Métrica para establecer el estado de avance de la etapa de implementación de un proyecto de desarrollo de software

Metric to establish the advance of the software development project in the implementation phase

Arturo César Arias Orizondo, Dailys Díaz Fuentes, Rosario Rodríguez Torres

Universidad del las Ciencias Informática.

{arturo, dfuentes, rrtorres}@uci.cu

Resumen

La UCI[1] como institución ejecutora de grandes desarrollos de software, necesita establecer mecanismos fiables para mantener el control de sus proyectos. El seguimiento del cronograma de trabajo, controlando por fechas el cumplimiento de sus principales hitos, constituye una técnica empleada para estos fines. Sin embargo, cuando aún la UCI no cuenta con un registro histórico de tiempos asociados al desarrollo, los plazos impuestos en los cronogramas suelen ser empíricos y no necesariamente realistas.

Para la crítica etapa de implementación, en la que muchos se interesan por conocer su avance, este problema se agudiza cuando no existen tareas "definidas", ni relaciones entre si y se dificulta el seguimiento del progreso de las mismas. Como consecuencia, la información del avance de la implementación que se brinda es subjetiva, los problemas no se detectan a tiempo y los proyectos concluyen, casi siempre, fuera de fecha. Por esta razón, el presente trabajo propone una métrica que le permita al líder de proyecto cuantificar objetivamente el estado de avance del software durante esta etapa.

La métrica toma en consideración por un lado, los casos de uso a desarrollar valorando su complejidad y por otro, las tareas para implementarlos de acuerdo al flujo de trabajo establecido por la arquitectura del software, teniendo en cuenta que estas tareas tienen un peso específico dentro del tiempo total necesario para implementar un caso de uso.

De la aplicación de esta métrica se obtiene cuantitativamente el estado de avance del desarrollo, así como su evolución en el tiempo, resultando en información precisa, necesaria para la toma de decisiones gerenciales del proyecto. Adicionalmente, el registro histórico de datos que la soportan, constituye una base objetiva para futuras estimaciones.

Palabras clave: Estado de avance, estimación, flujo de implementación, gestión de proyecto, medición, métrica, planificación.

Abstract

The University of Computer Sciences, as a large software projects developing institution, has the need to establish the adequate viable mechanisms to keep a systematic control of the mentioned projects. The following of a correct step-by-step work schedule, in order to go through each one of the goals, is a method used for that mean. However, the university does not count with a time registry associated to the development of the projects, the time-limit in the schedules are usually empiric and sometimes even un-realistic.

For the critical time of implementation, in which several people show their interest in knowing the progress, a problem gets worse, there´s no "defined" tasks, neither relationships among them, all this interferes with the evolution of the plan. As a consequence, the information of this phase is subjective, problems aren´t detected on time and the project concludes, almost every time, out of schedule. For this reason, the present paper proposes a new metrics that will allow the leader of the project to quantify objectively the state of development of the software through this period.

[1] UCI: Universidad de las Ciencias Informáticas

The metrics takes in consideration on one side, the cases of use to develop, measuring the complexity, and on the other side, the tasks to implement them according to the workflow established by the structure of the software, counting on that the tasks have a specific weight inside the total necessary time to accomplish each case.

From the application of the metrics you can obtain quantitatively the state of development and the evolution in time, as exact information, necessary for important managing decisions. Additionally the registry that supports that application is an objective base for futures estimations.

Key words: *Advance state, estimate, flow of implementation, measuring, metric, planning, project management.*

Introducción

La Universidad de las Ciencias Informáticas (UCI) se adentra cada vez más en la producción de software empresarial con el reto de convertir a la industria de cubana de software en un renglón fundamental de la economía del país. Una de sus misiones consiste en obtener productos y brindar servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación, logrando una fuerte relación Universidad-Empresa.

Como universidad joven, la UCI no alcanza aún obtener una base de registros históricos que permitan plantear una estrategia común para estimar y planificar con precisión todos los proyectos.

A pesar de existir varios sistemas de métricas, probadas y aceptadas en cientos de instituciones y empresas prestigiosas del mundo con resultados importantes, en la UCI no se cuenta con un sistema de métricas adaptado a sus características, que le permita evaluar el avance de los proyectos de software de gestión. Por tanto, la institución necesita utilizar un sistema de medida que le permita cuantificar los procesos asociados a la gestión de proyectos.

Para poder triunfar en este sector es imprescindible que estos productos cuenten con la calidad requerida.

Existen diversas formas de gestionar esta calidad, una de ellas, -quizás la más objetiva- es haciendo uso de las métricas del software.

Las métricas no son más que medidas o colecciones de datos de las actividades de los proyectos y recursos. Estas deben ser simples, objetivas, fáciles de coleccionar, fáciles de interpretar y difíciles de malinterpretar. (RUP 2003) Las mismas producen indicadores a partir de los cuales se pueden tomar decisiones importantes.

Las mayores desviaciones que se producen en un proyecto son debidas a deficiencias en el control del mismo. Para asegurar la eficacia de la ejecución es necesario que el líder de proyecto realice un buen seguimiento. Si se miden los procesos y los productos con métricas rigurosas, se puede mejorar y controlar el proceso a partir de los resultados en su ejecución. (Febles 2003)

Las métricas del software son analizadas y evaluadas por el líder de proyecto. Mediante la medición, el líder puede señalar las tendencias (buenas o malas), realizar mejores estimaciones y llevar a cabo una verdadera mejora sobre el proceso de desarrollo.

La medición proporciona al líder de proyecto la evidencia cuantificable que este necesita como apoyo a la toma de decisiones, basándose en anotaciones realistas y no en la subjetividad, hace más visible el desarrollo y le permite anticiparse a los problemas, ayudando así, a que este pueda mantener el control. Además, estas observaciones le permitirán al líder en caso de atrasos durante el proceso de desarrollo, trazar las estrategias necesarias para cumplir con la planificación.

Uno de estos proyectos desarrollados con éxito en la UCI fue el Sistema de Gestión Penitenciaria (SIGEP), el cual tuvo como propósito dar respuesta a las necesidades de gestión, información y apoyo a la toma de decisiones de la Dirección General de Custodia y Rehabilitación del Recluso (DGCRR) de la República Bolivariana de Venezuela. (ARIAS, 2006)

En el proceso de desarrollo del SIGEP, muchas personas estuvieron interesadas en conocer su avance (clientes y directivos). Por lo general la información que se brindaba al principio, tenía carácter subjetivo y no contaba con un alto grado de precisión. Por tanto, fue necesario emplear un método que permitiera medir dicho avance a partir del cual se pudieran tomar las decisiones gerenciales necesarias derivadas de esa información.

Hasta ese momento, no existía una métrica que se ajustara al flujo de trabajo que rigió la etapa de implementación del proyecto SIGEP, teniendo en cuenta que la medición de las actividades en esta disciplina depende de la estructura arquitectónica de cada proyecto y de su equipo

de desarrollo, y que la métrica si bien puede tener un principio general y aplicable a cualquier proyecto, sus parámetros deben reflejar las características del proyecto y del equipo de desarrollo.

Por consiguiente, en el proyecto SIGEP, se crea una métrica para medir el avance del software durante la etapa de implementación. La evaluación de los resultados de su aplicación serviría de base para futuras estimaciones y planificaciones. Los resultados del avance tendrían una base objetiva y no dependería de la subjetividad de la dirección del equipo de desarrollo. La métrica es adaptable teniendo en cuenta que se basa en factores de ponderación que podrían ser calculados teniendo en cuenta las características de cada proyecto.

Fue necesario identificar las tareas de implementación que se ejecutaron en el proyecto SIGEP según su estructura arquitectónica y definir los parámetros de los cuales depende la complejidad de las tareas de la implementación. Se describió cómo se calcula cuantitativamente el avance en la etapa de implementación, se diseñó la métrica mediante una solución matemática y se validó en la práctica.

Formulación de la propuesta

Para conocer el estado del software durante la implementación es necesario medir el avance a lo largo de este flujo. Este puede expresarse como el por ciento de cumplimiento de las tareas de la implementación. Por esta razón, para medir el avance es necesario considerar las tareas de implementación de acuerdo al flujo de trabajo impuesto por la arquitectura del software, su complejidad y los casos de uso a implementar.

De acuerdo a la arquitectura definida para desarrollar una solución de software se definen los roles y actividades que se requieren para la implementación del producto. En una arquitectura por capas podrían ser necesarios roles específicos para implementar cada una de las capas arquitectónicas y en función de la complejidad de estas tareas el tiempo que se requiere para implementar un caso de uso puede variar.

El proyecto de desarrollo que se toma como base para formular esta propuesta utilizó una arquitectura de tres capas sobre frameworks de peso ligero (spring, hibernate) y sobre el diseño arquitectónico se definió un conjunto de tareas de implementación.

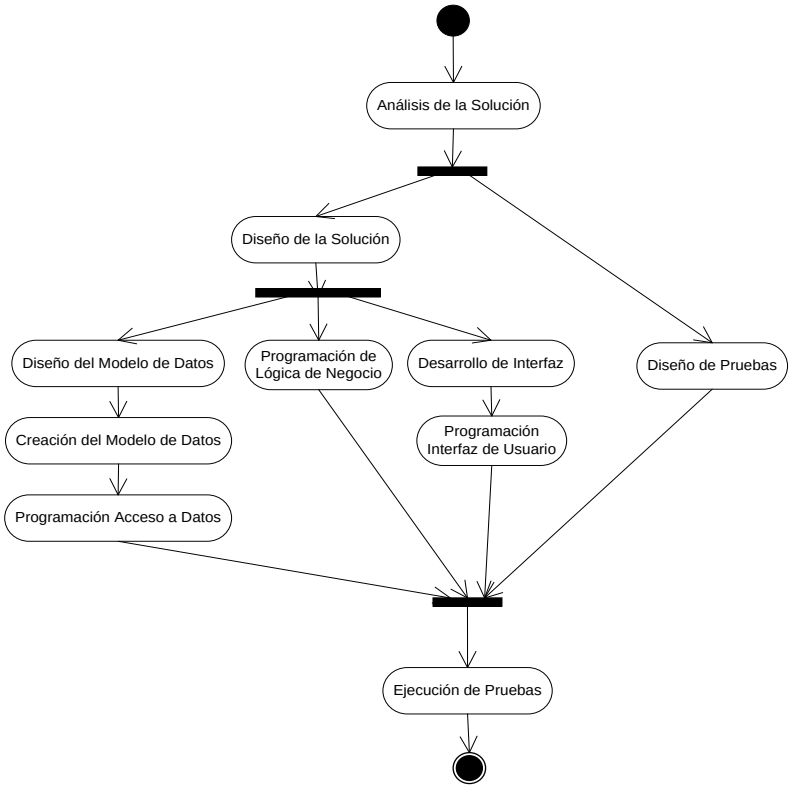

Fig. 1. Secuencia de tareas que definen el flujo de implementación

A cada una de las tareas se le asignó un peso, que representa el por ciento del tiempo total que toma su ejecución como parte de la implementación de un caso de uso. Nótese por tanto, que la suma de todos los pesos resultaría un 100%. Estos pesos se fueron ajustando a lo largo de las iteraciones del desarrollo para que expresaran con mayor fidelidad la complejidad de las tareas y el esfuerzo en tiempo requerido para ejecutarlas.

Tabla 1. Descripción y peso de las tareas de implementación

Tarea de implementación	Descripción	Involucrados	Peso
Análisis de la solución	Consiste en el análisis de la documentación resultante de la captura de requisitos para conocer a profundidad el sistema que se desea construir. Concluye una vez que los involucrados en la implementación de las funcionalidades tengan claridad de lo que se desea implementar.	Analistas, Diseñador, Programadores, Diseñador de base de datos, Diseñador de pruebas	4%
Diseño del modelo de datos	Partiendo de un modelo lógico obtenido durante la captura de requisitos, se definen las estructuras de base de datos que darán soporte de persistencia a la solución de software.	Analista, Diseñador, Diseñador de base de datos	7%
Creación del modelo de datos	Creación de las estructuras y objetos de base de datos en el sistema de gestión seleccionado.	Administrador de base de datos	2%
Desarrollo de la interfaz	Construcción del prototipo de interfaz de usuario a partir del prototipo resultante de la captura de requisitos y basado en las pautas de diseño establecidas. Constituye el punto de partida para la programación de interfaz	Realizador gráfico, Analista	11%
Diseño de la solución	Es considerada una macro actividad que agrupa actividades relacionadas con varios roles y áreas del desarrollo: • Diseño de interfaces de negocio: Consiste en crear las interfaces que encapsulan la lógica de negocio sobre una o varias entidades. • Diseño de entidades de dominio: Definir las entidades de dominio involucradas en la solución del problema. • Diseño de interfaces de acceso a datos: Consiste en crear las interfaces de los DAOs, así como los métodos necesarios para responder a las funcionalidades. • Diseño de la capa de presentación: Se definen las vistas y el flujo de navegación, las posibles peticiones del usuario y los controladores que las atienden. Finalmente se definen componentes del cliente, como son clases JavaScript, documentos HTML, imágenes, etcétera.	Diseñadores, Programadores de Interfaz de usuario, Programadores de acceso a datos	10%
Programación de acceso a datos	Consiste en implementar la capa de acceso a datos. Tiene que estar creada la base de datos y las entidades de dominio. Se concluye cuando se hayan realizado las siguientes actividades: • Realizar los ficheros de mapeo: Consiste en crear los ficheros de mapeo del Framework Hibernate. • Programación de interfaces: Consiste en crear las clases que implementan las interfaces de los DAOs. • Realizar pruebas unitarias: Realizar pruebas unitarias a las implementaciones de los DAOs.	Programador de acceso a datos	9%

Programación de lógica de negocio	Se implementan los métodos definidos para cada una de las interfaces de los managers, ajustándose a las funcionalidades previstas.	Diseñador	9%
Programación de interfaz	Implementar los controladores de spring, validaciones, programar la lógica del cliente, realizar pruebas al funcionamiento de la interfaz.	Programador de interfaz de usuario	33%
Pruebas de calidad	Considerada una macro actividad debido al grupo de actividades que deben desarrollarse para darle cumplimiento. Estas actividades son: • Estudio de la documentación: Consiste en el estudio de la documentación generada por el analista durante la captura de requisitos, hasta lograr ser un especialista del negocio. • Diseño de Casos de Prueba: Construcción de todos los posibles caminos de ejecución, o escenarios, de cada caso de uso. Se obtiene como resultado un listado final con los casos de prueba identificados a partir de los posibles escenarios, los resultados esperados para cada caso y las condiciones o valores requeridos para la ejecución de los distintos escenarios. • Diseño de Juego de Datos: Para cada escenario de caso de prueba se identifican sus valores de prueba, es decir, se precisan los datos de prueba. • Ejecución de las Pruebas: Garantiza la calidad del sistema desarrollado, verificando que todas las funcionalidades han sido correctamente implementadas. No concluye hasta que las funcionalidades sean liberadas una vez resueltos los defectos detectados.	Diseñador de casos de prueba, Probador, Analista	14%

La realización de estas tareas hace posible que se realicen los propósitos de la implementación. Desde el punto de vista de los casos de uso, ninguno habrá concluido hasta tanto no hayan sido realizadas cada una de las actividades anteriores.

Por este motivo a cada una de las tareas definidas se le otorga, para cada caso de uso, un Estado de completamiento entre 0 y 1: 0 si aún no se ha realizado y 1 si ya la tarea se encuentra terminada; un valor decimal que se encuentre entre estos dos números representa el progreso de la tarea dado por el responsable de ejecutarla y basado en las actividades definidas dentro de su flujo de trabajo que aún le restan por terminar.

Así el estado de avance de un caso de uso puede definirse como:

$$\text{Avance por Caso de Uso } (ACU_i) = \sum_{j=1}^{n} (\text{Estado de completamiento}_{ij} \times \text{Peso de la tarea}_j)$$

Donde: Peso de la tarea corresponde al valor asignado a la tarea (j).

Otra dimensión a analizar para establecer el avance de la implementación es la complejidad de los caso de uso a desarrollar. La existencia de casos de uso de diferente complejidad implica que el tiempo de realización de una tarea no sea el mismo para todos los casos. Por ello, para medir el avance de un módulo conformado por varios casos de uso, sería erróneo calcularlo como el promedio del avance de estos.

Los casos de uso se clasifican según su complejidad en: alto, medio o bajo. Esta clasificación es concedida por el Diseñador en función de los siguientes aspectos:

- Número de elementos de datos.
- Lógica de negocio.
- Interfaces externas, dígase dispositivos u otro sistema.

En un módulo solo se presenta una de las combinaciones posibles de complejidad de los casos de uso que lo conforman. Para cada combinación se estableció un peso específico, valor llamado Peso de la complejidad del caso de uso (Pccu) y es el que permite ponderar la diferencia de esfuerzo para desarrollar los casos de uso de diferente complejidad.

Tabla 2. Relación de los Pesos de la complejidad del caso de uso según las combinaciones posibles.

Caso	Combinación posible	Peso CU complejidad alto ($Pccu_{alta}$)	Peso CU complejidad medio ($Pccu_{media}$)	Peso CU complejidad bajo ($Pccu_{baja}$)
1	Existen casos de uso con complejidad alta, media y baja	0,5	0,3	0,2
2	Todos los casos de uso tienen complejidad alta	1	0	0
3	Existen casos de uso con complejidad alta y media	0,6	0,4	0
4	Existen casos de uso con complejidad alta y baja	0,8	0	0,2
5	Todos los casos de uso tienen complejidad media	0	1	0
6	Existen casos de uso con complejidad media y baja	0	0,6	0,4
7	Todos los casos de uso tienen complejidad baja	0	0	1

Utilizando las posibles combinaciones de complejidad de los casos de uso que componen un módulo y partiendo del valor del avance de cada caso de uso, es posible determinar el avance total del módulo. Esto se obtendría mediante la suma del producto entre el Peso de complejidad y el promedio de avance de los casos de uso para cada uno de los niveles de complejidad, lo que quedaría representado por la siguiente expresión:

Avance por Módulo (AM)= $Pccu_{alto}$ x $PAcu_{alto}$ + $Pccu_{medio}$ x $PAcu_{medio}$ + $Pccu_{bajo}$ x $PAcu_{bajo}$

Donde PAcu alto es el promedio de avance de los caso de uso de complejidad alta, PAcu medio es el promedio de avance de los caso de uso de complejidad media y PAcu bajo es el promedio de avance de los caso de uso de complejidad baja.

		4%	7%	2%	11%	10%	9%	9%	33%	14%					
	Complejidad	Análisis de la solución	Diseño del modelo de datos	Creación de modelo de datos	Desarrollo de interfaz	Diseño de la solución	Programación de acceso a	Programación de lógica de negocio	Programación de interfaz	Pruebas internas de calidad	% avance	% avance relativo			
Capacidades (Iskael)												**97%**		alta	0
Gestionar locales del área de reclusión.	Media	1	1	1	1	1	1	1	1	0,8	97%			media	0,6
Gestionar cupos.	Media	1	1	1	1	1	1	1	1	0,8	97%			baja	0,4
Gestionar afectaciones de locales y cupos.	baja	1	1	1	1	1	1	1	1	0,8	97%				
Consultar estado de cupos o locales	baja	1	1	1	1	1	1	1	1	0,8	97%				

Fig. 2. Seguimiento de tareas de implementación para un módulo**.**

Para determinar el avance de la implementación, se puede considerar aceptable, el promedio de avance de cada módulo.

$$Avance\ del\ Proyecto = \frac{\sum_{i=1}^{n} Avance\ del\ Módulo}{n}$$, donde n representa la cantidad de módulos.

La siguiente tabla muestra un resumen, para una semana dada, de la situación del proyecto, donde se puede observar la lista de los módulos que corresponden a la iteración y la cantidad de funcionalidades (casos de uso) a implementar en cada uno de ellos.

Total de semanas	**9**
Semana actual	**8**
Semanas que quedan	**1**

No	Módulos	Funcionalidades	Avance	Semanas para terminar	Estado del desarrollo
1	Requisas y decomisos	22	98%	0,2	En tiempo
2	Ubicación	3	98%	0,2	En tiempo
3	Pertenencias	7	96%	0,3	En tiempo
4	Novedades	10	97%	0,3	**En tiempo**
5	Medidas disciplinarias	6	90%	0,8	**En tiempo**
6	Capacidades	4	97%	0,3	En tiempo
7	Armamento	17	85%	1,4	**Atrasado**
8	Dotación del interno	5	95%	0,4	En tiempo
9	Educación	12	92%	0,7	En tiempo
10	Trabajo	6	86%	1,3	**Atrasado**
11	Deporte y Cultura	20	91%	0,8	En tiempo
12	Evaluaciones Técnicas	6	97%	0,3	En tiempo
13	Reportes	20	65%	4,4	**Atrasado**
		138	**91%**	**0,8**	En tiempo

Fig. 3. Resumen del avance por módulos para una semana dada.

Se puede observar que la tabla expresa las semanas que necesita el módulo para terminar. De los datos anteriores se puede establecer la siguiente relación:

$$\frac{Total\ Semanas}{100\%} = \frac{Semana\ Actual}{Avance\ por\ Módulo}$$

Para obtener las semanas que quedan para terminar, suponiendo que se mantiene el ritmo de trabajo, se puede realizar la siguiente consideración:

$$Semanas\ para\ Terminar = \frac{Semana\ Actual \times 100\%}{Avance\ por\ Módulo} - Semana\ Actual$$

Nótese que existe otro campo en la tabla cuyo nombre es Estado de Desarrollo. El valor por módulo de este campo depende totalmente de los resultados arrojados en el cálculo anterior. Si el resultado obtenido en Semanas por Terminar supera las semanas que quedan para la fecha de entrega establecida por el cliente, el módulo está Atrasado, esto significa que deben tomarse decisiones derivadas de esta información que permitan efectuar la entrega en la fecha acordada. En el caso contrario, es decir, si el resultado obtenido es menor que las semanas que quedan para la fecha de entrega establecida entonces el módulo está En Tiempo.

Análisis de los resultados

La forma más cómoda y visual de lograr un seguimiento del avance por módulos, es a través del análisis del gráfico del estado de avance. Este gráfico muestra claramente en qué estado de avance se encuentran los módulos y por tanto advierte al jefe del proyecto en cuáles tiene que volcar los mayores esfuerzos.

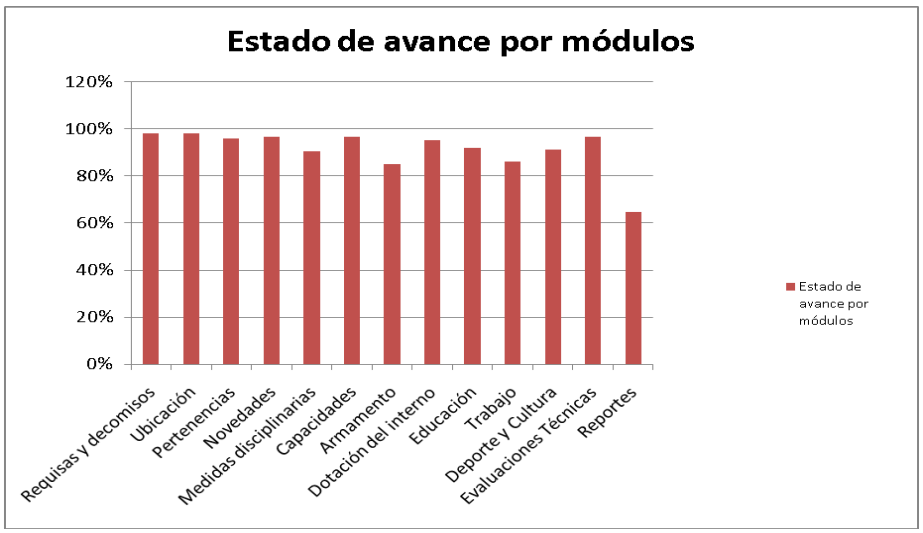

Fig. 4 Estado de avance por módulos

Se puede visualizar además como se ha comportado semanalmente el avance de cada uno de los módulos.

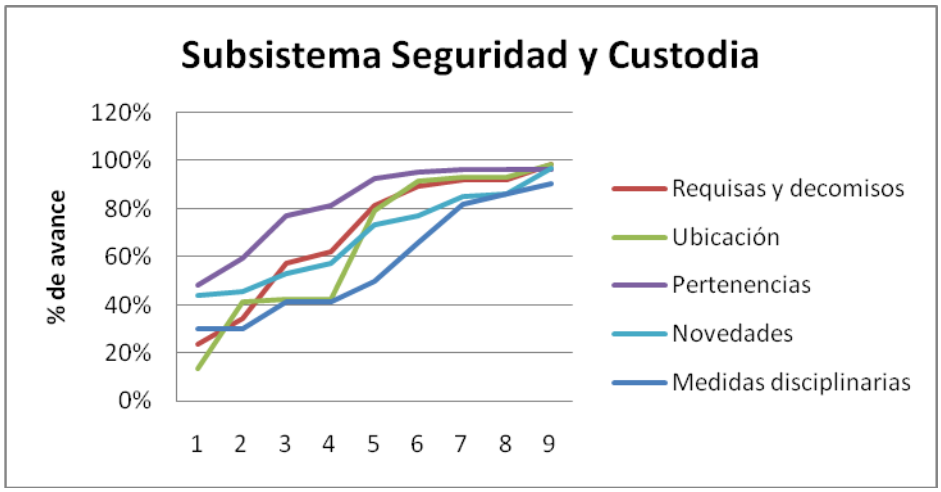

Fig. 5. Evolución semanal de los módulos

En la siguiente figura se muestra un histórico por semanas del progreso de los diferentes módulos y total del proyecto durante el proceso de implementación del software.

		semana1	semana2	semana3	semana4	semana5	semana6	semana7	semana8	semana 9
No	Módulos	7-3-2008	14-3-2008	21-3-2008	28-3-2008	4-4-2008	11-4-2008	18-4-2008	25-4-2008	2-5-2008
1	Requisas y decomisos	24%	34%	57%	62%	81%	89%	92%	92%	98%
2	Ubicación	13%	41%	42%	42%	79%	91%	93%	93%	98%
3	Pertenencias	48%	60%	77%	81%	93%	95%	96%	96%	96%
4	Novedades	44%	46%	53%	57%	73%	77%	85%	86%	97%
5	Medidas disciplinarias	30%	30%	41%	41%	50%	66%	82%	86%	90%
6	Capacidades	25%	47%	67%	75%	84%	92%	93%	93%	97%
7	Armamento	7%	18%	18%	19%	34%	58%	71%	86%	85%
8	Dotación del interno	11%	32%	55%	56%	93%	94%	95%	95%	95%
9	Educación	18%	48%	61%	76%	89%	89%	91%	92%	92%
10	Trabajo	15%	49%	49%	55%	77%	80%	86%	86%	86%
11	Deporte y Cultura	37%	61%	76%	79%	80%	85%	85%	91%	91%
12	Evaluaciones Técnicas	26%	53%	78%	83%	88%	91%	91%	91%	97%
13	Reportes					26%	31%	46%	65%	65%
		25%	43%	56%	61%	73%	80%	85%	89%	91%

Fig. 6. Histórico del avance por semanas

Una representación que resume el comportamiento semanal del avance de cada módulo quedaría de la siguiente manera:

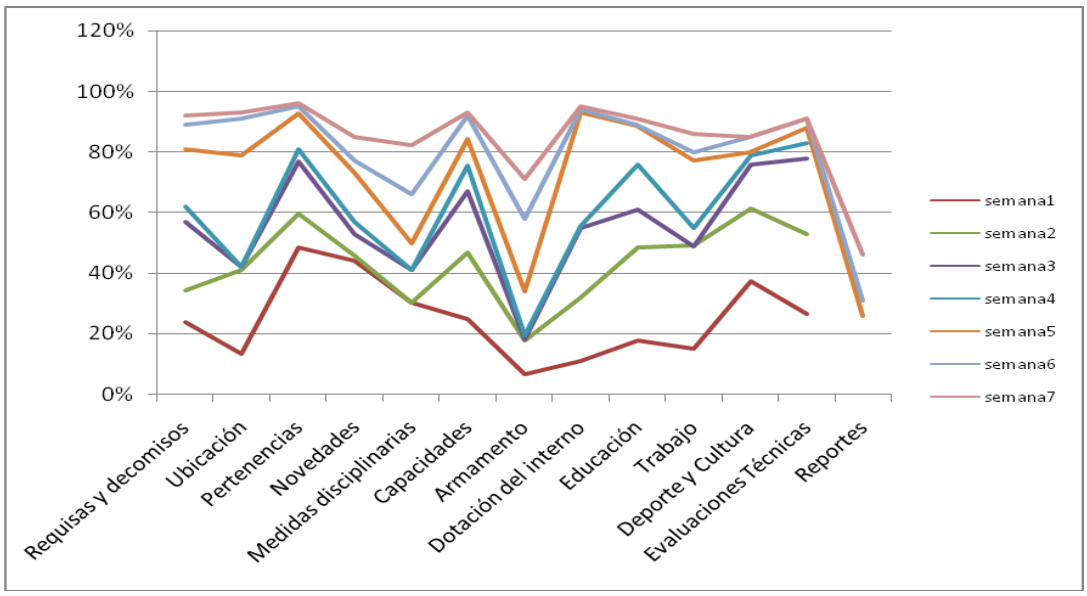

Fig. 7 Comportamiento general por módulos

El avance general del proyecto a partir de los datos de cada semana se representaría gráficamente de la siguiente manera.

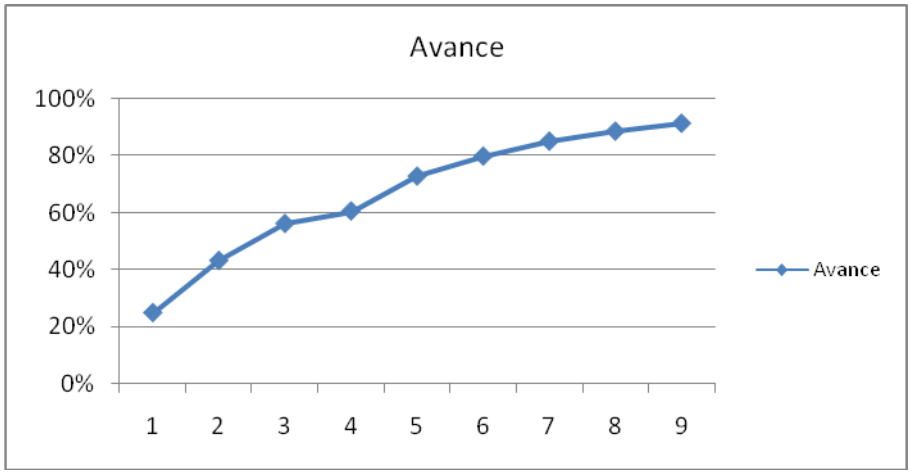

Fig. 8. Histórico del avance del proyecto

Esta gráfica permite observar las tendencias a lo largo del tiempo. Nótese que durante las primeras semanas del desarrollo existe un avance relativo alto. Luego se observa un menor avance en la semana 4 respecto a la semana 3, período durante el cual se efectuaban en la Universidad los IV Juegos Deportivos, los que afectaron como mostró la métrica, el proceso de desarrollo, debido a que la mayoría de los miembros del equipo de proyecto son estudiantes. Luego se observa nuevamente un avance relativamente alto hasta aproximadamente el 80% punto en el cual comienza la etapa de pruebas y donde el avance fue más lento.

Un comportamiento ideal estaría representado por una línea recta en la gráfica del avance general del proyecto; pero esto solo es posible bajo condiciones ideales, las cuales son muy difíciles de obtener debido entre múltiples factores, a que los miembros del grupo de desarrollo no tienen la experiencia suficiente y no se dedican al trabajo a tiempo completo. Además las variables de ajustes introducidas al método deben ser evaluadas para ganar en exactitud. Un ejemplo de esto lo demuestra la desaceleración del avance durante la etapa de pruebas, lo que podría indicar que el peso dado a esta fase debería ser mayor con respecto al otorgado al resto de las tareas de implementación.

Lo anterior determina que el método propuesto no es exacto, ya que en él influyen múltiples factores y las variables son cambiantes; sin embargo, su aplicación arrojó resultados que en la práctica no se alejaron de la realidad y el proyecto cumplió en el tiempo previsto sus compromisos productivos, en parte porque pudieron ser tomadas a tiempo las medidas gerenciales pertinentes cuando se detectaron desviaciones.

Conclusiones

La métrica diseñada permitió obtener cuantitativamente el estado de avance de la implementación del software y constituye en sí un registro histórico de la productividad del trabajo.

Para establecer correctamente el estado de avance de la implementación deben quedar bien definidas las tareas de esta disciplina en función de la arquitectura utilizada, las características del equipo de proyecto y su estructura de roles.

Mientras mayor nivel de detalle se tenga de las tareas de implementación y le sean asociadas a estas un peso que se corresponda con la fracción del tiempo total que se necesita para ejecutarse, mayor exactitud tendrá el valor del avance que genera la métrica.

Se puede determinar hasta qué cantidad de casos de uso un equipo de desarrollo puede implementar en un tiempo determinado. Para el equipo de desarrollo del SIGEP se estima que en 11 semanas de trabajo se pueden implementar de 130 a 140 casos de uso.

Se creó un punto de partida para desarrollar un método científico de medición de software.

Se sentaron las bases para comenzar a obtener registros históricos de los proyectos desarrollados en la UCI.

La métrica puede ser utilizada en otros proyectos de desarrollo de software, redefiniendo las tareas de implementación y su peso especifico dentro del tiempo total de desarrollo de un caso de uso.

Recomendaciones

La aplicación de esta métrica debe hacerse por el líder de proyecto en reunión semanal con los equipos de desarrollo. Los resúmenes semanales proveen una síntesis del estado del proyecto desde el punto de vista del trabajo realizado en la semana.

Es recomendable que estos resúmenes se usen como punto de partida para la planificación de la semana siguiente.

La información obtenida debe ser enviada tanto a los directivos como a los miembros del equipo de desarrollo. Puede ser gratificante para los desarrolladores ver gráficamente el estado de avance de los módulos en los cuales trabajan y su evolución. El proyecto en general conoce donde está y lo que le resta para terminar.

Referencias Bibliográficas

ARIAS, O. C. A. *"Proyecto Técnico de Asesoría Especializada, Colaboración Médica Odontológica, Comunicación Institucional y Solución Tecnológica para apoyar la modernización del Sistema Penitenciario de la República Bolivariana de Venezuela"*. 2006. p.

FEBLES, A. *Un modelo de Referencia para la Gestión de Configuración en la PYME de Software*. Ciudad de la Habana, Instituto Superior Politécnico “José Antonio Echeverría”, 2003. p.

PRESSMAN, R. S. *"Ingeniería del Software. Un Enfoque Práctico"*. 1998a. p.

RUP. *Rational Unified Process*. 2003. p.