¿Por qué fracasan la mayoría de los proyectos de Software?

Por: Ing. Raydelto Hernández Perera

Gran impacto nos causó leer las estadísticas publicadas por el **Standish Group***, las cuales revelan que sólo el 12% de todos los proyectos de software a nivel mundial son entregados a tiempo y dentro del presupuesto planificado.

Las técnicas tradicionales de ingeniería se han adaptado al campo de desarrollo de software precisamente para proveer herramientas que permitan gestionar los proyectos, administrar los requerimientos y riesgos, así como crear métricas confiables para hacer estimaciones acertadas de tiempo y costo; sin embargo, ha creado gran asombro ver las estadísticas antes mencionadas a pesar de que la ingeniería del software existe hace varias décadas.

Esta realidad ha sido motivo de estudio durante años. Capers Jones, al igual que otros conocidos autores, coinciden en que gran parte del problema radica en el optimismo excesivo en la estimación de tiempos y costos.

La experiencia ha demostrado que este alto índice de fracaso en los proyectos de software está relacionado a fechas de entrega inalcanzables dadas sin utilizar métrica alguna para llegar a una estimación eficiente aproximada. Las empresas maduras en desarrollo de software son atinadas a la hora de hacer estimaciones sobre un proyecto debido a que, a lo largo de su vida como empresa, han desarrollado métricas eficientes para medir la duración y costo de un producto tomando como punto de partida los requerimientos del cliente.

En la actualidad, los requerimientos del software cambian constantemente. Esto así, porque las empresas cambian, y junto con ellas debe de cambiar su tecnología Si no se hace una eficiente administración de los requerimientos puede convertirse el proyecto en un bucle sin final hasta que finalmente este es cancelado. En este orden de ideas, es considerada buena práctica documentar los requerimientos del cliente, incluso, es recomendado que éste los firme, de modo de tener una herramienta para validar que el producto que se está entregando cumple con todo lo solicitado inicialmente y todos los nuevos requerimientos que surjan sean manejados en futuras versiones.

Las situaciones inesperadas pueden marcar la diferencia entre un proyecto exitoso y uno fallido. Es por eso que al momento de iniciar el proyecto de software se deben de administrar los riesgos, de modo que esté bien definido que sucederá en caso de que se presente una situación adversa previamente contemplada o no. Los dos riesgos más comunes en nuestro país son el riesgo de perder datos a causa de fallas eléctricas y la rotación del personal. Políticas eficientes de copias de seguridad (backups) reducen en gran medida el riesgo de pérdida permanente de datos; por otra parte, la documentación eficiente de los procesos ayuda a que el nuevo desarrollador continúe el trabajo de la persona saliente.

Lamentablemente, tanto en nuestro país como en el resto del mundo, existe un gran porcentaje empresas y profesionales independientes del desarrollo de software, que optan por no acuñar ningún modelo, y siguen el proyecto de manera empírica, dando pie a las estadísticas expuestas al inicio de este artículo. La disciplina de Ingeniería de Software establece que la fase de codificación representa tan solo un 20% del proceso completo de creación del producto de software, si nos limitamos solo a esta parte sin las demás actividades de soporte estamos dando pie al caos.

Algunas corrientes dentro del ámbito del desarrollo de software, entienden que los procedimientos propuestos por la ingeniería de software son poco flexibles y que por ende hacen que el proyecto sea más lento. Varias ramificaciones de las técnicas tradicionales de ingeniería de software se han creado en los últimos años con fines de crear modelos más flexibles, livianos y eficientes.

Ken Beck, creador de la metodología conocida como Programación Extrema, convocó en febrero del 2001 a todos los representantes de modelos de desarrollo de software alternativos, a fin de defender los intereses comunes. A partir de entonces se conocen estas disciplinas alternas como *modelos ágiles*, y se publicó a raíz de este encuentro el documento denominado Manifiesto Ágil, el cual propone valorar más a los individuos y su interacción que a los procesos y las herramientas

Algo en común entre las metodologías ágiles es la forma en como responden al cambio de los requerimientos, pues todas proponen realizar lanzamientos pequeños y frecuentes, a modo de evitar que el software se convierta en obsoleto en el lapso entre un lanzamiento y otro.

Una metodología ágil que ha tenido auge en los últimos años ha sido la metodología SCRUM, en la cual se reduce el tiempo empleado en crear documentos, se integra el cliente al equipo, y se hacen reuniones diarias de no más de diez minutos, donde el objetivo es que cada miembro exponga lo que hizo ayer, lo que hará hoy y si hay algún impedimento en su camino. Este es un modelo autogestionado, no hay supervisores, cada quien sabe lo que tiene que hacer, tiene tiempo para hacer su trabajo, y cuenta con el apoyo necesario para sobrepasar las trabas que puedan existir en el camino. Grandes empresas han tenido éxito con este modelo poco tradicional, tal es el caso de Google, Yahoo, Microsoft, Electronic Arts, Nokia y Philips, entre otras.

Partiendo de los planteamientos antes expuestos concluimos que las entidades que desarrollan productos de software deben de adoptar algún modelo de desarrollo del cual haya sido probada su eficiencia. Importante es conocer al equipo de trabajo con que se cuenta, para fines de decidir cual modelo utilizar, ya que con personas ágiles, motivadas y dinámicas (como el caso de Google) una metodología ágil pudiera ser sumamente efectiva; no obstante, si el equipo no reúne las características antes mencionadas, las técnicas tradicionales ayudarán a dar un seguimiento más detallado del proyecto y se podrá tener un mayor control sobre éste.

** Standish Group es una empresa estadounidense formada en 1985 con el objetivo de recolectar información sobre fallos en el campo de la tecnología de la información.*