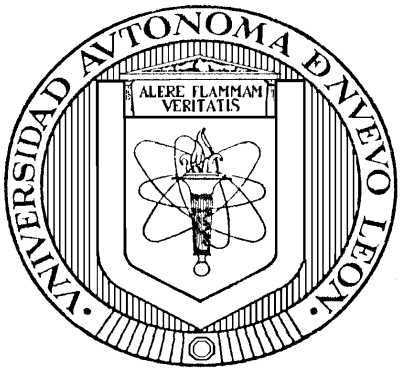
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUDIOS DE POSGRADO



"EVALUACIÓN, SELECCIÓN E IMPLEMENTACIÓN DE UN MODELO DE DESARROLLO DE SOFTWARE PARA LA INDUSTRIA DE LA TRANSFORMACIÓN"

POR

ING. LUIS CÉSAR GONZÁLEZ RAMÍREZ

TESIS
EN OPCIÓN AL GRADO DE MAESTRO EN CIENCIAS DE LA ADMINISTRACIÓN CON ESPECIALIDAD EN SISTEMAS

SAN NICOLÁS DE LOS GARZA, N.L. Diciembre 6 2007

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUIDOS DE POST-GRADO

"EVALUACIÓN, SELECCIÓN E IMPLEMENTACIÓN DE UN MODELO DE DESARROLLO DE SOFTWARE PARA LA INDUSTRIA DE LA TRANSFORMACIÓN"

POR

ING. LUIS CÉSAR GONZÁLEZ RAMÍREZ

TESIS
EN OPCIÓN AL GRADO DE MAESTRO EN CIENCIAS DE LA ADMINISTACIÓN CON ESPECIALIDAD EN SISTEMAS

SAN NICOLÁS DE LOS GARZA N.L. Diciembre 6 2007

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
División de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis "Evaluación, Selección e Implementación de un Modelo de Desarrollo de Software para la Industria de la Transformación"**,** realizada por el alumno **Luis César González Ramírez** matrícula **690456**, sea aceptada para su defensa como opción al grado de Maestro en Ciencias de la Administración con Especialidad en Sistemas.

El Comité de Tesis

_____

Asesor
M.C. Valeria Paola González Duéñez

_____ _____

Revisor
M.C. Carlos Bernardo Garza Treviño

Revisor
M.C. Oscar Rincón Almaraz

_____

Vo. Bo
Dr. Guadalupe Alan Castillo Rodríguez
División de Estudios de Posgrado
de la Facultad de Ingeniería Mecánica y Eléctrica
de la Universidad Autónoma de Nuevo León

San Nicolás de los Garza, Nuevo León a 07 de Diciembre del 2007

PRÓLOGO

En las últimas décadas del siglo XX asistimos a un conjunto de transformaciones económicas-sociales y culturales cuya vertiginosidad y complejidad no admite precedente y nuestro país no se encuentra ajeno a ello, caen rápidamente todo tipo de muros y barreras entre las naciones al mismo tiempo que se amplía la brecha en el nivel de desarrollo humano al que acceden los distintos pueblos.

La globalización implica un proceso de creciente internacionalización del capital financiero, industrial y comercial, nuevas relaciones políticas internacionales y el surgimiento de nuevos procesos productivos, distributivos y de consumo deslocalizados geográficamente, una expansión y uso intensivo de la tecnología sin precedentes.

La globalización significa el aumento de la vincularidad, la expansión y profundización de las distintas relaciones sociales, económicas y políticas, la creciente interdependencia de todas las sociedades entre sí, promovida por el aumento de los flujos económicos, financieros y de comunicación y catapultada por la tercera revolución industrial que facilita que los flujos puedan ser realizados en tiempo real.

Como fenómeno de mercado, la globalización tiene su impulso básico en el progreso técnico y, particularmente, en la capacidad de éste para reducir el costo de mover bienes, servicios, dinero, personas e información.
Esta reducción de la diferencia económica ha permitido aprovechar las oportunidades de arbitraje existentes en los mercados de bienes, servicios y factores disminuyendo la importancia del papel de la geografía y la efectividad de las barreras de la política.

La inestabilidad de las condiciones económicas, de marginamiento y de limitación de las oportunidades se encuentra entre las más sobresalientes. En efecto la aldea global es portadora de oportunidades y riesgos.

El proceso de globalización plantea la oportunidad de mejorar las condiciones de acceso a los mercados que anteriormente se hallaban fragmentados. Los flujos de información, tecnología y capital han sido los que han incrementado su movilidad y por consiguiente constituyen los mercados donde más han mejorado las condiciones de acceso para economías con menor capacidad relativa de generación interna. También crea nuevas oportunidades en tanto incrementa la competencia, sienta las bases para el establecimiento de nuevas alianzas empresarias y sociales y contribuye a la desarticulación de los oligopolios establecidos. La que más se destaca es la heterogeneidad de un fenómeno que se aplica a los bienes, servicios, capitales y de manera bastante desigual, a los hombres. Aquellos que detentan un empleo de producción o de servicio de carácter personal, ven el empeoramiento de su nivel de vida.

En Monterrey, en la industria del acero, tenemos el último gran caso, Grupo Imsa con más 70 años en el mercado, y con un gran numero de empresas asociadas (Corportivo Grupo Imsa, Imsa, APM, Multypanel, VarcoPruden, SteelScape, etc) e Hylsa (Pilar por muchos años de Grupo Alfa), junto con su empresa hermana Galvak. Se fusionaron las dos leyendas en una sola empresa, Ternium. Esta mega fusión solamente es la muestra de la globalización. Ahora la nueva empresa, de capital Italo-Argentino, controla mas del 75% del acero del país.

La globalización es la respuesta de las empresas privadas al entorno cambiado y cambiante de los negocios internacionales. El proceso tiene sus raíces en cuatro aspectos claves, los primeros dos de los cuales son con razón los más importantes: Los cambios de política en la economía mundial, concretamente la liberalización de las corrientes de capital y de comercio, la desregulación de los mercados, y el desarrollo del sector privado incluida la privatización, que han abierto nuevas oportunidades de inversión en la mayoría de los países.

La aceleración del progreso tecnológico que ha cambiado las reglas del juego tanto para los países en desarrollo anfitriones y las empresas de estos países.

América Latina, Europa oriental y central y el Asia central están clasificadas como integradores contingentes, en los que el proceso de posterior integración estará impulsado por las corrientes de capital.

Los países en desarrollo más avanzados están perdiendo las ventajas competitivas en sectores de índice de mano de obra muy elevado. La elevación de los salarios reales ha erosionado sus ventajas de costos, mientras que las cuotas impuestas por los países industrializados limitan su acceso a los mercados, obligándolos a reubicar algunas de sus operaciones de elevado índice de mano de obra en economías de salarios bajos.

La reducción del empleo en la industria manufacturera a medida que las nuevas tecnologías genéricas sustituyen mano de obra por capital, pese a las ventajas comparativas de una mano de obra abundante. Un desplazamiento hacia una mayor dependencia de la producción primaria y algunos servicios (turismo), causados por la pérdida de las ventajas comparativas en la manufactura debida a una actualización tecnológica inadecuada, o al cierre de industrias de sustitución de importaciones que no pueden competir en condiciones de economía abierta.

El lazo indisoluble que se genera en el siglo XX entre la ciencia y la tecnología posibilita acelerar, ampliar y consolidar el proceso de globalización, especialmente, en sus aspectos económicos y culturales.

La digitalización de las comunicaciones humanas ha revolucionado la producción, el almacenamiento y el acceso a la información. Si la revolución industrial multiplicó la fuerza del hombre, la evolución informática multiplica la capacidad del cerebro humano. Hoy la información se ha democratizado, y está al alcance de quien posea una computadora y un módem para acceder a Internet.

Las nuevas tecnologías están creando un mundo donde los valores y las economías repercuten de en un lado a otro; la cultura y los valores humanos están siendo modelados por un medio electrónico.

Estas transformaciones se apoyan en una aceleración sin precedentes en los procesos tecnológicos, tanto en lo que atañe al ritmo mismo de las innovaciones como en lo que se refiere al lapso que transcurre entre la innovación y su incorporación en la producción. Tal proceso se inició en los años 70 y ha llegado a ser tildado como la "tercera revolución tecnológica e industrial". Se ha asentado en la electrónica, la informática, la robótica, los nuevos materiales, la genética y la biotecnología.

ÍNDICE GENERAL

Resumen 1

1. Introducción 3
2. Marco Teórico 5
   - 2.1 Cambios de Paradigmas 5
     - 2.1.1 La Globalización 5
     - 2.1.2 Integración de Empresas 5
   - 2.2 Innovación tecnológica 6
     - 2.2.1 Software basado en componentes 6
     - 2.2.2 Modelo de Madurez de Capacidades 6
     - 2.2.3 ¿Puede Industrializarse el Software? 7
     - 2.2.4 Economías de Escala y Alcance 8
     - 2.2.5 Software Basado en Componentes 10
     - 2.2.6 El modelo CMM 18
   - 2.3 Modelos de Desarrollo de Software, Ventajas y Desventajas 20
     - 2.3.1 Desarrollo interno 20
     - 2.3.2 Personal por Asignación 20
     - 2.3.3 Personal por Proyecto 20
     - 2.3.4 Fábrica de Software 21
3. Proyecto 6sigma: Migración al Concepto de Fabrica de Software 29
   - 3.1 Problemática 29
   - 3.2 Análisis y Representación 30
   - 3.3 Procedimientos 30
     - 3.3.1 Fase de Definición
     - 3.3.2 Fase de Medición 37
     - 3.3.3 Fase de Análisis 42
     - 3.3.4 Fase de Implementación 46

3.3.5 Fase de Control 46
4. Conclusiones y Recomendaciones 48
Bibliografía 50
Lista de Gráficas 51
Lista de Tablas 52
Apéndice A: Cuestionario Aplicado a Proveedores de Desarrollo de Software 53
Glosario de Términos 54
Resumen Autobiográfico 55

RESUMEN

El desarrollo de software, como se practica actualmente, es lento, costoso y sujeto a errores, usualmente obteniendo productos con un gran número de defectos, causando serios problemas de operabilidad, disponibilidad, velocidad, seguridad y otras calidades de servicio.

En un estudio de Standish Group , hace 10 años en EEUU, las empresas gastaban US$250 billones en desarrollo de software al año.

- Sólo el 16% de estos proyectos finaliza en tiempo y bajo el presupuesto planeado.
- Un 31% eran cancelados, por problemas de calidad, con pérdida de US$ 81 billones.
- El 53% excedía su presupuesto en razón de 189%, con pérdidas por US$59 billones.
- Y solo 42% de los proyectos concluidos están relacionados a la definición original.

Estas estadísticas siguen siendo hoy en día validas, seguimos desarrollando software de la misma manera que lo hacíamos diez años atrás. Nuestros métodos y prácticas no han cambiado mucho, y tampoco lo han hecho los costos y riesgos asociados.

El desarrollo de software es un trabajo intensivo, consumiendo más capital humano por dólar, de valor producido de lo que esperamos de una industria moderna. Por supuesto, fuera de estos problemas, los productos del desarrollo de software obviamente ofrecen un valor importante a los consumidores, lo que queda demostrado por una demanda creciente a largo plazo. Esto no significa que los consumidores queden totalmente satisfechos, bien con el software que les es entregado, o con la manera en que se lo entregamos. Sólo significa que ellos

valoran tanto el software que están dispuestos a soportar grandes riesgos y pérdidas en orden de intentar alcanzar los beneficios que el mismo produce.

Sin embargo, esta situación esta a punto de cambiar, ya que la demanda global total de software esta proyectada en aumento de gran magnitud durante los próximos 10 años –dirigida por nuevas fuerzas de la economía global- un ejemplo de estas fuerzas, es la India, que para el desarrollo de su país, aporta el gobierno beneficios fiscales, lo que ha sido punto de lanza para acelerar el crecimiento tecnológico de la región.

1

INTRODUCCION

DESCRIPCIÓN DEL PROBLEMA:

Hoy en día, en un mundo globalizado donde se requiere reducir de forma significativa los costos, optimizar recursos, mejorar y acelerar la integración de las empresas, donde el área de Tecnología de Información (TI) se ha convertido en la piedra angular donde se fundamenta y soporta los retos de las organizaciones.

OBJETIVO:

Determinar el esquema óptimo de desarrollo de software para las empresas con alta demanda de automatización tecnológica, con la finalidad de tener una estructura organizacional flexible, reducir los costos y gastos de operación, asegurando los compromisos de la Gerencia de TI ante la Dirección General en tiempo y forma.

HIPÓTESIS:

El concepto de Fábrica de Software (en ingles “Software Factory”) debe ser considerado, en este lustro, como una de las mejores alternativas en el desarrollo de Software para la empresa con un alto nivel tecnológico que requiere una fuerte demanda de horas de desarrollo.

JUSTIFICACIÓN:

El mercado global donde la empresa compite con márgenes de utilidad muy reducidos, se requiere un alto nivel de desarrollo de software para que el cliente perciba la diferenciación entre nuestra empresa vs. la competencia. Para poder sustentar este alto nivel de desarrollo de software, se requiere una metodología de primer nivel.

Hoy en día, el concepto de fábrica de software está muy desarrollado y existen proveedores en todo el mundo que ofrecen sus servicios con estrictos estándares de calidad a precios muy competitivos; como es el caso de las grandes fábricas de software en la India, las cuales ofrecen servicios de desarrollo en todo el mundo.

LÍMITES DE ESTUDIO:

El estudio será limitado a la demanda que tiene uno de los corporativos más grandes de nuestra localidad, bajo la visión 2007-2011; además ofrecerá un benchmark de las diferentes ofertas de servicio que hoy en día se pueden conseguir en el mercado global y contemplará las ventajas / desventajas de cada tipo.

METODOLOGÍA:

Es un caso práctico de negocio, el cual se está llevando a cabo desde hace más de dos años en una empresa regiomontana. Al inicio de este año fue nuevamente validada la estrategia por la Dirección General, y será replicada en las demás empresas del grupo. A través del sistema de medición 6 Sigma se analizó la problemática, utilizando la metodología DMAIC (Definir, Medir, Analizar, Implementar y Controlar) y con las herramientas estadísticas como Gráficas de Pareto, Histogramas, Correlaciones, Desviaciones Estándar, etc.

2

MARCO TEORICO

2.1. CAMBIO DE PARADIGMAS

Como industria, hemos estado colectivamente en este punto antes. La historia del desarrollo del software es un ejemplo de la complejidad y cambio, el progreso crea cada vez demanda. Mientras el progreso continua a la alza, el desarrollo tecnológico, no ha sido continuo. Al contrario, ha seguido el conocido patrón de curvas de innovación, como se ilustra en la Figura 1

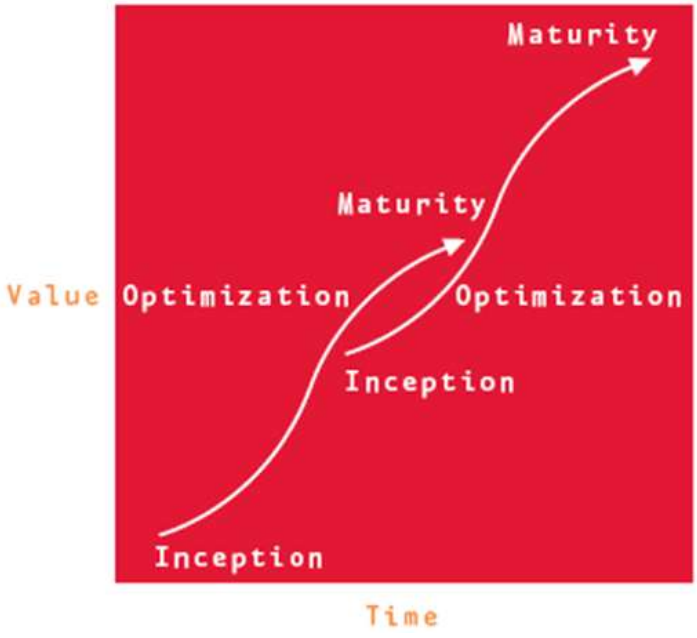


Figura 2.1. Curvas de Innovación

Típicamente, una innovación discontinua establece los cimientos para una nueva generación de tecnologías. El progreso de la nueva tecnología es inicialmente rápido, pero luego comienza gradualmente a desacelerarse, mientras

la tecnología se estabiliza y madura. Eventualmente, la tecnología pierde su habilidad para mantener la innovación, y un punto es alcanzado. En este punto, otra innovación discontinua establece otra tecnología para otra generación de nuevas ideas, y el patrón se repite.

2.2 INNOVACION TECNOLOGICA

2.2.1- ENFRENTANDO LOS FUTUROS CAMBIOS, NUEVAMENTE

Dado que la capacidad de la industria depende del tamaño de la base de desarrolladores competentes y la productividad de los mismos, aumentar la capacidad de la industria requiere o bien más desarrolladores utilizando los métodos y prácticas actuales, ó un número de desarrolladores similar al actual utilizando métodos y prácticas diferentes.

Mientras que la cultura de aprendizaje cultivada durante los últimos diez años parecería haber aumentado el número de desarrolladores competentes y la competencia del desarrollador promedio, no podrán equipar a la industria para satisfacer el nivel de demanda esperado.

La solución debe incluir por lo tanto el cambio de nuestros métodos y prácticas, debemos encontrar maneras de hacer a los desarrolladores mucho más productivos.

2.2.2- INDUSTRIALIZANDO EL DESARROLLO DE SOFTWARE

Otras industrias han incrementado su capacidad mediante el cambio desde artesanía, donde productos enteros son creados desde cero por individuos o equipos muy pequeños, hacia fabricación, donde un rango de variantes de producto es rápidamente ensamblado a partir de componentes reciclables creados por múltiple proveedores, y donde las máquinas automatizan las tareas de memoria o repetitivas. Ellas han estandarizados los procesos, diseños y empaquetados, usando líneas de producto para facilitar el reciclaje sistemático, y cadenas de provisión para reducir costos y riesgos. Algunas son ahora capaces de ajustes en masa, donde las variantes de productos son producidos rápidamente, a un menor costo y bajo demanda para satisfacer los requerimientos específicos de clientes individuales.

2.2.3- ¿PUEDE INDUSTRIALIZARSE EL SOFTWARE?

Los productos de software son en alguna medida como los productos de disciplinas de ingeniería convencional como puentes, construcciones y computadores. Pero hay también ciertas diferencias importantes que le dan al desarrollo de software un sabor único. Debido a que el software es lógico y no físico, sus costos están concentrados en el desarrollo en lugar de producción, y ya que el software no se viste, su buen funcionamiento depende de calidades lógicas como su robustez y correcta programación, en lugar de calidades físicas como dureza y maleabilidad.

Algunas de las discusiones han involucrado comparaciones del tipo “manzanas con naranjas” entre la producción de bienes físicos, por un lado, y el desarrollo de software, por otro. La clave para aclarar la confusión es entender las diferencias entre producción y desarrollo, y entre economías de escala y economías de alcance.

Para poder ofrecer ROI (por sus siglas en ingles, “Return on Investment”, Retorno sobre la Inversión), los componentes re-usables deben ser reutilizados lo suficiente para recuperar el costo que tuvo su desarrollo, bien directamente mediante la reducción de costos o indirectamente, mediante la reducción de riesgos, de tiempo de mercadeo o mejoras de calidad. Los componentes re-usables son activos financieros desde una perspectiva de inversión. Debido a que el costo de desarrollar un componente re-usable es generalmente alto (figura 2), obtener niveles de ganancia de re-uso no es algo tan sencillo.

Figura 2.2. Elemento reciclado

Un enfoque sistemático bajo desarrollos recicladso es entonces requerido. Esto generalmente involucra identificar un dominio en el cual múltiples sistemas van a ser desarrollados, identificar problemas recurrentes en ese dominio,

desarrollar conjuntos de activos integrados de producción que resuelvan estos problemas, y luego aplicarlos mientras los sistemas son desarrollados en ese dominio.

2.2.4- ECONOMÍAS DE ESCALA Y ALCANCE

El reciclaje sistemático puede brindar tanto economías de escala como economías de alcance. Estos dos efectos son ampliamente conocidos en otras industrias. Mientras los dos reducen tiempo y costos, y mejoran la calidad del producto, mediante la producción colectiva de múltiples productos en lugar de la producción individual, difieren en la forma en la que producen estos beneficios.

Las economías de escala aparecen cuando múltiples instancias idénticas de un mismo diseño son producidas colectivamente, en lugar de individualmente, como se ilustra en la Figura 3. Aparecen en la producción de cosas como herramientas donde son utilizadas para producir múltiples instancias idénticas de producto. Un diseño es creado, en conjunto con instancias iniciales, denominadas prototipos, mediante un proceso intensivo en recursos, denominado desarrollo, llevado a cabo por ingenieros. Varias instancias adicionales, denominadas copias, son luego producidas por otro proceso, denominado producción, realizado por máquinas y/o fuerza de trabajo de bajo costo, para satisfacer la demanda del mercado.

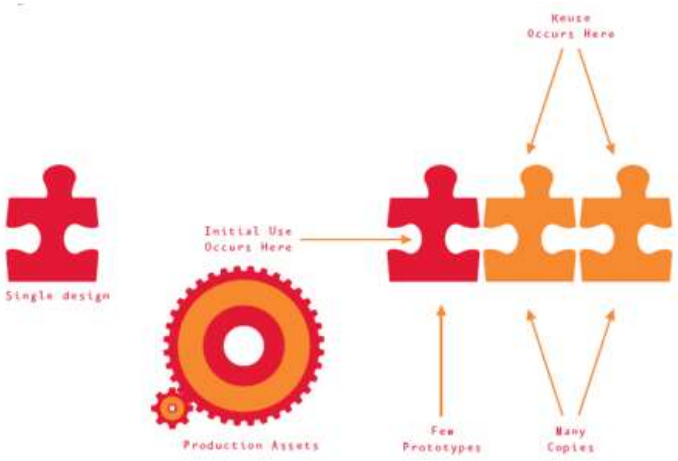


Figura 2.3. Economías de Escala

Las economías de alcance aparecen cuando múltiples diseños similares pero no idénticos y prototipos son construidos colectivamente, en lugar de

individualmente, como se ilustra en la Figura 4. En la fabricación de automotores, por ejemplo, múltiples diseños de automotores similares pero no idénticos son comúnmente desarrollados a partir de la composición de diseños existentes de subcomponentes, como chasis, cuerpo, interior y variantes o modelos son creados mediante la variación de las prestaciones como potencia del motor, de diseños existentes. En otras palabras, las mismas prácticas, procesos, herramientas, y materiales son usados para diseñar y generar prototipos múltiples, es decir productos similares mas no iguales. Lo mismo es cierto en la construcción comercial, donde varios puentes o edificios raramente comparten un diseño en común. Sin embargo, una particularidad interesante de la construcción comercial es que usualmente solo una o dos instancias son producidas de cada diseño exitoso, por lo que las economías de escala son raras de encontrar en estos casos. En la fabricación de automotores, donde varias instancias idénticas son producidas de cada diseño exitoso, las economías de alcance son complementadas por las de escala, como se ilustra por las copias de cada prototipo mostrado en la Figura 4.

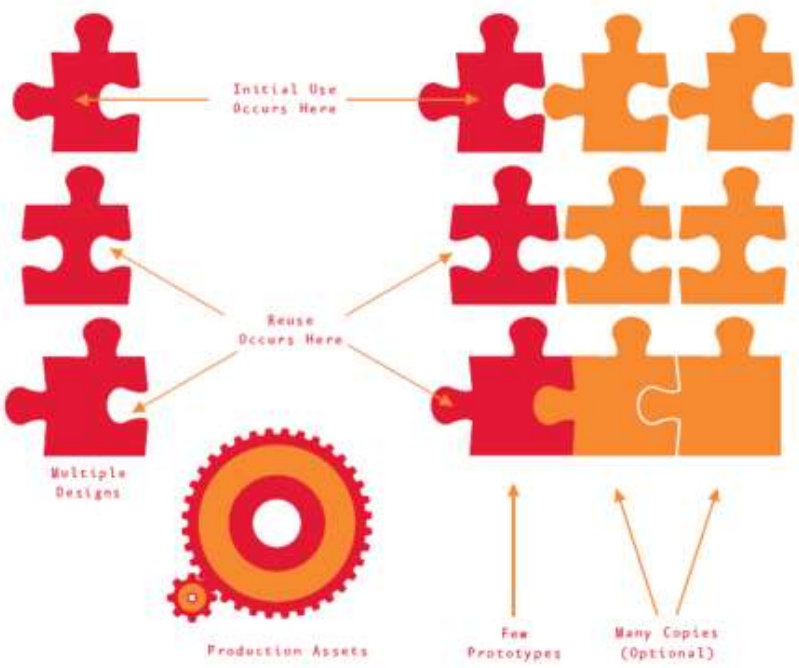


Figura 2.4. Economías de Alcance

Por supuesto, hay importantes diferencias entre software y la fabricación de automotores o la construcción comercial, pero en algunos puntos encuentra similitudes.

- En mercados como el consumidor de escritorio, donde copias de los productos como sistemas operativos y aplicaciones de productividad son producidas en masas, el software exhibe una economía de escala, como la fabricación de automotores.
- En mercados como el empresarial, donde las aplicaciones de negocios desarrolladas para obtener ventajas competitivas no son prácticamente nunca desarrolladas en masa, el software exhibe sólo economías de alcance, como la construcción comercial.

Ahora podemos entender como las manzanas han sido comparadas con naranjas. La producción en industrias físicas ha sido erróneamente comparada con desarrollo en software. No tiene ningún sentido buscar economías de escala en el desarrollo de cualquier naturaleza, sea de software o de bienes físicos. Sí podemos, sin embargo, esperar que la industrialización del desarrollo de software explote las economías de alcance.

2.2.5- DESARROLLO DE SOFTWARE BASADO EN COMPONENTES

La complejidad de los sistemas computacionales actuales nos ha llevado a buscar la reutilización del software existente. El desarrollo de software basado en componentes permite reutilizar piezas de código preelaborado que permiten realizar diversas tareas, conllevando a diversos beneficios como las mejoras a la calidad, la reducción del ciclo de desarrollo y el mayor retorno sobre la inversión. Al comparar la evolución del ambiente de IT con el crecimiento de las metrópolis actuales, podemos entender el origen de muchos problemas que se han presentado históricamente en la construcción de software y vislumbrar las posibles y probables soluciones que nos llevarán hacia la industrialización del software moderno.

Este proceso de industrialización ha dado ya sus inicios con implementaciones como la plataforma .net, la cual impulsa la idea de industrializar el software utilizando tecnologías de componentes. Los avances y mejoras presentados en esta plataforma van mucho más allá de las implementaciones iníciales como COM y CORBA, convirtiendo a los componentes .net en verdaderas piezas de ensamblaje, en un estilo muy similar a las líneas de

ensamblaje modernas. Asimismo, los nuevos paradigmas como las Fábricas de Software proveen de los medios para hacer la transición desde el 'hacer a mano' hacia la fabricación o manufactura de software.

2.2.5.1-Beneficios del Desarrollo de Software basado en Componentes

En esencia, un componente es una pieza de código preelaborado que encapsula alguna funcionalidad expuesta a través de interfaces estándar. Los componentes son los "ingredientes de las aplicaciones", que se juntan y combinan para llevar a cabo una tarea. Es algo muy similar a lo que podemos observar en el equipo de música que tenemos en nuestra sala. Cada componente de aquel aparato ha sido diseñado para acoplarse perfectamente con sus pares, las conexiones son estándar y el protocolo de comunicación está ya preestablecido. Al unirse las partes, obtenemos música para nuestros oídos.

El paradigma de ensamblar componentes y escribir código para hacer que estos componentes funcionen se conoce como Desarrollo de Software Basado en Componentes.

El uso de este paradigma posee algunas ventajas:

- Reutilización del software. Nos lleva a alcanzar un mayor nivel de reutilización de software.
- Simplifica las pruebas. Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados.
- Simplifica el mantenimiento del sistema. Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.
- Mayor calidad. Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo.
- De la misma manera, el optar por comprar componentes de terceros en lugar de desarrollarlos, posee algunas ventajas:

- Ciclos de desarrollo más cortos. La adición de una pieza dada de funcionalidad tomará días en lugar de meses ó años.
- Mejor ROI. Usando correctamente esta estrategia, el retorno sobre la inversión puede ser más favorable que desarrollando los componentes uno mismo.
- Funcionalidad mejorada. Para usar un componente que contenga una pieza de funcionalidad, solo se necesita entender su naturaleza, más no sus detalles internos. Así, una funcionalidad que sería impráctica de implementar en la empresa, se vuelve ahora completamente asequible.

2.2.5.2-Metrópolis: Analogía Evolutiva del Software basado en Componentes

Pat Helland, arquitecto de Microsoft, ha desarrollado una metáfora llamada Metrópolis, en la cual compara la evolución de las tecnologías de la información con la forma como las ciudades de Estados Unidos han evolucionado durante los 2 últimos siglos. Al comprender la evolución de las ciudades actuales, podemos darnos una idea del futuro promisorio que tiene el desarrollo de software basado en componentes.

Para entender entonces lo que está ocurriendo en este momento con el ambiente IT hay que explorar 6 facetas de esta analogía:

- Ciudades - Casas de Software

  Las ciudades evolucionaron gradualmente como lugares para hacer comercio y manufactura. En estas ciudades existían edificios con poca o ninguna conexión entre ellos. Las ciudades tenían un contacto muy limitado con sus ciudades aledañas y desarrollaron su propia cultura, estilo y forma de hacer cosas (Ver Figura 5). De la misma forma, las casas de software evolucionaron gradualmente mientras nuevas aplicaciones fueron construidas y luego extendidas. Cada aplicación separada e independiente de sus similares en la misma casa de software. Cada casa de software tenía su propia cultura, estilo y forma de hacer las cosas (Ver Figura 6):

Figura 2.5: Evolución de las ciudades.

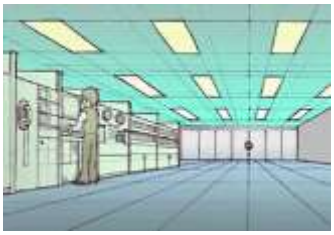


Figura 2.6: Evolución de las casas de software.

- Fábricas y Edificios - Aplicaciones

  En los primeros años del siglo XIX, la manufactura típicamente era simple e independiente. Los bienes producidos estaban limitados tanto por las necesidades del mercado local, como por la sofisticación del proceso de manufactura. Las fábricas producían todas las partes del ensamblado final, armaban el ensamblado e incluso lo vendían. Si uno quería un par de zapatos, tenía que irse a la fábrica de zapatos. Esta no era la forma más eficiente de fabricar bienes y los ítems fabricados eran caros y usualmente no de la mejor calidad (Ver Figura 7). Muchas de nuestras aplicaciones actuales son como aquellas fábricas. Producen datos procesados independientemente unos de otros, los cuales se entregan en 'mercados' limitados. Están integradas verticalmente y usualmente no aceptan el trabajo de otras aplicaciones como entrada (Ver Figura 8).

Figura 2.7: Fábricas y edificios.

Figura 2.8: Aplicaciones.

- Transporte - Comunicaciones

  A mediados del siglo XIX llegó el ferrocarril (figura 9). Se hicieron enormes cantidades de dinero moviendo personas, carbón y trigo de un lugar a otro. Con una demanda arrolladora de transporte, eventualmente todo USA estaba interconectado por ferrocarril. No solo la gente empezó a viajar a nuevos lugares para conocer nuevas culturas, sino que ahora los comerciantes podían vender artículos de formas nunca antes pensadas. Pero más importante aún, el movimiento

de los artículos despertó la expectativa de que las cosas funcionen en conjunto. Antes del ferrocarril simplemente no importaba si los bienes de un fabricante eran incompatibles con los de otro fabricante.

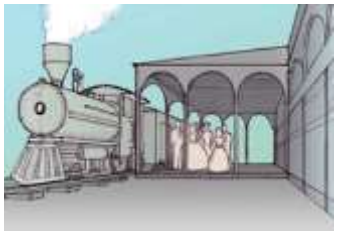


Figura 2.9: Transporte.

Figura 2.10: Comunicaciones.

Al final del siglo XX llegó Internet (figura 10). Se invirtieron montos enormes en navegación, correo electrónico, JPEG y MP3's. Se tendieron los hilos para permitir la navegación y el movimiento de las formas más simples de datos. El navegador le permitió a la persona el transportarse para interactuar directamente con una aplicación distante. Sin embargo, el movimiento de los datos aún no funciona bien en conjunto, haciendo muy limitados los procesos de negocios a través del Internet. Las nuevas conexiones implicaron nuevos cambios en la estandarización de los artículos y los datos. Pronto esto implicaría cambios en los procesos de negocios.

- Bienes Fabricados - Datos Estructurados

A inicios del siglo XVIII los bienes se hacían a mano. Si una llave no encajaba en la cerradura, se ajustaba la cerradura de alguna manera para que permita el paso de la llave. Al establecer controles severos sobre la especificación y producción de las partes componentes, se pudieron realizar masivas producciones de todo tipo de artículos. Sin embargo, esta era una estandarización hacia dentro de las empresas. Pero para finales del siglo XVIII la idea ya se había expandido entre los fabricantes y se produjeron todo tipo de estándares para las partes comunes. Había tamaños y medidas para los artículos, con la expectativa de que aquellos producidos por una fábrica serían intercambiables e interoperables con componentes similares y complementarios producidos por otra. Las compañías que produjeron las partes con un alto grado de precisión tuvieron éxito; lo que tenían un proceso menos consistente, fracasaron (Ver Figura 11 y Figura 12)

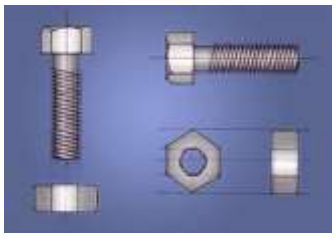


Figura 2.11: Bienes fabricados.

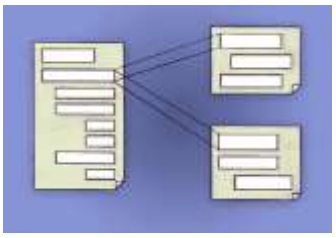


Figura 2.12: Datos estructurados.

Hoy en día aún tenemos estructuras de datos no estandarizadas. Cada aplicación modela la información a su propia manera y dependemos de operadores humanos para 'ajustar' las aplicaciones y así lograr integrarlas. Es necesario agregar semántica para hacer que las aplicaciones se entiendan. De la misma forma como el mercado demandó que se pudieran intercambiar artículos a finales de los '80, el mismo demandará el intercambio de datos en un futuro cercano. Esto significa estandarizar la funcionalidad de conceptos de negocio como 'orden de compra'. Las organizaciones que no se percaten de las eficiencias de la integración-por-diseño perderán a la larga frente a los que persigan estas eficiencias. El resultado de este cambio será un boom económico para las compañías que sobrevivan a él y un dramático mejoramiento para el diario vivir de las personas.

- Ensamblados Fabricados - Empresas Virtuales

  La mayoría de los fabricantes de bicicletas no producen llantas, de la misma forma como quienes hacen camisas no producen sus propios botones. Al crear ensamblados con los mejores componentes disponibles, los fabricantes de bicicletas pueden crear productos más sofisticados y de mayor calidad. Para lograr esto, necesitan especificaciones detalladas de las partes componentes, así como deben también considerar el contexto en el que cada parte será usada. Las compañías de hoy en día están 'creando ensamblados' de su funcionalidad de negocios (Ver Figura 13). En lugar de crear un departamento de distribución y entrega, el trabajo se entrega como outsourcing. En lugar de fabricar el producto, se delega la construcción del mismo a una compañía que se especialice en producción de bajo costo y alta calidad. La definición de “compañía” evoluciona (Ver Figura 14).

Figura 2.13: Ensamblados fabricados.

Figura 2.14: Empresas virtuales.

Las comunicaciones de alta velocidad e información estructurada ofrecen beneficios similares, produciendo la virtualización de las organizaciones. Se puede crear un modelo de componentes de negocio definiendo claramente la semántica y requerimientos operacionales de nuestras capacidades de negocios. Al definir interfaces claras, se puede encapsular los detalles de cómo estas capacidades son implementadas y cada componente puede ser orquestado como miembro de cualquier número de procesos.

- Comercialización y Distribución - Procesos de Negocio

A finales del siglo XIX los centros de comercio urbanos se habían desarrollado. Los bienes se habían vuelto más sofisticados y las opciones del consumidor habían aumentado. Sin embargo, el ir de compras era algo tedioso. Un día de compras podía implicar el tomar el tren hacia la ciudad, luego ir donde el carnicero y luego donde el panadero, pasar comprando las verduras y por último una vuelta por la farmacia. Sin embargo, muy pronto la estandarización permitió la producción masiva y la habilidad de transportar bienes a ubicaciones centrales para la venta y dieron origen a los centros comerciales y el supermercado. Así por ejemplo, Wal-Mart logró nuevas eficiencias al llevar al poder al comercio sobre los productores(proveedores)

Figura 2.15: Comercialización y distribución.

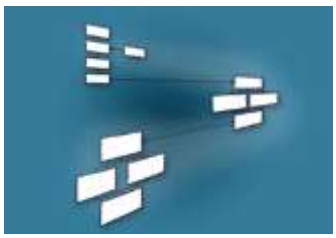


Figura 2.16: Procesos de negocios.

Examinando la situación actual de los procesos de negocios, podemos observar dos tipos de integración. Una de ellas se basa simplemente en enviar un fax y esperar que el mismo haya sido recibido

y que tal vez nos respondan. Otra técnica que reduce errores es la conocida integración A-B, la cual permite el uso de la porta papeles para copiar datos entre aplicaciones. Pero si se quiere hacer algo realmente mejor, se necesita lograr el intercambio y estandarización de datos y operaciones. Los procesos de negocios aún son hechos a mano y los estándares son muy pobres. En lo futuro, los procesos de negocios crecerán para ser la fuerza que le dé la forma y defina los estándares para las nuevas aplicaciones, de la misma forma como Wal-Mart impone los estándares para cientos y miles de artículos (Ver Fig. 16).

2.2.5.3-Fábricas de Software: el Nuevo Paradigma

Las Fábricas de Software son una iniciativa que plasma la necesidad y provee de los medios para hacer la transición desde el 'hacer a mano' hacia la fabricación o manufactura. En sí, una Fábrica de Software es un ambiente de desarrollo configurado para soportar el desarrollo acelerado de un tipo específico de aplicación.

Uno de los elementos clave de este patrón es el elevar el nivel de abstracción de los desarrolladores. Actualmente usamos UML (Lenguaje Unificado de Modelado, por sus siglas en ingles “Unified Modeling Len”) para la documentación. Sin embargo, lo que buscamos hoy en día es la productividad. Mientras los estereotipos y los tags pueden ser usados para decorar modelos UML, la experiencia muestra que se necesitan características más precisas del lenguaje para soportar compilación, depuración, pruebas y otros tipos de tareas de desarrollo.

Pero aún cuando los modelos juegan un rol importante, no lo son todo. Para llegar a niveles más altos de productividad necesitamos la habilidad de configurar, adaptar y ensamblar rápidamente componentes desarrollados independientemente, auto descriptivos e independientes de ubicación, para producir así familias de sistemas similares, pero diferentes. Para ello, será necesaria la creación de patrones con dominio específico, frameworks y herramientas que se alineen tanto con la arquitectura del producto como con el

ciclo de vida del producto. Más aún, debemos considerar los procesos que usamos para analizar requerimientos, desarrollar software y ponerlo en producción. Necesitamos disponer de las mejores prácticas, contenido reutilizable y distintos tipos de patrones. La aplicación de todo esto en conjunto se conoce como Fábrica de Software.

Las fábricas de software son posibles hoy en día y representan el intento de aprender de otras industrias que encaran problemas similares, y aplican patrones específicos de automatización a tareas de desarrollo manual existentes. Las fábricas de software vuelven más rápida, barata y fácil la construcción de aplicaciones, concretando así la visión de la industrialización del software moderno.

2.2.6- EL MODELO CMM

Este modelo establece un conjunto de prácticas o procesos clave agrupados en Áreas Clave de Proceso. Para cada área de proceso define un conjunto de buenas prácticas que habrán de ser:

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

A su vez estas Áreas de Proceso se agrupan en cinco "niveles de madurez", de modo que una organización que tenga institucionalizadas todas las prácticas incluidas en un nivel y sus inferiores, se considera que ha alcanzado ese nivel de madurez.

Los niveles son:

1 - Inicial. Las organizaciones en este nivel no disponen de un ambiente estable para el desarrollo y mantenimiento de software. Aunque se utilicen técnicas correctas de ingeniería, los esfuerzos se ven minados por falta de planificación. El éxito de los proyectos se basa en el esfuerzo personal.

2 - Repetible. En este nivel las organizaciones disponen de unas prácticas institucionalizadas de gestión de proyectos, existen unas métricas básicas y un razonable seguimiento de la calidad. La relación con subcontratistas y clientes está gestionada sistemáticamente.

3 - Definido. Además de una buena gestión de proyectos, a este nivel las organizaciones disponen de correctos procedimientos de coordinación entre grupos, formación del personal, técnicas de ingeniería más detallada y un nivel más avanzado de métricas en los procesos.

4 - Gestionado. Se caracteriza porque las organizaciones disponen de un conjunto de métricas significativas de calidad y productividad, que se usan de modo sistemático para la toma de decisiones y la gestión de riesgos. El software resultante es de alta calidad.

5 - Optimizado. La organización completa está volcada en la mejora continua de los procesos. Se hace uso intensivo de las métricas y se gestiona el proceso de innovación.

Así es como el modelo CMM establece una medida del progreso conforme avanza, en niveles de madurez. Cada nivel a su vez cuenta con un número de áreas de proceso que deben lograrse. El alcanzar estas áreas o estadios se detecta mediante la satisfacción o insatisfacción de varias metas claras y cuantificables. Con la excepción del primer Nivel, cada uno de los restantes Niveles de Madurez está compuesto por un cierto número de Áreas Claves de Proceso, conocidas a través de la documentación del CMM por su sigla inglesa: KPA.

Las prácticas que deben ser realizadas por cada Área Clave de Proceso están organizadas en 5 Características Comunes: Compromiso de la realización, la capacidad de realización, las actividades realizadas, las mediciones y el análisis, la verificación de la implementación.

Se considera típico que una organización dedique unos 18 meses para progresar un nivel, aunque algunas consiguen mejorarlo. En cualquier caso requiere un amplio esfuerzo y un compromiso intenso de la dirección.

2.3 MODELOS DE DESARROLLO DE SOFTWARE, VENTAJAS Y DESVENTAJAS

2.3.1- DESARROLLO INTERNO

El desarrollo con personal interno, si lo vemos de valor de conocimiento, es el mejor, ya que el mismo personal que esta por nomina hace el desarrollo, atiende los mantenimientos y tiene todo el conocimiento del negocio, sin embargo es el desarrollo mas caro que puede pagar una empresa, debido a todas las implicaciones legales que lleva a tener personal por nomina. Pocas empresas manejan este esquema, y las empresas que lo llevan, lo utilizan en los principales temas del negocio.

El personal interno tiene lazos legales muy fuertes con la compañía con la que trabajo, por lo que el tiempo donde esta ligado la empresa-empleado suele ser largos (varios años)

2.3.2- PERSONAL POR ASIGNACIóN

El desarrollo con personal por asignación, es lo mas parecido que el desarrollo con personal interno, sin embargo los lazos con la empresa son casi nulos, por lo que este tipo de personal son de alta rotación. Pocas empresas logran tener un lazo “afectivo” con dicho personal. El costo de la fuga de conocimiento es grande, sin embargo los lazos legales son muy económicos.

2.3.3- PERSONAL POR PROYECTO

El desarrollo con personal por proyecto, es efectivo. El personal solo llega a la empresa por un periodo de tiempo determinado. Con una clara definición de tareas y tiempos asignados. Una vez terminado el proyecto, el desarrollador deja la organización. Este tipo de contratación, se da en proyectos donde el conocimiento del personal externo solo es necesario, por un periodo de tiempo finito.

2.3.4- FABRICA DE SOFTWARE

Una fábrica de software (FS) es un conjunto empaquetado de procesos, herramientas y otros activos integrados que se usan para acelerar las tareas del ciclo de vida para un determinado tipo de componente de software, aplicación o sistema. Este esquema es muy productivo, cuando la empresa que lo contrata, visualiza un esquema de trabajo de largo plazo. Es posible acelerar los proyectos si se brinda a los profesionales una orientación que les ayude a saber lo que deben hacer, cuándo, por qué y cómo. Para ello, proporcionamos orientación con una formalización justa de los procesos, con componentes que se puedan ensamblar y configurar rápidamente, o marcos de trabajo que se puedan completar en poco tiempo, así como con herramientas especializadas que automaticen de forma completa o parcial las actividades repetitivas o que requieren baja calificación.

Uno de los principales objetivos que se pretende conseguir con una FS es aprender de las soluciones que hemos creado anteriormente para responder a problemas o requisitos comunes, y aplicar ese aprendizaje a futuros proyectos. Para ello, necesitamos un medio que nos permita describir esas soluciones reutilizables, y otro para organizarlas en torno a áreas de interés en que suelen encontrarse los problemas o requisitos a los que responden. Una organización de este tipo contribuye a reducir el conjunto de problemas o requisitos en cualquier momento, lo que facilita la identificación de las soluciones reutilizables pertinentes.

Configuración de características

Una de las claves para la creación de una fábrica eficaz consiste en definir productos de trabajo, actividades y activos que se puedan usar en un gran número de soluciones distintas. Por ejemplo, el punto de vista de biblioteca de acceso a datos puede proporcionar una biblioteca que le ayude a obtener acceso a una base de datos SQL.

Quizá no siempre use el mismo RDBMS para todos los proyectos, por lo que su biblioteca deberá adaptarse a otros RDBMS. Puede que los productos de trabajo usados con la biblioteca y las descripciones de las actividades realizadas

con la biblioteca también deben adaptarse en consecuencia. Cuanto mayor sea la variabilidad que acepte un punto de vista, más flexibilidad le ofrecerá para aplicarla a distintas soluciones, aunque su configuración también requerirá más trabajo. La adaptación a una mayor variabilidad implica un aumento del grado de complejidad. Deberá buscar un punto de equilibrio en la variabilidad para que la fábrica de software resulte eficaz. Cuanto más genérica sea la fábrica, menores serán los logros en productividad y capacidad de previsión.

Una buena forma de determinar el margen de variabilidad adecuado es a través de un análisis de características de las soluciones que ha previsto crear. Mediante una técnica de análisis de similitudes y variabilidad (commonality variability, C/V), es posible separar las características comunes (u obligatorias) que deben presentarse del mismo modo en todas las soluciones, de las características variables (u opcionales) que podrían aparecer sólo en algunas soluciones o que difieren en su presentación según la solución de que se trate. La descripción detallada de este tipo de análisis se sitúa fuera del ámbito de este artículo, aunque existen numerosas publicaciones que tratan sobre el tema.

En la fábrica que crea aplicaciones empresariales administrativas a partir de SOA, el acceso a datos es una característica obligatoria (dado que en todas las soluciones se llevará a cabo), pero el cliente web es una característica opcional (dado que en algunas soluciones habrá clientes web y, en otras, clientes de Windows o no habrá ningún cliente).

En una fábrica tiene la posibilidad de usar modelos de características para describir las características que pueden aparecer en un miembro de la línea de productos, para separar las características comunes de las variables y para indicar cómo pueden aparecer las características variables. En los modelos de características también se puede definir qué efectos tendrán las decisiones acerca de las características variables; por ejemplo, se puede indicar que una característica variable requiera la intervención de otra y que ésta, a su vez, sea incompatible con una tercera. Estas decisiones se toman para una aplicación determinada mediante la configuración del modelo de características. La configuración es un proceso simple en el que se debe especificar qué

características variables descritas por el modelo aparecerán en la aplicación, y de qué modo aparecerá cada una.

El trabajo con modelos de características es sólo uno de los mecanismos bien conocidos que se pueden usar para describir la variabilidad. Otros son los formularios, las tablas, los asistentes, los diseñadores, las plantillas, los patrones, los scripts y el código. Los mecanismos de variabilidad se pueden usar de forma independiente y en distintas combinaciones para especificar e implementar características variables en una fábrica de software.

Claves para el éxito:

- Encontrar las similitudes y variaciones en los productos.
- Medir el rendimiento del proceso actual de desarrollo de productos en términos de productividad y calidad.
- Definir y mejorar un proceso que permita un desarrollo eficaz de los productos.
- Proporcionar un modelo KM que ayude a todos los agentes a conocer los logros en materia de productividad y calidad.
- NO Diseñar más de un proyecto a la vez.
- Desarrollar de un modo rápido y económico activos reutilizables que encapsulen conocimientos y cuya reutilización resulte fácil; sobre todo en el caso de las herramientas personalizadas.
- Identificar dominios específicos y tratarlos con herramientas y procesos personalizados, en lugar de intentar aplicar procesos y herramientas generales a todos los dominios.

Medida de la calidad y la productividad

Como podemos ver, el esquema de fábrica proporciona un mecanismo útil para organizar métricas. Dado que cada punto de vista se orienta a un aspecto determinado del proceso de desarrollo del software, es posible usar puntos de vista para definir medidas de productividad y de calidad. El uso de estas medidas permite recopilar datos sobre aspectos determinados del proceso de desarrollo del software.

El análisis de los datos permite determinar qué puntos de vista deben mejorar y de qué manera, así como lo que se obtiene a través de esas mejoras. Para implementar este método necesitaremos una forma de expresar el tamaño del producto, así como el tiempo y el presupuesto dedicados y la calidad del producto. Estas medidas podrán servir para cuantificar la capacidad de previsión, la productividad y la calidad de cada punto de vista. También se pueden usar para evaluar los productos finales de la fábrica. Midiendo cada punto de vista y el rendimiento general de la fábrica, puede determinar de qué modo afecta cada punto de vista al rendimiento general y, por lo tanto, cuánto debe invertirse para las actividades en un punto de vista determinado con activos reutilizables.

Este proceso ayuda a obtener el máximo rendimiento de la inversión por lo que respecta a la capacidad de previsión, productividad y calidad. Le ayuda a comparar los resultados de los objetivos establecidos inicialmente antes de empezar el desarrollo de la fábrica.

Uso de puntos de función para expresar el tamaño del producto

Para cuantificar la productividad, se necesita una métrica que pueda usarse para expresarla en términos de volumen de producto de software creado en un período de tiempo. Cuando es posible predecir el tamaño del sistema y medir el crecimiento del tamaño del producto durante el desarrollo, podemos anticipar mejor el tiempo necesario para completar el proyecto y medir la productividad en horas por unidad de tamaño del producto. Al medir el tamaño y el crecimiento reales, podemos identificar diferencias entre los valores reales y previstos, y empezar a analizar y administrar las diferencias cuando se hacen evidentes.

Si usamos una fábrica de software, dispondremos de dos ventajas que nos permitirán aumentar considerablemente la capacidad de previsión. En primer lugar, desarrollamos un miembro de una familia de productos determinada, con características conocidas, no una aplicación arbitraria. Puesto que la fábrica nos permite describir una familia de productos y sus características destacadas y, lo que es más importante, mejorar esa descripción a medida que se adquiere experiencia en el curso de diversos proyectos, sabremos mucho más acerca de una aplicación que se desarrolle con una fábrica que de una aplicación arbitraria.

En segundo lugar, desarrollamos la aplicación aplicando la orientación prescriptiva que suministra la fábrica. Por lo tanto, realizamos muchas tareas de desarrollo de forma muy similar en aplicaciones sucesivas, con ayuda de los mismos patrones, plantillas, bibliotecas y herramientas. Si estandarizamos determinados procedimientos, con una fábrica tienden a suprimirse las variaciones innecesarias del proceso de desarrollo, de modo que resulta mucho más probable que el tamaño del producto y su crecimiento sigan patrones similares en distintas aplicaciones.

Si usamos una fábrica de software, la medida de estos valores y la identificación, el análisis y la administración de las diferencias entre los valores planeados y reales pueden resultar de suma utilidad. Ya hay muchos métodos de estimación que pueden ayudarle a determinar el tamaño del sistema. Si desea una métrica para expresar el tamaño y la productividad, necesitará una cuantificación objetiva. Estos puntos de función se pueden considerar una métrica aproximada para determinar el tamaño de un sistema y calcular el esfuerzo y la programación. Durante el desarrollo, la métrica puede servir para determinar si el proyecto requiere más o menos trabajo que otros proyectos similares.

Si examinamos con más detenimiento la capacidad de previsión, vemos que una fábrica nos permite separar las características comunes de todos los miembros de una familia de productos de las características variables presentes únicamente en algunos miembros, o que presentan distintos tamaños o características según de qué miembros se trate. En lugar de recopilar requisitos para un determinado producto a partir de cero, podemos suponer que presenta las características compartidas por todos los miembros de la familia y concentrarnos en la especificación de sus requisitos únicos o variables.

Volviendo al análisis de puntos de función en el contexto de una fábrica, vemos que podemos empezar con un tamaño de producto mínimo fijo que siempre tendremos, dado que en nuestra familia de productos siempre hay ciertas partes fijas. Este tamaño de producto mínimo fijo constituye una medida de las características comunes de la familia de productos. También podemos definir

tamaños computables para muchas de las características variables que podrían agregarse o no a nuestro perfil de producto básico. A partir de esta información, tenemos la posibilidad de calcular el costo de una determinada configuración de características. Esa información, a su vez, puede ayudarnos a decidir qué características deben crearse. Dicho de otro modo, el análisis de puntos de función basado en la configuración de características nos proporciona un medio para evaluar sobre una base más sólida los pros y los contras con respecto a los costos y la funcionalidad.

La precisión de estas predicciones depende mucho del nivel de las métricas recopiladas y de la similitud de los futuros proyectos de desarrollo del sistema con los que ha trabajado y para los que ha recopilado métricas. Cada punto de vista de una fábrica de software admite un cierto margen de variabilidad en el desarrollo del sistema. A su vez, el grado de variabilidad determina qué clases de activos puede ofrecer el punto de vista al usuario de la fábrica y, por lo tanto, el nivel de coherencia entre un proyecto y el siguiente por lo que respecta a ese aspecto del producto. Por ejemplo, en la primera versión de nuestra fábrica, quizá sólo haya algunos puntos de vista en los que se describa la arquitectura del sistema y que proporcionen únicamente instrucciones para los desarrolladores acerca de la puesta en práctica de la implementación. Los desarrolladores que usen esta fábrica deberán escribir una gran cantidad de código manualmente. Sin embargo, tras crear varios sistemas con esta fábrica, es posible que la construcción de la parte de la interfaz de usuario de esos sistemas tienda a variar considerablemente, ya que los desarrolladores interpretan las indicaciones de maneras muy distintas. A partir de esa observación, podría llegar a la conclusión de que el punto de vista de interfaz de usuario admite un amplio margen de variabilidad. Cuando tiene en la fábrica muchos puntos de vista con una gran variabilidad, las medidas y predicciones serán menos precisas que cuando la variabilidad de los puntos de vista es muy limitada.

Llegados a este punto, podríamos preguntarnos si el nivel de variabilidad asociado a un punto de vista determinado es realmente necesario. Si no es así, podríamos aumentar la precisión de nuestras medidas y predicciones quitando la variabilidad sobrante o gratuita. Fijémonos de nuevo en el punto de vista de

interfaz de usuario, por ejemplo; pero esta vez, proporcionaremos un conjunto de componentes de biblioteca compatibles con las instrucciones y una herramienta gráfica que configure las rutas de navegación de los usuarios. Al proporcionar estos activos, formalizamos algunos aspectos del proceso de desarrollo de la interfaz de usuario que antes se definían de un modo menos estricto. Con esta formalización se reduce el margen de variabilidad que admite el punto de vista de interfaz de usuario. Los sistemas que se desarrollen con esta fábrica presentarán una mayor uniformidad en la creación de la interfaz de usuario, de modo que nuestras medidas y predicciones resultarán más precisas. Dado que la magnitud de los errores en los cálculos se reduce cuando disminuye la variabilidad que admite la fábrica, la capacidad de predicción puede mejorar con la evolución de la fábrica hacia una reducción de la variabilidad excesiva o innecesaria. En la práctica, la productividad y la calidad también tienden a mejorar con una mayor capacidad de predicción, dado que la reducción de variabilidad comporta una disminución del tiempo necesario para crear una determinada característica, así como del número de defectos. En este punto, se impone una llamada a la prudencia: La variabilidad debe reducirse en la medida de lo posible, pero nunca hasta tal punto que el proyecto se prolongue o sea más costoso debido a las restricciones impuestas a los desarrolladores.

Cuando empiece a usar puntos de función, al principio puede recurrir a datos históricos de organizaciones del entorno que figuren en la documentación de que disponga para realizar sus primeras estimaciones. Los datos históricos resultan útiles porque incluyen las influencias de la organización. La misma idea se aplica al uso de datos históricos en la fábrica de software. Los proyectos desarrollados con una fábrica de software tendrán mucho en común. Aunque no cuente con datos históricos de proyectos anteriores, puede recopilar datos de su proyecto actual y usarlos como base para calcular la parte restante del proyecto. Su objetivo debería ser una transición lo más rápida posible desde el uso de datos de organización o "estándar" para el sector a datos de fábrica y de proyecto

Uso de medidas para mejorar una fábrica

Al establecer una línea de base o calibrar efectivamente los parámetros de productividad y calidad medidos en horas por punto de función y defectos por punto de función, tiene la posibilidad de analizar los datos de proyecto e identificar

actividades que puedan requerir mucho tiempo, o bien puntos de vista que suelan tener asociados muchos defectos. Tras la calibración puede empezar a cambiar la organización de su fábrica y mejorarla de diversas formas; por ejemplo, las técnicas que requiere, el proceso que define o los activos reutilizables que proporciona. Es esencial identificar las áreas en las que se precisan mejoras, de modo que la orientación de las inversiones resulte acertada. Sin embargo, eso debería ser relativamente sencillo, ahora que dispone de un medio para determinar la contribución de cada punto de vista en términos de capacidad de previsión, productividad y calidad. Cuando disponga de una línea de base con datos iníciales, puede ejecutar un bucle continuo para analizar el rendimiento de cada punto de vista, usar esa información para determinar qué se debe mejorar, realizar las mejoras y después repetir el proceso.

3

PROYECTO 6SIGMA: MIGRACION AL CONCEPTO DE FABRICA DE SOFTWARE

3.1 PROBLEMÁTICA

¿Los procesos del desarrollo del software se pueden caracterizar completamente por tres medidas simples?

- Tiempo: ¿el tiempo requerido para realizar una tarea?
- Tamaño: ¿el tamaño del producto del trabajo producido?
- Defectos: el número y el tipo de defectos,

Las metas típicas son alcanzar costos más bajos del desarrollo y de mantenimiento, ciclos de desarrollo más rápidos, y una calidad del producto más alta.

La duración de la productividad y de ciclo no se puede manejar directamente. En un sentido son salidas del proceso usado para desarrollar el software. Se controlan por otras variables, calidades del producto y veces en tarea. La calidad de los productos de software intermedios controla directamente la cantidad de tiempo pasada en la integración y la prueba. Éste es el típicamente

30% - 50% de costo del desarrollo. La calidad se puede manejar directamente usando las técnicas estadísticas que rinden los productos que se mueven a través de prueba muy rápidamente y tiene muy pocos defectos latentes restantes cuando está desplegada.

Al mejorar las prácticas del área de calidad de Software puede rendir mejoras del 15% - 25% en la reducción del ciclo. El tiempo en una tarea se define como la cantidad de tiempo que pasó para poder sacar un producto. Los datos han demostrado que la mayoría de las organizaciones consiguen tiempo efectivos entre el 30% - 50% en tareas de una semana del trabajo. El tiempo en tarea es otra área que se presta a la mejora cuando existe la información para ser validada via técnicas estadísticas. Hemos visto con frecuencia mejoras del 50% en el plazo de 6 meses. Mientras que se mejora la calidad del producto y menos defectos se encuentran al llegar a esta etapa. Cuando la revisión se convierte en el mecanismo primario para encontrar y para quitar defectos, el número de defectos se reducen considerablemente antes de llegar a la etapa de pruebas.

3.2 ANÁLISIS Y REPRESENTACIÓN

La gerencia TI tenía la duda, de basar nuestro desarrollo en personal de Outsourcing vía asignación de recursos (como tradicionalmente se había llevado a cabo en los últimos 10 años dentro de la empresa) ó buscar el desarrollo a través de una “fabrica de Software”

La hipótesis a defender es: El desarrollo de software que requieren las empresas con alta dependencia de desarrollos a la medida, debe ser desarrollado a través de “fabrica de software”, por ser más eficiente y por tener costos más competitivos, y no por Outsourcing (personal por administración)

3.3- PROCEDIMIENTOS

En nuestro caso, el área de Tecnología de Información desarrolla Sistemas hechos a la medida según los requerimientos de los usuarios a lo largo de toda la organización.

Una de las grandes diferencias entre la metodología 6 sigma y las típicas metodologías de Desarrollo de Software (SDLC, sus siglas en inglés), es el concepto de la calidad. Para 6isgma se habla del tema desde la misma primera etapa del proyecto, sin embargo las metodologías tradicionales de desarrollo, la calidad suele ir al final del desarrollo. La otra gran diferencia es que desde el inicio del proyecto bajo 6sigma, te piden basar el proyecto bajo ahorro/beneficios, mientras que el SDLC no.

A través de un análisis basado en la metodología 6SIGMA apoyados en el DMAIC, demostramos que el desarrollo bajo el concepto de fabrica de software ofrece más ventajas a la organización y reducción de costos importantes al área de Tecnología vs el método tradicional de personal por administración.

La empresa cuenta con más de cinco años de experiencia en la implementación de proyectos 6sigma, apoyados con el Master Black Belt y de la gerencia de mejora continua.

Sabemos que típicamente los proyectos 6sigma se llevan de 4 a 6 meses para ver los impactos. (Reducción del 15 al 25% en costos, en este caso costos del desarrollo del software, manteniendo este ahorro de manera constante)

Al seleccionar un proyecto a implementar bajo la metodología 6SIGMA, inicialmente se define un objetivo general, un alcance básico y las métricas en cuestión de beneficios y KPI, esta selección debe estar alineada a la estrategia del negocio (En este momento tenemos una idea general de nuestra variable “Y”).

Figura 3.1 Proyecto ligado al plan estratégico del negocio

A continuación se hace la representación de cada etapa de metodología, para finalmente mostrar las conclusiones y beneficios encontrados.

3.3.1-FASE DE DEFINICION

La primera fase de un proyecto de seis sigma, consta de cuatro pasos:

- Identifica al Cliente del proceso a ser mejorado y identifica el COPQ (Costo de Pobre Calidad), el cual será el foco a reducir en el proyecto. El COPQ se puede también referir en el proyecto como la "Y" [como en Y=f(x)].
- Necesitamos desarrollar el Objetivo y Alcance
- Necesitamos entender el flujo del proceso, para poder en las siguientes etapas buscar mejoras en cada paso del proceso, en esta etapa se construye el diagrama del proceso (Flujo del Proceso)
- Definir el compromiso de reducir del COPQ. (Y a reducir)

Como regla general, el alcance de un proyecto 6 sigma DMAIC es limitado para poder terminar el proyecto en el plazo de cuatro a seis meses, esta pauta evita los proyectos se alargan indefinidamente.

La experiencia establece claramente que los proyectos grandes tienen un alto porcentaje de probabilidad de fracaso. Por lo tanto, es necesario descomponer el objetivo inicial del proyecto en una serie de objetivos de nivel inferior que se conviertan en proyectos individuales ó fases.

El diagrama siguiente ilustra una forma que dividir en fases una "Y grande". Logrando una serie de objetivos más pequeños, más manejables, que contribuyan a la realización de la meta global - en este caso, mejorando "Reducción del gasto del desarrollo de software" (entregar proyectos el tiempo, con esfuerzo razonable, y con un número aceptable de defectos lanzados) bajo el concepto de fabrica de software. ¿Esta descomposición plantea la aplicación de prioridades y la selección del proyecto – tomando en cuenta nuestras limitaciones en recursos

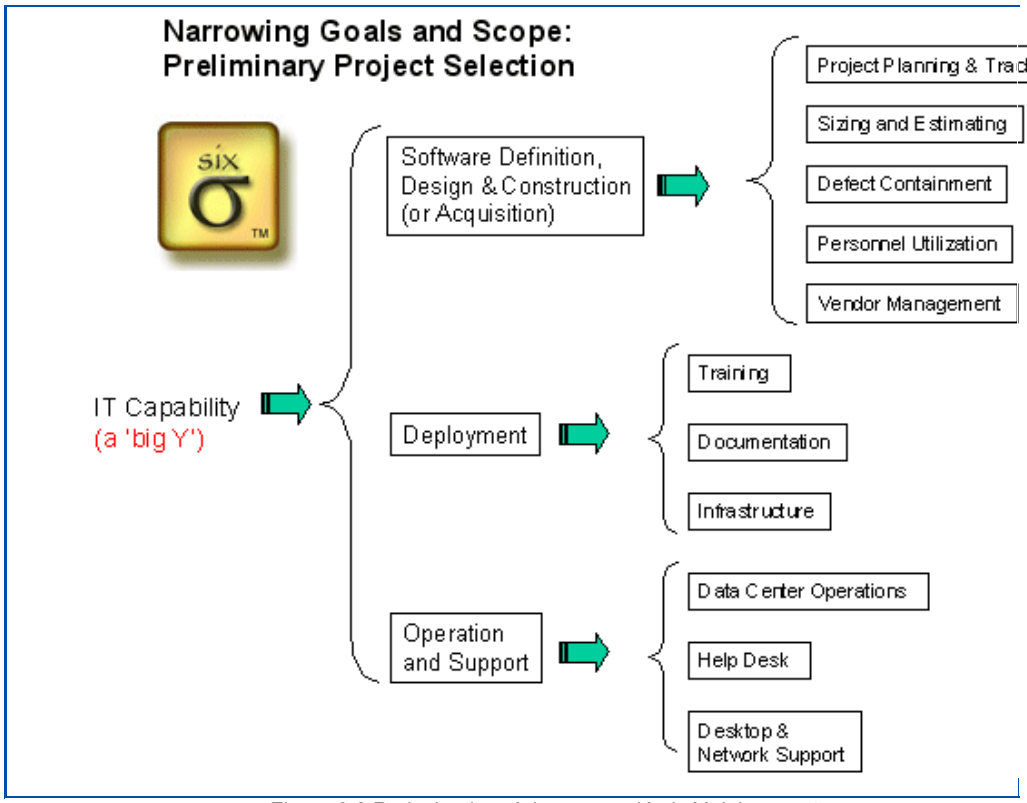


Figura 3.2 Reducir a la máxima expresión la Y del proyecto

Si profundizamos mas sobre nuestra variable “Y”, debemos incluir tiempo de construcción, costo total del desarrollo y calidad entregada en términos de defectos. Reconocemos que hay potencialmente muchos factores que influencian estos resultados, así que necesitamos descomponer más la variable Y para conseguir un proyecto con un alcance manejable.

Después, debemos enfocarnos en una de las gerencias (divisiones) a las que le damos servicios. Para esto podemos ver la estadística apoyados con las graficas de Pareto (Figura 19) . Para tomar las áreas que más demandan nuestros servicios de desarrollo. Así seleccionamos la gerencia con la que hemos invertido más recursos en los últimos 12 meses.

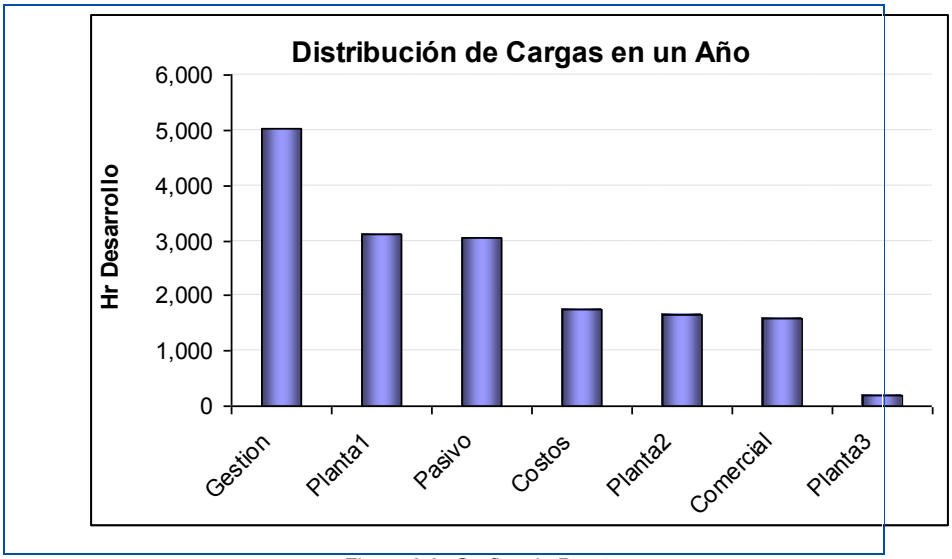


Figura 3.3- Grafica de Pareto

Una vez identificado el área de negocio a trabajar y su problemática básica.

La declaración del problema: Esta es la definición del problema a solucionar o mejorar "El desfase en los tiempos de entrega del desarrollo (software) al cliente final, afecta la satisfacción del cliente-proveedor, le afecta a la operación del día con día del cliente y causa mayor gastos a TI." (Mas horas de desarrollo, mas horas de pruebas)

Objetivo: Aquí debemos indicar el objetivo en términos financieros Y. Esto podría ser como "Mejorar el porcentaje de entregas en tiempo de entrega en los proyectos, subiendo el objetivo del 66% al 90% en los próximos 12 meses."

Equipo de proyecto - por ejemplo, el líder de proyecto será el Black Belt del área de TI (Puede ser un Green Belt - depende del monto esperado de ahorro) ¿El patrocinador de proyecto (Champion) es el Gerente de desarrollo de Sistemas, y equipo que los miembros incluirán al Administrador de Proyectos, tres líderes de proyectos (los mas proactivos) y al encargado de calidad de desarrollo.

Alcance y límites del proyecto: "El proyecto solo incluye desarrollos con componentes comunes, con fecha de inicio y final comprometidas con las gerencias".

Oportunidad financiera – “Las tarifas por hora de desarrollo en contratos bajo concepto de fábrica de software son entre 20 y 30% más económicos vs Personal por Administración, debido a la mejor administración que se lleva de los recursos”. En cálculos previos visualizamos ahorros entre los $82,000 a $100,000 usd + mejoras en la satisfacción del usuario + Indirectos (Liberación de espacio de oficina—los recursos vía fabrica, por lo general no están dentro de las oficinas del cliente)

Figura 3.4- Flujo del proceso

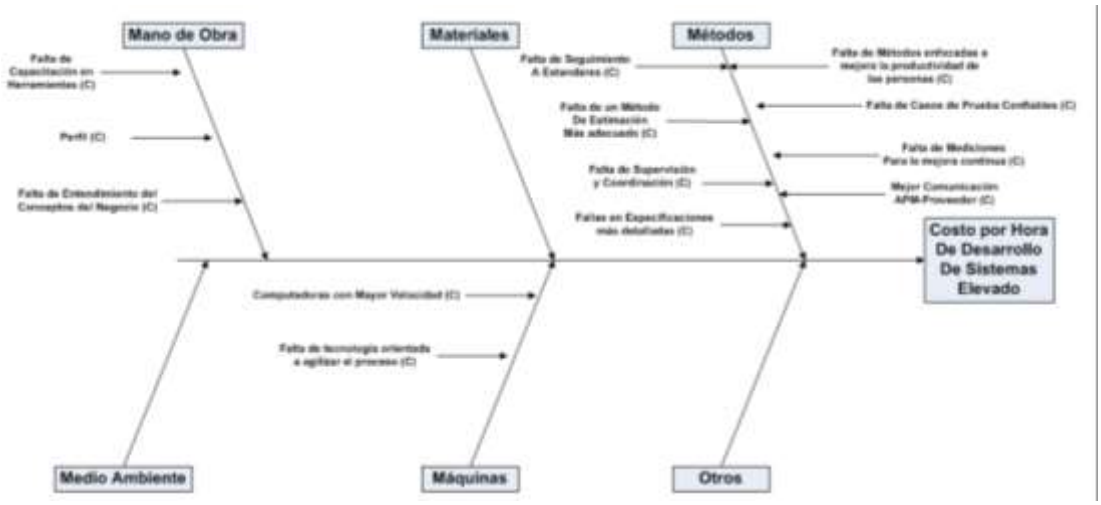


Figura 3.5- Diagrama de Causa y Efecto

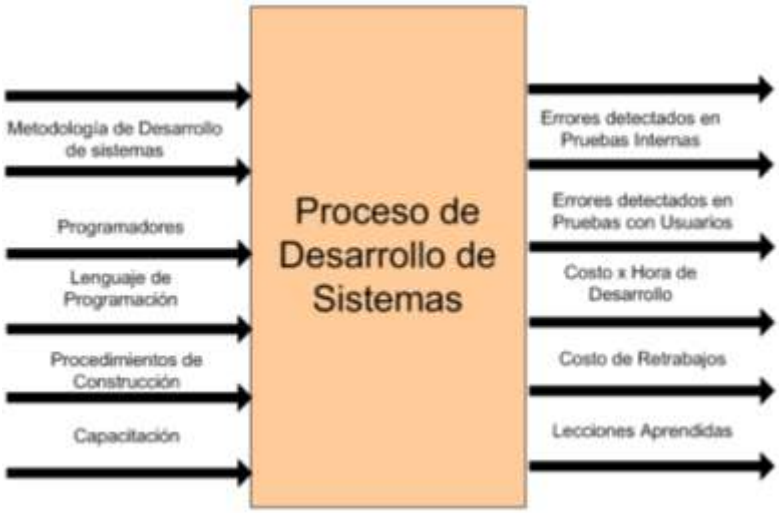


Figura 3.6 Diagrama IPO

3.3.2- FASE DE MEDICION

Esta fase se puede definir en 8 pasos:

- Confirmar / Ajustar la variable Y (COPQ) – Tenemos que medir y evaluar la validez de los datos disponibles, y confirmar nuestra hipótesis inicial en cuanto a la magnitud del problema.
- Defina las metas o los estándares del funcionamiento - aquí establecemos nuestra metas para la mejora en la Y, fijando un objetivo numérico que sea agresiva pero alcanzable.
- Identifique los factores de la segmentación para el plan de la colección de datos - aquí identificamos los factores que dividen naturalmente nuestro proyecto en segmentos Y (por proyecto, el producto, la organización, etc.) y Xs que puedan influenciar la Y, y definir cómo, cuando y donde los mediremos. Generalmente, los factores que influencian los resultados son: *cosas que podemos controlar y otros factores ("ruido") que no podemos controlar.
- Determine las medidas que se utilizará. ¿Son confiables y constantes? ¿Qué tan exactos son los datos?
- Colección de datos - recopilamos los datos necesarios.
- ¿Describa y exhiba la variación en funcionamiento actual - cuáles son los rangos de los valores de X y de Y actualmente?
- ¿Plan de contención - si el proceso actual está en condiciones críticas, qué que cosas podemos arreglar rápido (do it!) para mitigar el problema,

mientras se realizan las actividades necesarias para cortar el problema de raíz.

- Revisión de Ahorros/Beneficios en la Fase De la Medida. Trabajemos con cada uno de estos pasos, continuando nuestro foco en mejorar nuestro proceso.

Confirme el proyecto Y

Para confirmar o refinar la Y que necesitaremos validar las fuentes de información, y la definición inicial de las cosas, por ejemplo, ¿cuál es la definición de "terminación"?, ¿Eso significa la fecha que el cliente aceptó el sistema?, ¿O significa la fecha que el equipo del desarrollo lo declaró construido? Hay a menudo una diferencia grande entre estas fechas, el que realmente cuenta, es la fecha del cliente.

En el caso nuestro, el área de “Centro de Competencia”, lleva el registro de cada proyecto, sus fechas de inicio y fin estimados, las fechas reales, así como una serie adicional de fechas y minutas, que dan la radiografía de cada proyecto.

Defina las metas o los estándares

Después de que recogimos los datos que permitan confirmar la estimación inicial de la tarifa del tiempo (el 66%) esté correcto. Debemos re-validar la viabilidad de nuestra meta. Podemos ser un poco más conservadores y solo ofrecer un ahorro del 50%.

Identifique los factores de la segmentación para la recolección de datos

En los proyectos, la estimación inicial, nace a partir del plan de trabajo que cada líder de proyecto especifica.

Revisar las prácticas para armar un plan de trabajo, esto nos puede ayudar a determinar algunos detalles… ¿cómo podemos dividir nuestros proyectos en segmentos para el análisis? Puede ser que, por ejemplo, decidamos investigar los cuatro factores asociados al planeamiento eficaz, que dicen los libros:

- ¿Duraciones cortas de las tareas?

- ¿Relación clara y definidas de predecesor /sucesor entre tareas?
- ¿Recursos "nivelados"? (No planear, semanas de 80 horas para cada miembro del equipo)
- ¿Definición de entregables por etapa?

Dependiendo del tamaño de nuestra organización y del número de proyectos que hemos terminado en el año pasado, podemos dividir en segmentos o estratificar los datos. Podemos, por ejemplo, separar los datos según el grupo del desarrollo que hizo el trabajo, o podemos estratificar según el tipo del proyecto del software. Generalmente podemos con esta segregación determinarlos diferentes niveles de éxito.

Evalúe la forma de medir sistemáticamente:

En este caso, nuestro "sistema de la medida" está muy relacionado a todas las bitácoras y seguimientos que hace el Administrador de Proyectos, el cual, para los datos iniciales se base en el plan de trabajo de los lideres. El Centro de Competencia Nuestra siguiente actividad es auditar los planes de trabajo que fueron desarrollados en cada proyecto y comprarlos con la base de datos de las bitácoras, para medir la diferencias entre los tiempos y así medir nuestra Y histórica. Si en esta auditoria salen dudas deberemos validar cada caso con los responsables /lideres de proyectos, para clarificar todo.

¿Los planes reflejan el trabajo que fue hecho realmente? ¿Las tareas agregadas o canceladas se reflejan en los planes? ¿La fecha de compromiso se le dio seguimiento, de modo que podamos dejar en claro, exactamente la fecha prometida de la terminación vs la fecha real? ¿Cómo los cambios de los requisitos del proyecto fueron manejados? ¿Si el cliente agregó nuevos requisitos significativos durante el proyecto, la fecha de compromiso fue ajustada apropiadamente para reflejar el cambio en alcance? ¿Cómo se hizo el análisis para hacer tales ajustes? ¿El cliente estuvo de acuerdo que los ajustes en las fechas?

Las respuestas a las preguntas tales como éstas, pueden, conducirnos a hacer varios ajustes a los datos, para hacerlo más constante y validos, antes de iniciar análisis. Datos de entrada equivocadas, generan conclusiones equivocadas.

El mensaje de la filosofía 6sigma, es muy claro "entiende la calidad de los datos antes de hacer conclusiones." Además, cuando tengamos información cualitativa, siempre debemos convertir esta en datos cuantitativos. Hay muchas maneras que poder hacer esto - aquí ofrecemos un acercamiento conveniente a esta situación.

Las cualidades mencionadas arriba son Xs potencial que influyen en el resultado del funcionamiento de los tiempos de entrega - creemos que si hacemos un buen trabajo que satisface estas cualidades seremos más probables entregar el proyecto a tiempo.

Criterios	Medición	Score
Duración	90 al 100% <= 5 dias	5
	80 al 90% <= 5 dias	3
	70 al 80% <= 5 dias	1
	Menor 70% <= 5 dias	0
Tareas ligadas	95 al 100% <= 5 dias	5
	90 al 95% <= 5 dias	3
	85 al 90% <= 5 dias	1
	Menor 85% <= 5 dias	0
Carga Nivelada (hasta 105% carga)	95 al 100% <= 5 dias	5
	90 al 95% <= 5 dias	3
	85 al 90% <= 5 dias	1
	Menor 85% <= 5 dias	0
Entregable por etapa	95 al 100% <= 5 dias	5
	90 al 95% <= 5 dias	3
	85 al 90% <= 5 dias	1
	Menor 85% <= 5 dias	0

Tabla 1: Clasificaciones de Criterios

Nuestra hipótesis entonces es que las actividades con alto numero de días en el Xs (tareas no desglosadas correctamente) serán correlacionados con grados

más altos de Y. Una forma para poder determinar si esta hipótesis está correcta sería instalar un esquema donde anotemos las posibles Xs para cada uno de nuestros proyectos históricos, para ver su comportamiento en los tiempos de entrega.

Recolección de Datos

Nuestro paso siguiente es recoger los datos de X y de Y para cada uno de los proyectos en nuestra línea base. Eso pudo producir un sistema de datos algo como eso:

# Proyecto	% de Retrazo*	Duración	Tareas ligadas	Carga Nivelada	Entregable por etapa	Resultado
1	-55	1	5	3	1	10
2	2	5	5	3	3	16
3	-14	1	1	3	3	8
4	-49	1	3	3	1	8
5	-2	3	5	1	5	14
6	-56	1	3	1	3	8
7	-54	3	1	5	1	10
8	-62	1	3	1	3	8
9	-45	3	1	0	1	5
10	12	5	5	5	3	18
11	-33	1	3	3	3	10
12	-11	3	5	3	1	12
13	-42	3	0	1	0	4
14	-45	3	3	3	3	12
15	-84	1	1	5	1	8
16	11	3	5	5	5	18
17	-75	1	0	1	0	2
18	-82	1	1	0	1	3
19	-8	1	3	3	5	12
20	6	5	3	3	5	16

* fecha comprometido al cliente vs fecha de aceptación del cliente

Tabla 2: Recolección de datos

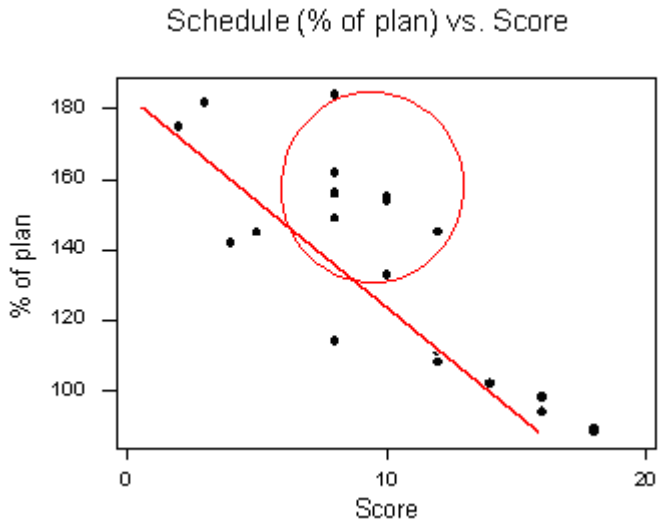


Figura 3.7- Grafia de Tendencia Plan vs Resultado

Al graficar los valores de la matriz, podremos ver la relación entre la Y (la mejora en las fechas de entrega, definido como % del plan vs X (El resultado Total)

Esto nos da como resultado el gráfico colindante que nos demuestra una relación entre nuestro X y Y, según lo sugerido por la línea de la tendencia, se ve un incremento cuando aumentamos el "Resultado" y nuestra Y (% del plan) mejora - los proyectos con planes definidos tienden estar más en tiempo. Pero también notamos que hay algunos proyectos (ésos dentro del círculo) fuera del patrón general - éstos sugieren otros factor(s) que todavía no hemos considerado pudieran afectar el resultado. No tenemos bastante evidencia todavía para asegurar que hay una relación de la causa y del efecto, pero estos datos "aberrantes" requieren un análisis más a detalle.

3.3.3- FASE DE ANALISIS

Es un proceso con 7 pasos:

- Medir la capacidad del proceso existente
- Redefinir las metas
- Identificar los segmentos de datos/los patrones significativos
- Identificar Xs posible.
- Identificar y verificar que Xs crítico

- Validar los beneficios estimados
- Analizar la medida de la revisión

Debemos validar los beneficios cuando examinamos los datos recolectados en la fase anterior, si detectamos cambios en la información, esto significa cambios en nuestra LINEA BASE, esto también implica también redefinición de los benéficos estimados.

Podemos utilizar esta información en para determinar el Cpk (Capacidad del Proceso). Haciendo los cálculos, encontraríamos que el valor que conseguimos es 0.2 (no es recomendado, según los libros estadísticos) Quisiéramos ver un valor de al menos 1, el ideal es 1.5. Tenemos que redefinir las metas de la mejora con estos datos. ¡Si la meta la mantenemos al 90% implica una mejora de mas 200%

Esto puede no ser imposible, pero en una sola intervención es poco probable producir un aumento de esa magnitud, así que debemos fijar una meta que sea más realista y alcanzable en el término cercano, dentro del alcance de nuestro proyecto actual.

Cuando la diferencia es muy grande, debemos ser conservadores en nuestro proyecto, y estar consientes que podemos replicarlo varias veces, en fases subsecuentes, para así llegar a los valores potenciales en el estudio inicial.

Tenemos que identificar los segmentos de datos significativos según lo indicado anteriormente, podríamos dividir los datos en segmentos por el grupo del software o por el tipo del software - si lo hiciéramos así, seguiríamos el patrón del análisis discutido aquí para cada segmento independientemente.

La mayoría de los resultados del proyecto parecen ser relacionados con las mejores cualidades de las prácticas del planeamiento reflejadas en los datos que recogimos, pero hay cinco datos atípicos que parecen ser influenciados por uno o más factores. Identifique Xs posible que nuestras observaciones sobre el patrón en los datos nos conducen al paso siguiente en nuestro análisis.

¿Qué otros factores no identificados pueden explicar los datos atípicos que hemos observado? Uno de los factores que influencian resultados del proyecto del software es el tiempo de entrega, cuando comenzamos con una fecha de entrega poco realista,. Esto nos da una pista que la fecha compromiso debe ser prevista y podría ser uno de los factores que explica los atípicos que observamos.

Podemos investigar esa hipótesis recogiendo más detalle de información sobre cada uno de estos proyectos - es decir, ¿cómo la fecha de entrega prevista vs el benchmark de la industria para los proyectos similares?

	"Y"	Total	Individual Scores (Xs)				
Project	% of plan	Score	Duration	Predecessors	Leveled	Defined	Plan %
1	155	10	1	1	5	3	92
2	98	16	5	3	5	3	105
3	114	8	1	3	1	3	102
4	149	8	1	1	3	3	90
5	102	14	3	5	5	1	110
6	156	8	1	3	3	1	85
7	154	10	3	1	1	5	44
8	162	8	1	3	3	1	55
9	145	5	3	1	1	0	76
10	88	18	5	3	5	5	145
11	133	10	1	3	3	3	82
12	111	12	3	1	5	3	81
13	142	4	3	0	0	1	72
14	145	12	3	3	3	3	81
15	184	8	1	1	1	5	68
16	89	18	3	5	5	5	114
17	175	2	1	0	0	1	56
18	182	3	1	1	1	0	64
19	108	12	1	5	3	3	96
20	94	16	5	5	3	3	98

Tabla 3: Matriz Ponederada

Aunque ahora tenemos una buena lista de factores controlables, podemos profundizar más, para ver si descubrimos mas porqués?.

Surgen los siguientes razonamientos: ¿Por qué no fraccionamos las tareas grandes, en tareas menores a una semana? ¿Por qué no definimos relaciones de predecesor / sucesor entre nuestras tareas? Una de las herramientas 6sigma, es conocida como los "5 porqués", nos apoya a profundizar más sobre un tema, preguntando porqué cinco veces en un esfuerzo de conseguir a la raíz verdadera del problema:

¿Por qué no definimos a precursores? - no sabíamos que era importante

¿- por qué no se proporcionó ningún entrenamiento

¿- por qué no se determino un presupuesto para el entrenamiento?

¿- por qué? - el encargado no pensó que importante este análisis nos dice algo sobre ediciones que necesitaremos tratar para hacer una mejora al proceso.

Al identificar los Xs críticos, podemos determinar cuáles de estos factores son los más influyen a los resultados del funcionamiento de las fechas de entrega. La forma en que podemos hacerlo, es utilizar análisis de regresión múltiple.

Las conclusiones que alcanzamos de análisis de esta muestra indican que el cerca de 78% de la variabilidad que vemos 'es explicado' por tres factores (el Xs)

Duración de la tarea, precursores, y plan %. Por lo tanto nuestro proyecto (probable esto nos indique que demos hacer dos proyectos - uno a ocuparse del proceso de la planeación y uno a ocuparse del problema de estimación de las tareas) se centrará en acciones podemos tomar para mejorar nuestro control sobre estos Xs.

Se debe hacer un ajuste a la ventaja financiera pronosticada para determinar qué mejoras se pueden hacer, estas pueden ser atractivas al negocio, de otro modo se analizan de nuevo los casos del negocio para re-estimar los costos de oportunidad.

Manejando una tarifa promedio de 25usd con un total estimado al año de 16,358 horas de desarrollo nos da un gasto total de $408,950 usd. Si manejamos en promedio 34% de nuestros proyectos tienen retrazo (representa $139,043 usd).

Si del gasto total descartamos las horas extras, nos da un total de $269,907 usd, si ha este monto le cargamos un 10% (nuestra meta, lograr el 90% de entregas a tiempo), el gasto sería de solo $296,897 usd. Con esto tendríamos un ahorro total neto de $112,000 usd.

3.3.4-FASE DE IMPLEMENTACION

Tomando en cuenta que en nuestro proyecto lo estamos enfocando a como mejorar las prácticas de planeación de proyectos, podemos decir que en esta etapa son 6 pasos los más críticos:

1 Identifique los alternativas de la solución
2 Optimizar las variables (“tunning”)
3 Identifique la relación entre X y Y.
4 Redefine, si es necesario la solución
5 Maneje un piloto/prototipo para implementar la solución
6 Verifique Ahorros / Beneficios

Para identificar soluciones alternas:

(1) Todos son responsable del plantear las mejores prácticas,

(2) Asignar a los mejores como Championes de los procesos y validar los draft antes de iniciar o en etapas iniciales

(3) El uso una cierta combinación de estas opciones.

Las relaciones variables de Optimización entre Xs y Ys en este caso no es necesaria - vemos que hay una relación directa entre grados más altos de X y Ys. Vamos a evaluar cada uno de las alternativas de la solución con respecto a criterios aplicables.

La prueba piloto, por ejemplo, anotemos los planes que estos equipos han producido usando el mismo acercamiento aplicado a nuestros datos históricos. Si un método demuestra una diferencia (positiva) significativa seleccionamos muy probablemente esa opción si está razonablemente en línea con la segunda mejor opción con respecto a costos y al plazo de obtención.

3.3.5- FASE DE CONTROL

El propósito de la fase del control es cerciorarse de que nuestras mejoras están sostenidas y reforzadas. Deseamos ser seguros que ponemos en el lugar todo de las acciones que ayudarán al cambio para ser acertado y duradero.

El control se puede describir en 5-pasos:

- Desarrolle el plan del control
- Determine la mejora en la capacidad del proceso.
- Implemente el control
- Cierra el proyecto
- Revisa y verifica ahorros/beneficios finales

También especificaremos qué acción debe ser tomada si las métricas están por debajo de las metas definidas previamente. Debemos definir en esta etapa como se van a estar midiendo los KPI (Por sus signlas en Ingles “Key Performance Indicator”, indicadores de desempeño)

Se debe determinar la capacidad de proceso mejorada y documentar el nuevo nivel del funcionamiento. Después se tienen que cerrar el proyecto incluyendo una transferencia formal de la responsabilidad del equipo de la 6sigma al personal operacional que sostendrá el proceso. Como parte del proceso de cierre el equipo archivará todos los expedientes y datos del proyecto, y publicará las lecciones aprendidas y los éxitos.

En nuestro caso, el cierre del proyecto dejamos un indicador en nuestro “Scord Card”. El cual utilizamos el Microsoft Project Portfolio Server 2007 donde se van a través de semáforos el status de los proyectos en construcción, el semáforo de los proyectos que están fuera de presupuesto y el semáforo de “Schedule”, donde se la da seguimiento a fecha de liberación (fecha del visto bueno del usuario, líder de requerimiento vs. fecha de compromiso originales, además vs. fecha última negociación.

4

CONCLUSIONES Y RECOMENDACIONES

Las Fábricas de Software están basadas en la convergencia de ideas claves en rehusó sistemático, desarrollo dirigido por modelos, desarrollo por ensamblado y marcos de proceso. Varias de estas ideas no son nuevas. Lo novedoso es su síntesis en un enfoque integrado que permita a las organizaciones con experiencia en dominios implementar el patrón de Fábrica de Software, construyendo lenguajes, patrones, marcos y herramientas para automatizar el desarrollo en dominios más particulares.

Una Fábrica de Software es un ambiente de desarrollo configurado para soportar el desarrollo rápido de un tipo específico de aplicación. Mientras que las Fábricas de Software son realmente el próximo paso lógico en la continua evolución de los métodos y prácticas de desarrollo de software, estas cambiarán la industria del software mediante la introducción de patrones de industrialización.

Por otro lado la implementación de los procesos Six Sigma requiere un cambio fundamental en la manera que la mayoría de la gente del software realizan sus trabajos. Para realizar estos aumentos de productividad es esencial reconocer que un cambio cultural significativo debe ocurrir.

La realización de este cambio cultural es lograda proporcionando el entrenamiento de la 6sigma para todos los gerentes, administradores de proyectos, y desarrolladores, con una mezcla de Champions y el equipo de Belts.

El existo de cada proyecto, se centra primero en una visión muy clara de la meta, en un plan de trabajo detallado al nivel mas bajo y en un seguimiento diario del plan, con revisiones semanales. Con planes de acción efectivo para cada evento que empiece a notar retrasos.

En las consideraciones futuras de este estudio, será muy importante medir, los costos de los desarrollos vs. lo tiempos que se tiene que dar en los proyectos. Mientras las organizaciones son estables, es fácil darle seguimiento a los costos bajo tiempos estándares, pero en los procesos de globalización, los costos dejan de tener prioridad, cuando los tiempos por si solos, generan millones y millones de dólares de ahorros.

Es necesario en un futuro analizar los impactos que cada metodología de desarrollo tiene, al momento de las grandes integraciones. Será muy interesante validar cual de las metodologías se mantiene luego de estos procesos.

REFERENCIAS BIBLOGRAFICAS

[1] Mark j. Kiemele, Stephen R. Schimidt, Ronald J. Berdine, “Basic Statistics”, Air Academy Press, 2004. USA

[2] Mark j. Kiemele, Stephen R. Schimidt, Ronald J. Berdine, “Knowledge Managment”, Air Academy Press, 2004. USA

[3] Six Sigma Black Belt Volumen 1 y 2
Air Academy Press, 2004. USA

[4] Jack Grenfield, “Fábrica de Sottware: Ensablando Aplicaciones con Patrones, Modelos, Marcos y Herramientas”, MSDN, Noviembre 2004. .USA

[5] Jack Grenfield, “The Case for Software Factories”
The Architecture Journal, Microsoft . USA

[6] Jack Grenfield and Mauro Regio, “Design and Implementing a Software Factory” , The Architecture Journal, Microsoft . USA

[7] Tome Fuller, “A Foundation for the Pillars of Software Factories”
The Architecture Journal, Microsoft . USA

[8] Marcel de Vries, “Measuring Success with Software Factories”
The Architecture Journal, Microsoft . USA

[9] Steve Eadie, “A GSI’s Perspective of Software Factories”
The Architecture Journal, Microsoft . USA

[10] Sten and Per Sundblad, “Business Improvement Through Better Software Architecture”, The Architecture Journal, Microsoft . USA

[11] Lewis Curtis and George Cerbone, “The Perspective Based Architecture Method”, The Architecture Journal, Microsoft . USA

[12] Julio C. Terreros, “Desarrollo de Software basado en Componentes”
MSDN, Santiago de Chile

RESUMEN DE GRAFICAS

Figura 2.1: Curvas de Innovación
Figura 2.2: Elemento re-usable
Figura 2.3: Economías de Escala
Figura 2.4. Economías de Alcance
Figura 2.5: Evolución de las Ciudades
Figura 2.6: Evolución de las casas de Software
Figura 2.7: Fabricas y Edificios
Figura 2.8: Aplicaciones
Figura 2.9: Transportes
Figura 2.10: Comunicaciones
Figura 2.11: Bines Fabricados
Figura 2.12: Datos Estructurados
Figura 2.13: Ensamblados Fabricados
Figura 2.14: Empresas Virtuales
Figura 2.15: Comercialización y Distribución
Figura 2.16: Procesos de Negocio
Figura 3.1: Proyectos ligados al Plan Estratégico del Negocio
Figura 3.2 Reducir a la Máxima Expresión la Y del Proyecto
Figura 3.3: Grafica de Pareto
Figura 3.4: Flujo de Proceso
Figura 3.5: Diagrama de Causa y Efecto
Figura 3.6: Grafica de IPO
Figura 3.7- Grafia de Tendencia Plan vs Resultado

RESUMEN DE TABLAS

Tabla 1: Clasificaciones de Criterios
Tabla 2: Recolección de datos
Tabla 3: Matriz Ponederada

GLOSARIO DE TERMINOS

Black Belt:	Líder de proyectos, con alto nivel de conocimiento del negocio y con un fuerte perfil estadístico.
Champion:	Alto nivel de conocimiento de entorno de trabajo, conocimientos estadísticos básicos. Guía para belts par orientar los esfuerzos análisis en proyectos 6sigma
CMM:	Capability Maturity Model. Modelo de Madurez de Capacidades, la cual mide la calidad de las empresas desarrolladoras de software sobre un estándar internacional.
COM:	Component Object Model Technologies
COPQ:	Costo de Pobre Calidad. En otras palabras es lo que la empresa pierde o dejar de ganar, por no tener un seguimiento a alguna variable del negocio
CORBA:	Common Object Request Broker Architecture — arquitectura común de intermediarios en peticiones a objetos
Cpk:	Capacidad del Proceso. Calculo manejado en SixSigma, para medir la capacidad de procesamiento de un modelo.
DMAIC:	Metodología de trabajo que consiste en 5 etapas: Definición, Medición, Análisis, Implementación y Control.
Green Belt:	Empleado de alto nivel, con conocimientos estadísticos y puede en tiempos parciales apoyar con proyectos para mejorar variables de su entorno laboral
IPO:	Input Process Output, es un tipo de grafica donde se muestran las entradas a un proceso y las salidas de este.
KPA:	Áreas Clave de Proceso
KPI's:	Key Performance Indicator. Indicador clave de desempeño, técnicas de medición desarrolladas por Ronald Daniel & Jack Rockart
Línea Base:	Es la fotografía del como esta la variable a controlar en proyecto 6sigma, esta fotografía generalmente se mide con los resultados de al menos 6meses de historia, máximo 12 meses.
Master Black Belt:	Responsible de Black & Greens Belt en una organización. Alto conocimiento de herramienta estadística y alto perfil de Servicio.

RDBMS:	Relational Data Base Management System. Sistema administrador de bases de datos relacionales
ROI:	Return on Investment
SDLC:	Metodologías de Desarrollo de Software
SOA:	Service Oriented Architecture (SOA)
UML:	Unified Modeling Len. Lenguaje Unificado de Modelado

AUTOBIOGRAFIA

Nací en Monterrey NL, mis estudios básicos e intermedios fue en colegios Maristas, luego ingrese a la Universidad Autónoma de Nuevo León, orgullosamente egresado de la Facultad de Ingeniero Mecánica y Eléctrica donde cruce la carrera de Ingeniero en Administrador de Sistemas.

Luego de egresar de la universidad tuve la oportunidad de estudiar un año el Marrymount Collage en Nueva York. Al regresar de USA, inicie mi vida profesional la cual se ha basado en dos grandes y bellas etapas.

La primera en Cervecería Cuauhtemoc, en donde esos dos años, fueron de gran aprendizaje, donde trabajé como programador de sistemas. Mi segunda casa fue Grupo IMSA, donde inicie como analista de sistemas, luego me dieron la gran oportunidad de entrar al fascinante mundo de las finanzas y planeación estratégica, donde por tres años desarrolle los planes de negocio de APM (la división donde se fabrican los rollos de acero), el seguimiento a los bonos de compensación del comité directivos, así como el seguimiento a las principales variables del mercado local, nacional e internacional.

Luego con el “boom” del Internet y del eBusiness, regrese al área de sistemas para ser el encargado de montar toda la infraestructura tecnológica, para soportar todas las estrategias eCommerce, además de ser el responsable del Portal y de los servicios en línea para los Clientes.

El reto mas grande hasta el momento ha sido la migración de nuestro ERP, el cual por mas de tres años fui responsable la migración de todo el proceso comercial. En el año 2005, me ofrecen la oportunidad de entrar a los procesos de mejora continua, como Black Belt. Durante los siguientes dos años trabajo en proyectos de reducción de costo y mejoras en la rentabilidad del negocio.

A final 2006 con la venta de Grupo Imsa de la familia Clariond a la familia Canales, soy coordinador a nivel Grupo Imsa de la planeación tecnológica en la integración de las Imsas (USA, México y Centroamérica)

A mediados del 2007 la empresa nuevamente es vendida, a un grupo de inversionistas argentinos, durante este proceso estoy encargado de la integración del los sistemas comerciales a la plataforma estandarizada que maneja Ternium.