informática

MÉTRICAS PARA EL CONTROL DE CONFIGURACIÓN DE SOFTWARE: DEFINICIÓN Y ALGUNAS VALORACIONES

Resumen / *Abstract*

Los métodos de producción de software están evolucionando desde formas artesanales a la producción industrial en gran escala. La industria de software cubana no está ajena a esos cambios. Para incidir positivamente en el desarrollo de la industria y lograr establecer en ella parámetros de excelencia es imprescindible implantar modelos de procesos tomando en consideración las mejores prácticas internacionales y adaptándolas creativamente a las condiciones concretas de Cuba. Para esto, entre otros aspectos, es necesario medir. El área de las mediciones de software, a pesar de ser una de las áreas en la ingeniería de software donde se ha investigado desde hace más de 30 años, todavía no ha sido bien comprendida, ni ampliamente aplicada en la industria del software, por tanto, la implementación de las mediciones en una organización requiere un cambio tecnológico, educacional y cultural importante. Las autoras del presente artículo proponen un conjunto de métricas para aplicar en el proceso de control de configuración. Estas métricas pueden ser obtenidas a partir del modelo de procesos de control configuración de software definido en trabajos anteriores por el grupo de Ingeniería de Software del Centro de Estudios de Ingeniería y Sistemas (CEIS). También se enuncian algunas reglas básicas a tener en cuenta cuando se aplican estas métricas en una empresa.

The software production is evolving from manufacture to industrializing. Cuban software industry is not unaware to those changes. To impact positively in the development of the industry and to be able to settle down in her excellence parameters it is indispensable to implant models of processes applying the best international practices and adapting them creatively to cuban conditions. In this relation, it is necessary to measure. Software metrics and standardization, in spite of being one of the areas in the software engineering where it has been investigated for more than 30 years, it has not still been well understood, neither broadly applied in the industry of the software. Therefore, the implementation of the metrics and standards in different organizations and enterprises constitutes a transcendental technological, educational and cultural change. Authors propose a metric set that it is convenient to apply in the control process of configuration. These software metrics can be obtained starting from the model of control processes of software configuration defined in previous works by CEIS Software Engineering scientific group. Therefore, some basic rules to keep in mind when these metric ones are applied in a company are enunciated in the paper.

Palabras clave / *Key words*

Gestión, configuración, métricas, defectos, cambios

Management, metrics, defects, change

Ailyn Febles Estrada, *Licenciada en Ciencias de la Computación, Máster en Informática Aplicada, Centro de Estudios de Ingeniería y Sistemas (CEIS), Instituto Superior Politécnico José Antonio Echeverría, cujae, Ciudad de La Habana, Cuba*
e-mail:afebles@ceis.cujae.edu.cu
Isabel Pérez Esteves, *Ingeniera Informática, CEIS, Instituto Superior Politécnico José Antonio Echeverría cujae, Ciudad de La Habana, Cuba*

Recibido:octubre del 2002
Aprobado:diciembre del 2002

INTRODUCCIÓN

El software se ha convertido en un tema crítico en la sociedad moderna mundial. Todos parecen necesitar mejores software en menos tiempo y a menor costo. Los métodos intuitivos de desarrollo de software que se usan actualmente son, básicamente, aquellos que los propios individuos

artesanalmente siguen, y solo servirán mientras la sociedad pueda tolerar la falta de predicción que ellos acarrean.

Siguiendo la idea de que cuando un modelo de proceso estándar ha sido especificado, el desarrollo de software puede ser un proceso definido, repetible, en lugar de una actividad *ad hoc* reinventada para cada nuevo proyecto;[1] en trabajos anteriores, se propuso el **proceso de control de configuración**, el cual incluye el **proceso de control de cambios** y el **proceso de control de versiones**, y para la industria de software cubana

Con la aplicación de estos procesos es posible, además de disciplinar a los involucrados, almacenar y disponer de los datos históricos necesarios para lograr un trabajo más predecible y eficiente. No obstante, hasta ese punto la aplicación del proceso solo garantiza la recolección de los datos. Para que los proyectos que sigan el modelo de proceso definido puedan ser planificados, supervisados (monitoreados), y controlados, será necesario definir y usar métricas, las cuales se pueden definir en términos de los componentes del propio modelo de procesos. Este artículo presenta un conjunto de métricas para ser aplicadas en las distintas áreas de procesos en la gestión de configuración, así como un grupo de sugerencias para la aplicación de estas métricas.

CALIDAD DE SOFTWARE

La administración de la calidad total es un término que se originó en 1985, que describe un estilo de administración japonés para mejorar la calidad. La TQM (qué es TQM ?), toma varios significados en dependencia de quién lo interprete y de cómo lo apliquen. En general, representa un estilo de administración dirigido a lograr éxitos a largo plazo enlazando la calidad con la satisfacción del cliente. Según TQM, las métricas y el análisis de estas, son los elementos fundamentales para lograr un mejoramiento continuo de la calidad.

Varios ambientes de trabajo han sido propuestos para mejorar la calidad que pueden ser usados para substanciar la filosofía de TQM. Algunos ejemplos son:[2]

- *Plan-Do-Check-Act*.
- *Quality Improvement Paradigm (QIP) / Experience Factory Organization*.
- *Capability Maturity Model (CMM) del Software Engineering Institute (SEI)*.
- *Lean Enterprise Management*.

MODELO DE MADUREZ DE LAS CAPACIDADES DEL SEI (CMM)

"Los procesos son como los hábitos: difíciles de establecer e incluso mucho más difíciles de romper.[3] Los procesos para desarrollar software a gran escala, pueden ser muy grandes y complejos. Pueden ser difíciles de definir e incluso mucho más difíciles de introducir. Por esta razón fue que el Software Engineer Institute (SEI) de la universidad Carnegie-Mellon desarrolló un ambiente de trabajo (*framework*) de madurez de procesos de software.[4] Este *framework* es una forma ordenada para las organizaciones de determinar las capacidades de sus procesos actuales y establecer prioridades en su mejora. Se hace a través del establecimiento de 5 niveles de prioridad progresivos logrando procesos de capacidades más maduros.[5] Por cada nivel se han definido los principios elementales o KPAs (*Key Process Areas*) que proveen las metas y ejemplifican las prácticas a llevar a cabo (figura 1).

CMM ha sido revisado y refinado por varios especialistas y representa el mejor juicio actual y el método más efectivo para conseguir los objetivos de cada nivel de madurez.

PROCESO PERSONAL DE SOFTWARE

El proceso personal de software (PSP, *Personal Software Process*) es un proceso de automejoramiento diseñado para ayudar a controlar, administrar y mejorar la forma en que se trabaja individualmente. Está estructurado por formularios, guías y procedimientos para desarrollar software. Si es usado apropiadamente, brinda los datos históricos necesarios para trabajar mejor y lograr que los elementos rutinarios del trabajo sean más predecibles y eficientes.[3]

Usando PSP, se pueden construir programas de más de 10 000 líneas de código (LOC), sin embargo, hay dos problemas típicos en los grandes programas. Primero, mientras se crece en tamaño, también lo hace el tiempo y el esfuerzo requerido. Esto puede ser una situación particular si solo existe un ingeniero en el proyecto. Segundo, la mayoría de los ingenieros tienen problemas en la visualización de todas las facetas importantes de un programa, incluso cuando su tamaño es moderado. Existen muchos detalles e interrelaciones que deben tenerse en cuenta, muchas dependencias lógicas, interacciones en el tiempo o condiciones excepcionales. Una de las formas más poderosas de resolver estos problemas es el proceso de software del equipo (*Team Software Process*, TSP).

PROCESO DE SOFTWARE DEL EQUIPO

Una definición acertada de un equipo es la dada por Dyer,[6,7] donde un equipo consiste en al menos dos personas que trabajan para lograr una meta/ objetivo/ misión común, donde cada persona tiene asignado un rol específico o funciones específicas que desarrollar, y donde el completamiento de la misión requiere alguna forma de dependencia entre los miembros del equipo.

El proceso del equipo de software (TSP, *Team Software Process*) es un proceso que al igual que el PSP, está basado en el modelo CMM, TSP está diseñado para ayudar a controlar, administrar y mejorar la forma en que trabaja un equipo de software. Al igual que PSP, está estructurado por formularios, guías y procedimientos para desarrollar software.[7]

MEDICIONES

Para lograr una buena calidad del producto es necesario identificar las mediciones y los criterios que serán utilizados para identificar el nivel deseado de la calidad y determinar si se está alcanzando. Las mediciones describen el método para capturar los datos que serán utilizados para evaluar la calidad, mientras

que los criterios definen el nivel o el punto en el cual el producto logra la calidad aceptable (o inaceptable).[8]

Es una afirmación indiscutible que las mediciones son cruciales en el progreso de todas las ciencias. El progreso científico se logra a través de observaciones y generalizaciones basadas en datos y mediciones y la derivación de teorías como resultado de estas.[2] Sin la verificación a través de los datos y las mediciones, las teorías y las proposiciones permanecerían en un nivel abstracto.

Los principios fundamentales que deben seguir las métricas son:[8]

• Deben ser simples, objetivas, fáciles de coleccionar, fáciles de interpretar y difíciles de malinterpretar.

• Su recolección debe ser automática y no intrusiva, o sea, no interferir en las actividades de los desarrolladores.

• Deben contribuir a la evaluación de la calidad temprana en e ciclo de vida, cuando los esfuerzos por mejorar la calidad de software son efectivos.

• Los valores absolutos y las tendencias de las métricas, deber ser usados activamente por el personal administrativo y e personal ingenieril, para comunicar progreso y calidad en ur formato coherente.

• La selección de un mínimo o más extensivo conjunto de métricas, dependerá de las características y contexto del proyecto Si es muy grande o si tiene restricciones de seguridad o de confiabilidad de los requerimientos; y si el equipo de desarrollo y de valoración (evaluación) es conocedor de las métricas, lo cua hará muy útil coleccionar y analizar las métricas técnicas.

	Características	KPAs	Resultado
5 Optimizado	Bases cualitativas para una inversión capital continua en la automatización y mejoramiento de la retroalimentación de los procesos	-Prevención de defectos -Administración de la tecnología de cambio -Administración del proceso de cambio	***Productividad y calidad***
4 Administrado	(Cuantitativo) -Procesos medidos -Control estadístico razonable sobre la calidad del producto	-Medición y análisis de los procesos -Administración de la calidad	
3 Definido	(Cuantitativo) -Costo y planificación confiables -Procesos definidos e institucionalizados -Rendimiento de la calidad mejorado pero impredecible	-Definición y mejoramiento de los procesos organizacionales -Programas de entrenamiento -Administración integrada de software -Coordinación intergrupal -Inspecciones entre compañeros -Ingeniería del producto de software	
2 Repetible	(Intuitivo) -Procesos dependientes de los individuos -Costo y calidad altamente variables- Métodos y procedimientos informales y *ad hocs*	-Administración de los requerimientos -Planificación y vigilancia de los proyectos de software -Administración de la subcontratación de software -Aseguramiento de la calidad del software -Administración de la configuración de software	
1 Lineal	-*Ad hoc* -Caótico -Costos, planificación y rendimiento de la calidad impredecibles		

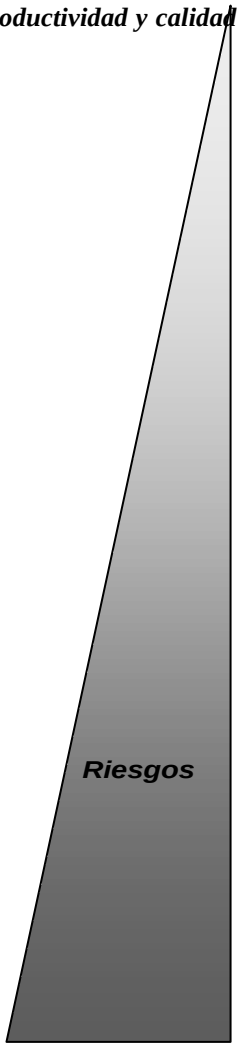

Fig. 1 *Representación esquemática de CMM.*

ANTECEDENTES

En trabajos desarrollados con anterioridad[9,10] se propuso un modelo de procesos de software para el control de configuración con el objetivo de ayudar a dar solución a algunos de los problemas existentes en la industria cubana de software. Con vistas a automatizar los procesos definidos se desarrolló igualmente la herramienta CASE, para el control, administración y mejoramiento de los métodos de trabajo, específicamente en lo que se refiere al desarrollo de software. El CASE propiamente usado provee una forma cómoda de almacenar y disponer de los datos históricos necesarios para lograr un trabajo predecible y eficiente. No obstante, hasta ese punto el CASE es simplemente una colección de datos. Por esta razón, surge la necesidad de ampliar la herramienta CASE a través de la definición e implementación de métricas que permitan que los proyectos que sigan el modelo de procesos definidos puedan ser planificados, monitoreados y controlados.

ESTÁNDARES PARA MEDIR EL CONTROL DE CONFIGURACIÓN

El SEI a partir de una análisis realizado, publicó un conjunto de 683 métricas para las áreas de proceso de los niveles 2,3,4,5 de CMM.[11] Dentro de las más importantes para el control de configuración se encuentran las siguientes:

1. Esfuerzo real realizado para las tareas de control de configuración.
2. Tareas terminadas en el proceso de control de configuración.
3. Resumen y estado de las solicitudes de cambio.
4. Estimado del trabajo actual completado en las actividades de control de configuración.
5. Esfuerzo estimado en las actividades de control de configuración.
6. Número de solicitudes de cambio liberadas por unidad de tiempo.
7. Resumen de las solicitudes de cambio.
8. Resumen de los defectos, incluidos los resueltos.
9. Tareas terminas en el control de configuración.

Teniendo en cuenta estas métricas fundamentales, las especificaciones de PSP, TSP y PSM, las características de la empresa cubana y analizando las características de la información que se utiliza en el proceso definido y se automatiza y almacena en la versión anterior del CASE, fueron definidas las siguientes métricas, las cuales se pueden obtener manualmente o mediante los nuevos módulos de la herramienta CASE.

MÉTRICAS DEFINIDAS

1. Del proceso

Seguimiento del avance del proceso

Un plan detallado de control de configuración ayuda a seguir y administrar el trabajo realizado.

Unas de las principales mediciones para el seguimiento del proyecto son el valor planificado (PV- *Planed Value*) y el valor devengado (EV- *Earned Value*). Estas mediciones proveen una forma de determinar la contribución de cada tarea al cronograma del proyecto. Durante la planificación se le llama valor planificado (PV- *Planed Value*) de una tarea o de una semana a lo que se planifica que esa tarea o semana contribuya al progreso del proyecto, y el valor de una tarea (o de una semana) será **devengado** solamente cuando esta tarea (o semana) se termine. No hay crédito parcial por tareas a medio concluir.[3]

Es recomendable obtener gráficos del valor devengado acumulado planificado y real semanalmente (figura 2).

Cuando se analicen los valores devengados planificados y reales se verá cuán adelantado o atrasado se está, aunque no se haya seguido el orden planificado.

Calidad de la planificación

La mejor forma de planificar con exactitud es medir y darle seguimiento a los procesos de desarrollo y trabajar para mejorarlos. Esto se puede lograr siguiendo un proceso de planificación consistente, analizando los datos históricos y planificando entonces al detalle.

Error de la planificación

Un método para medir la calidad de la planificación es durante el desarrollo de un trabajo o cuando se termine, tomar las mediciones reales de tiempo, costo o tamaño, y calcular el error entre las horas reales y las planificadas, el costo real y el planificado; o entre las LOC reales y estimadas, respectivamente.[3]

El error de la planificación se puede calcular como:

Error % = 100 * (Real - Estimado) / Estimado

Basándose en que:

Error = Real - Estimado

Estimado 100%

Error Error %

Será mejor la calidad de la planificación cuando el error esté oscilando en valores cercanos a cero.

Razón de costo de la planificación

Otra forma de medir la calidad de la planificación es a través del análisis de la **razón de costo de planificación (RCP)**.[3,12] La RCP es la razón del costo planificado, dividido entre el costo real hasta la fecha en que se calcule. Como se observa esta métrica es similar a la anterior, la única diferencia radica en que esta es una razón y la otra un porcentaje. La RCP, al igual que el error, se puede analizar por tiempo, dinero o LOC.

En caso de que dicha razón se iguale a 1, significa que se gasta en planificar lo mismo que se gasta en desarrollar, o sea, el costo no varía.

En caso de que la RCP sea menor que 1, se evidencia pobre desempeño del proyecto, se gasta más de lo que se gana. Si el análisis es de tiempo, puede ser que se esté gastando mucho tiempo en planificar los proyectos.

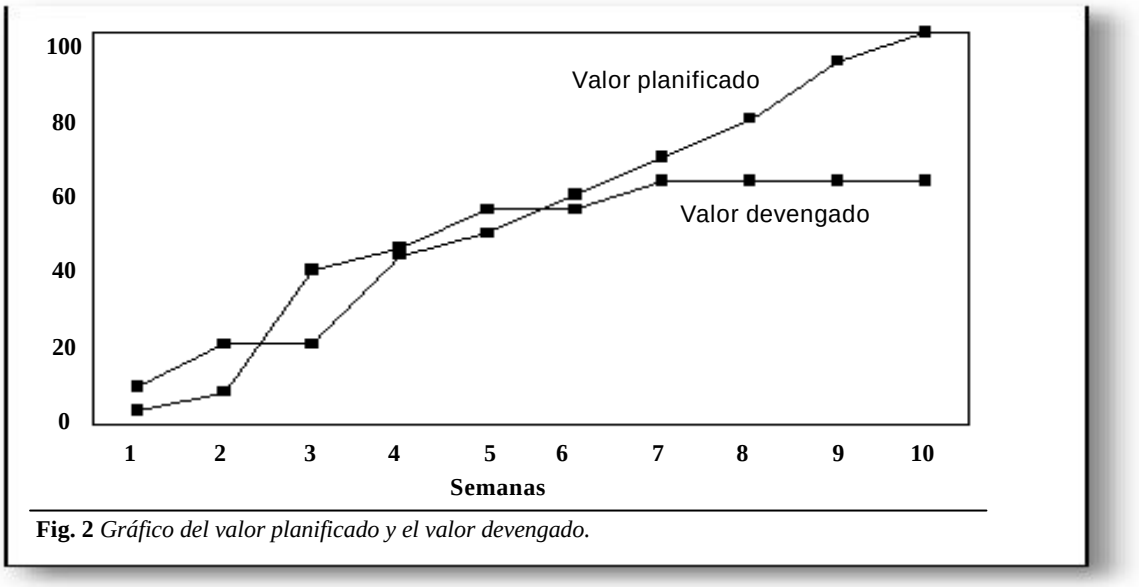

Fig. 2 *Gráfico del valor planificado y el valor devengado.*

Idealmente el RCP debe ser ligeramente mayor que 1, lo cual demostraría un buen desempeño del proyecto, o sea, se gasta menos de lo que se gana.

2. Pedidos de cambio

Estado de los pedidos de cambio

Con este resumen se puede llegar a la conclusión clara de en qué estado están estancados los pedidos de cambio y por tanto darle una advertencia al jefe del proyecto de dónde tiene que insistir e incluso si es necesaria una reunión de la Junta de Control de Cambios (figura 3).

En el proceso están definidos 5 estados de las peticiones de cambio:

• Aprobada: Peticiones de cambio que ya han sido aprobadas por la junta de control de cambios pero no se han comenzado a desarrollar por un especialista.

• Cerrada: Petición de cambio que ya fue resuelta.

• En cola: Petición de cambio por analizar por la junta de control de cambios.

• En desarrollo: Petición de cambio aprobada por la junta que está siendo resuelta por un desarrollador.

• Posible rechazada: Solicitudes de cambio que la junta tiene pendiente por tener incompleto los datos.

Esfuerzo del personal

Para que el jefe del equipo tenga más elementos a la hora de asignar, reasignar carga de trabajo y analizar el esfuerzo de personal bajo su mando se definieron las siguientes mediciones del esfuerzo del personal:

• Carga laboral real de cada uno de los trabajadores de proyecto.

• Horas planificadas y horas laboradas en un gráfico de barras (personal y del equipo completo).

• Horas mensuales de personal, planificadas y reales.

Con un análisis de los gráficos, como los que se muestran en la figura 4, se puede lograr un análisis más profundo de la asignación de esfuerzo.

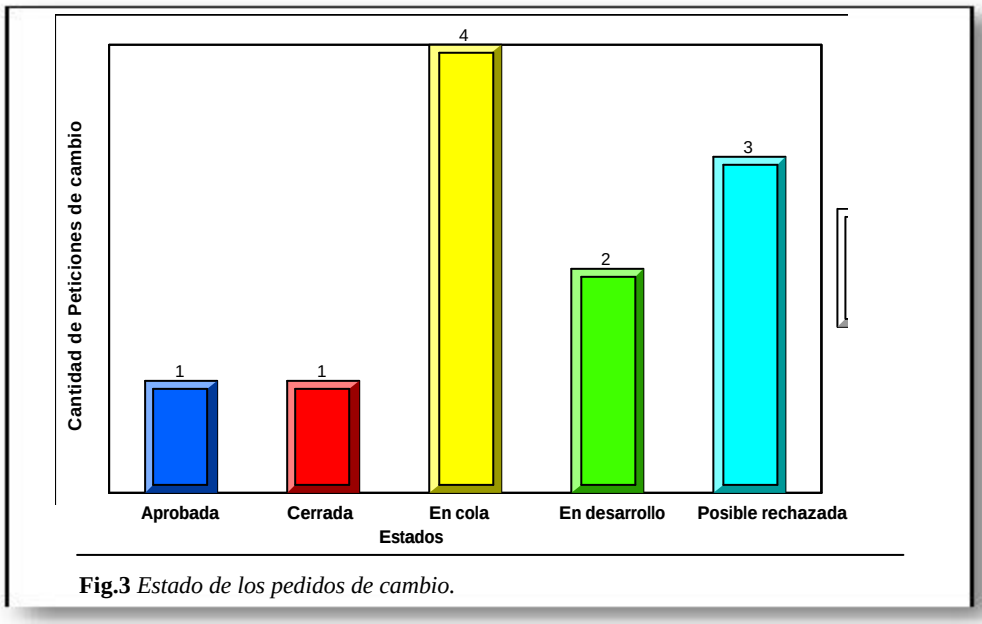

Fig.3 *Estado de los pedidos de cambio.*

Los gráficos de esfuerzo del personal están basados en propuestas de métricas del PSM.[13]

3. De los defectos

Durante el desarrollo del proceso de control de configuración, muchos de los cambios que son solicitados son producto de defectos en el software o el prototipo entregado al usuario final. Por esta razón es importante controlar los defectos que aparecen de esta manera en el producto.

Resumen de defectos

Una forma de analizar el estado de los defectos es a través del **resumen de defectos**, donde se analizan la densidad de defectos por etapas, y los escapes netos. Se considera escapes netos a todos los defectos que se insertaron antes o durante una etapa determinada, pero que no fueron encontrados en esa etapa, sino en otra etapa posterior.

Un ejemplo de un resumen de defectos, se muestra en la tabla 1.

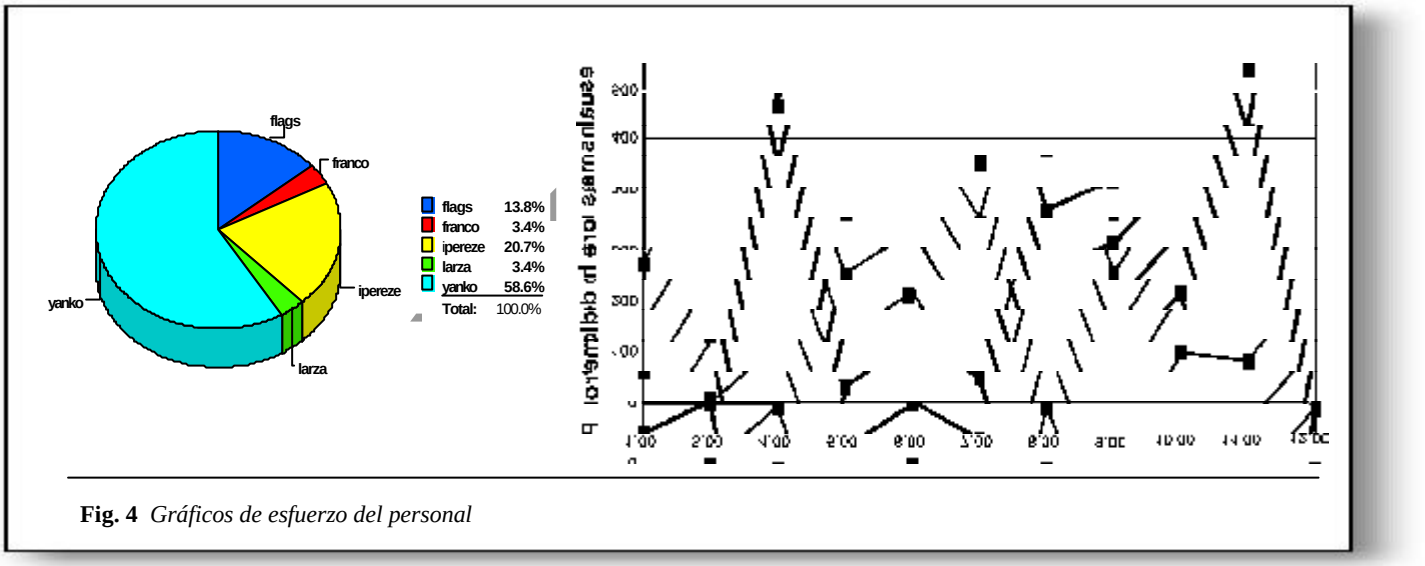

Fig. 4 *Gráficos de esfuerzo del personal*

TABLA 1
Resumen de defectos por etapas

Etapa	Insertados	Eliminados	Acumulativo Insertados	Acumulativo eliminados	Escapes netos	Rendimiento
Planificación	1	0	1	0	1	
Diseño	5	0	6	0	6	
Inspección Diseño	0	3	6	3	3	50 %
Codificación	15	1	21	4	17	
Inspección Código	0	8	21	12	9	47,1%
Prueba	0	6	21	18	3	
Control de cambios	0	3	21	21	0	
Total	21	21				57,1%

Nótese que con el resumen de defectos, no se observa el avance del rendimiento, solamente su valor final. Sin embargo, el método de calcularlo es mucho más fácil, el rendimiento por cada etapa se podrá calcular por:

Rendimiento = Defectos eliminados · 100 /(Defectos eliminados + Escapes netos)

En rendimiento total del proceso se calcula como:

Rendimiento Total Proceso = Defectos Eliminados antes de prueba ·100 / (Defectos Eliminados ante prueba + Escapes antes de prueba)

Nótese que estos valores son equivalentes a:

Rendimiento Total Proceso = Acumul. Defectos Eliminados inspec. Código · 100 /(Acumulado Defectos Eliminados inspec. Código + Escapes inspec. Código)

Debe quedar claro que un valor de rendimiento alto es mucho mejor que un valor bajo, el cual es un resultado pobre. El objetivo o meta es un rendimiento de 100 %, o sea, que se traten de eliminar los defectos tan pronto como sea posible.

Para analizar el avance del rendimiento de un equipo de proyectos, la métrica más eficiente es graficar el rendimiento total por cada proyecto, un ejemplo se muestra en la figura 5.

Una buena tendencia es la señalada por la línea más gruesa, o sea, en la medida que el equipo de proyecto gana en experiencia el valor del rendimiento debe ir aumentando.

Defectos por hora

Los defectos por hora indican la efectividad del tiempo dedicado a las inspecciones. En la medida que el rendimiento incrementa, una disminución de los defectos por unidad de tiempo es natural.

Los defectos eliminados por unidad de tiempo se pueden calcular como:

$$\text{Defectos encontrados por horas} = 60 \cdot \frac{\text{Defectos eliminados en la etapa}}{\text{Minutos dedicados a esa etapa}}$$

Nota: Cualquier otra conversión de unidad de tiempo es válida.

Eficiencia en la eliminación de defectos

La eficiencia en la eliminación de defectos provee una métrica útil de la efectividad relativa de varios métodos de eliminación de defectos. Da la razón de defectos eliminados por unidad de tiempo de 2 etapas cualesquiera que sean. Típicamente se hace comparando con la etapa de prueba.

Frecuencia de defectos

Es importante darle un seguimiento a los defectos que se detectan la forma más cómoda y visual de lograr este seguimiento es a través de los gráficos de frecuencia de defectos. Un ejemplo pudiera ser el que se muestra en la figura 6. En este gráfico se puede llegar a la conclusión clara de los defectos que más se suelen insertar a producto, los que más fácil se eliminan etcétera.

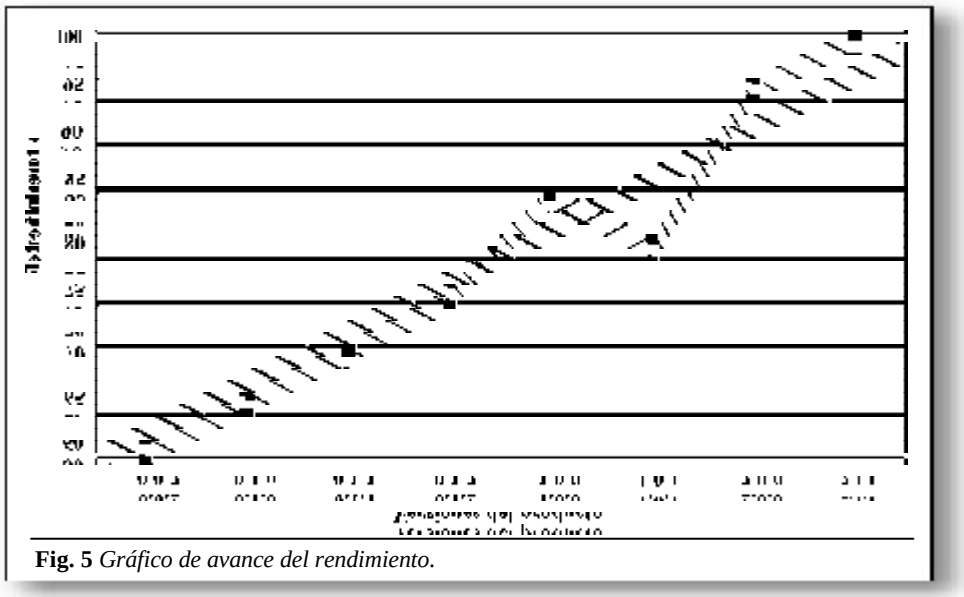

Fig. 5 *Gráfico de avance del rendimiento.*

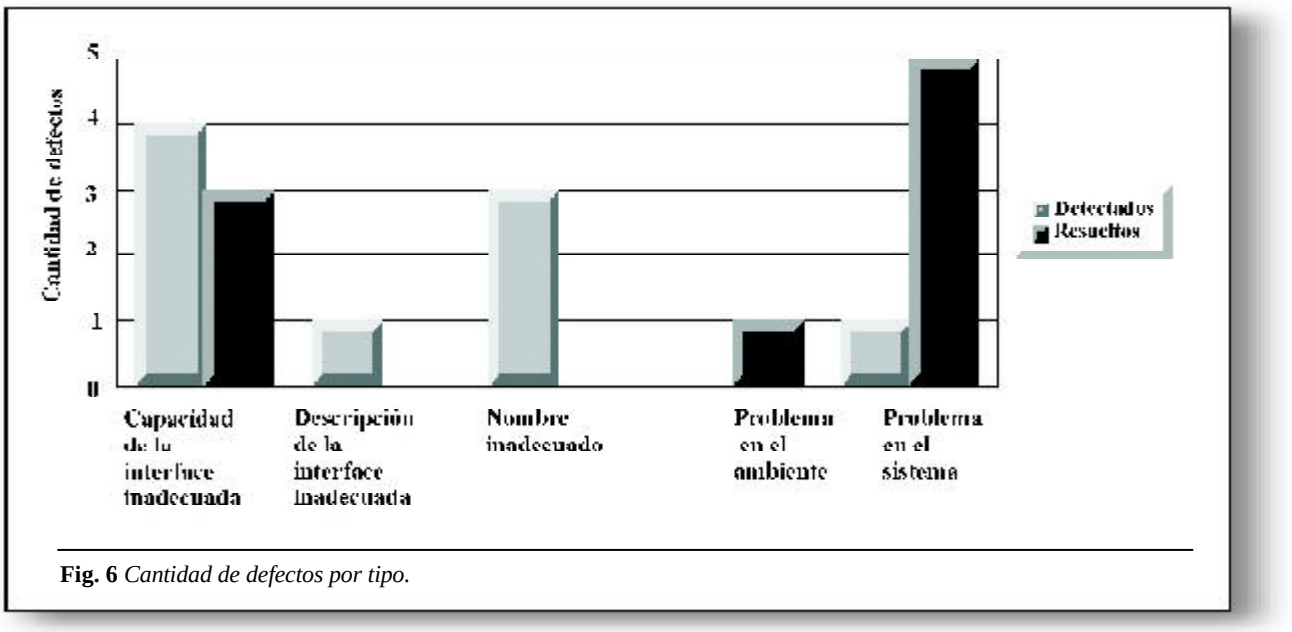

Fig. 6 *Cantidad de defectos por tipo.*

Algo similar se puede analizar de un gráfico que en lugar de los tipos de defectos muestre el estado en que están, dígase: detectado, resuelto, en desarrollo, etcétera.

ALGUNAS REFLEXIONES SOBRE LA APLICACIÓN DE LAS MÉTRICAS

Ninguna discusión de la selección y el diseño de métricas de software estaría completa sin abordar el tema de cómo las métricas afectan a las personas y cómo las personas afectan a las métricas. Aunque sea indiscutible, la utilidad de las métricas para la organización, siempre dependerán de las actitudes de las personas involucradas.

Suele haber preocupación de que las mediciones señalarán problemas en un proyecto o en una organización que no eran visibles antes de que el proceso de medición fuera implementado. Estas preocupaciones son reales, y sobreponerse a ellas, requiere un entendimiento de las mediciones, así como saber cómo usar los resultados de las mediciones apropiadamente en todos los niveles de la organización.

La mejor forma de evitar el problema del factor humano en el trabajo con las métricas es seguir algunas reglas básicas tales como las que se enuncian a continuación:

- **No mida a los individuos:** El estado del arte en las métricas de software no llega hasta ese punto. El ejemplo clásico de este error es medir la productividad de los individuos. Si se mide la productividad, en líneas de código (LOC) por horas, puede ocurrir que las personas se concentren en su propio trabajo en detrimento del equipo y del proyecto. Hay que enfocarse en los procesos y en los productos, no en las personas.
- **Nunca usar las métricas como un "garrote":** La primera vez que se usen las métricas en contra de los individuos o los equipos será la última vez que se obtendrán datos válidos.
- **No ignorar los datos:** Una forma segura de matar un programa de métricas es ignorar los datos cuando se toman decisiones.
- **Nunca usar una sola métrica:** Los software son complejos y multifacéticos. Un programa de métricas debe reflejar esa complejidad. Debe mantenerse un balance entre los atributos del costo, la calidad y los cronogramas de forma que se satisfagan todas las necesidades de los usuarios. Enfocarse en una única métrica puede causar que el atributo que es medido mejore a costa de otros atributos.
- **Proveer retroalimentación:** Proporcionando una retroalimentación regular al equipo sobre los datos que ellos ayudan a coleccionar tiene varios beneficios, por ejemplo; ayuda a mantener el foco en la necesidad de coleccionar los datos. Si los miembros del equipo se mantienen informados sobre los detalles específicos de cómo los datos son usados, ellos tendrán menos posibilidades de empezar a sospechar sobre su uso, y ayuda a educar a los miembros del equipo en la responsabilidad de coleccionar los datos. Los beneficios pueden ser datos más consistentes, exactos y oportunos.
- **Lograr "pertenencia":** Para lograr un sentido de pertenencia tanto en las metas como en las métricas en un programa de medición, se tiene que lograr la participación en la definición de las métricas.
- **Respetar la privacidad de los datos y de las métricas**, clasificando cada elemento de dato que se coleccione en alguno de los tres niveles que propone Wiegers.

1. Individual
2. Equipo de proyecto
3. Organización

CONCLUSIONES

- En el Centro de Referencia de Ingeniería de Software (CRIS) perteneciente al Centro de Estudios de Ingeniería de Software (CEIS) fue realizado el "CASE para la planificación y control de configuración de software",[10] donde se propuso un modelo de procesos de software con el objetivo de ayudar a dar solución a algunos de los problemas existentes en la industria cubana de software.
- Surgió la necesidad de ampliar la herramienta CASE a través de la definición e implementación de métricas que permitan que los proyectos que sigan el modelo de procesos definidos puedan ser planificados, monitoreados y controlados.
- Fueron definidas un total de 12 métricas que abarcan los aspectos fundamentales del control de la calidad de los procesos de desarrollo de software, del seguimiento y monitoreo de los procesos, y en alguna medida de la planificación de los proyectos, desde el punto de vista del control de cambios.

REFERENCIAS

1. **NUSENOFF, RONALD E. AND DENNIS C. BUNDE:** *A Guide Book and Spedsheet Tool for a Corporate Metrics Program*, Holanda, 1993.
2. **KAN H., STEPHEN:** *Metrics and Models in Software Quality Engineering*, Addison-Wesley, 2000.
3. **HUMPHREY, WATTS S.:** *A Discipline for Software Engineering*, Addison-Wesley, 1995.
4. **————.:** *Managing the Software Process*, Addison-Wesley, 1989.
5. **PAULK, MARK C.; BILL CURTIS AND MARY BETH CHRISSIS:** *Capability Maturity Model, The: Guidelines for Improving the Software Process*, Addison-Wesley, 1995.
6. **DYER, JEAN L.:** "Team Research and Team Training: A-Stateo fthe-Art Review", *Human Factors Reviews*, The Human Factors Society, Inc, 1984.
7. **HUMPHREY, WATTS S.:** *Introduction to the Team Software Process (sm)*". Addison-Wesley, 2000.
8. *Rational Unified Process*, Rational Software Corporation. Version 2001A.04.00, Copyright 1987-2001.
9. **FEBLES, AILYN:** "Case corporativo para el proceso de control de cambios", Tesis presentada en opción al título de Máster en Informática Aplicada, Ciudad de La Habana, 2001.
10. **HERNÁNDEZ, YANKO E. IDAEL BANDERAS:** "Case para la planificación y control de configuración de software", Trabajo de Diploma para optar por el título de Ingeniero Informático. Instituto Superior Politécnico "José Antonio Echeverría", Ciudad de la Habana, 2001.
11. SEI Level 2, 3, 4, & 51, *Metrics* (683), 2001.
12. **MUNRO, BILL:** *Measuring IT Project Performance*. http://www.tsepm.com/tsepm/spring01/article3.html (31/01/2002).
13. **McGARRY, JOHN** *et al*.**:** *Practical Software Measurement: A Foundation for Objective Project Management Version 3.1a*, 1998.