



SOFTWARE GURU CONOCIMIENTO EN PRÁCTICA

- Estimación de proyectos
- Aseguramiento de calidad
- Capas en J2EE

Año 01 No.01 • Enero-Febrero 2005

ENTREVISTA:
Martín Méndez
Director de TI en GNP



MUCHOS CAMINOS, UN DESTINO

Conociendo los diferentes modelos
de procesos de software

CASO DE ESTUDIO:
**Aplicando PSP
y TSP**

A Fondo
GXportal 4

Ahora pensar en grande es tu elección.

Con Visual Studio® .NET 2003 escribes menos código que con Microsoft® Visual® Basic® 6.0, logrando convertir esa gran idea en realidad, más rápido de lo que jamás imaginaste. El nuevo Ambiente Integrado de Desarrollo (IDE) concentra tus habilidades existentes en una herramienta que te brinda más código robusto, así como mejoras en la función IntelliSense® y una implantación más completa. Serás más productivo y estarás listo para trabajar con tus ideas más revolucionarias.

Para descubrir cómo Visual Studio .NET 2003 puede ayudarte a convertir rápidamente tus grandes ideas en realidad, visita:
<http://msdn.microsoft.com/mexico> o llama al centro de atención a clientes Microsoft: 5267 2120 o al: 01800 849 9998



Visual Studio

Tu potencial. Nuestra pasión.™

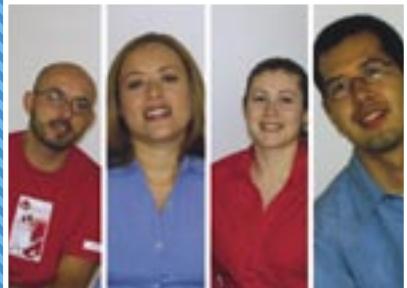
Microsoft



Microsoft Corporation, todos los derechos reservados. Microsoft, el logotipo de Microsoft y "Tu potencial. Nuestra pasión." son marcas registradas de Microsoft Corporation en Estados Unidos y/o otros países.

■ A >

EDITORIAL



Bienvenidos al número de Enero-Febrero 2005 de Software Guru. En esta ocasión hemos escogido a los procesos de software como tópico central. Este es un tema de gran interés en la industria de software mundial, pero que tiene especial importancia en México y el resto de América Latina.

La falta de madurez en capacidad de procesos es uno de los principales puntos débiles de la industria de software en nuestros países. Un estudio recientemente realizado por la Secretaría de Economía en México, mostró que el nivel de madurez de procesos promedio en la industria de software mexicana es de 0.9 (en base a la escala dada por ISO 15504, que va de 0 a 5). Esta estadística refleja nuestra falta de competitividad en este rubro, donde tenemos cerca de diez años de retraso respecto a los líderes de la industria mundial.

Afortunadamente estamos conscientes de esta situación y existen diversos esfuerzos para mejorarla. Justamente hace un par de meses se llevó a cabo la primera edición de la conferencia SEPG (Software Engineering Process Group) Latinoamérica, donde ingenieros de procesos de diversos países intercambiaron experiencias y promovieron mejores prácticas para desarrollar la capacidad de procesos en esta región. Poco a poco las empresas demuestran un mayor interés en este tema y asignan recursos para mejorar su madurez de procesos.

Sin embargo, este tema presenta gran complejidad debido a lo abstracto que es y la diversidad de posibilidades existentes. La gran mayoría de las empresas no sabe qué camino tomar hacia la madurez de procesos: ¿CMM? ¿CMMI? ¿ISO 15504? ¿MoProSoft? ¿RUP? ¿TSP? ¿PSP? ¿XP? Es muy fácil perderse entre tantas opciones. Es por ello que en nuestro artículo central nos enfocamos en explicar estas opciones y cómo se relacionan entre sí, para ayudar a las organizaciones a entenderlas y poder elegir el camino que más les convenga.

Equipo Editorial

P.D. Agradecemos el enorme apoyo que hemos recibido de parte de todos ustedes. Es alentador recibir sus comentarios, porras y opiniones para continuar este esfuerzo y generar un producto de gran valor para todos. Les reiteramos la invitación para que participen a través de los foros en www.softwareguru.com.mx o por correo electrónico en editorial@softwareguru.com.mx

■ DIRECTORIO

Edición Ejecutiva

Pedro Galván

Coordinación Editorial

Mara Ruvalcaba

Edición y Producción

Edgardo Domínguez

Dirección de Arte

Oscar Sámano

Consejo Editorial

Francisco Camargo, Guillermo Rodríguez y Raúl Trejo, ITESM Campus Estado de México; Hanna Oktaba, UNAM-AMCIS; Luis Cuellar, Softtek.

Colaboradores

Jorge Palacios, Antonio Reyes, Omar Ruvalcaba, Francisco López Lira, Paulina Olivares, Raul Achoy, Luz Ma. Luckie, Isabel Chilopa, Artemio Mendoza, Ricardo Vidrio, Ricardo Domínguez, Rodrigo García, Mariana Pérez-Vargas, Elizabeth Almeraz, Ramés Rodríguez, Ariel García.

Ventas

David González

Marketing

Natalia Sánchez

Webmaster

www.aguilahosting.com

Agradecimientos

Hilda López, Martha Kondo, AMCIS, Gastón Sánchez, Martín Méndez.

Contacto

info@softwareguru.com.mx

+52 55 5239 5502

Software Guru es una publicación editada por Brainworx, S.A. de C.V. Parque de la Malinche No. 6, Col. El Parque, Naucalpan, Estado de México. Ejemplar de cortesía. Derechos reservados por Brainworx, S.A. de C.V. Certificado de licitud de Derechos de Autor en trámite. Certificado de licitud de título e ISSN en trámite. Certificado de licitud de contenido en trámite. Prohibida la reproducción total o parcial sin previa autorización escrita de los editores. Se terminó de imprimir en el mes de enero de 2005 en Litográfica Roma S.A. de C.V. Todos los artículos son responsabilidad de sus propios autores y no necesariamente reflejan el punto de vista de la editorial.

contenido ene-feb 2005

número 01



EN PORTADA

Procesos de Software: Guía para el Viajero

Es un hecho que los procesos son la base para desarrollar software de calidad de manera predecible y repetible. Sin embargo, con tantas opciones, normas y modelos, es difícil saber cuál es el camino adecuado. Este artículo es una guía para ayudarlo a elegir el mejor camino en este viaje tan importante.

20

Productos

LO QUE VIENE 10

Apache Beehive y Amazon Simple Queue Service

A FONDO 12

GXportal 4

HERRAMIENTAS 14

Gestión de Portafolio de Proyectos

TIPS 16

El Síndrome ERwin

Columnas

Tejiendo Nuestra Red 06

por Hanna Oktaba

Mejora Continua 08

por Luis Cuellar

Cátedra y Más 41

por Raúl Trejo

Prácticas

ADMINISTRACIÓN DE PROYECTOS 30 Estimación de Proyectos

La estimación de tiempos y esfuerzos es un tema de gran interés para todo administrador de proyectos de software. En este artículo Rodrigo García nos explica las bases de la estimación, y aborda dos métodos para realizar esta difícil e importante tarea.

ASEGURAMIENTO DE CALIDAD 34 ¿SQA? ¡Sálvese quien pueda!

¿Qué es el aseguramiento de calidad? ¿En qué consiste el rol de SQA? ¿Cuáles son sus actividades y para qué las realiza? Mariana Pérez-Vargas y Elizabeth Almeraz nos contestan todas estas preguntas a través de este interesante artículo.

ARQUITECTURA 38 Capas Conceptuales para Sistemas J2EE

J2EE nos provee una plataforma poderosa para desarrollar sistemas empresariales. Sin embargo, es fácil perderse en ella. Ramés Rodríguez comparte con nosotros una estrategia basada en cinco capas conceptuales para organizar la arquitectura de un sistema basado en esta plataforma.

SG®



ENTREVISTA 18

Martín Méndez, Director de Tecnología de Información en GNP.



CASO DE ESTUDIO 26

QuarkSoft nos guía a través de la aplicación de PSP y TSP para rescatar un proyecto.

En Cada Número

Noticias y Eventos	04
Fundamentos	42
Tecnología	44
Biblioteca	46
Carrera	48

Noticias

Oracle



PARTNER
PROGRAM

CONSOLIDA RELACION CON ISVs

Oracle tiene en curso un programa para consolidar su relación con las empresas desarrolladoras independientes de software (Independent Software Vendors - ISVs), a fin de proveer a sus clientes de las soluciones tecnológicas que les permitan enfrentar los retos de la actualidad. Gerardo Flores, Gerente de ISV en Oracle México, comenta: "Los ISVs son socios de negocios críticos no sólo porque fortalecen el portafolio de soluciones verticales de Oracle, sino que fortalecen el modelo de alianzas —a través de una especialización respecto a las necesidades del negocio del cliente—. A nueve meses de implementar el modelo mencionado en México, se alcanzaron doce nuevas aplicaciones de industria que fortalecen el negocio y la operación de las organizaciones de nuestro país e incluso en muchos casos, del extranjero."

MoProSoft

PROYECTO DE PRUEBAS CONTROLADAS

El modelo de procesos para la industria de software (MoProSoft) y el método de evaluación de capacidad de procesos (EvalProSoft), actualmente se encuentran en pruebas controladas en cuatro empresas. Los objetivos de este proyecto son:

1. Probar que MoProSoft eleva la capacidad de procesos en las PyMEs de software donde es implantado.

2. Probar que EvalProSoft es aplicable para evaluar la capacidad de los procesos de una organización.
3. Establecer el costo, tiempo y esfuerzo necesarios para alcanzar un nivel de capacidad específico.



Este proyecto comenzó en julio del 2004 y se espera que termine en febrero del 2005. Durante este tiempo también se está capacitando a ocho personas del interior del país como consultores en formación.

ProSoft

APOYO PARA CLUSTERS

Los clusters de Aguascalientes, Baja California, Guanajuato, Jalisco, Morelos, Nuevo León, Puebla, Sinaloa, Sonora y Yucatán, recibieron apoyos del fondo ProSoft para diversos proyectos durante el cuarto trimestre del 2004. Los rubros más apoyados fueron: incubación de empresas, equipamiento de centros de investigación, adopción de metodologías de calidad. El pasado 30 de noviembre se llevó a cabo una reunión con la participación de Microsoft, la AMITI y los representantes de los clusters de Coahuila, Sinaloa y Yucatán para intercambiar experiencias sobre empresas integradoras de software.

Eventos Enero-Marzo 2005

SEPG LA



ÉXITO DE LA PRIMERA CONFERENCIA DEL SOFTWARE ENGINEERING PROCESS GROUP

La primera edición del SEPG LA fue celebrada con éxito del 8 al 10 de noviembre en Guadalajara, Jalisco. La conferencia atrajo más de 240 profesionales de todo Latinoamérica, pertenecientes a los principales sectores industriales implicados en el desarrollo de sistemas de software. El notable resultado subraya que no es sólo una conferencia necesaria y significativa entre los profesionales de software (98% de las evaluaciones dan un resultado de excelente); sino que el contenido es de gran importancia para el desarrollo de todos los sectores industriales de Latinoamérica que dependen de la calidad del software para sobrevivir y permanecer competitivos.

TI@mericas



CUMBRE INTERNACIONAL DE LA INDUSTRIA DEL SOFTWARE

Del 27 al 29 de octubre, la CANIETI llevó a cabo la segunda edición de su evento TI@mericas, en esta ocasión uniéndose esfuerzos con la Secretaría de Economía para apoyar a las empresas desarrolladoras de software. El objetivo fue promover el potencial de las empresas desarrolladoras de software en México, creando nichos de oportunidad para los empresarios y generar credibilidad al mercado interno y de exportación.

El evento tuvo sede durante dos días en Tijuana, Baja California y el tercer día en San Diego, California, con la participación de más de 700 asistentes de 19 diferentes Estados de la República.

7-10 Enero 2005 TALLER SOFTWARE.NET.MX – MICROSOFT Y PROSOFT

Universidad de Monterrey
Monterrey, Nuevo León
Info: www.microsoft.com/spanish/msdn
Tel: 01800-849-9959
e-mail: desarrollador@microsoft.com.mx

26-28 Enero 2005 INTRODUCTION TO CMMI – POR GIUSEPPE MAGNANI

Oficinas de Avantare
Ciudad de México
Info: www.avantare.com
Tel: 52 (55) 5544-3321
e-mail: informacion@avantare.com

8-11 Febrero 2005 EXPO COMM MÉXICO 2005

Centro Banamex
Ciudad de México
Info: www.expocomm.com.mx
Tel: 1087-1664, 01800-000-2322
e-mail: conferencias@ejkrause.com

17 Febrero y 17 Marzo 2005 SEMINARIO LLEGANDO RÁPIDAMENTE AL CMMI 2 Y 3 A TRAVÉS DE RUP

Itera Cd. de México – Itera Monterrey
17 de Febrero, Cd. de México
17 de Marzo, Monterrey
Info: www.ITERA.COM.MX
Tel: 52 (55) 5281-7670
e-mail: contactsalescenter@ITERA.COM.MX

2-3 Marzo 2005 SUN TECH DAYS

World Trade Center
Ciudad de México
Info: www.suntechdays.com.mx
e-mail: info@suntechdays.com.mx

7-10 Marzo 2005 SEPG 2005 - DELIVER WINNING PRODUCTS THROUGH PROCESS IMPROVEMENT

Washington State Convention & Trade Center
Seattle, Washington
Info: www.sei.cmu.edu/sepg
e-mail: customer-relations@sei.cmu.edu

14-18 Marzo 2005 SD WEST 2005 – DEVELOP AN ADVANTAGE

Santa Clara Convention Center
Santa Clara, California
Info: www.sdexpo.com
e-mail: rrobles@cmp.com

Investigación de Procesos

PRIMEROS TEMAS ABIERTOS

En la entrega anterior de esta columna les platiqué sobre la creación del International Process Research Consortium (IPRC), cuyo objetivo es definir la ruta para investigación de procesos de software para los próximos 5-10 años. En el primer taller se presentaron propuestas individuales y posteriormente se formaron cinco espacios de sesión abierta sobre los temas identificados de interés común. Los participantes nos repartimos libremente entre estos espacios para generar ideas. A continuación les presento un resumen sobre los temas abordados, el problema que intentan atacar y los posibles temas por investigar.

Verificación y Validación de Procesos

o Problema: no queda claro qué significa la verificación o validación de procesos.

o Temas por investigar:

- Cómo demostrar que un proceso se está siguiendo (*process fidelity*)
- Cómo demostrar que un proceso hace lo que queremos que haga (*process suitability*)
- ¿Cuáles son los requerimientos de un proceso?
- ¿Cuál es la estructura de un proceso (componentes y elementos)?
- Ciclo de vida para el desarrollo de procesos

Repository de Procesos de Software

o Problema: falta de ejemplos y casos de éxito para usar en la práctica, educación e investigación de Ingeniería de Software.

o Temas por investigar:

- Desarrollar un repositorio global con diferente tipo de información que ayude a implementar prácticas y que resguarde evidencias de casos exitosos.

Desarrollo distribuido (“outsourcing”)

o Problema: no se conocen los criterios para la toma de decisión con respecto a *outsourcing*. No se sabe cómo coordinar el *outsourcing* con el negocio.

o Temas por investigar:

- Identificar la variedad de tipos de desarrollos distribuidos
- Identificar y analizar las razones de *outsourcing* (competencia, costo, efectividad)
- Analizar qué partes de negocio conviene dejar en *outsourcing*
- Procesos comunes o no entre el negocio y la organización que proporciona *outsourcing*
- Mecanismos de coordinación e integración

PyME o SME (Small & Middle Enterprise)

o Problemas

- Cómo se define (¿desde 5 personas hasta cuantas?)
- Una persona lleva varios roles
- Un producto contiene varios productos de trabajo
- Tiene pocos recursos
- Ve a los procesos como su muerte
- Ve a las evaluaciones como exámenes

o Temas por investigar:

- Entender qué significa la mejora de procesos para PyMEs
- Proporcionar un conjunto mínimo de procesos que sean fáciles de utilizar y evolucionar
- Necesitan ser parcialmente subsidiadas por gobierno y/o integradoras de sistemas
- Creación de “clusters”
- Casos de estudio específicos para PyMEs

Factores Humanos

o Problema: no está suficientemente entendido el valor de los recursos humanos en el desarrollo de software.

o Temas por investigar:

- Relación de factores humanos con la productividad
- Disposición de recursos humanos a cambiar
- Cómo afecta la distribución geográfica, la educación y la cultura (de país, de la organización, del equipo)
- Cómo afecta la política pública (reactiva o proactiva)

Estos problemas y temas de investigación no nos resultan ajenos y reflejan el estado incipiente de esta área a nivel mundial. Yo me uní al grupo de Verificación y Validación de Procesos, coordinado por Terry Rout, un australiano responsable por la publicación de la ISO/IEC 15504. En las propuestas de temas de investigación que allá se mencionaron, tenemos algo adelantado en MoProSoft con respecto a la estructura de procesos, sus componentes y elementos definidos a través de un patrón.

Como curiosidad les puedo contar que el desarrollo distribuido fue la mayor preocupación de los alemanes, y los representantes de la India se concentraron en factores humanos. Los miembros del SEI fueron los más interesados en las PyMEs. La razón es, como ellos mismos lo explicaron, que las grandes empresas subcontratan a las pequeñas y, si el único marco de procesos disponible es CMMI, están en problemas. Esta preocupación del SEI me dio aún mayor confianza de que en México vamos por buen camino con MoProSoft. Sin embargo, el tema que une a todos es el del repositorio de procesos de software, ya que éste podría tener un impacto inmediato en la industria de software mundial. ¿Por qué no empezamos a hacer algo al respecto en México?

- **Hanna Oktaba**



La Dra. Hanna Oktaba es profesora en la Facultad de Ciencias de la UNAM. Es fundadora y vicepresidenta de la Asociación Mexicana para la Calidad en la Ingeniería de Software. Actualmente dirige el proyecto para la creación de una norma mexicana para la industria de software.

Podrías tener mejores resultados...



Rational. software

Optimización de Procesos de Ingeniería de Software y TI

- Rational Unified Process
- CMMI
- CMMI by RUP
- ITIL
- Instalación y Configuración de herramientas Rational
- Optimización del área de Ingeniería de Software
- Alineación de objetivos de negocio con los servicios de TI
- Consultoría y capacitación en procesos de TI

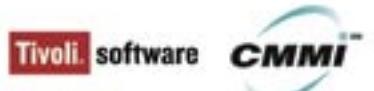
Adoptar un proceso de ingeniería de software confiable, eficiente y basado en las mejores prácticas de la industria • Asegurar el desarrollo exitoso y con calidad de proyectos de software • Mejorar la productividad de las operaciones de TI • Elevar el nivel de satisfacción de los servicios de TI • Facilitar la administración de la infraestructura de TI • Reducir el tiempo de implementación de procesos de mejora • Obtener visibilidad clara del desempeño de los procesos y proyectos de TI • Facilitar la toma de decisiones basada en información cuantitativa.

itera
it & development process

INSCRÍBETE
al Seminario Gratuito
**'Llegando Rápidamente al
CMMI 2 y 3 a través de RUP'**

Méjico 17 de Febrero, 2005
Monterrey 17 de Marzo, 2005

contactsalescenter@itera.com.mx
(55) 5281 7670



www.itera.com.mx

Centro de Formación

- Rational University
- CMMI
- ITIL

El Proceso a Seguir

¿Y PARA QUÉ PROCESOS?



Luis R. Cuellar es Director de Calidad a nivel mundial de Softtek Information Services. Luis es reconocido por la American Society for Quality (ASQ) como Certified Quality Manager, Certified Software Engineer, y Six Sigma Black Belt. En los últimos cinco años ha estado a cargo de la definición e implantación de la estrategia para CMM5 y Six Sigma a través de las diferentes áreas del centro de desarrollo de Softtek.

Hola!, y bienvenidos a este espacio en nuestra revista. Como saben, la industria de software está más competida que nunca, nuevas compañías con nuevos productos y servicios surgen a diario, haciendo más complejo el mercado. Adicionalmente la industria se encuentra saturada de teorías, modelos y herramientas que prometen aumentar la calidad y productividad de las empresas, pero que a la hora de implementar, generan resultados muy por debajo del costo de la inversión. Esta columna está enfocada a generar y compartir ideas prácticas para aplicar estos temas en las pequeñas y medianas empresas. Abordaremos temas como CMM, CMMI, Six Sigma y otros de una forma práctica que nos ayude a entender qué hay detrás y cómo podemos, en forma sencilla, lograr los beneficios prometidos.

Para iniciar esta columna y durante las siguientes dos publicaciones, me gustaría iniciar con el elemento más básico de toda teoría de calidad y mejora: el proceso. El proceso es la base de todo modelo y metodología de mejora, y aunque parece sencillo de entender y definir, la definición incorrecta de los procesos de la organización es la principal causa de fracaso de las iniciativas de mejora.

Organización orientada a procesos

Comencemos por acordar algunas definiciones, por ejemplo, ¿qué es un proceso? Según el diccionario, “conjunto de operaciones lógicas y aritméticas ordenadas, cuyo propósito es la obtención de resultados determinados”. En otras palabras, un proceso está compuesto por **actividades** con una secuencia de ejecución, **responsables** para cada actividad, **formatos**, y **procedimientos** para llevar a cabo una tarea. Bajo esta definición, se podría decir que las actividades que sigo para llegar a tiempo a mi trabajo por las mañanas cumple con la definición de proceso, que abarca desde las actividades y procedimiento que ocupo para levantarme a cierta hora, hasta las actividades que llevo acabo para transportarme al trabajo. Otra definición necesaria para iniciar esta discusión es la **efectividad**, definida como la “capacidad para producir el efecto deseado”, lo que quiere decir que el proceso de llegar a mi trabajo en la mañana es efectivo en la medida en la que realmente llegue a mi trabajo a la hora deseada. Finalmente, tenemos la **eficiencia**, que es la “capacidad para lograr un fin empleando los mejores medios posibles”. Así que mi proceso es eficiente en base a la consistencia con la cual se ejecuta en forma eficaz.

Para que mi organización funcione correctamente, mis procesos deben estar definidos y deben llevarse a cabo en forma efectiva y eficiente. Esto con la finalidad de crear repetibilidad, facilitar la comunicación y depositar el conocimiento de la organización.

Repetir éxitos de la organización

La idea de tener procesos definidos viene del modelo de manufactura y las cadenas de producción, donde las actividades se definen en procesos repetibles para ayudar a cada persona de la cadena productiva a volverse expertos en su trabajo específico. Este concepto se puede transportar a la cadena de servicio, y aunque existen fuertes diferencias en los niveles de detalle y tipos de proceso, el objetivo es el mismo: la repetibilidad, poder garantizar que los éxitos de hoy se puedan repetir mañana. Los procesos son la base para asegurar que exista un método probado para continuamente proveer un producto o servicio en forma eficaz y eficiente.

Facilitar la comunicación

Una gran cantidad de problemas en los proyectos medianos de software se deben a fallas en la comunicación. Desde el no definir claramente el alcance del proyecto con el cliente, hasta el hecho de que quienes estimaron el proyecto olvidaron pasar los supuestos que utilizaron a la persona que ahora lleva la problemática de entregarlo a tiempo y en presupuesto. Contar con un proceso predefinido nos ayuda a que todos los partícipes del proyecto conozcan su responsabilidad y sepan cuál es la información mínima que deben comunicar. El proceso nos da las pautas que todos vamos a seguir durante el proyecto para trabajar como un equipo.

Depositar el conocimiento de la organización

Finalmente, el proceso también nos sirve como repositorio o marco de referencia para guardar el conocimiento que la organización adquiere a través del tiempo. Las teorías de administración del conocimiento se basan en la idea de que la organización debe generar repositorios de consolidación del conocimiento de todos los individuos, para que todos puedan aprender de los errores de todos. El proceso puede funcionar como un repositorio específico de ejemplos, checklists, templates y demás artefactos, que representan los logros, mejoras y lecciones aprendidas a través de toda la organización. El proceso que sigue la organización es el marco perfecto para identificar qué es lo que la organización necesita aprender y cómo se referencia con respecto a otros aprendizajes antes mencionados.

Esto nos da una base para la conversación de la siguiente ocasión en donde profundizaremos en los problemas más comunes para definir procesos y porqué pueden llevar al fracaso total de una estrategia de mejora.

- Luis Cuellar

La consolidación de MGE en México



MGE es una empresa líder a nivel mundial, proveedora de soluciones de energía de calidad, diseñadas para incrementar la disponibilidad y continuidad en procesos y aplicaciones de misión crítica, ya sea desde una computadora hasta los grandes centros de cómputo o plantas industriales. “En aplicaciones especiales y en equipos de grandes capacidades MGE es el proveedor mundial número 1”.

Trabajar sin interrupciones

Con más de 40 años de experiencia y una amplia gama de servicios y productos, se han distinguido siempre por brindar soluciones adaptadas a las necesidades de empresas que requieran de equipos tecnológicamente probados para el cuidado y respaldo de sus computadoras y procesos productivos en línea. Los UPS's son equipos de alto rendimiento para el ahorro de energía que permiten contar con soluciones de energía estable y permanente, que garantizan la continuidad operativa en procesos y aplicaciones de misión crítica.

Los sistemas de MGE incluyen fuentes de energía continua, interruptores de transferencia estática, unidades de distribución de energía, filtros de armónicos, supresores de picos, inversores y acondicionadores de energía.

MGE tiene programas de mantenimiento preventivo y correctivo de servicio para la gama de productos que comercializa, desarrollados por el Ing. David Rubio, director de servicio. Para ello cuentan con una plantilla de ingenieros de campo capacitados en estándares internacionales para ofrecer servicio los 365 días, las 24 horas, con un tiempo de respuesta inmediata.

Consolidación de MGE en México

MGE tiene tres años de operar con esta denominación en México y ha logrado alcanzar una cuota de mercado en el primer año, por arriba del 7% de sus más cercanos competidores, gracias a su estrategia de servicios, lo cual les ha permitido alcanzar cuentas corporativas importantes en los sectores médico, financiero,

industrial, educativo, manufacturero, de gobierno y telecomunicaciones.

Para MGE el mercado de UPS's ha crecido exponencialmente en concordancia con las necesidades de protección de información. Por ello, el involucramiento de los proveedores en los requerimientos de las empresas es fundamental, pues a través de un análisis previo se pueden elaborar estrategias de prevención, mantenimiento y soporte adecuado a las zonas en las que se encuentran establecidas las compañías. De esta manera, “es como ofrecemos soluciones adecuadas y adaptadas a las necesidades de nuestros clientes, nuestro servicio está enfocado a brindar calidad en cada uno de nuestros productos”, comenta el Ing. Miguel Gutiérrez, director comercial de MGE UPS Systems México.

El futuro de los UPS's

Con sus 40 años de experiencia a nivel mundial y tres en México, MGE tiene un futuro prominente, pues según afirma Cristina Gamero, directora de administración y finanzas, su perspectiva de crecimiento para este año es de un 20% respecto a las ventas del año anterior.

Con la fortaleza y la ventaja competitiva que les da contar con un equipo de gente altamente capacitada y dispuestos a entender e involucrarse en los proyectos de sus clientes, MGE se ha planteado nuevas metas de consolidación de sus productos. La alianza comercial que ha mantenido con Grupo Schneider les permitirá penetrar estos mercados y consolidar su presencia de marca en los países de la región.



De izquierda a derecha:

Ing. Miguel Ángel Gutiérrez,
Lic. Cristina Gamero,
Ing. David Rubio

A pesar de las condiciones económicas adversas que se han registrado en el mundo en los últimos meses, MGE considera que el mercado de UPS's está en crecimiento, pues como explica la Lic. Gamero “los UPS so indispensables para salvaguardar la información; el que su empresa se encuentre protegida contra cualquier eventualidad natural o humana, genera confianza en los inversionistas toda vez que el núcleo de su negocio sigue siendo productivo. Los empresarios consideran a los UPS como una inversión redituable en corto plazo y un factor importante en la reducción de riesgos”.

“Somos una empresa líder mundial, joven en México pero con gran potencial de crecimiento, que refuerza día a día la confianza de nuestros clientes, distribuidores y accionistas”.



MGE UPS Systems México
Av. Congreso de la Unión 524
Col. Santa Anita, Del. Iztacalco
México 08300 D.F.
Tel 5538-9687 Fax 5530 7625



Apache Beehive

VE LA LUZ

BEA Systems anunció durante el ApacheCon 2004 que ya está disponible la primera versión alfa de su proyecto de código abierto Apache Beehive, un framework que pretende simplificar el desarrollo de aplicaciones empresariales y arquitecturas orientadas a servicios (SOA) a través de un modelo de objetos sencillo que utiliza J2EE y Struts.

Hasta el momento, Beehive está compuesto por tres elementos:

- **NetUI PageFlows** – Framework para aplicaciones web construido sobre Struts, que facilita la creación de los archivos de configuración de Struts a través del uso de metadatos.
- **Controls** – Framework de componentes que permite incorporar metadatos utilizando las nuevas capacidades de Java 5 para anotaciones (JSR-175).
- **Web Services** – Una implementación del JSR-181, que es un modelo de programación basado en anotaciones para desarrollar web services.

El proyecto actualmente se encuentra en estatus de incubación dentro de la fundación de software Apache. El objetivo de esta versión, llamada v1Alpha, es permitir que los desarrolladores lo comiencen a utilizar para crear aplicaciones SOA que se puedan ejecutar en diferentes servidores de aplicaciones. Beehive actualmente soporta los servidores de aplicación JOnAS y Apache Geronimo, así como el contenedor de web Tomcat. Al ser un proyecto de código abierto, el objetivo es que pronto se genere soporte para una amplia gama de plataformas. Con el v1Alpha también se liberaron herramientas adicionales, como es el caso

de controles para Hibernate, una herramienta para object-relational mapping (ORM). ORM es la capacidad de mapear clases en un lenguaje orientado a objetos —en este caso Java— hacia tablas de bases de datos relacionales para el manejo de persistencia de datos.

Para desarrollar con Beehive se puede utilizar el BEA Weblogic Workshop 8.1, o herramientas de código abierto como Pollinate, un plug-in de Eclipse creado para soportar esta tecnología. Además, Borland ya anunció que en el futuro próximo sus productos también soportarán este modelo de programación.

Para mayor información:
Beehive - Apache Software Foundation
incubator.apache.org/beehive/

Pollinate Project
www.eclipse.org/pollinate/

BEA Systems
www.bea.com

■ PRODUCTOS



Amazon.com

LANZA SIMPLE QUEUE SERVICE

Amazon.com ha lanzado el beta del Amazon Simple Queue Service (SQS), un servicio de colas de mensajes entre aplicaciones distribuidas. Con este lanzamiento, la empresa anunció sus planes para vender un servicio que administre colas de mensajes entre componentes de aplicaciones distribuidas. Durante el beta, que es gratuito para usuarios registrados, Amazon SQS funcionará como almacenamiento transitorio de mensajes pequeños (menores a 4 KB) y que permanecerán en la cola por un máximo de 30 días. Un mismo usuario podrá tener almacenadas un máximo de 4,000 peticiones. Amazon ha comentado que cobrará por el servicio una vez que sea oficialmente liberado, pero hasta ahora no ha dado información de precios.

Este es el primer producto que surge como resultado de la iniciativa de Amazon Web Services, lanzada en octubre de 2004. Esta iniciativa refleja una creciente tendencia entre las principales empresas del web para ofrecer su propia plataforma como infraestructura de TI para terceros. El servicio se puede acceder a través de una interfaz de web service, que permite siete operaciones: crear colas, configurar colas, listar colas, eliminar colas, agregar mensajes, leer datos y eliminar mensajes. Este servicio ha sido específicamente diseñado para ser usado en aplicaciones distribuidas, por lo tanto una misma cola pueda ser accedida de manera concurrente sin necesidad de que los componentes se coordinen entre sí para acceder la cola.

Microsoft, con MSMQ (Microsoft Message Queuing) e IBM con WebSphere MQ, ofrecen capacidades similares. Sin embargo, Amazon lo ofrece en un esquema de Application Service Provider (ASP), donde las empresas pagarían una cuota mensual para utilizar el sistema que reside en Amazon.com.

Este esquema puede ser atractivo para empresas pequeñas, ya que permite evitarse costos de instalación y mantenimiento, aunque involucra el riesgo de no tener el control absoluto sobre la información manejada.



Suscríbete **GRATIS**
a **SOFTWARE GURU**

Averigua cómo en
www.softwareguru.com.mx

Donde encontrarás: Noticias, Calendario de Eventos,
Foros, Suscripciones, Boletín Electrónico, etc.



GXportal

ARTech

www.gxportal.com

Calificación: 4 de 5

**Precio y Disponibilidad**

- Motor y 1 usuario: \$750 a \$2,800 USD.
- Usuarios adicionales: \$140 a \$350 USD.
- Contactar a la oficina de ARTech más cercana para disponibilidad en tu región.

Requerimientos de Sistema**Servidor**

- Windows 2000+.
- SQL Server 2000+.
- Internet Information Server 5.0+.
- .NET Framework 1.1.

Cliente

- Internet Explorer 5.5+, Netscape 6.0+, Mozilla 1.2+.

■ VEREDICTO

En general, consideramos que GXportal es una muy buena opción cuando se requiera un CMS sencillo que pueda ser administrado por los usuarios finales sin necesidad de intervención de personal técnico. Pero no se queden con la curiosidad, los invitamos a que soliciten una evaluación de este producto.

PROS:

1. Fácil de usar por personal no técnico.
2. Disponible en español.
3. Precio accesible.

CONTRAS:

1. Restringido a plataforma Microsoft por el momento.
2. No existe un lugar dónde obtener componentes que extiendan la funcionalidad base.

GXportal 4

VALOR Y SENCILLEZ



Gxportal es un motor para portales web cuyo objetivo es poder diseñar, administrar y mantener portales escalables sin necesidad de programar. Éste es un producto de ARTech, empresa de origen uruguayo con oficinas en México, Estados Unidos y Brasil. El producto insignia de ARTech es GeneXus, una plataforma para automatización de desarrollo de software. De hecho, GXportal forma parte de esta plataforma.

provee todas las funciones básicas de un CMS, como el diseño basado en plantillas —que permite manejar el diseño por separado del contenido—, administración de contenido con base en un workflow, búsqueda de contenido, personalización de páginas, conectores para contenido HTML, Flash y Javascript, componentes para foros de discusión, newsletters, FAQs y encuestas.

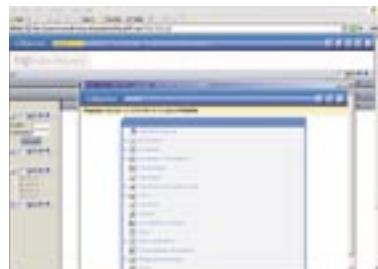
Además de esto, la versión completa ofrece:

• **Interacción con aplicaciones.** - Con GXportal, un portal no sólo es un sitio informativo, sino que las aplicaciones de negocio también se pueden habilitar para el web.

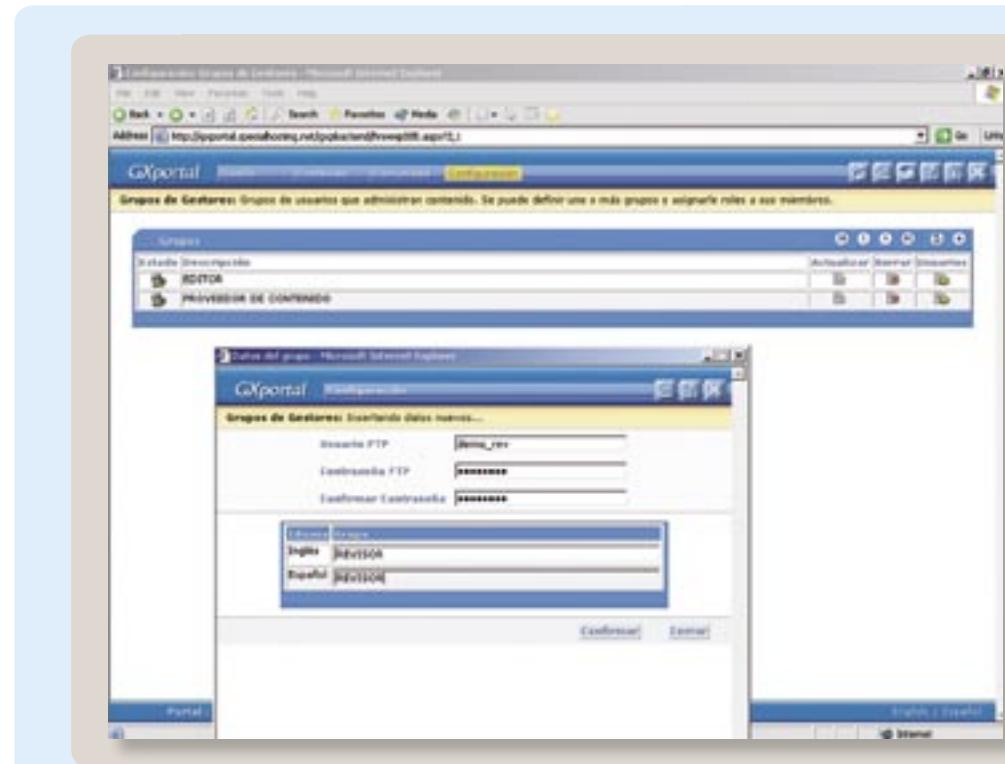
• **Single Sign On.** - Un objetivo común de un portal es que sirva de punto de entrada para diferentes aplicaciones habilitadas en web. La capacidad de Single Sign On (firma única), permite que un usuario se identifique una sola vez y pueda utilizar diferentes aplicaciones sin necesidad de estarse autenticando en cada una.

El principal atractivo de este producto es la capacidad de que personal no técnico pueda administrar un portal sin soporte del personal de sistemas.

Aún así, el principal atractivo de este producto es la capacidad de que personal no técnico pueda administrar un portal por sí mismo. Esto se logra a través del uso de asistentes (wizards) y menús para todas las actividades de administración de diseño y contenido. En pocas palabras, es posible crear un sitio completo sin necesidad de escribir una sola etiqueta de HTML. Tal vez esto parezca irrelevante a nuestros lectores, que seguramente conocen la especificación de HTML y DOM al derecho y al revés, pero lo importante es que permite que los usuarios finales puedan manejar un portal entero sin necesidad de recurrir al personal de sistemas... imaginén la belleza.



Otro atractivo de GXportal es su precio, el cual es bajo al compararlo con herramientas similares en el mercado. El esquema que maneja GXportal es que se adquiere una licencia que incluye el motor y un usuario administrativo, y por separado se puede adquirir licencia para usuarios administrativos adicionales. La licencia de la versión Lite tiene un precio de \$750 dólares, mientras que la licencia de la versión completa cuesta \$2,800. Los precios por usuario administrativo adicional dependen del volumen, pero van de \$140 a \$350 dólares por usuario. Todo esto es independiente de la cantidad de usuarios no administrativos, la cual es ilimitada. Estos precios son atractivos al compararlos con soluciones para portales como la de Bea u Oracle, que típicamente superan los \$80,000 dólares.



Trabajar en base a grupos de trabajo permite controlar los privilegios de los usuarios.

Por otro lado, es cierto que existen CMS de código abierto con disponibilidad gratuita, que brindan gran flexibilidad en cuanto a sus capacidades. El problema con ellos es que aún no tienen la facilidad de uso de los productos comerciales, por lo que requieren mayor conocimiento técnico y están fuera del nicho en que se enfoca GXportal.

La instalación de este producto es bastante sencilla. Simplemente se ejecuta un instalable y se va siguiendo el asistente para configurar la conexión a la base de datos, el servidor web y el servidor FTP. Posteriormente se utiliza un módulo de administración de licencias para alimentar las licencias disponibles. Despues de esto ya estamos listos para crear y publicar un website. La documentación incluida en GXportal es bastante completa, ya que incluye manual de instalación, un tutorial para guiar al usuario en la funcionalidad básica, y ayuda detallada en formato Microsoft Help.

La versión 4.0.1 de GXportal solamente funciona sobre plataforma Microsoft, ya que requiere Internet Information Server como servidor web, y SQL Server como manejador de base de datos, además de utilizar el .NET Framework 1.1. Sin embargo, Eugenio García, Project Manager de GXportal,

nos comenta que la versión 4.1 —que estará disponible cuando ustedes estén leyendo esto— ya se ejecuta sobre plataforma Java, pudiendo utilizar como DBMS tanto Oracle como DB2. Esto hace posible que GXportal sea instalado en ambientes Linux/Unix, utilizando cualquier motor de servlets (Websphere, Tomcat, Oracle IAS, JRun, etc.) y Apache como servidor web. Otra característica importante para esta versión es el uso de cache en las sentencias a la base de datos, lo cual brinda una mejora considerable en el desempeño y escalabilidad.

Algo que sentimos que le hace falta a GXportal es una comunidad de desarrolladores que contribuyan componentes para extender la funcionalidad base de este producto. Hemos visto que otros productos cuentan con esto, y es de gran ayuda. En este sentido, Eugenio nos comenta que en la versión 5.0 se podrá extender GXportal a través del estándar WSRP (Web Services for Remote Portlets). “Esto nos permitirá integrarnos a la comunidad que hoy en día existe en torno a esta especificación, así como crear nuestra propia comunidad para que las empresas puedan intercambiar información mediante este mecanismo.”, agrega Eugenio. ☐

Por Isabell Chillopa

Gestión de Portafolio de Proyectos

ALINEANDO LAS TECNOLOGÍAS DE INFORMACIÓN CON EL NEGOCIO

En los últimos años, el alineamiento entre las Tecnologías de la Información (TI) y la estrategia del negocio ha estado y se mantiene en el nivel más alto de prioridad de las organizaciones. Este interés radica en la manera en que las inversiones en TI, recursos, oportunidades de negocio y el portafolio de aplicaciones pueden estar en armonía con los objetivos estratégicos de la organización, de tal manera que permitan apoyar el proceso de toma de decisiones y mejorar el uso de recursos existentes para incrementar la productividad, haciendo así más efectiva y eficiente a la organización.

De acuerdo con información reportada en diversos estudios, las organizaciones enfrentan una diversa problemática al respecto. Pocas empresas seleccionan de manera exitosa un portafolio de proyectos que sea consistente con su estrategia de negocio. Un proyecto puede ser exitoso desde la perspectiva de tiempo, presupuesto y alcance, pero si falla en cumplir o satisfacer los objetivos de negocio, fracasa de manera completa.

Al igual que la selección de proyectos, la ejecución de éstos también acostumbra estar descentralizada y fragmentada. Las mejores prácticas de la industria y lecciones aprendidas derivadas de la ejecución de los proyectos no se identifican y gestionan para ser aplicadas sistemáticamente en la organización, desaprovechando la sinergia potencial entre proyectos. En otros casos, no se cuenta con procesos definidos para revisar propuestas de proyectos, ni un rastreo adecuado que permita identificar los proyectos que fracasan en el cumplimiento del valor de negocio prometido. Incluso, llega a suceder que los niveles directivos ni siquiera cuentan con una lista completa de los proyectos de TI en curso dentro de la organización. En resumen, no se cuenta con una visibilidad adecuada de lo

que en realidad se está haciendo en la organización.

El resultado de estas deficiencias se ve reflejado en la ejecución de demasiados proyectos, una gran cantidad de complejidad y redundancia; así como fallas, retrasos y excesos en el presupuesto de los proyectos. Es evidente por lo tanto, que las organizaciones que no tienen control sobre sus portafolios de proyectos de TI están condenadas al fracaso.

Para solucionar esta problemática, una de las estrategias de negocio que más fuerza está tomando es la Gestión de Portafolio de Proyectos o Project Portfolio Management (PPM). PPM permite a las organizaciones alinear sus proyectos de TI y recursos con los objetivos de negocio corporativos. PPM brinda a las organizaciones una visión integral de su estrategia de TI, permite ganar control sobre sus proyectos y ayuda a generar valor al negocio.

Características y Beneficios de PPM

PPM involucra desde la identificación y priorización de oportunidades de negocio (se examinan las propuestas de proyecto con respecto a los objetivos corporativos), hasta la ejecución y cierre de proyectos, organizándolos en portafolios.

Con PPM se desarrollan y monitorean mediciones que tratan los activos de TI de igual manera como se tratarían activos o portafolios de diversas inversiones financieras; por ejemplo, inversiones estratégicas más riesgosas se balancean con inversiones más conservadoras y la mezcla se monitorea constantemente para evaluar cuáles proyectos siguen su curso, cuáles necesitan ayuda y cuáles deben ser terminados. Al mantener un portafolio balanceado, se reduce el riesgo en cada proyecto, se obtiene un mayor entendimiento de los aspectos económicos de cada uno y se genera una tasa más alta de retorno de inversión general del portafolio. Asimismo, se tiene mayor visibilidad y un uso eficiente de los recursos entre los diferentes proyectos.

PPM brinda claros y múltiples beneficios a las organizaciones. En esencia, los ejecutivos y gerentes pueden monitorear sus portafolios de proyectos facilitando la administración integrada del alcance, tiempo, costo, recursos, habilidades, adquisiciones, comunicación, reporte, predicciones y riesgos, y alineando estos proyectos a los objetivos de negocio para incrementar la productividad, apoyar la toma de decisiones oportuna e informada, y generar mayor valor al negocio.

Isabel Chillopa es consultor y responsable de la Oficina de Proyectos en Itera. Anteriormente laboró en el Instituto de Investigaciones Eléctricas (IIE), donde participó en proyectos de desarrollo de software y coordinó la certificación del sistema de calidad conforme a la norma ISO 9001. Isabel es Licenciada en Informática del Instituto Tecnológico de Záratepec, y Maestra en Administración de Tecnologías de la Información del ITESM.

Al mantener un portafolio balanceado, se reduce el riesgo en cada proyecto y se obtiene un mayor entendimiento de los aspectos económicos de cada uno.

Cómo Iniciar PPM

No hay una manera única de implantar PPM. Diferentes empresas manejan diferentes modelos y metodologías. Sin embargo, Todd Datz, Editor Ejecutivo de la revista CIO, establece ciertos pasos clave en la creación y administración de portafolios de proyectos de TI:

- **Reunir.**- Crear un inventario de proyectos es una tarea ardua pero bien vale la pena. En muchos casos, puede ser la primera vez que se tenga una vista completa de los proyectos de TI, y permite encontrar redundancias. Un buen inventario es la base para desarrollar proyectos alineados con los objetivos estratégicos del negocio.
- **Evaluar.**- Despues de inventariar los proyectos, se establece un portafolio de éstos. Los líderes de las unidades de negocio, en conjunto con los líderes de TI, deben sopor tar los proyectos con casos de negocio que muestren estimación de costos, ROI, análisis de riesgos y beneficios esperados.

• **Priorizar.**- Aún despues de evaluar los proyectos, la mayoría de las empresas tendrán más proyectos de los que pueden realizar. El proceso de priorización permite asignar recursos a los proyectos que estén más alineados con los objetivos estratégicos de la organización.

• **Revisar.**- Una vez que se tiene una lista de proyectos aprobados, es vital administrarlos activamente. Esto involucra monitorear los proyectos a intervalos frecuentes. Contar con la visión de portafolio también facilita la decisión de cancelar proyectos cuando sea necesario. Jeff Chasney, Vicepresidente de Planeación Estratégica y CIO de CKE Restaurants, comenta: "No requieres completar todos los proyectos simplemente porque los empezaste.".

Herramientas de Apoyo a PPM

Existen diversas herramientas de software para asistir en la implantación y automatización de esta práctica. Uno de los productos más conocidos es Primavera (www.primavera.com). Se espera que en el 2005 el mercado de estas herramientas sea de \$540 millones de dólares anuales. Seguramente esto es lo que motivó a IBM para entrar a este mercado con el reciente lanzamiento del Rational Portfolio Manager que, al integrarse con el resto de la plataforma de desarrollo de IBM, puede ofrecer grandes beneficios.



Las herramientas de software son vitales para implantar PPM. Imagen: Rational Portfolio Manager.

Como toda iniciativa nueva, la implementación de PPM requiere tiempo y esfuerzo. Sin embargo, los retos que conlleva son mínimos en comparación con el valor y beneficios que brinda a la organización. ☺

Referencias:

Datz, Todd. "Portfolio Management, How To Do It Right".
CIO Magazine, Mayo 2003

Reddy, Ashok. "PPM: Aligning Business and IT".
IBM developerWorks, Mayo 2004

Autores varios. "Centralizing Management Of Project Portfolios".
Meta Group. Enero 2002



CHIAPAS
Centro Turístico
por excelencia...



Expertos en RUP
Generadores de Código
Microsoft.NET
Rich Clients
N-Tiers
Maquila de Software
Componentes Móviles
MCSD
CRM

s sistemas
Software Global

6a. Sur Poniente #1248 4to. Piso
Colonia La Lomita
Tuxtla Gutiérrez, Chiapas
ventas@ssistemas.com
Tel y Fax: +52 (961) 6131963
www.ssistemas.com

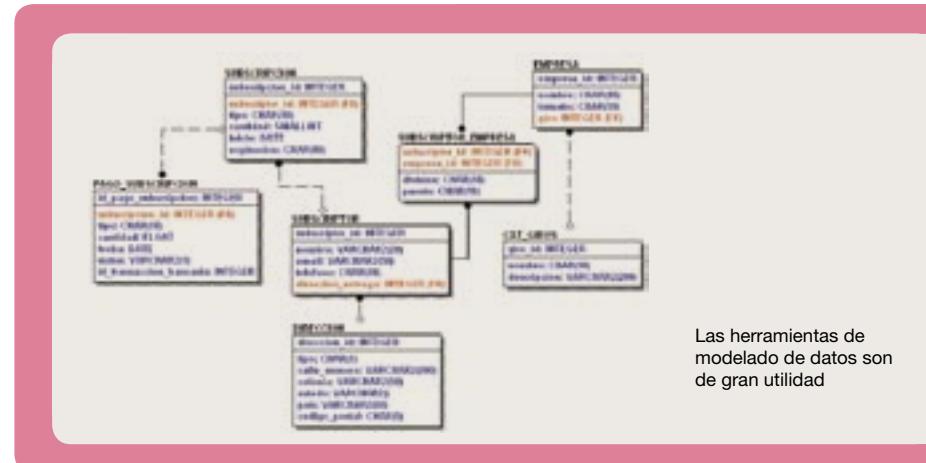
Por Artemio Mendoza

El Síndrome ERwin

Los modeladores de datos son herramientas indispensables para el desarrollo de software. Es importante que conozcamos su funcionalidad, así como las limitaciones que pueden tener. En este artículo, Artemio Mendoza nos narra su experiencia con ERwin Data Modeler de Computer Associates.

ERwin es uno de los productos más conocidos para modelado de datos. Su funcionalidad va más allá de simplemente dibujar diagramas, ya que también genera los scripts de creación de la base de datos, permitiendo escoger entre diferentes DBMS destino como Oracle, DB2, Sybase, SQL Server y otros más. También es posible conectarse al DBMS de manera nativa —en el caso de Oracle, utilizando SQL*Net— o por ODBC para generar en línea los objetos y sus relaciones o hacer ingeniería en reversa para obtener el diagrama Entidad Relación (ER) de una base de datos existente. En fin, si observamos el conjunto de servicios que ofrece, podemos entender por qué ERwin goza de tanta popularidad.

Sin embargo, aún con todas sus gracias, si dejamos demasiadas cosas en las manos de ERwin terminaremos con bases de datos altamente inefficientes. ¿Por qué esta afirmación? Bueno, para justificarla voy a hablar acerca del RDBMS que conozco, el cual es Oracle. Sucede que he llegado a varios proyectos porque tienen problemas con la fragmentación de los discos de la base de datos, tablas que no pueden crecer más, problemas con integridad referencial —no se pueden borrar datos de alguna tabla con Foreign Key—, y algunos otros más sencillos que no se ven a simple vista pero que afectan al desempeño global del DBMS como encadenamiento en bloques



Las herramientas de modelado de datos son de gran utilidad

de datos, espacios más grandes de lo necesario, niveles de concurrencia inadecuados. ¿Por qué sucede esto? La respuesta es muy simple, resulta que en el caso de Oracle, la creación de un objeto, como una tabla o un índice, necesita parámetros de almacenamiento, tales como PCTFREE, PCTINCREASE, INITIAL, MAXEXTENTS y otros más. Pero, si se realiza automáticamente por un usuario que no conoce la existencia de estos, los valores se toman por omisión y rara vez resultan ser los adecuados. Un ejemplo típico es el PCTINCREASE, que indica el porcentaje de espacio que se va a reservar para el objeto creado cuando se acabe el espacio (*extent*) actual. El valor por omisión es de 50. ¿Qué significa esto? Bueno, que si tengo una tabla que de INITIAL tiene 10MB pero sigue creciendo y rebasa estos 10MB, la próxima extensión que se va a tomar será 50% más grande, es decir, de 15MB, y la siguientes de 22MB, 33.8 MB, 50.7MB y 76MB. Así, en tan sólo cinco iteraciones se incrementó en casi 800% el valor inicial. ¿Resultado? Fragmentación y espacio mal utilizado, además de que no permite saber si el próximo *extent* va a caber en el tablespace. Otro ejemplo es el parámetro MAXEXTENTS que indica el número máximo

de *extents* que se permiten alojar para ese objeto. Así, si se indica un valor demasiado bajo, puede ser que la tabla rebase estos valores y marque un error *ORA-01631 max # of extents num reached in [table name]*.

Evidentemente, después de que se corren los scripts y se crea toda la base de datos, no existen problemas y se procede al desarrollo de las aplicaciones. Sin embargo, no pasa mucho tiempo cuando empiezan a surgir las consecuencias. Todos sabemos que cuesta más arreglar un problema que prevenirlo —el costo de arreglar un problema durante el desarrollo debe de tomar en cuenta a los programadores que quedan parados sin poder trabajar—. ¿Y qué decir cuando estos sistemas se liberan a producción con los parámetros por omisión mencionados? Siempre se termina por llamar a alguien que arregle los dimensionamientos.

¿Entonces no debemos utilizar para nada ERwin y hacer nuestros diagramas y scripts a mano? ¡Por supuesto que no! El planteamiento es que subordinemos esta excelente herramienta a los criterios de quien conoce los requerimientos específicos del DBMS para el que está generando, de tal forma que realice los cálculos pertinentes e indique los valores óptimos. Si no hacemos esto, aún cuando generemos nuestros objetos a mano, el riesgo será el mismo. Toma en cuenta que “Es mejor prevenir que remediar”. Este es un consejo muy sabio y económico. ¿No lo creen? ☺

Artemio Mendoza es Director de Operaciones de Towa Software, empresa de consultoría en TI de la cual es socio fundador. Durante más de once años se ha desempeñado como consultor, gerente y director en empresas de tecnología en México y Estados Unidos para clientes como GE Plastics, McKinsey & Company, Bancomer y Alestra. Artemio tiene el grado de Maestría en Administración de TI, otorgado por el Tec de Monterrey.

Desde 1997 la **amcis** ha reunido a los profesionales y académicos interesados en promover la calidad en la Ingeniería de Software.

Nuestro objetivo es compartir el conocimiento y experiencia a través de:

- Diplomado en Calidad de Software (único en México)
- Conferencias y reuniones mensuales
- Seminarios



Si quieres aprender lo mejor de la Calidad en Ingeniería de Software y compartir tus experiencias **ven a conocernos.**

amcis

A large, stylized logo for "amcis" in white and purple. The letters are lowercase and have a modern, rounded font. The letter "m" has a vertical purple bar extending downwards, and the "c" has a horizontal purple bar extending to the right.

Mayores informes: www.amcis.org.mx, e-mail: info_amcis@yahoo.com.mx, Tel.: 5563.3059
Entrada gratis a las conferencias y reuniones mensuales presentando este anuncio.



Martín Méndez

DIRECTOR DE TECNOLOGÍA DE INFORMACIÓN EN GNP

Por Jorge Palacios

Desde sus inicios en la industria del software hace 24 años, Martín ha tenido la gran inquietud de qué hacer para cumplir los proyectos a tiempo y con calidad. Esta mentalidad lo ha llevado a emprender importantes esfuerzos, como fue el caso de Tecnosys, empresa fundada por Martín que en 1999 se convirtió en la primera de América Latina en ser reconocida como SW-CMM3. Actualmente, Martín se desempeña como Director de Tecnología de Información en GNP, donde sigue tan cautivo como siempre del afán de mejorar a través de los procesos.

Software Guru.- ¿Cómo y cuándo empiezas a dar cauce a esta inquietud?

Martín Méndez.- En 1991, mientras visitaba una biblioteca del MIT, me llamó la atención un libro titulado "Managing The Software Process" de Watts Humphrey. Al leerlo me di cuenta que el software no era algo artesanal, sino que era administrable.

¿Cuál es el valor de los procesos en las organizaciones?

Soy un firme creyente de los procesos y

además me ha tocado ver empresas muy exitosas alrededor de los procesos, en IBM es impresionante lo que han hecho.

Actualmente se habla de que la competencia ya no va a estar alrededor de que tu producto sea mejor, sino de que generes las mejores experiencias a tus clientes. Esto sólo se puede lograr con operaciones bien definidas, medidas y continuamente mejoradas, y eso sólo te lo dan los procesos.

¿Qué importancia tiene la ingeniería de software para ayudar a los "sistémicos" a escalar la pirámide organizacional?

Un artista se hace solito, porque es una labor de él. En el momento en que te conviertes en ejecutivo, ya no eres artista, y tu misión no es un arte, sino una función en una organización. La ingeniería de software brinda un marco para que algo que estaba considerado como un arte, se pueda industrializar y administrar.

“Los dos obstáculos mayores (para implantar CMM) son convencer al que pone el dinero y lograr que la gente se discipline.”

Cuéntanos tu experiencia encabezando un proyecto de implantación CMM.
Los dos obstáculos mayores son convencer al que pone el dinero y lograr que la gente se discipline. En cuanto a esto último, es importante generar un mensaje adecuado donde se explique el beneficio tanto para la organización como para las personas, hay diferentes cosas que se pueden hacer.

¿Nos podrías compartir alguna experiencia en este sentido?

En mercadotecnia existe algo llamado “el cruce del abismo”. Esto se refiere a que entre los pioneros (early adopters) y la mayoría, hay un abismo que tienes que ver cómo cruzar. En nuestro caso, un día hicimos una sesión donde presentamos los proyectos de estas personas para que el resto vieran los resultados, y al mejor proyecto le dimos un premio que consistía en un cheque (simbólico), los pusimos en frente de la empresa y les dimos un reconocimiento. El costo de esto no fue significativo y, en cambio, jaló a la gente muchísimo. Con base a los early adopters tienes que jalar a la mayoría.

¿Cómo convenciste a IBM de invertir en un proyecto de mejora de Tecnosys?

Les expliqué que habían comprado una empresa de la cual seguramente habían pedido referencias, y las cuales normalmente iban llenas de referencias personales, por ejemplo “si tú te traes a Martín Méndez y a este y a este otro, tu proyecto saldrá bien”. Pero nunca fueron referencias hacia la empresa ni al método ni al proceso, y lo que yo quería cambiar era eso. Les comenté que yo no podía clonar 50 personas en un mes, pero lo que sí podía hacer era generar un programa de capacitación e inducción para que conforme se incorporara personal nuevo, se fuera alineando a la manera de hacer las cosas.

La verdad es que al principio sólo me dieron el dinero para ISO 9000, que era sustancialmente menor que el necesario para CMM, y medimos resultados. En un año ya estábamos certificados en ISO 9000.

Para los que ponen el dinero hay dos cosas importantes: la rentabilidad y la satisfacción del cliente. La rentabilidad la medimos mes a mes, en los estados de resultados; y para medir la satisfacción del cliente, generamos encuestas y demostramos que una vez implantando el sistema de calidad, dejando pasar el tiempo necesario, había subido la satisfacción del cliente. Entonces ya justificaba la inversión en CMM, con una visión de incursionar en el mercado de la exportación.

¿Es aplicable CMMI a las PyMEs desarrolladoras de software mexicanas?
Yo creo que el (modelo) continuo es aplicable, y el ejemplo es Ultrasist, que habla del nivel en el que ya estamos en México. No se necesita mucho capital financiero, sí capital humano. Ellos tienen un capital intelectual y humano muy sólido, de llamar la atención. Ese tipo de empresas sí lo van a poder hacer.

La otra es el modelo flexible, al cual sí le veo posibilidades en las PyMEs mexicanas. También les recomiendo que vean la segunda versión del MoProSoft como una serie de recomendaciones para configurar el modelo continuo de CMMI.

Para mí el valor del MoProSoft es agarrar y decir: “aquí está este modelo mexicano que es barato. Úntrenle y adquieran la disciplina, y después vayan un paso adelante”. Eso fue lo que sucedió con el ISO 9000 en Tecnosys, el principal valor fue que empezamos

a entender lo que significaba trabajar en forma disciplinada, el otro fue demostrarlo al consejo de administración que sí se podía hacer y que tenía buenos resultados.

¿Cómo te integras a GNP?

GNP en mí busca dos cosas: una experiencia en el mercado de desarrollo y la capacidad de generar un proceso disciplinado. Tengo a mi cargo la tarea de integrar los servicios al negocio, mas la tarea no formal de ir ordenando el proceso de desarrollo. Como dice Humphrey: “*Every business is a software business*”.

¿Cómo va el proceso de generar disciplina en GNP?

Tanto mi jefe como mis compañeros, creemos que podemos profundizar y obtener mejores resultados. Estamos buscando poner orden y luego reducción del gasto, no solamente a través de la eficiencia del desarrollo. Se están obteniendo formas de hacer más con lo mismo, o hacer más con menos. Todo el mundo dice que le eches ganas y que se queden más a trabajar, pero eso ya sabemos que no funciona. Mi rol consiste en mejorar la satisfacción de mi cliente y un componente importante es entregar a tiempo y en costo lo que él necesita. La otra parte es entender el negocio y cómo solventar sus requerimientos. Mi rol pasó de preocuparme por cómo mejorar el desarrollo del software a cómo generar las soluciones que mi cliente está buscando.

¿Qué mensaje le das a nuestros lectores?

Escojan algo en lo que quieren ser buenas y dedíquense a eso por una temporada larga, el tiempo les dirá si pudieron lograrlo o no. **G**



Procesos

En un mundo de cambios constantes y competencia global, las organizaciones de desarrollo de software son presionadas a alcanzar mayor eficiencia con menores costos. Para poder lograr este objetivo, es necesario adoptar una forma de trabajo que permita entender, controlar, comunicar, mejorar, predecir y certificar el trabajo realizado.

Actualmente existe una gran diversidad de opciones relacionadas con procesos de desarrollo. Constantemente se escuchan diferentes acrónimos como CMM, CMMI, RUP, ISO, PSP, TSP, etc., que causan confusión, principalmente debido a la mala interpretación de los mismos.

Revisemos entonces los principales procesos de desarrollo y modelos más utilizados al momento, así como los estándares relacionados.

de

Software

GUÍA DEL VIAJERO

Por Mara Ruvalcaba

¿Por qué contar con un proceso de software?

Hasta hace poco tiempo, la producción de software era realizada con un enfoque artístico, a diferencia de un enfoque industrial. Ante la constante presencia de proyectos fallidos, y con el objetivo de mejorar la calidad de los productos, en los últimos años las organizaciones introdujeron los métodos de ingeniería de software. (Ver Fundamentos – Desarrollar software es mucho más que programar, pág. 42)

A partir de estos, se formalizó el enfoque de ingeniería de producto para desarrollar software. Factores como la globalización han obligado a las organizaciones a contar con marcos de trabajo que las ayuden hacer las cosas de la manera más eficiente. Fue entonces que se incorporó la ingeniería de procesos al desarrollo de software.

Proceso

Antes de definir lo que es un proceso de desarrollo de software, entendamos lo que es un proceso. Una definición sencilla de proceso es “serie de acciones que conducen a un final”. Esta definición parece coincidir con las ideas generales de la gente sobre procesos, pero deja muchas preguntas abiertas. ¿El proceso es la forma en que la organización opera —desde mercadotecnia hasta recursos humanos— o es la forma en que un desarrollador diseña, produce código, o prueba el software? ¿El proceso se refiere a administración, ingeniería, o ambas? ¿El proceso implica demasiada documentación y nos abstiene de desarrollar el producto objetivo?

La respuesta a éstas puede variar dependiendo de la perspectiva. Sin embargo, siempre que para alcanzar algún fin deseado necesitemos ejecutar una serie de acciones, y estas acciones tengan cierto orden, dependencias, roles responsables, resultados, tiempos de ejecución y herramientas de apoyo, estaremos hablando de procesos, que pueden ser predefinidos y personalizados.

Proceso de software

La meta de la ingeniería de software es construir productos de software, o mejorar los existentes; en ingeniería de procesos, la meta es desarrollar o mejorar procesos.

Un proceso de desarrollo de software es un conjunto de personas, estructuras de organización, reglas, políticas, actividades y sus procedimientos, componentes de software, metodologías, y herramientas utilizadas o creadas específicamente para definir, desarrollar, ofrecer un servicio, innovar y extender un producto de software.

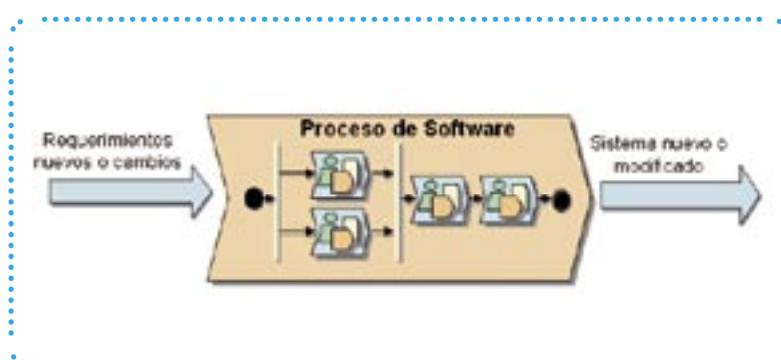
Un proceso de software efectivo habilita a la organización a incrementar su productividad al desarrollar software:

- Permite estandarizar esfuerzos, promover reuso, repetición y consistencia entre proyectos.
- Provee la oportunidad de introducir mejores prácticas de la industria.
- Permite entender que las herramientas deben ser utilizadas para soportar un proceso.
- Establece la base para una mayor consistencia y mejoras futuras.

Un proceso de software mejora los esfuerzos de mantenimiento y soporte:

- Define cómo manejar los cambios y liberaciones a sistemas de software existentes.
- Define cómo lograr la transición del software a la operación, y cómo ejecutar los esfuerzos de operación y soporte.

Necesitamos un proceso de software cuya funcionalidad esté probada en la práctica, y personalizado para que cumpla con nuestras necesidades específicas.



¿Y las Metodologías Ágiles?

Las metodologías ágiles son otro ejemplo de prácticas específicas para desarrollar software. Uno de los métodos ágiles más conocidos es Extreme Programming (XP), que se describe como una disciplina de desarrollo de software basada en valores de simplicidad, comunicación, retroalimentación, y coraje. XP funciona al poner a todo el equipo en presencia de prácticas sencillas, con suficiente retroalimentación para habilitar al equipo para ver donde están y afinar las prácticas para una situación única. No hemos listado XP como un modelo específico, ya que no se considera un proceso formal y completo, sino que más bien es un conjunto de prácticas que se pueden aplicar dentro de los modelos mencionados.

Elementos Típicos del Proceso de Software

Actividad	Definen las acciones que se llevan a cabo en un momento dado del desarrollo de software.
Flujo de Trabajo	Colección estructurada de actividades y elementos asociados (artefactos y roles), que producen un resultado de valor.
Rol	Son responsables por llevar a cabo las actividades del proceso, pueden ser personas o herramientas.
Producto o Artefacto	Son las entradas y salidas de las actividades, pueden ser de diferentes tipos, como documentos, modelos, componentes, planes, reportes, etc.
Disciplina	Conjunto integrado por actividades relativas a una rama particular de conocimiento. Ej. Análisis y diseño.

Diversidad en Modelos

Actualmente existe una gran variedad de modelos para procesos de software. Podemos entenderlos más fácilmente si los clasificamos en dos tipos: genéricos y específicos.

Modelos Genéricos

Abarcan todos los procesos relacionados con el desarrollo de software

CMM – modelo de madurez de capacidades – estándar de facto

CMMI – modelo integrado

ISO 9001-2000 – sistema para administración de la calidad – estándar

ISO/IEC 15504 – marco para evaluación de procesos de software – en vías de ser estándar, por ahora reporte técnico

MoProSoft – modelo de procesos para la industria de software en México – en vías de ser norma mexicana

Modelos Específicos

Enfocados a la ingeniería de productos de software

UP – proceso de desarrollo

RUP – proceso de desarrollo

PSP – enfocado en individuos

TSP – enfocado en equipos (incluye PSP)

Revisemos estos modelos para entender mejor su objetivo y estructura.

Modelos Genéricos

CMM (Capability Maturity Model) - Modelo de Madurez de Capacidades

Marco que describe elementos clave de procesos efectivos de software. Creado por el Software Engineering Institute (SEI) en conjunto con Carnegie Mellon University. La primera versión se publicó en 1994.

CMM describe un camino evolutivo en 5 niveles de mejora de procesos para lograr su madurez. Cubre prácticas de planeación, ingeniería y administración del desarrollo y mantenimiento de software.

Niveles de Madurez

Nivel

1 – Inicial

Proceso caótico, impredecible. El éxito depende del esfuerzo heroico de individuos.

2 – Repetible

Institucionalizar procesos efectivos de administración de proyectos de software, que permiten a las organizaciones repetir prácticas exitosas desarrolladas en proyectos previos.

3 – Definido

El proceso estándar para desarrollar y mantener software en la organización está documentado, incluyendo procesos de administración e ingeniería de software, y estos procesos están integrados.

4 – Administrado

Se establece un conjunto de metas cuantitativas para medir el nivel de calidad y desempeño de los proyectos y del proceso organizacional.

5 – Optimizado

No es simplemente detectar y resolver defectos, sino prevenirlas y evitarlas al implementar actividades proactivas. Mejora continua de procesos.

Áreas clave del proceso

Ninguna

- Administración de Requerimientos.
- Planeación de Proyecto de Software.
- Seguimiento y Control del Proyecto de Software.
- Administración de Subcontratos de Software.
- Aseguramiento de Calidad de Software.
- Administración de Configuración de Software.

- Enfoque en Procesos de la Organización.
 - Definición de Procesos de la Organización.
 - Programa de Capacitación.
 - Administración Integral de Software.
 - Ingeniería de Productos de Software.
 - Coordinación Intergrupal.
 - Revisiones entre Colegas.
- Administración cuantitativa de procesos.
- Administración de la calidad del producto de software.

- Prevención de defectos.
- Administración de cambio de tecnología.
- Administración de cambio de procesos.

ISO 9001-2000

ISO (International Standards Organization) en 1987 crea la norma ISO 9000, conjunto de estándares que establecen los requerimientos para la gestión de los sistemas de calidad. ISO 9000:2000 está formado por:

- ISO 9000 Fundamentos y Vocabulario.
- ISO 9001 Requisitos.
- ISO 9004 Recomendaciones.

La parte de Requisitos - ISO 9001:2000, está estructurado en 8 secciones:

1. Alcance.
2. Normas para la Consulta.
3. Términos y Definiciones.
4. Sistema de Gestión de la Calidad.
5. Responsabilidad de la Dirección.
6. Gestión de los Recursos.
7. Realización del Producto.
8. Medida, Análisis y Mejora.

Aunque ISO 9001:2000 no otorga un estándar específico para sistemas de desarrollo de software, es decir, no abarca todos los procesos relacionados con el desarrollo de software, muchas organizaciones de software han optado por gestionar su sistema de calidad en base a este estándar, y obtener una certificación reconocida de manera internacional.

En el año 2000, el SW-CMM fue reemplazado por CMMI. El SEI ya no mantiene el modelo SW-CMM, sus métodos de evaluación asociados, ni el material de entrenamiento.

CMMI (Capability Maturity Model Integration) - Modelo de Madurez de Capacidades Integrado

El proyecto de CMMI fue concebido como una iniciativa para reunir los diferentes CMMs en un conjunto de modelos integrados, más consistentes entre ellos. Los modelos fuente que sirvieron como bases incluyen: CMM Software, CMM Ingeniería de Sistemas, y CMM Desarrollo Integrado de Producto .

CMMI proporciona una guía para desarrollar procesos, que además ayuda a evaluar la madurez de la organización o capacidad de un área de procesos. CMMI incluye los procesos de ingeniería de software e ingeniería de sistemas.

El modelo está representado de forma continua y escalonada. Contiene 22 áreas de procesos. Cada área de proceso está formada por: Objetivos específicos, Prácticas específicas, Objetivos genéricos, y Prácticas genéricas.

CMMI Modelo Continuo

Esta formado por 5 niveles de capacidad del proceso:

0. Incompleto
1. Desempeñado
2. Administrado
3. Definido
4. Administrado cuantitativamente
5. Optimizado

Y por cuatro categorías de áreas de procesos: Administración de Procesos, Administración de Proyectos, Ingeniería, Soporte. Estas categorías agrupan a las diferentes áreas de proceso, dividiéndolos en procesos básicos y avanzados.

CMMI Modelo Escalonado

El modelo escalonado, al igual que CMM, describe un camino evolutivo en 5 niveles de madurez de procesos, además integra nuevas áreas de proceso.

La selección entre el modelo escalonado y el continuo depende del objetivo de la organización, además de tener que considerar la situación (si ya se tiene CMM, cultura en procesos, etc.). Por ejemplo, si el objetivo es llevar a la organización a cierto nivel de capacidad, deberá seleccionarse la forma escalonada; en cambio si el objetivo es mejorar cierto proceso, será mejor seguir la forma continua.

Algunos Beneficios de CMMI vs. CMM

Algunos de los beneficios que han experimentado las organizaciones que pasan de CMM a CMMI son los siguientes:

- Mejor alineación a objetivos de negocio.
- Mejor visibilidad hacia las actividades de ingeniería, con el objetivo de asegurar que el producto o servicio cumple las expectativas del cliente.
- Aprovechamiento de mejores prácticas adicionales (e.j., medición, riesgo, administración, y manejo de proveedores).
- Acoplamiento más estrecho con estándares de ISO.

ISO/IEC 15504

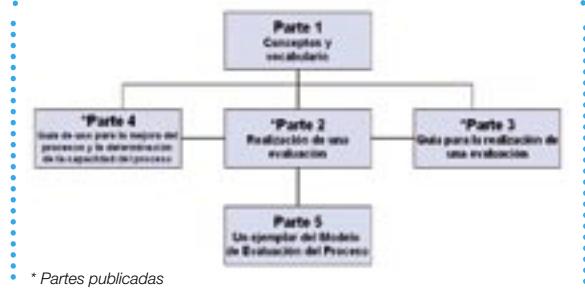
Estándar internacional que ofrece un marco para la evaluación de procesos. Fue iniciado en 1991 como el proyecto SPICE (Software Process Improvement and Capability dEtermination). La versión de Reporte Técnico fue aceptada y publicada en 1998, enfocada únicamente en procesos de software.

En el transcurso de su desarrollo ha evolucionado, de ser un modelo de referencia de buenas prácticas de software, para convertirse en un marco de trabajo para evaluación de múltiples modelos (de software o no). Su dirección actual es poder aplicarse a múltiples disciplinas y permitir a las diferentes comunidades definir sus propios procesos específicos, modelos de referencia o buenas prácticas. ISO 15504 está en vías de convertirse en estándar internacional, y se espera que esté completo para 2006. Esta versión esta compuesta de cinco documentos, a diferencia del reporte técnico que consta de nueve partes (Ver gráfica 1 - Estructura de documentos).

La parte 2 de este estándar es de especial interés, ya que es la que define como se realiza una evaluación. Establece requisitos tanto para modelos de procesos de referencia como para los métodos de evaluación sin establecer alguno en particular.



Gráfica 2 - Modelo de Capacidades 15504-2



Gráfica 1 - Estructura de Documentos 15504

Niveles de Capacidad (Ver gráfica 2):

0. Incompleto.- Falta de cumplimiento del proceso.
1. Realizado.- Genera los productos de trabajo esperados.
2. Administrado.- Proceso y productos administrados y controlados.
3. Establecido.- Proceso definido para la organización y utilizado adecuadamente.
4. Predecible.- El proceso opera dentro de los límites estadísticos establecidos.
5. Optimizado.- El proceso mejora continuamente.

Las organizaciones de software no tienen que seleccionar entre 15504 y su modelo actual. El rol de 15504 es proveer un marco de trabajo en el que los modelos y métodos de evaluación puedan existir de una manera armónica. No está diseñado para ser utilizado solo. La selección radica en decidir si el ajustarse a 15504 es importante para el negocio (¿El negocio compite en el mercado global?, ¿Provee software a algún cliente que requiera 15504?, ¿Existen varios modelos de evaluación en la organización?). De ser así, deberán elegir modelos que se ajusten a 15504.

Compatibilidad entre Modelos

ISO/IEC 15504

- Evaluación de procesos de software.
- En vías de ser estándar.
- Dirección amplia.
- Niveles de capacidad.
- Evaluación de capacidades por proceso individual.
- Guía para realizar la evaluación.
- Toma como referencia ISO 9001 y CMMI.

CMMI

- Modelo de madurez de procesos de software.
- Modelo – estándar de facto.
- Dirección detallada.
- Pasos progresivos (niveles) - Escalonada.
- Categorías de procesos - Continua.
- Guía para institucionalización e implementación.
- Modelo de evaluación será conforme al modelo de evaluación de 15504.

ISO 9001-2000

- Sistema de Gestión de la Calidad.
- Estándar internacional.
- Dirección amplia.
- Un conjunto de requerimientos a ser cubierto.
- No hay lineamientos para su implementación.
- Usado como referencia en actividades de gestión de calidad por CMMI y 15504.

CMMI Modelo Escalonado

Nivel de Madurez

1 – Inicial
Madurez de un proceso caracterizada por resultados impredecibles.

2 – Administrado
Madurez del proceso caracterizada por el desempeño repetible del proyecto. El foco clave del proceso está en actividades y prácticas a nivel proyecto.

3 – Definido
Madurez del proceso caracterizada por mejorar el desempeño del proceso dentro de una organización.

4 – Administrado Cuantitativamente
Caracterizada por mejorar el desempeño organizacional.

5 – Optimizado
Madurez del proceso caracterizada por un desempeño organizacional rápido y configurable, Mejora continua y cuantitativa de procesos.

Áreas de proceso

Ninguna

- Administración de Requerimientos.
- Planeación de Proyectos.
- Monitoreo y Control de Proyectos.
- Administración del Acuerdo con el Proveedor.
- Administración de Configuración.
- Aseguramiento de Calidad de Proceso y Prod.
- Mediciones y Análisis.
- Desarrollo de Requerimientos.
- Solución Técnica.
- Integración de Producto.
- Verificación, Validación.
- Enfoque Organizacional en Proceso.
- Definición de Procesos de la Organización.
- Capacitación Organizacional.
- Administración Integral de Proyectos.
- Administración del Riesgo.
- Análisis de Decisión y Resolución.
- Desempeño de Procesos Organizacionales.
- Administración Cuantitativa de Proyectos.
- Innovación y Despliegue Organizacional.
- Análisis Causal y Resolución.

Modelos Específicos

PSP – Personal Software ProcessSM

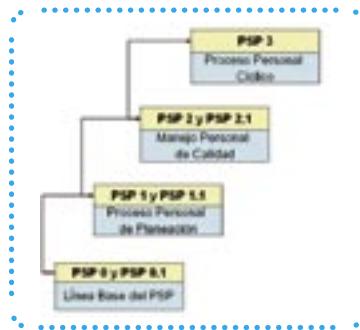
Personal Software Process (PSP) es un proceso diseñado para ayudar a los ingenieros de software a controlar, manejar y mejorar su trabajo. PSP está basado en una motivación: La calidad de software depende del trabajo de cada uno de los ingenieros de software. Debido a que los costos de personal constituyen 70% del costo del desarrollo de software, las capacidades y hábitos de trabajo de los ingenieros determinan en gran medida los resultados del desarrollo de software.

Basado en prácticas encontradas en CMM, el PSP puede ser usado por ingenieros para estructurar y disciplinar el desarrollo de software. El ingeniero de software podrá planear mejor el trabajo, conocer con precisión el desempeño, medir la calidad de productos, y mejorar las técnicas.

PSP puede ser aplicado en:

- Desarrollo de programas.
- Definición de requerimientos.
- Documentación.
- Pruebas de sistemas.
- Mantenimiento de sistemas.

Evolución de PSP



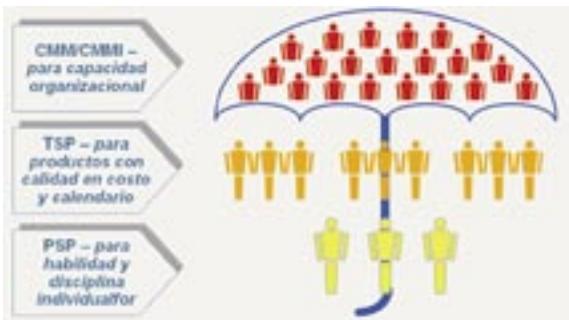
TSP - Team Software ProcessSM

Team Software Process (TSP) es un marco para el desarrollo de software que pone igual énfasis en el proceso, producto y trabajo en equipo. Al igual que PSP, TSP fue propuesto por Watts Humphrey.

TSP se basa en PSP, y se fundamenta en que el software, en su mayoría, es desarrollado por equipos, por lo que los ingenieros de software deben primero saber controlar su trabajo, y después saber trabajar en equipo. TSP le enseña a los ingenieros a construir equipos autodirigidos y desempeñarse como un miembro efectivo del equipo. También muestra a los administradores como guiar y soportar estos equipos.

Estrategia de TSP

- Proveer un proceso sencillo basado en PSP
- Desarrollar productos en varios ciclos. Ciclo de TSP: Lanzamiento, Estrategia, Plan, Requerimientos, Diseño, Implementación, Pruebas, Postmortem
- Establecer medidas estándares para calidad y desempeño
- Proveer definiciones de roles, y evaluaciones de rol y de equipo
- Requiere disciplina de proceso
- Provee guía para manejo de problemas de trabajo en equipo



RUP

El Rational Unified Process (RUP) es un proceso de software desarrollado y comercializado por Rational Software (ahora parte de IBM). RUP está diseñado alrededor de seis mejores prácticas para el desarrollo de software:

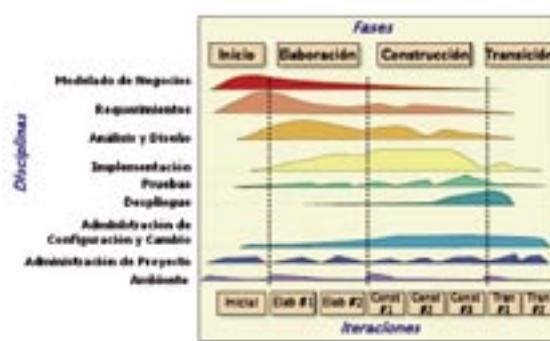
- Desarrollar de manera iterativa.
- Administrar los requerimientos.
- Utilizar arquitecturas basadas en componentes.
- Modelar el software visualmente.
- Verificar la calidad de manera continua.
- Controlar los cambios.

En sí, RUP es una guía que define roles, actividades, flujos de trabajo y lineamientos para ejecutar proyectos de software de acuerdo a estas mejores prácticas.

RUP organiza los proyectos de software en dos dimensiones: la del tiempo y la de las actividades. En base al tiempo, los proyectos se dividen en cuatro fases secuenciales:

- Concepción – Definición de alcance, identificación de riesgos.
- Elaboración – Resolución de riesgos, establecimiento de arquitectura.
- Construcción – Generación del producto.
- Transición – Disponibilidad a la comunidad de usuarios finales.

Las actividades se organizan en nueve diferentes disciplinas que son ejecutadas durante las diferentes fases.



En realidad RUP es un framework (marco de trabajo) que pretende ser personalizado o configurado para organizaciones y proyectos específicos. RUP no se puede aplicar de la misma forma en todos los proyectos de una organización. Es por esto que pretender seguir RUP a través de ir cumpliendo con la lista de artefactos que define, es una estrategia poco efectiva. Lo que las organizaciones deben hacer es entender la razón de ser de RUP – las prácticas citadas anteriormente – y en base a esto aplicar lo que decidan que es conveniente para cada área o proyecto específico.

RUP es una instancia particular del Proceso Unificado, definido por Ivar Jacobson, Grady Booch y James Rumbaugh en el libro "The Unified Software Development Process" de 1998. Adicionalmente existen otras instancias de este proceso, tales como el Proceso Unificado Mejorado (Enhanced Unified Process), el cual agrega soporte multiproyectos y fases y disciplinas para el mantenimiento y retiro de sistemas de software.

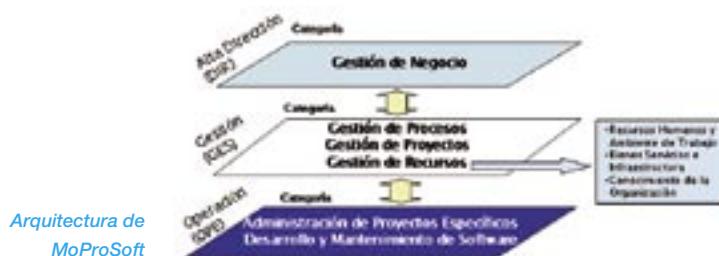
Compatibilidad de PSP y TSP con CMMI

CMM y CMMI se enfocan en la mejora organizacional basada en modelos. TSP contiene prácticas operativas para el desarrollo de equipos. PSP se enfoca en la mejora a nivel individual.

¿Qué hay para la Industria Mexicana?

Méjico quiere y puede darse a conocer con una fuerte industria de software. Las condiciones básicas están dadas: se generan recursos humanos capacitados, se tiene acceso a los equipos y herramientas de desarrollo, el mercado local tiene necesidades lejos de estar satisfechas, el mercado externo de habla hispana no está saturado y la cercanía con los E.U. trae oportunidades de exportación potenciales.

MoProSoft, modelo de procesos para la industria de software mexicana, fue desarrollando pensando en facilitar a las organizaciones dedicadas al desarrollo y mantenimiento de software la adopción de las mejores prácticas reconocidas internacionalmente a través de modelos como SW-CMM, CMMI, TSP, PSP, ISO/IEC 15504, PMBOK y SWEBOK.



Conclusión

Las organizaciones de desarrollo de software, además de entender los principales modelos y procesos existentes, deben considerar varios factores para poder decidir que camino seguir, como: tamaño de la organización, recursos, enfoque en mercado global o local, habilidades, etc.

Las empresas grandes con miras a la exportación, pueden considerar una certificación en CMMI, principalmente niveles 2 y 3, con el objetivo de evaluarse y ganar ventaja competitiva. Las empresas pequeñas y medianas con miras a mejorar los procesos y ser competitivos, pueden adoptar ISO 9001, ya que es un modelo más barato y sencillo. Las PyMEs mexicanas tienen la opción de seguir MoProSoft como camino inicial. Las áreas internas de sistemas pueden adoptar algún proceso específico, además pueden requerir a sus proveedores algún nivel de madurez comparable con CMMI o ISO 15504, o bien, que adopten el mismo proceso específico.

Es recomendable considerar otros modelos que apoyen y complementen la correcta implantación del proceso de software. Como ISO/IEC 9126:2001 enfocado en modelar la calidad de productos de software, o bien, modelos para implementar procesos de mejora como IDEAL. Otros modelos especializados, que pueden ser de gran utilidad son PMBOK (Project Management Body of Knowledge) y SWEBOK (SW Engineering Body of Knowledge).

Para muchas de las organizaciones a nivel mundial, los 90s fueron la década de mejora de procesos. Aunque muchas aprendieron cómo implementar exitosos programas de mejora, otras lucharon sin lograr mejoras duraderas. Estas últimas organizaciones pasarán la siguiente década tratando de ponerse al día, mientras que los líderes emprenden nuevos retos. Las metas relacionadas con procesos en esta década incluyen integración de procesos, armonización, y aceleración. Las organizaciones deben luchar por alcanzar un nivel de calidad que les permita ser competitivas en el mercado global. El punto de inicio es definir la meta, entendiendo la situación actual y las diferentes opciones, para poder así emprender el viaje por el camino correcto.

Referencias:

- Proceso de Desarrollo de Software, Hanna Oktaba, Facultad de Ciencias – UNAM
- ¿Porqué usar MoProSoft como modelo de procesos de referencia?, Hanna Oktaba, www.amcis.org.mx
- Software Engineering Institute – Carnegie Mellon University, www.sei.cmu.edu
- Diplomado en Calidad de Software, AMCIS 2002 – www.amcis.org.mx
- Process Diversity in Software Development – IEEE Software, julio-agosto 2000

REFLEXIONES

El Desarrollo de Software

¿ARTE O INGENIERÍA?

Por Francisco López Lira

"La virtud está en el medio."
—Horacio

Existen en la industria dos visiones aparentemente encontradas sobre lo que debería ser el desarrollo de software, aquella que lo considera una arte y la que plantea que es ingeniería.

La primera considera al desarrollador como un artista, no sujeto a reglas ni métodos, enfatizando la creatividad como la principal virtud. Rechaza los procesos considerando que estos restringen la creatividad.

La segunda visualiza al desarrollador como un ingeniero, que debe utilizar procesos, métodos y técnicas probadas en un ambiente disciplinado y hace énfasis en establecer procesos definidos y controlados.

Aunque ambos enfoques tienen algo de verdad, debemos tener en cuenta algunas consideraciones:

- Arte no equivale a Caos. La idea de Arte está estrechamente relacionada con un sistema de trabajo. Etimológicamente Arte deriva del latín Ars que, a su vez, deriva del griego Tecné y que para Aristóteles era "el uso sistemático del conocimiento para la acción humana inteligente".

- El arte requiere disciplina. Por ejemplo, a Miguel Angel le tomó cuatro años terminar la Capilla Sixtina. Se dice también que Leonardo Da Vinci podía trabajar las 24 horas del día.

- Producción artística no es producción comercial. Los grandes artistas del renacimiento tenían mecenas que les permitían vivir cómodamente hasta que creaban su obra maestra. Por otro lado, el desarrollo de software en el entorno actual implica cumplir con tiempos establecidos, lo cual a su vez requiere de procesos predecibles.

- La creatividad es muy importante, pero no es lo único. Un proyecto de desarrollo de software requiere además: planear, controlar, organizar personas, dirigir, comunicar, coordinar, negociar, interactuar, verificar y validar, por ejemplo. Todo esto requiere de un entorno organizado.

- Lo importante es la gente, lo secundario los procesos. Sin embargo, hay que considerar que la gente no puede funcionar adecuadamente en un sistema caótico. "El sistema siempre ganará" dice Rummel. Por ello, es necesario contar con los procesos adecuados que posibiliten un sistema estable.

- Un mal proceso es peor que ninguno. Los procesos que se utilicen deben ser diseñados a partir del conocimiento de los desarrolladores y de las condiciones de cada organización.

Quizá la virtud yazca en el medio. Podemos utilizar procesos y modelos de procesos procurando crear un entorno organizado para potenciar el valor de las personas y de creatividad.

¹ Geary A. Rummel, "Serious performance consulting", International Society for Performance Improvement, 2004

El autor es fundador y actual presidente de la Asociación Mexicana para la Calidad en Ingeniería de Software (AMCIS).
flopezlira@amcis.org.mx



Aplicando PSP y TSP

QUARKSOFT COMPARTE SU EXPERIENCIA
Por Ricardo Vidrio y Ricardo Domínguez

Una de las experiencias que QuarkSoft, S.C. ha tenido en el desarrollo de software utilizando metodologías como Personal Software ProcessSM (PSP) y Team Software ProcessSM (TSP), ha sido realizar un proyecto para generar un cambio a la nómina a nivel nacional de una dependencia de gobierno.

Cuando QuarkSoft tomó el proyecto, éste ya llevaba un retraso de varias semanas con respecto a la fecha de inicio planeada, aún no se realizaba la fase de análisis de requerimientos y la fecha de terminación, como en muchos otros casos, no era negociable, por lo que para terminar el proyecto en tiempo y forma, el equipo de desarrollo debía ser muy eficiente.

Los problemas principales que enfrentamos fueron los siguientes:

- Falta de toma de requerimientos.
- Muy poca gente conocía la regla del negocio.
- El proyecto de inicio ya contaba con un retraso importante.
- No existía una planeación del proyecto.
- Los participantes por parte del cliente no estaban familiarizados con PSP y TSP.

QuarkSoft decidió tomar esta oportunidad ya que era un claro ejemplo de falta de organización en la administración de un proyecto y era un buen momento para probar si realmente los procesos funcionaban en tiempos de crisis.

Nuestro principal reto era decidir cuáles serían los procesos mínimos a implementar para trabajar bajo esta metodología, tomando en consideración que la mayor parte del equipo no contaba con capacitación en PSP y TSP, requerida para asegurar el control del proyecto.

Toda la información que se genera durante el proceso de lanzamiento es necesaria para evaluar si el proyecto es factible en el tiempo que se requiere.

El proyecto inició con un proceso de lanzamiento como lo plantea TSP. Este lanzamiento consiste de una serie de 9 juntas que se realizan en un periodo de 3 a 4 días con el equipo completo de desarrollo. Los objetivos de cada sesión son los siguientes:

Sesión 1 – Definir objetivos del cliente y del producto a construir.

Sesión 2 – Asignar roles y responsabilidades de cada miembro del equipo.

Sesión 3 – Obtener diseño conceptual del producto, estrategia de desarrollo y lista de principales componentes.

Sesión 4 – Identificar y estructurar actividades necesarias para la construcción.

Sesión 5 – Generar plan de calidad basado en métricas cuantitativas para monitorear la calidad de los componentes creados.

Sesión 6 – Detallar plan de trabajo.

Sesión 7 – Analizar riesgos.

Sesión 8 – Preparar presentación para el cliente con resultados del lanzamiento.

Sesión 9 – Presentar al cliente.

Toda la información que se genera durante el proceso de lanzamiento es necesaria para evaluar si el proyecto es factible en el tiempo que se requiere, además de tener todo un plan de acción detallado a seguir durante el desarrollo del proyecto con metas específicas y, sobre todo, verificables.

Los mecanismos seleccionados para la administración del proyecto fueron los siguientes:

1. Registro del tiempo real invertido por cada ingeniero en el proyecto para compararlo contra el tiempo planeado.
2. Juntas de comunicación entre los integrantes del equipo.
3. Juntas de estatus semanales con el equipo y el cliente.
4. Control y seguimiento cuantitativo del avance de los componentes a nivel individual.

El plan del proyecto contaba con actividades específicas y con tiempos planeados para realizarlas. Los tiempos que se registraron se dividieron en 3 tipos: tiempo real, que es el tiempo que se invirtió al realizar una actividad del plan de trabajo; tiempo de espera, que es el tiempo muerto que un ingeniero tiene por dependencias con actividades de otros ingenieros o por esperar decisiones o actividades por parte del cliente; y, finalmente, el *overhead*, que se define como el tiempo que se tiene que invertir en actividades que no se planearon pero que se deben realizar para terminar el proyecto. Cada semana se convocabía a una junta de estatus con el objetivo de mantener informado al cliente del avance cuantitativo del proyecto.

Se generó un reporte en el cual se incluía el número total de programas o componentes por módulo, la etapa del ciclo de desarrollo en la que estaban cada uno de ellos y el porcentaje de avance de cada módulo como se muestra en la siguiente tabla:

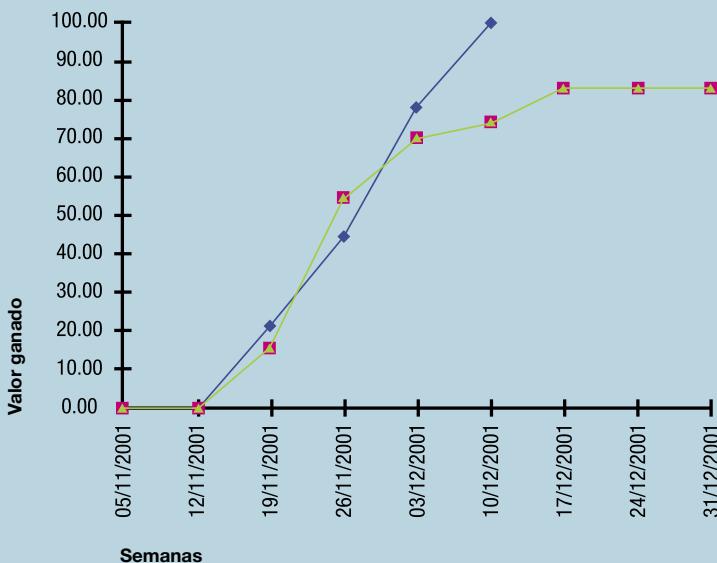
Estatus de Módulos						
Módulo	Número de componentes	Análisis y Codific.	Pruebas	Porcentaje	Nuevos	Faltantes
Altas	10	10	10	100.00	0	0
Terceros	9	9	9	100.00	6	0
Nómina	1	1	1	100.00	1	0
Admon.	43	20	20	46.51	4	23
Rep.	57	21	1	50.00	0	1
Nuevos	18	11	11	61.11		7

La tabla nos muestra de manera sencilla y puntual los módulos a desarrollar dentro del proyecto, el número de componentes de los que consta y las diferentes fases por las que tiene que pasar cada componente, desde el análisis hasta las pruebas necesarias para verificar que efectivamente se cumplen con los requerimientos solicitados por el cliente.

PSP propone la realización de revisiones e inspecciones de los componentes al término de las fases de diseño detallado, codificación y pruebas de unidad. El objetivo de las revisiones e inspecciones es que el ingeniero de software identifique mediante un checklist que contiene los puntos más importantes a revisar, el mayor número de defectos posibles en cada componente. Posteriormente se genera una inspección por un ingeniero que es ajeno a la construcción de ese componente y, apoyado por un checklist de inspección, identifica el mayor número posible de defectos que no se detectaron en la revisión. Esto permite que el producto contenga la menor cantidad de defectos cuando se encuentre en la etapa de pruebas de integración de componentes, con lo cual se pretende reducir los tiempos de las fases de pruebas de integración y de sistema, que regularmente dentro de un proyecto de desarrollo de software son impredecibles y costosas.

Adicionalmente, y con el objetivo de mantener un mayor control del proyecto, se incluyó una gráfica para conocer el porcentaje de avance real del equipo con respecto al plan. La gráfica se genera mediante la consolidación de datos que cada uno de los ingenieros de software registra. La información contenida en la gráfica nos brinda la oportunidad de identificar el avance real del proyecto mediante el concepto de valor ganado. Cada una de las actividades planeadas en la construcción de los componentes tiene asignado un determinado valor ganado. El valor ganado del proyecto se obtiene considerando únicamente todas aquellas actividades que han sido concluidas satisfactoriamente y no se asignan valores parciales por actividades inconclusas. El valor ganado se define como el valor otorgado a cada una de las actividades del plan detallado como un porcentaje de su duración con respecto al tiempo total del proyecto.

Ricardo Vidrio es director de operaciones de QuarkSoft. Es instructor de PSP autorizado por el SEI y coach de equipos TSP. Es egresado del ITESM y tiene una maestría en Ingeniería de Software en la Universidad Carnegie Mellon.



Valor Ganado

En la gráfica se muestran tres diferentes líneas. La línea azul (rombos) nos indica el valor ganado planeado del proyecto y representa el tiempo que llevarán las actividades incluidas en el plan original. Esta línea se vuelve nuestro eje, ya que ahí está indicado el inicio y fin del proyecto. La línea rosa (cuadros) representa el valor ganado real que llevamos semana a semana y que refleja todas las actividades terminadas en el plan de cada ingeniero. Esta nos permite ver de forma cuantitativa el avance real del proyecto. Finalmente, la línea verde (triángulos) nos indica una proyección del tiempo que requeríramos para terminar todas las actividades planeadas en un inicio si tomamos en cuenta el desempeño real del proyecto. La información de la línea verde nos predice el posible retraso que se tendría si no se toman las medidas correctivas necesarias para evitarlo.

Esta información nos permitió saber en todo momento cuál era el estatus real del proyecto, qué se había realizado y lo que faltaba por ejecutar. En los proyectos normalmente no tenemos registros del tiempo que se pierde por falta de definición o por cambios de requerimientos por parte del cliente. Cuando el cliente tuvo conocimiento de esta información, se empezaron a tomar acciones correctivas y posteriormente preventivas para evitar mayores retrasos en el proyecto. Finalmente, el proyecto se logró controlar y se concluyó con dos semanas de retraso, lo cual fue considerado por el cliente como un éxito rotundo.

Los planes individuales son la clave para darle seguimiento al plan general original. Estos planes permiten informar el estatus de cada actividad, revisar constantemente los riesgos importantes del proyecto, reorganizar la carga de trabajo entre los ingenieros en caso de ser necesario, y producir los informes semanales del estatus del proyecto que se le entregan al cliente.

Disciplinas como las planteadas en PSP y TSP nos ayudan a que el desempeño de cada uno de los ingenieros sea mayor, dando la oportunidad a cada uno de planificar y dirigir su propio plan de trabajo e interactuar dentro de equipos autodirigidos que se hacen responsables de sus propios planes. Adicionalmente, el uso de modelos como PSP y TSP no garantizan que el proyecto nunca entre en crisis, pero sí nos proveen de información oportuna que nos permite tomar las decisiones adecuadas para mantener bajo control el proyecto.

La implementación de estos modelos depende del compromiso y disciplina de los ingenieros que intervienen en los proyectos, ya que son ellos los responsables de llevar a cabo todos los procesos necesarios para que la recolección de métricas sea adecuada. Sólo con

métricas confiables se puede crear una base de datos históricos que nos ayude a mejorar las estimaciones de futuros proyectos que puedan ser entregados a tiempo, en costo y con calidad.

PSP y TSP permiten que el proveedor de servicios se convierta en una organización cada vez más madura, ya que el cliente tiene la visibilidad de "cómo" se desarrollan sus proyectos, se incrementa la confianza en las estimaciones basadas en datos históricos y, sobre todo, con la terminación exitosa de los proyectos, se incrementa el ánimo de los equipos de desarrollo.

Para introducir correctamente estos modelos es necesario un apoyo total de los gerentes y directivos responsables de financiar estos proyectos. Es importante que la alta gerencia comprenda que las prácticas de ingeniería de software contenidas en estos modelos generarán mejoras en el proceso de desarrollo de software que se traducirán a mediano plazo en una mayor rentabilidad del negocio. Es vital para la continuidad de este proyecto que la alta gerencia esté completamente convencida de los beneficios que se obtendrán para no cancelar el apoyo al proyecto en caso de un retraso no planeado.

Actualmente QuarkSoft (www.quarksoft.net) está trabajando en el proceso para obtener la certificación CMMI nivel 3 y basa toda su iniciativa del programa continuo de mejoramiento de procesos de software en los modelos PSP y TSP. ☺

Team Software ProcessSM, TSPSM, Personal Software ProcessSM y PSPSM son Service Marks de la Universidad de Carnegie Mellon.

Ricardo Domínguez es líder de proyecto en QuarkSoft con más de nueve años de experiencia en desarrollo de software. Es ingeniero de software certificado en PSP con entrenamiento en TSP. Ricardo es egresado de la Universidad Popular Autónoma del Estado de Puebla (UPAEP).



**Soluciones reales,
flexibles...**

¡Y a tu alcance!

www.deintec.com

ESTIMACIÓN DE PROYECTOS

ENTENDIENDO LAS BASES

Por Rodrigo García

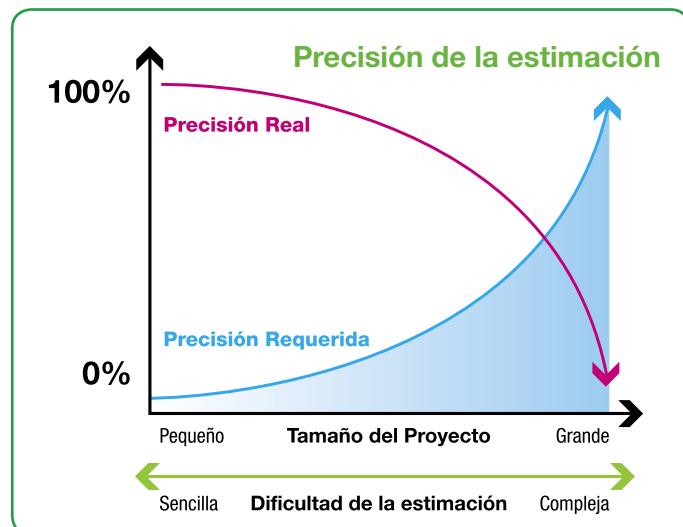
Según Aberdeen Group, 90% de los proyectos de software se liberan tarde, mientras que Gartner comenta que 50% de los proyectos de software excede su presupuesto inicial. Existen muchas razones para esto, pero una de las más comunes es que no se realiza una estimación adecuada.

La información que se espera obtener a través de la estimación en un proyecto de software es la siguiente:

- Tamaño del producto
- Esfuerzo requerido
- Duración del proyecto
- Recursos necesarios
- Calidad esperada

Como ya mencionamos, las malas estimaciones —o las no estimaciones— son una de las principales causas de fracaso en los proyectos. Esto típicamente sucede por no contar con un proceso estructurado para la estimación, lo cual hace casi imposible definir equipos de trabajo de magnitud adecuada y plazos de cumplimiento razonables. La situación empeora cuando no se cuenta con información de proyectos anteriores. Todo esto hace que la estimación sea uno de los temas que más preocupa a los administradores de proyectos de software.

Un problema inherente a la estimación es que conforme aumenta el tamaño del proyecto, disminuye la precisión de los resultados obtenidos (ver gráfica 1).

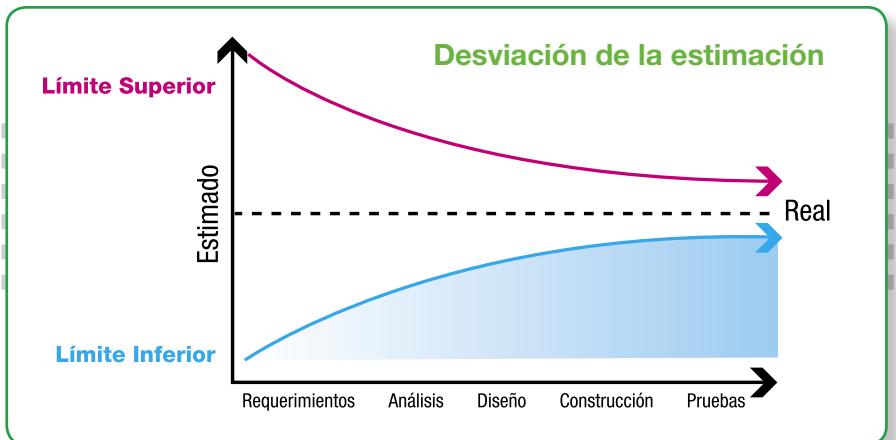


Gráfica 1. Conforme aumenta el tamaño del proyecto disminuye la precisión de la estimación

Entonces, ¿qué debemos hacer para que nuestra estimación sea precisa, exitosa y cumpla los objetivos que buscamos? Comencemos por entender las dos características principales de cualquier estimación: que sea explicable y que sea revisable.

- **Explicable.**- Es común que al preguntar a las personas cómo llegaron al resultado de una estimación, ellas mismas no lo puedan explicar. Una buena estimación debe de tener suficiente claridad y modularidad para que sea entendible y cualquier persona pueda comprender los valores de todos sus componentes.

Rodrigo García es Technical Solutions Manager en Softtek Information Services en Monterrey, N.L., donde es responsable de la estimación de propuestas para proyectos de desarrollo. Rodrigo es Ingeniero en Sistemas Computacionales del ITESM y actualmente cursa el MBA en Global e-Management en la misma institución.



Gráfica 2. La desviación de la estimación disminuye conforme avanza el proyecto.

- **Revisable.**- A medida que el proyecto va avanzando, la cantidad de información que se tiene va siendo cada día mayor, y la misma nos ayuda a ir corrigiendo la estimación. Es importante definir límites superiores e inferiores para la estimación, dependiendo de la etapa en la que se encuentre (ver gráfica 2).

Gracias a estos límites, y con la ayuda de cierta información histórica, podemos ir revisando nuestro proyecto para saber si seguimos dentro del estimado. Tomemos como ejemplo un proyecto de 500 horas hombre. Nuestra información histórica nos indica que el levantamiento de requerimientos lleva cerca de 20% del tiempo, el diseño 30%, el desarrollo 25%, y las pruebas, junto con la puesta en producción, 25% restante. Imaginemos que habíamos planeado el levantamiento de requerimientos en 100 horas, pero en realidad se lleva 120. En este momento es cuando se tiene que revisar el estimado, ya que sería incorrecto asumir que solamente vamos con 20 horas de retraso, cuando en realidad llevamos un retraso de 20%. En este caso, tendríamos que revisar el proyecto y agregar 20% al esfuerzo en todas las fases. Nuestro nuevo estimado sería de 600 horas.

La duración definida para un proyecto rara vez surge como resultado de una estimación, sino que es negociada entre el cliente (ya sea interno o externo) y el equipo de desarrollo. Suele suceder que se busque acortar el tiempo en función de aumentar el número de recursos, pero sabemos que esta relación no es del todo correcta. De aquí la existencia de frases como "no porque una persona pinte una casa en diez días debemos concluir que diez personas lo harán en un día". Otro error común es no tomar en cuenta las limitaciones de recursos disponibles. Debemos aprender a estimar en base a los recursos que se tienen, no a los que quisiéramos tener.

Sentando estos antecedentes, expliquemos dos métodos comunes para estimar de proyectos de software.

COCOMO

Desarrollado en la Universidad del Sur de California con Barry Boehm como principal autor. Su nombre significa "Constructive Cost Model" (modelo constructivo de costo). El modelo original data de 1981, pero en el 2000 se liberó COCOMO II, una versión ajustada para el tipo de proyectos realizados actualmente. Este modelo se basa en la siguiente información para realizar una estimación:

- **Tamaño del sistema.**- Normalmente medido en KSLOCs o miles de líneas de código —la K representa un millar, y SLOC es el acrónimo de "source lines of code"—. Esta información se puede obtener en base a sistemas existentes, aunque también existen métodos para estimar el tamaño de un sistema en base a su funcionalidad (ver puntos de función más adelante).

- **Factores de escala.**- Contabilizan las economías o deseconomías de escala en el proyecto. Cuando un proyecto exhibe economías de escala, significa que si el tamaño del producto se duplica, el esfuerzo necesario es menos del doble, mientras que cuando hay deseconomías de escala, el esfuerzo requerido aumenta a un ritmo mayor que el tamaño del producto. Entre los factores que mejoran las economías de escala están la cohesión del equipo y la madurez de procesos en la organización. Sin embargo, en general los proyectos de software exhiben deseconomías de escala.

- **Multiplicadores de esfuerzo.**- Características del proyecto que afectan el esfuerzo requerido para desarrollarlo. Algunos ejemplos son el nivel del personal disponible, las herramientas utilizadas o la complejidad del lenguaje.

Transformamos la Calidad en TI

Servicios de Consultoría, Capacitación y Evaluación, con experiencia y reconocimiento a nivel mundial en la implantación de prácticas innovadoras de Administración de Proyectos e Ingeniería de Software para la Industria de Tecnologías de Información.



CMMI®
PMBOK®
ISO9000
CMMI SM

No porque una persona pinte una casa en diez días debemos concluir que diez personas lo harán en un día.

La fórmula para estimar esfuerzo es del tipo:

$$\text{Esfuerzo} = A \times \text{Tamaño}^E \times \prod ME$$

Donde *A* es una constante, *E* es un indicador de economías de escala (calculado en base a los factores de escala), y *ME* es el promedio de valores de los multiplicadores de esfuerzo.

La fórmula para estimar la duración es del tipo:

$$\text{Tiempo} = C \times \text{Esfuerzo}^F$$

Donde *C* es una constante y *F* es un valor derivado de los factores de escala.

Existen valores por defecto para las constantes, pero lo ideal es calibrarlas en base a la información histórica de proyectos en la organización.

En general, COCOMO es una buena opción para realizar estimaciones “macro”, con el objetivo de asignar presupuesto a un proyecto, o decidir si vale la pena desarrollar/reemplazar un sistema.

Puntos de Función

Otra alternativa para dimensionar un sistema es utilizando “function points” (FPs) o puntos de función. Este método fue creado por Allan Albrecht, y es una manera de cuantificar la capacidad funcional de un sistema de software. Los pasos para determinar los puntos de función en un sistema son:

1. Identificar los elementos generadores de funcionalidad. Estos pueden ser de cinco tipos: transacciones de entrada, transacciones de salida, transacciones de búsqueda, archivos lógicos internos y archivos de interfase externa.
2. Calificar cada elemento y asignarle un valor. Cada elemento se califica como “simple”, “promedio” o “complejo”, de-

pendiendo de la cantidad de datos que maneje; posteriormente, asignar una cantidad de puntos a cada elemento. Por ejemplo, una búsqueda de complejidad baja corresponde a 4 puntos, mientras que una salida de complejidad alta corresponde a 7. Existen tablas con los valores correspondientes a cada caso (ver las referencias).

3. Sumar los puntos asignados a cada elemento para generar un total conocido como “puntos de función sin ajustar” (UFP por sus siglas en inglés).

4. Determinar el factor de complejidad técnica (TCF). Identificar las características técnicas relevantes al proyecto, tales como requerimientos de desempeño, confiabilidad e interacción con otros sistemas. Con esto se calcula un factor de ajuste cuyo valor va de .65 a 1.35.

5. Estimar los puntos de función ajustados. $FP = UFP * TCF$

Con esto ya podemos determinar el tamaño de un sistema. Pero ahora, ¿cómo podemos obtener el esfuerzo y duración del proyecto? Imaginemos que vamos a desarrollar un sistema de 100 FPs, y que usaremos un ciclo de vida tradicional en cascada con las siguientes etapas: análisis de requerimientos, diseño, construcción, pruebas y liberación a producción.

Basándonos en registros históricos, podemos determinar la relación que hay entre la cantidad de FPs de un sistema y el esfuerzo requerido para cada etapa. Por ejemplo, podríamos concluir que el análisis requiere 2 horas-hombre por cada FP, y como nuestro sistema tiene 100 FPs entonces podemos esperar que el análisis se lleve 200 horas-hombre. La tabla 1 muestra el ejemplo completo.

Modelo de estimación de esfuerzo

Etapa	Esfuerzo por FP	Esfuerzo Esperado
Requerimientos	2	200
Diseño	3	300
Construcción	5	500
Pruebas	4	400
Liberación	2	200
Total		1600

Tabla 1

Este ejemplo sólo llega al nivel de detalle de etapas. Un modelo real debe contemplar todas las actividades a realizar dentro del proyecto, incluyendo juntas de revisión y otras tareas que rara vez se toman en cuenta en una estimación. La idea de este método es que sirva de base para la planeación de actividades y asignación de recursos en un plan de trabajo detallado.

Conclusión

Hemos delimitado la importancia y dificultades de la estimación de proyectos de software. Mencionamos dos métodos de estimación: uno orientado a generar valores agregados para la toma de decisiones, y otro orientado a generar información detallada para la planeación. En próximos artículos hablaremos más a fondo sobre ellos. Es un hecho que ningún modelo va a dar una alta precisión en un inicio, poco a poco deben irse ajustando en base a la información obtenida cada que se realiza un proyecto. G

Referencias:

Gramajo, E. et al. Combinación de Alternativas para la Estimación de Proyectos de Software. CAPIS - Centro de Actualización Permanente en Ingeniería de Software.

Boehm, Barry et al. Software Cost Estimation with COCOMO II. Prentice Hall, 2000.

Longstreet, David. "Fundamentals of Function Point Analysis". www.ifpug.com/fpafund.htm



Microsoft Learning

DESEO: nuevas oportunidades

Conviértete en un experto en la tecnología Microsoft.

Realiza tu carrera en tres semestres y al finalizar cada uno obtendrás una certificación Microsoft.

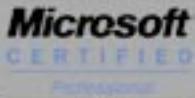


Microsoft CERTIFIED

Solution Developer

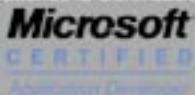
Primer Semestre
Módulo I

Obtienes la Certificación



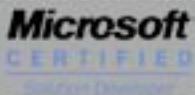
Segundo Semestre
Módulo II

Obtienes la Certificación



Tercer Semestre
Módulo III

Obtienes la Certificación



- Clases por las tardes y sesiones dos veces por semana o sabatinas
- Laboratorios prácticos
- Exámenes de certificación incluidos
- Bolsa de trabajo
- Planes de crédito educativo
- 12% de descuento al pagar de contado

Inversión por semestre:

Inscripción: \$3,990*

5 Mensualidades: \$2,990*

*IVA incluido

12 meses sin intereses Bancomer

Microsoft CERTIFIED

Systems Engineer

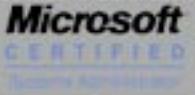
Primer Semestre
Módulo I

Obtienes la Certificación



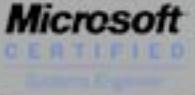
Segundo Semestre
Módulo II

Obtienes la Certificación



Tercer Semestre
Módulo III

Obtienes la Certificación



Te invitamos a nuestras sesiones informativas todos los miércoles a las 19:00 hrs

Confirma tu Asistencia: (55) 5488 0827 / (55) 5488 0673

www.intersoftware.com.mx/microsoftlearning

programas@intersoftware.com.mx

¿SQA? ¡SÁLVESE QUIEN PUEDA!

Por Mariana Pérez-Vargas y Elizabeth Almeraz

El desarrollo de software es una de las actividades más complejas que se realizan hoy en día. Es considerado como algo abstracto ya que el producto que se construye no es algo físico, es una caja negra. Podemos decir sin temor a equivocarnos, que los requisitos que pedimos para desarrollar software son mínimos y por lo tanto éste tiene un alto grado de incertidumbre, el cual aumenta si en el proyecto se presentan los siguientes problemas:

- Falta de planeación
- Falta de administración del proyecto
- Falta de soporte ejecutivo
- Cambios no controlados
- Expectativas poco realistas
- Altos costos
- Falta de recursos
- Requerimientos incompletos

La pregunta es, ¿cómo se puede tener mayor certidumbre en el desarrollo de software y obtener productos con calidad, con el presupuesto planeado y en el plazo estimado?

Los problemas que enfrentamos en el desarrollo de software tienen solución a través de la adopción de procesos, lo cual consiste en que las organizaciones usen buenas prácticas que han demostrado ser efectivas y que definen de manera precisa cómo se construye el software. En el mercado existen diferentes modelos y estándares que pueden ser adoptados por la industria de TI: CMMI, ISO15504, MoProSoft, ISO9000, PMBoK, Trillium y Bootstrap, por mencionar algunos. Todos estos de alguna manera consideran y recalcan la importancia de establecer

prácticas de aseguramiento de calidad en los proyectos de desarrollo de software.

Aseguramiento de Calidad

Actualmente, muchas organizaciones de TI, así como áreas internas de sistemas, sin la necesidad de establecer un programa de mejora o seguir todas las prácticas de un modelo o estándar de referencia, están adoptando la práctica de aseguramiento de calidad de software para incrementar la calidad de sus productos, y lograr que el desarrollo de software sea más predecible.

Hasta aquí la historia parece ir muy bien, sin embargo, a pesar de que las organizaciones, desde el punto de vista competitivo, reconocen el valor del SQA (Software Quality Assurer / Advisor), en la práctica este rol es muchas veces subvalorado, incluso por aquéllas donde se han establecido ciertas prácticas. Se cuestiona el valor agregado que puede aportar a los proyectos, siendo un recurso que no está involucrado al 100% en los mismos. En ocasiones, se visualiza al SQA más como un enemigo y una barrera para realizar el trabajo asignado, que como el aliado que puede avisarnos sobre los peligros latentes que se encuentran en el proyecto al no seguir adecuadamente un proceso o al fijarnos expectativas que se encuentran fuera de lo humanamente alcanzable. Esta visión resta importancia y respeto al papel que este rol desempeña, por lo cual es importante reivindicarlo para que pueda mostrar sus bondades al resto de la organización y una revisión de aseguramiento de calidad se convierta en algo que nos brinde un valor agregado más que en una pesadilla que no nos deje conciliar el sueño en la noche.

Mariana Pérez-Vargas es Directora General de Avantare y Lead Assessor Certificada y Autorizada por el SEI. Cuenta con amplia experiencia para proporcionar consultoría estratégica a diferentes empresas desarrolladoras de software, áreas de desarrollo de software y organismos gubernamentales. Pionera en el área de calidad y procesos, entre sus principales logros está haber sido responsable de coordinar exitosamente el programa de mejora en la primera organización en México que se evaluó en CMM.



Desarrollando
para una producción
más EFICIENTE

Para lograr una reivindicación adecuada del SQA, es necesario contestar las siguientes preguntas: ¿qué es el aseguramiento de calidad?, ¿quién es un SQA?, ¿cuáles son las actividades que realiza?, y ¿para qué las realiza?

Contestando a las preguntas, iniciaremos diciendo que el aseguramiento de calidad es el conjunto de actividades desarrolladas dentro del proceso de producción de software, necesarias para garantizar que una organización obtiene un determinado nivel de calidad en el producto o que cumple con los requisitos establecidos.

Haciendo Aliados

El secreto para que un SQA sea visualizado como un rol que aporta valor, es la creación de sinergia entre los equipos de trabajo y él o ella. De esta manera se estarán logrando varios objetivos, como por ejemplo establecer un ambiente de confianza y cooperación para realizar el trabajo de ambas partes, y así crear una visión compartida y objetivos comunes.

El establecimiento de esta visión y objetivos comunes debe hacerse para cada nivel representado en la organización.

Una de las principales tareas del SQA es la de proporcionar guía y asesoría a los miembros de los equipos sobre el uso y aplicación de los procesos.

Un SQA es el rol en una organización que se encarga de revisar y auditar los productos y actividades para verificar que éstos cumplen con los procesos aplicables al proyecto y los estándares establecidos. Proporciona a la gerencia la visibilidad del estado en que se encuentran los procesos y los productos de los diferentes proyectos.

Las actividades del SQA no se limitan a realizar auditorías y revisiones; una de las principales tareas del SQA es la de proporcionar guía y asesoría a los miembros de los equipos de trabajo sobre el uso y aplicación de los procesos descritos en la organización, al nivel de sus proyectos. Así pues, podemos decir que el SQA realiza principalmente las actividades de auditar/revisar los proyectos y asesorar sobre el uso o aplicabilidad de los procesos. Adicionalmente, brinda soporte para facilitar la adopción de prácticas, monitorear la adecuada implantación de las mismas, proporcionar entrenamiento específico y facilitar la comunicación intergrupal.

¿Y para qué realiza todas estas actividades? Para evaluar la administración y buen desarrollo del proyecto, previniendo posibles riesgos o incidentes que se pudieran presentar durante el ciclo de vida del proyecto.

De tal forma que la mejor manera encontrar aliados y mostrar el valor de los SQA's es buscando estrategias que puedan ayudar a los sectores seleccionados a reconocer el valor del SQA en sus tareas cotidianas y del día a día.

Una estrategia para que los diferentes niveles de la organización vean al SQA como un aliado es establecer acciones que muestren cómo el papel del SQA, en su función de asesor, los ayuda a realizar su trabajo mediante la implantación de prácticas y procesos, coadyuvando con ello a alinear el trabajo diario con los procesos definidos, sin que éstos últimos se perciban como una actividad extra. Así se pueden empezar a identificar actividades sencillas y prácticas que ayudarán para mejorar las relaciones del grupo de SQA con los diferentes niveles de la organización, tales como: los ingenieros de software, los administradores de proyecto, la gerencia media y la gerencia alta.

Este enfoque ayudará al SQA a crear un ambiente de sinergia que mejore su credibilidad, reivindique su función y coadyuve a mejorar la productividad de los equipos, la calidad de su trabajo y los productos generados.

Somos una empresa especializada en el desarrollo de software a la medida, nuestra metodología se basa en la aplicación de las mejores prácticas para el desarrollo y el uso de herramientas CASE para desarrollo y auditoría del software.

También contamos con modelos o productos base, los cuales son fácilmente adaptables a las necesidades específicas de su empresa, algunas de esas soluciones incluyen aplicaciones para Facturación, Punto de Venta, Almacenes, CRM, Control de obras, activos fijos, Biblioteca, Videoteca, etc.



Contamos con especialistas para ofrecer soluciones con distintos manejadores de bases de datos, lenguajes de programación y plataformas, desde AS 400 hasta Pocket PC.

Torres Adalid # 707
despacho 802
Col. Del Valle, México, D.F.
C.P.03100
Tel: (55) 5687 5052

www.imexsoft.com.mx

“Cuando la calidad es vital, son necesarias algunas revisiones independientes, no porque no confiemos en la gente, sino porque son humanos.”

—Watts S. Humphrey

Managing the Software Process. SEI Series
in Software Engineering.

Estas son algunas sugerencias de lo que el SQA puede hacer para obtener el apoyo y la cooperación de los diferentes grupos en la organización:

Con el área operativa:

- Asesorarlos en sus funciones con respecto a la calidad.
- Ser guía y mentor en prácticas de ingeniería de software y técnicas de verificación y validación como revisiones entre colegas, pruebas, simulaciones, etc.
- Ayudar a los ingenieros de software a visualizar cómo sus actividades influyen en la calidad del producto final.
- Sensibilizarlos y mostrarles cómo los procesos les ayudan en la ejecución de sus tareas diarias.

Con la administración del proyecto:

- Asesorarlos en aspectos relacionados con su proyecto.
- Recompensarlos por sus esfuerzos, dándoles visibilidad correspondiente.
- Señalando las desviaciones de los procesos y/o productos, pero aplaudiendo los logros.
- Capacitándolos en los procesos y procedimientos para mejorar la calidad en sus proyectos.

- Escuchándolos y escalando sus sugerencias de mejora a los procesos.

Con la gerencia media:

- Mantenerlos informados sobre estado de los proyectos a su cargo.
- Asesorarlos en las funciones que corresponden a su nivel y realizar las revisiones que les correspondan de acuerdo a los procesos definidos por la organización.
- Proporcionarles visibilidad sobre los problemas comunes de aseguramiento de calidad en sus áreas y/o de otras áreas de la organización.
- Sugiriendo mejoras a prácticas administrativas y de ingeniería de software en sus proyectos.

Con la alta gerencia:

- Vigilando el apego a procesos, procedimientos, estándares y políticas definidos en la organización.
- Escalando problemas que deban ser atendidos a este nivel
- Reportando los resultados de las diferentes áreas.
- Brindando visibilidad sobre la calidad de los productos.
- Manteniéndolos informados sobre la operación del negocio y el apego a los procesos.

En la actualidad, se cuentan con datos de la industria que pueden ayudar a las organizaciones a estimar el esfuerzo que se invierte en las actividades de aseguramiento de calidad, como por ejemplo:

- El esfuerzo total invertido en actividades de aseguramiento de calidad por los proyectos representa entre 3 y 5% del esfuerzo total invertido en el mismo.
- El número de proyectos que un SQA puede atender varía entre 5 y 10 proyectos, dependiendo de la madurez del SQA, la complejidad de los proyectos, el tiempo (parcial o completo) dedicado a actividades de SQA y la madurez de los procesos revisados en los proyectos.

Finalmente, podemos concluir que si la organización pone en marcha esta práctica de aseguramiento de calidad, aunada con otras prácticas sencillas como verificación y validación, esto puede significar auténticos ahorros para todos los proyectos, eliminando defectos, previniendo riesgos, visualizando problemas en sus etapas tempranas y ayudando a los proyectos a potenciar los beneficios de los procesos. ☺

Elizabeth Almeraz es pionera en México en la realización de actividades de Aseguramiento de Calidad del Software (SQA), participando en el primer grupo de SQA que ayudó a lograr una evaluación CMM en México. Actualmente es consultor de Avantare y especialista en las áreas de técnicas de verificación de productos de software y mejora de procesos para la industria de TI.

Llevate el kit de programación
Por \$32 dlls Más iva

Cuando se es El # 1 en el mundo, ¿Qué se hace para repetirlo?



REINVENTING SOFTWARE PROTECTION & LICENSING

Presentamos HASP HL - La nueva generación en seguridad basada en hardware para proteger su software y su propiedad intelectual. Implemente la más robusta seguridad del líder indiscutible, Aladdin. Protéjalo una vez. Protéjalo como debe ser.

Compruébalo Usted mismo con el demo en linea "Protección de Software 1-2-3", o solicite su Kit de Desarrollador en HaspHL.com.mx/Encore.

- Robusta protección de software basada en algoritmos AES y RSA.
- Implemente innovadores modelos de licenciamiento independientemente del esquema de protección
- Administración de Licencias Monousuario o Multiusuario
- Integración de API y herramientas intuitivas y fáciles de utilizar
- Llave altamente confiable, compacta y multiplataforma

* Aladdin es el proveedor #1 en el mercado de llaves de autenticación y licenciamiento de software tanto en 2002 como en 2003.

-- Boletín IDC # 31432, 2004



Distribuidor en México: Seguridad Integral para Software e Internet, S.A de C.V.
5208-7472, 5207-8222, 01800-021-4277, 01800-552-8300
www.sisoft.com.mx sisoft@sisoft.com.mx



Una realidad que no se puede ocultar es el hecho de que el desarrollo y la codificación de software gozan de complejidad. A pesar de su relativa sencillez, resulta difícil entender todo este grupo de tecnologías convergentes que llamamos J2EE (Java 2 Enterprise Edition). Esta plataforma, que tiene como objeto codificar sistemas de software de nivel empresarial, goza de gran complejidad. EJBs de sesión, de entidad y de mensajes, JSPs, Servlets, Web Services, JDBC, HTML, código SQL, etc., es complicado figurar en la mente todo un sistema software basado en J2EE. En ocasiones, no resulta muy claro el lugar que ocupa cada uno de estos elementos, y resulta aún más difícil explicarlo a un grupo de desarrolladores que está por introducirse en esta plataforma.

Las Capas Conceptuales

En este artículo les recomendamos cómo organizar sistemas basados en J2EE, utilizando una estrategia de organización en cinco capas conceptuales. Esta organización es un medio abstracto cuyo objetivo es el designar un orden lógico a cada elemento, agruparlo dentro de una capa y lidiar con la complejidad dentro de cada una. A fin de cuentas el objetivo es hacer más claro el desarrollo en J2EE.

Las capas en cuestión son: Presentación, Aplicación, Servicios, Dominio y Persistencia. Algunas de estas capas pueden no estar presentes en un sistema. Dependiendo de las características particulares de éste (el dominio del problema), de los marcos de trabajo que se empleen y de la estrategia con que es abordado el desarrollo, alguna de estas capas no serán codificadas. Teniendo esto último en mente, veremos cada una de ellas.

Presentación.- Compuesta por todos los elementos relacionados con la interfaz de usuario, tales como botones, ventanas, campos de textos, etiquetas, etc., si hablamos de clientes Win32, AWT, SWT o Swing; HTML, JSP, JavaScript, aplicaciones Macromedia Flash si se tiene a un navegador Web por cliente; Xlets o Midlets, y sus correspondientes controles de interfaz de usuario si se trata de una aplicación inalámbrica basada en J2ME (Java 2 Micro Edition). Ésta es la capa de interfaz de usuario, el rostro visible y, en cierta forma, estético del sistema. Sin lugar a dudas, la parte más importante para el usuario final del sistema.

Aplicación.- Responsable del control del flujo de trabajo del lado del cliente y de componentes de IU reutilizables. Éstos se caracterizan por presentar un acoplamiento con la capa inmediata siguiente (servicios). Para el caso del control del flujo de trabajo, se puede emplear HttpSession o el marco de trabajo Apache Struts. En el caso de componentes, por ejemplo, se puede tener un control heredado de un "Combo Box" que despliega los nombres de clientes que están almacenados en la base de datos, y estos son obtenidos a través de un método de un EJB (capa de servicios) que es ejecutado en el constructor de dicho componente. Dependiendo del sistema que estemos construyendo, puede ser que no necesitemos administrar el flujo de control ni se empleen componentes, por tanto, esta capa no estaría presente en tal situación.

Servicios.- Si la capa de presentación es la más importante para el usuario final, la de servicios lo es para los desarrolladores. Sin lugar a dudas, la más importante e indispensable de un software, el API (Application

CAPAS CONCEPTUALES PARA SISTEMAS J2EE

Por Ramés Rodríguez

A fin de cuentas, el objetivo es hacer más claro el desarrollo.

Ramés Rodríguez es desarrollador de J2EE desde el 2002. Su especialidad es el empleo de software libre aplicado a Java y el diseño de arquitecturas orientadas a objetos. Actualmente se desempeña como Arquitecto de Sistemas en la Universidad Loyola del Pacífico.

Programming Interface) del sistema. Es el punto de entrada a la lógica de negocios, la cual se encuentra encapsulada a través de un conjunto de componentes de servicio.

En el caso de J2EE, los componentes de esta capa típicamente se implementan como EJBs de Sesión sin estado (stateless). La razón de que no tengan estado es que no guarden información de clientes particulares, y estén completamente desacoplados de éstos, además de que una instancia puede atender múltiples peticiones brindando un mayor desempeño. Estos Beans acostumbran servir como fachadas que proveen servicios utilizando información de las entidades de la capa de dominio.

Esta capa requiere un gran esfuerzo para las pruebas unitarias, ya que es imperativo asegurarse de que estos componentes respondan adecuadamente a las diferentes condiciones posibles de ejecución. De tenerlos, es aquí donde residirían los servicios web (web services) del sistema.

Dominio.- Si se desean los beneficios de un diseño orientado a objetos (el ser reutilizable y de fácil mantenimiento), esta capa debe estar presente dentro del sistema que se deseé desarrollar. En ella está contenido nuestro modelo conceptual, mejor conocido como modelo de dominio –de allí el nombre de la capa.

Este modelo consiste en todas aquellas entidades (facturas, productos, órdenes de compra, etc.) y actores (clientes, bancos, empleados, etc.) que componen y/o interactúan en los procesos del negocio, y son creadas a raíz de la abstracción de sus atributos significativos para el sistema. Este modelo típicamente es ilustrado con diagramas de estructura, como el diagrama de clases de UML. Una característica de estas entidades es su capacidad para asociarse con

otras entidades o, incluso con sí mismas. Es posible que un sistema no cuente con capa de dominio; en estos casos la capa de servicios estaría codificada de tal manera que dentro de los mismos métodos de sus objetos se accederían directamente los datos en donde estos se almacenen, por ejemplo usando JDBC en el caso de una base de datos relacional. Esta opción tiene la desventaja de que es de difícil mantenimiento y poco reutilizable, pero puede ser factible para sistemas de vida corta o para aquellos que no tendrán cambios significativos a través del tiempo, incluso para prototipos.

Persistencia.- Al hablar de persistencia hablamos de almacenamiento de información en un medio no volátil (bases de datos, archivos de texto plano, etc.). Los elementos de esta capa típicamente funcionan como objetos de acceso a datos o DAOs (Data Access Objects), proveyendo métodos para leer y almacenar los datos de los objetos de negocio. Es por esto que la existencia de esta capa típicamente está condicionada a la existencia de la capa de dominio. Si no hay un modelo de dominio presente, no hay razón para hacerlo persistente.

Incluso con la presencia de un modelo de dominio, podría no ser necesario codificar esta capa cuando se empleen marcos de trabajo tales como persistencia manejada por el contenedor (Beans de tipo CMP), JDO (Java Data Objects) o Hibernate. Al codificar las entidades en la capa de dominio, estos marcos de trabajo se encargarán de la persistencia de cada una de ellas, liberando a los programadores de esta responsabilidad.

La persistencia manejada por el Bean (EJB de Entidad de tipo BMP) es un caso clásico de implementación de



esta capa, ya que existe un modelo de dominio que debe ser persistente, y para ello los programadores codifican clases con sentencias SQL cuyo objeto es crear, modificar, eliminar y consultar registros en una tabla.

Orden de Desarrollo

Ya describimos las cinco capas recomendadas. Ahora hablemos sobre cómo nos pueden ayudar para definir el orden en que se debería desarrollar un sistema. ¿Cómo es eso? Sigue que estas capas presentan una característica adicional: cada una de ellas depende de la capa inmediata inferior. Ya sea en tiempo de compilación o de ejecución, los tipos en las capas superiores dependerán de los tipos de las capas inferiores. Por ejemplo, volviendo al ejemplo del componente que muestra los nombres de clientes, para que la capa de aplicación, en donde se encuentra codificado el componente, muestre esta información, necesita invocar un método que se encuentra en la capa de servicios. Es esta dependencia la que marca el orden de desarrollo.

De acuerdo con Floyd Marinescu, autor del libro EJB Design Patterns, lo primero que debe ser codificado es la capa de dominio, seguida por las de persistencia, servicios y por las de interfaz de usuario (aplicación y presentación). La de dominio es la primera debido a que aquí residen las entidades que deben estar listas y compiladas para poder ser utilizadas por la que capa de servicios, que es la que contiene la lógica de negocio; adicionalmente, estas entidades típicamente son sencillas de implementar. Una vez lista la capa de dominio, ésta desencadena ordenadamente el desarrollo de las otras capas. Es recomendable tener en operación la capa de servicios lo más pronto posible para lograr el desarrollo de la interfaz de usuario en paralelo con las demás capas.

Una importante observación, asumiendo que se está creando un sistema bajo un proceso de desarrollo iterativo e incremental: no es necesario desarrollar totalmente y con lujo de detalle cada capa. Sólo deben ser definidas aquellas piezas de código directamente relacionadas con el o los casos de uso que se estén llevando a cabo en la iteración actual.

Aunque pareciera ser que esta estrategia es la única forma posible de abordar el desarrollo, existe otra que he concebido a través de mi experiencia. En los proyectos de J2EE en los que he estado involucrado, es una constante dividir el desarrollo en dos partes: el cliente y el servidor. Con el objeto de que el desarrollo del cliente se lleve en paralelo con el desarrollo del lado del servidor, primero se debe codificar la capa de servicios. Una vez concluida esta capa, los dos equipos de desarrollo (cliente y servidor) pueden iniciar su trabajo de manera independiente uno del otro. ¿Qué es lo que se gana? El equipo de desarrollo del cliente no debe esperar a que se termine la programación del lado del servidor (las capas de servicio, dominio y persistencia) para iniciar su trabajo, como sucede con la otra estrategia.

En este punto surge una pregunta obvia: ¿cómo se puede terminar la capa de servicios sin haber terminado la de dominio y las otras inferiores? La respuesta es sencilla: la capa de servicios sólo tiene los esqueletos vacíos de cada uno de sus métodos, los cuales retornan datos de prueba; la capa de servicios

realmente no está terminada. Por ejemplo, empleando el componente de nombres de clientes citado anteriormente, suponiendo que estos datos son obtenidos a través del método `getNombresDeClientes()`, que devuelve un `java.util.Collection` de objetos de tipo `String`, entonces la implementación de este método creará una instancia de una clase que implemente la interfaz `Collection` - como un `ArrayList` - y le agregará nombres de clientes arbitrarios. La razón de ser del componente es desplegar un conjunto de nombres que se localizan en la base de datos. Sin embargo, para efectos de tener una capa de servicios funcional lo más rápido posible, se omite el acceso a la base de datos (capa de persistencia), y en su lugar enviamos datos de prueba.

Para que esto funcione se necesita cierto lujo de detalle al diseñar cada uno de los métodos de la capa de servicios. Siendo más específicos, si durante el flujo de trabajo de diseño de la iteración actual, se especificaron cada uno de los métodos que deben tener nuestras fachadas, entonces serán estos métodos los primeros a codificar. Se necesita saber el nombre definitivo del método, los

valores de retorno, el tipo y número de parámetros que recibe, y las excepciones que arroja. Estos datos son los que conforman el esqueleto de los métodos.

Conclusiones

Esta práctica nos permite, por un lado, lidiar con la complejidad de J2EE a través de un enfoque de organización y, por el otro, nos ofrece dos estrategias para iniciar el desarrollo de un Sistema J2EE. Empero, esta no es la única forma en que puede ser organizado un sistema J2EE, ni tampoco son todos los enfoques para empezar a desarrollarlo. Sin lugar a dudas, la forma más efectiva es aquella que comprendamos mejor y con la que nos sintamos cómodos. Pero si no se tiene un enfoque ni estrategia, entonces esto es un buen punto de partida. ☺

Referencias:
Marinescu, Floyd. EJB Design Patterns.
Wiley, 2002.

*Desarrolle aplicaciones de misión crítica
basadas en su propio y exclusivo
conocimiento del negocio, de forma
independiente de la tecnología.*

GENEXUS™
THE FIRST INTELLIGENT TOOL

GXflow

GXplorer

GXportal

GXquery

www.genexus.com

Descargue una versión de prueba gratuita:
www.genexus.com/trial

Ciudad de México - México
Calle Leibnitz N° 20, desp. 801
(52 55) 5255 4733

Certificación de Calidad para PyME

¿QUÉ TAN NECESARIA ES?

El desarrollo de software con calidad es una necesidad de las empresas productoras de software del país. La competencia de la industria extranjera, como India y China, así como las exigencias de algunos clientes estadounidenses, hacen que la calidad se haya convertido en algo más que una palabra de moda o una urgencia por obtener certificaciones prestigiosas. Para tener ventaja competitiva, un desarrollador de software debe cuidar no sólo la calidad de su producto, sino la calidad en su proceso de desarrollo. Por calidad del producto entendemos software que sea válido y verificado, que sea resistente a fallas, y que tenga el desempeño requerido según especificaciones. Por calidad del proceso entendemos aquellas prácticas que hacen que el tiempo de los desarrolladores sea utilizado de manera óptima, que aquellos procesos repetitivos sean debidamente documentados, y que en general hacen que el ciclo de desarrollo de un proyecto sea tan suave y sin incidentes como sea posible.

Es esta búsqueda de calidad lo que ha llevado a la creación y adopción de modelos y metodologías de calidad, como CMM o SPICE. De hecho, los modelos de calidad se han vuelto tan importantes que una empresa desarrolladora de software de gran tamaño no puede concebirse si no tiene algún tipo de certificación o avalúo de parte de algún organismo internacional. La capacidad de estas empresas —Softtek, por ejemplo— para exportar software hacia Estados Unidos está a menudo supeditada a poseer algún tipo de certificación de calidad internacional. No queda ninguna duda de que estas empresas deben incluir y mantener en sus programas de aseguramiento de calidad de software una certificación internacional, tanto para optimizar el uso de sus recursos como para poder competir en el mercado de exportación.

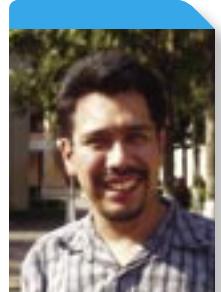
¿Cuál, entonces, debe ser la postura de una pequeña o mediana empresa productora de software con respecto a las certificaciones? Y, ¿de qué manera deberían apoyar las comunidades y grupos de software del país a estas empresas para la obtención de certificaciones? Como ya es sabido, las PyMEs tienen recursos limitados. En particular, resulta difícil asignar roles para las actividades de un modelo como CMM entre los pocos miembros del personal de una pequeña empresa, que seguramente ya juegan distintos roles en el proceso de desarrollo. Pero esta aparente paradoja no significa que estas empresas no deban aspirar a la adopción de prácticas y procedimientos que mejoren su ciclo de desarrollo de software. En el caso extremo en el que la empresa desee entrar al mercado de

exportación, es probable que no haya opción y se tenga que aspirar a una certificación formal, con el costo que eso representa. Pero existen otras opciones. Se puede aspirar a implantar un modelo de calidad con el objetivo de optimizar procesos y adoptar mejores prácticas, sin optar por pagar el costo de hacer oficial la certificación. O se puede implantar un modelo de calidad regional, adaptado a la realidad de la PyME. En estos casos, lo que se busca es la mejora de los procesos de desarrollo de software y un mejor producto final, con el fin de optimizar costos y tener un producto confiable que consolide a la empresa en el mercado.

Algo que es importante mencionar es la postura que se está permeando en la comunidad de software sobre el mercado potencial de las PyMEs desarrolladoras de software. Si bien es cierto que se desea que el país compita contra las potencias productoras de software, como la India, también es cierto que existe un gran mercado nacional que, paradójicamente, consume software producido por empresas extranjeras. La visión presentada en varios foros de la industria es precisamente esa: la PyME debe empezar por abarcar este amplio mercado nacional antes de preocuparse por el mercado de exportación. Es la opinión de varios miembros de la industria que éste enfoque al mercado nacional crearía un círculo virtuoso en la economía del país.

Esto nos lleva de nuevo al tema de las certificaciones. Según esta visión, no es una prioridad la certificación de metodologías de calidad de software para las PyMEs. Lo verdaderamente importante es la adopción cabal de estas metodologías en los procesos de desarrollo. En todo caso, es de suma importancia la elaboración de estándares locales y, ¿por qué no? de certificaciones nacionales que avalen estándares de calidad acordes a las necesidades de los consumidores locales y las características únicas de las pequeñas y medianas empresas. Es por esta razón que los esfuerzos por crear estas certificaciones locales deben ser apoyados tanto por los grupos de ingeniería de software, así como por las instituciones académicas y la industria en general, y que es de suma importancia la adaptación y creación de modelos de calidad de software para la pequeña y mediana empresa. Estos modelos serán, entonces, una antesala para optar, en su momento, por una certificación de nivel internacional, y un paso necesario en el camino hacia un país que sea competitivo en la industria del software mundial. **G**

- Raúl A. Trejo



El Dr. Raúl A. Trejo es profesor investigador de Sistemas de Información en el Tec de Monterrey, Campus Estado de México. Sus áreas de especialidad incluyen Ingeniería de Software y Algoritmos Computacionales. El Dr. Trejo gusta de participar en proyectos que involucren el trabajo cercano con estudiantes, como el Proyecto Principia de Integración Curricular del Tec, o el Concurso FIRST de Robótica de la NASA. El Dr. Trejo ha participado en conferencias como el Americas Conference on Information Systems y publicado artículos en la revista Artificial Intelligence.

Ingeniería de Software

DESARROLLAR Es MUCHO MÁS QUE PROGRAMAR

¿Qué conocimientos se necesitan para desarrollar software? Esta pregunta atormenta a muchos, desde jóvenes estudiantes hasta personal de recursos humanos. Hay quienes piensan que basta con saber programar en algún lenguaje. Sin embargo, todos aquellos con experiencia en este medio reconocen que para desarrollar software de calidad, más que de programadores se requiere de ingenieros de software.

Yentonces, ¿qué conocimientos debe tener un ingeniero de software? Precisamente esto es lo que busca responder el *Software Engineering Body of Knowledge (SWEBOK)*, un proyecto auspiciado por la IEEE para lograr un consenso mundial de lo que es esta disciplina y el lugar que tiene junto a otras ingenierías. El SWEBOK define diez *Knowledge Areas (KAs)* o áreas de conocimiento.

se generan modelos que sirven como “planos” para la construcción. Típicamente se divide en dos tipos:

- Diseño arquitectónico - Describe la estructura y organización de alto nivel de un sistema. Identifica los componentes e interfaces entre éstos.
- Diseño detallado - Describe individualmente cada componente con suficiente detalle para ser construido.

Esta área concentra una gran cantidad de conocimiento. Para empezar, el diseño de software requiere entender a fondo principios como la abstracción, acoplamiento, cohesión, descomposición y encapsulación, ya que son la base para diseñar sistemas robustos. También es necesario saber resolver aspectos prácticos como la persistencia de datos, sistemas distribuidos, peticiones concurrentes, manejo de eventos, recuperación a fallas, etc. Por último, un diseñador que no quiera “reinventar la rueda” cada vez que se le presente un problema, debe estar familiarizado con “patrones”, soluciones documentadas a problemas comunes.

Construcción de Software
Esta área de conocimiento se refiere a la creación de software útil a través de la programación, depuración (*debugging*), pruebas unitarias e integración de componentes. La construcción lida con la creación y aplicación de algoritmos para la resolución de problemas, así como su implementación utilizando algún lenguaje de programación. Todo esto se debe hacer buscando minimizar la complejidad y cumpliendo estándares para que el código generado sea entendible y extensible por otros, además de que esté optimizado para consumir la menor cantidad de recursos posible.

ÁREAS DE CONOCIMIENTO (KAs) DE LA INGENIERÍA DE SOFTWARE

- | | |
|---|--|
| <ul style="list-style-type: none"> • Requerimientos • Diseño • Construcción • Pruebas • Calidad • Mantenimiento | <ul style="list-style-type: none"> • Administración de la Configuración • Administración (de Proyectos) • Proceso • Herramientas y Métodos |
|---|--|

Requerimientos de Software
Los requerimientos de software expresan las necesidades y restricciones que debe satisfacer un producto para contribuir a la solución de un problema real. Esta área de conocimiento considera la obtención, análisis, especificación y validación de los requerimientos, así como el rol que juegan dentro del proceso de desarrollo de software. Un especialista en requerimientos tiene conocimiento y experiencia en técnicas para obtener, cuantificar, negociar, clasificar, priorizar, modelar, documentar y validar los requerimientos de software. Además debe saber administrar adecuadamente los cambios a éstos.

Diseño de Software

El diseño juega un rol clave en el desarrollo de software ya que es donde

Pruebas de Software

Las pruebas de software consisten en la verificación dinámica del comportamiento real de un programa comparado con su comportamiento esperado en un conjunto finito de casos de prueba seleccionados de un dominio de ejecuciones típicamente infinito. Las pruebas se realizan para evaluar la calidad de un producto a través de la detección de fallas en éste. Sin embargo, las pruebas de software han evolucionado para dejar de ser consideradas como algo que comienza sólo hasta que la programación termina, con el limitado propósito de detectar fallas. Es aceptado que las pruebas deben abarcar el proceso completo de desarrollo, que su planeación comienza durante las primeras etapas del proceso de requerimientos, y que los planes y procedimientos de prueba se deben desarrollar y refinar durante el ciclo completo de desarrollo.

Las pruebas pueden ser de diferentes tipos, ya sea por su alcance (unitarias, integrales, de sistema) o su objetivo (funcionalidad, confiabilidad, desempeño, regresión, aceptación, beta, etc.). Para esto se utilizan diferentes técnicas como tablas de decisión, análisis de fronteras, máquinas de estados, y la experiencia misma. Por último, para cuantificar la calidad de un producto de software es necesario entender métricas como la densidad de defectos y la evaluación de confiabilidad.

Calidad del Software

El área de conocimiento de calidad se enfoca en la aplicación de técnicas estáticas para evaluar y mejorar la calidad del software. Esto difiere del acercamiento utilizado en pruebas, donde las técnicas utilizadas son dinámicas, ya que requieren la ejecución del software. El área de conocimiento de calidad involucra los subprocesos de aseguramiento de calidad, verificación, validación, revisión y auditoria. Además considera

Los métodos de ingeniería de software establecen una estructura para sistematizar las actividades con el objetivo de aumentar las posibilidades de éxito.

tópicos como la clasificación de defectos, control estadístico de calidad, modelos de predicción y análisis de tendencias.

Mantenimiento de Software

El mantenimiento se refiere a las modificaciones a un producto de software previamente liberado para prevenir fallas (preventivo), corregirlas (correctivo), mejorar su desempeño (perfectivo) o adaptarlo a cambios en el ambiente (adaptativo). Históricamente no se le ha dado tanta atención al mantenimiento como al desarrollo de software. Sin embargo, esto está cambiando debido a que las empresas buscan sacar el mayor jugo posible a sus productos existentes. Algunos temas clave en el mantenimiento son la reingeniería, análisis de impacto, pruebas de regresión, y *outsourcing* del mantenimiento. Además hay que tener en cuenta que el proceso a seguir en el mantenimiento es diferente al que se sigue para el desarrollo, por lo que involucra actividades y artefactos diferentes.

Administración de la Configuración del Software (SCM)

La configuración de un sistema se refiere al conjunto de elementos de hardware y software que lo forman. SCM es la disciplina de identificar la configuración en distintos puntos del tiempo con el propósito de controlar los cambios a ésta, manteniendo su integridad y rastreabilidad durante el ciclo completo de vida del software. SCM va más allá del simple control de versiones, y requiere saber identificar los elementos de configuración, definir un proceso de control de cambios, auditar y reportar el estatus de la configuración, y administrar la integración y liberación del sistema completo.

Administración de la Ingeniería del Software

Esta área de conocimiento es lo que típicamente llamamos Administración de Proyectos o *Project Management*. Consiste en la aplicación de actividades administrativas —como la planeación, coordinación, medición, monitoreo, control y reporte— para asegurar que el desarrollo y mantenimiento de software se lleva a cabo de manera sistemática, disciplinada y cuantificable. Entre los tópicos más importantes de esta área de conocimiento están la planeación de proyec-

tos, estimación de esfuerzo, asignación de recursos, administración de riesgo, manejo de proveedores, manejo de métricas, evaluación, y cierre de proyectos.

Proceso de Ingeniería de Software

Cada área de conocimiento considera un proceso para las actividades técnicas y administrativas que deben realizarse para adquirir, desarrollar, mantener y retirar software; éste es considerado como un primer nivel de procesos. Adicionalmente existe un segundo nivel, o meta-nivel, que se enfoca en la definición, implantación, evaluación, mejora y administración del cambio de los procesos de primer nivel. A éste es al que se refiere el área de conocimiento de proceso de ingeniería de software.

Herramientas y Métodos de Ingeniería de Software

Las herramientas permiten la automatización de tareas repetitivas y bien definidas, habilitando al ingeniero de software para que se concentre en los aspectos creativos del proceso. Existen una gran cantidad de herramientas para asistir todas las áreas de conocimiento, desde la administración de requerimientos hasta las pruebas automatizadas.

Los métodos de ingeniería de software establecen una estructura para sistematizar las actividades con el objetivo de aumentar las posibilidades de éxito. Esta área de conocimiento se enfoca en los métodos que abarcan múltiples KAs, ya que los que son relativos a una sola área de conocimiento se incluyen en el área correspondiente. Los métodos pueden aplicar técnicas heurísticas (informales), formales, y basadas en prototipos.

Estas son las diferentes áreas del conocimiento de la ingeniería de software. Si quieras conocer más sobre estos temas, no te pierdas futuras ediciones de esta revista, ya que nuestros artículos ahondarán en cada uno de ellos.

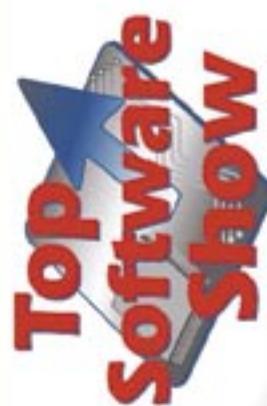
Más información:
Guide to the SWEBOK
www.swebok.org

**TODOS LOS DESARROLLADORES DE SOFTWARE DEMOSTRANDO, COMERCIALIZANDO Y
ASESORANDO A EMPRESARIOS DE MICRO,
PEQUEÑAS Y MEDIANAS EMPRESAS.
RESERVA TU ESPACIO.
NO TE PIERDAS ESTA OPORTUNIDAD.
TEL. 5536 4120.
contacto@mayer-project.com.mx
ventas@mayer-project.com.mx**



...Creando Puntos de Encuentro

AGOSTO 2005
¡YA VIENE!
W.T.C. CIUDAD DE MEXICO
EL PRIMER
TOP SOFTWARE SHOW





El tiempo de respuesta por parte del administrador o líder de proyecto tiene un impacto directo en la entrega de soluciones. Esta situación es muy común en cualquier empresa, y como en todas, es necesario estar listos para tratar de solucionar estos problemas en tiempo y forma. Un sistema de servidores blades correctamente configurado y administrado, nos permite atacar de forma rápida y sencilla este tipo de problemas, y en base a estos beneficios, nos facilita justificar la inversión en nueva tecnología.

La mayor parte de las organizaciones de TI se encuentran en la problemática de hacer más con menos, pero, ¿en qué consiste un sistema de servidores blade? ¿Cómo saber si es una solución que se adecúa a las necesidades de cada empresa? ¿Qué opciones existen en México?

Sistema de Servidores Blade
Un sistema ideal de servidores blades se compone de servidores blades, almacenamiento compartido, red compartida y un sistema de administración centralizada. Los equipos blades son servidores ultra compactos que contienen procesador, memoria y almacenamiento propio. Cada uno de estos servidores se inserta en un chasis, como libros en una estantería. A través del chasis los servidores comparten ventiladores, energía eléctrica, conexiones de fibra canal, conexiones



SERVIDORES BLADES

HACER MÁS CON MENOS

Por Ariel García

Son las seis de la tarde y llega nuestro líder de pruebas a comunicarnos que uno de los servidores que están usando presentó una falla y necesitan reinstalar el sistema operativo. Además, se requiere de un equipo extra para poder liberar la nueva versión del programa en cuestión, pues es necesario un equipo más robusto.

de red, unidad de disco floppy, CD-Rom, etc. Dependiendo de la configuración de nuestro sistema de blades, podemos brindar acceso mediante una sola conexión de distintas tecnologías —red local, red SAN— a todos los servidores blades del chasis. Los beneficios de esta tecnología son evidentes para cualquier persona encargada del mantenimiento y administración de los equipos. El trabajo de cableado de red y/o fibra y el consumo de energía disminuyen. La densidad del espacio físico usado se maximiza.

Otro componente importante de un sistema de blades es la herramienta de administración de los mismos. A través de ella podemos llegar a configuraciones que realicen la instalación y configuración de los servidores de forma automática, que únicamente requieren la inserción de los servidores en el chasis. Con el software de administración podemos realizar copias o imágenes de los servidores de producción, esto nos brinda flexibilidad y protección en las configuraciones de nuestros servidores, sobre todo en situaciones de liberación de nuevas versiones o aplicación de parches críticos a sistema operativo o aplicaciones. Esto incrementa la seguridad y disponibilidad de nuestros servicios.

Dependiendo del fabricante, se pueden construir configuraciones avanzadas de sistemas de servidores blade. Estos pueden proporcionarnos soluciones de

alta disponibilidad y/o de procesamiento simétrico a gran escala que nos permiten migrar de sistemas propietarios RISC/SMP a tecnología nueva de alta escalabilidad y bajo costo. Estos beneficios reducen de forma dramática la administración y los costos de pertenencia del equipo. La funcionalidad de esta tecnología nos permite adaptarnos a las necesidades del negocio, incrementando la disponibilidad y los tiempos de respuesta en caso de fallas o nuevas necesidades de equipo cómputo.

Blades a la Medida

Existen escenarios muy claros en los cuales debemos evaluar la adquisición de un sistema de servidores blades:

- Se renovará la planta de servidores actuales del centro de cómputo.
- Se tiene un proyecto nuevo que demanda la compra de equipo y requiere escalabilidad a futuro.
- Se tiene un problema de espacio físico del centro de cómputo. Esto demanda la consolidación de servidores, sin comprometer la disponibilidad de los servicios.
- Se construirá un nuevo centro de cómputo que requerirá de escalabilidad y alta disponibilidad en su infraestructura.

En lo personal, considero que si tenemos planeada la adquisición de cinco o más servidores durante el año en curso, debemos considerar esta tecnología.

Ésto y más en:



HP Pavilion zd8000 Notebook PC

Esta línea de computadoras portátiles de HP es la inversión ideal para trabajo en oficina y entretenimiento en casa. Viene preinstalada con Windows XP Professional y el Service Pack 2, para mayor protección en contra de ataques vía red, los molestos pop-ups y virus recibidos por e-mail.

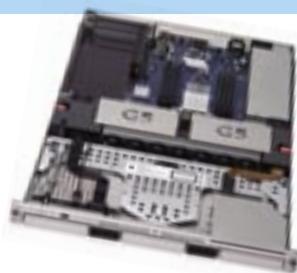
Las opciones de personalización varían, pero se puede elegir un procesador Pentium 4 a 3.2GHz con Hyperthreading, escalar hasta 1GB de RAM, disco duro de 100GB y elegir una unidad DVD-RW/R + CD-RW con soporte para discos de doble capa, y una tarjeta de video ATI Radeon X600 con 265MB de memoria. Además, tiene puertos ethernet, USB 2.0, Firewire, S-video y módem de 56K. Por si fuera poco, las bocinas integradas son Harman-Kardon, y junto con la pantalla Brightwide de 17 pulgadas, son ideales para disfrutar de la versión 2005 del Windows Media Center, que da la posibilidad de sintonizar dos canales de televisión simultáneos, además de estaciones de radio de FM.

Por si fuera poco, incluye software para edición de video y traslación a DVD, además de un lector de tarjetas de memoria 6 en 1. Con una HP Pavilion zd8000, tanto diversión como productividad están garantizadas.

Apple Xserve G5

El servidor Xserve G5 de Apple monta dos procesadores PowerPC G5 a 2.0GHz, que entregan más de 30 gigaflops de poder de procesamiento por sistema, 60% más que el Xserve G4. El procesador G5 de 64-bit, ofrece un desempeño sin paralelo para aplicaciones basadas en UNIX. El Xserve G5 incluye un nuevo controlador de sistema con hasta 8GB de memoria PC3200 EEC, tres modulos seriales para unidades ATA que brindan hasta 750Gb de almacenamiento, hardware RAID interno opcional, dos puertos PCI-X y dos puertos Ethernet Gigabit para alto desempeño en trabajo en redes.

El Xserve G5 es ideal para soluciones de impresión profesional, gestión de grupos de trabajo, stream de video, aplicaciones de bases de datos, aplicaciones gráficas de alto desempeño y montaje de sitios web, así como gestión de servidores de correo electrónico. El sistema Mac OS X Server viene preinstalado, e integra software de código abierto y sus estándares con herramientas fáciles de usar, para ofrecer a sus usuarios lo mejor de dos mundos.



Iomega Mini USB Drives



Los medios de almacenamiento portátil de Iomega no sólo han elevado su capacidad hasta 1GB, sino que además han crecido en funcionalidad. Ahora, estos versátiles cartuchos de memoria flash soportan conectores USB 2.0, que incrementan la velocidad de escritura hasta 7.0MBps, pero lo más interesante de esta nueva generación es la incorporación de una tecnología denominada Active Disk, que permite transferir aplicaciones al dispositivo para que corran desde el mismo, eliminando problemas en caso de que alguna computadora no tenga instalada determinada aplicación. Esta característica sólo funciona en aplicaciones para PC, y respeta todos los copyrights de los desarrolladores, pues las aplicaciones no se pueden copiar desde el medio hacia otra máquina. La línea también incluye los Micro USB drives, con capacidades desde 64MB y precios que van de los \$64 hasta los \$149 dólares.

01 The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad

Michael A. Cusumano
Free Press, Marzo 2004

“Si el negocio del software fuera como los demás, éste libro no sería necesario.”

Con estas palabras comienza el libro “The Business of Software ...”, que fue uno de los más sonados del 2004 en el área de negocio/TI. A través de esta obra, Michael Cusumano nos explica que hay varios aspectos de la industria de software que la hacen única, y por lo tanto requieren estrategias y modelos de negocio únicos.

El libro reflexiona sobre diversos temas como diferentes modelos de negocio, métricas financieras, segmentación de mercado, mejores prácticas y outsourcing. Cusumano también define ocho factores que se deben tomar en cuenta para evaluar una empresa de software.

Sin embargo, el tema central de “The Business of Software ...” es el de los modelos de negocio, donde el autor plantea tres opciones para esta industria: ser una empresa de productos, de servicios o un híbrido. Cusumano describe las características de cada uno, sus fortalezas, debilidades y cuál puede ser el más apropiado dependiendo de la etapa en que se encuentra una empresa, así como de la situación económica. También se incluyen diversos casos de estudio con valiosas lecciones que aprender.

En general, consideramos que esta es una excelente lectura para cualquier persona involucrada en la industria, principalmente los ejecutivos. Muy probablemente después de leerlo lleguen a la conclusión de que, a diferencia de lo que muchos piensan, en esta industria es más importante el modelo y estrategia de negocio que la tecnología.

02 Agile Software Development Ecosystems

Jim Highsmith
Addison-Wesley, Marzo 2002

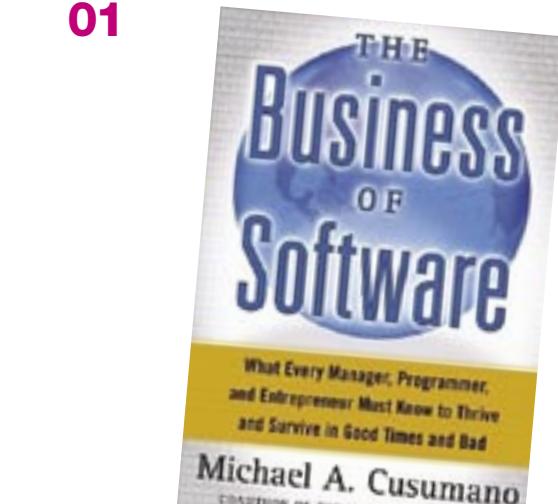
Y dado que hemos hablado bastante sobre procesos de desarrollo, creemos adecuado incluir un libro relacionado con este tema. Atendiendo al gran interés que hay en América Latina y el mundo sobre el “movimiento ágil” y sus diferentes métodos, les recomendamos el libro “Agile Software Development Ecosystems” de Jim Highsmith.

Más que una receta para implementación, este libro es un viaje a través de la filosofía y actitud de los principales impulsores de las metodologías ágiles. El autor hace uso de su experiencia y la de colegas como Kent Beck, Martin Fowler y Alistair Cockburn para

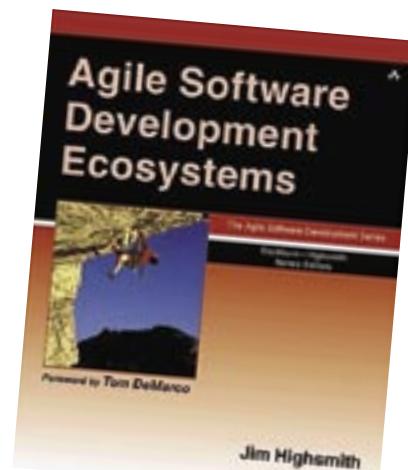
hablar sobre la agilidad en un sentido económico y de negocios. Highsmith hace uso de la teoría del caos para ilustrar la necesidad de balancear el orden y el desorden en un mundo de cambio constante, explicando porqué es mejor reemplazar el control por la libertad con disciplina. Todo esto se analiza bajo un énfasis recurrente en la gente y la manera en que piensan y se comportan. El mensaje central es que es la gente —y no los procesos— quién te salvará, pero necesitas darle la oportunidad.

En el libro se explican principios como la necesidad de entregar valor al cliente, promover la colaboración, confiar en la gente, ser adaptable y hacer las cosas de la manera más sencilla posible. Posteriormente se definen uno por uno los principales métodos ágiles como Extreme Programming (XP),

01



02



SCRUM, Adaptive Software Development (ASD), Feature Driven Development (FDD), Lean Development y otros. Se incluye poca información detallada sobre los procesos, ya que es un libro de principios más que de prácticas. El objetivo no es que después de leer esta obra el lector ya pueda “hacerse ágil”, sino que tenga la suficiente información para decidir si hacerse ágil es lo adecuado para su proyecto u organización. La implantación se maneja en otros textos.

Una queja que puede haber respecto a este libro es que resulta bastante repetitivo en cuanto a las ideas que maneja. Esto es comprensible, ya que es un esfuerzo para sumergir al lector en todo lo que es el “movimiento ágil”. Además, la narrativa tiene un ritmo adecuado para mantener la fluidez y evitar hacerlo aburrido.

**Adquiera,
Convierta y
Retenga más
clientes con
WebTrends 7**



Campaing Performance

Search Engine Results

Customer Self-Service

WebTrends 7
SOFTWARE + ON DEMAND

Commerce Analysis

Audience Segmentation

Navigatio Analysis

Content Effectiveness

WebTrends®
RELENTLESS ABOUT RESULTS

COMPLETE WEB ANALYTICS FOR BETTER RESULTS

Distribuido en México por:

Abits
S O F T W A R E
01 (55) 5273-7078 y 5272-1122
www.abits.com

■ INDEX

DIRECTORIO

**TENEMOS UN ESPACIO
RESERVADO PARA TI**

Si deseas anunciarte contáctanos
en el (55) 5239 5502 o en
ventas@softwareguru.com.mx

<i>Anunciante</i>	<i>Páginas</i>	<i>Sitio</i>
Abits Software	47	www.abits.com
AMCIS	17	www.amcis.org.mx
ARTech	40	www.genexus.com/mx
Avantare	31	www.avantare.com
Deintec	29	www.deintec.com
EXPOCOMM	F3	www.expocomm.com.mx
IBM	F4	www.ibm.com/mx
Imexsoft	35	www.imexsoft.com.mx
Intersoftware	33	www.intersoftware.com.mx
Itera	07	www.itera.com.mx
Mayen PM	43	www.mayen-project.com.mx
MGE	09	www.mgeups.com
Microsoft	F2-1	www.microsoft.com/mexico
Software Guru	11	www.softwareguru.com.mx
Sisoft	37	www.sisoft.com.mx
Ssistemas	15	www.ssistemas.com

Desarrollador Cinco Estrellas

BUSCA TU ESTRELLA

Si quieras formarte como desarrollador en la plataforma .NET, un excelente punto de partida es el programa de capacitación Desarrollador Cinco Estrellas (DCE) ofrecido por Microsoft. Bajo esta iniciativa, Microsoft pretende brindar entrenamiento y capacitación en los principales productos y servicios de esta plataforma. El programa DCE está diseñado para permitir a los suscriptores incrementar sus conocimientos y habilidades de forma progresiva, en una serie escalonada de etapas. Las principales ventajas del programa son que es gratuito y que la mayoría del material está disponible en español y portugués.

DCE está dirigido a:

- Desarrolladores profesionales que adopten .NET sin importar su experiencia previa y quieran actualizar sus conocimientos a este nuevo software.
- Estudiantes universitarios que deseen incorporar en su currículum los conocimientos necesarios para desarrollar aplicaciones con la nueva generación de tecnologías .NET.

Durante la participación en este programa, los suscriptores realizan una serie de evaluaciones que les permiten ir avanzando de nivel, obteniendo las estrellas correspondientes hasta obtener la quinta, que los califica como desarrollador experto en .NET.

El programa ofrece diversos beneficios. Con sólo registrarse se tiene acceso gratuito a las guías de estudio y software para entrenamiento. Los desarrolladores con al menos una estrella son listados en el directorio de desarrolladores de MSDN, el cual sirve como referencia curricular y para que las empresas que buscan personal puedan contactarlos. Posteriormente, conforme se van obteniendo estrellas, se van ganando ventajas adicionales, además del conocimiento, claro está.

Este programa tiene más de un año de haber sido lanzado, y las estadísticas generadas hasta el momento de ésta publicación son las siguientes:

País	Total de registrados	1a Estrella	2a Estrella	3a Estrella	4a Estrella	5a Estrella
Argentina	28.454	2.800	881	225	49	17
Perú	12.552	2.307	432	124	74	50
México	15.422	2.181	399	129	45	19
España	7.590	1.450	411	163	45	19
Colombia	9.311	1.297	330	139	74	20
Ecuador	4.191	1.057	261	98	57	7
Venezuela	6.904	1.029	161	78	46	22
Chile	5.528	682	168	56	10	5
Otros	10.627	548	116	37	14	4
Uruguay	1.518	313	71	31	6	-
Costa Rica	1.836	230	50	26	21	5
Bolivia	2.097	225	46	19	8	-
República Dominicana	1.260	214	39	17	9	5
Total por estrella	107.290	14.333	3.365	1.142	458	173

Para inscribirte o conocer más sobre este programa, visita: www.microsoft.com/latam/msdn/comunidad/dce/

■ HUMOR

```
# include <stdio.h>
int main (void)
{
    int i;

    for (i=1; i <= 500; i++)
        printf("No hakeare el website de la escuela.");

    return 0;
}
```



La exposición más grande de Telecomunicaciones y TI's en México



La nueva generación de negocios en telecomunicaciones y TI's

Visite los pabellones especializados
e internacionales



Conozca el Programa de Conferencias
más especializado



El pre-registro ya
está en línea
**¡Registrese
HOY mismo!**
www.expoconnectioncenter.com.mx

www.expocomm.com.mx

Organizada por:



Reed Exhibitions

Apoyado por:



Organismos de apoyo:



Para mayor información:

1087-1664 • 5340-2322 • 01 800 000 2322

conferencias@ejkrause.com

conferencias@connectioncenter.com.mx

IBM® ibm.com/ondemand/mx

TLAXCALA ON DEMAND

EN EL ESTADO DE TLAXCALA, LOS CIUDADANOS DEMANDABAN LA AUTOMATIZACIÓN DE PROCESOS ADMINISTRATIVOS JUNTO CON LA SEGURIDAD Y CONFIABILIDAD DE SUS DATOS. AHORA, CON IBM INFORMIX LOS TRÁMITES SON MUCHO MÁS CONFIABLES Y FÁCILES DE REALIZAR.

ON DEMAND BUSINESS™

IBM, el logo de IBM, e-business y el logo de On Demand Business™ son marcas registradas de International Business Machines Corporation. © 2004 IBM Corporation. Todas las otras marcas y productos mencionados son marcas registradas de las compañías que las producen. Todos los derechos reservados.