Rev. Fac. Ing. Univ. Antioquia N.° 58 pp. 199-207. Marzo, 2011

Colección automática de métricas hacia un repositorio de mediciones

Automatic metric collection to a repository of measurements

José Alejandro Lugo García[1], Ana María García Pérez[2]*

[1] Dirección Técnica de la Producción. Universidad de las Ciencias Informáticas (UCI), Carretera de San Antonio de los Baños. Km 2 ½. Torrens. Ciudad de la Habana. Cuba.

[2] Centro de Desarrollo de Software Villa Clara. Carretera a Camajuaní, Km 5.5. CP 54830. Santa Clara, Villa Clara, Cuba.

(Recibido el 05 de mayo de 2010. Aceptado el 19 de agosto de 2010)

Resumen

En este trabajo se propone un proceso de colección automática de medidas del software producido. Esto permitiría obtener indicadores de la productividad y del esfuerzo asociados, a través de un entorno construido completamente sobre software libre.

----- *Palabras clave:* Colección automática de medidas software producido

Abstract:

This paper proposes an automatic collection process from measurements of produced software. It would be possible to obtain indicators of productivity and associated effort, through an environment built entirely on free software.

----- *Keywords:* Automatic measure collection produced software

* Autor de correspondencia: teléfono: + 53 + 5 + 237 73 46, correo electrónico: jalugo@uci.cu (J. A. Lugo García)

Introducción

Los indicadores derivados del proceso de recolección de medidas y análisis de métricas son extensamente utilizados para la evaluación del estado de las organizaciones productoras de software. Si bien los términos *medida*, *medición* y *métricas* se emplean indistintamente con frecuencia, es fundamental resaltar las diferencias entre ellos. Dentro del contexto de la ingeniería del software, una *medida* proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto. La *medición* es el acto de determinar una medida [1]. Una *métrica* es una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado [2]. Dicho en otras palabras, la medida expresa una característica individual, la medición permite capturar dicha característica y la métrica permite relacionar y comparar mediciones.

Cada empresa es responsable de definir las métricas que va a implementar teniendo en cuenta sus objetivos organizacionales y necesidades de información, por ello las métricas deben estar alineadas con el calendario, costo y niveles de calidad propuestos. A esto se añade que cada Organización debe establecer un proceso para la colección de los datos y definir este proceso lo más natural posible para sus trabajadores.

Hoy día, la mayoría de las organizaciones estima la productividad de sus trabajadores basándose en las preguntas que el analista hace al programador en el módulo donde trabaja, y mediante híbridos de métodos de estimación basados en COCOMO II y puntos de función o casos de uso. Por otra parte, no abundan por lo general, herramientas de software libre integradas al desarrollo que almacenen y obtengan medidas para el apoyo a las estimaciones y por ende el grado de subjetividad con que esta tarea se realiza es alto.

El objetivo de este artículo es exponer un proceso de colección automática de medidas del software producido, con la posibilidad de obtener indicadores de la productividad y del esfuerzo asociados, a través de un entorno construido completamente sobre software libre.

La organización del trabajo es la siguiente. Primero, son presentados los conceptos relacionados con indicadores y línea base. Luego, se discuten y son comparadas las técnicas y enfoques actuales sobre el establecimiento, colección de medidas y qué métricas son más convenientes para dimensionar el tamaño del software que se construye. Se muestran herramientas útiles que pueden ser explotadas para la obtención de medidas en proyectos de software. A continuación se describen las nuevas herramientas que se construyeron, el proceso de colección automática de medidas, y la aplicación de la herramienta a un caso de estudio. Finalmente se exponen las conclusiones.

Modelo teórico

Indicadores y línea base

La figura 1 muestra que un *proceso de medición y análisis* conlleva a la aplicación continua de mediciones basadas en técnicas para el proceso de desarrollo del software y sus productos, a fin de suministrar información relevante a tiempo. Así, la Organización, mediante el empleo de estas técnicas mejorará el proceso y sus productos.

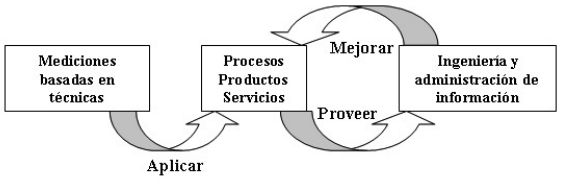

Figura 1 Proceso de medición y análisis

Para dotar de capacidad de supervivencia a un sistema de mejora continua, es necesario disponer de un sistema de información que permita la identificación sistemática de oportunidades de mejora relevantes para los responsables de la Organización. Los elementos de tales

oportunidades pueden encontrarse a partir del análisis de los *indicadores* que puedan ser obtenidos por el propio sistema de información.

Los indicadores, en el caso de organizaciones de software, pueden consistir en métricas o combinaciones de métricas que proporcionen una visión profunda del proceso de desarrollo del software, de los proyectos de software y de los productos en sí.

Un punto de partida para realizar estimaciones es establecer una *línea base* con estos indicadores siguiendo el proceso descrito en la figura 2.

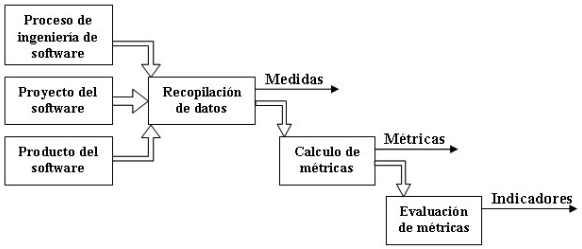

Figura 2 Proceso de recopilación de indicadores del software [1]

Si se aplica el proceso descrito para la recopilación de métricas de software y se aplica en la determinación y almacenamiento de los indicadores, se dispondría de una poderosa información para conocer dos pronósticos: el primero relacionado con la mejora de los procesos de la Organización y el segundo vinculado a la estimación más certera de proyectos futuros.

Los indicadores deberán ser posibles de obtener de cada proyecto que forme parte de la institución, y publicarse al personal que lo solicite, según el nivel de acceso correspondiente, en un repositorio central.

Por tanto, se hace deseable poder contar con un repositorio que contenga datos relativos a medidas, métricas, documentos asociados e indicadores según sea el interés u objetivo de los dueños del proceso, proyectos y productos [3].

Establecimiento y colección de métricas

Si bien la disponibilidad de los datos sobre los indicadores de mejora es un elemento clave, uno de los mayores problemas al que se enfrenta cada Organización es el establecimiento y colección de sus medidas.

En este sentido Pressman [1] reconoce que constituye un reto el hecho de lograr que los ingenieros del software cobren conciencia de la importancia de registrar sus actividades.

El Proceso Personal de Software (PSP) y el Proceso de Software en Equipo (TSP) [4], incluyen reconocidas técnicas para el asentamiento de una base de información útil en la obtención de indicadores, pero aún así requieren de mucho trabajo por parte del ingeniero para lograr llevar a cabo el control necesario de sí mismo y del equipo respectivamente.

Otros modelos incluyen métricas con el objetivo de evaluar distintos atributos de calidad del producto, entre estos se encuentra GQM (Goal Question Metric) [5], en el que, partiendo de los objetivos y las preguntas respecto a qué hace una Organización para mejorar, se logran establecer las métricas correspondientes.

Patrocinado por el Departamento de Defensa y la Armada de los Estados Unidos, Practical Software Measurement (PSM) [6], es considerado como el esquema base a partir del cual se ha elaborado el nuevo estándar ISO/IEC 15939 [7] relativo a la medición del software para el que se proporcionan detalles adicionales respecto a las actividades y tareas relacionadas con el establecimiento de métricas e interpretación de indicadores. Propone una herramienta gratis, el PSM Insight, en la cual se establecen manualmente o por importaciones los datos primarios que servirán para obtener gráficos con indicadores sobre la medición del software.

Sin embargo, el enfoque actual de recolección de las medidas, al realizarse manualmente, tiene sus repercusiones en el producto, pues consume a los ingenieros del software parte del tiempo de desarrollo.

Además implica disparidades en las interpretaciones de las métricas dado que cada individuo las interpreta a su manera. Es importante que la gestión de un producto software y su calidad sea un proceso que no estorbe a quienes lo construyen y siguiendo este principio, cada proceso de recolección, análisis y generación de reportes de indicadores debe tratar de ajustarse a la forma más simple posible.

El trabajo de García [8] propone diseñar, implementar y validar un proceso automático de formación de los indicadores de productividad y costo, a partir de la producción de componentes de la Organización de software. Para lograr esto se propone crear una herramienta que calcule una métrica para tamaño del software, acoplable a un sistema de gestión de configuración.

Métricas para dimensionar el tamaño del software

Entre las métricas para medir el tamaño del software se encuentran los puntos de función [9], los puntos de característica [10] y los puntos de casos de uso [11] que si bien suponen una buena aproximación al arte de evaluar productos de software, ya que hacen que la medida sea independiente del lenguaje o herramienta utilizada en el desarrollo del proyecto, adolecen de la facilidad para hacer el cómputo automático y requieren intervención del hombre para mejorar la precisión con que se estima.

Una de las características de la mayoría de los sistemas que se construyen, es que sus componentes cuentan con líneas de código. Aunque los principales problemas de utilizar líneas de código como métrica de tamaño para estimación del esfuerzo son la falta de una definición universal de *línea de código*, su dependencia del lenguaje de desarrollo y la dificultad de estimar en fases tempranas del desarrollo la cantidad de líneas que tendrá una aplicación, hoy en día sigue siendo una medida atractiva, habida cuenta que la mayoría de los editores de programas proporcionan esta medida de manera automática, o en su defecto, resulta muy fácil hallarla.

Además, si los indicadores a partir de líneas de código son calculados en función de la plataforma de desarrollo (.NET, Java, C++, PHP, entre otras), se podrán establecer comparaciones con otros proyectos sobre plataformas similares. [12]

Herramientas útiles en la obtención de medidas

Al aconsejarse en la actualidad la gestión de proyectos de manera iterativa e incremental, se hace imprescindible el empleo de Sistemas para la Gestión de la Configuración (SGC) ya que el control de las versiones es vital para un desarrollo organizado. Dichos sistemas de control de la configuración ofrecen en efecto el mecanismo adecuado para el control de la actividad sobre las líneas de código (tamaño), tiempos y responsables. Entre ellos, Subversion (SVN) [13] constituye uno de los más reconocidos y utilizados a nivel mundial por proyectos de software.

Por otra parte, en el año 2006 surgió en Internet un proyecto open source denominado StatSVN (Estadísticas basadas en SVN) liderado por Jason Kealey y Gunter Mussbacher [14].

Construida sobre tecnología Java, la herramienta que lleva el mismo nombre, retorna información que existe en un repositorio SVN y es capaz de generar varias tablas y gráficos que describen la fase de implementación de un proyecto dado (línea del tiempo sobre las líneas de código, contribución total en líneas de código y sobre las diferentes carpetas del SVN de cada programador, etc.).

La versión más reciente (0.7.0) es capaz de generar una suite estática HTML o documentos XDOC donde aparecen contenidas las tablas y gráficas descritas. La herramienta, de código abierto, está liberada bajo los términos de la licencia LGPL (Lesser General Public License) basada en StatCVS, otra herramienta similar pero destinada a analizar repositorios CVS, ambas disponibles en SourceForge.net [15], conocido

sitio donde se alojan cientos de proyectos de software libre.

Resultados y discusión

Marco del proceso propuesto

Al haber identificado la herramienta StatSVN que permite recolectar estadísticas de desarrollo a partir de un repositorio SVN, se propone basar la medición del tamaño en líneas de código, con las que se puede medir además la productividad de los trabajadores al poder contar con el control del tiempo.

Sin embargo, los resultados de StatSVN son solo posibles de alcanzar siguiendo una serie de pasos manuales para poder obtener los indicadores de interés sobre un sitio web.

Para automatizar aún más este proceso, se construyó un script denominado *SVNStatBash*. Este servicio es un archivo bash (.sh) para plataforma UNIX, implementado a partir de códigos modificados de varias librerías de StatSVN y una nueva aplicación, *SIEStatSVN*, la cual emplea Hibernate [16] para insertar las medidas colectadas (que se pueden ver en la suite HTML) en una base de datos relacional diseñada sobre PostgreSQL [17].

Por otra parte, es nuestro interés lograr indicadores de productividad que muestren no sólo información sobre la etapa de codificación, sino que además reflejen los datos de las personas que trabajaron y el tiempo que emplearon durante la fase de diseño del software.

Para estos fines, se construyó un producto denominado *Sistema de Gestión de Indicadores de Productividad (SGIP)* que permite gestionar la base de datos PostgreSQL y facilita el registro sobre la misma de atributos relacionados con el proyecto y con los documentos del diseño. Esta herramienta fue construida empleando la plataforma de desarrollo Java.

El marco para la utilización del entorno de herramientas propuesto es mostrado con detalle en el proceso descrito en la figura 3.

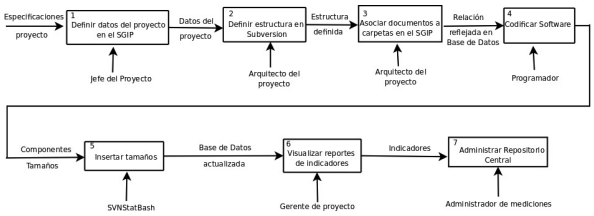

Figura 3 Proceso de obtención de indicadores utilizando control de versiones sobre código fuente y datos iniciales sobre documentación del proyecto

Descripción detallada del proceso

Se ha enumerado en la parte superior izquierda cada actividad del proceso.

Actividad 1**:** El Jefe del Proyecto define los datos generales descriptivos del proyecto de software en el *SGIP*: fecha de inicio, nombre y plataforma de desarrollo. También se capturan atributos de sus documentos de diseño: cantidad de analistas involucrados y las fechas de creación y modificación de los mismos. Estos datos permitirán el cálculo final de los indicadores de productividad y esfuerzo.

Actividad 2**:** El Arquitecto del proyecto define la estructura arbórea de las carpetas del proyecto en el SVN con vistas a la construcción del código fuente. La estructura en principio es libre, es decir, el arquitecto la puede definir en función de la plataforma de desarrollo y utilizando una organización acorde con la arquitectura del proyecto.

Actividad 3: El arquitecto utiliza una ventana auxiliar del *SGIP* para asociar qué documentos de diseño van a guiar el desarrollo de código en cada carpeta del SVN. Para esto aprovecha la misma estructura que tiene definida como copia local del proyecto (ver figura 4). Dicha información se hará persistente en la base de datos del SGIP.

Esta acción permitirá posteriormente al *SVNStatBash*, conocer sobre cuáles carpetas del SVN se desea colectar los tamaños producidos. En caso de ocurrir modificaciones en el árbol de directorios del SVN, el *SGIP* permite que esta información pueda ser actualizada en su base de

datos, manteniendo una correcta correspondencia entre las carpetas actuales del SVN y los documentos de diseño asociados a las mismas.

Figura 4 Ventana auxiliar del SGIP. Relaciona qué carpetas del Subversion servirán de alojamiento al código fuente resultante de los documentos de diseño a implementar

Establecer este tipo de nexo recolectando medidas de las fases de diseño y codificación, conocida la plataforma de desarrollo, incide positivamente en la estimación del tamaño de un módulo de diseño del producto software, ya que se tiene registrada la cantidad de líneas de código con que cuenta dicho módulo.

Actividad 4: El programador realiza la codificación del software y guarda su trabajo en el SVN del proyecto.

Actividad 5: El servicio se ejecuta diariamente a una hora determinada y se encarga de generar, de manera automática, una suite HTML con tablas reportes de las estadísticas del SVN (líneas de código total del proyecto por período de tiempo, actividad de los programadores, etc.), insertando los tamaños de las carpetas del SVN definidas en la actividad 3, en la base de datos del SGIP.

Sirviéndose de las medidas sobre los tamaños y tiempos colectados por *SVNStatBash* y tomando los datos generales de los proyectos, el *SGIP* realiza los cálculos adicionales necesarios, a saber: tamaño por personas meses (productividad) y esfuerzo (personas meses).

Actividad 6: En este punto el Gerente de Proyecto más las instancias interesadas están listas para obtener reportes sobre los indicadores de productividad y de esfuerzo que podrán accederse desde la ventana principal del *SGIP*.

Actividad 7: Las métricas se almacenan en un *Repositorio Central de Medición*. El administrador de mediciones administra el repositorio por medio de un sitio web donde se definen usuarios y roles para el acceso, configurando las opciones del sitio y la apariencia en contenido. Este sitio hace disponibles distintos tipos de gráficas para hacer análisis de los indicadores por meses o por programador.

Con esta serie de pasos y *SVNStatBash* ejecutándose todos los días, se proporciona una base de datos actualizada que funciona como *Línea Base de la Organización del Software* y un *Repositorio Central* de Medición como medio para almacenar los indicadores de productividad y esfuerzo a cualquier nivel deseado (programador, producto, proyecto, organización).

Es importante hacer notar que el tiempo de desarrollo se actualiza cada día, hasta que se defina una fecha de liberación del producto en su totalidad. Por consiguiente, la productividad cambia día a día con los tamaños de los componentes registrados y las personas participantes, y al cierre de cada proyecto se puede contar con las estadísticas finales.

Aquellos componentes que sean eliminados en cualquier SVN de un proyecto por poseer defectos, no se tienen en cuenta por el *SGIP* en los tamaños entregados, de esta manera el nivel de aceptación por parte de los clientes de los productos elaborados se refleja en los cálculos de productividad y esfuerzo.

Implantación del proceso

La figura 5 muestra la propuesta de implantación del proceso dentro de una organización.

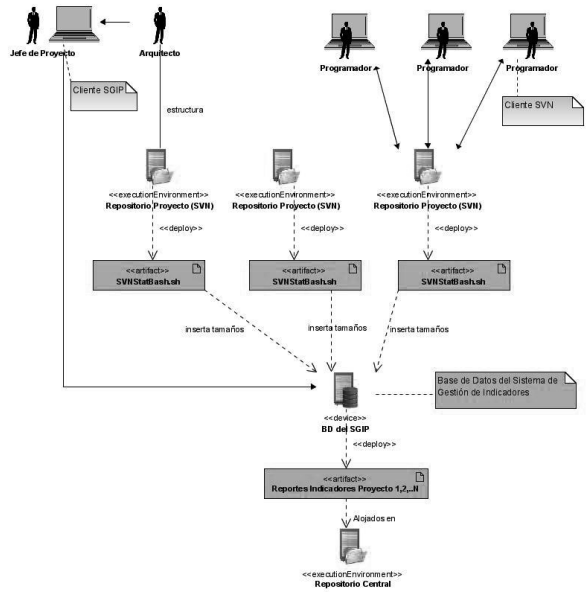

Figura 5 Implantación del proceso propuesto en la organización

En el extremo superior izquierdo se puede apreciar cómo el Jefe de Proyecto utiliza un cliente del SGIP para registrar los datos generales del proyecto en la base de datos (Actividad 1). Por su parte, el arquitecto del proyecto define la estructura arbórea de las carpetas del proyecto en el SVN con vistas a la construcción del código fuente. (Actividad 2). También emplea un cliente de SGIP para relacionar qué carpetas del Subversion sirven de alojamiento al código fuente resultante de los documentos de diseño a implementar (Actividad 3), tal y como se muestra en la pantalla de la figura 4. Los programadores, valiéndose de un cliente de SVN proceden a guardar su trabajo en el repositorio SVN del proyecto. (Actividad 4). Cada ordenador donde esté instalado el SVN aloja el script SVNStatBash, que se ejecuta de manera automática utilizando el administrador de tareas programadas del Sistema Operativo activo (Actividad 5). De esta manera los tamaños producidos se registran diariamente sin necesidad de métodos manuales de medición.

Llegado a este punto la base de datos se encuentra lista para ser consultada y generar reportes con los indicadores de productividad y esfuerzo a partir de la información que contiene. (Actividad 6). Finalmente se hace notar que si cada medida obtenida por proyecto se almacena en un Repositorio Central de Medición (Actividad 7), es posible obtener indicadores de productividad y de esfuerzo a cualquier nivel de la organización y período de tiempo deseados.

La implantación de este entorno de trabajo permite a una organización llevar a cabo un ciclo de mejora continua para el proceso de codificación de productos, tal y como se muestra en la figura 6:

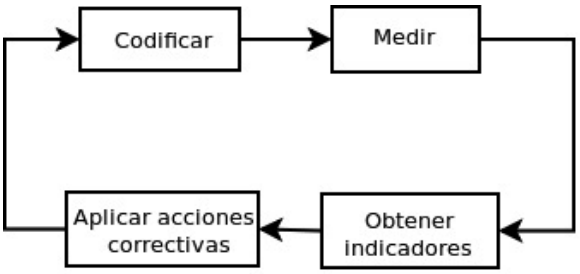

Figura 6 Ciclo de mejora en torno a la colección automática de métricas hacia un repositorio de mediciones

Hacemos notar que la obtención de indicadores debe ir acompañada de un proceso de control del trabajo, donde gráficos de líneas o barras ayuden a encontrar los parámetros de control de la productividad en la codificación. En este caso, medias y tendencias deben apoyar un proceso de toma de decisiones que ayuden a disminuir progresivamente los problemas principales de cada programador, identificados con técnicas de trabajo en grupo. Por ello es que se propone la aplicación de acciones correctivas hacia los programadores que permitan producir un nuevo ciclo mejorado.

Aplicación a un caso de estudio

A continuación se muestra el proceso de obtención automática de indicadores durante

el desarrollo de los productos expuestos en el presente trabajo. La Organización cuenta con un proyecto denominado Calisoft en el cual se desarrollan dos productos que reciben el nombre de SGIP y SVNStatBash. El período de tiempo analizado corresponde a todo el año 2009 (aunque es posible analizar otros períodos de tiempo). Ambos productos se implementan sobre la plataforma de desarrollo Java.

Por medio del sitio web conectado al *Repositorio Central de Medición* se generan distintos tipos de gráficas que permiten analizar el comportamiento del indicador de productividad durante el desarrollo individual por productos (ver figura 7) y del proyecto.

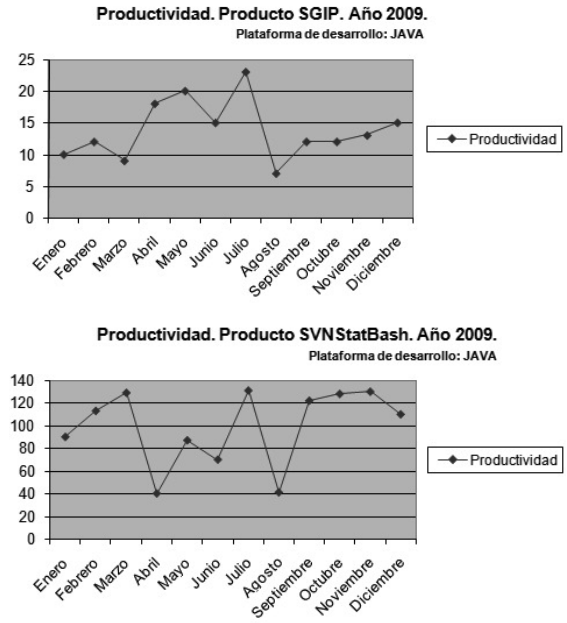

Figura 7 Reporte anual del indicador de productividad para los productos SGIP y SVNStatBash del proyecto Calisoft

El análisis de productividad para el proyecto Calisoft se muestra en la figura 8. Con este gráfico, la Organización encuentra que están fuera del control las productividades en los meses *abril* y *agosto*.

En la figura 9 se pueden comparar las productividades en codificación de los programadores del proyecto Calisoft. Por lo que se concluye que hay que buscar las causas de la baja productividad de María.

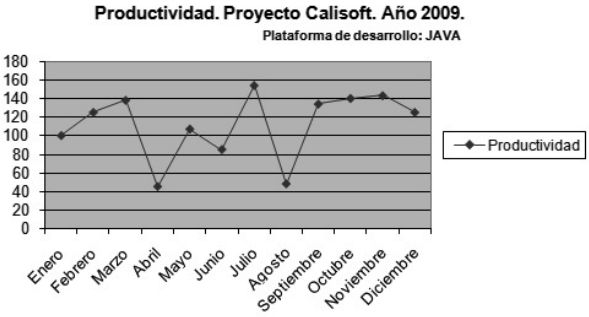

Figura 8 Análisis del indicador de productividad del proyecto Calisoft. Año 2009

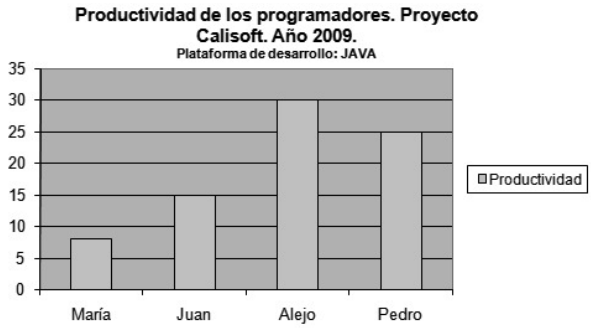

Figura 9 Análisis de productividad por programador del proyecto Calisoft

La figura 10 muestra alguna de las estadísticas recolectadas del producto *SGIP* por *SVNStatBash.* Igualmente pueden ser utilizadas como apoyo para el análisis del indicador de productividad de dicho producto.

Conclusiones

Al elevar el nivel de automatización de la recolección de medidas, aumenta la efectividad y la eficiencia de dicho proceso. El empleo de mecanismos como los expuestos en este trabajo, hace un aporte considerable hacia la cuantificación de los procesos y productos de software.

Tanto el esfuerzo como la productividad producidos se calculan automáticamente durante el trabajo, sin necesidad de emplear actividad

controlada por personas que registren los eventos durante el desarrollo.

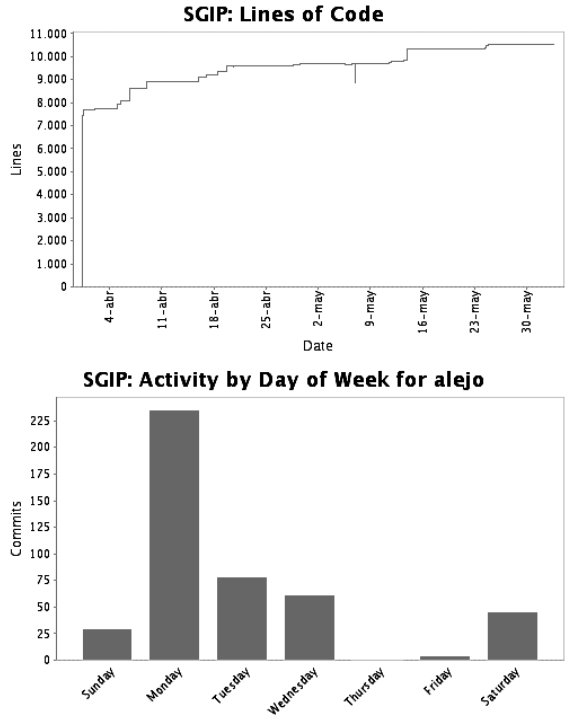

Figura 10 Gráficos generados por SVNStatBash durante un período de tiempo. En la parte superior: el progreso del tamaño (líneas de código) del producto SGIP. En la parte inferior: el número promedio de commits semanales hechos al SVN por uno de sus programadores

Se propone el cálculo de indicadores de productividad y de esfuerzo, por medio del uso combinado de los datos que se obtienen de las estadísticas colectadas por SVNStatBash y de las fórmulas calculadas por SGIP.

El empleo de software libre en el entorno propuesto, permite independencia de licencias privativas y un bajo costo para lograr la implantación del proceso.

La aplicación de técnicas estadísticas a los datos almacenados en la línea base, así como técnicas de softcomputing, abren un campo de investigación relacionado con la estimación de los nuevos proyectos por comparación con los de la línea base.

Referencias

1. R. S. Pressman. *Ingeniería de Software: un enfoque práctico*. 5ta ed. Ed. McGraw-Hill. México.2001. pp. 53-75.
2. IEEE. *IEE Software Engineering Standards*. St. 610.12-1990. pp. 47-48.
3. http://www.computer.org/portal/web/csdl/doi/10.1109/SEW.2003.1270721. Consultada el 27 de abril 2010.
4. http://www.sei.cmu.edu/library/abstracts/news-at-sei/backgroundjun99.cfm. Consultada el 27 de abril 2010.
5. V. R. D. M. Weiss. "A Methodology for Collecting Valid Software Engineering Data". *IEEE Trans. Software Engineering*. Vol. 1984. pp. 728-738.
6. http://www.psmsc.com/. Consultada el 2 de diciembre 2008.
7. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=44344. Consultada el 16 de julio de 2010.
8. G. P. Ana María. "Una nueva métrica para tamaño en ayuda a la gestión de la calidad del software". *2do Taller de Calidad en la Informática y las Comunicaciones.* Evento Internacional Informática. Palacio de las Convenciones. La Habana. Cuba. Vol. 2005. pp. 3.
9. S. Sparks, K. Kapczynski. "The art of sizing projects". *ITWorld.com*. Vol. 1999. pp. 3-4.
10. A. J. Albrecht. "Measuring application development productivity". *Proc. Joint Share/Guide/IBM Symposium on Application Development.* Monterey. Vol. 1979. pp. 83-92.
11. Jones, C. *Applied Software Measurement - Assuring Productivity and Quality*. Ed. McGraw-Hill. New York. 1999. pp. 1-5.
12. G. P. Ana María. "Automatización de la gestión de la calidad de una organización de software partiendo de la gestión de configuración". *Memorias del Evento UCIENCIA*. La Habana (Cuba). 4-6 de julio. 2006. pp. 3-4.
13. http://subversion.tigris.org/. Consultada el 28 de abril 2010.
14. http://www.softwareengineering.ca/statsvn/CSI5140_StatSVN.pdf. Consultada el 8 febrero 2008.
15. http://sourceforge.net/. Consultada el 28 de abril 2010.
16. http://www.hibernate.org/6.htm. Consultada el 15 de diciembre 2007.
17. http://www.postgresql.org/download/. Consultada el 27 de abril 2010.