*A publicarse en las Memorias del Coloquio de Investigación Multidisciplinaria CIM 2012. Instituto Tecnológico de Orizaba. Octubre 2012.* 

# DEPYM: Modelo de Mejora de Procesos de Software con enfoque de Desarrollo Dirigido por Pruebas

*A.Centeno Téllez [1] , P. Gómez Gil [2],*
*[1] Universidad Veracruzana, USBI Campus Ixtac, Ixtaczoquitlán Veracruz México,*
*[2] Coordinación de Computación, Instituto Nacional de Astrofísica, Óptica y Electrónica.*
*Tonantzintla, Puebla México*
**acenteno@uv.mx*

**Área de participación**: Sistemas Computacionales

**Resumen**

El presente trabajo describe un modelo Ágil para la mejora de procesos de software llamada DEPYM para aplicarse en el proceso de Desarrollo y Mantenimiento de software y está basado en el desarrollo dirigido por pruebas. El área de aplicación del presente modelo son empresas pequeñas y medianas con equipos de desarrollo menores de 10 integrantes y en proyectos de no mayor 18 meses de duración.

El artículo incluye las bases teóricas sobre las que se basa el proyecto así como los resultados del experimento de validación. Los resultados muestran que es posible generar productos de calidad en tiempos y costos presupuestados con el modelo y mediante los procesos definidos en este trabajo. El experimento donde se aplicó el modelo DEPYM fueron 3 proyectos reales logrando resultados aceptables en cuanto a las métricas definidas por el instrumento de evaluación como velocidad de entrega, satisfacción del cliente y densidad de defectos.

***Palabras clave**: DEPYM, AGIL, TDD*

**Abstract**

*This paper describes an Agil method to software process improvement named DEPYM to be applied in the development software process and is oriented to Test driven development approach. This model should be used in small and medium enterprises with teams not greater than 10 engineers and time budget not greater than 18 months.*

*This works explain technical concepts related to DEPYM project so the obtained results from the test experiment. The results shows is possible create hight quality product in time and cost budgets using this model. The studies case was 3 real life projects in diferent contexts getting good results using Agil metrics defined like release velocity, customer acceptance and defect density.*

## Introducción

Los problemas presentes en las empresas de la construcción de software mexicanas están íntimamente relacionados con la falta de procesos maduros y repetibles usados en la elaboración de sus productos. Esta falta de procesos institucionalizados se refleja en la mala calidad de los productos, entregas fuera de calendario y presupuestos rebasados.

Las Metodologías Ágiles son un ejemplo de prácticas específicas para desarrollar software. La programación extrema (XP) es una disciplina de desarrollo de software basada en la simplicidad, comunicación, retroalimentación y entregas frecuentes. XP utiliza prácticas sencillas a ser utilizadas por el equipo de desarrollo y con suficiente retroalimentación para motivar al equipo a conocer el estado real del proyecto [1]. El desarrollo dirigido por pruebas (*Test Driven Development*, TDD) es una de las 12 prácticas claves de la Programación Extrema. En TDD los desarrolladores de software "prueban primero, después codifican," enfocando-se inicialmente en la verificación y validación de los requerimientos de software, mediante la construcción de pruebas unitarias automatizadas. En este paradigma, el diseño evoluciona como un nuevo código que es escrito para satisfacer la pruebas que fallaron [2]. La figura 1 muestra el modelo de procesos del desarrollo dirigido por pruebas.

## Trabajos Relacionados

En un grupo de desarrollo de software de IBM *Retail Store Solutions*, se construyó un software de sistemas usando el enfoque de *Test Driven Development* (TDD) [3]. El equipo estaba compuesto por 9 ingenieros de tiempo completo, cinco de los ingenieros estaban localizados en Raleigh NC, Estados Unidos, (incluyendo al líder del equipo) y cuatro estaban localizados en Guadalajara, México. Adicionalmente se usaron

recursos humanos de tiempo parcial en el equipo para la administración y desarrollo del proyecto. Aunque el equipo de desarrollo tenía amplia experiencia con la especificación *JavaPOS* 2 usada en sus proyectos y con dispositivos *POS (hardware* para punto de venta*)*, se notó que en cada revisión de entregables la densidad de defectos después de las pruebas de verificación funcional (FVT) no se había reducido como se esperaba. Como resultado de esto, el equipo de desarrollo estuvo abierto a nuevos enfoques de desarrollo como el TDD. Usando TDD, se logró reducir la densidad de defectos en alrededor del 50% comparado a un sistema similar construido usando un enfoque de pruebas unitarias a la medida. El proyecto fue completado a tiempo y con mínimo impacto en la productividad. Adicionalmente, el conjunto de casos de prueba automatizados creados vía TDD fue usado como un activo reusable para mejorar la calidad durante el tiempo de vida del sistema.

Microsoft realizó una investigación acerca de la utilidad del TDD desde el punto de vista de la calidad del software y de la productividad, en términos del tiempo de desarrollo. Este caso de estudio se desarrolló con profesionales del software en equipos de desarrollo de *Windows* y *MSN* con diferentes objetivos organizacionales, plataformas y ambientes[4]

En Microsoft, al igual que en IBM, se construyó un activo de conocimientos empíricos sobre la eficacia de TDD para ser replicado dentro de la compañía. Además dicha investigación tiene la intención de realizar un análisis costo-beneficio sobre la utilidad de TDD. Esto involucrará un análisis del retorno de la inversión para determinar la desventaja entre el incremento en el tiempo de desarrollo y el resultado en las mejoras en la calidad del software. Se espera con estas medidas ayudar a los administradores de proyectos a tomar decisiones importantes acerca de la utilidad de la implantación de TDD en sus organizaciones

El presente trabajo analiza a fondo el desarrollo dirigido por pruebas y la mejora de procesos de software, se investiga su aplicación y se propone una modelo formal de desarrollo de software que se adapta en el diseño ágil y la metodología de desarrollo por Pruebas. Esta investigación se enfoca en la Gestión del proyecto y el proceso de Desarrollo de Software con enfoque Ágil, a fin de enfrentar y proponer soluciones a la problemática de implementación de estos procesos.

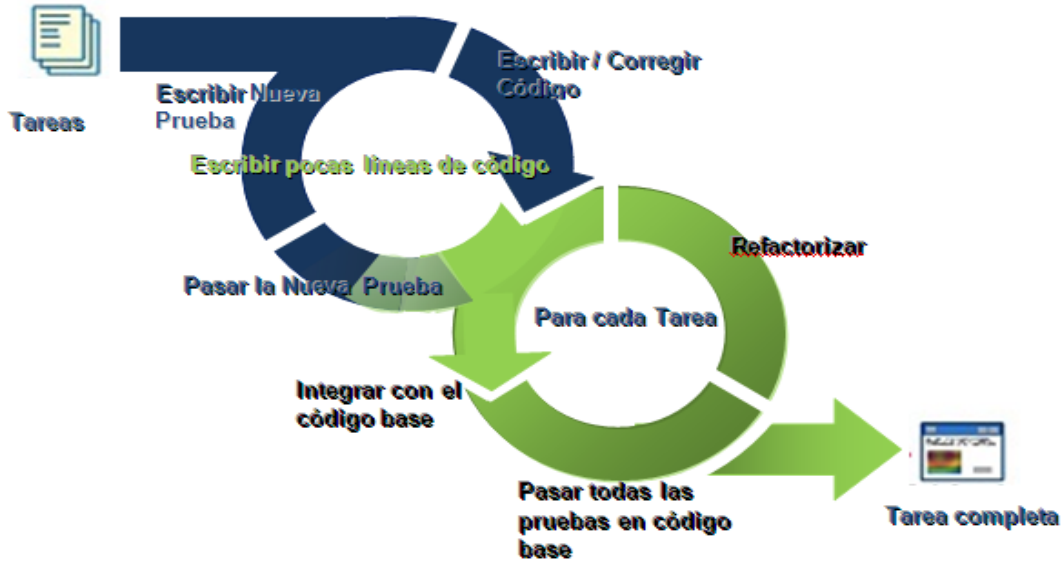


**Figura 1.- Modelo de procesos de TDD**

## Metodología

### Materiales y Métodos

La Metodología de Desarrollo **D**irigido por **P**ruebas y **M**ejora de procesos para PyMEs toma las ventajas del framework TDD definido por Kent Beck [5] y se ajusta a las especificaciones de los métodos Ágiles en su proceso de Desarrollo del producto y Gestión del proyecto. *El modelo DEPYM esta es un modelo de mejora de procesos de software Agil que incluye: Scripts, formas y estándares.*

Los roles que propone el modelo DEPYM para el proceso de desarrollo de software son: Líder de proyecto, analista, arquitecto de software, ingeniero de software y *tester.*

## Modelo propuesto

DEPYM define un documento maestro denominado *Project charter* cuya misión es servir como documento de visión y alcance que permite definir información general sobre el proyecto tal como el nombre, descripción, estimación de fechas de arranque y terminación, involucrados (*stakeholders),* entregables por etapa *(releases)* en las que se dividirá el proyecto y las historias de usuario que compondrán cada entregable. Además este documento permite definir cuál es el estándar de codificación que se usará para construir los programas. De igual forma aquí se define cual será el documento de configuración de software que ayudará al nombrado de carpetas, archivos, formatos de fecha y horas, entre otros. La figura 2 muestra el diagrama general de procesos del modelo DEPYM.

El *Project charter* permite determinar el estado actual del proyecto a través del valor ganado obtenido por cada una de las historias de usuario. Las historias de usuario permiten definir de forma modular cada uno de los requerimientos funcionales del proyecto. Las historias determinan a detalle cada una de las tareas necesarias para llevar a cabo una historia, permitiendo además requerimientos específicos e información adicional sobre los procesos. El modelo DEPYM está totalmente orientado al desarrollo dirigido por pruebas, por tanto para construir el código de una historia de usuario, se definen sus tareas y de cada tarea se definen de forma detallada cada una de las pruebas unitarias que se deberán cumplir para terminar esa tarea. Para la definición de las pruebas de usuario se usa el patrón *Arranging-Act-Assert [6].*

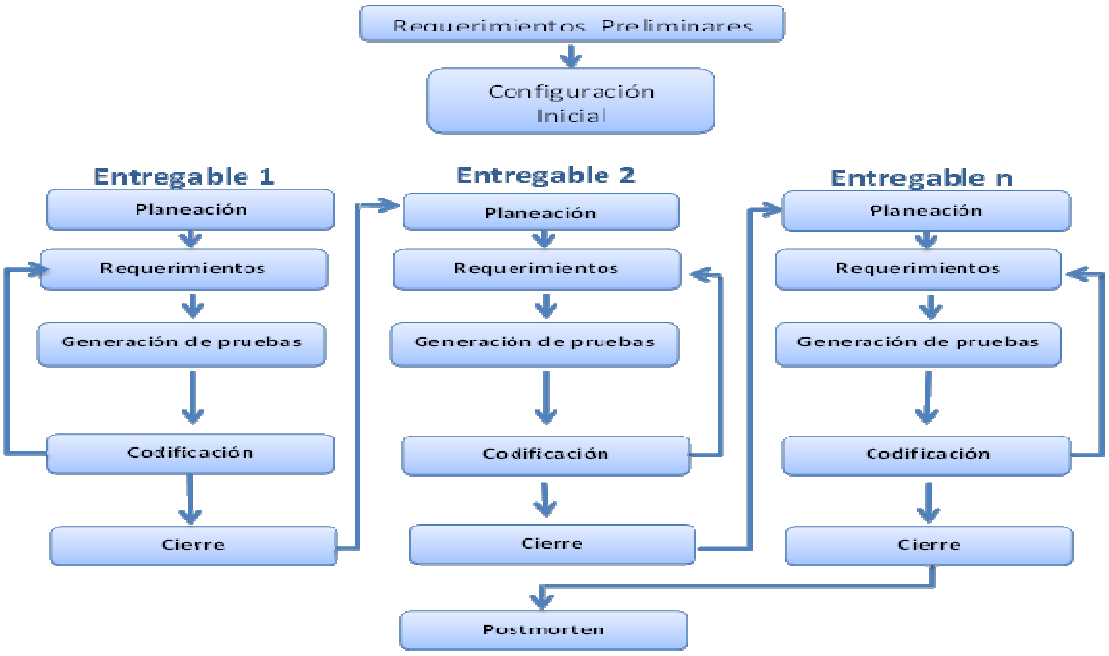


**Figura 2.- Diagrama en bloques del modelo DEPYM**

## Formas Propuestas

El modelo DEPYM propone formas agiles para la planeación, seguimiento y control del proyecto, la Tabla 1 muestra las 10 formas usadas en los diferentes scripts de procesos propuestos por el modelo, cabe mencionar que en la memoria técnica completa del modelo existen scripts para auxiliar en el llenado de dichas formas. En el experimento de validación se uso una hoja de cálculo para crear cada una de las formas.

**Tabla 1.- Formas propuestas por el modelo DEPYM**

| Id Forma | Nombre | Breve descripción |
|---|---|---|
| **A.1.** | Definición breve del proyecto | Esta forma deberá contener: Breve descripción del proyecto, datos generales del clientes, alcances del proyecto, tiempo empleado en las reuniones diarias, Número de semanas que tendrá un entregable, Funcionalidades básicas del proyecto. |
| **A.2.** | Definición de requerimientos no | Breve descripción de requerimientos no funcionales como desempeño, velocidad, sistema operativo, versión |

| | | |
|---|---|---|
| | funcionales | |
| **A.3.** | Definición de arquitectura | Definir la arquitectura a usar dentro del proyecto, n capas, MVC, SOA, entre otros |
| **A.4.** | Entregables del proyecto | Fecha de compromiso para el entregable,<br>Definición breve del entregable |
| **A.5.** | Integrantes del equipo | Nombre del integrante, Rol(es) que desempeñara de forma preponderante |
| **A.6.** | Planeación de historias de usuario (Project Charter) | Nombre de la historia de usuario, Prioridad de 1 a 3, Numero de releáse en que se asignara esta historia de acuerdo a su prioridad, Desarrollador responsable de esta historia, Numero de días en que el desarrollador estima se realizara esta historia, Fecha inicio de la historia, Fecha final de la historia, Valor porcentual planeado en base al 100% que esta historia aporta al proyecto, Valor actual una vez desarrollada esta historia. |
| **A.7.** | Historia de usuario | Historia de usuario ID, nombre, descripción, requerimientos funcionales, notas adicionales, lista de tareas que deben realizarse para completar la historia (Id,nombre, prioridad, fecha inicio, Tiempo en horas, valor planeado y valor ganado |
| **A.8** | Definición de pruebas unitarias para cada Tarea | Lista de pruebas unitarias requeridas para completar la tarea actual ( id Tarea, id prueba, nombre, descipción,<br>Definicicion de la prueba unitaria de acuerdo al patrón<br>*Arranging, Act Assert* |
| **A.9.** | Plan de Riesgos | Descripción, impacto, probabilidad, Estrategia de mitigación, fecha de levantamiento, realizado por, fecha de revisión |
| **A.10.** | Plan de comunicación | Información, medio, Forma de comunicación, quien lo genera, quien lo recibe |

**Estándares propuestos**

El modelo define solo 2 estándares usados en todo el proyecto, al igual que las formas existe un script para auxiliar en la creación de estos artefactos. La tabla 2 muestra los estándares propuestos por DEPYM.

**Tabla 2.- Estándares propuestos por el modelo DEPYM.**

| ID | Nombre | Descripción |
|---|---|---|
| **B.1.** | Estándar de codificación | Convención para definición de comentarios,, declaración de clases e interfaces, indentación, longitud de líneas de código, convención de nombrado de clases, interfaces, variables, métodos, |
| **B.2.** | Administración de configuración del software | Acrónimos, estructura de carpetas, formatos de fecha y hora, nombrado de archivos y carpetas, manejo de versiones |

## Patrón de proceso

En esta sección se desarrolla el *script* general que rige el ciclo de vida de software del modelo DEPYM. Se analiza el patrón de procesos a usar, así como cada una de las etapas que conforman cada entregable: Planeación, requerimientos, diseño, generación de pruebas, codificación, pruebas de integración y cierre. También se analiza el *script* de procesos para llevar a cabo las reuniones

Para la definición de los scripts del modelo DEPYM se sigue un patrón de procesos que deben cumplir siempre con los siguientes elementos:

- Nombre del proceso
- Propósito
- Responsables del proceso
- Criterios de entrada
- Tareas
- Criterios de salida

Dentro del modelo DEPYM se definen criterios y documentos, para efectos de sintaxis los criterios se pondrán en cursivas y los documentos en negritas. La Tabla 3 muestra el script del proceso general de DEPYM.

**Tabla 3.- Script general del modelo DEPYM**

| | | |
|---|---|---|
| Nombre: Script General de DEPYM | | |
| Propósito: Guiar al Líder del proyecto y al equipo en todo el proceso de desarrollo del proyecto | | |
| Responsable del proceso: Líder del proyecto | | |
| Diagrama de proceso | | |
| Criterios de entrada | | *Proyecto autorizado y alineado a la misión de la empresa*<br><br>**Descripción breve del proyecto (Forma A.1.- Definición breve del proyecto)**<br><br>*Roles, integrantes del equipo, horarios* |
| Paso | Actividad | Descripción |
| 1 | Configuración inicial | *1. Realizar la reunión inicial*<br><br>**2. Definición de arquitectura (**Forma A.3. Definición de arquitectura**)**<br><br>**3. Requerimientos no funcionales (**Forma A.2.- Definición de requerimientos no funcionales**)**<br><br>**4. Estándar de codificación (**Forma B.1.- Estándar de codificación**)**<br><br>**5. Administración de la configuración del software (**Estándar B.2.- Administración de configuración del software **)**<br><br>*6. Determinar roles*<br><br>*7. Alcances del proyecto*<br><br>*8.* **Determinar el equipo de trabajo (**Forma A.5.- Integrantes del equipo**)**<br><br>9. **Plan de comunicación (**Forma A.10.- Plan de comunicación**)**<br>*10. Definir la lista de funcionalidades*<br><br>*11. Determinar el numero de entregables y funcionalidades que cubrirá cada uno de ellos.*<br><br>**12. Llenar la forma de entregables del proyecto (Forma A.4.- Entregables del proyecto)** |
| 2 | Planeación | *1. Para el entregable actual realizar un proceso de planeación ágil que permita alcanzar los tiempos y calidad requeridos.*<br><br>*2. Identificar las necesidades del cliente para convertirlas en historias de usuario*<br><br>*3. Realizar un análisis de tiempos*<br><br>*4. Valor planeado que representa cada historia*<br><br>**5. Llenar la forma de planeación de historias de usuario (Forma A.6.- Planeación de historias de usuario)** |

| | | |
|---|---|---|
| 3 | Requerimientos | 1. *Identificar las historias de usuario*<br>2. **Para cada una de ellas se definen cada una de las tareas necesarias para cumplir dicha historia, se llena la Forma de Historia de usuario (Forma A.7. Historia de usuario)**<br>3. *Así mismo se analizan los riesgos, prioridades, esfuerzo requerido y valor planeado de cada tarea* |
| 4 | Generación de pruebas unitarias | 1. Analizar cada una de las tareas de las historias de usuario y se generan las pruebas que deberán pasarse para cumplir con los requerimientos de esa tarea.<br>2. **Se llena la forma de pruebas unitarias (Forma A.8.- Definición de pruebas unitarias para cada tarea)** |
| 5 | Codificación | 1. *Realizar el código más simple que funcione y haga pasar las pruebas definidas en la etapa anterior de acuerdo al estándar de codificación*<br>2. *Refactorizar el código para mejorar la funcionalidad existente.* |
| 6 | Pruebas de integración | 1. *Se prueba la funcionalidad de todas las historias de usuario del entregable y se integran al código base* |
| 7 | Cierre | 1. *Planear el siguiente entregable*<br>2. *Realizar un análisis del proceso.*<br>3. *Si el entregable actual fuera el cierre del proyecto se pasa a la etapa de Post Morten para realizar el proceso de mejora de todo el proyecto* |
| | Criterios de salida | 1. *Planeación del siguiente entregable*<br>2. **Estándar de codificación actualizado**<br>3. **Documento de Configuración de software actualizado (SCM)**<br>4. **Definición de las historias de usuario del siguiente release**<br>5. **Plan de Riesgos actualizado para el siguiente release**<br>6. **Plan de comunicación actualizado** |

## Resultados y discusión

Actualmente se tiene el modelo terminado con el patrón de proceso, el script general cada una de las etapas de desarrollo, las formas definidas y estándares. Para el instrumento de evaluación se definieron métricas apropiadas para entornos Agiles y de acuerdo a su naturaleza las métricas apropiadas son:

**1.- Valor entregado al cliente**.- Permite determinar si el proyecto esta cumpliendo con las expectativas del cliente de acuerdo a cada entregable, ya que en un entorno ágil debe maximizar el valor que se entrega, los resultados deben ser medidos en términos de valor entregado al cliente. Mediante esta métrica, el cliente puede conocer la velocidad con que retorna su inversión. La evaluación se realizo mediante la siguiente ponderación: **A** - Se cumple con la expectativa y se supera, **B** – Cumple la expectativa satisfactoriamente, **C** - Cumple regularmente la expectativa, **D** - Cumple la expectativas con problemas de calidad y **E** - No cumple la expectativa.

**2.- Velocidad de desarrollo**.- Permite evaluar la fecha de finalización del proyecto y/o saber de qué requerimiento/entregable se dispondrá en una fecha determinada en términos de días hábiles de proyecto.

**3.- Horas pendientes en la iteración**.- Permite conocer el porcentaje de avance del proyecto en base al valor ganado de las horas invertidas en el proyecto y a los entregables realizados.

**4.- Numero de defectos por KLOC** (porcentaje de defectos por cada 1000 líneas de código).- Con esta métrica medimos la densidad de defectos en cada entregable con el propósito de disminuirla.

Para la validación del trabajo de investigación se realizaron tres experimentos para aplicar el modelo DEPYM en proyectos reales y diferentes contextos:

1. Proyecto de control de rutas de reparto y logística de ventas (Fase 1), de una compañía transnacional (Univeler) con 9 meses de duración y 11 entregables, el equipo estuvo formando por 3 ingenieros de tiempo completo con jornadas de trabajo de 40 hrs/sem. El proyecto tuvo un total de 4320hrs y una duración de 180 días hábiles. La tabla 4 muestra los resultados obtenidos aplicando las 4 métricas del modelo DEPYM en el proyecto 1.

**Tabla 4.- Resultados obtenidos del proyecto 1 del experimento de evaluación de DEPYM**

| ID | Nombre Entregable | Horas planeadas | Prioridad | %Valor | Horas Acumuladas | Evaluación del Cliente | Vel. Dias | horas pendientes | Defectos KLOC Plan | Defectos KLOC Actual |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Rutas | 777.6 | ALTA | 18 | 777.6 | A | 32.4 | 3542.4 | 35 | 55 |
| 2 | Ventas | 648 | ALTA | 15 | 1425.6 | A | 59.4 | 2894.4 | 32 | 49 |
| 3 | Control Activos | 475.2 | ALTA | 11 | 1900.8 | A | 79.2 | 2419.2 | 30 | 51 |
| 4 | Almacen | 388.8 | ALTA | 9 | 2289.6 | A | 95.4 | 2030.4 | 27 | 36 |
| 5 | Reportes | 388.8 | MEDIANA | 9 | 2678.4 | A | 111.6 | 1641.6 | 26 | 50 |
| 6 | Facturacion | 432 | MEDIANA | 10 | 3110.4 | A | 129.6 | 1209.6 | 24 | 47 |
| 7 | Compras | 259.2 | MEDIANA | 6 | 3369.6 | A | 140.4 | 950.4 | 22 | 42 |
| 8 | Adm. Financiera | 259.2 | MEDIANA | 6 | 3628.8 | B | 151.2 | 691.2 | 19 | 37 |
| 9 | Interfaz Contable | 172.8 | BAJA | 4 | 3801.6 | B | 158.4 | 518.4 | 15 | 32 |
| 10 | Control de Bancos | 216 | BAJA | 5 | 4017.6 | B | 167.4 | 302.4 | 13 | 25 |
| 11 | Interfaz Rec. Hum. | 302.4 | BAJA | 7 | 4320 | C | 180 | 0 | 13 | 19 |
| | Totales | 4320 | | 100 | 4320 | | 180 | | | |

2. Proyecto para el control del Presupuesto de obra, control de recursos humanos, materiales y maquinaria para compañías de Construcción de la región de Orizaba-Córdoba Ver. México. El proyecto fue de 7 meses (140 días hábiles) de duración y 7 entregables, con un equipo de 2 personas. La tabla 5 muestra los resultados obtenidos aplicando DEPYM en el proyecto 2.

**Tabla 5.- Resultados obtenidos del proyecto 2 del experimento de evaluación de DEPYM**

| ID | Nombre Entregable | Horas planeadas | Prioridad | %Valor | Horas Acumuladas | Evaluacion del Cliente | Vel. Dias | horas pendientes | Defectos KLOC Plan | Defectos KLOC Actual |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Control Obra | 515 | ALTA | 23 | 515.2 | A | 32.2 | 1724.8 | 34 | 55 |
| 2 | Estimación de obra | 381 | ALTA | 17 | 896 | A | 56 | 1344 | 32 | 49 |
| 3 | Control de Materiales | 336 | ALTA | 15 | 1232 | B | 77 | 1008 | 31 | 51 |
| 4 | Control Mano Obra | 314 | MEDIANA | 14 | 1545.6 | C | 96.6 | 694.4 | 29 | 36 |
| 5 | Maquinaria y Equipo | 269 | MEDIANA | 12 | 1814.4 | A | 113.4 | 425.6 | 29 | 50 |
| 6 | Adquisiciones | 246 | MEDIANA | 11 | 2060.8 | B | 128.8 | 179.2 | 27 | 47 |
| 7 | Facturación | 179 | BAJA | 8 | 2240 | A | 140 | 0 | 25 | 31 |
| | Totales | 2240 | | 100 | 2240 | | 140 | | | |

3. Un proyecto actualmente en ejecución en un ambiente académico (Generación de una herramienta para automatizar un Modelo de Gestión de indicadores de Logro en el Instituto Tecnológico de Orizaba en Ver. México) planeado a 14 meses con 17 entregables, el equipo está formado por 2 ingenieros, el proyecto tiene un **25%** de avance de acuerdo al valor ganado obtenido a la fecha y al numero de horas acumuladas ejercidas de 1120 contra un total de 4480 del proyecto. La tabla 6 muestra los resultados obtenidos en el proyecto 3.

**Tabla 6.- Resultados obtenidos del proyecto 2 del experimento de evaluación de DEPYM**

| ID | Nombre Entregable | Horas planeadas | Prioridad | %Valor | Horas Acumuladas | Evaluacion del Cliente | Vel. Dias | horas pendientes | Defectos KLOC Plan | Defectos KLOC Actual |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Conocer la empresa | 134.4 | ALTA | 3 | 134.4 | A | 8.4 | 4345.6 | 35 | 55 |
| 2 | Compromiso del CEO | 89.6 | ALTA | 2 | 224 | A | 14 | 4256 | 32 | 49 |
| 3 | Definición del equipo de trabajo | 134.4 | ALTA | 3 | 358.4 | A | 22.4 | 4121.6 | 21 | 51 |
| 4 | Integración del equipo de trabajo | 179.2 | ALTA | 4 | 537.6 | A | 33.6 | 3942.4 | 29 | 51 |
| 5 | Organizar el plan de acción | 403.2 | MEDIANA | 9 | 940.8 | A | 58.8 | 3539.2 | 28 | 51 |
| 6 | Determinar la visión | 179.2 | MEDIANA | 4 | 1120 | A | 70 | 3360 | 28 | 51 |
| 7 | Lluvia ideas | 268.8 | MEDIANA | 6 | 0 | - | 0 | 4480 | 27 | 0 |
| 8 | Definir Valores | 268.8 | MEDIANA | 6 | 0 | - | 0 | 4480 | 25 | 0 |
| 9 | Análisis del entorno | 313.6 | MEDIANA | 7 | 0 | - | 0 | 4480 | 24 | 0 |
| 10 | Clasificación de la información | 313.6 | MEDIANA | 7 | 0 | - | 0 | 4480 | 26 | 0 |
| 11 | Definición de proyectos estratégicos | 448 | MEDIANA | 10 | 0 | - | 0 | 4480 | 25 | 0 |
| 12 | Determinación de prioridades | 134.4 | BAJA | 3 | 0 | - | 0 | 4480 | 23 | 0 |
| 13 | Formulación de la misión | 179.2 | BAJA | 4 | 0 | - | 0 | 4480 | 21 | 0 |
| 14 | Direccionar los objetivos estratégicos | 224 | BAJA | 5 | 0 | - | 0 | 4480 | 20 | 0 |
| 15 | Desarrollar el mapa estratégico | 313.6 | BAJA | 7 | 0 | - | 0 | 4480 | 17 | 0 |
| 16 | Diseñar el sistema de medición | 224 | BAJA | 5 | 0 | - | 0 | 4480 | 15 | 0 |
| 17 | Elaborar el cuadro integral de mando | 672 | BAJA | 15 | 0 | - | 0 | 1120 | 13 | 0 |
| | Totales | 4480 | | 100 | 1120 | | 280 | | | |

## Trabajo futuro

Refinar el modelo en base a resultados del experimento, publicar los resultados de la investigación y la construcción de una herramienta de software en la nube para automatizar el modelo y este disponible para la comunidad en general

## Conclusiones

El resultado de la investigación es favorable ya que se han obtenidos resultados de evaluación satisfactorios a la fecha por parte de los clientes y de acuerdo a las métricas establecidas.

Las contribuciones de la investigación son las siguientes:

1. Creación una metodología Ágil de desarrollo de software adaptable las empresas mexicanas y basado en las mejores prácticas de los métodos Ágiles
2. Script de Procesos, formas y Estándares adaptables a empresas pymes, procesos definidos y documentados para el seguimiento de los scripts, llenado de formas y estándares
3. Experimento de aplicación del modelo en 3 proyectos reales con resultados satisfactorios de acuerdo a las métricas definidas en su instrumento de evaluación

## Referencias


[1] Ruvalcaba Mara, Software Guru Edición Enero-Febrero 2005, México Pág. 4.

[2] E. Barriocanal, M. Urb'an, I. Cuevas, and P. P'erez. An experience in integrating automated unit testing practices in an introductory programming course. ACM SIGCSE Bulletin, 34(4):125–128, December 2002

[3] Maximilien Michael E. Assessing Test-Driven Development at IBM., pages 1. Computer (IEEE Computer Society) 2003.

[4] Bat Thirumalesh, Nagappan Nachiappan. Evaluating the Efficacy of Test-Driven Development: Industrial Case Studies. ISESE'06, September 2006, ACM. Pagina 2. Rio de Janeiro, Brazil.

[5] Beck Kent, "Test Driven Development by example."

[6] William c. wake. OOPSLA 2000


## Autorización y renuncia