 Estándares y la NOM para la calidad en el desarrollo.

Para garantizar que MDSIC-M sigue los estándares de la NOM a continuación se establece con claridad cada uno de los elementos de los procesos. En (Flores R., Astorga V., Rodríguez E., Ibarra E., & del Carmen A., 2014) se explica que para cumplir con la NMX-I-059 (Moprosoft), se requiere un amplio entendimiento de sus requisitos. Por tal motivo, algunas organizaciones suelen contratar consultores para que las preparen y guíen en el proceso de implantación y verificación, dependiendo en gran medida del conocimiento que estos posean. La NMX-I-15504 (proceso de madurez) apoya a la organización para realizar evaluaciones consistentes y confiables, reduciendo la subjetividad para lograr resultados válidos y comparables a los de la verificación oficial. Sin embargo, existen personas que desconocen su contenido o la interpretan de forma incorrecta. Siguiendo el Estándar NMX-059-NYCE-2005 se establece:

01: DEFINICIÓN DE CONCEPTOS Y PRODUCTOS. Aclarar cada uno de los elementos que conforman el proyecto.

02: REQUISITOS DE PROCESO (Moprosoft). Se define en 2 partes:

- I. Presenta los elementos normativos por cada uno de los 9 procesos:
 1. Gestión de Negocio (GN), responsable del proceso (RGN)
 2. Gestión de Procesos (GPR), responsable del proceso (RGP)
 3. Gestión de Proyectos (GPY), responsable del proceso (RGPY)
 4. Gestión de Recursos (GR), responsable del proceso (RGR)
 5. Recursos Humanos y Ambiente de Trabajo (RHAT), responsable del proceso (RRHAT)
 6. Bienes, Servicios e Infraestructura (BSI), responsable del proceso (RBSI)
 7. Conocimiento de la Organización (CO), responsable del proceso (RCO)
 8. Administración de Proyectos Específicos (APE), responsable del proceso (RAPE)
 9. Desarrollo y Mantenimiento de Software (DMS), responsable del proceso (RDMS)
- II. Atributos de Proceso (AP). Descripción de Categoría, Proceso, Propósito, Objetivo(s), Actividades (Prácticas Base - PB) y Producto de Trabajo (PT). Además se debe describir desde el nivel 1 hasta el nivel 5 la capacidad de cada proceso. Ver en la figura 5.13 el ejemplo de los AP para nivel 1 y 2 de capacidad.

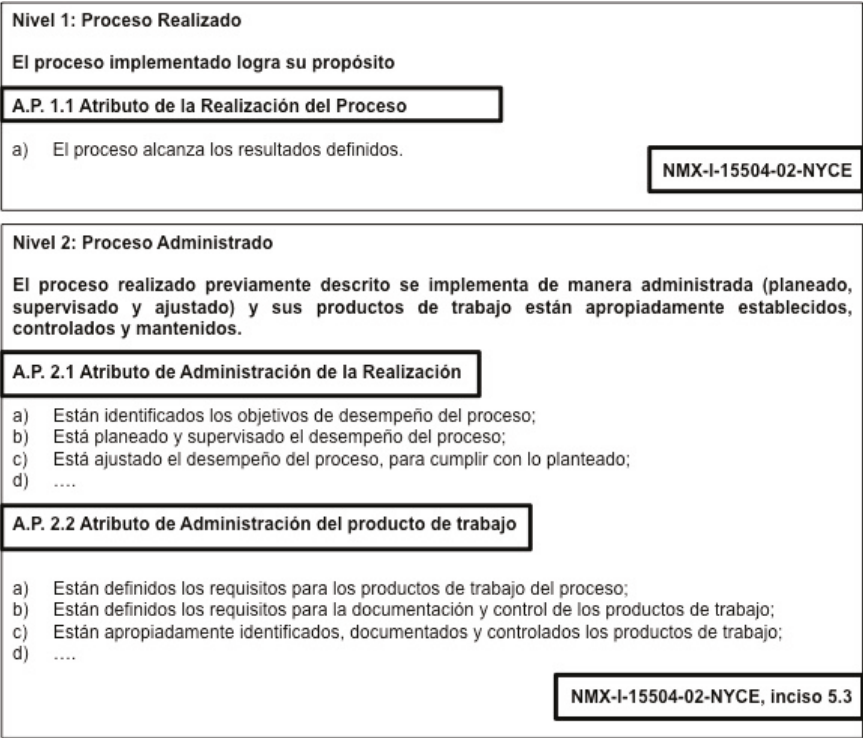


Figura 5.13. Ejemplo de los atributos de una proceso para el nivel de capacidad 1 y 2.
Fuente: (Flores R., Astorga V., Rodríguez E., Ibarra E., & del Carmen A., 2014).

Además para cada proceso se hace su validación de madurez, ver figura 5.14.

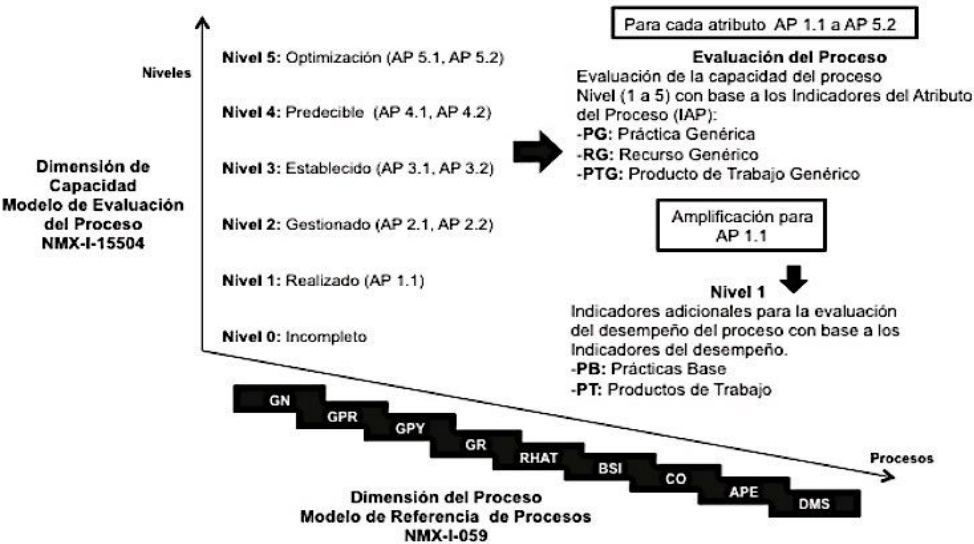


Figura 5.14 Especificación del proceso y de su capacidad NMX.
Fuente: Flores R. et. al. (2014).

03: GUÍA DE IMPLANTACIÓN DE PROCESOS. Descripción de cómo se implementarían.

04: DIRECTRICES PARA LA EVALUACIÓN DE PROCESOS (EvalProSoft). La evaluación de la capacidad considera un conjunto de AP. Para cada AP, la medida de capacidad se basa en la medición de los Indicadores de Atributo de Proceso (IAP) que integran Prácticas Genéricas (PG), Recursos Genéricos (RG) y Productos de Trabajo Genéricos (PTG). Para evaluar el logro del AP es necesario asignarle una calificación en una escala porcentual de medición, con base en los IAP y su ampliación de nivel 1 a partir de los ID. En la figura 5.15 se describe.

Calificación		Descripción	Porcentaje
N	No alcanzado	Existe poca o nula evidencia del cumplimiento del AP del proceso evaluado	0-15%
P	Parcialmente alcanzado	Existe evidencia de una aproximación, o se ha alcanzado parcialmente el AP del proceso evaluado	>15% hasta 50% del alcance
A	Ampliamente alcanzado	Existe evidencia de una aproximación sistemática, o ya se ha alcanzado hasta cierto grado el AP del proceso evaluado	>50% hasta 85% del alcance
C	Completamente alcanzado	Existe evidencia de una aproximación sistemática y completa, o ya se ha alcanzado hasta cierto grado el AP del proceso evaluado	>85% hasta el 100% del alcance

Figura 5.15. Evaluación de la capacidad de un proceso y sus atributos.

Fuente: Flores R. et. al. (2014).

Para estar acordes a la NMX-I-15504 a continuación se coloca una tabla con los KPI y los procesos definidos en MDSIC-M, además de su analogía con la NMX-I-15504 (NMX-059).

Tabla 5.9. Mapeo de los procesos y KPIs de MDSIC-M hacia el estándar NMX-I-15504 para la evaluación.

Fuente: elaboración propia.

Proceso	KPI	Proceso NMX-I-155054	Evaluación (IAP) NMX-I-15504
1. Planear	Tiempo	GPY, APE	RG, evaluación; N y C.
	Costo	GR	RG, evaluación; N, P, A y C.
	Riesgo	GR	PTG, evaluación; N y C.
	Métricas	DMS	RG, evaluación; N y C.
2. Elaborar Prototipos	Tiempo	GPR	RG, evaluación; N y C.
	Desempeño	DMS	PTG, evaluación; N, P, A y C.
	Pruebas	DMS	PG, evaluación; N, P, A y C.
3. Controlar el desarrollo	Riesgo	GR, GPY	PTG, evaluación; N y C.
	Costo	GR, DMS	RG, evaluación; N, P, A y C.

	Tiempo	GPR	RG, evaluación; N y C.
	Equipos	RHAT	RG, evaluación; N, P, A y C.
	Roles	RHAT	PG, evaluación; N, P, A y C.
4. Refactorizar componentes, artefactos	Costo	GPY	RG, evaluación; N, P, A y C.
	Riesgo	GR, DMS	PTG, evaluación; N y C.
	Tiempo	GPR, DMS	RG, evaluación; N y C.
	Aprendizaje	APE	PG, evaluación; N, P, A y C.
5. Asegurar la calidad	Madurez	APE	PTG, evaluación; N, P, A y C.
	Revisión	DMS	PG, evaluación; N, P, A y C.
	Costo	APE	RG, evaluación; N, P, A y C.
	Tiempo	GPY	RG, evaluación; N y C.
6. Cerrar proyecto	Reuso	DMS	PG, evaluación; N, P, A y C.
	Costo	GR	RG, evaluación; N, P, A y C.
	Riesgo	GR	PTG, evaluación; N y C.
	Tiempo	GPY	RG, evaluación; N y C.
	Aprendizaje	DMS	PG, evaluación; N, P, A y C.

Estos procesos en su evaluación deberán cumplir con los atributos marcados por la NMX-059, permitiendo la evaluación de los niveles de madurez de cada uno. En los temas siguientes se muestran los procesos y subprocesos de MDSIC-M y su evaluación de madurez.

5.8 Modelo de Madurez para MDSIC-M.

De acuerdo a la NMX-059 cada proceso debe ser evaluado en sus atributos e indicadores de calidad, se establecen desde el nivel 0: Proceso incompleto hasta el nivel 5 proceso optimizado. Cada proceso es evaluado en sus KPIs con la clasificación; **N**: no alcanzado (0-15 % avance), **P**: parcialmente alcanzado (15-50 %), **A**: ampliamente alcanzado (50-85 %) y **C**: completamente alcanzado (100 %). En la figuras 5.16, 5.17, 5.18, 5.19, 5.20 y 5.21 se muestran los niveles de madurez de los procesos y los subprocesos de MDSIC-M en la evaluación de calidad. Esta evaluación se apega a los atributos y procesos definidos en la NMX-059.

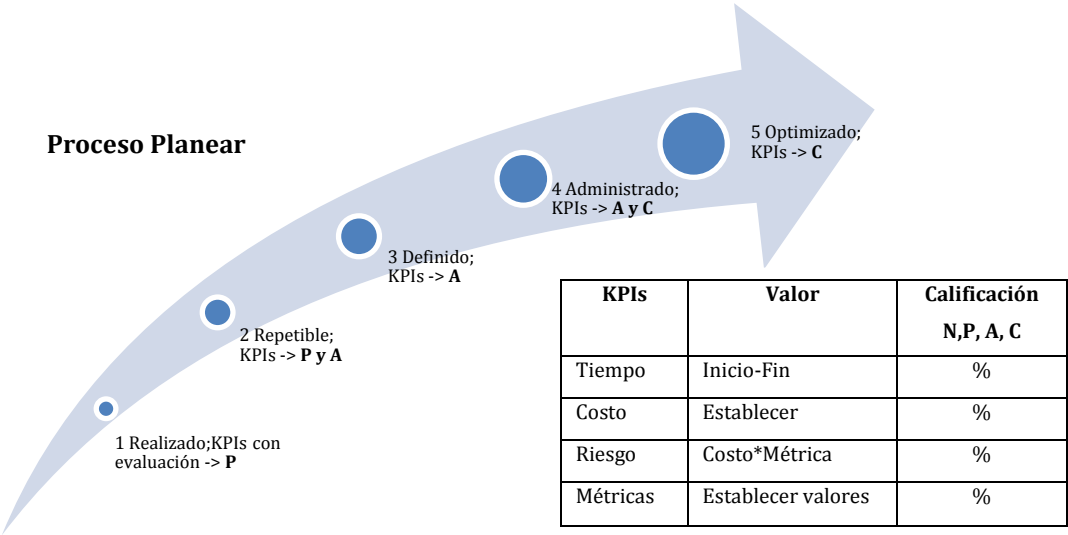


Figura 5.16 Establecimiento del nivel de madurez del proceso Planear y sus KPIs.
Fuente: elaboración propia.

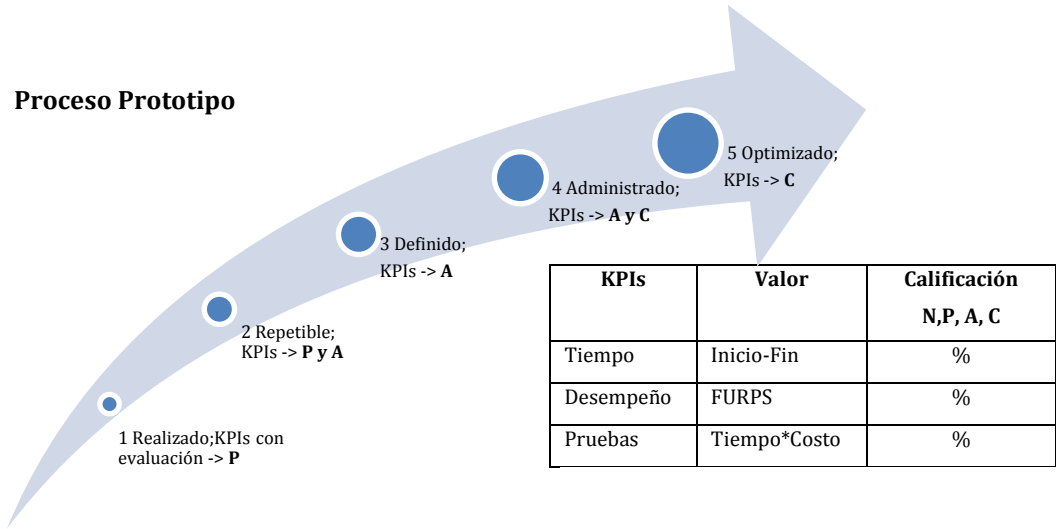


Figura 5.17 Establecimiento del nivel de madurez del proceso Prototipo y sus KPIs.
Fuente: elaboración propia.

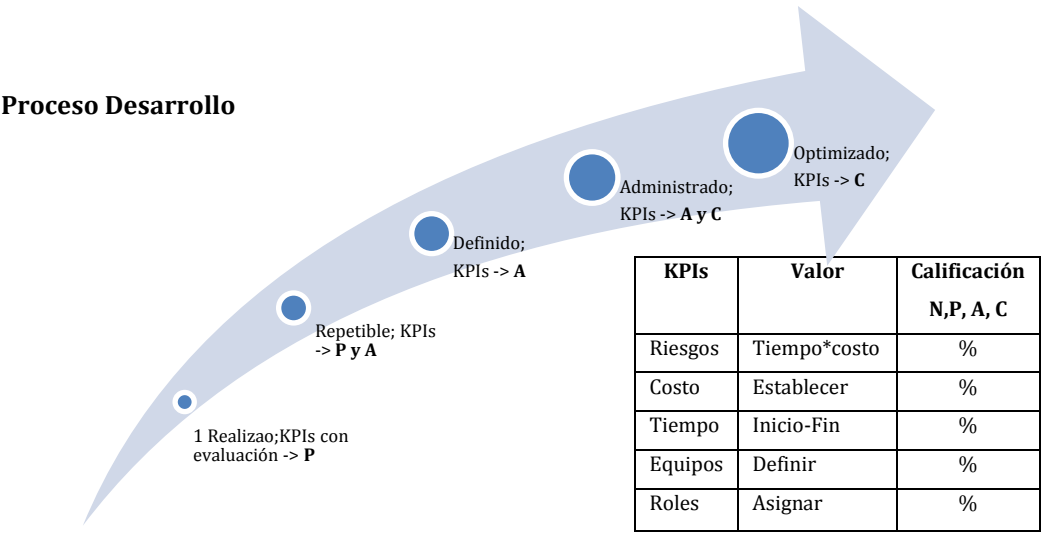


Figura 5.18 Establecimiento del nivel de madurez del proceso Desarrollo y sus KPIs.
Fuente: elaboración propia.

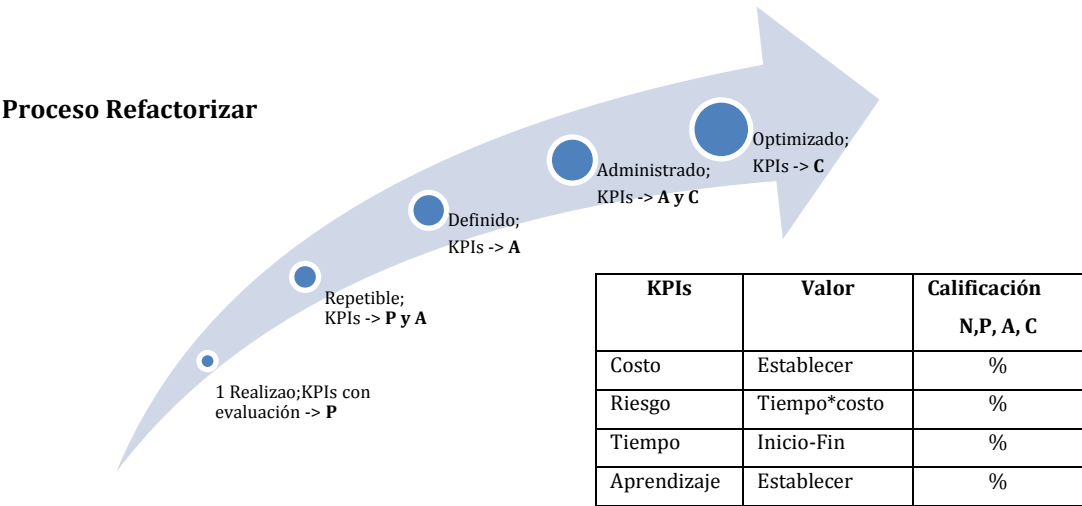


Figura 5.19 Establecimiento del nivel de madurez del proceso Refactorizar y sus KPIs.
Fuente: elaboración propia.

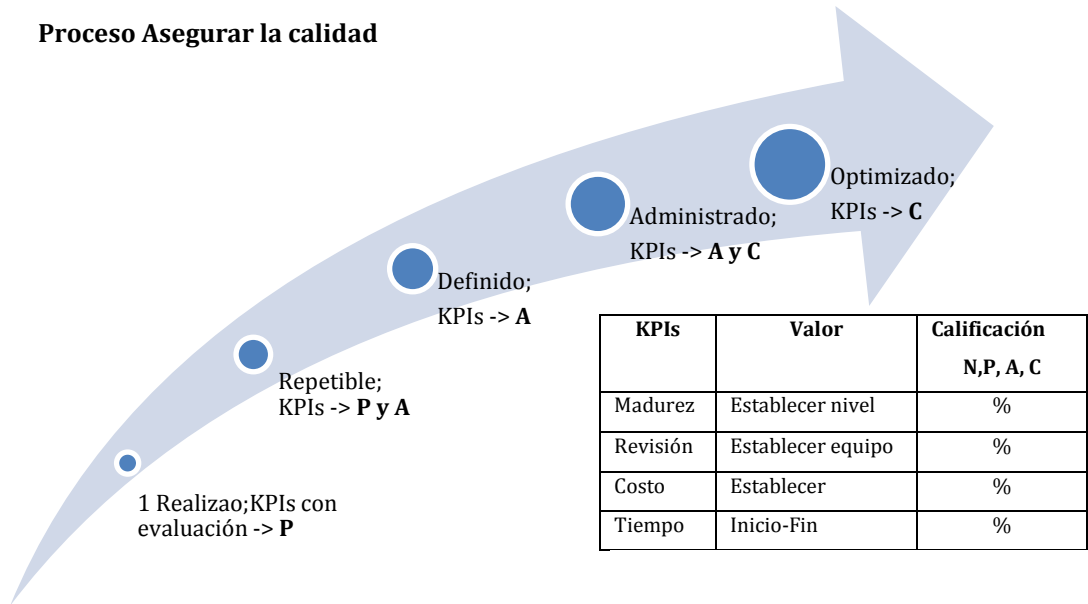


Figura 5.20 Establecimiento del nivel de madurez del proceso Asegurar la calidad y sus KPIs.
Fuente: elaboración propia.

Proceso Cerrar proyecto

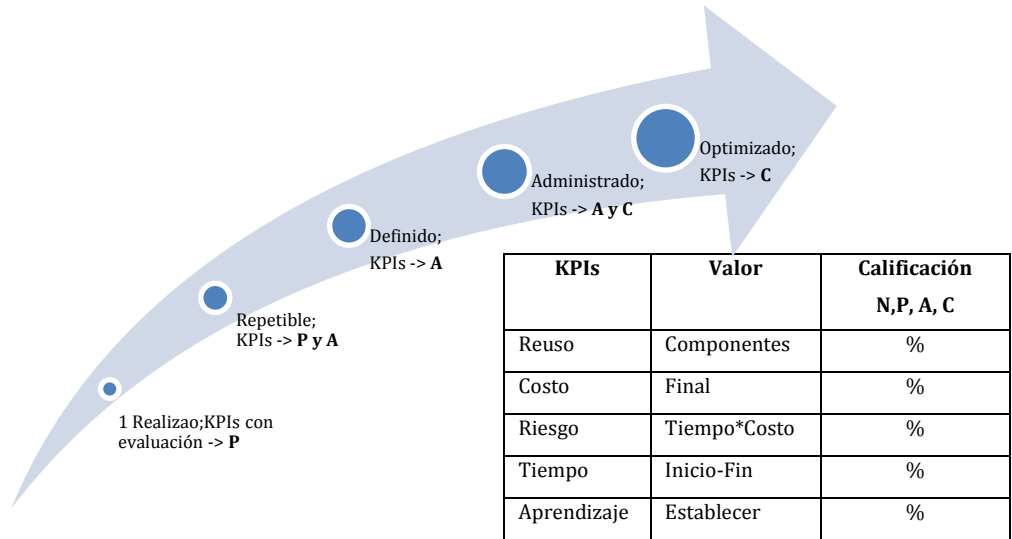


Figura 5.21 Establecimiento del nivel de madurez del proceso Cerrar proyecto y sus KPIs.
Fuente: elaboración propia.

5. 9 Calidad en los equipos y roles.

Para lograr la integración de equipos de trabajo y la asignación de roles de trabajo de acuerdo a las capacidades de los miembros del equipo se pueden seguir las directivas del PSP y TSP definidos por (SEI, 2013). ¿Qué es el TSP? Es una metodología para dirigir el trabajo de mejora y desarrollo de software además de establecer un entorno donde el trabajo efectivo de equipo sea normal y natural. En la figura 5.22 se muestra este proceso.

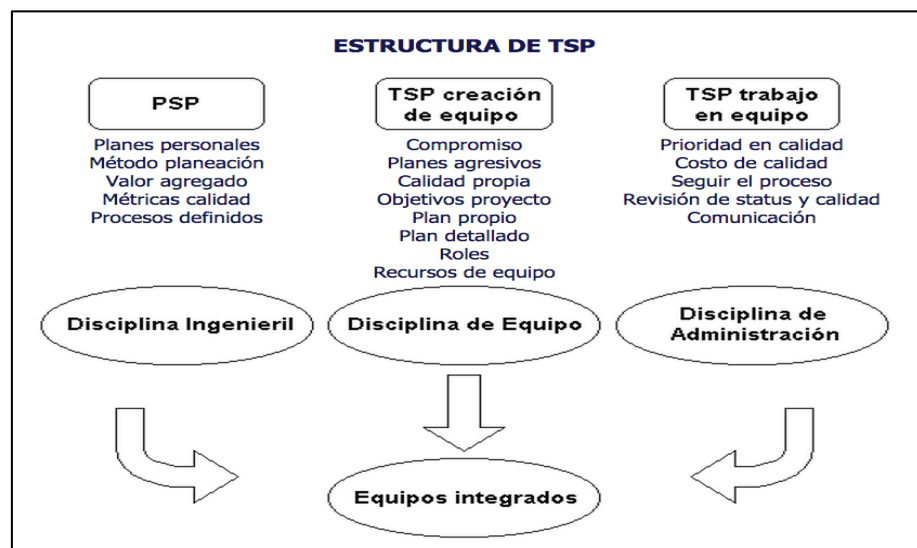


Figura 5.22 Establecimiento de equipos y roles de trabajo propuesto por TSP.
Fuente: (SEI, 2013)

Para lograr una calidad en el establecimiento de los equipos y roles se deben tomar en cuenta los siguientes elementos:

- ✓ Establecer el producto esperado del equipo de trabajo,
- ✓ Realizar el trabajo en forma planeada,
- ✓ Comunicación libre y frecuente,
- ✓ Se establece en forma clara los roles de cada uno de los miembros y
- ✓ Se establece un Plan para el manejo de la calidad y se documentan, defectos, compilación, entregables, artefactos que se deben desarrollar.

Capítulo 6. Pruebas de la metodología MDSIC-M.

6.1 Requerimientos para la herramienta MDSIC 2.0.

Parte esencial de este proyecto es el desarrollo de un sistema que ayude a la implantación de la metodología en FS que son PyMEs. Como se revisó en el capítulo anterior MDSIC-M establece una serie de formas para medir la calidad de cada uno de sus 6 procesos básicos; cada proceso se descompone en subprocesos y estos tienen KPIs que miden la calidad de un desarrollo. Como se describe en el marco teórico, el desarrollo de una APP debe estar apegado a las necesidades de los clientes o los promotores, es necesario entonces utilizar una metodología ágil que permita apegarse a las necesidades del cliente y que permita la generación rápida de aplicaciones que reflejen los requerimientos y que permita en corto plazo actualizaciones y mejoras; con esto se trata de lograr la competitividad en un mercado global, que es cada vez más competido.

Los principios que se deben tomar en cuenta son: prioridad en la satisfacción del cliente, considerar que los requerimientos son cambiantes, entregables en un corto tiempo, retroalimentación diaria entre los clientes y los desarrolladores, equipos motivados de desarrollo, enfatizar la conversación cara a cara para tomar decisiones, el software funcionando es la medida principal de progreso, basarse en procesos para un desarrollo autosustentable, atender la calidad y el buen diseño, la simplicidad como regla de desarrollo, equipos auto organizados y finalmente en intervalos regulares el equipo debe reflejar las mejores estrategias y realizar los ajustes necesarios para un mejor comportamiento.

El mercado de las APPs está totalmente controlado por las grandes corporaciones que construyen dispositivos móviles, sin embargo es posible participar en este mercado comprando licencias que permiten la distribución de APPs en una tienda global, que a través de compras

remunera ganancias para quién la desarrolló (con un porcentaje establecido). Este esquema el que más utilizan las PyMEs en México.

Como se observó, de la encuesta aplicada a empresas PyMEs de la región centro occidente surgieron varios indicadores importantes para los desarrolladores y son los siguientes:

1. Establecimiento de requerimientos.
2. Proyecto inicial.
3. Establecer procesos y tareas.
4. Control de tareas.
5. Definición de entregables.
6. Control de entregables.
7. Control de riesgos.
8. Control de costos y métricas.
9. Garantizar la calidad.
10. Herramienta colaborativa.
11. Plan de pruebas y entregas.
12. Lecciones aprendidas.

Estos requerimientos de las FS que desarrollan APPs son parte de las buenas prácticas descritas en la metodología PMI-ACP para la administración de proyectos mediante el enfoque ágil y adelgazado, en la figura 6.1 se muestran las buenas prácticas propuesta por el PMI-ACP. Se describen la técnicas y herramientas, los conocimientos y habilidades para las 6 actividades de la administración: 1) Entregas basadas en el valor, 2) participación de los interesados, 3) impulsar las prácticas de rendimiento del equipo, 4) planeación adaptativa, 5) detección y resolución de problemas y 6) mejora continua.

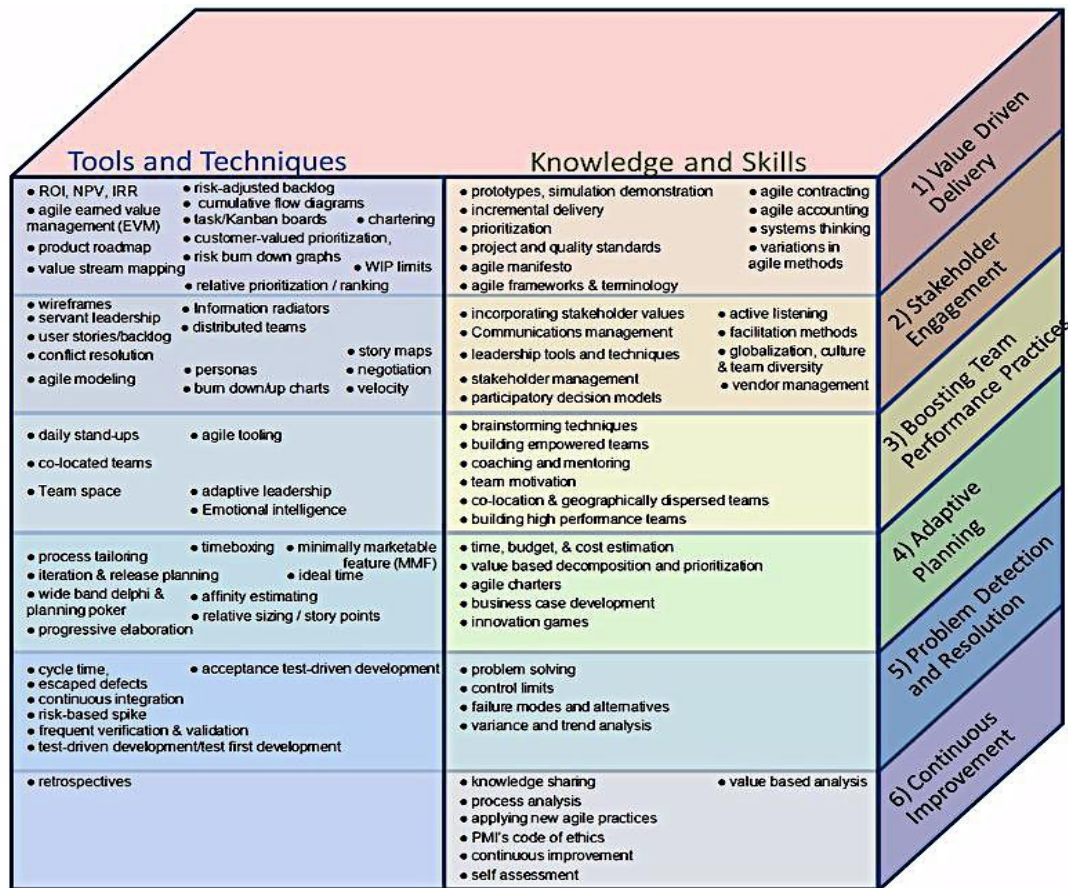


Figura 6.1 Técnicas, herramientas y conocimientos, habilidades para la administración de proyectos ágiles.
Fuente: (PMI-ACP, 2015).

Utilizando los resultados de las encuestas a los desarrolladores de APPs, se resalta la importancia de administrar un desarrollo de acuerdo a las buenas prácticas siguientes:

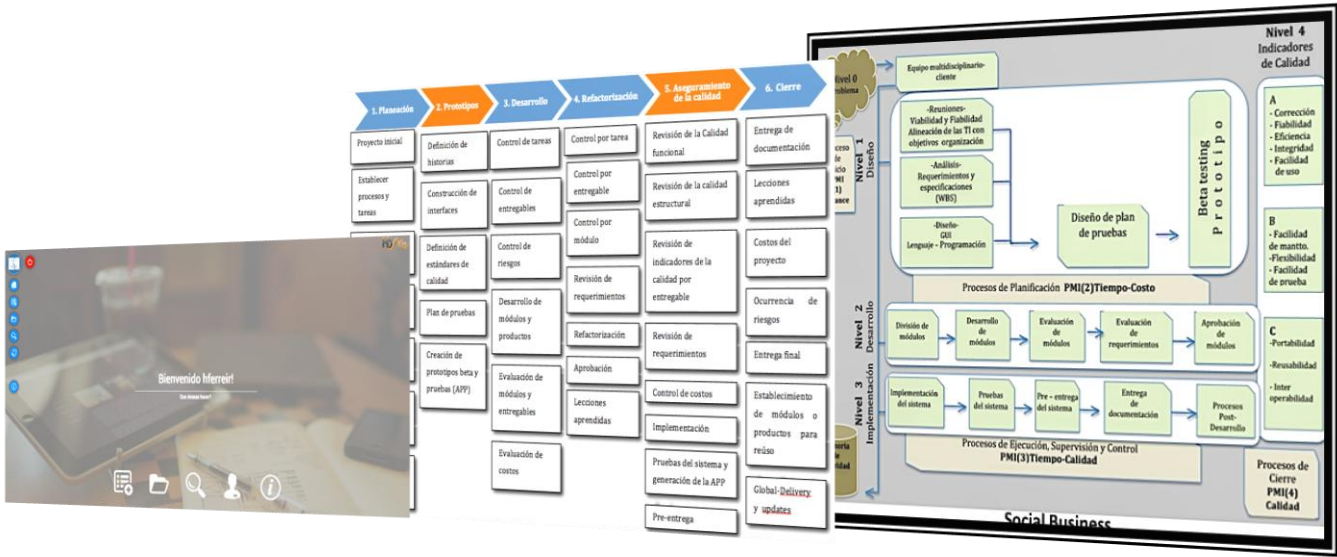
1. Tomar en cuenta los indicadores FURPS y el reuso de componentes.
2. Apegarse a los tiempos y costos proyectados del proyecto (calidad).
3. Revisar continuamente la compatibilidad de la APP y utilizar procesos para organizar el desarrollo (competitividad).
4. Usar de una herramienta para desarrollo ágil, asignar recursos para la calidad, la utilización de una metodología y la utilización de una herramienta IDE.
5. Utilizar un método, un estándar de calidad y herramientas de vanguardia para el desarrollo (ventaja competitiva).
6. Utilizar de tiendas para la distribución de APPs y el uso de métricas en las diferentes etapas del desarrollo.

7. Emplear el *outsourcing* como una forma de competir.
8. Controlar riesgos y usar métricas en la etapa de planeación (certidumbre en el proyecto).

En el diseño de la herramienta también se tomó en cuenta la definición de procesos y las formas de interactuar de los diferentes roles y equipos de trabajo que participan en un desarrollo. De estas consideraciones surgen las necesidades siguientes:

- Se tiene que alimentar al planificador con tareas; asignación, ejecución y liberación de recursos.
- Apegarse a entregables, tiempos y tareas definidos en el plan (WBS).
- Desarrollo apegado a equipos de trabajo colaborativo que promuevan la comunicación a través de redes sociales.
- Utilizar notificaciones del avance en todo momento.
- Facilidad de utilizar varios proyectos a la vez.
- Promover un ambiente colaborativo.
- Facilidad de censar la calidad en las diferentes etapas del desarrollo.
- Usar el control de versiones, prototipos y actualizaciones.
- Almacenar lecciones aprendidas.
- Guardar la información del proyecto y componentes de reuso.
- Facilitar la utilización de la metodología mediante; una interfaz amigable, buen desempeño, confiable y con soporte.

El diseño de la herramienta se muestra en la figura 6.2 A), B) y C).



A) Herramienta MDSIC 2.0 B) Metodología MDSIC-M C) Modelo MDSIC

Figura 6.2 A), B) y C). Diseño de la herramienta MDSIC 2.0 que utiliza la metodología MDSIC-M.
Fuente: elaboración propia.

6.1.2 Requerimientos funcionales y no funcionales para el diseño de MDSIC 2.0.

En la tabla 6.1 se describen los diferentes tipos de requerimientos para el software MDISC 2.0 que utilizará la metodología MDSIC-M.

Tabla 6.1 Diferentes tipos de requerimientos para el software MDSIC 2.0.
Fuente: elaboración propia.

Funcionales	No funcionales	Encuesta a PyMEs	Expertos en el desarrollo
Establecer características. Establecer Proyecto inicial. Establecer procesos y tareas. Creación y control de tareas. Creación y control de entregables. Control de riesgos. Control de costos y métricas. Ayudar a garantizar la calidad. Herramienta colaborativa.	Que ayude a utilizar MDSIC-M. Fácil de usar. Confiable. Buen desempeño. Con soporte. Bajo costo. Estable. Multiproyecto.	Permita censar indicadores. Reuso de componentes. Apegarse a los tiempos planeados. Apegarse a los costos establecidos. Compatibilidad de la APP. Utilización de procesos.	Ayude a la calidad. Fácil de utilizar. Colaborativa. Social Business. Responsiva. Ubicua. Multiplataforma. Uso de estándares. Ayude a la madurez. Ayude al crear equipos.

Ofrezca un plan de pruebas. Almacene lecciones aprendidas.		Herramienta desarrollo ágil. Control de la calidad. Utilización de una metodología. Uso de una herramienta IDE. Uso de un método. Uso estándares de calidad. Uso de tiendas. Distribución de APPs. Métricas en todas las fases. Que fomente el outsourcing. Control de riesgos.	Asignar roles. Control de riesgos. Control de costos. Control del tiempo. Jerarquía de tareas. Tiempo de Sprint. Tiempo de entrega.
---	--	---	---

6.2 Establecimiento del Plan para el desarrollo de MDSIC 2.0.

Para dar seguimiento a la metodología MDSIC-M, se requiere construir una herramienta que ayude a controlar el desarrollo de una APP con calidad. A continuación se describen las actividades para este propósito, se define el plan del proyecto, los entregables, las tareas y los costos de desarrollo de la herramienta. En la figura 6.3 se muestra el WBS que describe las principales tareas para el desarrollo del software MDSIC 2.0.

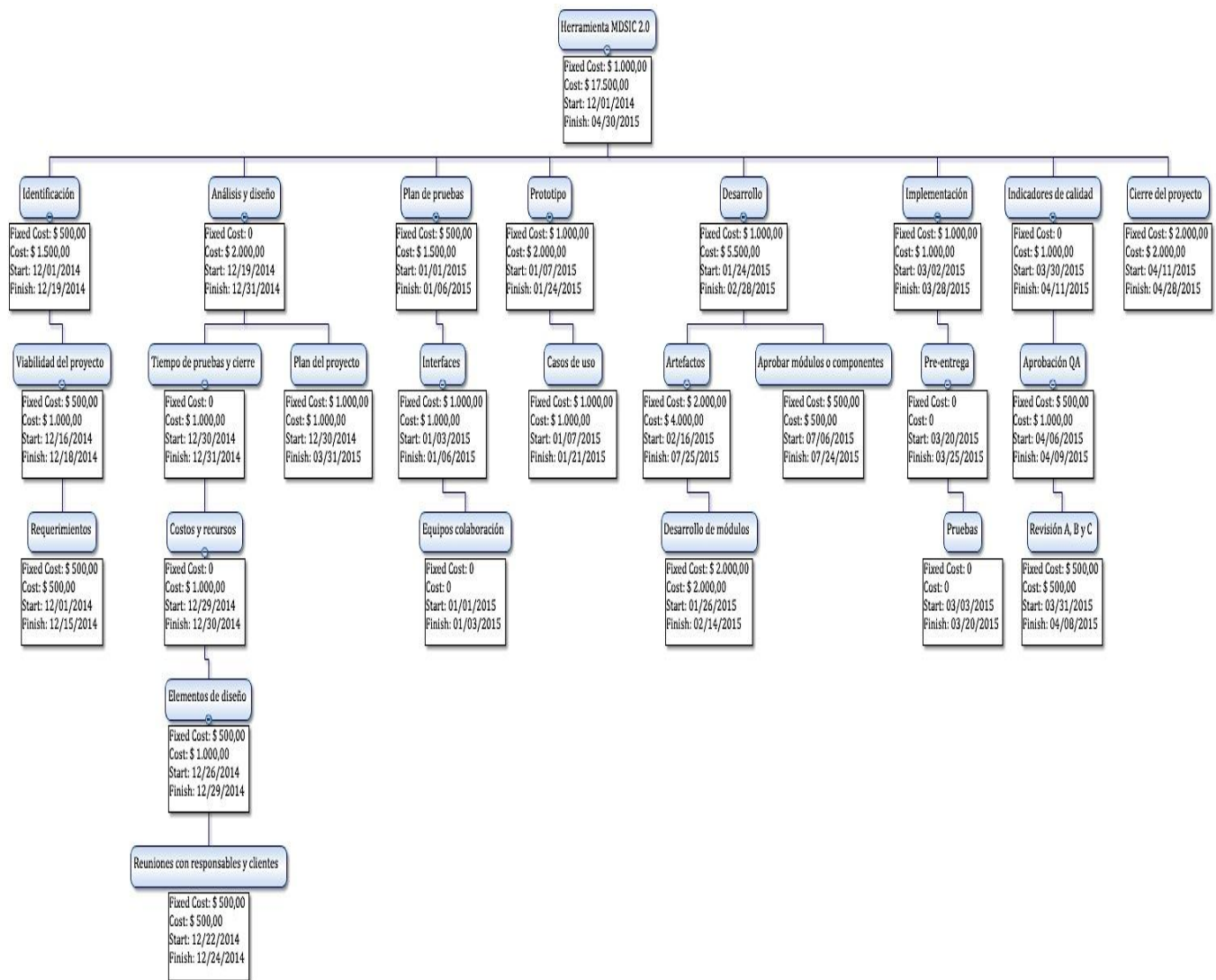


Figura 6.3 WBS construido para el desarrollo de la herramienta MDSIC 2.0.

Fuente: elaboración propia.

Los principales entregables del plan son los siguientes (basado en MDSIC):

- ✓ Identificación de las necesidades del proyecto; requerimientos y viabilidad del proyecto.
- ✓ Análisis y diseño de la herramienta; reuniones con clientes y promotores, elementos de diseño, costos y recursos necesarios para el desarrollo, tiempos para pruebas y cierre y generación del plan.
- ✓ Plan de pruebas; establecer la interacción por roles (colaboración) y diseño de interfaces.

- ✓ Prototipo; Establecer casos de uso en las interfaces.
- ✓ Desarrollo de la herramienta; construcción de módulos, construcción de artefactos y la evaluación de la calidad de componentes por el equipo de QA.
- ✓ Implementación de la herramienta; aplicar pruebas de funcionamiento, programar una pre-entrega y liberar la herramienta funcionando.
- ✓ Censar indicadores de calidad; revisión de los indicadores A, B y C y aprobación de módulos, componentes y artefactos de la herramienta.
- ✓ Finalmente programar el cierre y entrega de la herramienta.

6.3 Diseño del plan de pruebas.

Se establecieron las fechas y los componentes que se deben probar en el diseño de la herramienta, se especificaron las interfaces y los casos de uso, esta actividad es previa al diseño de interfaces y debe describir con claridad los elementos de calidad que se probarán. En la tabla 6.2 se muestra el formato de las pruebas realizadas.

Tabla 6.2. Formato de pruebas para el desarrollo de MDSIC 2.0.
Fuente: elaboración propia

Formato de caso de pruebas
Tipo de prueba: De caja blanca o caja negra del componente Establecer Tareas y Roles Objetivo: Validar fallas en el establecimiento de las tareas del WBS y la asignación de roles de acuerdo al plan del proyecto.
Caso No. 1
Descripción; Permite la definición y establecimiento de tareas que se definen para el WBS.
Entradas; Tareas del WBS; tiempo, recursos, roles, costos y riesgos de ejecución.
Salida esperada; Listado de tareas, para iniciar ejecución en el scheduler.

En la tabla 6.3 se hace una síntesis del plan de pruebas que se aplicó a las interfaces de la herramienta.

Tabla 6.3 Pruebas que se aplicarán a las interfaces de la herramienta MDSIC 2.0.
Fuente: elaboración propia.

Tipo de prueba	Caso de prueba	Resultado	Criterio de QA
Acceso al sistema.	1. Funcionamiento de la Interfaz de acceso.	Permitió el acceso a los usuarios que crearon cuenta.	FURPS.
Almacenamiento.	2. Base de datos, usuarios, proyectos, roles, métricas, tareas, lecciones, componentes.	Se almacenó en forma correcta los elementos.	Tablas con información
Uso de interfaz.	3. Iconos de proyectos, tareas, reportes, equipos de trabajo.	Todos los elementos respondieron al clic.	Respuesta eficiente.
Carga de proyecto.	4. Cargar un proyecto creado y revisar el avance.	Se observó la carga en forma correcta; tareas, entregables, fechas y costos.	Datos del proyecto correctos.
Acceso al proyecto con roles diferentes.	5. Acceder con roles diferentes y cargar el mismo proyecto.	Se observó el avance del proyecto y notificaciones de acuerdo al rol.	Avance del proyecto y notificaciones.
Creación de tareas, tiempos y recursos.	6. Crear un conjunto de tareas para un entregable y verificar, tiempos, recursos, costos y métricas para los riesgos.	Se crearon tareas, se asignaron recursos, tiempos y riesgos. Se revisó la coherencia con los entregables y requerimientos.	Establecimiento de métricas y cálculos de costos correcto. Validación del planificador.
Notificaciones y colaboración.	7. Dar seguimiento a la ejecución de las tareas en el planificador, revisar notificaciones y coherencia en la colaboración.	Se evaluó el funcionamiento del planificador de acuerdo a las tareas, costos y riesgos definidos.	Ejecución confiable de las tareas, tiempos y costos correctos.
Firmas de los participantes.	8. Verificar la participación de un actor de acuerdo a su rol y validación en el proyecto.	Se verificó la correcta integración de equipos de colaboración.	Interacción entre colaboradores y notificaciones ad-hoc al avance.
Reportes de avances.	9. Revisar la generación correcta de los reportes de acuerdo a la fase de la metodología.	Se verificó la correcta generación de los reportes.	Elementos de reportes e información correcta.
Evaluación de	10. Revisar la evaluación de los módulos	Se verificó que se permitirá	Nivel de madurez

Calidad.	de calidad que permite validar la madurez por proceso.	la evaluación de procesos por nivel.	ad-hoc al proceso.
Social Business.	11. Revisar la generación de la información en redes sociales de acuerdo al avance del proyecto.	Se verificó que la información arrojada en redes sociales fuera correspondió al proyecto.	Información de redes sociales ad-hoc al proyecto.
Cierre del proyecto.	12. Revisar el cierre correcto de un proyecto.	Se revisó que se permitirá cerrar el proyecto de acuerdo a los criterios definidos en el plan.	Cierre y cálculo de costos y tiempos correctos.
Componentes de reuso.	13. Revisar que se permite la documentación de componentes a reusar en otros proyectos.	Se verificó que se permitiera documentar componentes de reuso.	Definición de componentes de reuso.
Lecciones aprendidas.	14. Verificar que la información del proyecto se almacene.	Se revisó que la información almacenada fuera coherente con el proyecto.	Informe de lecciones aprendidas.

En la sección siguiente se establece el diseño de las interfaces de acuerdo a los requerimientos de la herramienta MDSIC 2.0.

6.4 Diseño del prototipo

Como método de validación, MDSIC propone la construcción de un prototipo que refleje las necesidades de los clientes de acuerdo a las reuniones con el equipo de desarrollo, un prototipo es un modelo que muestra el comportamiento del sistema que se está diseñando, se utiliza para entenderlo completamente o ciertos aspectos de él y así clarificar los requerimientos del cliente. Un prototipo es la representación de sistema, aunque no sea el sistema completo, debe reflejar principalmente los casos de uso definidos en la etapa de diseño de la herramienta.

6.4.1 Casos de uso.

Es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo un proceso. Los personajes o entidades que participarán en un caso de uso se denominan actores. Es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios u otros sistemas. Una relación es una conexión entre los elementos del modelo, por ejemplo la especialización y la generalización son relaciones. Estos pueden ser programados con orientación a objetos (POO).

Los casos de uso se utilizan también para ilustrar los requerimientos del sistema al mostrar cómo se relacionan eventos y componentes que se producen para cada uno de los actores. Los más comunes son los que representan a los requisitos funcionales del sistema.

En la figura 6.4 se muestran los principales casos de uso de la herramienta MDSIC 2.0. Los actores principales son:

- Administrador. Encargado de promover, administrar y costear el proyecto.
- Cliente. Es el usuario final o consumidor de la aplicación a desarrollar.
- Programador o diseñador. Desarrolla o diseña los componentes, artefactos del prototipo y de la aplicación.
- Calidad. Verifica la correcta construcción del prototipo, componentes y artefactos, mediante la validación en cada una de las fases de la construcción de la aplicación.

De estos casos de uso se desarrollaron los objetos necesarios (DOO) para construir la herramienta MDSIC 2.0.

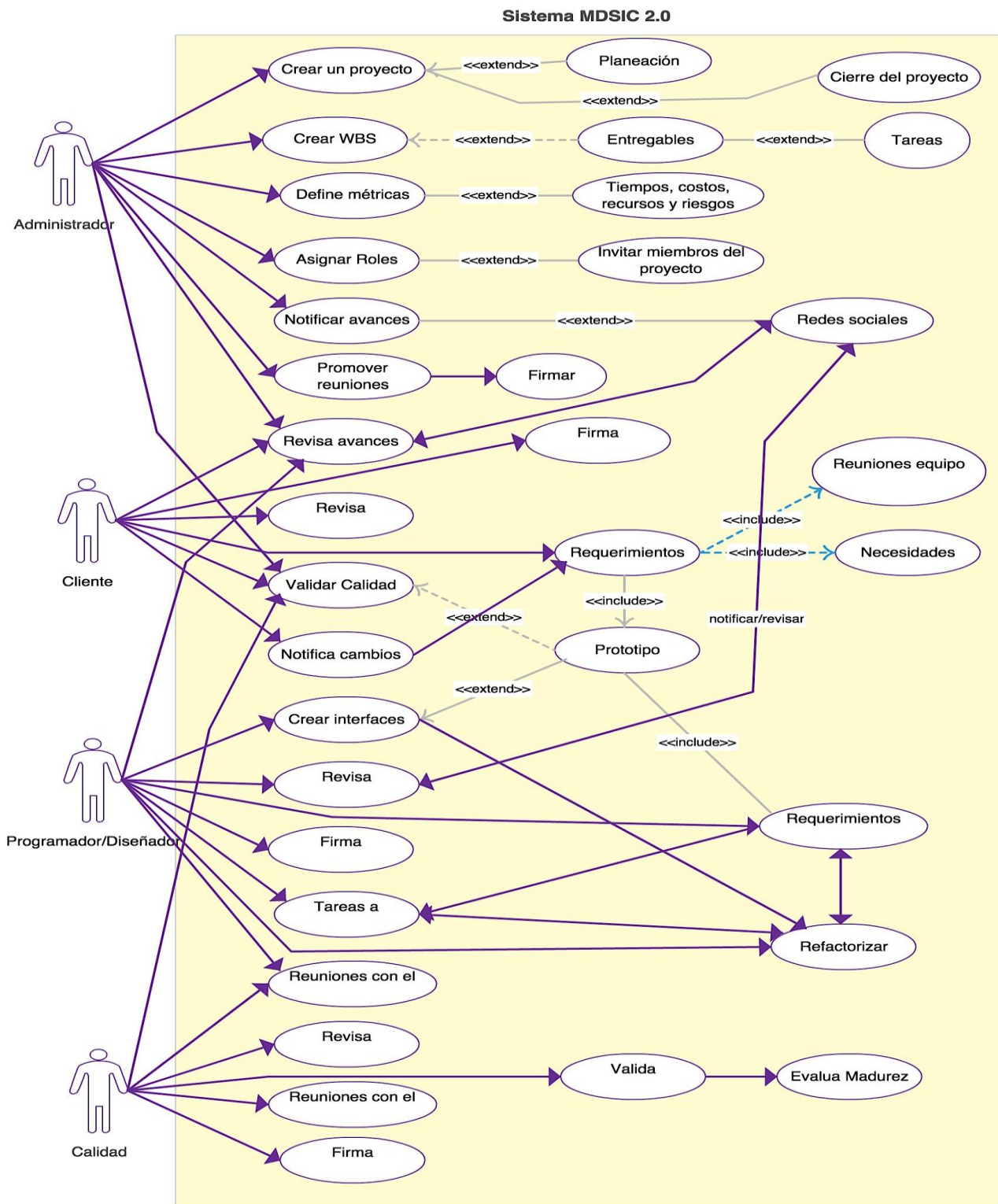


Figura 6.4 Casos de uso para el desarrollo con POO de MDSIC 2.0.
Fuente: elaboración propia.

6.4.2 Diseño de la base de datos (BD).

Considerando los casos de uso y los requerimientos funcionales se desarrollo una BD que permitiera almacenar la información referente a las diferentes fases del desarrollo de un APP siguiendo la metodología MDSIC-M, los elementos de almacenamiento son:

- Datos generales del proyecto.
- Especificación de requerimientos.
- Plan de pruebas y prototipo.
- Tareas y entregables del proyecto.
- Métricas de costos y riesgos.
- División de módulos y desarrollo.
- Componentes refactorizados y errores.
- Implementación, plan de pruebas y pre-entrega de la APP.
- Lecciones aprendidas.
- Evaluación de calidad.
- Evaluación de madurez.
- Gráficas de avance por fase.

A continuación se muestra el diagrama de la bases de datos en formato entidad relación que especifica lo anterior descrito. En la figura 6.5 se muestra el diseño de la base de datos.

Fuente: elaboración propia.

6.5 Desarrollo de la herramienta MDSIC 2.0.

Como requerimiento principal del software es; “construir una herramienta que permita la colaboración y sea de fácil uso para los usuarios de la metodología, desde cualquier dispositivo”. Se decidió desarrollar una *WebAPP* (Aplicación que puede ejecutarse en cualquier explorador de Internet y que se adapta a diferentes resoluciones para distintos dispositivos). Se utilizaron las siguientes herramientas y *framework* (marco de programación) para el desarrollo de MDSIC 2.0 (basado en un desarrollo tipo Modelo, Vista y Controlador, MVC), ver tabla 6.4.

Tabla 6.4 Herramientas y Framework utilizados para el desarrollo de MDSIC 2.0.
Fuente: elaboración propia

Herramienta	Descripción
HighCharts	Es un <i>framework</i> para el diseño y construcción de gráficas automatizadas, utiliza JavaScript y JQuery en un diseño MVC (mysql, JQuery, JavaScript). En el servidor se reciben llamadas de Ajax (OpenAjax, 2012) para generar la vista de la gráfica. (HighCharts, 2014).
JSF <i>JavaServer Faces</i>	Es una tecnología y <i>framework</i> para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java. JSF usa <i>JavaServer Pages</i> (JSP) como la tecnología que permite hacer el despliegue de las páginas (Vista), pero también se puede acomodar a otras tecnologías como XUL (<i>XML based Use interface Language</i>). (ORACLE, 2010).
jQuery	Es una biblioteca de JavaScript, que simplifica la forma de interactuar con los documentos HTML, manipular el árbol DOM (<i>Document Object Model</i>), manejar eventos, desarrollar animaciones y agregar interacción con Ajax en páginas web. (jQuery Foundation, 2006).
Netbeans	Es un entorno de desarrollo integrado (IDE), que utiliza el lenguaje de programación Java. Tiene varios módulos para extenderlo y mejorar el desarrollo de un API (java EE, web, desktop, Mobile, etc.). (ORACLE, 2015).
PHPMyAdmin/MySQL	Es una herramienta (desarrollada en PHP) que apoya a la administración de la BD elaboradas en MySQL utilizando páginas web. Ayuda a usar una BD y también a; crear, eliminar y alterar tablas, a borrar, editar y añadir campos, a ejecutar sentencias SQL, administrar campos, administrar privilegios, exportar datos entre otras características. Se encuentra disponible bajo la licencia GPL Versión 2. (phpMyAdmin, 2015).

PrimeFaces	Es una biblioteca de componentes para JSF de código abierto, que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web. Tiene licencia de Apache V2. Permite la integración con otros componentes que mejoran las GUI en web. (PrimeFaces, 2014)
Tomcat/Linux	Apache Tomcat es un contenedor de <i>servlets</i> (aplicación java que se ejecuta en un servidor Linux u otro) desarrollado bajo el proyecto <i>Jakarta</i> de <i>Apache Software Foundation</i> . Implementa las especificaciones de los <i>servlets</i> y de JSP de Oracle y ayuda al despliegue de páginas web desarrolladas en java. (Apache Tomcat, 2015).

Todas las herramientas anteriores fueron utilizadas por el equipo de desarrollo (2 programadores) que siguieron la especificación de los casos de uso definidos anteriormente.

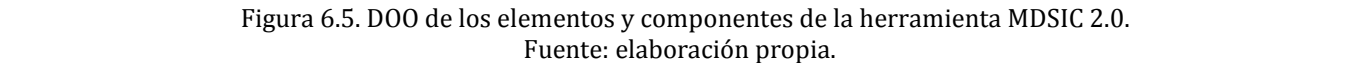
6.5.1 Herramientas y Diseño Orientado a Objetos (DOO).

El DOO se utilizó como base para el desarrollo del software, funciona como una guía para los programadores y para planificar el código. El DOO de un objeto agrupa datos encapsulados y procedimientos para representar las distintas entidades. Los objetos que fueron identificados y documentados durante el análisis orientado a objetos (AOO) se describen a continuación:

- NavManager. Clase de objeto que define la interfaz principal de la herramienta; acceso, manejo de ventana, menú de proyectos, menú de metodología y lienzo principal.
- Register. Clase de Objeto que define las propiedades de los usuarios que acceden al sistema.
- DbConnector. Clase de objeto que define la conexión con la base de datos de MySQL, la que a su vez almacena los datos del proyecto.
- Proyectos. Clase de objeto que define las propiedades de los proyectos y que se especializa en fechas y datos de un proyecto.
- Requerimiento. Clase de objeto que representa los entregables del proyecto y se especializa en Tareas, Requerimientos (entregables) y la conexión con el scheduler.
- Scheduler. Clase de objeto que define las características generales de un proyecto y controla el tiempo de ejecución de las tareas, además de lanzar alertas y notificaciones.

- Form. Clase de objeto que integra todos los formularios (pantallas) de los proyectos en desarrollo y permite su despliegue control y ejecución. Se especializa en los formularios FormDA, FormEB, FormDB, FormCB, FormEA, FormBA, FromCA y FormAA (para despliegue de diferentes pantallas de las fases de la metodología).
- Miembro. Clase de objeto que define los atributos de un usuario del proyecto, se especializa en la clase Miembros (conjunto de usuarios) y en la clase Roles (que define los atributos del rol de cada uno de los miembros)

En figura 6.5 se muestra la interacción entre los objetos del DOO.



Metodología MDSIC-M para desarrollo de aplicaciones móviles en PyMEs	209
--	-----

todas las tareas definidas en la planeación del proyecto. Debe identificarse para cada proyecto la fecha de inicio y fin (real y planeada), los entregables que son parte de los requerimientos, el costo proyectado, el costo actual (con riesgos incluidos), costo excedido en requerimientos, tiempo del proyecto excedido, además de controlar las alertas que indican costos, tiempos y fechas excedidos. En la figura 6.6 se muestran los atributos y métodos de cada clase.

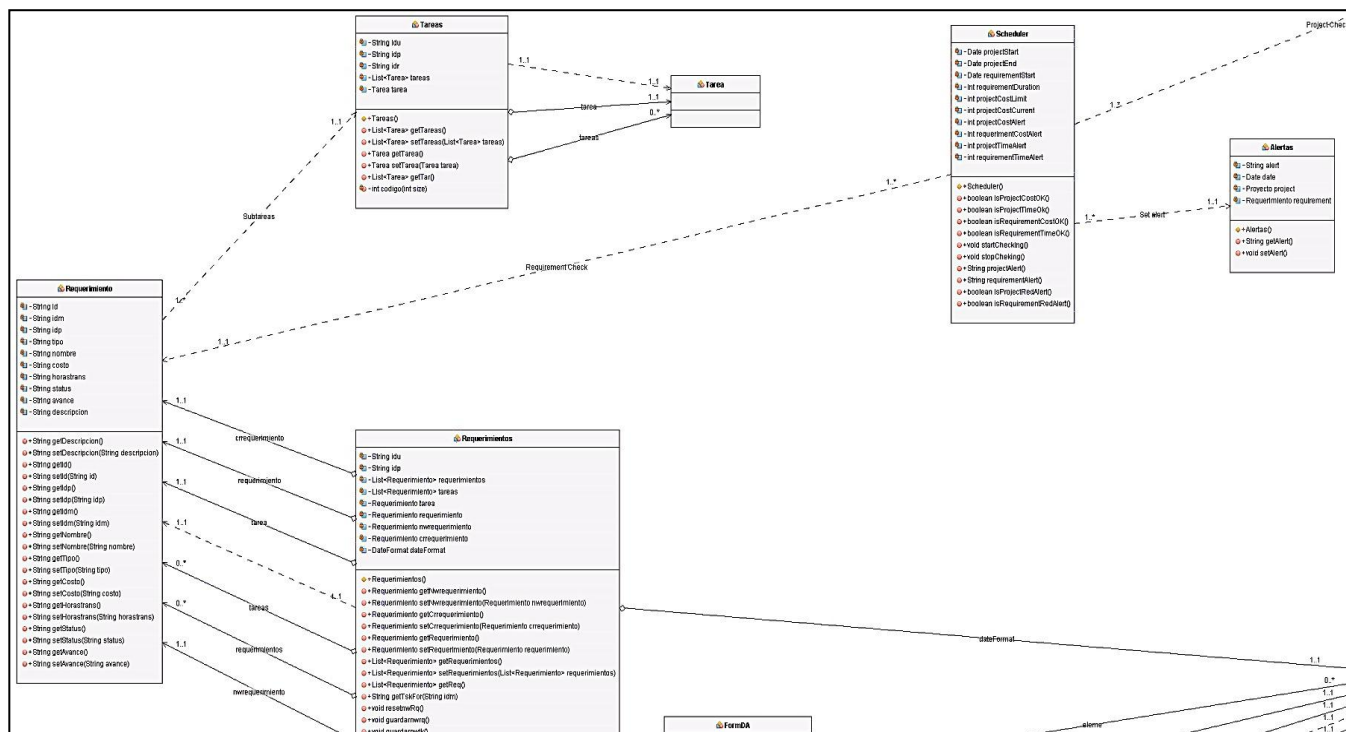


Figura 6.6. Atributos y métodos de las clases scheduler, Requerimiento, Tareas, Tarea y requerimientos.
Fuente: elaboración propia.

De este diseño se procedió a la programación de cada una las interfaces y componentes, se realizó el desarrollo de acuerdo a las fechas planteadas en el WBS para el desarrollo de MDSIC 2.0.

6.5.3 Componentes de desarrollo.

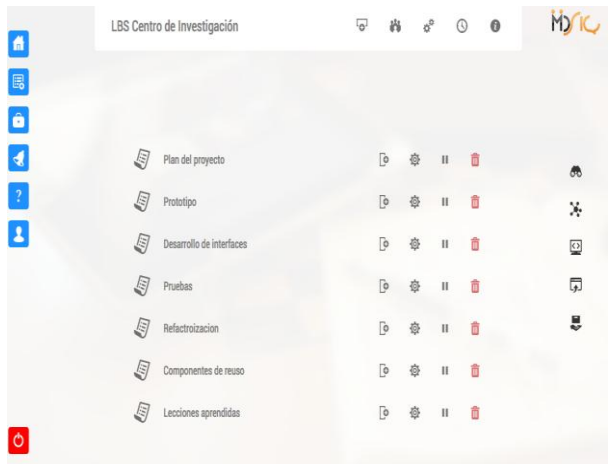
Del prototipo, del DOO y el diagrama para la POO, se obtienen los componentes a desarrollar para la herramienta MDSIC 2.0. En la figura 6.7 A), B), C) y D) se muestran las interfaces y componentes del prototipo.



A) Acceso al sistema MDSIC 2.0.



B) Pantalla de inicio.



C) Edición de tareas y entregables en WBS.



D) Edición de un formulario para la fase de identificación.

Figura 6.7. Interfaces del prototipo de MDSIC 2.0 para su desarrollo.
Fuente: elaboración propia.

En la tabla 6.5 se muestran los principales componentes desarrollados para las interfaces diseñadas en el prototipo.

Tabla 6.5. Componentes desarrollados para creación y modificación de proyectos en MDSIC 2.0.
Fuente: elaboración propia

Componente	Descripción	Icono
Inicio	Muestra la pantalla de inicio.	
Nuevo proyecto	Crear un nuevo proyecto; nombre, cliente, fecha inicio y termino, presupuesto, descripción y horario de trabajo.	

Proyectos	Muestra un listado de los proyectos a los que pertenece el usuario.	
Notificaciones	Muestra alertas y notificaciones pendientes para el usuario, incluye la interacción con redes sociales.	
Información ayuda	Ayuda e información del uso de la herramienta.	
Mi cuenta	Muestra los datos del usuario y rol en el proyecto.	
Almacenar	Guardar los cambios realizados.	
Regresar	Regresar al menú anterior.	
Mostrar	Visualizar el proyecto para su seguimiento y firma en caso de ser necesario.	
Editar	Modificar los datos del proyecto.	

En tabla 6.6 se muestran todos los componentes desarrollados (agrupados por menú), para el seguimiento de un proyecto, sus tareas y fases en la metodología MDSIC-M.

Tabla 6.6. Componentes agrupados para creación y modificación de tareas y fases en MDSIC 2.0.
Fuente: elaboración propia

Componente	Descripción	Icono
Administración	1. Agregar tareas al WBS, 2. Administrar miembros del proyecto, 3. Configurar proyecto, 4. Historial del proyecto y 5. Información del proyecto	 1  2  3  4  5
Definición y edición de tareas	1. Agregar tarea o subtarea, 2. Configurar entregable, 3. Pausar tarea y 4. Eliminar tarea	 1  2  3  4
Fases de la metodología	i. Fase de identificación del proyecto. i. Fase de diseño del proyecto i. Fase de desarrollo de componentes v. Implementación del proyecto v. Calidad en el proyecto	 1  2  3  4  5
Interacción entre fases	1. Regresar a fase anterior 2. Agregar nuevo elemento o formulario 3. Firma digital de un formulario	 1  2  3

6.6 Pruebas y Cierre del proyecto.

Como se describe en el plan del proyecto y en el diseño de los casos de uso, se realizaron las pruebas con 2 tipos de proyectos (ficticios) de diferentes tamaños; proyecto pequeño de 3 personas y componentes reducidos, proyecto mediano para 15 personas y con muchas tareas. En el capítulo 7 se concluye sobre el funcionamiento y pertinencia de la metodología.

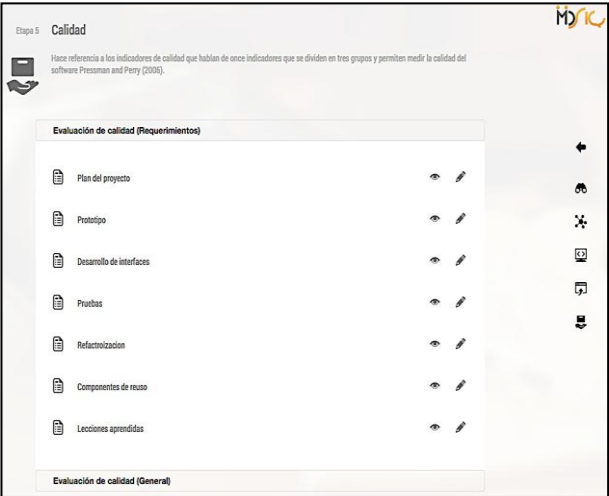
6.6.1 Indicadores de calidad.

El aseguramiento de la calidad se realiza de acuerdo al modelo propuesto por (Nathan-Regis & Nasira, 2013). Una metodología ágil que utilice técnicas de calidad debe considerar lo siguiente:

- 1) Asegurar la calidad en los requerimientos y artefactos, metamorfosis, arquitectura en pico y retroalimentación del cliente en el sitio.
- 2) En el desarrollo y diseño de componentes debe garantizar la calidad mediante la refactorización, la programación por pares, reuniones de avances y simplificar problemas.
- 3) La implementación y aceptación de las pruebas deben hacerse al 100% con retroalimentación de los clientes. Estas etapas deben generar un producto con calidad.

Para controlar en forma adecuada la calidad en el desarrollo de una APP, se propone el desarrollo de productos (componentes) utilizando la técnica de refactorización (reconstrucción de elementos con fallas), además se considera como parte importante la asignación de roles y recursos a quiénes tengan las habilidades adecuadas; esta asignación se realiza por parte del promotor del proyecto y el líder de cada una de la áreas (en la PyMEs es principalmente el administrado). Es importante señalar que la técnicas de QA deben ser revisadas y censadas para garantizar un correcta construcción de los componentes del proyecto, algunas de las prácticas propuestas por Nathan-Regis et. al. (2013), son; 1) Refactorización, 2) Desarrollo basado en pruebas, 3) Integración continua, 4) Programación por pares, 5) Comunicación cara a cara, 6) Participación continua del cliente, 7) Retroalimentación frecuente del cliente, 8) Metáfora del sistema para obtener un abstracción simple. En la figura 6.8 A), B) y C) se muestra las interfaces

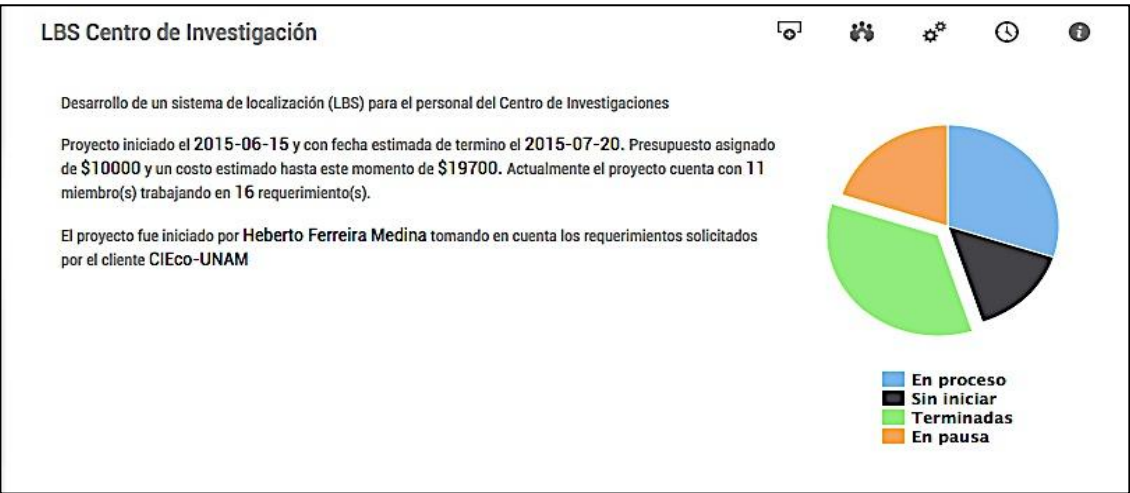
que permiten garantizar la calidad por cada uno de los diferentes elementos de desarrollo de una APP. En el anexo C se puede revisar el manual técnico y de usuario.



A) Evaluación de la calidad por requerimientos



B) Evaluación de la calidad por fase



C) Evaluación de la calidad por proyecto

Figura 6.8 A), B) y C) Interfaces que se diseñaron para validar la calidad por requerimientos, por fase y por proyecto.
Fuente: elaboración propia.

Capítulo 7. Análisis e interpretación de los resultados y trabajos futuros.

7.1 Discusión.

El reúso de componentes se considera un tema importante entre los desarrolladores de APPs, por lo que MDSIC 2.0 fue diseñado para almacenar esta información (se guardan en la tabla de características del proyecto). De igual forma la herramienta también almacena la información de los procesos genéricos definidos en MDSIC-M, para cada fase; en donde se censan los KPIs y se evalúan de acuerdo a la NOM (calidad).

MDSIC-M surge del análisis del marco teórico, de la filosofía del desarrollo ágil, de la filosofía Lean y sobretodo de la experiencia de los desarrolladores de APPs en la región. En la figura 7.1 se muestra el contexto en el que se diseño.

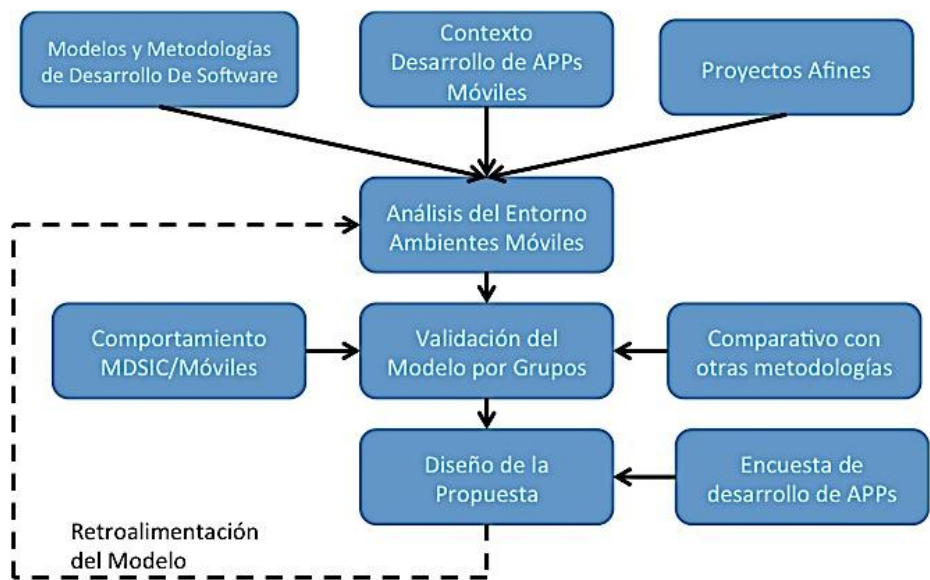


Figura 7.1. Contexto de desarrollo de la metodología MDSIC-M.
Fuente: elaboración propia.

Como se observa en la figura MDSIC-M es el resultado de la investigación, de la experiencia y la aplicación de las buenas prácticas en el desarrollo de APPs. Una parte importante de esta propuesta es la retroalimentación mediante la aplicación de la metodología en desarrollos reales en PyMEs, analizando así su factibilidad de implementarse en FS y su funcionalidad.

En un desarrollo basado MDSIC-M se controlan los riesgos, esto ayuda a que una APP sea codificada con calidad y compatibilidad: ayuda a la entrega planeada de los productos a través de tiendas estándares y además se define un proceso de cierre de un proyecto, que establece la estrategia de mercadotecnia de la APP.

La utilización de métricas ofrece a una ventaja competitiva al permitir establecer los recursos y costos de cada uno de los componentes que se desarrollan. También ayuda a garantizar el presupuesto proyectado y solo los riesgos podrán cambiar el escenario final del cierre del proyecto. Es una propuesta de desarrollo ágil que ayuda a las PyMEs a adoptar un método simple y eficaz de desarrollo de APPs.

7.2 Validación de un cuasi experimento.

Para validar la calidad en un desarrollo se clasificó los KPAs utilizando el modelo de Chappel (2014), que define el QA estructural (QAs), funcional (QAf) y la calidad del proceso (QAp), en la tabla 7.1 se muestra la correspondencia de los KPAs y la calidad que censa. Se verificó cada uno de estos en proyectos reales y se obtuvo el porcentaje de los componentes que si se lograron terminar (utilizando el porcentaje de terminación alcanzado).

Para el QAs se establecieron 6/6 KPAs, para QAf 6/6 y finalmente para QAp 6/6 indicadores. Ver tabla 7.1.

Tabla 7.1. KPAs y su correspondencia con el modelo de calidad de Chappel (2014).
Fuente: elaboración propia.

KPAs	QA Funcional	QA Estructural	QA Proceso
Establecimiento de requerimientos.	✓		
Construcción del Plan.	✓		
Definición de métricas.	✓	✓	
Control de entregables.	✓		✓
Prototipo.	✓	✓	
Control del tiempo.		✓	✓
Revisión continua de costos.		✓	✓
Control de riesgos.		✓	✓
Control de tareas.			✓
Refactorización.	✓		
Reúso de componentes.		✓	✓

Para realizar el cuasi-experimento se llevó a cabo la implementación de la metodología en dos proyectos en diferentes FS de la región y se aplicó el modelo de validación a los procesos de la metodología, el cual se muestra en la tabla 7.2 en donde se observa la historia del proyecto y la validación de la calidad. Se definieron dos proyectos; una para el grupo 1 de empresas pequeñas y otro para el grupo 2 de empresas medianas.

Tabla 7.2. Validación de la calidad en 2 proyectos en PyMEs de la región.
Fuente: elaboración propia.

	Proyecto 1 Grupo de empresa Pequeñas	Proyecto 2 Grupo de empresa medianas
Nombre	APP para localización para el personal (LBS).	Creación de un catálogo de especies y una APP para su visualización.
Objetivo	Sistema de Localización del personal LBS, basado en GPS	WebAPP para generar un APP de catálogos de especies
# Requerimientos	4	12
# Participantes	3	10
Roles	1 cliente, 1 desarrollo y 1 calidad	1 administrador, 3 clientes, 3 desarrollo, 1 diseñador y 2 de calidad.
Tareas	10	24

Entregables	6	8
Prototipos	3	4
Componentes de desarrollo	14	25
Refactorización	6	10
Tiempo	56 días	70 días
Costo*	US\$2240	US\$3600
KPIs censados	11	11
Riesgos	Ocurrieron	Ocurrieron

*Nota: el costo está asociado a las métricas de la región occidente y se basa en salarios mínimos en US\$ dólares americanos (INEGI, 2015).

Calidad (QA)	Proyecto 1	Proyecto 2
QA Funcional	80 %	91 %
QA Estructural	60 %	60 %
QA Proceso	60 %	60 %
Validación	Ocurrieron riesgos, tiempo y entrega tardías	Ocurrieron cambios en los requerimientos, tiempo y entrega tardías

Como se observa en los primeros desarrollos al utilizar la metodología la curva de aprendizaje mostró un nivel de madurez en crecimiento en los procesos censados; en el primer proyecto se obtuvieron porcentajes bajos de calidad (proyecto 1 en empresa pequeña), se ofreció un curso de capacitación básico (dos días).

En el segundo proyecto se inicio con un curso de capacitación más completo, donde se explica cómo funciona la metodología (1 semana). Esta estrategia permitió mejorar en el proceso de planeación, en donde la QA funcional fue del 91%, esto demuestra un aumento en el nivel de madurez en los procesos de planeación, prototipado, desarrollo y refactorización. En donde no se logro mejorar significativamente fueron en los procesos de desarrollo y refactorización, que fueron afectados por el proceso de planeación y en el cálculo continuo de costos por que los requerimientos eran cambiantes, lo que llevo un QA estructural del 60% (el control del tiempo, revisión continua de costos, control de riesgos tuvieron tiempos excedidos), esto llevo a retrasos

en los tiempos de desarrollo y refactorización. Los cambios en los requerimientos impactaron en el tiempo de entrega de la APP.

Finalmente se observó que la curva de aprendizaje al tratar de utilizar la metodología no permitió un QA de proceso con mejores resultados; retrasos en los entregables, problemas de control de tiempos, control de tareas y el establecimiento de componentes de reuso fueron algunos de los problemas.

El modelo de validación de la calidad y el uso de una metodología apegada a las necesidades de la FS ayudaron a madurar en sus procesos de desarrollo de software, los estándares Moprosoft y CMMI pueden ser adoptados rápidamente por estas empresas. Como conclusión se puede observar que MDSIC-M ayuda a la adopción de la NOM.

7.3 Comparativa con otras metodologías.

Uno de los objetivos de esta tesis es construir un metodología que ayude a los desarrolladores ofrecer la construcción de APPs de calidad. Esta debe ser ágil y adelgazada de tal forma que las FS puedan adoptarla en fácil y rápida, convirtiéndose así en una alternativa en esta industria. Para justificar su utilización se realizó una comparativa con otras metodologías encontradas en el estado del arte, en donde algunas son adaptadas como método de construcción de APPs; Scrum, Kanban y XP, son la más utilizadas para este propósito. También existen las metodologías como DIMP, MASAM y Mobile-D que son totalmente orientadas al desarrollo de APPs y cuyas características son más afines para este propósito.

En la tabla 7.3 se comparan las características principales de las metodologías utilizadas para el desarrollo de APPs encontradas en la literatura.

Tabla 7.3. Comparativa entre metodologías para el desarrollo de APPs.
Fuente: elaboración propia.

Características	MDSIC-M	Scrum	Kanban	MASAM	DIMP	SleSS	Mobile-D
Establecimiento de requerimientos	✓	✓	✓	✓	✓	✓	✓
Definición de un plan	✓	✓	✓	✓	✓	✓	✓
Definición de métricas	✓	✓	✓			✓	✓
Control de entregables	✓	✓	✓	✓		✓	✓
Prototipado	✓	✓			✓		✓
Control del tiempo	✓	✓	✓	✓		✓	✓
Revisión continua de costos	✓					✓	✓
Control de riesgos	✓	Opcional	Opcional				✓
Control de tareas	✓	✓	✓	✓		✓	✓
Refactorización	✓	✓	✓	✓	✓	✓	✓
Reuso de componentes	✓	✓	✓	✓	✓	✓	✓
Establecer equipos y roles	✓	✓	✓			✓	✓
Uso del <i>social business</i>	✓						
Apegado a normas y estándares	✓						
Madurez de procesos	✓					✓	
Para PyMEs	✓	✓	✓	✓	✓	✓	✓
Censa indicadores FURPS	✓	✓	✓	✓	✓	✓	✓
Número de etapas o fases	6	5	4	4	4	5	5
Filosofía ágil	✓	✓	✓	✓	✓	✓	✓
Roles	5	6	1	7	5	8	8
Filosofía en su aplicación	Organización integral y colaborativa para PyMEs	Organizar equipos y motivarlos	Orientado a cambios y entrega a tiempo	Adaptarse a la empresa y desarrollo rápido	Plataforma colaborativa basada en la distribución	Equipo colaborativo para remover basura	Organizar el desarrollo por fases y etapas

La comparativa está orientada hacia las características que los desarrolladores de aplicaciones móviles expresaron en las encuestas. Es importante señalar que de todas las metodologías anteriores Scrum y Kanban son las que más se utilizan en la FS de México.

En (Letelier & Penadés, 2006) se comparan algunas metodologías ágiles basándose en la características ad-hoc al desarrollo de una APP, éstas son: sistema cambiante, colaboración entre los miembros del equipo, simplicidad, adaptable, resultados rápidos, uso CMM, excelencia técnica, entre otras. De esta comparativa surge el ranking de agilidad, el cual muestra en orden descendente las mejores metodología con estas características:

- 1) XP y ASD (Adaptable Software Development).
- 2) Scrum.
- 3) Desarrollo Lean.
- 4) Desarrollo dirigido por características (FDD) y desarrollo de sistemas dinámicos (DSDM).

En (Pérez P. & González C., 2012) se hace otra comparativa de las principales metodologías ágiles en donde por su impacto y utilización en empresas de España se obtuvo que las metodologías XP, Scrum, Kanban y una combinación de Scrum y Kanban (ScrumBan) son las más utilizadas. En la tabla 7.4 A), B), C) y D) se muestra la comparativa de estas metodologías con MDSIC-M, dividida por: A) uso, B) capacidad de agilidad, C) aplicabilidad y D) procesos y productos.

Tabla 7.4. A), B), C) y D). Comparativa de MDSIC-M con respecto a las principales metodologías ágiles.
Fuente: basado en (Pérez P. & González C., 2012).

			METODOLOGÍAS ÁGILES				
			ORIENTADA AL DESARROLLO DE SOFTWARE		ORIENTADA A LA GESTIÓN DE PROYECTOS		
			MDSIC-M	XP	SCRUM	KANBAN	SCRUMBAN
U S O	¿Por qué utilizar un método ágil?	Respeto de las fechas de entrega	✓		✓		
		Cumplimiento de los requisitos	✓	✓	✓	✓	✓
		Respeto al nivel de calidad	✓				
		Satisfacción del usuario final	✓		✓		
		Entornos turbulentos	✓	✓	✓	✓	✓
		Favorable al Off shoring	✓		✓		✓
		Aumento de la productividad	✓	✓	✓	✓	✓

A) Comparativa con respecto al uso.

			MDSIC-M	XP	SCRUM	KANBAN	SCRUMBAN
C A P A G C I L D I A D A D D E	¿Cuál es la parte de agilidad incluida en el método?	Iteraciones cortas	✓	✓	✓	✓	✓
			✓	✓	✓	✓	✓
		Colaboración					
		Centrado en las personas	✓	✓	✓	✓	✓
		Refactoring político	✓	✓			
		Prueba político	✓	✓	✓		✓
		Integración de los cambios	✓	✓	✓	✓	✓
		De peso ligero	✓	✓	✓	✓	✓
		Los requisitos funcionales pueden cambiar	✓	✓	✓	✓	✓
		funcionales pueden cambiar	✓			✓	✓
		El plan de trabajo puede cambiar	✓	✓		✓	✓
		Los recursos humanos pueden cambiar	✓	✓		✓	✓
		Cambiar los indicadores		✓			
		Reactividad (al inicio del proyecto, cada etapa, cada iteración)	Etapas	Iteración	Iteración	Iteración	Iteración
		Intercambio de conocimientos (bajo,alto)	BAJO	ALTO	BAJO	BAJO	BAJO

B) Comparativa con respecto a la capacidad de agilidad.

			MDSIC-M	XP	SCRUM	KANBAN	SCRUMBAN
A P L I C A B I L I D A D	¿Cuándo un ambiente es favorable para usar este método?	Tamaño del proyecto (Pequeño, Grande)	Pequeño	Pequeño	Grande/Pequeño	Pequeño	Grande/Pequeño
		La complejidad del proyecto (Baja, Alta)	Baja	Baja	ALTA	Baja	Alta
		Los riesgos del proyecto (Bajo, Alto)	Alto	Bajo	Alto	Bajo	Alto
		El tamaño del equipo (Pequeño, Grande)	Pequeño	Pequeño	Pequeño	Pequeño	Pequeño
		El grado de interacción con el cliente (Baja, Alta)	Alta	Alta	Alta	Bajo	BAJA
		Grado de interacción con los usuarios finales (Baja, Alta)	Alta	Bajo	Alta	Bajo	BAJA
		Grado de interacción entre los miembros del equipo (Baja, Alta)	Alta	Alta	Alta	Baja	Alta
		Grado de integración de la novedad (Baja, Alta)	Baja	Alta	Alta	Baja	Alta
		La organización del equipo (Auto-organización, Organización-jerárquica)	Auto-organización	Auto-organización	Auto-organización	Auto-organización	Auto-organización

C) Comparativa con respecto a su aplicabilidad.

			MDSIC-M	XP	SCRUM	KANBAN	SCRUMBAN
P R O C E S O S Y P R O D U C T O S	¿Cómo están caracterizados los procesos del método?	Gestión de proyectos	✓		✓		✓
		Descripción de procesos	✓	✓			
		Normas y orientaciones concretas sobre las actividades y productos	✓	✓			
		Las actividades cubiertas por el método ágil					
		Puesta en marcha del proyecto	✓				
		Definición de requisitos	✓	✓	✓		✓
		Modelado	✓	✓	✓		
		Código	✓	✓	✓	✓	✓
		Pruebas unitarias	✓	✓	✓	✓	✓
		Pruebas de integración	✓	✓	✓	✓	✓
		Prueba del sistema	✓	✓	✓	✓	✓
		Prueba de aceptación					
		Control de calidad	✓				
		Sistema de uso					
		Productos de las actividades del método ágil					
		Modelos de diseño	✓		✓		✓
		Comentario del código fuente	✓	✓	✓	✓	✓
		Ejecutable	✓	✓	✓	✓	✓
		Pruebas unitarias	✓	✓	✓	✓	✓
		Pruebas de integración	✓	✓	✓	✓	✓
		Pruebas de sistema	✓	✓		✓	✓
		Pruebas de aceptación					
		Informes de calidad	✓				
		Documentación de usuario					

D) Con respecto a los procesos y productos.

Como se observa la metodología MDSIC-M tiene sus fortalezas y debilidades, sin embargo se posiciona muy bien con respecto a otras metodologías que se han utilizado entre los desarrolladores de APPs, el reto es promover la difusión y uso de esta para que se convierta en una alternativa real en la industria del desarrollo de software.

7.4 Comparativas de MDSIC 2.0 con otras herramientas.

Las metodologías comparadas ofrecen diferentes herramientas para el control automatizado del desarrollo ágil, algunas herramientas hacen énfasis en la colaboración, en el seguimiento, control de actividades y sobretodo ayudan al administrador del proyecto para el control de costos.

Existen en la actualidad muchas herramientas que utilizan las metodologías ágiles y *Lean*, cada una ofrece características similares basadas en las cuatro principales metodologías; Scrum, Kanban, XP y Lean entre otras. Para reducir el número de herramientas que se requiere comparar con MDSIC 2.0, se investigó en diferentes listas o blogs en Internet sobre “herramientas para la administración de proyectos basadas en metodologías ágiles”, los rankings utilizados para la selección son de acuerdo a los artículos de (Krill, 2013), (Peter, 2010), (de Haaff, 2012), (Erickson, Dierking , & Thomas, 2014), (Agile-Tools, 2013), (Ronen-Harel, 2015), (Shojaee, 2012) y (GetApp, 2015). De las 26 herramientas mostradas en estas listas se seleccionaron las 6 mejores de acuerdo a su ranking (las que más aparecen en los ranking de mayor puntaje), éstas son las siguientes:

- Jira Agile. Herramienta orientada a los expertos o principiantes en el uso de métodos ágiles, para la creación de historias, la construcción de un *sprint backlog*, identificar el compromiso del equipo y la velocidad, visualización de las actividades del equipo, informes del progreso del equipo, entre otras. Soporta diferentes metodologías (Atlassian, 2015).
- Rally. Promueve el uso de métodos de agilidad como ventaja competitiva. Organiza la cadena de valor para conectar al cliente a su organización y construir una cultura de colaboración. Se basa en la planeación dinámica y en la colaboración de equipos de trabajo, permite la generación de una APP basada en el diseño y se basa en la metodología Scrum. Ofrece licencias Freemium (Rally, 2015).
- Version One. Conjunto de herramientas para desarrollo de software con filosofía agile y Lean, permite la colaboración con equipos de trabajo, programas y portafolios. Ofrece servicios entrenamiento, asesoría y uso de metodologías ágiles. Se compone de una serie de productos, artículos, seminarios, entre otros elementos para desarrollo de software (VERSIONONE, 2015).
- Pivotal Tacker. Permite el desarrollo de aplicaciones basadas en historias de usuario, que deben ser ordenadas y codificadas posteriormente, genera código automático para el desarrollo de una aplicación, ayuda con la entrega y distribución de aplicaciones (PivotalTracker, 2015).

- Axosoft. Ayuda en la planeación, en el uso de una metodología, en la integración de equipos, a mejorar la calidad del producto, permite la integración con otros sistemas de gestión y tiene soporte en la nube (Axosoft, 2015).
- ScrumDo. Herramienta para planeación, seguimiento, control de tareas, reportes, avances, notificaciones y principalmente promueve un entrega predecible de un producto de software. Ofrece diferentes métodos de planeación y organización en el desarrollo (ScrumDo, 2015).

En la tabla 7.5 se muestra la comparativa de estas herramientas con respecto a las facilidades ofrecidas por MDSIC 2.0.

Tabla 7.5. Comparativa de herramientas para la administración de proyectos ágiles ad-hoc a una APP.
Fuente: elaboración propia.

Características	MDSIC 2.0	Jira Agile	Rally	Version One	Pivotal Tracker	Axosoft	ScrumDo
Metodología	MDSIC-M	Scrum, Kanban	Scrum, Agile, XP y propietaria	Scrum, Lean, SAFE y otras	Agile	Scrum, Kanban y otras	ScrumBan
Entorno integrado	✓	✓	✓	✓	✓	✓	✓
Fácil de usar	✓	✓	✓	✓	✓	✓	✓
Confiable	✓	✓	✓	✓	✓	✓	✓
Buen desempeño	✓	✓	✓	✓	✓	✓	✓
Soporte		✓	✓	✓	✓	✓	✓
Control tiempo	✓	✓	✓	✓	✓	✓	✓
Control costos	✓	✓	✓	✓	✓	✓	✓
Redes sociales	✓	✓				✓	✓
Notificaciones	✓	✓	✓	✓	✓	✓	✓
Documentación		✓	✓	✓	✓	✓	✓
Reportes	✓	✓	✓	✓	✓	✓	✓
Colaboración	✓	✓	✓	✓	✓	✓	✓
En la nube		✓	✓	✓	✓	✓	✓
Licencia	Libre	Freemium	Freemium	Freemium	Freemium	Freemium	Freemium

Costo usuario		US\$ 20 /año	US\$ 588 /año	US\$ 468 /año	US\$ 36 /año	US\$ 10 /año	US\$ 35.6 /año
Generación de código automático		✓	✓	✓	✓		
Compatibilidad con otras herramientas		✓	✓	✓	✓	✓	✓

Como se observa MDSIC 2.0, es una alternativa para el desarrollo de APPs en un mercado global, sin embargo características como el soporte, la documentación, uso en la nube, generación de código automático y la compatibilidad con otras herramientas todavía faltan por desarrollarse. Otro punto importante es que MDSIC 2.0 si ofrece la generación de información acorde la evaluación de la calidad apegado a la NOM. En México no se reportan en la literatura desarrollos que ofrezcan entornos de planeación y desarrollos de aplicaciones. Esta es quizá la ventaja competitiva de MDSIC 2.0, una herramienta ad-hoc al mercado nacional.

7.5 Experiencias de desarrollo con las PyMEs.

Como parte de las pruebas piloto de la herramienta y la metodología se realizaron 2 proyectos reales, para censar el comportamiento de los equipos en un ambiente de desarrollado de una APP. Se realizaron los proyectos mencionados en capítulo 4, en donde se revisó el impacto y pertinencias de la metodología ad-hoc a las necesidades y las expectativas de las PyMEs.

El proyecto 1 fue aplicado en una empresa un grupo de empresas pequeñas (FS) de la región que cuenta con 5 personas; el administrador, el de soporte y tres programadores. El personal de desarrollo son ingenieros computación egresados de universidades locales con conocimientos en el desarrollo de APPs, pero con pocas bases en la administración y ejecución de un proyecto. Para el desarrollo de la APP que requería el cliente se realizaron varias reuniones para establecer los requerimientos e iniciar la planeación. Para conocer la funcionalidad de MDSIC-M, se realizó un curso rápido con las características y las buenas prácticas propuestas. Se observó que el liderazgo del administrador de proyectos es importante ya que es quién decide

junto con el cliente la factibilidad del proyecto. Se revisó junto con el personal (curso de 4 horas), la propuesta de MDSIC-M y sus fases. Se explicó la importancia de llevar el control de tiempos y costos para generar una APP con calidad. La resistencia a una metodología es quizá el mayor reto en este tipo de empresas.

Se les explicó también el funcionamiento de MDSIC 2.0 y como aplicarlo, se elaboró un ejemplo rápido de los principales elementos de la metodología y se guió en los tres días siguientes el arranque del proyecto y la ejecución. Los resultados de este proyecto piloto se muestran en tabla 7.2.

Un conclusión importante es que la madurez de los procesos y la validación de la calidad estructural, funcional y de procesos quedó al 60 % un nivel de madurez 2; parcialmente alcanzado. Esto demuestra que las PyMEs deben madurar con el uso de la metodología hasta alcanzar un nivel óptimo en todos los procesos.

En el caso del proyecto 2 se aplicó en una FS en un grupo de empresas medianas, en donde existe un equipo de 10 personas con los roles de: 1 promotor/administrador, 2 de calidad, 5 desarrolladores y 2 diseñadores. El proyecto atendía a las necesidades de 3 clientes (diferentes) y 10 usuarios finales. El desarrollo de la APP en un entorno de requerimientos cambiantes fue el desafío mayor. Se realizó un curso de 5 días donde explicó el funcionamiento de MDSIC-M y MDSIC 2.0. Se apoyó continuamente al promotor de este proyecto con entrenamiento y seguimiento del proyecto. En este caso se observó que el proceso de planeación tuvo mejores resultados ya la calidad funcional censada fue acorde a los tiempos y costos establecidos en plan inicial, se llegó a un nivel de 91 % con madurez 4; proceso terminado y repetible. Sin embargo en el desarrollo y generación de la APP se tuvieron muchos retrasos por la organización entre los diseñadores, los desarrolladores y los requerimientos cambiantes de los usuarios. Ocurrieron muchos componentes con fallas y tuvieron que ser refactorizados, esto llevó a una calidad estructural y de procesos con un 60%, lo que llevó a un nivel de madurez 2; proceso terminado no repetitivo. Es importante mencionar que tanto, el promotor, como los programadores y diseñadores son

personal capacitado con mucha experiencia en el desarrollo de APPs. En la figura 7.2 se muestra el nivel de madurez por proceso y por proyecto.

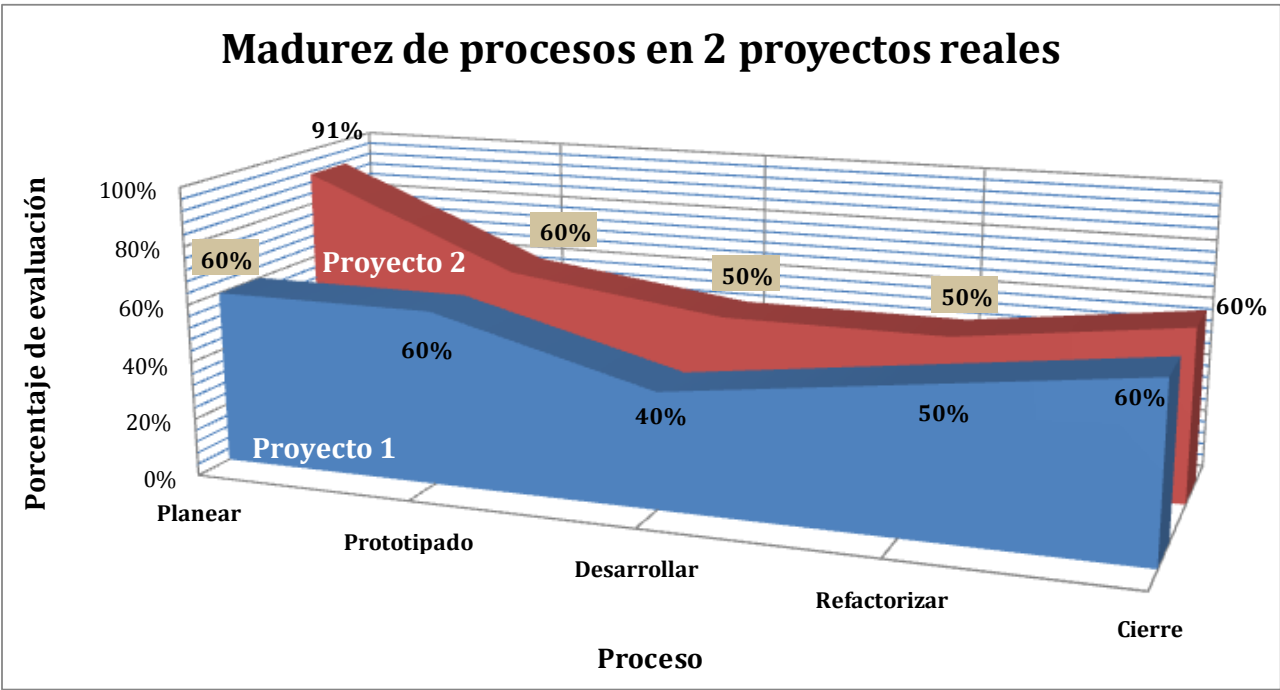


Figura 7.2. Evaluación de la madurez de procesos MDSIC-M en dos proyectos reales.
Fuente: elaboración propia.

7.6 Trabajos futuros.

Como se observó en las tablas comparativas anteriores, todavía hay temas que mejorar en la metodología, a pesar de su aplicación en proyectos reales falta más pruebas sobre el correcto funcionamiento de esta. La validación de la calidad y el apego a la NOM todavía es un tema subjetivo que debe afinarse con la evaluación de los KPIs en más proyectos y que den certidumbre a la calidad que se necesita en el desarrollo de una APP. A continuación se describen las áreas de oportunidad para mejorar la metodología:

- A) Mejoras en scheduler. Aumentar la experiencia del usuario con una interfaz más amigable, que permita la definición y el establecimiento de los recursos, tiempos, costos y riesgos de una tarea. Que ofrezca también un seguimiento preciso de la calidad en los procesos y que ayude a establecer el nivel

- B) Considerar nuevas tendencias en el desarrollo. Una tema que no fue desarrollado en la herramienta MDSIC, fue el seguimiento al desarrollo de módulos, artefactos y componentes, de tal forma que el desarrollo sea guiado en forma precisa. Permitiendo con esto certidumbre en el desarrollo de componentes, en las pruebas por componentes, en las pruebas de integración, en las pruebas de comunicación y en las pruebas del sistema. Esto ayudaría a mejorar en gran medida la refactorización de componentes al generar una APP.
- C) Considerar PSP y TSP, métricas e impacto en la calidad. A pesar de que la calidad se censa por KPAs y se evalúan los KPIs por fases. Todavía falta integrar las métricas en la fase de desarrollo y en el PSP; las habilidades del equipo de desarrollo, la disciplina, las métricas de rendimiento, la estimación y planeación de tareas en el desarrollo, la administración de la calidad en compontes de código, entre otros. Falta también integrar métricas del TSP; en los planes por equipo, la comunicación en el equipo, el establecimiento de metas e hitos en el trabajo en equipo y censar la calidad en equipo. Se observo que este punto es vital ya que la calidad y madurez del proceso de desarrollo siempre fue uno de menor valor.
- D) Resaltar las ventajas de la metodología. MDSIC-M es una conjunto de buenas prácticas organizados por fases y procesos, esto no garantiza que en el desarrollo de una APP sea aplicadas todas las recomendaciones y seguimientos propuestos, sin embargo los procesos y subprocesos definidos si ayudan a los desarrolladores a considerar temas que quizá no conocían y que son importantes para una APP de calidad.
- E) Reducir la curva de aprendizaje de la metodología MDSIC-M. Se observó que en los proyecto piloto en dos PyMEs reales, un curso de capacitación y el “*coaching*” son favorables, ya que durante el desarrollo de los primeros proyectos es de vital importancia conocer la metodología, se requiere entonces de aumentar el acervo de documentos de ayuda, videos y asesorías para una mejor absorción de parte de la PyMEs de la metodología.
- F) Transferencia del conocimiento. El establecimiento de un modelo para la transferencia del conocimiento técnico de MDSIC-M, permitiría a las PyMEs establecer en el corto, mediano y largo plazo mejoras en la competitividad e innovación, en la industria del desarrollo móvil. Este ayudaría al desarrollo económico de las PyMEs y de la región. El modelo debe centrarse en mejorar el conocimiento explicito sobre las prácticas que ayudan a la calidad

en la construcción de APPs; debe enfatizar la absorción en el individuo, el equipo, la organización y la inter-organización. Creando con esto un conocimiento tecnológico especializado en la disciplina de las TICs.

- G) Transferencia tecnológica. La metodología ofrece ventajas al tratar de seguirla y adoptarla en una PyME, sin embargo es necesario un esquema de transferencia tecnológica en donde los equipos de desarrollo, de calidad, de diseño y de administración de proyectos sean apoyados para seguir el modelo y la metodología, logrando con esto una cultura de innovación, desarrollo sustentable, competitividad y emprendurismo en el desarrollo de APPs, cada vez más complejas y con características nuevas.
- H) Análisis de decisiones para la selección del tipo de proyecto que puede ser viable para la PyME (planeación estratégica). Un tema que no se desarrolló en la propuesta de MDSIC-M fue el establecimiento de un árbol de decisiones que permita las PyMEs analizar la viabilidad de un proyecto en base a su experiencia y lecciones aprendidas. La construcción del diamante de Porter y el análisis FODA de la PyME es el camino a seguir para el establecimiento del análisis de decisiones. Establecer un método que ofrezca con claridad la factibilidad y la viabilidad de un proyecto se vuelve importante para los administradores de una PyME evitando con esto costos y e inversiones infructuosas.
- I) Generar conocimiento fundamentado en el Social Business. Explotar el *social business media* se convierte en una prioridad para las PyMEs, el conocimiento y la aplicación de nuevos métodos y técnicas en el desarrollo móvil es el objetivo. La escucha, los grupos conversacionales, la información viral, el soporte y la integración de los clientes a través de este medio se convierte en una prioridad. El cambio cultural hacia la apertura de la PyMEs y el intercambio de información es una forma de empoderarse en esta industria, donde el *social business* permite acelerar la productividad, la innovación y el reconocimiento de los usuarios y clientes potenciales. Se requiere entonces construir un modelo de competitividad que permita explotar esta nuevo entorno de negocios.
- J) Fortalecimiento de los CTI. La utilización de MDSIC-M permitirá a las PyMEs innovar, el desarrollo con calidad permitirá acceder a los incentivos promovidos por programas o estrategias de fortalecimiento de las TI. Fomentando el fortalecimiento al capital humano, exportaciones de APPs y la competitividad a nivel regional, nacional e internacional.

Permitirá también ayudar a mejorar la madurez de los procesos de desarrollo, logrando con esto el acceso al financiamiento, certeza jurídica, la difusión de las TI y finalmente a la integración en clúster de TI que permita el intercambio de conocimiento.

7.7 Aportación de MDSIC-M como metodología ágil para FS.

MDSIC-M mejora aquellos desarrollos que requieren de tiempos rápidos de respuesta y la generación de versiones beta, permite un desarrollo organizado y contempla los estándares de calidad al proponer la utilización de procesos y el censado de la calidad.

El “*Global Delivery*” queda fuera de esta metodología sin embargo puede ser una oportunidad de mejora a futuro para ayudar al proceso completo del desarrollo móvil. Se están realizando pruebas experimentales con empresas de la región para observar la utilidad de la metodología y conocer si es una buena alternativa para fábricas de software en México.

Se ha observado un creciente desarrollo de aplicaciones Web que es una de las principales demandas en el mercado, la metodología que proponemos puede ser aplicada en este tipo de desarrollo y ayudar así a obtener una ventaja competitiva; un proyecto a futuro es probar mediante este enfoque el desarrollo Web y observar cual será su comportamiento.

CMMI y Moprosoft proponen etapas o niveles de madurez en el desarrollo de una fábrica de software, los KPIs censados en MDSIC-M ayudarán a las empresas PyMEs a lograr el objetivo de apegarse a un estándar o NOM de calidad, la documentación y generación de reportes de MDSIC 2.0 ofrece una forma fácil y sencilla de lograr este objetivo.

Como resultado de las comparativas de la MDSIC-M con respecto otras metodologías se puede observar en la figuras 7.3 a la figura 7.10 el comportamiento al censar la agilidad.

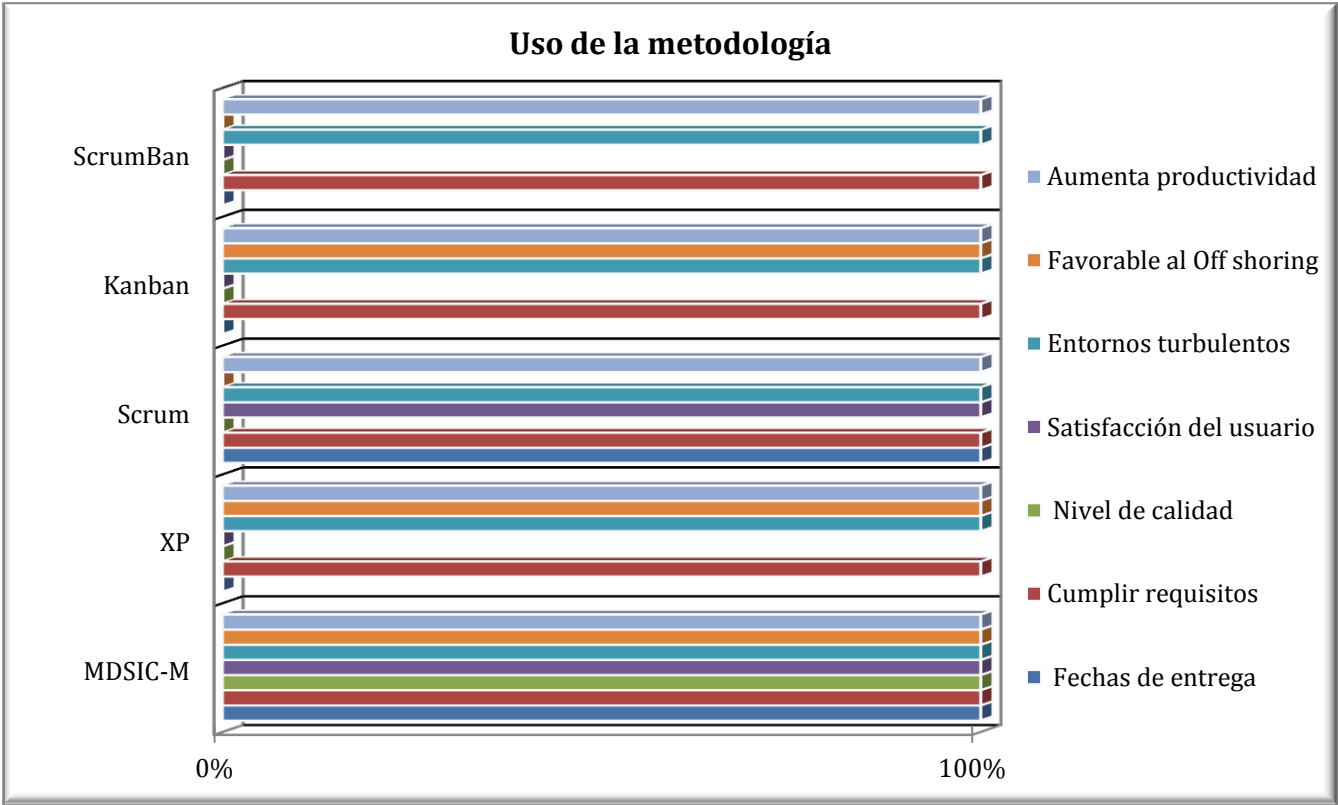
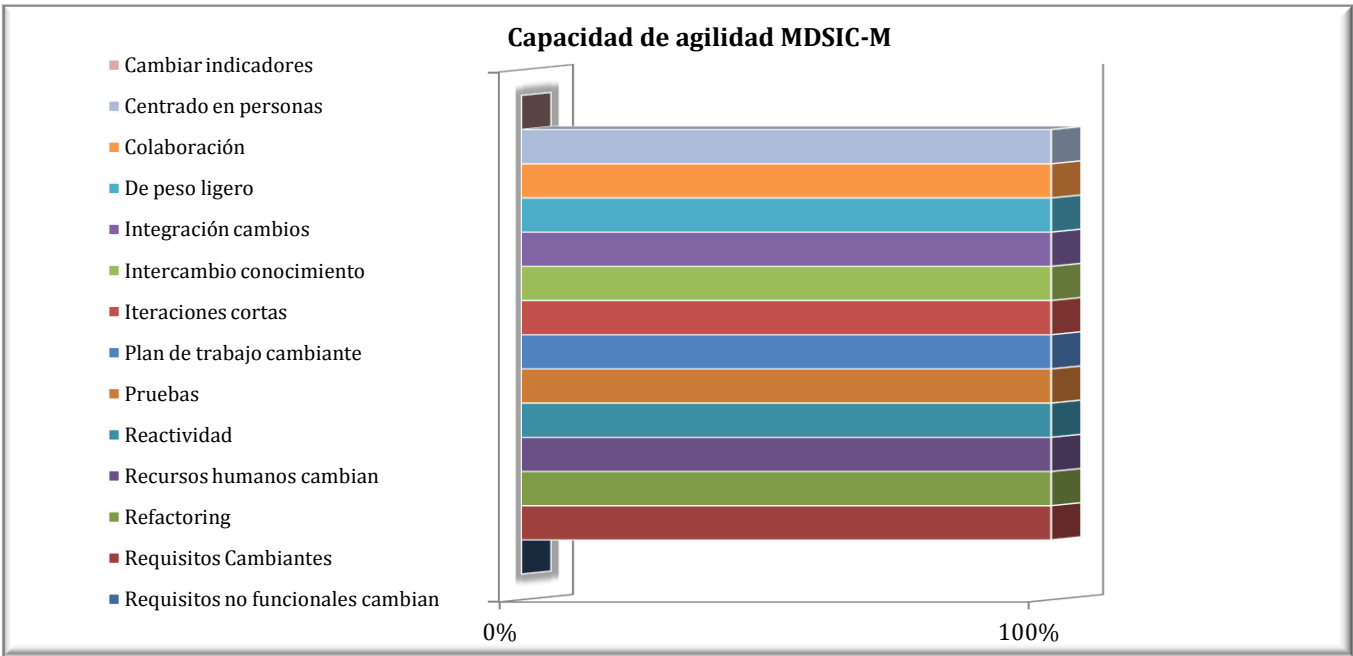
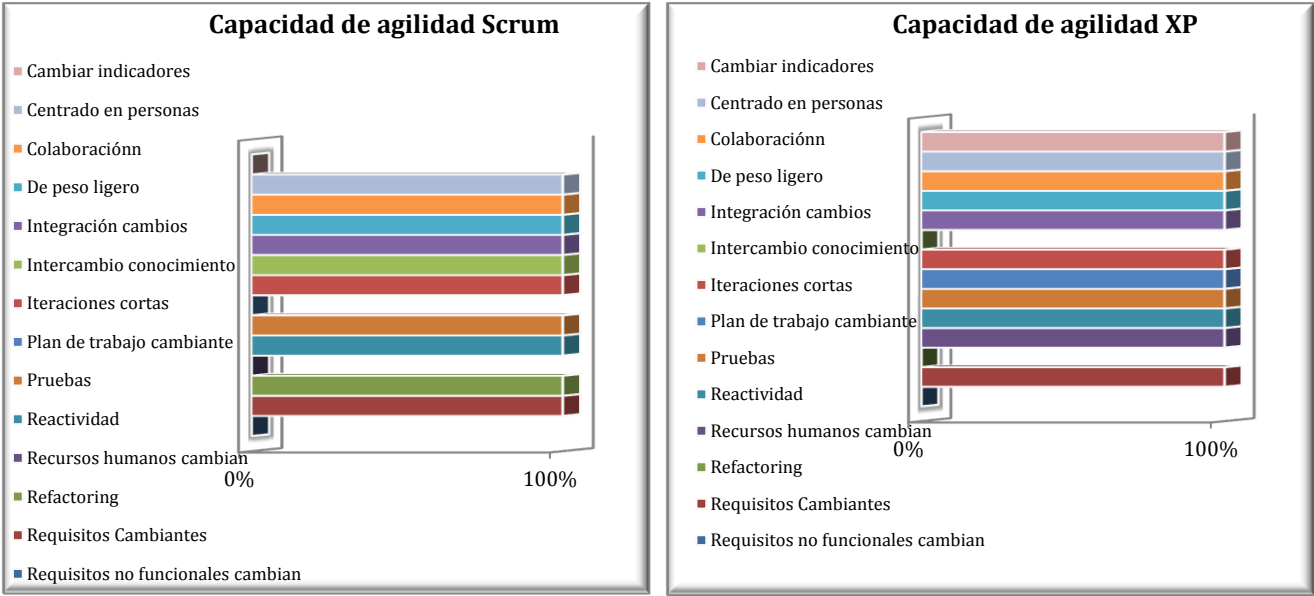


Figura 7.3. Comparativa del uso de la metodología en entornos ágiles.
Fuente: basado en (Pérez P. & González C., 2012).



A) Capacidad de MDSIC-M como método ágil.



B) Capacidad de agilidad de Scrum.

C) Capacidad de agilidad de Scrum.

Figura 7.4 A), B) y C). Capacidad de agilidad de las metodologías analizadas.
Fuente: basado en (Pérez P. & González C., 2012).

De igual forma se compara la aplicabilidad de las metodologías en APPs y el resultado se puedes observar en la figura 7.5.

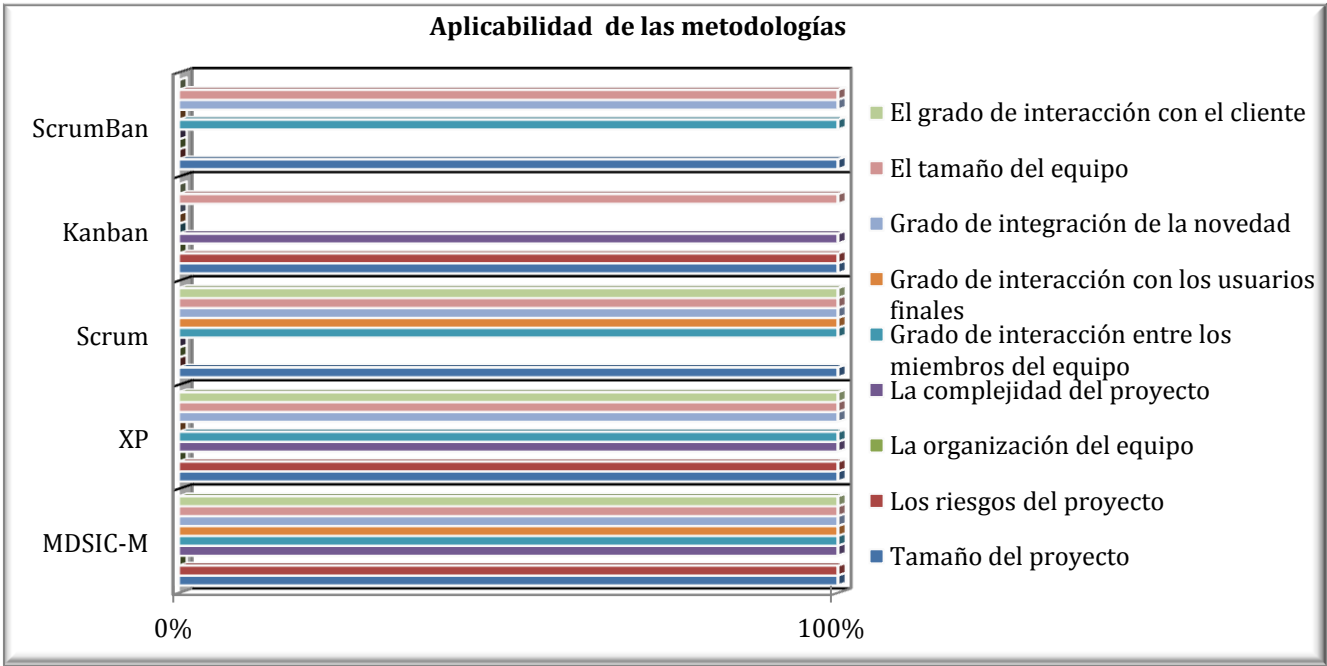


Figura 7.5. Aplicabilidad de las metodologías analizadas.
Fuente: basado en (Pérez P. & González C., 2012).

Se comparan también en lo referente a procesos y normas de un desarrollo; A) en el nivel de abstracción, B) en las actividades cubiertas por el método ágil y C) en los productos de las actividades del método ágil. Finalmente se comparan las metodologías con los totales de uso, capacidad de agilidad, aplicabilidad y procesos y productos. En la figuras 7.6, 7.7, 7.8 y 7.9 se muestran los resultados.

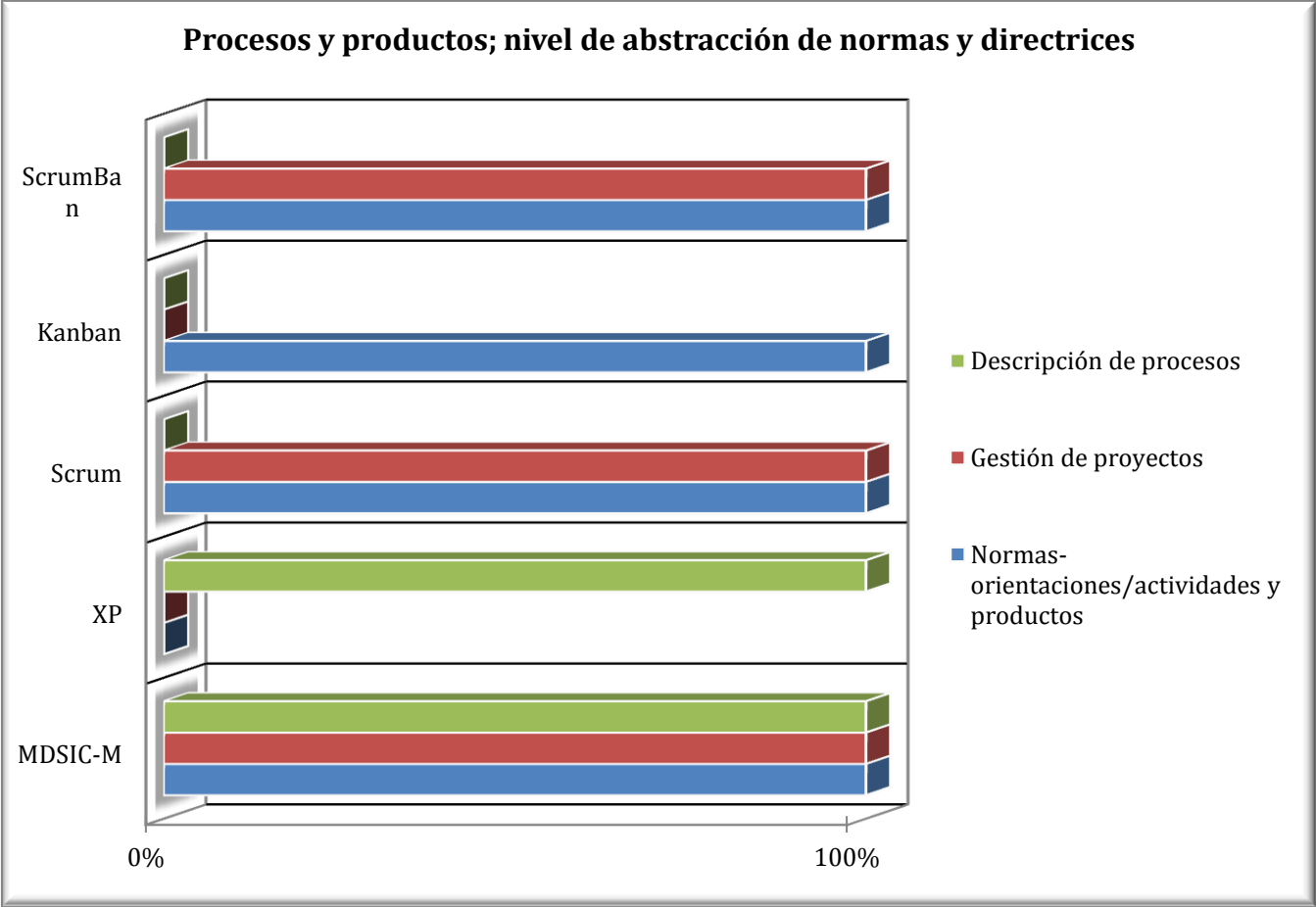


Figura 7.6. Comparativa con respecto a procesos y productos, nivel de abstracción de normas y directrices.
Fuente: basado en (Pérez P. & González C., 2012).

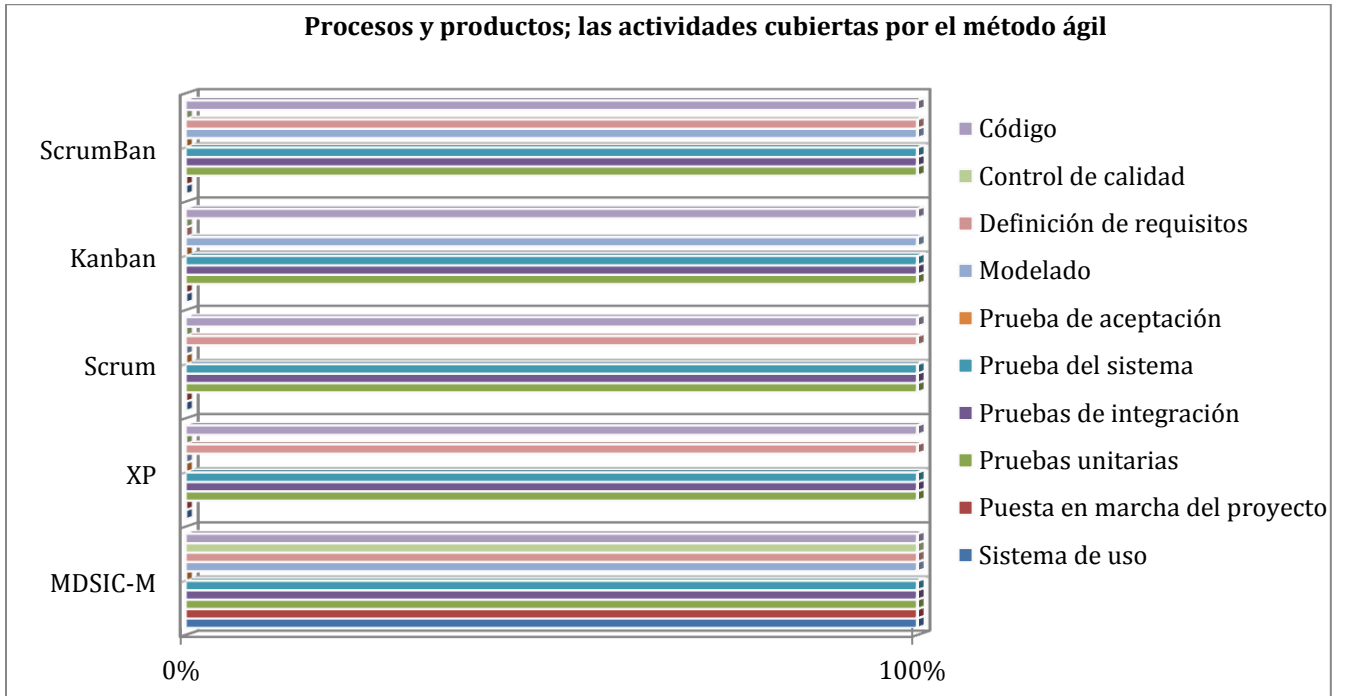


Figura 7.7. Comparativa con respecto a procesos y productos, las actividades cubiertas por el método ágil.
Fuente: basado en (Pérez P. & González C., 2012).

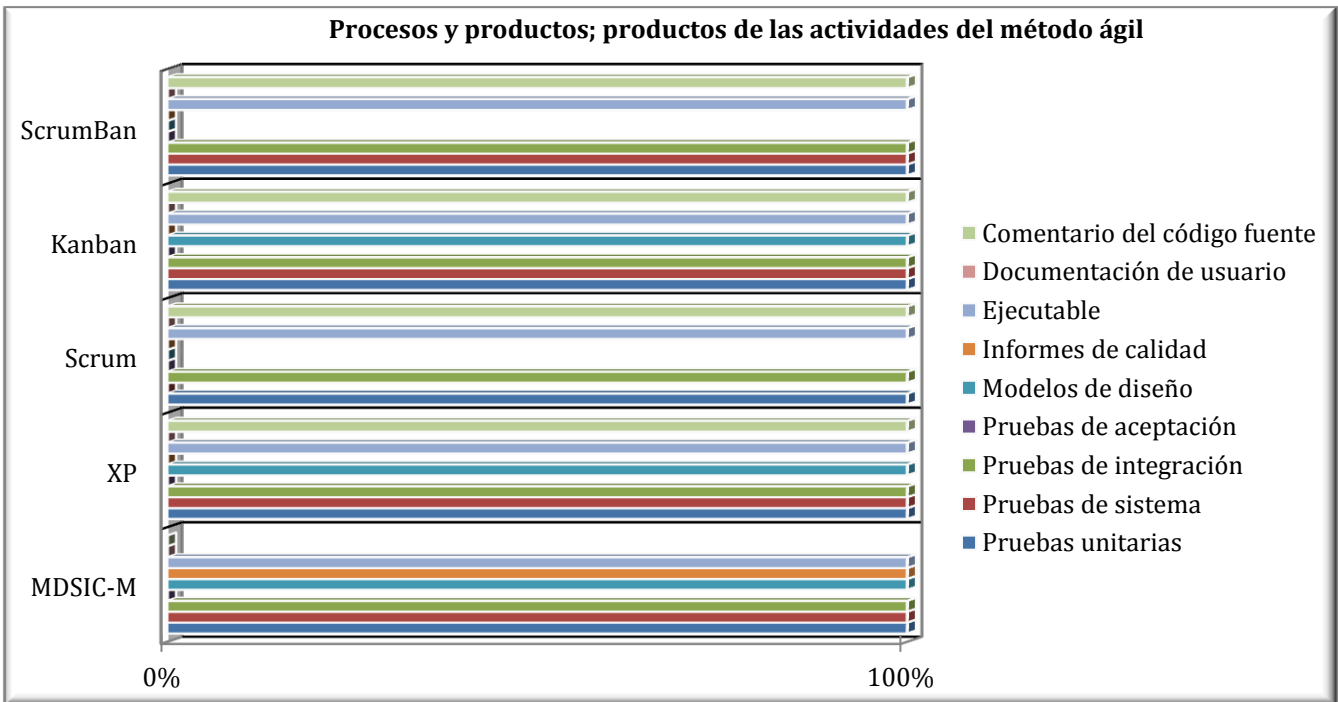


Figura 7.8. Comparativa con respecto a procesos y productos, productos de las actividades del método ágil.
Fuente: basado en (Pérez P. & González C., 2012).

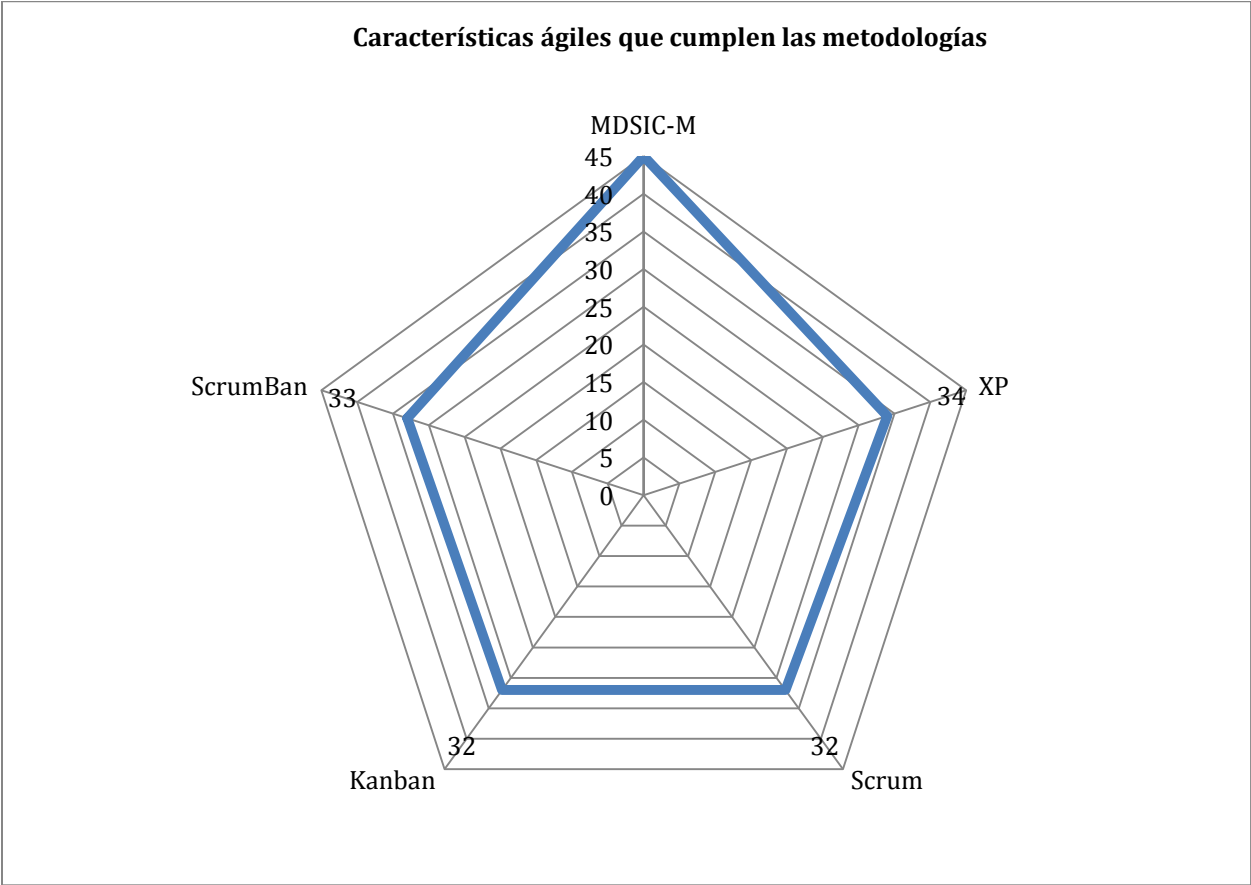


Figura 7.9. Totales que si se logran en las características censadas de las metodologías ágiles.
Fuente: basado en (Pérez P. & González C., 2012).

Como se observó en las comparativas la metodología MDSIC-M sale bien librada en lo referente a las características de un método ágil, sin embargo esto no demuestra que es mejor metodología, solo afirma que los objetivos de diseño que se buscaron desde la definición de la metodología se cumplen. Como se mostró en la tabla 7.5, se realizó un comparativa de acuerdo a los indicadores de desempeño que las metodologías mencionan en la literatura, de esto se observa que nuevamente MDSIC-M ofrece una cantidad mayor de KPIs (que se basan en sus procesos de evaluación de la calidad). En la figuras 7.10, 7.11 y 7.12 se muestran las comparativas de los KPIs y las metodologías más ad-hoc al desarrollo de una APP.

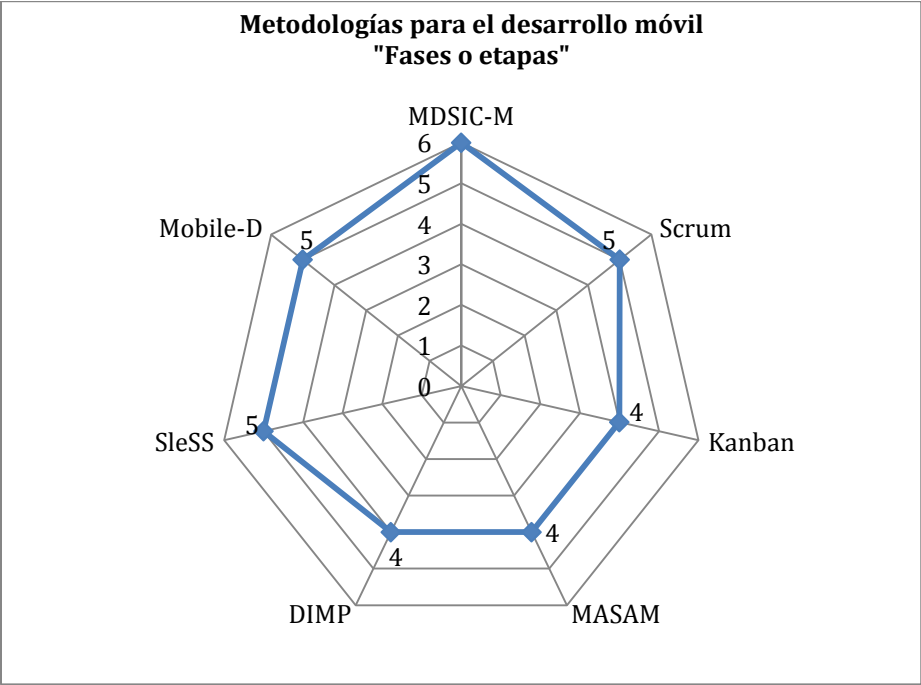


Figura 7.10. Fases o etapas que considera la metodología para desarrollo de APPs.
Fuente: elaboración propia.

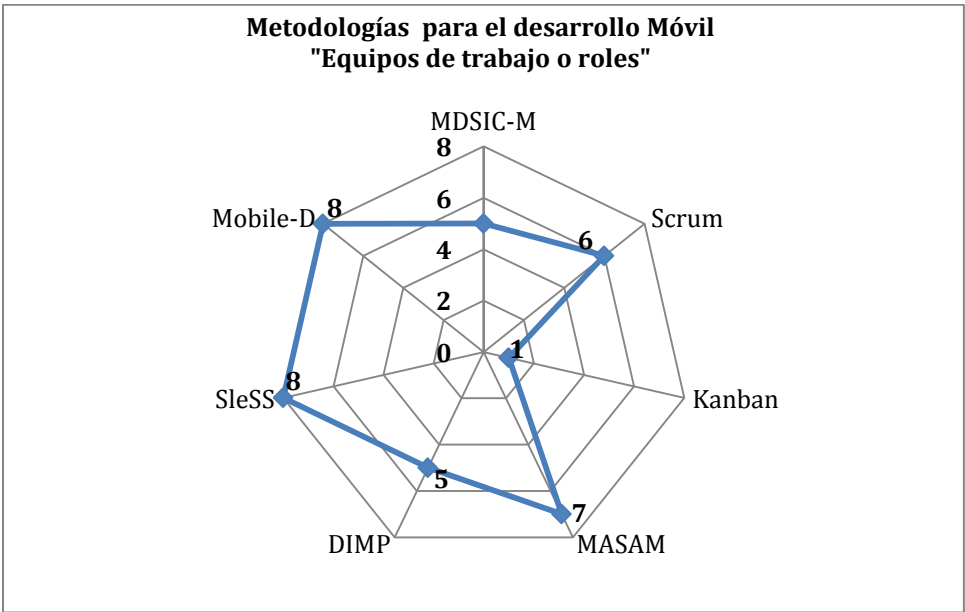


Figura 7.11. Número de miembros en los equipos de trabajo y roles de las metodologías.
Fuente: elaboración propia.

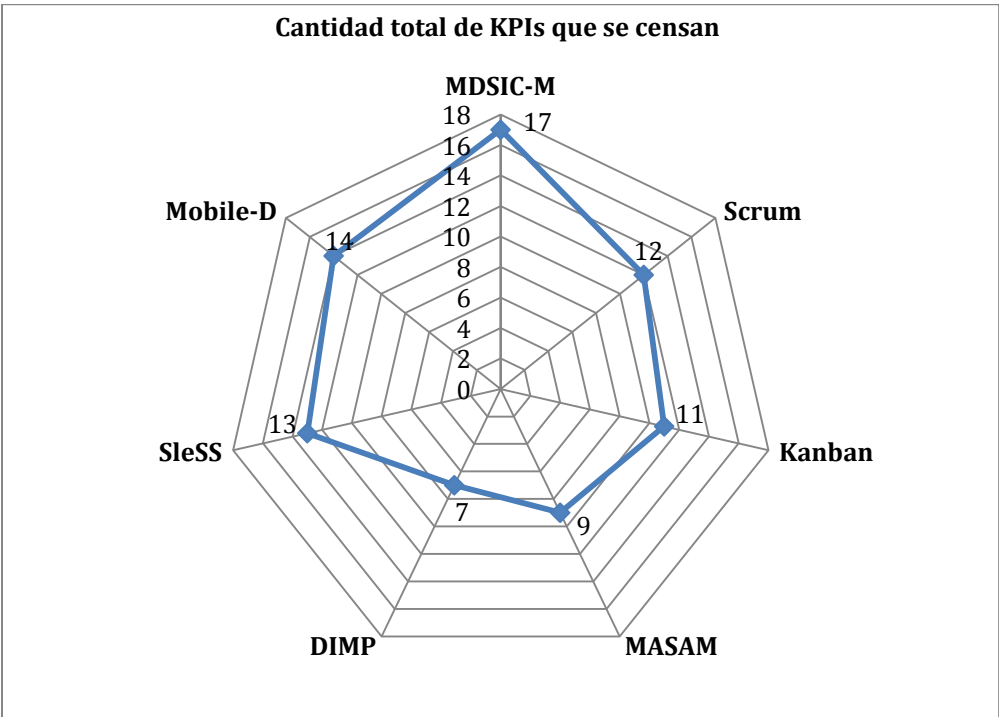
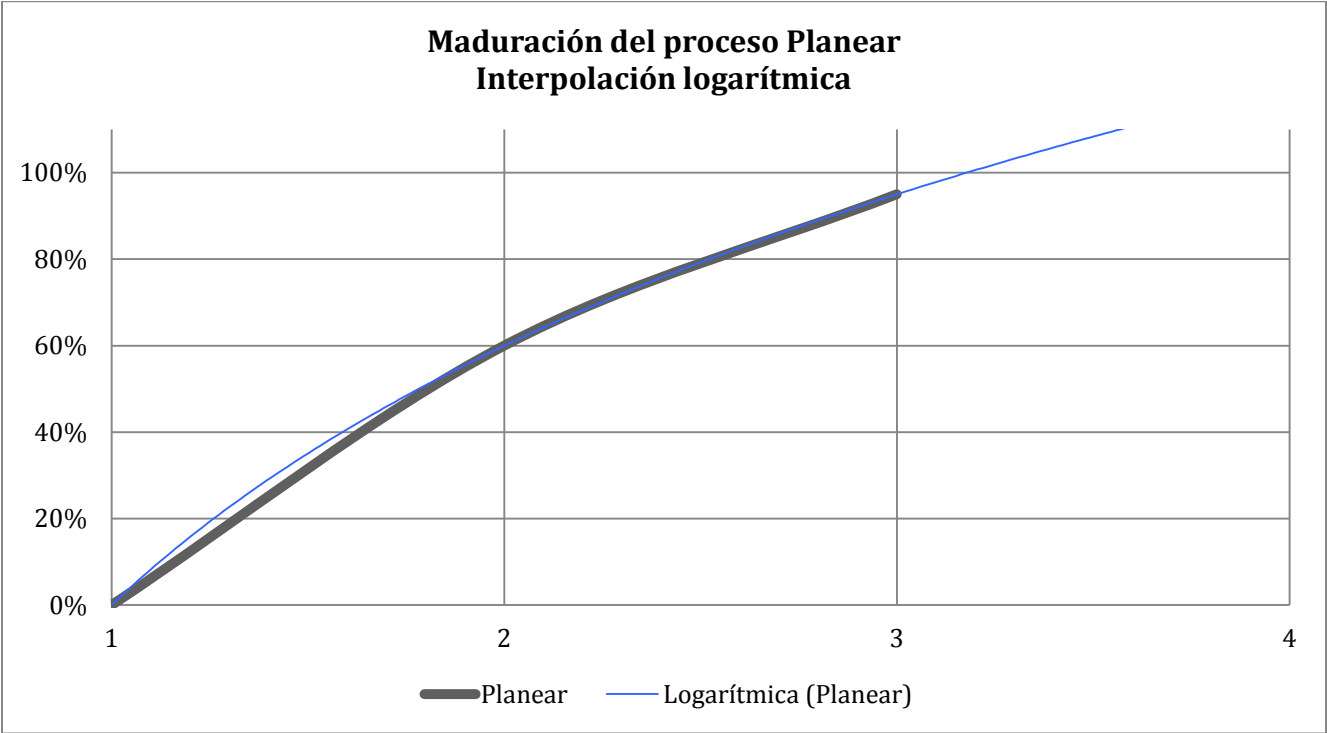
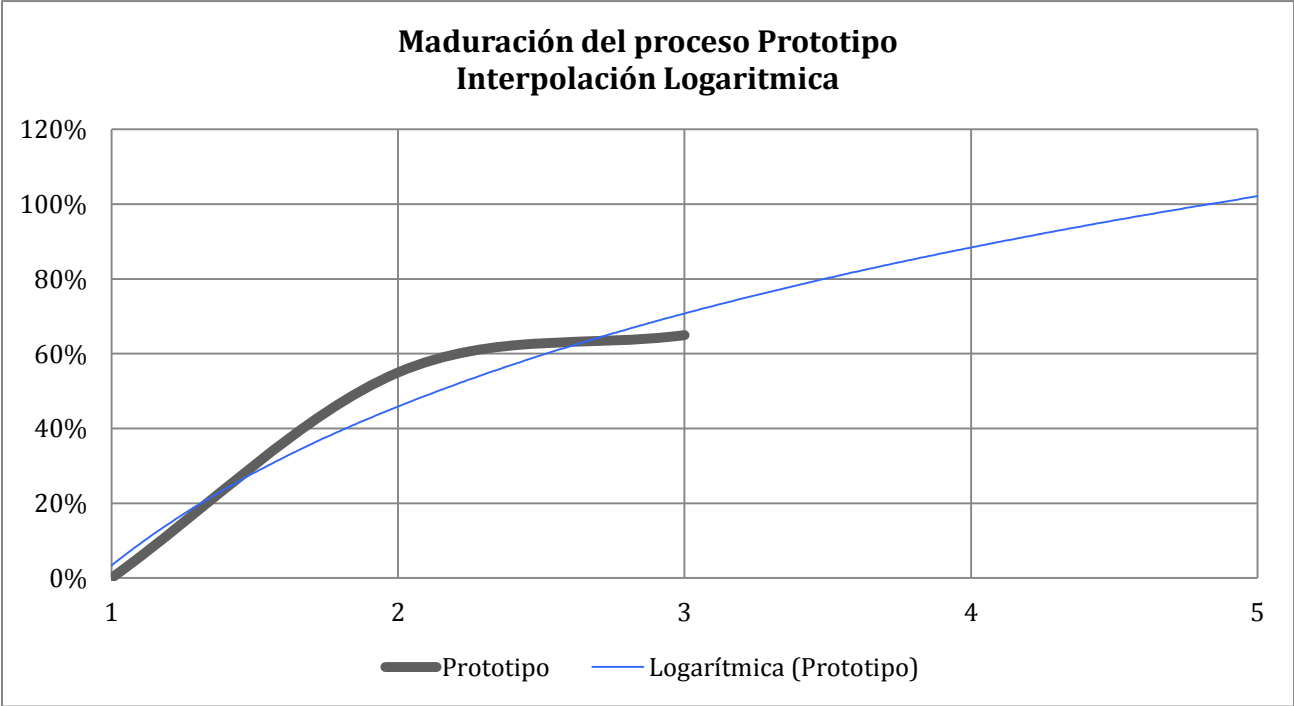


Figura 7.12. Número de KPIs que censan las metodologías en la evaluación de calidad.
Fuente: elaboración propia.

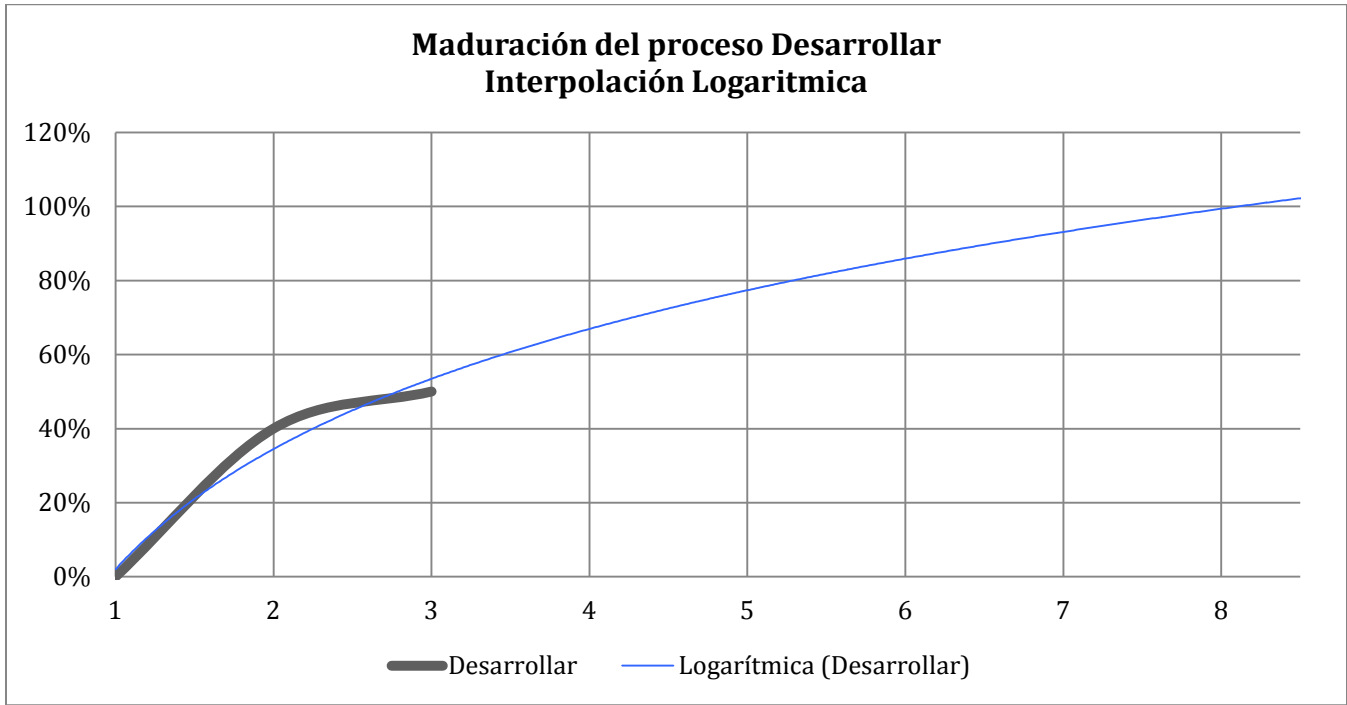
Para la evaluación de la calidad en MDSIC-M se censaron de acuerdo al modelo de Chappel la calidad estructural, funcional y de proceso en desarrollo de una APP. Al evaluar la calidad es posible acumular por KPI y por proceso la cantidad de indicadores logrados y su porcentaje de logro, de esta evaluación se puede establecer el nivel de cada uno de los procesos de la metodología. Como parte esencial del cuasi experimento se evaluó la calidad en estos proyectos y se logró establecer por interpolación polinomial el número de proyectos que deben probarse en una PyME para llegar a al 100% calidad en el proceso y alcanzar un nivel madurez 6; optimizado. En la figura 7.12 A), B), C), D) y E) se muestra la interpolación logarítmica como estimación y la posibilidad de llegar a un 100% de madurez aplicando MDSIC-M para el desarrollo de una APP.



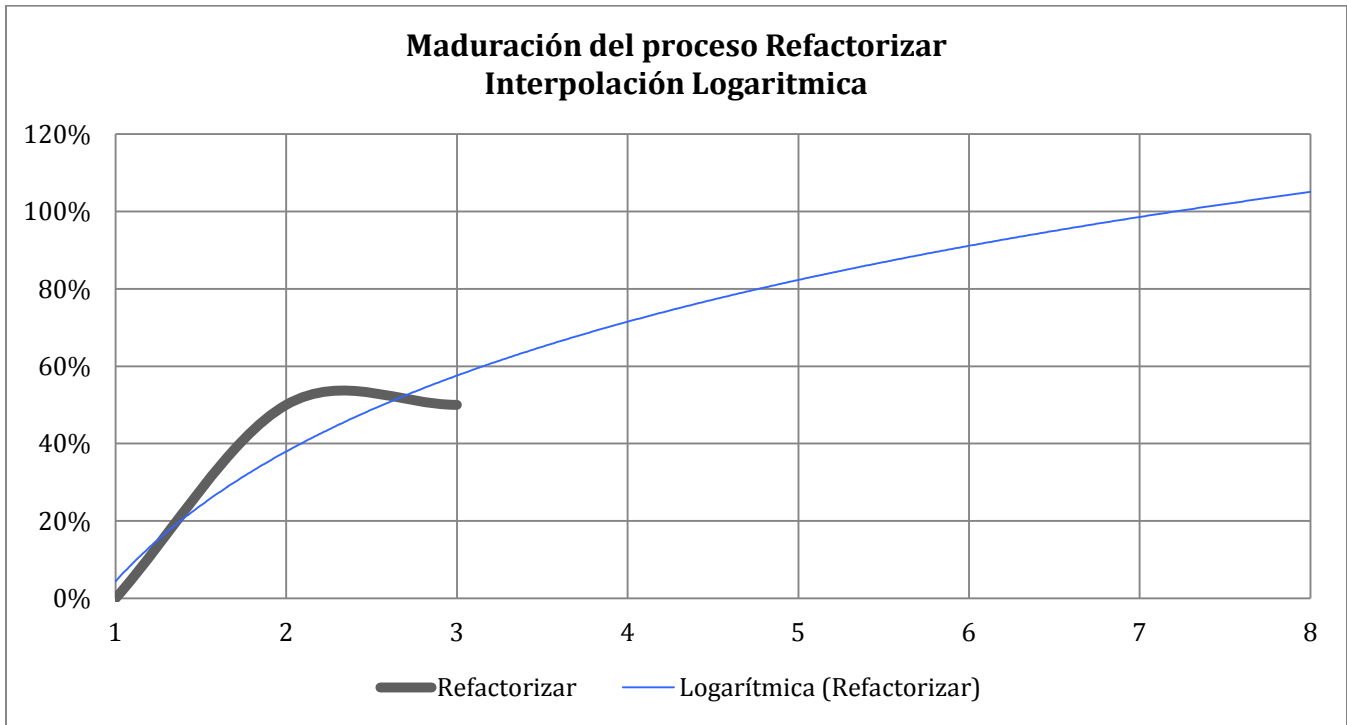
A) Interpolación de la madurez del proceso Planear para el cuasi experimento, se requieren 3 proyectos.



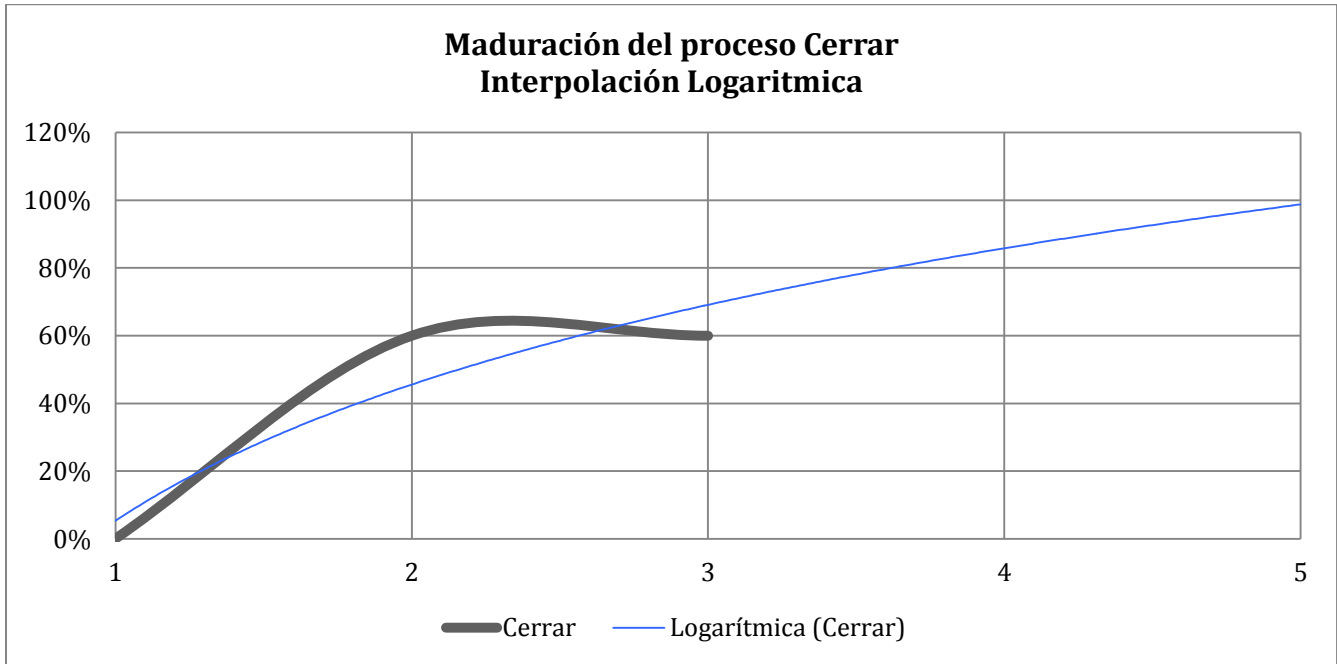
B) Interpolación de la madurez del proceso Prototipo para el cuasi experimento, se requieren más de 4 proyectos.



C) Interpolación de la madurez del proceso Desarrollar para el cuasi experimento, se requieren más de 8 proyectos.



D) Interpolación de la madurez del proceso Desarrollar para el cuasi experimento, se requieren 7 proyectos.



E) Interpolación de la madurez del proceso Cerrar para el cuasi experimento, se requieren más de 4 proyectos.

Figura 7.13 A), B), C), D) y E). Madurez de los procesos de MDSIC-M de acuerdo a la calidad censada en el cuasi experimento. Fuente: elaboración propia.

Finalmente como ventaja de la metodología se propone almacenar historias de proyectos, en donde el tamaño el tiempo, los costos y los riesgos que ocurrieron durante un proyecto son guardados y documentados, permitiendo a una empresa consultar éstas experiencias y no repetir errores, mejorando así en su proceso de desarrollo. Se almacenan también los componentes de reuso y esta es la una de las mejores alternativas para competir en el mercado de desarrollo móvil, además de fomentar el desarrollo económico de las FS de la región.

En el anexo A se muestra el diccionario de datos de la base de datos utilizada. En el anexo B se documentan los procesos y subprocesos de MDSIC-M. En el anexo C se presentan los manuales de usuario y técnico de la herramienta MDSIC 2.0. En el anexo D se muestra el desarrollo del scheduler utilizando las herramientas de desarrollo.

Bibliografía

1. Abowd, G., Hayes, G., Iachello, G., Kientz, J., Patel, S., Stevens, M., & Truong, K. (2005). Prototypes and paratypes: designing mobile and ubiquitous computing applications. *Pervasive Computing, IEEE*, 67 - 73.
2. Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jäälinoja, J., Korkala, M., . . . Salo, O. (2004). Mobile-D: an agile approach for mobile application development. *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications* (págs. 174-175). New York, USA: ACM.
3. Adler, B. (2014). *The 8 Mobile APP Metrics That Matter*. Recuperado el 1 de 2 de 2015, de localytics: <http://info.localytics.com/blog/the-8-mobile-app-metrics-that-matter>
4. agile manifesto. (2001). *agilemanifesto.org*. Recuperado el 20 de 2 de 2015, de Manifiesto por el Desarrollo Ágil de Software: <http://www.agilemanifesto.org/iso/es/>
5. Agile2011. (7 de 4 de 2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, 85, 1213-1221.
6. Agile-Tools. (2013). *Agile Project Management Tools*. Recuperado el 5 de 1 de 2015, de Agile Tools: <http://www.agile-tools.net/>
7. AIMSÍ. (2014). *Clúster de Tecnologías de información Colima*. Recuperado el 11 de 1 de 2014, de AIMSÍ: <http://aimsi.org/cluster-de-tecnologias-de-informacion-colima/>
8. Ambler, S. W. (2014). *Agile Modeling and the Rational Unified Process (RUP) - See more at:* <http://www.agilemodeling.com/essays/agileModelingRUP.htm#sthash.iJt6OalS.dpuf>. Recuperado el 10 de 1 de 2015, de agilemodeling.com: <http://www.agilemodeling.com/essays/agileModelingRUP.htm>
9. AMESOL. (2015). *AMESOL; Asociación Mexicana Empresarial de Software Libre*. Recuperado el 15 de 1 de 2015, de amesol.ac: <http://www.amesol.ac/>
10. AMIPCI. (2014). *Asociación Mexicana de Internet*. (AMPSI, Productor) Recuperado el 16 de 1 de 2015, de amipsi.org.mx: <https://www.amipci.org.mx/es/>
11. AMITI. (2015). *AMITI; Mejores empresas de TI para México*. (AMITI, Productor) Recuperado el 10 de 1 de 2015, de AMITI.org.mx: <http://amiti.org.mx/>
12. AMITI. (s.f.). *ASOCIACIÓN MEXICANA DE LA INDUSTRIA DE TECNOLOGÍAS DE INFORMACIÓN*. (AMITI) Recuperado el 5 de 12 de 2014, de <http://amiti.org.mx/>
13. AMITI, INFOTEC. (2013). <http://amiti.org.mx/>. Recuperado el 1 de 2 de 2015, de ESTUDIO DE PERSPECTIVAS Y ESTRATEGIAS DE DESARROLLO Y DIFUSIÓN DE APLICACIONES MÓVILES: http://amiti.org.mx/wp-content/uploads/2013/10/RE_Estudio-APPS.pdf
14. ANADIC. (2015). *ANADIC; Nadie Mejor que todos juntos*. Recuperado el 15 de 1 de 2015, de anadic.com: <http://www.anadic.com.mx/>



UPAEP – Secretaría General

Dirección General de Apoyos Académicos

Dirección del Centro de Recursos para el Aprendizaje y la Investigación.

Biblioteca Central - **Karol Wojtyla**

Tesis Digitales Restricciones de uso:

DERECHOS RESERVADOS ©

PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de textos, imágenes, gráficas, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente de donde la obtuvo mencionando el autor o autores involucrados en el documento.

Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

15. ANIEI. (2014). *ANIEI; Asociación Nacional de Instituciones de Educación en Tecnologías de la Información*. Recuperado el 20 de 1 de 2015, de ANIEI.org: <http://www.aniei.org.mx/ANIEI/>
16. Apache Tomcat. (2015). *Apache Tomcat*. Recuperado el 12 de 2 de 2015, de Tomcat: <http://tomcat.apache.org/>
17. Apple IOS. (2015). *IOS Apple*. Recuperado el 20 de 2 de 2015, de What is IOS: <https://www.apple.com/ios/>
18. Atlassian. (2015). *JIRA Agile*. Recuperado el 15 de 2 de 2015, de JIRA: <https://es.atlassian.com/software/jira/agile>
19. Axosoft. (2015). *Take back your development time*. Recuperado el 24 de 2 de 2015, de Axosoft: <http://www.axosoft.com>
20. Bednar, J. A., & Robertson, D. (2005). *University of Edinburgh Informatics*. Recuperado el 15 de 1 de 2015, de Teaching Courses: http://www.inf.ed.ac.uk/teaching/courses/seoc2/2004_2005/slides/quality.pdf
21. Bergström, F., & Engvall, G. (2011). *Development of handheld mobiles applications for the public dectro in Android and iOS using agile Kanban process tool*. Linköping University , Science and Thecnology. Norrköping, Sweden: Linköping Electronic Press.
22. BlackBerry. (2015). *Blackberry Global*. Recuperado el 15 de 2 de 2015, de Blackberry: <http://global.blackberry.com/es.html>
23. Bohem, B. (2002). Get ready for agile methods, with care. (IEEE, Ed.) *Computer*, 35(1).
24. CANIETI. (2015). *CANIETI; Cámara Nacional de la Industria Electrónica, de Telecomunicaciones yTecnologías de la Información*. Recuperado el 2 de 1 de 2015, de canieti.org: <http://www.canieti.org/Inicio.aspx>
25. Cendejas V., J. L., Vega L., C. A., Careta I., A., Gutierrez S., O., & Ferreira M., H. (Nov de 2014). Design of the integrated collaborative model for agile development software in the central-western companies México. *Nova Scientia*(13), 1-12.
26. Chappel, D. (10 de 1 de 2014). *DavidChappel & Associates*. Recuperado el 1 de 12 de 2014, de www.davidchappel.com: http://www.davidchappell.com/writing/white_papers/The_Three_Aspects_of_Software_Quality_v1.0-Chappell.pdf
27. CIAT. (2014). *Cluster Integradores de Alta Tecnología Jalisco*. Recuperado el 10 de 11 de 2014, de CIAT: <http://www.ciat.mx/>
28. CLUSTERTIM. (2014). *Cluster de Tecnologías e Información de Michoacán*. Recuperado el 16 de 8 de 2014, de clustertim: <http://www.clustertim.com.mx/>
29. CMMI. (2015). *The Capability Maturity Model Integration (CMMI)*. Recuperado el 22 de 2 de 2015, de CMMI.com: <http://cmminstitute.com/#home>
30. Cockburn, A. (2004). *Crystal Clear: A Human-Powered Methodology for Small Teams*. USA: Pearson Education.
31. Cohen, N., & Elalouf, A. (May de 2014). How to Manage an Agile/Kanban Software Project Using EVM. (ijcit.com, Ed.) *International Journal of Computer and Information Technology*, 3(3), 533-536.

-
32. Cunha, T., Dantas, V., & Andrade, R. (2011). SLeSS: A Scrum and Lean Six Sigma Integration Approach for the Development of Software Customization for Mobile Phones. En IEEE (Ed.), *Software Engineering (SBES), 2011 25th Brazilian Symposium on*. 25, págs. 283-292. Sao Paulo: IEEE.
 33. CYTED. (2015). *CYTED*. (CYTED, Productor) Recuperado el 10 de 2 de 2015, de PROGRAMA IBEROAMERICANO DE CIENCIA Y TECNOLOGIA PARA EL DESARROLLO: <http://www.cytel.org/>
 34. de Haaff, B. (2012). *Which is the best agile software project management tool?* Recuperado el 20 de 1 de 2015, de Quora: <http://www.quora.com/Which-is-the-best-agile-software-project-management-tool>
 35. De Silva, K., Darshana, H., Silva, W., Basnayake, B., & Wickramaratne, C. (2013). Distributed Integrated Development Environment for Mobile Platforms. *The 8th International Conference on Computer Science & Education (ICCSE 2013)*, (págs. 324-329). Sri Lanka.
 36. Dyba, T., & Dingsøyr, T. (1 de 2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, 50.
 37. e-encuesta. (2014). *e-encuesta.com*. Recuperado el 15 de 1 de 2015, de Encuesta en línea. Desarrollo de aplicaciones móviles en México: <http://www.e-encuesta.com/answer?testId=HLraFEHrV6w=>
 38. Erickson, K., Dierking, R., & Thomas, M. (2014). *A Comparative Look at Top Agile Tools*. Recuperado el 5 de 1 de 2015, de Captech; Insights: <https://www.captechconsulting.com/blogs/a-comparative-look-at-top-agile-tools>
 39. Escutia Núñez, G. (Septiembre de 2010). *Cámara Nacional de la Industria Electrónica y Telecomunicaciones y Tecnologías de la Información*. Recuperado el Septiembre de 2013, de Sede Occidente: <http://www.caniet.org/sedes/occidente/QuienSomosOccidente.aspx>
 40. Estorach, V. (2014). *Las 7 métricas que importan en las apps móviles*. Recuperado el 1 de 2 de 2015, de YeePLY; Blog de desarrollo de Apps, Webs y juegos móviles: <https://www.yeePLY.com/blog/7-metricas-para-apps-moviles/>
 41. Expósito-Langa, M., Molina-Morales, F. X., & Capó-Vicedo, J. (2009). Influencia de las dimensiones de la capacidad de absorción en el desarrollo de nuevos productos. *Investigaciones Regionales*, 17, 20-59.
 42. Ferreira M., H. (2010). *Optimización de árboles de actividades (WBS) en la planeación de proyectos, utilizando simulación de proceso en una máquina paralela*. Universidad Interamericana para el Desarrollo, UNID, Maestría en Tecnologías de la Información. Morelia: UNID.
 43. Flores R., B. L., Astorga V., M. A., Rodríguez E., O. M., Ibarra E., J. E., & del Carmen A., M. (Junio de 2014). Interpretación de las normas mexicanas para la implantación de procesos de software y evaluación de la capacidad bajo un enfoque de gestión de conocimiento. *Revista de la Facultad de Ingeniería*(71), 85-100.
 44. Flores Rios, B. L., Astorga Vargas, M. A., Olguín Espinoza, J. M., & Andrade Peralta, M. d. (2008). Experiences on the Implementation of MoProSoft and Assessment of Processes under the NMX-I-059/02-NYCE-2005 Standard in a Small Software Development Enterprise. En IEEE (Ed.), *Mexican International Conference on Computer Science*. 1, págs. 323-328. Mexicali, Mx: IEEE.

45. Fonseca, E. (2014). *Mobile Camp México 2014*. Recuperado el 10 de 12 de 2014, de El estado del arte móvil en México: <http://mobilecamp.mx/>
46. Fortier, A., Rossi, G., Gordillo, S. E., & Challiol, C. (4 de 11 de 2009). Dealing with variability in context-aware mobile software. *Journal of Systems and Software*, 83(6), 915-936.
47. Gallegos, R., Grandet, C., & Pavel, R. (2014). *Los Emprendedores de TIC en México: Recomendaciones de política pública para su nacimiento, crecimiento y consolidación*. Instituto Mexicano para la Competitividad, Estudio realizado por Microsoft de México. México D.F.: IMCO.
48. Gandlgruber, B. (2010). *INSTITUCIONES, COORDINACIÓN Y EMPRESAS* (Vol. 1). (U. Cuajimalpa, Ed.) D.F., México, México: Anthropos.
49. GetApp. (2015). *Get Agile Project Management Applications*. Recuperado el 1 de 2 de 2015, de GetApp: <https://www.getapp.com/project-management-planning-software/agile-project-management/>
50. Gooogle. (2015). *Android*. Recuperado el 12 de 1 de 2015, de Sistema Android para dispositivos móviles: <https://www.android.com/>
51. Gooogle Play. (2015). *Gooogle Play Store*. Recuperado el 15 de 2 de 2015, de Gooogle Play: <https://play.google.com/store>
52. Gobierno de la Republica. (2013). *Plan Nacional de Desarrollo 2013-2018*. Recuperado el 10 de 2 de 2015, de pnd.gob.mx: <http://pnd.gob.mx/>
53. goleansixsigma. (2014). *Lean Six Sigma*. Recuperado el 20 de 1 de 2015, de goleansixsigma.com: <https://goleansixsigma.com/the-basics-of-lean-six-sigma/>
54. González, A. (2012). *canieti.org*. (CANIETI, Productor) Recuperado el 5 de 1 de 2015, de CANIETI: http://canieti.org/Comunicacion/noticias/vista/12-04-09/Hay_38_cl%C3%B4steres_mexicanos.aspx
55. Greenfield, J., Short, K., Cook, S., & Kent, S. (2004). *Software Factories; Assembling Applications with Patterns, Models, Frameworks and Tools* (1a. edición ed., Vol. 1). Incianapolis, Indiana, USA: Wiley Publishing, Inc.
56. Guerra C., L. (2013). *Conferencia CNE - Los Clusters como Impulsores del Desarrollo y la Innovación*. Grupo SUSTAVA, Planeción y Desarrollo. slideshare.net.
57. Haar, J., Leroy-Beltrán, C., & Beltrán, Ó. (2004). *Revista del comercio exterior*. Recuperado el 10 de 12 de 2014, de <http://www.revistacomercioexterior.com/>: <http://www.revistacomercioexterior.com/rce/magazines/67/3/haar0604.pdf>
58. Harlem, F., Swati, C., & Xiaofeng, W. (May de 2014). Adopting an Agile Approach for the Development of Mobile Applications. (A. Ahmed, Ed.) *International Journal of Computer Applications*, 94(17), 43-50.
59. Hernández S., R., Fernández C., C., & Baptista L., P. (2010). *Metodología de la investigación*. (J. Mares C., Ed.) México, Perú: McGRAW-HILL.
60. HighCharts. (15 de 2 de 2014). *HighCharts API*. (HighCharts.com, Productor) Recuperado el 2015, de HighCharts Make Your Data Come Alive: <http://www.highcharts.com/>

61. IMCO. (2014). *Los Emprendedores de TIC en México: Recomendaciones de política pública para su nacimiento, crecimiento y consolidación*. Instituto Mexicano para la Competitividad, Estudio realizado por Microsoft de México. México D.F.: IMCO.
62. ImpulsaTI. (2014). *IMPULSA Tecnologías de Información*. Recuperado el 15 de 1 de 2015, de impulsati.org: <http://impulsati.org/>
63. INEGI. (2014). *Indicadores de ocupación y empleo al cuarto trimestre de 2014*. Recuperado el 5 de 10 de 2014, de Instituto Nacional de Estadística y Geografía: <http://www3.inegi.org.mx/sistemas/temas/default.aspx?s=est&c=25433&t=1>
64. INEGI. (s.f.). *Instituto Nacional de Estadística y Geografía*. (INEGI) Recuperado el 16 de 8 de 2014, de <http://www.inegi.org.mx/sistemas/bie/cuadrosestadisticos/GeneraCuadro.aspx?s=est&nc=618&c=25436>
65. infoinfo. (2014). *infoinfo*. Recuperado el 5 de 10 de 2014, de Directorio de empresas México - listado de empresas México - comunidad de empresas México: <http://www.infoinfo.com.mx/>
66. infoinfo. (2015 йил 15-Ene). *Directorio de Empresas en México*. (infoinfo) Retrieved 2013 йил 15-Feb from infoinfo: <http://www.infoinfo.com.mx/>
67. Infotec. (2014). *Experiencias y desafíos en la apropiación de las TICs por las PyMEs Mexicanas*. Recuperado el 20 de 1 de 2015, de www.infotec.com.mx: http://www.infotec.com.mx/en_us/infotec/Experiencias_y_desafios_de_las_TICs
68. ISO. (2015). *ISO Standards*. (ISO, Productor) Recuperado el 10 de 1 de 2015, de www.iso.org
69. ITIL Foundation. (2014). *metrics*. (ITIL, Editor) Recuperado el 1 de 12 de 2014, de ITIL Foundation: http://itilv3.osiatis.es/proceso_mejora_continua_servicios_TI/metricas.php
70. Jensen, K., & Kristensen, L. M. (2009). *Coloured Petri Nets; Modelling and Validation of Concurrent Systems* (Vol. 1). (Springer, Ed.) Berlin, Heidelberg, Germany: Springer-Verlag.
71. Jensen, K., Christensen, S., Kristensen, L. M., & Westegaard, M. (13 de 12 de 2014). *Coloured Petri Net Tools, 4.0*. (C. Group, Productor) Recuperado el 22 de 1 de 2015, de CPN Tools 4.0: <http://cpntools.org/>
72. Jeong, Y.-J., Lee, J.-H., & Shin, J.-H. (2008). Development Process of Mobile Application SW Based on Agile Methodology. *Development of mobile application software environment technology supporting multi-platform, IEEE*, 362-366.
73. Jones, C. (2008). Development Practices for Small Software Applications. *The Journal of Defense Software Engineering*(9-13).
74. jQuery Foundation. (2006). *jQuery write less, do more*. Recuperado el 10 de 2 de 2015, de jQuery: <https://jquery.com/>
75. K. Dey, A., D. Abowd, G., & Salber, D. (2001). A conceptual Framework and a Toolkit for supporting the Rapid Prototyping of Context-Aware Applications. *Human Computer Interaction*, 16, 97-166.
76. Kan, S. H. (2003). *Metrics and Models in Software Quality Engineering* (2a. ed.). Boston, USA: Addison-Wesley.
77. Kothari, C. (2004). Methods of Data Collection. En C. R. Kothari, *Research Methodology methods and thecniques* (págs. 95-151). New Delhi: NEW AGE INTERNATIONAL (P) LIMITED, PUBLISHERS.

78. Krill, P. (2013). *Developer picks: 7 hot tools for agile development*. Recuperado el 15 de 1 de 2015, de InfoWorld: <http://www.infoworld.com/article/2606822/agile-development/125232-Developer-picks-7-hot-tools-for-agile-development.html#slide7>
79. Laporte, C. Y., Simon, A., Renault, A., & Crowder, K. V. (2008). The Developmnet of International Standards for Very Small Enterprises. *The International Council on Systems Engineering (INCOSE)*, 1-6.
80. leadinganswers.com. (2015). *Leading Answers*. Recuperado el 15 de 1 de 2015, de Leading Answers; leadership and agile project management ideas, observation and links: http://leadinganswers.typepad.com/leading_answers/resources.html
81. Letelier, P., & Penadés, M. C. (4 de 2006). Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). *Ciencia y Técnica Administrativa*, 5(26).
82. Lopez M., D. A. (2013). *Implementación de Fábricas de Software en el sector públic Colombiano*. Asociación Colombiana de Ingenieros de Sistemas. Información de cursos y conferencias, Publicaciones. Bogota, Col: ACIS.
83. Lozada Carrillo, A. (2013). Crónica de una Certificación Fracasada de MoProSoft. *Software Guru*, 1(38), 61-64.
84. Maglyas, A., Nikula, U., & Smolander, K. (2012). Lean Solutions to Software Product Management Problems. *IEEE Software*, 12, 40-46.
85. Matus R., M., Ramírez A., R., & Buenrostro M., E. (2013). *Capacidades adquiridas y por desarrollar en los cluster TI de México: Análisis comparativo centrado en el caso inteQsoft A.C.* INFOTEC, Fondo de Información y Documentación para la Industria. D.F. México: INFOTEC.
86. MexicoFirst. (2015). *MEXICO FIRST*. Recuperado el 15 de 2 de 2015, de mexico-first.org: <http://www.mexico-first.org/>
87. MexicoIT. (2014). *México IT*. Recuperado el 15 de 2 de 2105, de mexico-it.net: <http://www.mexico-it.net/>
88. Microsoft. (2015). *Windows Phone*. Recuperado el 10 de 2 de 2015, de Windos Phone: <http://www.windowsphone.com/>
89. MIT. (2013). *MIT App Inventor*. Recuperado el 15 de 1 de 2015, de App Inventor: <http://appinventor.mit.edu/explore/about-us.html>
90. Moprosoft, N. (2011). *Tecnología de la información - Ingeniería de Software - Calidad de producto*. (NYCE, Ed.) Recuperado el 2015, de <http://www.nyce.org.mx>: <http://www.nyce.org.mx/index.php/proceso-verif/moprosoft>
91. Myung H., S. (January de 2009). Process Quality Levels of ISO/IEC 15504, CMMI and K-model. *International Journal of Software Engineering and Its Applications (IJSEIA)*, 3(1), 33-42.
92. Nathan-Regis, B. N., & Nasira, G. (2013). Agile Methods and Quality _ A survey. *Computer Science & Information Technology*, 10, 167-182.
93. Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, 72-78.

94. NYCE. (2011). *NMX-I-059/02-NYCE-2011 "Tecnología de la información - Ingeniería de Software - Calidad de producto"(MoProSoft)*. (NYCE, Ed.) Recuperado el 10 de 3 de 2015, de NYCE Estándares:
<http://www.nyce.org.mx/index.php/proceso-verif/moprosoft>
95. Oberscheven, F. M. (2013). *Software Quality Assesment in a Agile Environment*. Faculty of Science. Düsseldorf, Germany: Radboud Universiteit Nijmegen.
96. OCDE. (2009). *Estudios de la OCDE de innovación regional; 15 estados mexicanos* (Vol. 1). (M. N. Milena, Trad.) D.F., México, México: OCDE.
97. Oktaba, H., García, F., Piatinni, M., Pino, F., Alquicira, C., & Ruiz, F. (2007). Software Process Improvement: The COMPETISOFT Project. (IEEE, Ed.) *IEEE Computer*, 40(10), 21-28.
98. OMG. (2015). *Software & Systems Process Engineering Metamodel Specification (SPEM) Version 2.0*. Recuperado el 20 de 1 de 2015, de omg.org: <http://www.omg.org/spec/SPEM/2.0/>
99. OpenAjax. (2012). *OpenAjax Alliance*. (O. A. Alliance, Productor) Recuperado el 15 de 1 de 2015, de OpenAjaxa Allance; standardizing Ajax Development: <http://www.openajax.org/index.php>
100. ORACLE. (2010). *JavaServer Faces Technology*. (ORACLE, Productor) Recuperado el 15 de 2 de 2015, de ORACLE Technology, Java, Java EE: <http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html>
101. ORACLE. (2015). *NetBeans IDE The Smarter and Faster Way to Code*. (ORACLE, Productor) Recuperado el 13 de 2 de 2015, de Netbeans org: <https://netbeans.org/>
102. Osorio, E. (2008). *La disciplina del aseguramiento de calidad con métodos ágiles*. Sun microsystems, Consultoría educativa Sun México. México D.F.: Sun.
103. Ovais A., M., Markkula, J., & Ovio, M. (2013). Kanban in software development: A systematic literature review. En IEEE (Ed.), *39th Euromicro Conference Series on Software Engineering and Advanced Applications*. 39, págs. 9-16. Santander, España: IEEE.
104. Pérez P., M., & González C., F. J. (2012). *Guía comparativa de metodologías ágiles*. Universidad de Valladolid, Escuela Universitaria de Informática. Universidad de Valladolid.
105. Peter. (2010). *Top Agile and Scrum Tools – Which One Is Best?* Recuperado el 1 de 1 de 2015, de agileScout: <http://agilescout.com/best-agile-scrum-tools/>
106. phpMyAdmin. (2015). *Bringing MySQL to the web*. Recuperado el 15 de 2 de 2015, de phpMyAdmin: http://www.phpmyadmin.net/home_page/index.php
107. Piatini V., M. G., & Garzas P., J. (2010). *FABRICAS DEL SOFTWARE: EXPERIENCIAS, TECNOLOGÍAS Y ORGANIZACIÓN*. 2a. edición. (Vol. 2). (Ra-MA, Ed.) Madrid, España: RA-MA EDITORIAL.
108. Piattini, M. G., & Garzas, J. (2010). *Fábricas de Software: Experiencias, tecnologías y orgnaización*. Madrid, España: RA-MA.
109. Pino, F. J., García, F., & Piatinni, M. (2008). Software process improvement in small and medium software enterprises: a systematic review. *Software Quality*, 16, 237-261.

110. PivotalTracker. (2015). *Buil Better Software faster*. Recuperado el 25 de 2 de 2015, de Pivotal Tracker:
<http://www.pivotaltracker.com>
111. PMI. (5 de Dic de 2014). *Project Management Institute*. (PMI, Productor) Recuperado el 13 de Ene de 2015, de Practice Standard for Work Breakdown Structures:
<http://marketplace.pmi.org/Pages/ProductDetail.aspx?GMProduct=00100084701>
112. PMI-ACP. (3 de Ene de 2015). *PMI Agile Certified Practitioner (PMI-ACP)*. (PMI, Productor) Recuperado el 3 de Ene de 2015, de PMI: <http://www.pmi.org/Certification/New-PMI-Agile-Certification.aspx>
113. Porter, M. E. (2009). *SER COMPETITIVO* (2009 ed.). (C. G. L., Trad.) Madrid, Madrid, España: Harvard Business Press/Deusto.
114. PrimeFaces. (2014). *PrimeFaces Ultimate JSF Framework*. (PrimeTek, Productor) Recuperado el 11 de 2 de 2015, de primefaces: <http://primefaces.org/>
115. Prosoft, SEDECO. (2012). *Prosoft 3.0*. (CANIETI, Ed.) Recuperado el 10 de 1 de 2015, de Programa para el desarrollo de la industria del Software 3.0:
http://www.prosoft.economia.gob.mx/Imagenes/ImagenesMaster/Estudios%20Prosoft/GREF_12.pdf
116. Rahimian, V., & Rasmin, R. (2008). Designing an agile methodology for mobile software development: A hybrid method engineering approach. *Research Challenges in Information Science, 2008* (págs. 337 - 342). Marrakech: IEEE.
117. Rai, A., Song, H., & Troutt, M. (18 de 6 de 1998). Software quality assurance: An analytical survey and research prioritization. *Journal of Systems and Software, 40*(1), 67-83.
118. Rally. (2015). *Rally Team Collaboration*. Recuperado el 2015, de Rally:
<https://www.rallydev.com/solutions/team-collaboration>
119. Ronen-Harel, S. (2015). *Quick compare agiel tools- evaluation guide*. Recuperado el 10 de 2 de 2015, de User Stories: <http://tracks.roojoom.com/r/28211#/trek?page=5>
120. Rueda G., A. B. (2010). *Propuesta de una guía para interpretar los procesos de MOPROSOFT de la categoría de operación usando una combinación de métodos ágiles*. Instituto Politécnico Nacional, UPIICSA. Sección de estudios de posgrado e investigación. México D.F.: IPN.
121. Salo, O., & Abrahamsson, P. (2006). An iterative improvement process for agile software development. *Software Process: Improvement and Practice, 81-100*.
122. Salvaneschi, G., Ghezzi, C., & Pradella, M. (19 de 4 de 2012). Context-oriented programming: A software engineering perspective. *Journal of Systems and Software, 86*(8), 1801-1817.
123. Saraiva, J. A., de Franca, M. S., Soares, S. C., Filho, F. J., & de Souza, R. M. (27 de 1 de 2015). Classifying metrics for assessing Object-Oriented Software Maintainability: A family of metrics' catalogs. *Journal of Systems and Software, 103*, 85-101.
124. Schaffer L., R., & Oktaba, H. (2008). *Guía para el desarrollo de aplicaciones WEB*. Postgrado UNAM, Maestría en Ciencias e Ingeniería de la Computación. México, D.F.: UNAM.

125. Scharff, C., & Verma, R. (Mayo de 2010). Scrum to Support Mobile Application Development Projects in a Just-in-time Learning Context. *ACM*, 978(1), 25-31.
126. Scrum. (2015). *Scrum.org*. Retrieved 2015 йил 12-1 from <https://www.scrum.org/>
127. ScrumDo. (2015). *Delivery Predictably*. Recuperado el 25 de 2 de 2015, de ScrumDo: <http://www.scrumdo.com>
128. Sección Amarilla. (3 de Ene de 2014). *Sección Amarilla*. Recuperado el 8 de Ago de 2013, de Sección Amarilla: <http://www.seccionamarilla.com.mx>
129. Secretaría de Economía. (mayo de 2014). *PROSOFT 2.0*. Recuperado el mayo de 2014, de PROSOFT de la Secretaría de Economía: <http://www.prosoft.economia.gob.mx/>
130. SEDECO. (2012). *ESTUDIO DE COMPETITIVIDAD DE CLUSTERS DE TECNOLOGÍAS DE INFORMACIÓN*. Dirección General de Comercio Interior y Economía Digital, Subsecretaría de industria y comercio. D.F.: SEDECO.
131. SEI. (2013). *TSP Performance and Capability Evaluation (PACE): Customer Guide*. Recuperado el 15 de 2 de 2015, de Software Engineering Institute: <http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=59407>
132. SEI. (2015). *Software Engineering Institute*. Recuperado el 20 de 2 de 2015, de SEI: <http://www.sei.cmu.edu/cmml/>
133. Select Estrategia. (2014). *Estado actual y perspectivas del capital humano en el sector TI y servicios relacionados Cuarto entregable: Versión final del estudio*. Secretaría de Economía. México D.F.: Select Estrategia S.C.
134. Shojaei, H. (2012). *Agile/Scrum Tools List and Pricing Comparison*. Recuperado el 1 de 2 de 2015, de Axosoft; company blog: <http://blog.axosoft.com/2012/04/11/agile-scrum-tools-list-pricing-comparison/>
135. SIEM. (5 de Ene de 2014). *Sistema de Información Empresarial Mexicano, 2010*. (SEDECO, Editor, S. D. ECONOMÍA, Productor, & SEDECO) Recuperado el 13 de Sep de 2013, de SIEM: <http://www.siem.gob.mx/siem/>
136. Spataru, A. C. (2010). *Agile Development Methods for Mobile Applications*. Tesis, University of Edinburgh, School of Informatics, Edinburgh.
137. Vega L., C., Rosano V., G., Ferreira M., H., & Cendejas V., H. (1 de Abr de 2013). Optimización de árboles de actividades (work breakdown structure), en la planeación de proyectos, utilizando simulación de procesos en una máquina paralela. (C. García M., Ed.) *Universita Ciencia*, 1.
138. Velasco E., M. N. (2013). *Experiencias y desafíos en la apropiación de las TICs por las PyMEs Mexicanas*. Recuperado el 20 de 1 de 2015, de infotec.com.mx: http://www.infotec.mx/en_us/infotec/Experiencias_y_desafios_de_las_TICs
139. VERSIONONE. (2015). *Enterprise Agility*. Recuperado el 22 de 2 de 2015, de VersionOne: <http://www.versionone.com>
140. VTT Electronics. (2008). *MOBILE-D*. Recuperado el 20 de 5 de 2014, de Software Thecnologies Research Programe: <http://agile.vtt.fi/mobiled.html>
141. Wells, D. (2013). *extremeprogramming*. Recuperado el 1 de 2 de 2015, de extremeprogramming.org: <http://www.extremeprogramming.org/>

142. Westergaard, M., & Verbeek, H. (. (2015). *CPN Tools. Home*. (C. Group, Productor, & Aarhus University)
Recuperado el 15 de 2 de 2015, de CPN Tools: <http://cpntools.org/start>
143. Xiaofeng, W., Kieran, C., & Oisin, C. (15 de 2 de 2012). "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. *Journal of Systems and Software*, 85, 1287-1299.
144. XP. (2015). *Extream Programming*. Recuperado el 12 de 12 de 2015, de
<http://www.extremeprogramming.org/>
145. Yang-Jae, J., Ji-Hyeon, L., & Gyu-Sang, S. (2008). Development Process of Mobile Application SW Based on Agile Methodology. En ICACT (Ed.), *International Conference on Advance Communications Technology ICACT 2014*, (pág. Korea). Phoenix Park.

ANEXO A. Diccionario de datos de la Herramienta MDSIC 2.0.

Considerando los casos de uso y los requerimientos funcionales se desarrolló una BD que permite almacenar la información referente a las diferentes fases del desarrollo de un APP siguiendo la metodología MDSIC-M, los elementos de almacenamiento son: datos generales del proyecto, especificación de requerimientos, plan de pruebas y prototipo, tareas y entregables del proyecto, , métricas de costos y riesgos, división de módulos y desarrollo, componentes refactorizados y errores, implementación, plan de pruebas y pre-entrega de la APP, lecciones aprendidas, evaluación de calidad y gráficas de avance por fase. En la tabla 1 se muestran las tablas de la BD:

Tabla 1. Tablas de MDSIC 2.0.

Base de Datos MDSCI 2.0	Tipo	Descripción
formaa_identificación	InnoDB	Definición del proyecto y sus características
formba_minuta	InnoDB	Tabla que almacena la minuta de reuniones
formbb_elementos	InnoDB	Elementos a considerar en el proyecto
formbb_prototipos	InnoDB	Defunción de prototipos
formbb_reporte	InnoDB	Fecha, tipo y participantes de un reporte
formca_divisionmodulos	InnoDB	Partes en los que se divide el desarrollo
formca_elementos	InnoDB	Descripción de cada elemento del WBS
formcb_evaluacionmodulos	InnoDB	Aprobación de participantes del módulo
formda_elementos	InnoDB	Aprobación de elemento por parte del participante
formda_pruebas	InnoDB	Definición del plan de pruebas
formdb_lecciones	InnoDB	Lecciones aprendidas por proyecto
formea_calidadrq	InnoDB	Validación de calidad por requerimiento
forme_b_calidadgen	InnoDB	Validación de calidad general del proyecto
miembros	MyISAM	Participantes del proyecto
prerequisitos	MyISAM	Prerequisitos necesarios de un proyecto
proyectos	MyISAM	Id y participantes por proyecto
requerimientos	MyISAM	Requerimientos del WBS por proyecto
roles	MyISAM	Roles de los participantes
users	MyISAM	Usuarios del sistema

El la figura 1 se muestra el diagrama entidad- relación de las tablas:

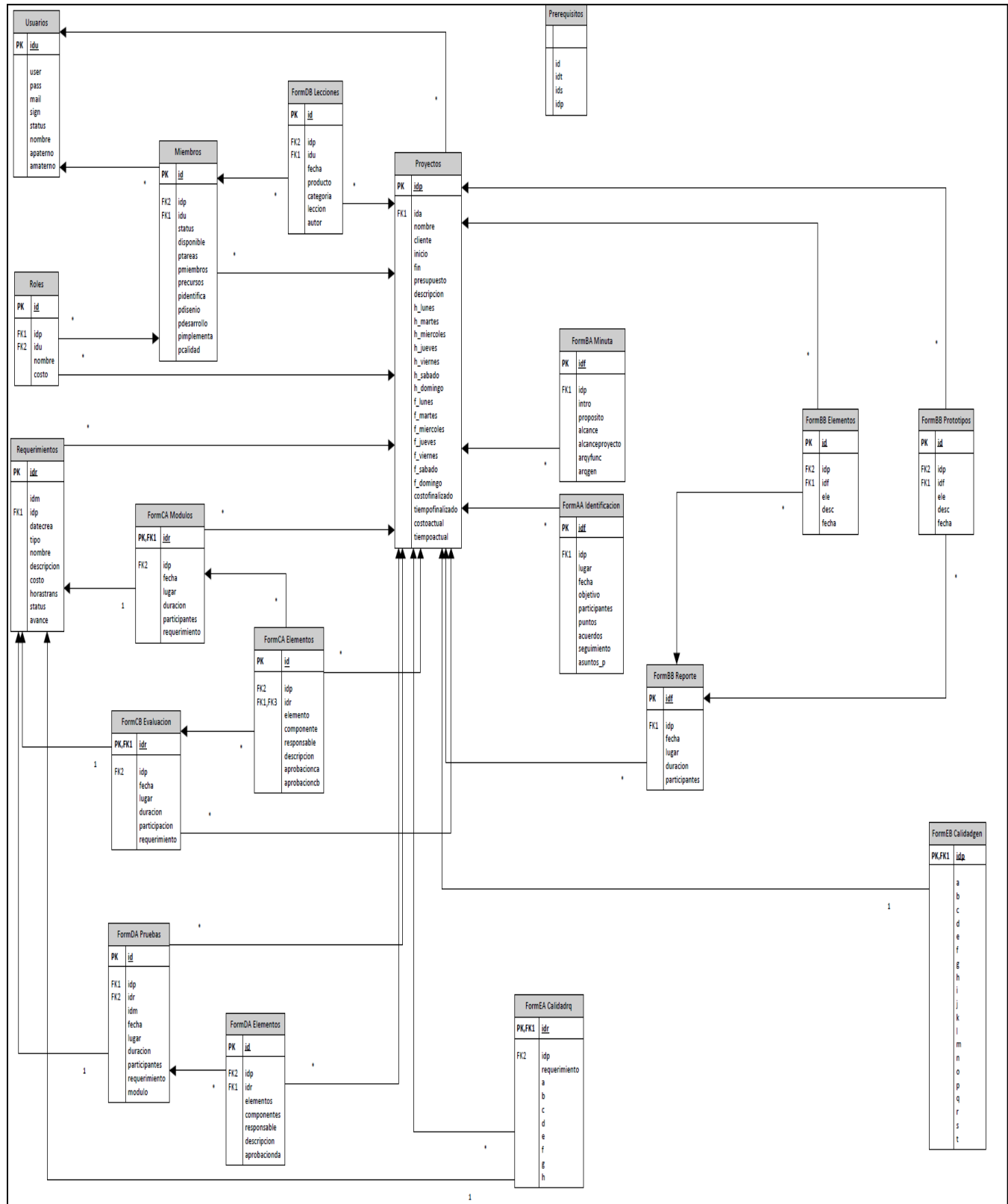


Figura 1. Diagrama entidad relación de la BD de MDSIC 2.0.

DICCIONARIO DE DATOS. A continuación se muestran las diferentes tablas y su descripción.

Tabla formaa_identificación			Tabla formba_minuta		
Campo	Tipo	Descripción	Campo	Tipo	Descripción
id	varchar(25)	Id identificación por proyecto	id	varchar(25)	Id de minuta por proyecto
idp	varchar(25)	Id del proyecto	idp	varchar(25)	Id del proyecto
lugar	text	Lugar de reunión	intro	text	Descripción de minuta
fecha	text	Fecha dd/mm/aaaa	proposito	text	Propósito minuta
objetivo	text	Motivo de la reunión	alcance	text	Alcance de la reunión
participantes	text	Participantes	alcanceproyecto	text	Alcance del proyecto
puntos	text	Puntos tratados	arqufunc	text	Arquitectura de la aplicación
acuerdos	text	Acuerdos de la reunión	arqgen	text	Arquitectura general
seguimiento	text	Puntos de seguimiento			
asuntos_p	text	Asuntos pendientes			

Tabla formbb_prototipos			Tabla formbb_reporte		
Campo	Tipo	Descripción	Campo	Tipo	Descripción
id	varchar(25)	Id de prototipo por proyecto	id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto	idp	varchar(25)	Id del proyecto
idf	varchar(25)	Id del elemento del prototipo	fecha	text	Fecha de reporte
ele	text	Nombre del elemento	lugar	text	Lugar del reporte
desc	text	Descripción del elemento	duracion	text	Horas del reporte
fecha	text	Fecha de diseño	participantes	text	Participantes del reporte

Tabla formca_divisiónmodulos			Tabla formca_elementos		
Campo	Tipo	Descripción	Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto	id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto	idp	varchar(25)	Id del proyecto
fecha	text	Fecha de la división de módulos	idf	varchar(25)	Id del elemento del WBS
lugar	text	Lugar de reunión para definir módulos	elemento	text	Nombre del elemnto
duracion	text	Duración de la reunión	componente	text	Compoenete que se desarrolla
participantes	text	Participantes	responsable	text	Responsable del desarrollo
requerimiento	text	Requerimieto que se diseño	descripcion	text	Descripción del elemento
			aprobacionca	text	Aprobación del responsable
			aprobacioncb	text	Aprobación del de calidad y administrador

Tabla formcb_evaluacionmodulos		
Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto
fecha	text	Fecha de evaluación
lugar	text	Lugar de la evaluación
duracion	text	Duración de la evaluación
participantes	text	Participantes en la evaluación
requerimiento	text	Requerimiento del WBS evaluado

Tabla formda_elementos		
Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto
idf	varchar(25)	Id del elemento del WBS
elementos	text	Elemento del WBS
componentes	text	Componentes del elemento
responsable	text	Responsable de desarrollo
descripcion	text	Descripción del elemento
aprobacionda	text	Aprobación de desarrollo

Tabla formda_pruebas		
Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto
idf	varchar(25)	Id del elemento del WBS
idm	text	Id de la prueba al elemento
fecha	text	Fecha de prueba dd/mm/aaaa
lugar	text	Lugar de las pruebas
duracion	text	Duración de la prueba
participantes	text	Participantes
requerimiento	text	Requerimiento al que pertenece
modulo	text	Módulo al que pertenece

Tabla formdb_lecciones		
Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto
idu	varchar(25)	Id de lección aprendida
fecha	text	Fecha de la lección
producto	text	Producto que se generó
categoria	text	Categoría de producto
leccion	text	Descripción de la lección
autor	text	Nombre del autor

Tabla formea_calidarq		
Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto
requerimiento	text	Nombre del requerimiento
a	text	Evaluación calidad fase 1
b	text	Evaluación calidad fase 2
c	text	Evaluación calidad fase 3
d	text	Evaluación calidad fase 4
e	text	Evaluación calidad fase 5
f	text	Evaluación calidad fase 6
g	text	Evaluación calidad fase 7
h	text	Evaluación calidad fase 8
i	text	Evaluación calidad fase 9

Tabla miembros		
Campo	Tipo	Descripción
id	varchar(25)	Id de miembros por proyecto
idp	varchar(25)	Id del proyecto
idu	varchar(25)	Id de usuario
status	text	Estaus del equipo
disponible	text	Habilitado/Deshabilitado
ptareas	text	Tareas del equipo
pmiembros	text	Número de miembros
precursos	text	Recursos necesarios
pidentifica	text	Identificación del administrador
pdisenio	text	Identificación del diseño
pdesarrollo	text	Identificación del desarrollador
pimplementa	text	Identificación de que implementa
pcalidad	text	Identificación del de calidad

Tabla formea_calidadgen		
Campo	Tipo	Descripción
id	varchar(25)	Id Proyecto
a	text	Calidad elemento a
b	text	Calidad elemento b
c	text	Calidad elemento c
d	text	Calidad elemento d
e	text	Calidad elemento e
f	text	Calidad elemento f
g	text	Calidad elemento g
h	text	Calidad elemento h
i	text	Calidad elemento i
j	text	Calidad elemento j
k	text	Calidad elemento k
l	text	Calidad elemento l
m	text	Calidad elemento m
n	text	Calidad elemento n
o	text	Calidad elemento o
p	text	Calidad elemento p
q	text	Calidad elemento q
r	text	Calidad elemento r
s	text	Calidad elemento s
t	text	Calidad elemento t

Tabla proyectos		
Campo	Tipo	Descripción
id	varchar(25)	Id proyecto
ida	varchar(25)	Id administrador
nombre	text	Nombre proyecto
cliente	text	Nombre clientes
inicio	text	Fecha inicio dd/mm/aaaa
fin	text	Fecha fin dd/mm/aaaa
presupuesto	text	Presupuesto inicial
descripcion	text	Descripción proyecto
h_lunes	text	Horas hábiles lunes
h_martes	text	Horas hábiles martes
h_miercoles	text	Horas hábiles miércoles
h_jueves	text	Horas hábiles jueves
h_viernes	text	Horas hábiles viernes
h_sabado	text	Horas hábiles sábado
h_domingo	text	Horas hábiles domingo
f_lunes	text	Horas fetivas lunes
f_martes	text	Horas fetivas martes
f_miercoles	text	Horas fetivas miércoles
f_jueves	text	Horas fetivas jueves
f_viernes	text	Horas fetivas viernes
f_sabado	text	Horas fetivas sábado
f_domingo	text	Horas fetivas domingo
costofinalizado	text	Costo final con riesgos
tiempofinalizado	text	Fecha fin real dd/mm/aaaa
costoactual	text	Costo actual
tiempoactual	text	Tiempo transcurrido

Tabla prerequisites		
Campo	Tipo	Descripción
id	varchar(25)	Id prerequisite
idt	varchar(25)	Id componente proyecto
ids	varchar(25)	Id requerimientos
idp	varchar(25)	Id proyecto

Tabla Roles		
Campo	Tipo	Descripción
id	varchar(25)	Id de reporte por proyecto
idp	varchar(25)	Id del proyecto
idu	varchar(25)	Id de lección aprendida
nombre	text	Nombre del rol
costo	text	Costo por hora del rol

Tabla requerimientos		
Campo	Tipo	Descripción
id	varchar(25)	Id requerimiento del WBS
idm	varchar(25)	Id administrador
idp	varchar(25)	Id proyecto
datecrea	text	Fecha de creación
tipo	text	Requerimiento/Tarea/Entregable
nombre	text	Nombre requerimiento
descripcion	text	Descripción
costo	text	Costos extraordinario
horastrans	text	Horas que lleva en ejecución
status	text	Estaus de la tarea, ejecución/pausa
avance	text	Porcentaje de vance QA

Tabla users		
Campo	Tipo	Descripción
idu	varchar(25)	Id usuario
user	text	Nombre de usuario
pass	text	Calve encriptada
mail	text	Correo electrónico
sign	text	Firma QR
status	int(1)	Estaus del usuario activo/inactivo
nombre	text	Nombre
apaterno	text	Apellido P
amaterno	text	Apellido M

ANEXO B. Descripción de procesos y Subprocesos de MDSIC-M.

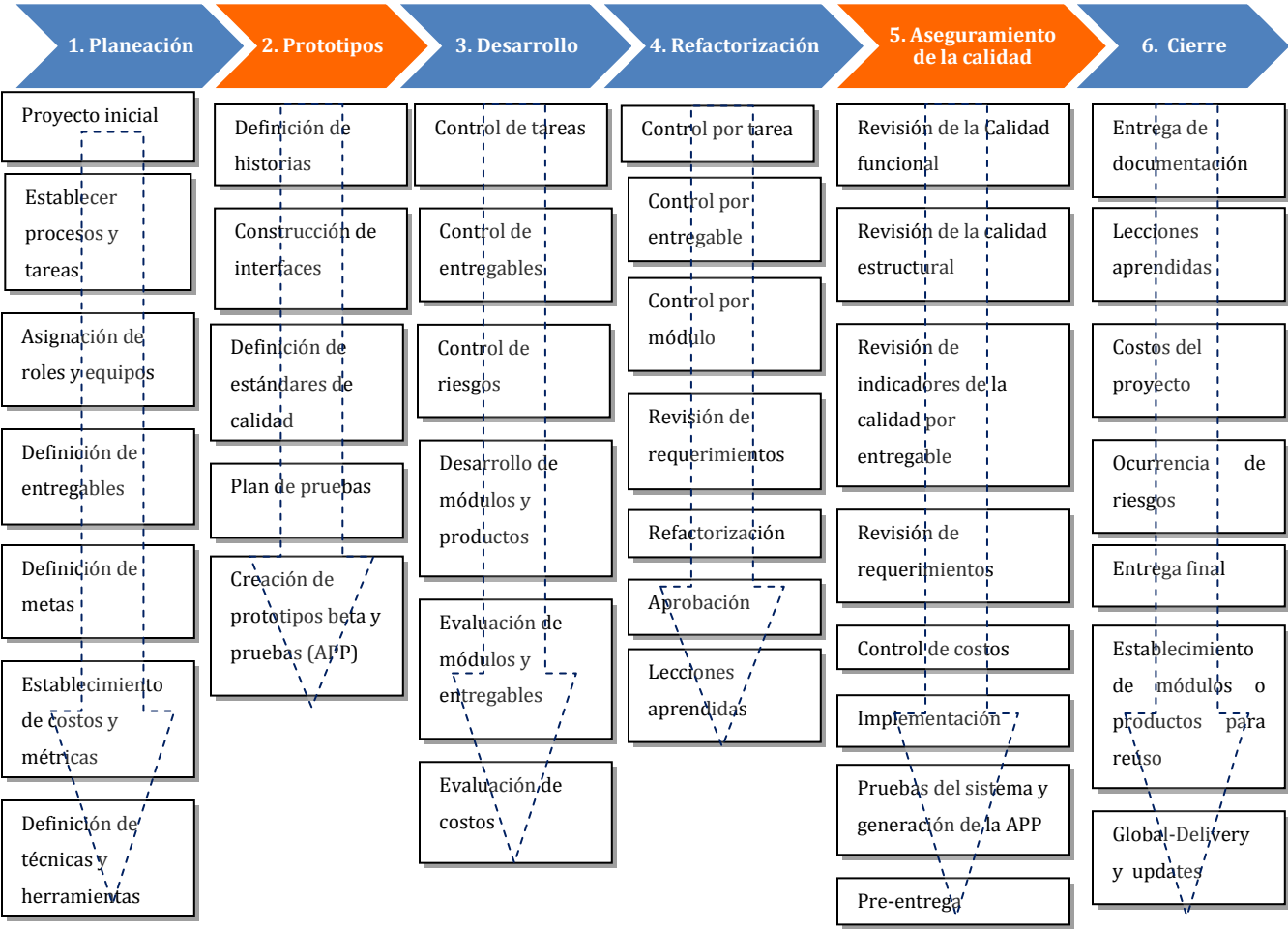


Figura 1. Metodología MDSIC-M procesos y subprocesos.

Tabla 1. Subprocesos y Roles

Proceso	Subprocesos en MDSIC-M	Roles
Planear		Administrador Cliente Diseño Calidad Desarrollo
Prototipado		Diseño Desarrollo Calidad Administrador Cliente Usuario
Desarrollar Aplicación		Desarrollo Calidad Administrador Cliente
Refactorizar		Desarrollo Calidad Administrador
Asegurar la Calidad		Calidad Administrador Cliente Diseñador
Cerrar el proyecto		Administrador Cliente Desarrollo Calidad

Fase de planeación.

Los subprocesos son:

1. Proyecto inicial. Se establecen objetivos, requerimientos y los participantes.
2. Establecer procesos y tareas. Se construye el WBS se organiza por entregable y tareas.
3. Asignación de roles y equipos de trabajo.
4. Definición de entregables y productos. Establecer los entregables (requerimientos) del proyecto, establecer hitos.
5. Definición de metas. Describir los logros y productos finales del proyecto.
6. Establecimiento de costos y métricas.
7. Definición de técnicas y herramientas. De acuerdo a los productos y entregables definidos establecer las herramientas o técnicas para el desarrollo de la APP.

Fase de prototipos.

Los subprocesos son:

1. Definición de historias. De acuerdo a los requerimientos iniciar los bocetos, diagramas o guiones de las interfaces.
2. Construcción de interfaces. Utilizando las herramientas construir las interfaces.
3. Definición de estándares de calidad. Establecer la definición de conceptos y productos, requerimientos de desarrollo, lineamientos para los procesos de implementación y los lineamientos para la evaluación.
4. Plan de pruebas. Definir las pruebas.
5. Creación de prototipos beta y pruebas.

Fase de desarrollo.

Los subprocesos son:

1. Control de tareas. Ejecución y aseguramiento de los recursos de cada una de las tareas definidas en el WBS.
2. Control de entregables. Censar continuamente el avance de los entregables, establecer fecha de inicio y fin de cada uno.
3. Control de riesgos. Contabilizar y agregar los costos para todos los riesgos, establecer nuevos tiempos y costos del proyecto.
4. Desarrollo de módulos y productos.
5. Evaluación de módulos y entregables. Revisar los KPI para cada uno de los artefactos, módulos y entregables establecidos.
6. Evaluación de costos. Establecer los costos incurridos por el desarrollo de módulos.

Fase de refactorización.

Los subprocesos son:

1. Control por tarea. En caso de detectar errores es necesario refactorizar el módulo.
2. Control por entregables. En caso de error enviar a refactorizar.
3. Control por módulo. En caso de error enviar a refactorizar.
4. Revisión de requerimientos. Censar cada uno de los entregables, productos, módulos acordes a los requerimientos establecidos, en caso de detectar errores enviar a refactorizar.

5. Refactorización. Para cada uno de los productos, tareas, módulos o entregables deben reprogramarse para entrar en desarrollo, se contabilizan y se calcula los nuevos costos.
6. Aprobación. Cada elemento que es refactorizado se revisa por el equipo de QA y si esta correcto es aprobado.
7. Lecciones aprendidas. Guardar la información de cada uno de los componentes refactorizadas y establecer los motivos de errores.

Fase de Aseguramiento de la calidad.



Los subprocesos son:

1. Revisión de la calidad funcional (QAf). Se revisan los requerimientos y el estado de la aplicación, se censa si es fácil de aprender, de usar y es confiable.
2. Revisión de la calidad estructural (QAs). Se revisa para el desarrollo de la aplicación el código desarrollado.
3. Revisión de los indicadores de la calidad por entregable. Se revisa cada uno de los KPI y se revisa la calidad por proceso (QAp).
4. Revisión de requerimientos. Se censa la calidad de los entregables.
5. Control de costos. Se verifica el costo actual del proyecto.
6. Implementación. Se establece el plan para implementar la APP para pruebas.
7. Pruebas del sistema. Una vez generada la App se aplica el plan de pruebas.
8. Pre-entrega, se genera la APP revisada y se coloca en una tienda para validar su calidad.

Fase de cierre.



Los subprocesos son:

1. Entrega de la documentación. Se generan los manuales.
2. Lecciones aprendidas. Se sintetiza todos los elementos del desarrollo de una APP.
3. Costos del proyecto. Se revisa el costo proyectado contra el costo real de la aplicación.
4. Ocurrencia de riesgos. Se contabilizan los riesgos en lo que se incurrió.
5. Entrega final. Se libera la APP en la tienda en línea y se documenta su aceptación.
6. Establecimiento de módulos o productos para reuso.

ANEXO C. Manual de usuario y técnico de MDSIC2.0.

MANUAL DE USUARIO.

Las actividades que debe realizar el usuario de la herramienta se muestra en la figura 3. En la figura 1 y 2 se describen las actividades del administrador y el seguimiento por proyecto y procesos descritos en el anexo B.

Creación del WBS para establecer un plan del proyecto.

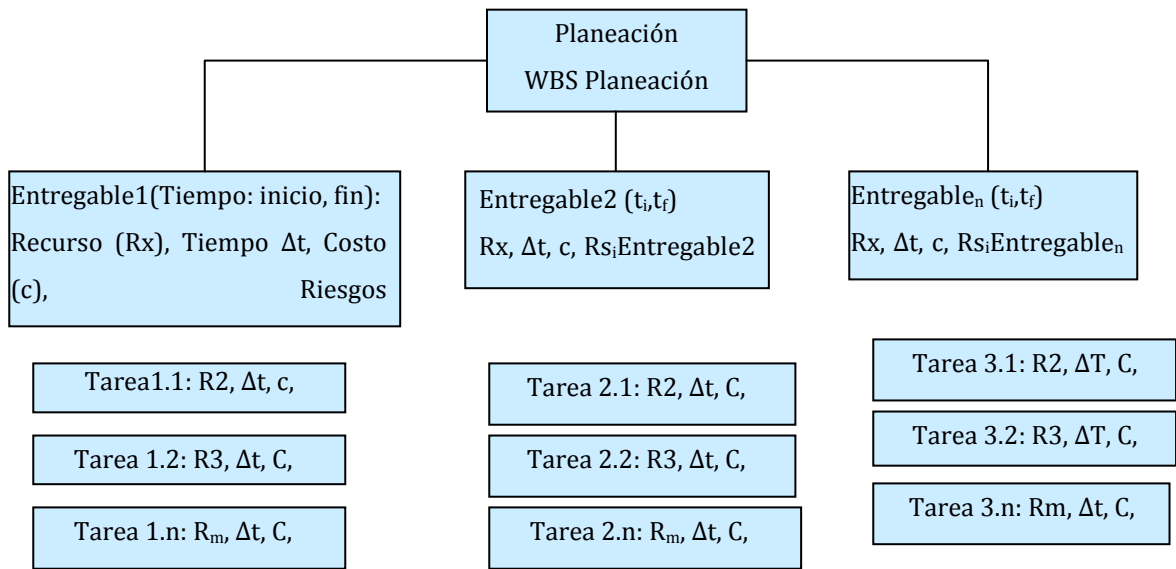


Figura 1. Guía de la estructura del árbol de actividades de un proyecto para su definición.

Fuente: Ferreira M. (2010).

Después de la planeación se inicia el seguimiento del proyecto por procesos y sus procesos, las tareas pueden ejecutarse en forma simultánea y no es necesario un orden estricto, ver tabla 1:

Tabla 1. Tareas para ejecutar los procesos, KPI a tomar en cuenta.

Proceso	KPI	Descripción
1. Planear	Tiempo	Representa el tiempo que se tarda en realizar la planeación de un proyecto, debe establecerse con claridad
	Costo	Representa el costo de desarrollo de la aplicación, debe tomar en cuenta la métricas de la APP para el cálculo del ROI
	Riesgo	Representa el porcentaje de ocurrencia de la falta de alguno de los elementos de desarrollo de la APP. El costo final deberá incluir el costo de cada uno de los riesgos ocurridos durante el proyecto
	Métricas	Deben tomarse en cuenta la métricas que definirán el mercado de la APP; usuarios, uso, adquisición, retención, etc.
2. Elaborar Prototipos	Tiempo	Representa el tiempo en que se desarrollarán prototipos para que representen los requerimientos del usuario; debe establecerse claramente para no incurrir en costos extras
	Desempeño	Representa el grado de satisfacción del cliente al generar un versión de un prototipo, el prototipo se apegue a los requerimientos de los clientes y muestra un GUI esperada
	Pruebas	Representa el número de pruebas realizadas a las Pantallas (GUI) de la APP por parte del cliente
3. Controlar el desarrollo	Riesgos	Representa el número de riesgos que ocurrieron durante el desarrollo de un producto, artefacto, módulo o entregable. Se establecen por tarea y se contabilizan para el costo final del proyecto
	Costo	Representa el costo extraordinario del desarrollo que difiere del costo planeado, esto se debe a la refactorización
	Tiempo	Representa el tiempo de desarrollo de cada una de las tareas que generan un entregable, artefacto, producto o módulo
	Equipos	Representa el personal (con diferentes roles) que intervienen en el desarrollo de una artefactos, productos, módulos o entregables
	Roles	Representa los elementos que participan en el desarrollo de un artefacto, producto, módulo o entregable. Estos deben firmar para validar el desarrollo
4. Refactorizar componentes, artefactos	Costo	Representa el costo extraordinario por refactorización de algún componente
	Riesgo	Representa el número de riesgos que ocurrieron durante la

		refactorización
	Tiempo	Representa el tiempo de refactorización de un componente con errores
	Aprendizaje	Describe el motivo de la refactorización de un componente y ayuda al aprendizaje para evitar errores
5. Asegurar la calidad	Madurez	Representa el grado de madurez de cada uno de los procesos que puede ser (NYCE, 2011), (Myung H., 2009); 0: Incompleto (no se logran los objetivos), 1: Realizado (desarrollo con poco control) 2: Gestionado (se logran tiempos y se cumplen requisitos) 3: Establecido (documentados y logran estándares) 4: Predecible (mediciones y bien planeado) 5: Optimizado (mejoras y mediciones)
	Revisión	Representa el número de veces que censó la calidad de un artefacto, producto, módulo o entregable
	Costo	Representa el costo extraordinario para medir la calidad de un artefacto, producto, módulo o entregable
	Tiempo	Representa el tiempo de realización de las tareas comparándolo con la planeación
6. Almacenar lecciones aprendidas y cierre del proyecto	Reuso	Representa el número de componentes que son factibles de reuso en proyectos subsecuentes
	Costo	Establece la diferencia entre planeado y el costo final del proyecto
	Riesgo	Establece los riesgos que ocurrieron en el desarrollo del productos
	Tiempo	Establece el tiempo de finalización el proyecto
	Aprendizaje	Almacenamiento de la métricas del proyecto, almacena el número de: requerimientos, tareas, componentes, prototipos, entregables, costos, métricas de la APP, equipos, roles, elementos de QA, componentes refactorizadas, etc.

En la figura 2 se muestra las actividades por rol que debe desarrollar el usuario al acceder al sistema, los roles son:

- Administrador. Se encarga de elaborar el plan, seguirlo, ejecutar tareas, guía el desarrollo, la implementación y ejecutar el cierre del proyecto.
- Cliente. Establece requerimientos, valida el prototipo, participa en el desarrollo como espectador y ayuda a censar la calidad.

- Programador/Diseñador. Diseña o programa las interfaces del prototipo y de la aplicación siguiendo el calendario definido en el plan.
- Calidad. Revisa que los requerimientos, el prototipo, los componentes de desarrollo y los elementos refactorizados cumplan de acuerdo al diseño inicial. También es el encargado de definir la madurez de los procesos.

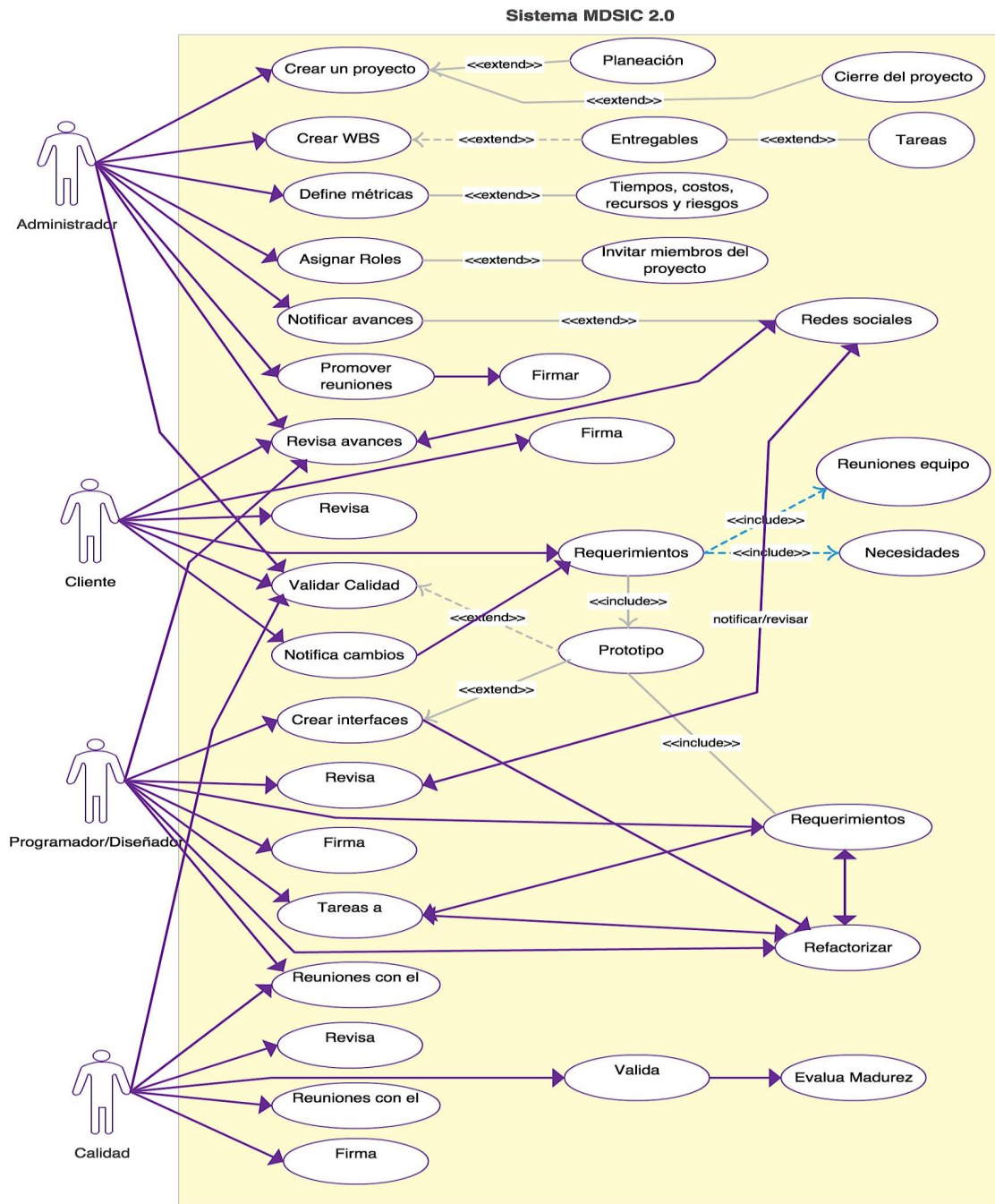


Figura 2. Actividades de los usuarios de la herramienta MDSCI 2.0.

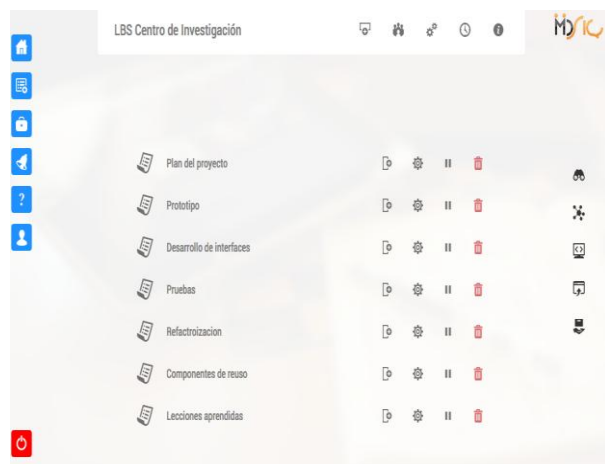
Para administrar y ejecutar un proyecto debe seguirse el orden de los subprocesos de acuerdo a las interfaces que se muestran a continuación, Ver figura 3 A), B), C) y D).



A) Acceso al sistema MDSIC 2.0.



B) Pantalla de inicio.



C) Edición de tareas y entregables en WBS.



D) Edición de un formulario para la fase de identificación.

Figura 3. A), B), C) y D). Actividades en el sistema MDSIC 2.0.

Como actividad final el sistema cuenta con un módulo de reportes que guía al administrador a través del avance del proyecto y genera la documentación técnica requerida para crear la historia del proyecto y futuras revisiones de calidad.

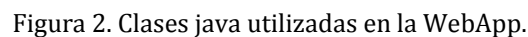
MANUAL TÉCNICO.

La herramienta fue diseñada con varios framework y que se describen brevemente en la tabla1.

Tabla 1. Frameworks utilizados para la WebApp MDSIC 2.0

Herramienta	Descripción
<i>HighCharts</i>	Para el diseño y construcción de gráficas automatizadas, utiliza JavaScript y JQuery en un diseño MVC (mysql, JQuery, JavaScript). En el servidor se reciben llamadas de Ajax (OpenAjax, 2012) para generar la vista de la gráfica.
JSF <i>JavaServer Faces</i>	Para la aplicación en Java que simplifica el desarrollo de interfaces. JSF usa <i>JavaServer Pages</i> (JSP) como la tecnología que permite hacer el despliegue de las páginas (Vista), pero también se puede acomodar a otras tecnologías como XUL (<i>XML based Use interface Language</i>).
jQuery	Simplifica la forma de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con Ajax en páginas web.
Netbeans	IDE que utiliza el lenguaje de programación Java. Tiene varios módulos para extenderlo y mejorar el desarrollo de un API (java EE, web, desktop, Mobile, etc.).
PHPMYAdmin/MySQL	Apoya a la administración de la BD elaborada en MySQL.
PrimeFaces	Componentes para JSF de código abierto, que cuenta con un conjunto enriquecido de objetos que facilitan la creación de las aplicaciones web.
Tomcat/Linux	Apache Tomcat es un contenedor de <i>servlets</i> que implementan las especificaciones de las interfaces y conexión con mysql.

En las figuras 1, 2 y 3 se muestra el desarrollo de las interfaces en el IDE así como las clases java, paquetes y plug-in utilizados en el desarrollo de la aplicación.



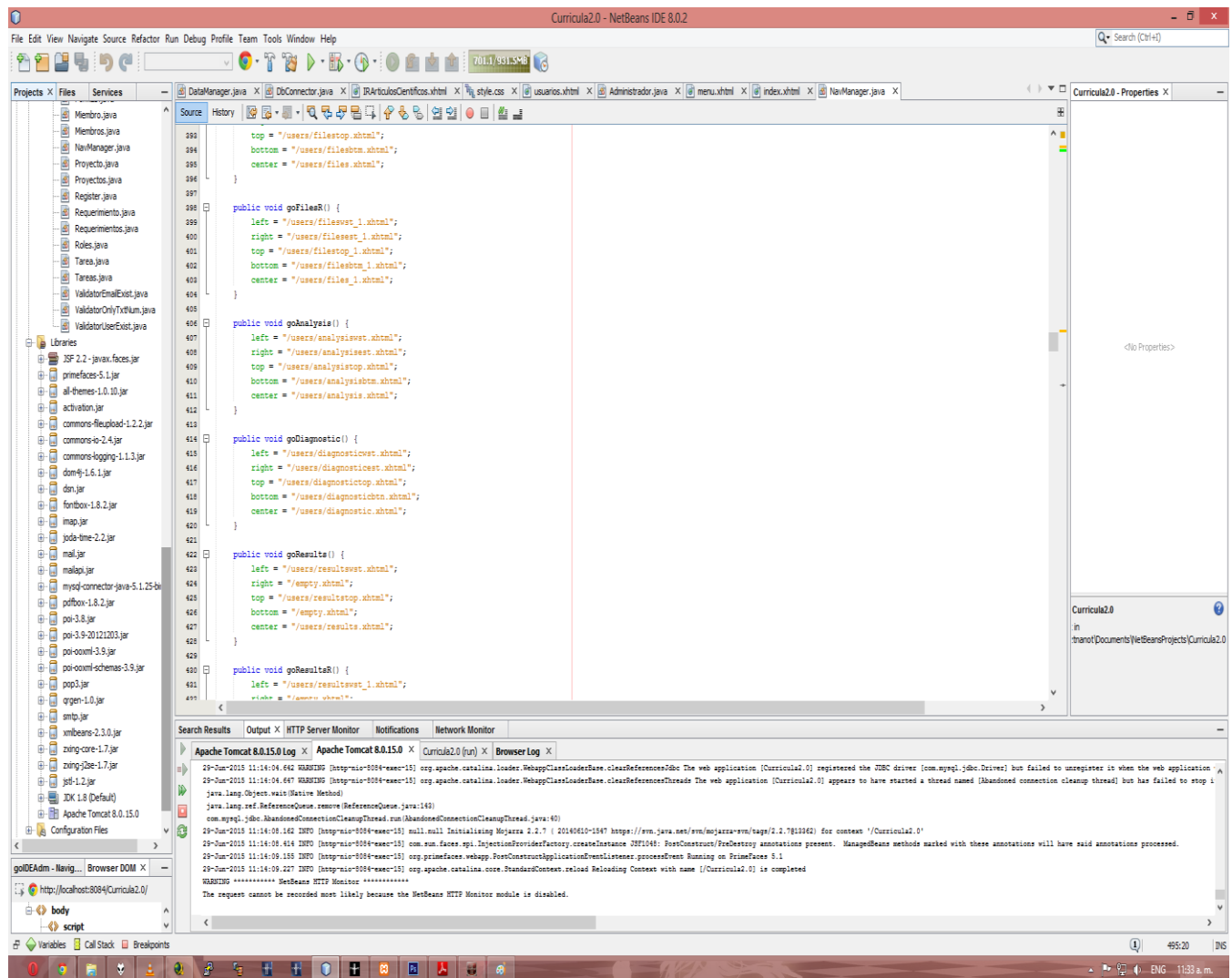


Figura 3. Plug-in .jar agregados para el correcto funcionamiento de la herramienta.

ANEXO D. Código ejemplo del scheduler.

La codificación del planificador se basa en el diseño elaborado en CPNTools y se observa en la figura 1.

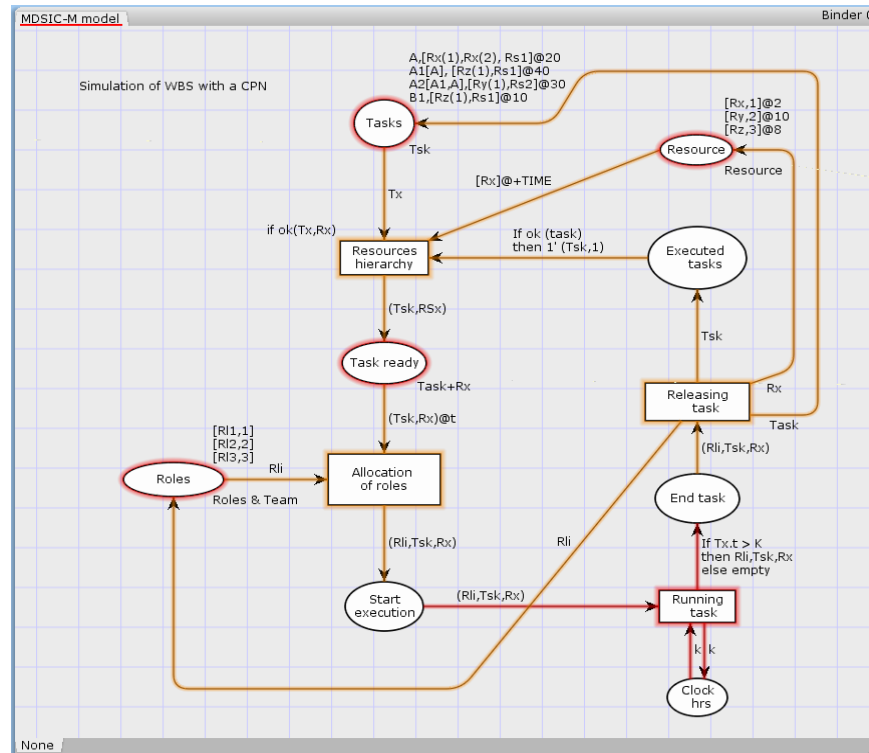


Figura 1. Diseño del planificador de tareas.

En la figura 2 se muestra la construcción de las tareas del planificador en la aplicación WebAPP que es parte esencial para la ejecución y seguimiento del proyecto.

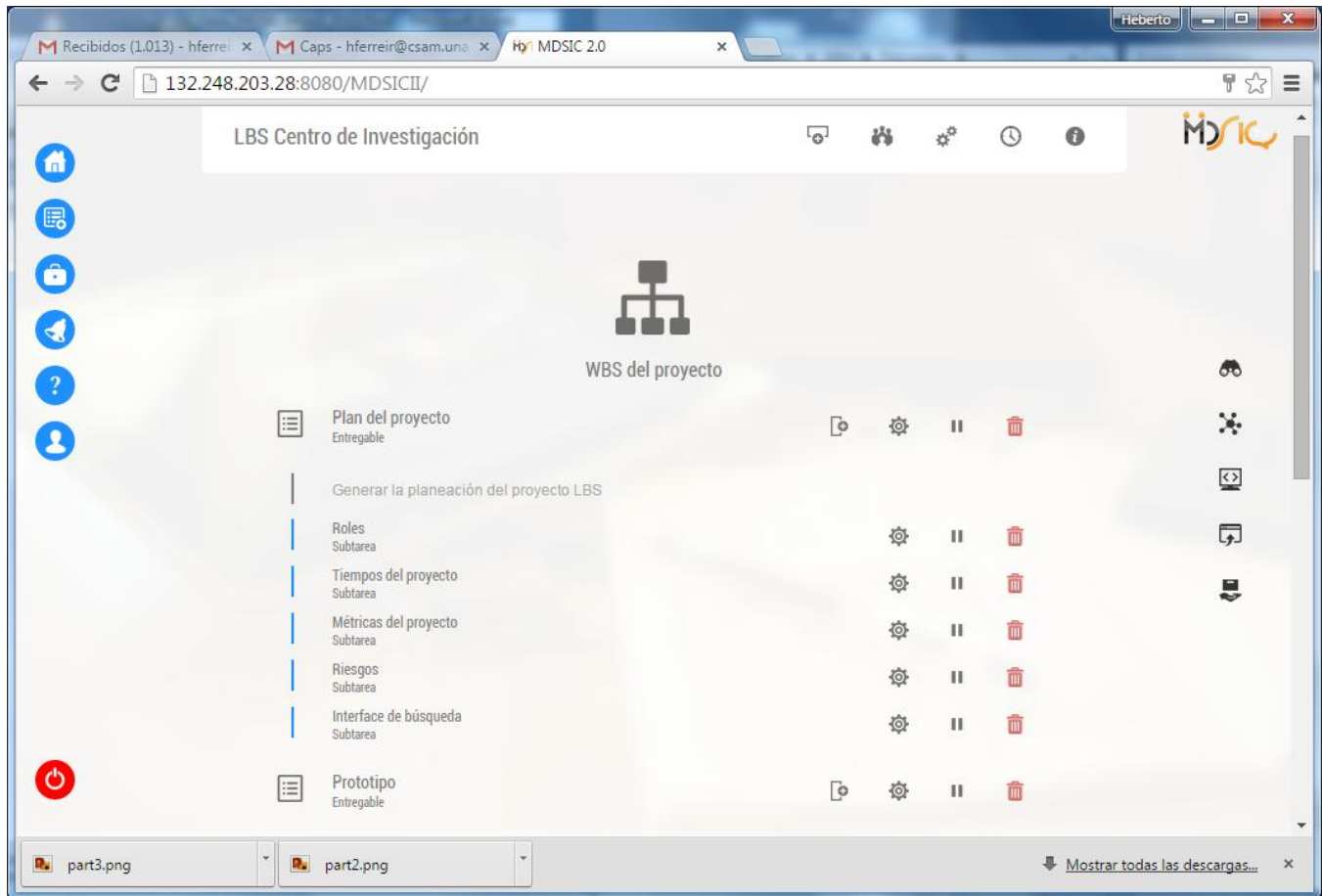


Figura 2. Ejemplo de la programación del planificador en MDSIC 2.0

En la figura 3 se muestra el diseño de las clases implementadas en el proyecto, donde se observa la interacción de los objetos y la ejecución de tareas en el planificador. En el manual técnico se muestra la codificación de cada una de las clases, los plug-in y los paquetes que interactúan en la herramienta desarrollada.

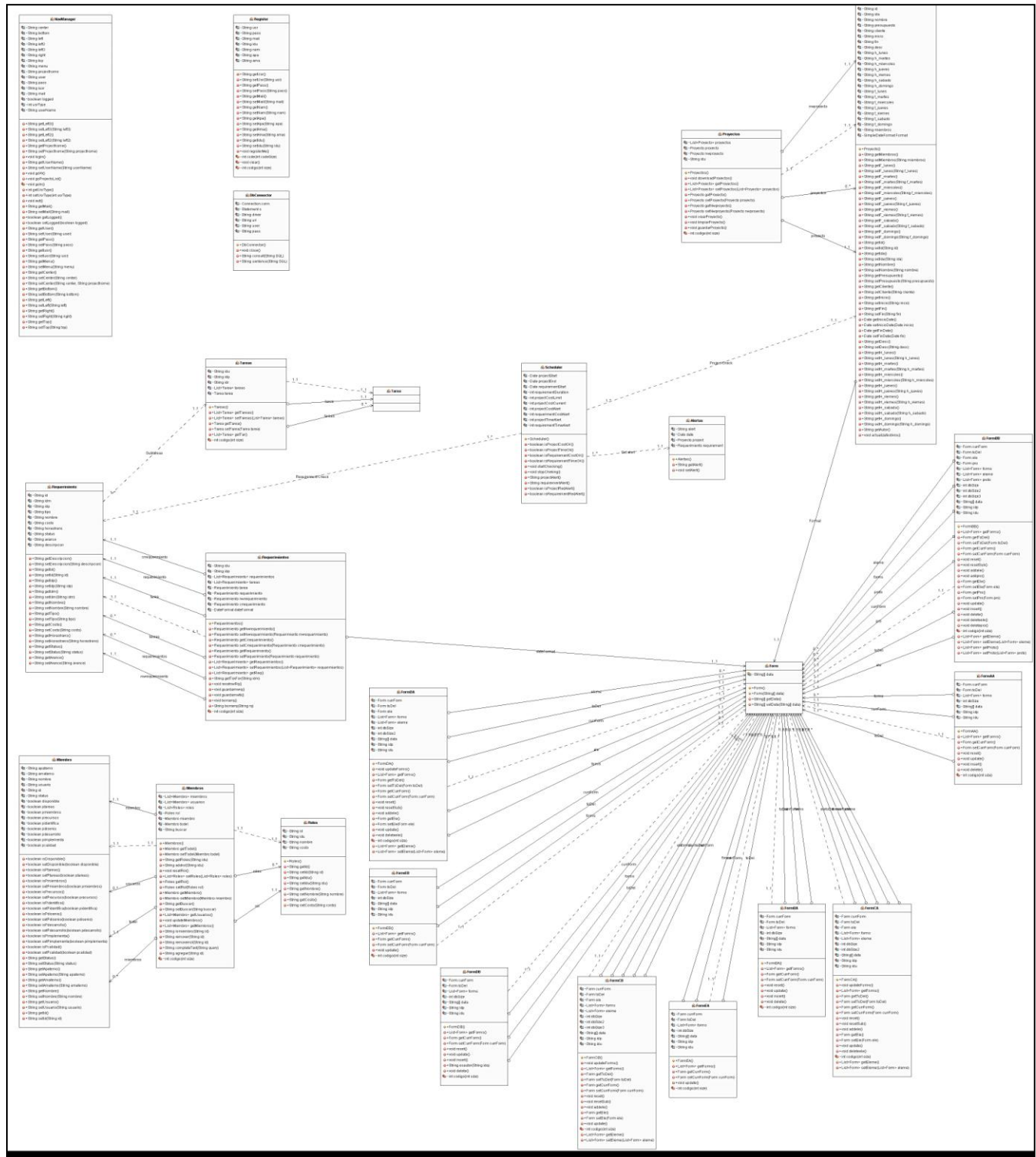


Figura3. Diseño de objetos y su interacción en MDSIC 2.0.