aemes TI
Asociación Española para la Gobernanza, la Gestión y la Medición de las Tecnologías de la Información

Revista de Procesos y Métricas

1 de diciembre

2011

VOLUMEN 8, NÚMERO 1, ENERO- DICIEMBRE 2011
ISSN 1698-2029

De las Tecnologías de la Información

Revista de Procesos y Métricas

De las Tecnologías de la Información

Volumen 8 Número 1

Revista fundada por la Asociación Española para la Gobernanza, la Gestión y la Medición de las Tecnologías de la Información (AEMES) <http://www.aemes.org>

ISSN: 1698-2029. Nº Depósito: M23879-2006

Revista de Procesos y Métricas

De las Tecnologías de la Información

Índice

Volumen 8 Número 1 Enero-Diciembre 2011

Editorial

La Revista de Procesos y Métricas que edita la Asociación Española para la Gobernanza, la Gestión y la Medición de las Tecnologías de la Información (AEMES) inicia una nueva etapa. El número uno del volumen ocho refleja un proceso evolutivo que ha supuesto un nuevo equipo editorial y un nuevo modo de publicación. En relación al primer aspecto, se han producido cambios en el rol de Editor, en el Consejo Editorial, así como en el Comité Científico de la revista. Así, Ricardo Colomo Palacios forma junto con José Domingo Carrillo el binomio de Editores de la revista. El profesor Dr. Carrillo, presidente de AEMES, venía ocupando el puesto de forma tradicional, sin embargo, el autor de este prefacio sí representa una renovación en el rol. El profesor Dr. Colomo-Palacios desempeña el mismo papel en la revista International Journal of Human Capital and Information Technology Professionals, siendo, además miembro del comité editorial de publicaciones como IET Software y Computer Standards & Interfaces.

En lo tocante al Consejo Editorial, su composición refleja la nueva junta directiva de AEMES en la que profesionales de reconocida trayectoria integran un colectivo de innegable solvencia e influencia profesional. Para cerrar el capítulo relativo al equipo editorial, el Comité Científico se ha refundado contando con la aportación de reputados investigadores del panorama español en la temática de la revista.

La segunda de las novedades es el modo de publicación. Desde la presente entrega de la revista, ésta ya no se edita en formato papel. Ahora, la Revista de Procesos y Métricas se encuentra accesible de forma exclusiva a través de internet. Este modo de publicación permite flexibilizar y agilizar los procedimientos editoriales sin que esta circunstancia represente un menoscabo en lo tocante a su difusión.

La presente entrega de la Revista de Procesos y Métricas contiene cuatro artículos de investigación y una reseña de traducción. El primer artículo por Jerónimo Puerta Hillman López, Ricardo Colomo Palacios y Ángel García Crespo de la Universidad Carlos III de Madrid se titula “Recomendaciones para la adopción de prácticas de gestión del capital humano en entornos ágiles bajo SCRUM”. Este trabajo, partiendo de iniciativas de reconocido éxito y prestigio, como SCRUM y People-CMM, dicta un conjunto de recomendaciones para la incorporación de prácticas People-CMM en diferentes Time Boxes de SCRUM.

El segundo artículo está firmado por Carmen Cobo García de Sopra Group. El trabajo, titulado “Un modelo de penalización para suministro de servicios” analiza los acuerdos de niveles de servicio (SLA) y proporciona un modelo matemático basado en una curva logística de crecimiento que pretende establecer sanciones para las empresas proveedoras de servicios en caso de incumplimiento de los SLA.

Seguidamente se incluye “NDT-Suite, una solución práctica para el uso de NDT”, un trabajo realizado por Julián Alberto García García, Daniel Rivero Capellán, María José Escalona Cuaresma e Isabel Ramos Román. En el artículo se presenta NDT-Suite, un conjunto de herramientas para el soporte a la Ingeniería Web.

El último artículo de investigación es “Medición de la Productividad de los Puestos de Trabajo en Ingeniería del Software”. En él, sus autores, Adrián Hernández-López, Ricardo Colomo Palacios y Ángel García Crespo de la Universidad Carlos III de Madrid llevan a cabo una revisión de la literatura relativa a las métricas de productividad definidas para los diferentes puestos de trabajo de la Ingeniería de Software.

El número cierra con una reseña firmada por D. José Luis Lucero, Vicepresidente de AEMES, en relación a la traducción que la Asociación ha llevado a cabo del Manual del IFPUG en su versión 4.3.1. Dicho documento se encuentra disponible en la web de la Asociación para los asociados.

El equipo editorial desea que esta nueva etapa cimiente el crecimiento de la revista y permita conseguir una mayor proyección e impacto y, con ello, atraiga temas de debate en un foro que, en el ámbito del idioma castellano, ostenta un liderazgo que se pretende afianzar.

Ricardo Colomo-Palacios

Recomendaciones para la adopción de prácticas de gestión del capital humano en entornos ágiles bajo SCRUM

Jerónimo Puerta Hillman López, Ricardo Colomo Palacios, Ángel García Crespo

Dpto. Informática

Universidad Carlos III de Madrid

Madrid - España

jeronimophl@gmail.com; {ricardo.colomo, angel.garcia}@uc3m.es

Abstract: *Over the last decade software engineering has been experimented agile methodologies appearance. Scrum is one which has many supporters. Due to its apparent ease of application, it has been adopted by different size organizations. However quality less management is evident in those methodologies, aside other problems. Also has been necessary to have a process and technology expertise team. Because of these factors many enterprises has been failed on adopting these methodologies. This paper is oriented to help organizations in Scrum adoption and improve, in order to fully exploit all the methodology benefits.*

Resumen: *En la última década la ingeniería de software ha experimentado la llegada de las metodologías ágiles. Scrum es una de ellas con muchos adeptos. Debido a su aparente facilidad de aplicación ha sido adoptadas por muchas organizaciones sin importar su envergadura. Sin embargo se evidencia que estas metodologías no se centran en la gestión de calidad entro otros problemas. Así también se ha visto que es necesario tener un equipo con conocimientos avanzados en tecnología y sobre todo en el proceso. Debido a estos factores muchas empresas han fracasado en la adopción de estas metodologías. Este artículo está orientado a ayudar a la adopción y mejora de Scrum por parte de las organizaciones para explotar completamente las bondades de la metodología.*

Keywords: *Scrum, People-CMM, Human Factors, Agile*

1. Introducción

En los últimos 50 años, el software ha pasado de ser una herramienta tecnológica para la solución de problemas específicos, a convertirse en industria, la cual está presente en la mayoría si no en todos los procesos de negocio de la actualidad. Sin embargo debido a esta evolución surgen mayores exigencias aparte de la funcionalidad, tales como calidad, reducción de costes, eficiencia, tiempo de salida al mercado entre otras. Al ser una industria relativamente joven, es que se detectan imperfecciones en el proceso de desarrollo de software. Muchos autores denominan como “crisis del software” al fenómeno que se presenta cuando el desarrollo de un software excede el presupuesto y/o no se entrega dentro de los límites establecidos, o en su defecto no cumple los requerimientos funcionales acordados [1].

En procura de solucionar la crisis, muchos investigadores han enfocado sus estudios en el desarrollo de marcos de trabajos, metodologías, modelos, con el fin de mejorar el proceso de desarrollo de software partiendo desde su concepción hasta la puesta en el mercado. Algunos han centrado su esfuerzo en la mejora de los procesos, institucionalizando ciertas prácticas para alcanzar mayores niveles de madurez y capacidad [2].

Por otra parte, en los últimos años han tomado forma las llamadas metodologías ágiles, las cuales se centran en “manifiesto ágil” [3], el cual partiendo de la experiencia de algunos expertos propone lo siguiente:

Basado en este manifiesto y tomando como referencia la importancia de las cuatro “P” (Proyecto, Producto, Procesos, Persona) [4], se empiezan a considerar a las personas como un factor clave en el rendimiento de la industria del software. Estas metodologías han sido de gran aceptación en las organizaciones, puesto que su

implementación no representa una inversión importante, y su adopción es relativamente sencilla y se realiza en corto tiempo. Sin embargo estas metodologías son muy criticadas por aportar un bajo nivel de calidad al producto, no realizar una gestión de riesgos adecuada y generar escasa documentación [5]. A su vez se necesita de un equipo de profesionales altamente competentes tanto en tecnología como en la metodología.

El siguiente artículo propone la aplicación de algunas prácticas sobre el capital humano adoptadas del modelo de madurez y capacidad para personas P-CMM [6], en entornos ágiles de desarrollo de software, tomando como ejemplo al marco de trabajo propuesto por SCRUM [7].

El resto del artículo se organiza de la siguiente manera. En el apartado segundo se aborda un resumen del marco teórico acerca de las herramientas utilizadas para la elaboración del artículo. A continuación se recomiendan algunas prácticas de gestión del capital humano para incrementar la fuerza de trabajo y aplicadas a ciertos "timebox's" de SCRUM. En el cuarto apartado, discutimos y presentamos los resultados esperados de este artículo, por último se presentan las conclusiones y recomendaciones para continuar con la investigación en un futuro trabajo.

2. Antecedentes

2.1. Scrum

Scrum es una metodología ágil de desarrollo de software, el cual tuvo a su primer equipo creado y liderizado por Jeff Sutterland en Easel Corporation en 1993, y formalizado bajo un marco de trabajo por Ken Schwaber en 1995 [8]. Scrum está fundamentado en la teoría empírica del control del proceso de desarrollo de software, bajo un enfoque iterativo e incremental, soportado por tres pilares fundamentales, la transparencia, la inspección y la adaptación [7].

El marco de trabajo de Scrum (ver figura 1) consiste en un conjunto de equipos, artefactos, reglas y cajas de tiempo, los cuales resumiremos a continuación.

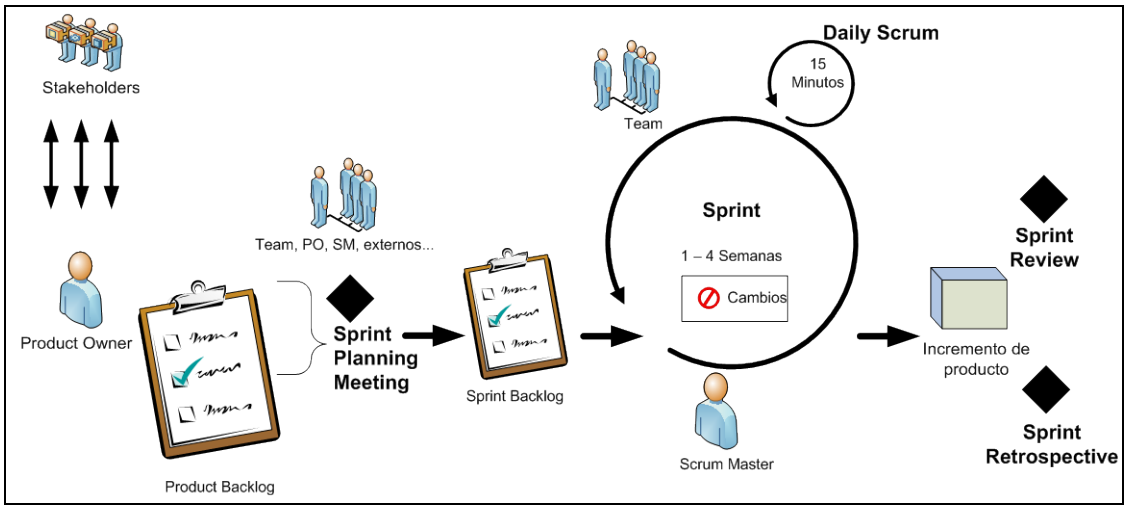

Figura 1. Scrum

El equipo de Scrum, está formado por tres roles [9]:

El *"Scrum Master"*, el cual es el encargado de llevar el proceso adelante dentro de la organización, además de asegurar que todos en el equipo sigan las practicas y reglas de Scrum. Este rol solo puede ser una persona, y no puede compartir el rol de *"Product Owner"*, pues tendría un conflicto de intereses de acuerdo a las competencias de estos roles [7].

El *"Product Owner",* es también una sola persona que representa los intereses de quién está detrás del proyecto, i.e. de los "clientes" (*stakeholders*), es responsable de maximizar el valor de trabajo que realiza el equipo y el retorno de inversión (ROI) esperado por el producto, ya sea de forma comercial o en desempeño [8]. Tiene como su principal tarea mantener actualizada y ordenar la lista de funcionalidades del proyecto, la cual en Scrum llamamos el *"Product Backlog"*.

Finalmente tenemos el *"Scrum Team"*, el cual es responsable del desarrollo de las funcionalidades requeridas, transformando el *"Product Backlog"* en incrementos de funcionalidades utilizables durante cada *"Sprint"*. El *"Scrum Team"* Tiene la particularidad de ser un equipo auto gestionable, y multifuncional, en donde los miembros presenten la habilidades necesarias para el lograr ese incremento. No existen sub- equipos dentro del *"Scrum Team"*, ni tampoco existen divisiones por aéreas técnicas o especialidades.

Para saber el estado del proceso de acuerdo a las funcionalidades requeridas para el producto, Scrum propone cuatro artefactos muy sencillos:

Se presenta una lista de funcionalidades requeridas para el producto, el *"Product Backlog"*. Esta lista debe estar ordenada de mayor importancia y nivel de riesgo.

El *"Release Burndown"*, es un grafico que nos dice cuanto trabajo queda por realizar del "Product Backlog", de acuerdo a la estimación de tiempo realizada.

Para cada iteración se tiene el *"Sprint Backlog"*, que es una lista de tareas para convertir las funcionalidades del *"Product Backlog"* seleccionadas, en un incremento de producto utilizable durante una *"Sprint".*

Por último tenemos el *"Sprint Burndown"*, el cual es un grafico que nos dice cuantos ítems quedan por realizarse del *"Sprint Backlog"* de acuerdo al tiempo que dura el *"Sprint"*.

En la nueva guía de Scrum, sus creadores han retirado los *"Burndown Charts"* [10], pudiendo así el equipo utilizar las herramientas o técnicas que mejor se les ajuste a su realidad y modelo de negocio actual.
Conociendo los artefactos a producir y los roles de las personas involucradas en el método nos quedan los *"time boxes"*. Estos son hitos que definen el funcionamiento del proceso. Scrum nos presenta los siguientes:

Se empieza con la *"Release Planning Meeting"* aunque es opcional y en la nueva versión de Scrum ha sido desestimada [10]. Esta es una reunión donde se realiza el plan de entregables del producto, este plan debe contener las funcionalidades, los riesgos, fecha y costo estimado que de cada entrega. La *"Release Planning Meeting"* produce la estimación y el orden del *"Product Backlog"* para la entrega.

Luego se lleva a cabo la *"Sprint Planning Meeting"*, reunión en donde se planea la iteración donde se desarrollan las tareas, está dividida en dos partes. La primera en donde se decide *Qué* es lo que se va a realizar

del *"Product Backlog"*. Y la segunda donde se dice *Como* se realizaran las tareas para obtener el incremento deseado. Generalmente cada ítem del *"Product Backlog"*se desglosa en una o varias tareas. Se recomienda que la duración total de esta reunión sea de 8 horas por mes de trabajo.

El *"Sprint"*, es una iteración en la cual el *"Scrum Team"*, desarrolla las tareas para obtener el incremento de funcionalidad del "Product Backlog" acordado, durante la *"Sprint"*, no se pueden introducir cambios a estos requerimientos. Se recomienda que la duración de una *"Sprint"* este entre 2 y 4 semanas.

Al finalizar la *"Sprint"* se lleva a cabo la *"Sprint Review"*, reunión en la que están presente el *"Scrum Team"* en su totalidad y los "stakeholders", para revisar cuáles objetivos se han logrado y cuáles no. A su vez revisar el *"Product Backlog"* para ver los cambios que se han solicitado y revisar el trabajo a realizar en la siguiente *"Sprint"*.

Luego tenemos la *"Sprint Retrospective",* que es otra reunión que se lleva a cabo después de la *"Sprint Review"* y antes de la siguiente *"Sprint Planning Meeting"*, en esta participa solo el *"Scrum Team"*. La finalidad de estas es inspeccionar el proceso en sí, el *"Scrum Master"* revisa y consulta a los demás miembros sobre las prácticas realizadas, herramientas utilizadas y la relación del equipo, esto se hace para mejorar el proceso, es en esta reunión donde se afianzan los pilares de ***inspección*** **y** ***adaptación***.

Finalmente tenemos el *"Daily Scrum"*, el cual es una reunión de 15 minutos, en las que los miembros del equipo se limitan a contestar las siguientes tres preguntas.

1. Qué he realizado desde la última reunión.
2. Qué voy a realizar antes de la siguiente reunión. y
3. Qué obstáculos existen para realizar mis tareas.

Esta reunión aumenta la fluidez de la comunicación entre los miembros del equipo y aumenta el nivel de conocimiento acerca del estado del proyecto y promueve la colaboración entre compañeros del *"Scrum Team"*, la efectividad de esta reunión estará de acorde a la ***transparencia*** que refleje cada miembro del equipo.

Luego de muchos casos de éxito Schwabber propone realizar la "Scrum kick off Metting" [11] para iniciar la adopción de Scrum en la organización. Esta reunión no se presenta como "timebox", básicamente es aquí donde se describe el proceso, se asignan los roles a las personas que formaran parte del proyecto, y se crea un "Transitional Product Backlog" que contiene las tareas necesarias para la implantación adecuada de Scrum.

People CMMPeople Capability Maturity Model® (People CMM®), fu publicado por primera vez en 1995 [12]. Es una guía para implementar prácticas que mejoren continuamente la capacidad de fuerza laboral *"workforce"* en una organización.

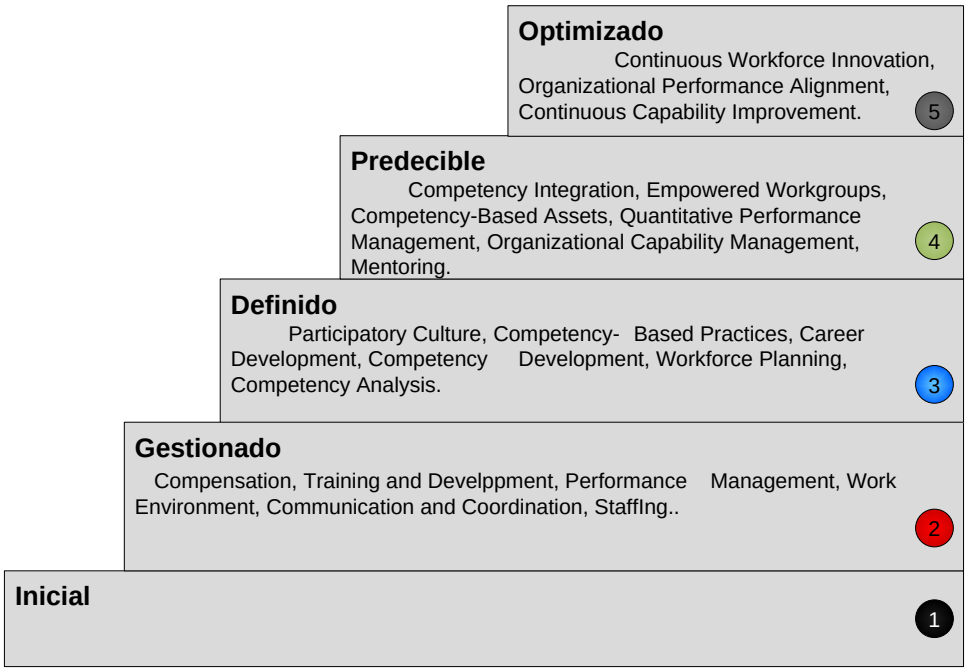

Figura 2. Niveles de People-CMM

PCMM está compuesto por cinco niveles, cada nivel presenta un conjunto de áreas de proceso, las cuales tienen ente tres y cinco objetivos. Para alcanzar un objetivo de un área de proceso es necesario realizar prácticas. La figura 2, nos muestra un resumen de los niveles y las áreas de proceso. A continuación presentamos una breve descripción de los objetivos generales de cada nivel [6].

En el nivel *Inicial*, las prácticas de la fuerza de trabajo tienden a ser inconsistentes, rutinarias y monótonas, es decir la organización realiza las actividades necesarias para su existencia, en este nivel existen muchos defectos en las políticas aplicadas a los recursos humanos y su fuerza de trabajo es la mínima necesaria para cubrir los objetivos.

En el nivel *Gestionado*, los gestores empiezan a realizar prácticas básicas de gestión de personal, obteniendo como resultado la capacidad de gestionar habilidades y el rendimiento de las personas dentro de la organización.

Una vez alcanzado el nivel *Definido*, la organización identifica y desarrolla el conocimiento, habilidades, talento y capacidades intelectuales que son requeridas para realizar las actividades del negocio. Se desarrolla una cultura y profesionalismo basado en conocer a fondo las competencias necesarias para desarrollar la fuerza de trabajo.

El cuarto nivel es llamado *Predecible*, debido a que las personas y/o grupos de trabajo pueden auto gestionar algunos procesos en base a sus competencias. En este nivel la organización se enfoca en explotar el conocimiento y la experiencia de la fuerza de trabajo desarrollada en el nivel 3. Las competencias a nivel de grupo de trabajo son entrelazadas para crear procesos integrados y multidisciplinarios a nivel de la organización. Estos grupos de trabajo se encargan de gestionar sus propios procesos y actividades. Estas prácticas son monitoreadas y corregidas si es necesario. Los mentores asisten a las personas ya sea individual o en grupos de trabajo para desarrollar sus capacidades.

En el nivel más alto de madures llamado *Optimizado*, las áreas de procesos se enfocan en continuar la mejora de la capacidad de fuerza de trabajo de la organización y sus prácticas. Las personas continúan mejorando sus procesos de trabajo personal y los grupos mejoran continuamente sus procesos operacionales a través de la integración de los procesos personales de sus miembros. La organización evalúa y mejora el rendimiento de sus individualidades y grupos de trabajo de acuerdo a los objetivos de negocio de esta.

Cada nivel de madurez de PCMM presenta ente tres a siete áreas de procesos, descritos con un propósito y dirigidos por objetivos. Un área de proceso es un conjunto de prácticas relacionadas que realizadas en la unidad de trabajo, satisfacen los objetivos que contribuyen a incrementar la capacidad y alcanzar un nivel de madurez. Sin embargo algunas áreas de proceso están enlazadas a través de los niveles de madurez por áreas comunes de interés que son dirigidas por el modelo. Estas áreas comunes de interés ayudan al desarrollo de los niveles a transformarse de acuerdo a una o más áreas de proceso. Existen cuatro áreas de comunes de interés, como nos muestra la Tabla 1:

Desarrollar capacidad individual.

Construir grupos de trabajo y cultura.

Motivar y gestionar rendimiento.

Configuración de la fuerza de trabajo.

Maturity Levels	Process Area Threads			
	Developing individual capability	**Building workgroups & culture**	**Motivating & managing performance**	**Shaping the workforce**
5 Optimizing	Continuous Capability Improvement		Organizational Performance Alignment	Continuous Workforce Innovation
4 Predictable	Competency Based Assets Mentoring	Competency Integration Empowered Workgroups	Quantitative Performance Management	Organizational Capability Management
3 Defined	Competency Development Competency Analysis	Workgroup Development Participatory Culture	Competency Based Practices Career Development	Workforce Planning

RPM 8 (1) (2011) ISSN 1698-2029

2 Managed	Training and Development	Communication & Coordination	Compensation Performance Management Work Environment	Staffing

Tabla 1. Áreas comunes de interés para los procesos

Como hemos podido notar, las prácticas a realizar son el núcleo para lograr los objetivos que nos llevan a un nivel de madurez en las áreas de proceso. Estas prácticas pueden ser de dos tipos, las llamadas prácticas de implementación, que describen las actividades a realizar para alcanzar los objetivos. Y por otra parte las prácticas de institucionalización las cuales están organizadas en cuatro categorías, El "compromiso a realizar" y la "habilidad a realizar" describen los prerrequisitos para implementar cada área de proceso y el "Medición y Análisis" y "Verificación de la Implementación", determinan si los prerrequisitos fueron logrados y el proceso ha sido institucionalizado. En la figura 4 se ilustra la arquitectura global de PCMM.

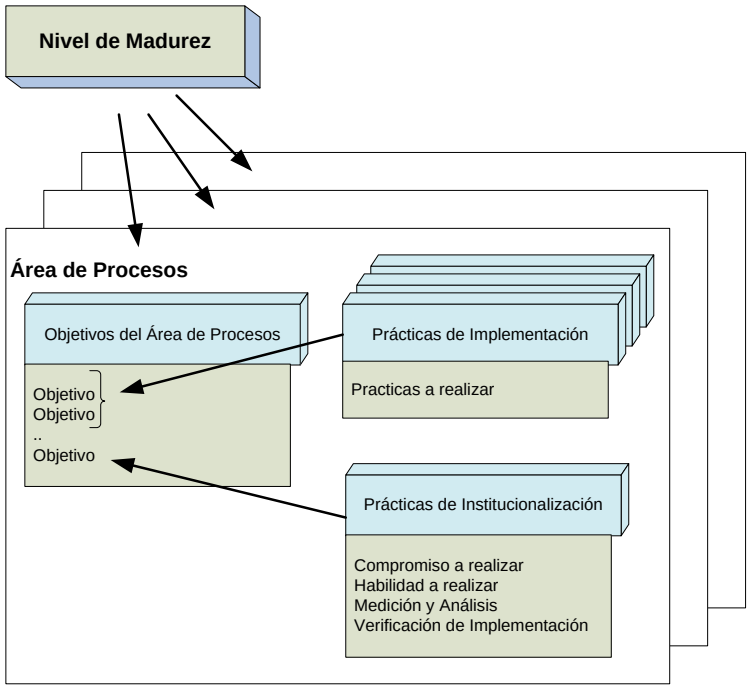

Figura 4. Arquitectura de PCMM

3. Recomendación de adopción de prácticas People-CMM en timeboxes de SCRUM

Existe bastante bibliografía en cuanto a metodologías o modelos que pueden servir de guía a las organizaciones que pretendan aplicar estos conceptos. Sin embargo aunque muchas empresas han tenido éxito, es cierto que también se presentan casos de fallo en la adopción de estas tecnologías. Inclusive existen estudios que presentan contradicciones dentro del marco de trabajo CMM que ofrecen el modelo de madurez del Instituto de Ingeniería de Software (SEI) [13].

Es así que en vista de mejorar el proceso de desarrollo de software, proponemos el siguiente artículo, donde pretendemos aplicar prácticas de capital humano institucionalizadas en entornos ágiles de desarrollo dirigidos por Scrum para la mejora del proceso de desarrollo de software, tomando como referencia algunos estudios realizados anteriormente sobre las prácticas y la mejora del proceso software [14].

El mapeo entre metodologías ágiles y modelos de madurez ha sido últimamente un tema de estudio en la comunidad científica, un ejemplo es el la combinación de prácticas de metodologías agiles, tanto de programación extrema (XP) y Scrum, con directivas de gestión para lograr el nivel dos de CMMI [15]. Por otro lado se ha intentado introducir practicas especificas de XP como la "programación por parejas" en las áreas de proceso *"Mentoring", "Training and Development", "Competency Development"*, entre otras pertenecientes a PCMM [16]. Otro ejemplo es la mezcla de las practicas de Scrum y las áreas de proceso de CMMI [17], en el cual tratan de acercar las diferencias entre estos para poder aplicarlo en empresas que tienen sus procesos dirigidos por el modelo CMMI y están considerando mejorar la agilidad de estos procesos. Y también podemos encontrar casos de estudio en los cuales han utilizado actividades de otras metodologías en Scrum [18].

Sin embargo nuestro estudio plantea un enfoque distinto, en el cual se toma como premisa que el marco de trabajo implantado es una metodología ágil, en el cual las prácticas del marco de trabajo son aplicadas de forma empírica y no institucionalizadas.

Para esto hemos estudiado los objetivos de cada área de proceso, alineado e interceptado con los objetivos de cada "timebox" de Scrum. Se han identificado algunas prácticas de las "áreas de proceso" con el fin de institucionalizar las actividades que se llevan a cabo dentro de los "timebox's".Y de esta forma poder realizar una evaluación del proceso y mejorar estas prácticas en cada "sprint", "release", o proyecto, según sea el caso establecido.

A continuación mostramos los resultados del estudio efectuado. Se han definido dos grandes bloques, recomendaciones de Adopción y de Mejora. En la Tabla 1 identificamos las prácticas que pueden ayudar la adopción de Scrum. Estas prácticas en su mayoría están presentes en los niveles 2 y 3 de madurez de P-CMM. En la Tabla 2 las prácticas para mejorar su aplicación, siendo estas correspondientes a los niveles 3 y 4, aunque se tienen algunas excepciones. Así también tomando en cuenta las prácticas de nivel 5 se plantea unas recomendaciones adicionales de mejoras globales que son ajenas a Scrum.

SCRUM Time Box	**Practica PCMM**	**Área de Proceso**	**Propuesta**
Scrum Kick Off Meeting	Miembros representativos de la unidad participan en las actividades de Staffing.	Staffing	Seleccionar personas que preferentemente: - Hayan trabajado juntas anteriormente. - Entienden el proceso de negocio y el objetivo del proyecto - Estén capacitada con la tecnología seleccionada. Definir la composición del equipo de Scrum, es decir asignar al Product Owner (PO), Scrum Master (SM) e integrantes del Scrum Team (ST).
	En cada unidad se identifican habilidades críticas para realizar cada tarea individual.	Training and Development	Asegurar que las personas seleccionadas entienden SCRUM, sus roles, artefactos y time boxes. Esta tarea es específica del SM.

	Se identifican entrenamientos necesarios para cada persona en habilidades críticas.		Definir un horario de entrenamiento para las actividades específicas requeridas ya sea grupal o individual de acuerdo a la necesidad del ST.
Release Planning Meeting	Cada unidad o equipo analiza el trabajo propuesto para determinar el esfuerzo y habilidades necesarias. Los individuos y grupos de trabajo participan en la toma de compromisos de trabajo que será responsable de realizar.	Staffing	Adecuar la carga laboral a la recomendada por SCRUM. Realizar estimaciones a 6hrs / persona para la primera sprint, tomando en cuenta la curva de aprendizaje del proceso. Utilizar "User Histories" o algún otro método ágil de captura de requisitos para la explicación en la adopción de Scrum.
	Las practicas y políticas de la organización relacionadas al personal deben ser comunicadas a estos.	Comunication and Coordination	Informar al Equipo sobre las fechas de realización de las reuniones dentro del release (Sprint Plannings, Sprint Retrospectives, Sprint Review, duraciones de la Sprint), y mantener esta información accesible a toda la organización.
	Para cada unidad, se identifican ambientes y recursos necesarios para realizar el trabajo comprometido. Se provee del ambiente necesario para realizar el trabajo.	Work Eviroment	Realizar solicitudes de recursos por cada Rol (ST, PO, SM) del equipo. Analizar las solicitudes del equipo, e incluirlos en el presupuesto. Proveer de los recursos (ambientes, herramientas, etc.) para realizar las prácticas de equipo que precisa Scrum y también para descanso del personal.
	Se desarrolla una estrategia de compensación. Se comunica a la fuerza de trabajo la estrategia de compensación de la organización	Compensation	Es necesaria la motivación extra salarial del participante, se recomiendan algunas técnicas de acuerdo a las posibilidades de la organización. - Acceso a cursos de especialización, o certificación de conocimientos empleados en los proyectos. Bonos por logros alcanzados con calidad dentro del tiempo acordado. Ej.: Utilizar un fondo común de bonos para recompensar el rendimiento del equipo en relación a los objetivos empresariales. Este fondo debe ser repartido de acuerdo al salario y responsabilidad de cada persona de modo transparente.
	Se identifican las competencias para el desarrollo del personal en el negocio de la organización Identificar el conocimiento, técnicas y habilidades de proceso para desarrollar las competencias anteriormente identificadas.	Career Development	Realizar una intersección entre los objetivos de negocio de la empresa, y la formación necesaria del personal para el trabajo necesitado en la Organización **(Plan de Desarrollo Personal)**. Para esto se recomienda: - Por ejemplo realizar una matriz con funciones, competencias y capacidades deseadas para los miembros del equipo. - Establecer un "Personal Product Backlog" con los objetivos necesarios del personal y realizar su seguimiento paralelo al proyecto.
Sprint Planning Meeting	La información requerida para realizar el trabajo comprometido es compartida a través de las personas afectadas de manera oportuna.	Comunication and Coordination	Publicar y compartir las especificaciones formales funcionales y no funcionales de cada uno de los ítems del "Sprint Product Backlog" entre los responsables y los afectados.
	Se establecen objetivos de rendimiento medibles basados en	Performance Management	Cada Ítem del "Sprint BackLog" debe ser medible en tiempo, funcionalidad, y calidad.

	el trabajo comprometido y para cada equipo.		Se recomienda adicionar a los requisitos información cuantificable y calificable, como ser: - Estimación de tiempo - Trabajo restante - Dificultad de la tarea. Esta información debe estar de acuerdo a la técnica utilizada para la especificación de los ítems.
Sprint	Las reuniones son conducidas de modo que los participantes utilicen de su tiempo de forma efectiva.	Comunication and Coordination	- El Daily Scrum no debe durar más de 15 minutos. - Cada "timebox" debe contar con una guía de actividades a realizar en ella, las cuales no deben ser alteradas ni quebrantadas.
	Las personas o grupos pueden plantear sus preocupaciones o problemas de acuerdo a un proceso documentado.		Definir un mecanismo para centralizar e informar de los impedimentos encontrados. - Ejemplo, formularios (físicos o automatizados) para que los miembros del ST expresen estas preocupaciones y problemas con el trabajo comprometido en el Daily Scrum. - Los impedimentos que no sean parte del trabajo, o aquellos que no se puedan auto solventar por el Scrum Team, deben ser remitidos al Scrum Master.
	Entrenar periódicamente a las personas o grupos de acuerdo a las necesidades para realizar sus tareas asignadas	Training and Development	Se recomienda designar un espacio de tiempo luego del Daily Scrum para solucionar las dificultades encontradas en este. El equipo auto gestiona la capacitación interna para superar estos obstáculos siempre y cuando la solución a estos no precisen de una capacitación externa. Este entrenamiento debe quedar registrado.
Sprint Review	Se realiza un seguimiento al rendimiento de la organización en el cumplimiento de objetivos estratégicos.	Workforce planning	Documentar las habilidades adquiridas y competencias desarrolladas en el logro del objetivo del sprint (como ser resolución de impedimentos, funcionalidades específicas, etc.) en un gestor de conocimientos manual o automatizado, en adelante llamaremos "Base de Conocimiento Funcional" (BCF). - A su vez agregar a esta información el tiempo y esfuerzo empleado para la obtención de estos objetivos.
	Cada persona documenta su objetivo de rendimiento par al desarrollar capacidades adicionales a las competencias laborales de la organización.	Competency Based - Practices	
Sprint Retrospective	Se realiza el seguimiento de las actividades relacionadas a la resolución de un problema hasta su cierre.	Comunication and Coordination	Establecer un proceso para relevar, seguir y remover impedimentos. Ya sea manual o automatizado. El encargado de gestionar esta información es el SM.
	Los problemas interpersonales o conflictos que degraden la efectividad de las relaciones de trabajo se manejan apropiadamente.		Se debe recordar que el ST es un equipo auto gestionable, y que cualquier conflicto interpersonal debe ser resuelto dentro de este, la gestión es estos conflictos es su competencia, pero se sugiere: - Expresar las molestias de forma abierta al equipo. - Proponer soluciones que no afecten a la unidad del ST. Si se presentase el escenario de apartar a un miembro del ST, esta decisión debe ser unánime, y sobre todo con la participación del SM.
	Se identifican y corrigen los factores ambientales que degradan o ponen en peligro la salud y seguridad del personal. Se Identifican y corrigen factores	Work Enviroment	Establecer políticas para las actividades extra laborales que se ejecutan en el ambiente de trabajo por ejemplo. - Uso de auriculares. - Llamadas telefónicas extra laborales. - Disponer de un lugar de reunión ajeno al ambiente del ST.

	físicos que mermen la eficacia del ambiente de trabajo. Se Identifican fuentes de interrupción o distracción que mermen la eficacia del ambiente de trabajo.		Brindar las condiciones climáticas ideales para el desempeño laboral. Establecer normas de higiene. (por Ejemplo) - Sobre la alimentación. - Espacios comunes de aseo. - Extintores, material sanitario y material de primeros auxilios básicos. - Medicinas para dolencias comunes (dolores de cabeza, menstruales, acidez estomacal, gripe, etc.).
	Se revisan periódicamente o de acuerdo a eventos los objetivos de rendimiento para cada persona.	Performance Management	La transparencia es un pilar importante en Scrum. Los objetivos no alcanzados o limitaciones de conocimiento técnico deben ser informados al SM en el Daily Scrum. Los Sprint Backlog Ítems que sean identificados como muy complejos, subdividirlos o asignar a la persona con mayor experiencia. (es compatible debido a la autogestión del ST).
	Los problemas de rendimiento son discutidos con las personas correspondientes.		Revisar con el individuo afectado las razones que producen el bajo rendimiento. De ser necesario liberar su carga de trabajo, cuando estos problemas sean de índole personal y afecten a la productividad del individuo (Problemas de salud, emocionales, o de cualquier índole relevante a su persona). Esta reunión es recomendable realizarla luego del Daily Scrum.
	Realizar el seguimiento del entrenamiento del equipo.	Training and Development	La preparación del equipo que va a adoptar Scrum en la organización, debe ser documentada, así se podrá aplicar a otros equipos dentro de la organización.

Tabla 2. Recomendación de Practicas de P-CMM para la adopción de Scrum

SCRUM Time Box	Practica PCMM	Área de Proceso	Propuesta
Release Planning Meeting	Se definen mecanismos para la resolución de conflictos y disputas.	Participation Culture	El ST con la colaboración del SM debe definir los procesos para la resolución de conflictos. - Personales(entre los miembros del equipo). - Laborales (técnicos, de calendario, de coordinación). - Extra-laborales (legales, de salud, etc.)
	Los planes de desarrollo de competencia son revisados y examinados periódicamente y basado en eventos.	Workforce Planning	Actualizar el artefacto (BCF) realizado en la recomendación de adopción de Scrum., De acuerdo a los objetivos de cada “Release“ y de negocio de la empresa así como también ante los procesos de Staffing.
	Los candidatos seleccionados son transitados a su nueva posición	Staffing	Establecer un plan de transición para nuevos miembros en el cual se tomen en cuenta. - Descripción del proceso de Scrum (Time Boxes, roles y personas). - Explicación de las políticas organizacionales y del objetivo de negocio de la institución. - Explicación de los recursos utilizados en el ambiente de trabajo (Artefactos, herramientas, etc.).
	Se define objetivos de rendimiento cuantitativos para lograr objetivos de negocio organizacionales.	Quantitative Performance Management	Crear un "**Enterprise Product Backlog**" con los objetivos de negocio de la organización y las tareas necesarias para la mejora de la aplicación de Scrum (no los objetivos del proyecto). Estos deben ser cumplidos y revisados por la administración. Para esto se recomienda que el PO este enlazado con un rol de autoridad dentro de la empresa (CTO, CIO , etc.).

	Dentro del equipo se analiza el trabajo para identificar las dependencias de procesos. El trabajo es estructurado para optimizar la coordinación entre equipos y mejorar el rendimiento en el trabajo interdependiente.	Workgroup Development	Dependiendo de la estructura de la organización se recomienda: - Mantener los ítems del "Enterprise Product Backlog"(EPB), visibles para toda la organización. (ideal para todo tipo de empresas, en especial para las que trabajan con múltiples equipos de Scrum, o realizando Scrum de Scrum). - A los "Product BackLogs" de los distintos Scrum Teams, adicionar una columna de referencia para los ítems interdependientes. (Ideal para Scrum Teams que tengan ítems interdependientes, ST que trabajen con otros tipos de equipos que no apliquen Scrum). - Mantener actualizados y visibles los " Enterprise Burndowns Charts" (correspondiente al EPB) a toda la empresa para poder consultar el estado de estos. - Si existen múltiples equipos aplicando Scrum en la organización, el o los SM deben identificar procesos comunes y entre los ST para aprovechar las bases de conocimientos.
Sprint Planning Meeting	Las actividades de desarrollo individual son seguidas y enlazadas con el plan de desarrollo personal.	Career Development	Los elementos del “Sprint Backlog“, deben ser asignados al personal que no sea experto pero que presente competencias afines a la tarea de acuerdo al artefacto realizado en la “Release Planning Meeting“. La potencia de aplicar Scrum se basa en que los equipos sean multifuncionales. - Identificar posibles áreas en la cual se puedan certificar los conocimientos del personal. como incentivo al crecimiento.
	La asignación de trabajo está diseñada para mejorar el desarrollo de los objetivos profesionales y personales.	Competency Based - Practices	
	Los equipos y las personas participan en las decisiones que incumben a su ambiente de trabajo.	Participation Culture.	Definir mecanismos para la solicitud y atención de los requerimientos del ST, SM y PO (formularios, plantillas de correo, etc.) específicos para cumplir los objetivos del sprint, hacia la parte administrativa de la empresa, involucrar a esta con el proceso propuesto por Scrum.
	Cuando se requiere, se utilizan los resultados cuantitativos del rendimiento obtenidos.	Quantitative Performance Management	- Analizar la (BCF), para identificar funcionalidades similares de anteriores Sprint's. - Analizar los "Product BackLogs" y "Sprint Burdown Charts" de otros equipos para identificar funcionalidades similares .
	Se identifican oportunidades para utilizar la experiencia del personal, para mejorar el rendimiento o alcanzar objetivos organizacionales.	Mentoring	Identificar los Ítems del Sprint Product BackLog que impliquen un esfuerzo adicional por el ST (nuevas tecnologías, alta complejidad), y de acuerdo a su análisis: - Solicitar sesiones formativas para miembros del ST. - Establecer mecanismos de comunicación avanzados para los casos que puedan ser solucionados de forma adhoc entre el personal de ST. - Elaborar un plan de mentoring durante el inicio del sprint.
Sprint	Las personas capacitadas en una competencia, fungen como mentores a aquellas con menos capacidad en esa competencia. La organización apoya a la comunicación entre las unidades que comparten una	Competency Development	Al recibir una solicitud de formación la organización o de colaboración entre miembros del ST se debe: - Consultar la (BCF) para poder identificar posibles temas relacionados y formadores capacitados. - Designar recursos necesarios para emular la solución o impartir la formación.

	competencia.		
	El ambiente soporta el trabajo de las personas o grupos que integran procesos basados en competencias. Las actividades y prácticas laborales están definidas y ajustadas para apoyar la integración de las actividades basadas en competencias.	Competency Integration	Adecuar el ambiente laboral de acuerdo al objetivo del sprint. Consultar el "Enterprise Product Backlog" para la integración de procesos afines entre los otros equipos de la organización, ya sea que estén utilizando Scrum o no. En caso de precisar personal ajeno al ST, solicitarlo su presencia en el Daily Scrum para atender los impedimentos surgidos y luego de esta reunión colaborar si se diera el caso.
	Se delega a los grupos de trabajo la responsabilidad y la autoridad para determinar métodos con los cuales podrán lograr el trabajo comprometido.	Empowered Workgroups	Una vez realizada la sprint planning meeting, el ST es el único responsable de producir el incremento de funcionalidad. Se recomienda. - Permitir que el ST autogestiones sus tareas, y como llevarlas a cabo. - Facilitar el acceso a las herramientas y artefactos. - No introducir cambios en el sprint. - No migrar personas en la sprint. - Realizar siempre el Daily Scrum. Documentar y reportar impedimentos al SM de cualquier índole. En caso de que el ST note que los ítems del "Sprint PBL " no pueden ser cumplidos (cualquiera sea el motivo), debe cancelarse el sprint con el visto bueno del SM y el PO.
	El ambiente de trabajo de la organización apoya el desarrollo y rendimiento de los grupos de trabajo.		El ST, SM y PO deben estar en un mismo ambiente de preferencia. Caso contrario proveer los recursos necesarios para la comunicación (audio, video, web meetings, escritorios remotos)
	Las personas y equipos utilizan los activos basados en competencias para realizar las actividades de negocio.	Competency-Bassed Assets	Se consulta la BCF, los PBL de distintos equipos o el EPB, para identificar funcionalidades o tareas documentadas que puedan servir de guía, o bien de base para el desarrollo de nuevas tareas similares a las encontradas.
	Se actualizan la descripción de las competencias dentro de un periodo de tiempo y orientada a eventos.	Career Development	El plan de desarrollo personal debe ser actualizado por cada miembro, este plan debe contener la información mínima necesaria para realizar un seguimiento adecuado y significativo: - Recursos empleados en lograr la capacidad. (Esfuerzo, Tiempo, Coste, etc.) - Complejidad de la capacidad. Ej. Si se han utilizado User Histories como técnica de especificación de las capacidades a lograr por la persona, realizar los respectivos History Points.
	Cada individuo documenta sus objetivos de rendimiento para desarrollar capacidades adicionales a las competencias laborales de la organización	Competency Based – Practices	
	Se comunican cada uno de los programas de mentoring para las personas y grupos. Los mentores asisten a personas o grupos a desarrollar las capacidades en las competencias laborales.	Mentoring	- Los programas de mentoring diseñados para la sprint, deben estar accesibles para otros ST que puedan acceder a estos en caso de ser necesario. - Ejecutar las formaciones al principio de la sprint, y de preferencia luego del Daily Scrum, o al final de la jornada laboral.

	Los mentores ayudan al desarrollo y mejora de los activos basados en competencias.		- Realizar un seguimiento al personal formado e identificar y actualizar la BCF con los posibles nuevos formadores.
Sprint Review	Las personas y equipos capturan información y artefactos resultantes de la realización de los procesos basados en competencias.	Competency Based Assets	Actualizar la Base de Conocimientos Funcional, el Plan de desarrollo personal y los Productos Backlogs de los ST.
	Se realiza un seguimiento al rendimiento de la organización en el cumplimiento de objetivos estratégicos.	Workforce Planning	
	Se obtienen registros cuantitativos del rendimiento de las personas o grupos	Quantitative Performance Management	
	Se publican la información organizacional y de rendimiento, para los equipos y personas.	Participation Culture	Artefactos de interés organizacional y propios de los Scrum teams tienen que estar publicados a toda la organización.(PBL's, Plan de desarrollo Personal, BCF, Burdown Charts, etc.). No existe razón para que no sean públicos puesto que estos artefactos muestran el **trabajo a realizar dentro de la organización (ya sea a nivel de producción o implantación de Scrum)**, y no contienen procesos de negocio internos, ni información confidencial de la organización.
Sprint Retrospective	Se evalúan y gestionan cuantitativamente el impacto de las prácticas y actividades laborales en la capacidad y rendimiento de las competencias basadas en procesos.	Organizational Capability Management	Según los resultados arrojados por el EPB, se obtiene la retroalimentación para evaluar el estado de: - El cumplimiento de los objetivos de negocio de la organización, - El estado de la aplicación de Scrum como metodología de desarrollo.
	El rendimiento de los procesos basados en competencias integrados es evaluado para identificar necesidades de ajustes y actualizaciones.	Organizational Capability Management	

Tabla 3. Recomendación de Practicas de P-CMM para la mejora de Scrum

Área de Proceso	Prácticas	Propuesta de aplicación
Continuous Capability Improvement	Las personas continuamente mejoran su capacidad y rendimiento en sus procesos laborales personales.	Aplicar prácticas de mejora personal basadas en algún proceso de mejora (PSP, SPI, Six Sigma).
	Los grupos evalúan la capacidad y rendimiento de sus procesos para identificar oportunidades de mejora. Las prácticas laborales de la organización, son ajustadas de acuerdo a lograr una mejora continua de las actividades personales y grupales.	Definir un análisis estadístico sobre el rendimiento del ST en base a tiempos, calidad, cantidad de Sprint Backlog Ítems realizados.

Organizational Performance Alignment	Las unidades alinean el rendimiento entre las personas, grupos y otras entidades dentro de la unidad.	El SM debe Mantener actualizados, detallados y visibles los artefactos que propone Scrum para toda la organización.
	La evaluación del impacto de las actividades y prácticas laborales en la alineación del rendimiento es utilizada en la realización de otras actividades de negocio.	Definir y evaluar estadísticas sobre los conflictos entre el ST con relación al rendimiento y logro de objetivos, y la solución a estos. Identificar problemas de contradicción entre procesos e incompatibilidades con el objetivo de negocio.
Continuos Workforce Innovation	Se establece un "framework" para la mejora continua de las practicas y actividades del personal.	Según lo recomendado tanto en adopción como en la mejora del proceso, es necesario institucionalizar ciertas prácticas, paralelas a las que afectan al desarrollo de software, entre estas tenemos - Mapeo de las actividades con el objetivo de negocio de la empresa. - Desarrollo personal, grupal y organizacional en Scrum. - Evaluación de la mejora (si es que se presenta) con la implantación de Scrum en la organización.
	Se evalúan y seleccionan nuevas prácticas laborales y tecnológicas y para su implementación.	- Utilizar alguna herramienta ALM orientada al proceso que define Scrum (TFS, TargetProcces, VersionOne, etc.). - Utilizar una herramienta de gestión documental (MOSS, Documentum , etc.) para los artefactos recomendados en este artículo. (BCF, Plan de desarrollo personal).

Tabla 4. Recomendaciones de prácticas ajenas a Scrum

4. Resultados esperados

Al adoptar Scrum en los primeros meses se notarán muchas falencias, y no sería raro tratar de adecuar la metodología a los procesos antiguos de la organización, sin embargo esto no es aconsejable por los autores, puesto que al modificar los procesos no se garantiza la efectividad esperada. Con la introducción las prácticas mencionadas en este artículo, se espera colaborar a las organizaciones en la adopción de Scrum y también en la mejora de su implantación. Al institucionalizar el proceso la organización cuenta con una experiencia documentada para futuras adopciones en otros equipos. También mejorar la capacidad de las personas que se encuentran aplicando la metodología.

Con respecto a las personas, se espera generar un ambiente colaborativo, donde la transparencia será un pilar fundamental entre los roles de Scrum y la organización. Al conseguir esto, la desviación en las estimaciones será menor, se agilizara el trabajo dependiente entre equipos, y se conseguirá un mejor ambiente laboral.

Al ser una metodología ágil y de sencilla aplicación los integrantes suelen obviar aspectos importantes de su rol, u obviar algunos “timebox” (“Release Planning”,””Daily Scrum”, “Sprint Retrospective”), los cuales sirven sobre todo para la proyección y retroalimentación del trabajo comprometido y la aplicación del proceso.

Otro factor importantes es clarificar y facilitar las tareas del “Product Owner” y del Scrum Master”, pues dependiendo del tamaño del equipo y de la magnitud de la empresa muchas veces suelen ser miembros del “Team”, y su carga de trabajo les lleva a confundir los roles.

Por último esperamos ayudar a crear un mecanismo de medición de procesos y actividades realizadas, con el fin de proponer nuevas prácticas de acuerdo a diferentes entornos de aplicación según las experiencias documentadas del proceso.

5. Conclusiones y trabajo futuro

PCMM presenta una serie de prácticas que ayudan a lograr los objetivos de las diferentes áreas de proceso que definen el nivel de madurez de gestión del personal en una organización. Las actividades que proponen dotan de solidez a la gestión del capital humano y recursos que se relacionan con este en la organización.

Si bien es cierto es una iniciativa basada en experiencias de software, pero su aplicación no es estrictamente orientada a este rubro. Al contrario, son procesos de negocio organizacionales genéricos. A su vez la aplicación de PCMM en organizaciones pequeñas y medianas no es muy común debido a lo extenso y complejo del marco de trabajo, la necesidad de expertos externos, o el tiempo a invertir en auto preparación.

Por otra parte Scrum es una metodología ágil de fácil aplicación en las organizaciones, escalable y comprensible. Su adopción no está condicionada por el tamaño de la organización ni los recursos que esta posea. Sin embargo, al ser sencillo y otorgar al equipo el poder de auto gestionarse, suelen ocurrir desviaciones en su aplicación, debido a la inexperiencia de los participantes, a las falencias de los procesos de negocio heredados de la organización y otros factores culturales, ambientales y/o temporales. Así también es bien sabido que una debilidad de las metodologías ágiles en especial de Scrum es su bajo enfoque en el aseguramiento de la calidad [5].

Luego de realizar un análisis de ambas tecnologías, y de acuerdo a algunos casos de éxito y experiencias recogidas [11], proponemos realizar una intersección entre algunas prácticas presentadas por PCMM en los "timebox" de Scrum, con el objetivo de fortalecer las falencias identificadas en la adopción de Scrum por parte de personas inexpertas, sin modificar su estructura y colaborar con quienes deseen progresar en la mejora del proceso.

Luego de la presente propuesta para alcanzar un nivel aceptable y predecible en cuanto al desarrollo de software aplicando Scrum, se estudiará a futuro implantar algún modelo especializado en mejora de procesos como Six Sigma, SPI entro otros.

Así también se espera llevar este estudio a otras áreas de interés de los autores, como el desarrollo de software en entornos distribuidos, mejora de procesos, gestión del ciclo de vida de las aplicaciones (ALM) entre otros.

Referencias

[1] Nauer, P. & Randall, B. (1969). Software Engineering NATO Scientific Affairs Division, Brussels.

[2] Humphrey, W.S. (1999). "The Team Software Process (TSP)", *Software Engineering Institute*, CMU/SEI-2000-TR-023, ESC-TR-2000-023.

[3] Beck, K. et. al (2001). *Manifesto for agile Software Development*, http://agilemanifesto.org/

[4] Pressman, R.S. (2005). *Software Engineering: A practitioner´s approach* (6th Edition). McGraw Hill.

[5] Timperi, O. (2004). An Overview of Quality Assurance Practices in Agile Methodologies. SoberIT, T-76.650 Seminar in software engineering, Finland.

[6] Curtis, B. Hefley, W.E. & Miller, S.A. (2001). "People Capability Maturity Model. Version 2.0 2nd Edition", *Software Engineering Institute*, CMU/SEI-2009-TR-003.

[7] Schwaber, K. & Sutherland, J. (1993). *The Scrum Guide (Feb. 2010 edition),* http://www.scrum.org/scrumguides

[8] Deemer, P. Benefield, G. Larman, C. & Vodde, B. (2009). *The Scrum Primer Version 1.1*

[9] Schwaber, K. (2004). *Agile project management with Scrum*, Microsoft Press (Redmond, DC).
[10] Schwaber, K. & Sutherland, J. (2011). *The Scrum Guide (Oct. 2011 edition),* http://www.scrum.org/scrumguides
[11] Schwaber, K. (2007). *The Enterprise and Scrum*. Microsoft Press (Redmond, DC).
[12] Curtis, B., Hefley, W.E. & Miller, S.A. (1995). Overview of the People Capability Maturity Model, Version 1.1. *Software Engineering Institute*, CMU/SEI-95-MM-001
[13] Ngwenyama, O. & Nielsen P. (2003). Competing Values in Software Process Improvement: An Assumption Analysis of CMM from an Organizational Culture Perspective*. IEEE Transactions on Engineering Management* 50(1) 100 -112.
[14] Colomo, R., Tovar E., Gomez J., Crespo A. (2006). Recomendaciones para el desarrollo del Capital Humano desde la Perspectiva de la Mejora del Proceso Software. *RPM-AEMES* 3(3).
[15] Bos, E. & Vriens, C. (2004). An agile CMM. C. Zannier et al. (Eds*.): XP/Agile Universe*, pp. 129–138.
[16] Srinivasa, G. & Ganesan, P. (2002). Pair Programming: Addressing Key Process Areas of the People-CMM. *Lecture Notes in Computer Science* (2418/2002), 115-120.
[17] Marçal A., de Freitas B., Furtado F., Furtado M., Maciel T. & Belchior A. (2008). Blending Scrum practices and CMMI project management process areas. *Innovations in Systems and Software Engineering* 4(1) 17–29.
[18] Caballero, E., Calvo-Manzano, J. A. & San Feliu, T. (2011). Introducing Scrum in a Very Small Enterprise: A Productivity and Quality Analysis. *SPI in Small and Medium Enterprises (SMEs) EuroSPI 2011*, Denmark.

Un modelo de penalización para suministro de servicios

Carmen Cobo Rivas

Sopra Group

Madrid - España

ccobo@sopragroup.com

Abstract: *The SLA's penalty models are often complex and imprecise. A mathematical model based on a logistic curve of growth is described in this paper to overcome these drawbacks. The function offers other advantages, among which the refinement of calculation.*

Resumen: *Los modelos de penalización para acuerdos de niveles de servicio adolecen con frecuencia de complejidad e imprecisión. Se describe en este artículo un modelo matemático basado en una curva logística de crecimiento, mediante el que se intenta superar estos inconvenientes. La función ofrece otras ventajas, entre las que destaca la finura de cálculo.*

Keywords: *Acuerdos de Niveles de Servicio, Penalizaciones, Curva Logística.*

1. Modelos de penalización

Pese a que las cláusulas de penalización deberían (idealmente) ser aplicadas muy raras veces [1], la verdad es que la sanción por servicios suministrados con niveles inferiores a los acordados es una forma de resarcir los daños causados al cliente y constituye un eficaz medio de control y un incentivo para que el proveedor proporcione los servicios según los niveles acordados [2].

El modelo de penalización es el método para calcular e imponer las sanciones. Debe incluir qué, cómo y cuánto ha de penalizarse.

En lo que respecta al objeto de la sanción, habrá que considerar los niveles de servicio acordados contractualmente, por ejemplo, en un servicio de atención telefónica, el número de llamadas respondidas antes de 10 segundos.

La forma de penalizar se basará en asociar un importe de sanción al desvío presentado por el servicio con respecto a un nivel mínimo acordado. Siguiendo con el ejemplo anterior, una forma de penalizar podría consistir en aplicar, por cada llamada respondida en más de 10 segundos, un porcentaje de reducción sobre la factura del servicio. Es de gran importancia el saber fijar en su justa medida la sanción, según la importancia del servicio: ni tan pequeña que el proveedor la considere como un coste aceptable, ni tan grande que le dificulte la prestación [2].

La cuantía de la sanción estará limitada por un tope máximo que, normalmente, vendrá expresado como un porcentaje sobre la facturación del servicio en un período de tiempo.

Los modelos de penalización suelen adolecer de varios defectos, entre los que se encuentran la innecesaria complejidad y la imprecisión.

La falta de simplicidad suele presentarse cuando se desea reflejar la diferente importancia de los servicios de forma poco homogénea, bien aplicando diferentes métodos para penalizar distintos servicios, bien introduciendo múltiples condiciones. Un modelo complejo de penalización suele estar asociado a un modelo complejo e ineficaz de acuerdos de niveles de servicio, que va contra el ideal de medición fina y precisa [3]. Un modelo complejo dificulta la aplicación de sanciones y dar lugar a conflictos sobre su interpretación.

La imprecisión aparece habitualmente por desatender el hecho de que un mayor alejamiento de los niveles acordados no sólo debe llevar aparejada una mayor sanción, sino también una sanción más severa.

Mediante el modelo que se desarrolla en la siguiente sección, basado en una función de crecimiento logístico, utilizada por Pierre François Verhulst [4] como modelo del crecimiento de poblaciones, se intenta superar los inconvenientes presentados.

2. Modelo propuesto

Se hará corresponder a cada servicio un nivel acordado mínimo que, si no se alcanzase, dará lugar a una sanción. Se medirá el porcentaje de desviación sobre dicho nivel mínimo, lo cual resultará en un número de puntos de penalización. El cálculo de los puntos a partir del porcentaje se llevará a cabo mediante la función de crecimiento *P(x),* que se presenta más adelante.

En función de los puntos acumulados para todos los servicios se aplicará una penalización sobre un porcentaje del importe a facturar por los mismos. Por ejemplo, si la factura por los servicios ascendiera a 1000, los puntos acumulados de penalización fueran 40, y el porcentaje del importe a facturar se situase en 15%, se aplicaría un (40/100) * 15 por ciento sobre la factura, es decir [(40/100)*15]*1000 /100 = 60. En general, si T representa los puntos acumulados, F el importe de la factura, S la cuantía de la sanción y *p* el porcentaje sobre el importe a facturar, tendremos

$$S = \frac{TFp}{10000}$$

Los puntos de penalización se calcularán mediante la siguiente función

$$P(x) = \frac{L}{1 + Ae^{-kx}}$$

x: desvío porcentual con respecto al mínimo nivel de servicio de un indicador.

k: multiplicador de severidad

A: parámetro relacionado con la importancia de servicio. Se presentan posibles valores:

A= 20 para servicios críticos.

A= 50 para servicios importantes.

A=100 para servicios ordinarios.

L es el límite de puntos para el servicio, es decir un valor que no podrá nunca alcanzarse con la función -por ser su asíntota horizontal superior-, sea cual fuere el desvío sobre el nivel mínimo. En efecto $\lim_{x\to\infty} P(x) = L$

Ejemplo: dado un servicio crítico, con nivel de servicio mínimo 95%, con límite de puntos igual a 30 y multiplicador de severidad igual a 0,1, para un nivel de servicio del 85% el desvío porcentual sería 11, y los puntos de penalización 4.

$$P(11) = \frac{30}{1+20e^{-0,1x11}} \approx 4$$

Para el mismo servicio, un nivel de 60% supondría un desvío porcentual de 37. Los puntos de penalización por dicho desvío se calcularían como sigue:

$$P(37) = \frac{30}{1+20e^{-0,1x37}} \approx 20$$

La gráfica que sigue presenta, con los parámetros del ejemplo, los puntos de penalización en función del desvío.

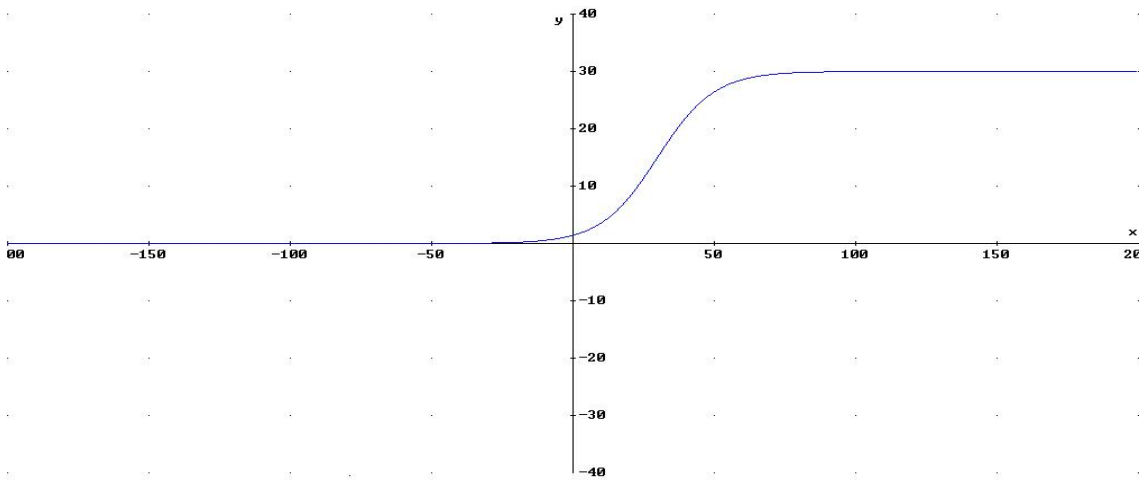

Figura 1. Gráfica de la función de penalización

3. El significado del multiplicador de severidad

El efecto de la variación de los valores de k sobre la curva consiste en una variación de su inclinación. Cuanto mayor es el multiplicador más pendiente presenta la gráfica. Esto puede apreciarse con claridad mediante la función derivada

$$P'(x) = \frac{AkLe^{xk}}{\left(e^{xk} + A\right)^2}$$

La siguiente gráfica muestran la derivada para los valores de k: 0,1 y 0,25, con los parámetros de P, ya vistos. Cabe destacar la gran sensibilidad de la pendiente a pequeñas variaciones del parámetro.

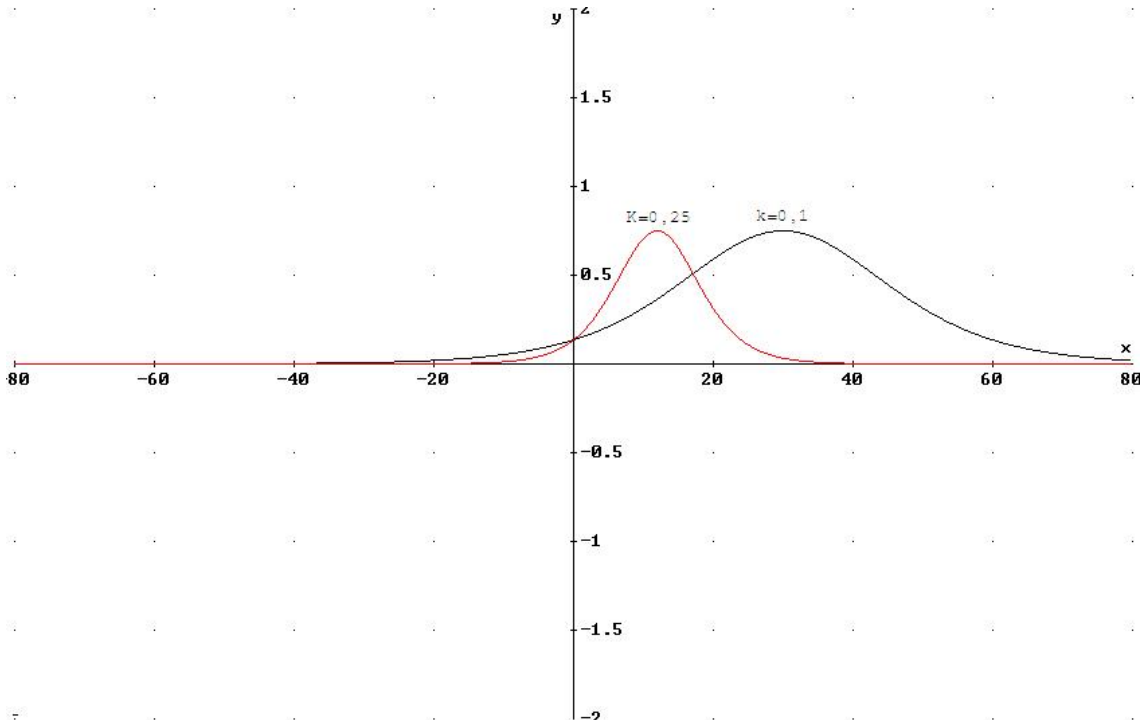

Figura 2. Pendientes de la curva de penalización

Nótese ahora que la segunda derivada, tal como aparece en la figura 3, nos indica que la aceleración de la penalización presenta dos tramos: creciente en la primera mitad de la parte cóncava de P y decreciente en la segunda parte. Esto significa que las desviaciones importantes, cuando comienzan a presentarse, son penalizadas con ganancia continua de aceleración, la cual se ralentiza hasta llegar a cero en el punto de inflexión de P.

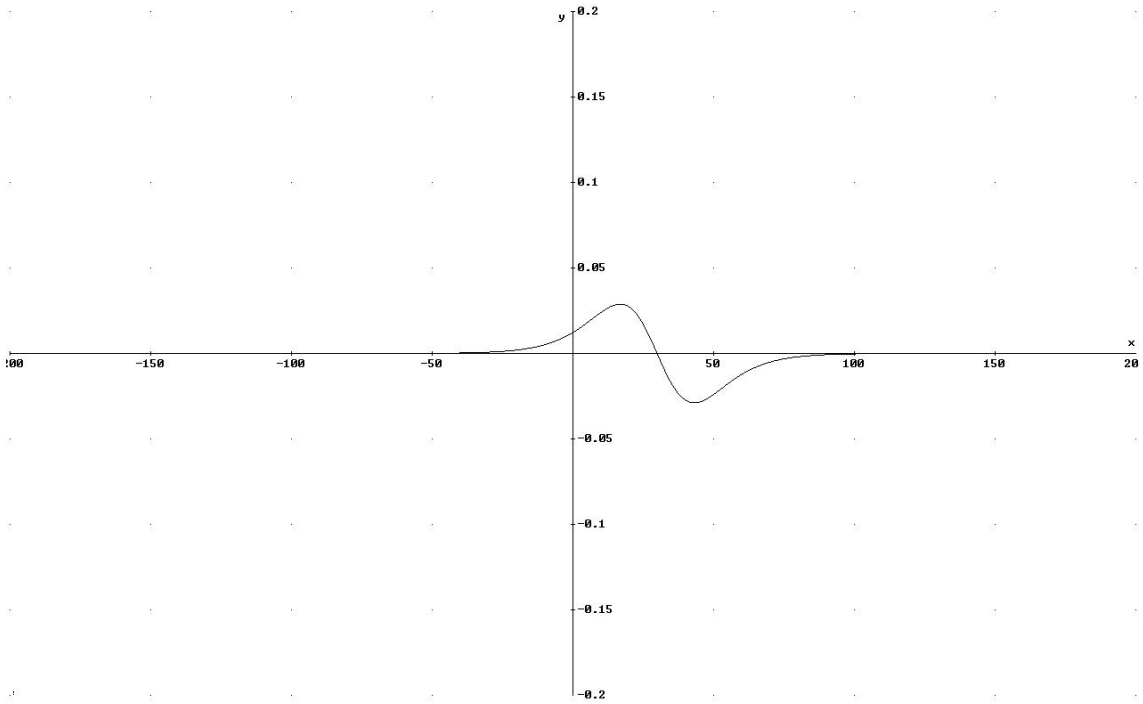

Figura 3. Aceleración de la penalización

4. El significado del parámetro A

El efecto del parámetro A es un desplazamiento de la curva en abscisas, sin variación de forma. Se puede utilizar para fijar en qué punto del desvío con respecto al mínimo nivel de servicio comienza a aplicarse una penalización fuertemente acelerada: cuanto más elevado el valor, más suave es la penalización en tramos inferiores de desvío.

Haciendo $Ae^{-k(x-c)} = Be^{-k(x)}$ podemos situar la curva *c* unidades a la derecha de la curva con parámetro A. B, el nuevo parámetro de importancia en la curva desplazada, será, como es fácil de comprobar $B = e^{\ln A+kc}$

A continuación se presenta una curva desplazada c=5 unidades a la derecha de nuestra gráfica original, con $B \approx 32{,}9744$

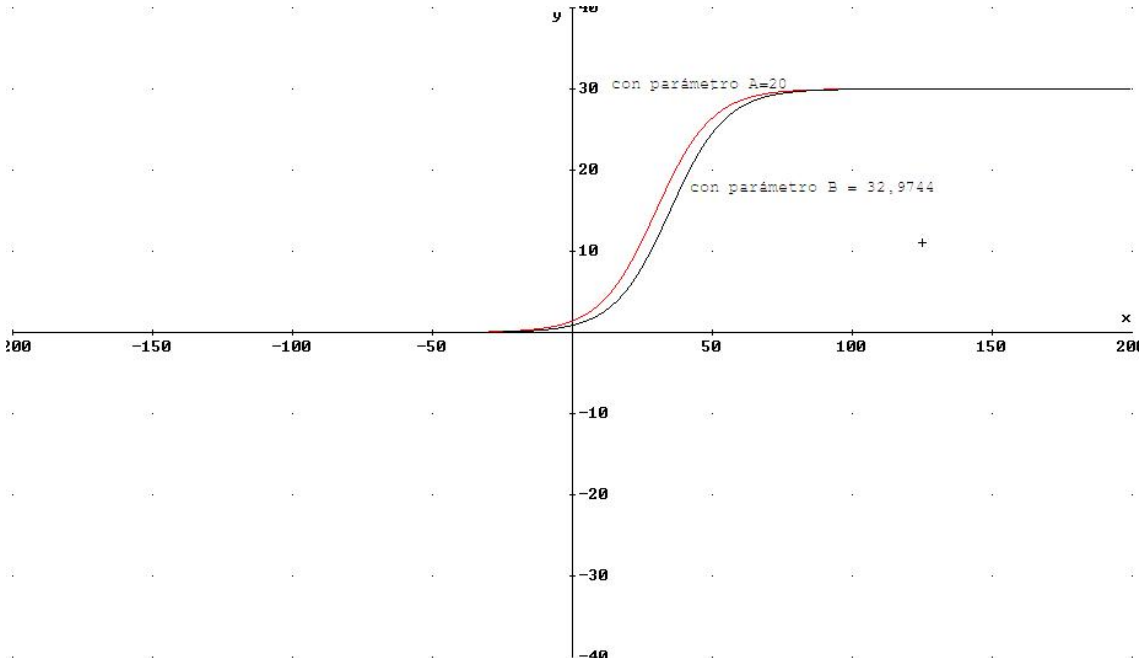

Figura 4. Variación en el parámetro A

5. Corrección en el origen

La curva presenta un punto de corte con el eje de ordenadas con ordenada positiva, lo cual significa que, sin haber desvío, existe una pequeña penalización. Esta puede ignorarse. Alternativamente puede corregirse la curva, de manera que pase por el origen de coordenadas. Para ello bastara transformar

$$f(x) = P(x) - P(0)$$, es decir $$f(x) = P(x) - \frac{L}{1+A}$$

6. Conclusiones

La sanción se obtiene mediante una fórmula, lo cual evita complicadas tablas de penalización.

La función crece de forma acelerada, reflejando la mayor severidad aplicable a los desvíos importantes.

Se puede controlar el crecimiento mediante el parámetro *k* de la fórmula: cuanto mayor sea su valor, más rápidamente crecerá la función; es decir, cuanta mayor importancia se otorgue al servicio, más puntos de sanción resultarán de pequeños desvíos. Se puede controlar asimismo, mediante el parámetro A, la cuantía del desvío a partir de la que comienza a aplicarse una fuerte penalización.

La cuantía máxima de la sanción, para un determinado servicio, depende de un límite establecido, lo cual permite asignar límites altos a servicios críticos y límites bajos a servicios ordinarios.

Referencias

[1] Minoli, D. (1995). *Analyzing Outsourcing: Reengineering Information and Communication Systems.* McGraw-Hill.
[2] Sturm, R. & Morris, W. (2000). *Foundations of Service Level Management*, SAMS.
[3] Govekar, M. (2009). Under Pressure: Evolving IT Operations to Best Deliver Business Value, Gartner Symposium, ITxpo 2009, France.
[4] Verhulst, P.F. (1845). Mathematical Researches into the Law of Population Growth Increase. *Nouveaux Memoires de l'Academie Royale des Sciences et Belles-Lettres de Bruxelles*, 18(1) 1-45.

NDT-Suite, una solución práctica para el uso de NDT

Julián Alberto García García, Daniel Rivero Capellán, María José Escalona Cuaresma, Isabel Ramos Román

Grupo IWT2. Departamento de Lenguajes y Sistemas Informáticos

Universidad de Sevilla

Sevilla - España

{julian.garcia, daniel.rivero}@iwt2.org, mjescalona@us.es, isabel.ramos@lsi.us.es

Abstract: *The importance of software engineering is well established and accepted by the research community. However, it is still common to find reticence in the business world against the discipline. There are plenty of proposals that call for agile methods for developing software. These methods propose very timely measures for the business environment but yet have not been used in large real projects. One of the most important aspects that lead to this situation is the need to implant these methodologies supported by development tools and environments that must be profitable for the business world. This paper presents a proposal framed within the paradigm of model-driven Web Engineering. This proposal has been adapted to provide a methodological environment currently is being used successfully in real environments.*

Resumen*: La importancia de la Ingeniería del Software está ampliamente demostrada y aceptada por la comunidad investigadora. Sin embargo, todavía es frecuente encontrarse reticencias en el mundo empresarial. Hay bastante propuestas que abogan por métodos ágiles y adecuados para el desarrollo de software que proponen medidas muy oportunas para el entorno empresarial pero, que sin embargo, no han sido usadas en grandes proyectos reales. Uno de los aspectos más importantes que llevan a esta situación es la necesidad de acompañar a esas propuestas de herramientas y entornos de desarrollo que realmente sean rentables para el mundo empresarial. Este artículo presenta cómo una propuesta enmarcada dentro del paradigma de Ingeniería Web guiada por modelos se ha adaptado para ofrecer un entorno metodológico que, actualmente, está siendo utilizado de forma satisfactoria en entornos reales.*

Keywords: *Model Driven Web Engineering, Web Requirements, Tools, Practical Experience, NDT*

1. Introducción

A lo largo del tiempo las metodologías web basadas en modelos han ido tomando una relevancia cada vez más importante en la Ingeniería Web, como resultado de esta evolución han surgido distintas propuestas metodológicas que han tratado y tratan de dar respuesta a las dificultades que ofrece el diseño de sistemas hipermedia y aplicaciones web. El éxito o fracaso de estas metodologías se debe a su capacidad para afrontar la realidad del desarrollo de software para la red.

A diferencia de la Ingeniería del Software, la Ingeniería Web ofrece otros retos como son:

Usuarios finales desconocidos. En el desarrollo de aplicaciones web no se debe asumir un rol de usuarios estándar ya que la naturaleza de la web es multicultural además de que cada usuario cuenta con capacidades y tecnología distintas.

Alta disponibilidad y requisitos cambiantes. La propia naturaleza dinámica de la web obliga a que las aplicaciones estén operativas la mayor parte del tiempo y que los cambios en los requisitos o el mantenimiento no supongan un cese de las actividades del usuario.

Ausencia de orientación. De forma general la web es un elemento textual en el que los únicos elementos de navegación son los hipervínculos. Debido a la gran flexibilidad de este mecanismo no es difícil diseñar aplicaciones en las que el usuario se sienta perdido por inconsistencias en la presentación o falta de contexto navegacional.

Por estas razones son necesarias metodologías que se centre en el usuario y que ofrezcan modelos para la fase de requisitos, modelos para el tratamiento de la navegación y modelos de interfaz.

Junto a los problemas a los que dan respuesta las metodologías actuales cabe tener en cuenta la importancia de que estas metodologías estén soportadas por herramientas. Este soporte facilita que los equipos de trabajo se centren en el problema de su dominio de negocio en lugar de preocuparse por los detalles de funcionamiento de la metodología subyacente. Igualmente el soporte de herramientas permite que los usuarios finales puedan validar los modelos generados si las herramientas proporcionan la adaptación de la información de modo que el usuario final no tenga por qué conocer los detalles metodológicos o técnicos del desarrollo para poder validar el trabajo realizado.

El presente artículo pretende ilustrar las herramientas de soporte de la metodología NDT para dar respuesta a las dificultades de desarrollo web junto a un ejemplo de uso y las experiencias derivadas de los proyectos reales realizados con NDT.

2. Trabajos Relacionados

NDT (Navigational Development Techniques) no es la única propuesta existente para la ingeniería basada en modelos para la web. Entre las metodologías más relevantes y que más han influido en NDT se encuentran OOHDM (Object-Oriented Hypermedia Design Method) [2], UWE (UML-Web Engineering) [3] y WebML (Web Modelling Language) [4], todas ellas tienen en común que los metamodelos y modelos están basados en UML y en los principios de orientación a objetos.

OOHDM supuso el punto de partida para las demás metodologías, esta se basa en HDM pero incluye un enfoque orientado a objetos en sus metamodelos. A su vez esta metodología ofrece tres modelos fundamentales que obligan a la separación de conceptos: modelo conceptual, modelo de navegación y modelo de interfaz abstracta.

UWE es una metodología de Ingeniería Web guiada por modelos y sus principales características son: un profile en UML [5] para modelar las fases de desarrollo en particular el modelo de requisitos, modelo de contenidos, modelo de navegación y modelo de procesos. Y soporte tool UWEet (basado en la herramienta UMLet) y MagicUWE (basado en la herramienta MagicDraw).

WebML es la propuesta del departamento de electrónica e informática del politécnico de Milán. Esta propuesta está formada por varios modelos (WeBML) y por una herramienta que da soporte a los modelos de WebML llamada WebRatio. La principal característica de esta metodología es su herramienta CASE webRatio basada e integrada con las herramientas de desarrollo eclipse, con capacidades para facilitar el diseño de diagramas BPMN.

En la actualidad podría decirse que NDT compite en el mercado con UWE y WebML por lo que es influido por dichas metodologías y a su vez influye en ellas.

3. Visión general de NDT

NDT (Navigational Development Techniques) [1] es una propuesta metodológica incluida dentro del paradigma MDE (Model-Driven Engineering, ingeniería Guiada por Modelos) que se definió inicialmente para cubrir las necesidades en el desarrollo Web.

NDT comienza con la definición de metamodelos formales para fase de requisitos y la definición de un conjunto de transformaciones definidas en QVT [12] para generar los modelos de fase Análisis. La etapa de ingeniería de requisitos comienza con la definición de los objetivos del sistema. En este sentido, NDT puede entenderse como una metodología guiada por objetivos, según la clasificación en [10].

Una vez definidos los objetivos, NDT propone ir capturando y definiendo los requisitos del sistema y propone dividirlos en diferentes tipos, de manera que puedan tratarse según su tipología. De esta forma, los requisitos se dividen en: requisitos de almacenamiento de información, requisitos de actores, requisitos funcionales, requisitos de interacción y requisitos no funcionales.

La división en tipologías de requisitos sigue la línea que otras propuestas han tenido en el entorno de Ingeniería Web en otras fases, principalmente análisis y diseño, en las que se han separado los modelos para trabajar con cada uno de ellos. Así, por ejemplo, se han definido modelos conceptuales, de navegación, de adaptabilidad, etc. Este tratamiento caracterizado permite tratar de manera particular cada tipología de requisitos y separar los conceptos. De esta forma, cada uno de esos elementos se define de manera formal en NDT mediante un metamodelo en el que cada tipología de esos requisitos, así como sus atributos y las relaciones que se establecen entre ellos, son representados haciendo uso de un diagrama de clases.

En su siguiente fase, el análisis, NDT trabaja de una forma similar. Siguiendo la línea de otras metodologías ampliamente aceptadas como UWE o WebML, NDT propone representar un modelo de contenido, un modelo de navegación y un modelo de interfaz abstracta. Todos ellos, también se encuentran definidos en un conjunto de metamodelos.

La propuesta, analizando tanto los metamodelos de requisitos como los metamodelos de análisis, establece una serie de relaciones entre los artefactos que en ellos aparecen. En base a estas relaciones se definen un conjunto de transformaciones.

Así estaba planteada inicialmente la metodología NDT. Sin embargo, en los últimos años ha evolucionado y ofrece soporte completo para todo el ciclo de vida. Abarca las fases de estudio de viabilidad, requisitos, análisis, diseño, construcción, implementación, así como las fases de mantenimiento y de pruebas durante el desarrollo de software, y además establecen nuevas reglas de transformación. En la figura 1 se muestra cómo a partir de la fase de requisitos es posible obtener los modelos de la fase de análisis y de la fase de pruebas.

Las reglas de transformación de NDT están representadas en la figura 1 mediante el estereotipo *«NDTTransformations»*, y una vez que han sido aplicadas, el equipo de analistas puede pueden realizar transformaciones controladas con el objetivo de completar y enriquecer los modelos para obtener el modelo definitivo. Este paso no es automático y requiere la experiencia del analista. Estas transformaciones están representadas en la figura 1 mediante el estereotipo *«NDTSupport»*.

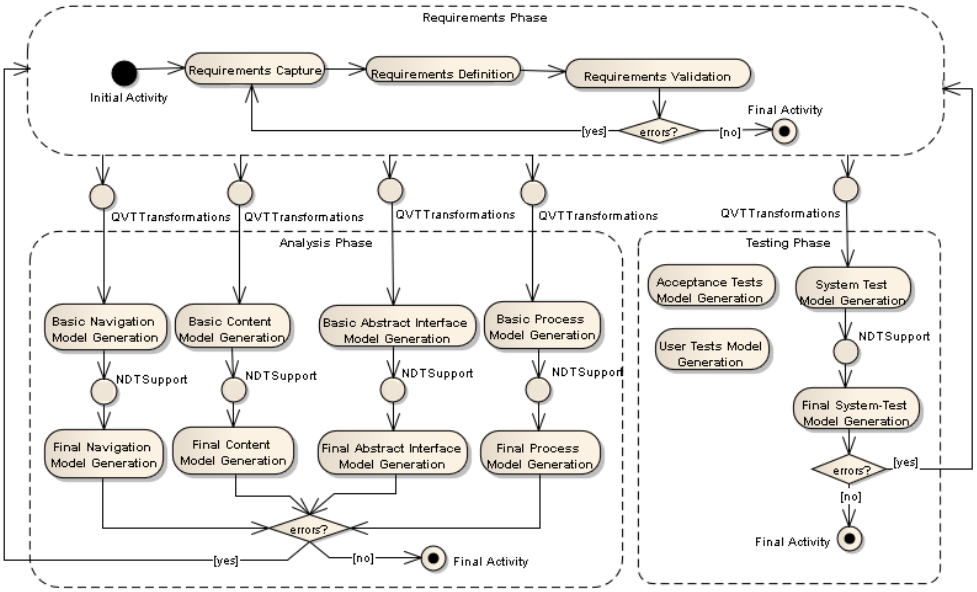

Figura 1. Generación de los modelos de análisis de pruebas desde los requisitos

Por otra parte, NDT es compatible con un conjunto de procesos para llevar a cabo la gestión de proyectos, seguridad y garantía de calidad. Este conjunto de procesos se definen en detalle en el trabajo NDTQ-Framework, presentado en la siguiente sección.

Una ventaja es que NDT se puede utilizar en el entorno empresarial de forma satisfactoria. Hoy en día, un elevado número de empresas en España trabajan con NDT en el desarrollo de software. Esto es posible debido al hecho de que NDT está totalmente apoyado por un conjunto de herramientas libres, agrupadas en NDT-Suite. En las siguientes secciones se detalla el conjunto de herramientas que constituyen esta suite.

4. NDT- Suite

Desde su comienzo, NDT ha sido una propuesta que ha tenido una amplia aplicación en el entorno empresarial. El feedback obtenido tras las aplicaciones prácticas, demostró que la primera herramienta diseñada para dar soporte a NDT, la herramienta NDT-Tool, no resultaba una solución óptima para el trabajo en proyectos reales de gran envergadura y heterogéneos debido principalmente a que NDT-Tool no era flexible; en [6] se justifica esta falta de flexibilidad de NDT-Tool. Por estos motivos, surge la necesidad de desarrollar la nueva suite de herramientas NDT: *NDT-Suite*.

Para desarrollar NDT-Suite lo primero que se hizo fue hacer una extensión de la propia metodología. Tomando las ideas de NDT y siguiendo las premisas marcadas por la metodología Métrica v3[1], UWE y OOHDM, se hizo una extensión para abordar todo el ciclo de vida. Esta extensión ha sido publicada en [7].

De esta forma, y aprovechando las premisas de estos entornos, suficientemente afianzados tanto en la Ingeniería del Software como la Ingeniería Web, y las ideas esenciales de NDT, se ha definido un entorno de trabajo compuesto por las fases de Requisitos, Análisis, Diseño, Construcción e Implantación, Pruebas y Mantenimiento. El hecho de que la elección haya sido Métrica se ha debido a que es la metodología de

[1] www.map.es

referencia en las Administraciones Públicas españolas. UWE fue seleccionado porque es una metodología Web fundamentada sobre UML, aspecto de gran interés debido a que es el lenguaje soportado por Métrica. Y OOHDM fue elegido por ser la primera metodología para la Web, cuyos modelos han sido adaptados y aprobados por toda la comunidad investigadora. La fusión realizada se basa en definir un proceso, similar al de Métrica pero haciendo uso de los modelos de UML y de las extensiones que NDT realiza de ellos, así como de sus procesos de ingeniería guiada por modelos.

A lo largo de este apartado se presentan todas las herramientas que actualmente componen NDT-Suite y otras que serán liberadas en breve.

4.1. NDT-Profile

El primer trabajo que dio inicio a NDT-Suite fue seleccionar un entorno de herramientas adecuado para trabajar. Tras diferentes estudios se optó por Enterprise Architect [8] (EA en adelante). Esta herramienta, a pesar de no ser de software libre, ofrece un soporte adecuado y su precio es bastante competitivo. Además, ofrece varias ventajas importantes como la posibilidad de definir profiles, herramientas para la gestión de documentación, etc. que han resultado de gran interés para nuestro trabajo.

La ampliación de la metodología NDT supuso la definición de nuevos metamodelos y transformaciones para todas las fases del ciclo de vida software. Para cada uno de los metamodelos de NDT se definió un profile en la herramienta EA. Este entorno es lo que se conoce como *NDT-Profile*. Así, por ejemplo, en la figura 2 se presenta una captura del entorno de trabajo en el que puede verse remarcado a la izquierda los distintos toolbox de NDT.

El uso de NDT-Profile ofrece la posibilidad de disponer de todos los artefactos de NDT de una manera sencilla, puesto que están integrados en la propia herramienta pero, además, también permite utilizar todos los modelos de UML e integrarlos fácilmente en la metodología.

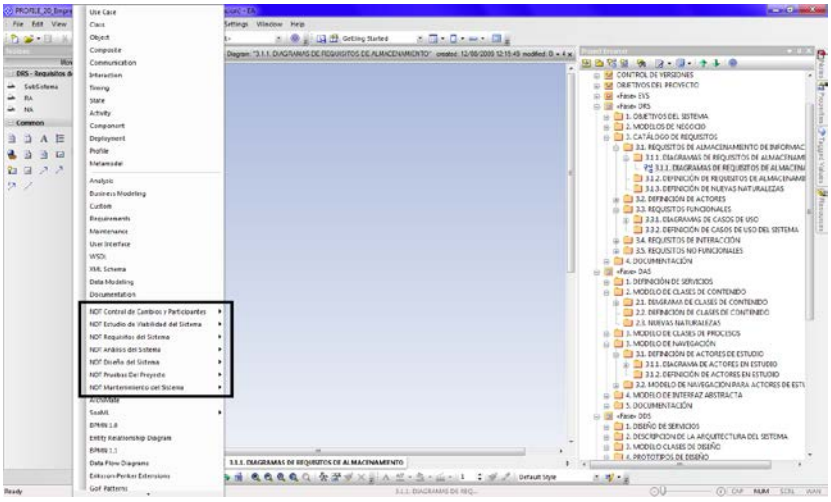

Figura 2. Interfaz principal de NDT-Profile. Toolbox y navegador del proyecto

4.2. NDT-Quality

El uso de NDT-Profile revela un problema importante y es la flexibilidad de trabajo que esta herramienta permite. Con NDT-Profile es posible infringir fácilmente las reglas y relaciones definidas en NDT, las cuales, son esenciales para trabajar con el entorno MDE de la propuesta. Por ello, se ha desarrollado otra herramienta dentro de NDT-Suite que se denomina NDT-Quality.

NDT-Quality, figura 3, es una herramienta que automatiza la revisión metodológica de un proyecto desarrollado con NDT-Profile. Se encarga de revisar tanto la calidad de un entregable en cuanto al uso de la metodología NDT en cada una de las fases del ciclo de vida software, como la trazabilidad de las reglas MDE que define NDT entre las distintas fases del ciclo de vida. Además, para facilitar el trabajo en proyectos que siguen ciclos de vida iterativos, la herramienta permite mediante una ventana de conformación seleccionar qué reglas concretas se desean verificar, figura 4.

Una vez realizada la revisión, la herramienta proporciona, para cada fase del ciclo de vida software revisada, un listado con los errores o inconsistencias metodológicas detectadas, figura 5. Estos errores se tipifican como leves, graves, o críticos. Entre los aspectos que son revisados por la herramienta se encuentran por ejemplo, completitud en definiciones, definiciones erróneas de restricciones, detección de ciclos en diagramas de clases, etc.

Para facilitar la creación de informes, la herramienta permite exportar en un documento en diferentes formatos el listado de inconsistencias encontradas durante la revisión.

Finalmente, como valor añadido, la herramienta no sólo se limita a resaltar los errores detectados durante la revisión, sino que también proporciona para cada uno de ellos una guía para resolverlo.

Como conclusión final, el uso de NDT-Quality permite garantizar la calidad de cualquier proyecto que siga la metodología NDT.

Figura 3. Ventana de configuración de NDT-Quality

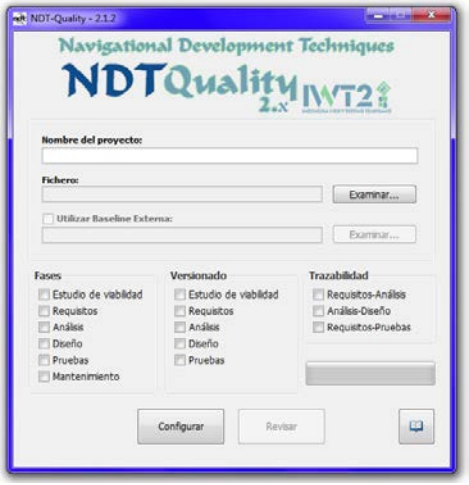

Figura 4. Ventana de NDT-Quality

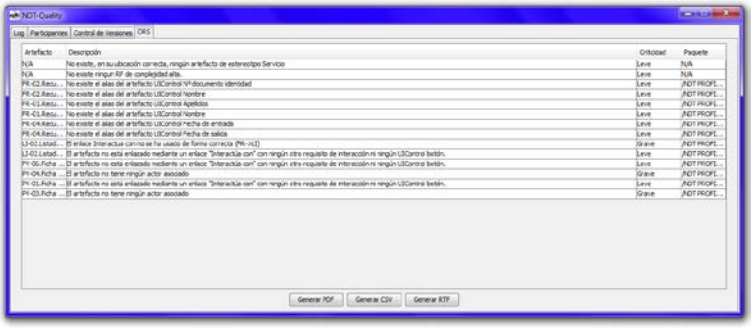

Figura 5. Ventana de informe de NDT-Quality

4.3. NDT-Driver

Inicialmente, NDT sólo definía la transformación del modelo de requisitos del proyecto en los modelos de la fase de análisis. A raíz de la ampliación de NDT al resto de fases del ciclo de vida, se tuvieron que definir nuevas transformaciones. Por este motivo, se hizo necesario incorporar esta nueva funcionalidad en la suite de NDT mediante una nueva herramienta: NDT-Driver.

NDT-Driver, figura 3, es una de las principales herramientas de soporte de la metodología NDT. Implementa un conjunto de procedimientos automáticos para llevar a cabo cada una de las transformaciones QVT definidas en NDT. Es capaz de generar los modelos de la fase de Análisis desde el modelo de la fase de Requisitos, los modelos de la fase de Diseño desde los modelos de Análisis, y el modelo de pruebas de sistema de la fase de Pruebas desde el modelo de Requisitos. Además, NDT-Driver permite obtener el modelo Requisitos a partir de los requisitos capturados durante la fase de Estudio de Viabilidad del proyecto. Además, para facilitar el trabajo en proyectos que siguen ciclos de vida iterativos, para cada transformación, NDT-Driver le permite al usuario seleccionar el modelo concreto a generar. En la figura 6 se muestra una ventana de configuración en la que aparecen los modelos disponibles para la transformación de Requisitos a Análisis.

La aplicación de alguna de las transformaciones que define NDT proporciona, sin duda, una posición aventajada al analista a la hora de desarrollar los distintos modelos de una determinada fase del ciclo de vida. Sin embargo, estos modelos generados de forma automática deben ser mejorados por el analista. Durante este

proceso de mejora es posible que se detecten requisitos del cliente que no han sido considerados en el modelo origen de la transformación, lo cual, conllevaría retocar el modelo origen y volver a generar el modelo en cuestión, lo cual además conllevaría la pérdida de todo el trabajo realizado por parte del analista en cuanto a la mejora del modelo generado.

Como solución a este aspecto, NDT-Driver permite dos modos de transformación: Reconstrucción o Actualización. El modo Reconstrucción consiste en volver a generar un modelo desde cero partiendo del modelo origen y descartando el modelo generado, si éste ha sido previamente generado. Mientras que gracias al modo Actualización, sólo se transforman aquellos elementos del modelo original que han sido modificados.

Por ejemplo, si todos los requisitos de almacenamiento han sido ya definidos en la fase de Requisitos, es posible generar el modelo conceptual de la fase de Análisis. Si llegados a este punto, se detectara que algún requisito de almacenamiento no se ha definido según las necesidades del cliente, no es necesario volver a generar de nuevo el modelo conceptual; NDT-Driver permite actualizarlo.

Como conclusión final, el uso de NDT-Driver permite reducir cuantitativamente el tiempo de desarrollo de cualquier proyecto que siga la metodología NDT.

Figura 6. Interfaz de NDT- Driver - Ventana de NDT Driver

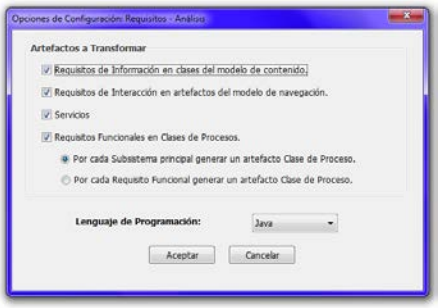

Figura 7. Interfaz de NDT- Driver - Ventana de configuración NDT Driver

4.4. NDT-Report

Toda la información descrita y desarrollada en un proyecto realizado en base a la herramienta NDT-Profile, aunque práctica y funcional, no es amigable a la hora de presentar a usuarios y clientes. Por este motivo, se desarrolló y liberó en 2008 la primera versión de una herramienta de generación documental para NDT denominada *NDT-Report*. Esta herramienta contemplaba la generación de un documento con todos los

requisitos recogidos en la fase de Requisitos y los modelos definidos en la fase de Análisis, generando el documento de análisis.

Debido a la ampliación que ha experimentado la metodología NDT en los últimos años, NDT-Report ha evolucionado con el objetivo de cubrir las necesidades documentales de todas las fases del ciclo de vida.

Actualmente, y aprovechando la potencia de EA para la generación de documentos basada en plantillas personalizadas, NDT-Report está constituido por un amplio conjunto de plantillas integradas dentro de la herramienta NDT-Profile. Con NDT-Report puede generar fácilmente un documento para cada fase del ciclo de vida de NDT: desde la fase de Requisitos hasta la de Mantenimiento, pasando por la fase de Análisis, la de Diseño y la fase de Pruebas del proyecto.

4.5. NDT-Glossary

Durante el desarrollo de un sistema de información, orientado o no a la Web, pueden surgir problemas de terminología debido principalmente a que en el proyecto participan grupos de personas muy heterogéneos y con diferente nivel de conocimiento del sistema a desarrollar: clientes, usuarios finales, jefes de proyecto , diseñadores, ingenieros, etc. Este problema se acrecienta en proyectos de sistemas de información Web en los que los equipos de trabajo son multidisciplinares. Para intentar paliar este problema NDT incluye en su suite una herramienta para la construcción de un glosario terminológico, denominada *NDT-Glossary*.

NDT-Glossary implementa un procedimiento automático que, a partir del análisis de los requisitos de un proyecto de un sistema de información, orientado o no a la Web, y que está realizado en base a la herramienta NDT-Profile, genera la primera versión del glosario terminológico de dicho proyecto. El conjunto de términos, o conceptos, recopilados pasarán a formar parte del glosario terminológico del proyecto en cuestión.

Cada entrada del glosario, correspondiente a un término, contendrá una descripción y un nombre, siendo este último único en el glosario.

Además de recopilar los conceptos del proyecto, NDT-Glossary ofrece la posibilidad de crear un documento en diferentes formatos (PDF o RTF), dónde se recoge de manera ordenada y estructurada toda la información del glosario de términos generado.

4.6. NDT-Prototypes

La única forma de que NDT pueda ser una realidad empresarial pasa por ofrecer herramientas que den soporte al desarrollo de sistemas de información Web. En esta línea de actuación se engloba *NDT-Prototypes*. Esta nueva herramienta de NDT-Suite ha sido liberada recientemente y está proceso de mejora y ampliación de funcionalidad.

Implementa un procedimiento que genera de manera totalmente automática un conjunto de prototipos XHTML a partir de los modelos de navegación descritos en la fase de Análisis de un proyecto desarrollado con la herramienta NDT-Profile. Estos prototipos proporcionan al equipo de desarrollo un punto de partida para poder llevar a cabo la construcción del sistema.

A efectos prácticos, NDT-Prototypes no sólo permite disminuir el tiempo empleado en la construcción de una aplicación Web sino que también proporcionan una herramienta útil para la validación de requisitos con usuarios y clientes.

4.7. Otras Herramientas

En esta sección se describen otras herramientas que se consideran relevantes para proporcionar al lector de una visión completa de NDT-Suite: NDT-Access, NDT-Counter y NDTQ-Framework.

En primer lugar, *NDT-Access* es una herramienta que tiene como objetivo agilizar el desarrollo de nuevas herramientas dentro del paquete NDT-Suite. NDT-Access supone el pilar maestro sobre el que se sustentan todas las herramientas que han sido, y serán, desarrolladas para proporcionar soporte a la metodología NDT para su uso en entornos prácticos. Proporciona una interfaz de comunicación totalmente transparente para llevar a cabo consultas, creaciones, modificaciones y actualizaciones de los datos del proyecto. La implementación de esta herramienta se ha llevado a cabo haciendo uso de la API que proporciona Deiser, empresa distribuidora de Enterprise Architect. Además de transparente, NDT-Access proporciona una interfaz independiente del sistema gestor de bases de datos que se utilice para almacenar los artefactos de NDT.

En segundo lugar se presenta la herramienta *NDT-Counter*. Cada una de las herramientas de NDT-Suite descritas anteriormente, están orientadas a cubrir una determinada necesidad del ciclo de vida software que define NDT. Sin embargo, ninguna de ellas proporciona soporte a la etapa de gestión del proyecto, y más concretamente, soporte a la tarea de estimación del coste que va a suponer el desarrollo de un proyecto software. Para cubrir esta necesidad se desarrolla una nueva herramienta, la cual aún no ha sido liberada, dentro de la suite de NDT: *NDT-Counter*. La técnica de estimación que implementa esta herramienta está basada en la técnica de puntos de casos de uso.

Por último, se presenta brevemente *NDT-Q-Framework*[2]. NDT-Q-Framework no es una herramienta software en sí misma, sino un marco de trabajo para el desarrollo de software de calidad. Ofrece un entorno de trabajo sostenible y ágil adecuado para tanto el desarrollo de aplicaciones software en entornos de las administraciones públicas y empresas privadas. En este marco de trabajo se plantea un desarrollo orientado por procesos y que contempla, no sólo las fases de desarrollo, sino también los aspectos de calidad, gestión, pruebas y mantenimiento. Este marco de trabajo contempla la definición de los procesos, la definición de sus políticas de orquestación y de las herramientas necesarias para su ejecución.

4.8. Ejemplo de uso de NDT-Suite

En este apartado se describirá cuál debería ser la secuencia de trabajo para lograr generar el modelo conceptual final de la fase de análisis. Para este supuesto, supongamos un sistema que permite la gestión de las reservas y estancias de un hotel.

En primer lugar, el equipo de analistas debe utilizar NDT-Profile para comenzar a registrar y catalogar los distintos requisitos del sistema a desarrollar. Por ejemplo, entre los requisitos de almacenamiento de

[2] Más información sobre NDTQ-Framework en: http://www.iwt2.org/iwt2/ ndt-qframework.php

RPM 8 (1) (2011) ISSN 1698-2029

información se pueden especificar los siguientes: datos del huésped, datos de una reserva, datos de una estancia, tipo de habitación, etc.

En la figura 8 se muestran el catálogo de requisitos de almacenamiento de información recogidos en el modelo de requisitos del proyecto. Centrándonos en un requisito concreto, el referido a un cliente del hotel (RA-01.Cliente), como se puede observar en la imagen, en la ventana de propiedades de este requisito aparece reflejada toda la información para su correcta definición: definición del requisito; lista de datos específicos del requisito con su tipo, cardinalidad y descripción; etc.

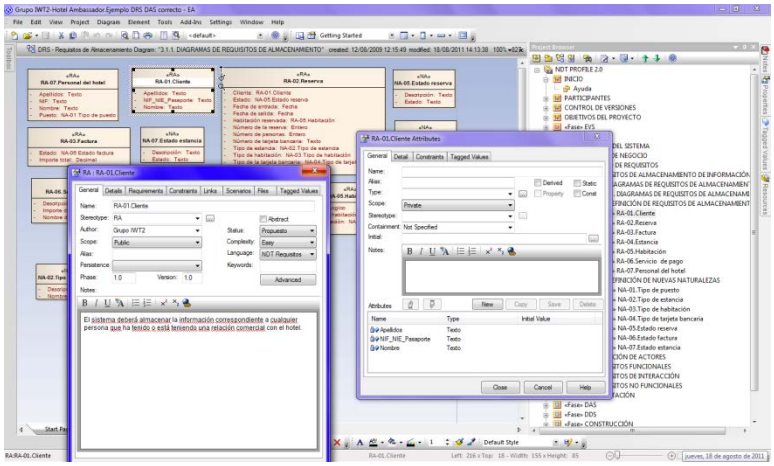

Figura 8. Visualización en NDT-Profile del catálogo de requisitos de almacenamiento de información

Una vez que el equipo termine la ingeniería de requisitos (o durante la misma si lo desea), se ejecutaría NDT-Quality para detectar todos los posibles errores metodológicos.

Una vez solventados todos estos errores, el equipo de analistas podría ejecutar NDT-Driver para generar todos los modelos básicos de la fase de Análisis. Concretamente, en el caso que nos ocupa, generaría el modelo conceptual básico a partir de los requisitos de almacenamiento de información. Visualmente, el equipo de analistas puede corroborar la generación de este modelo dirigiéndose a las carpetas oportunas dentro del NDT-Profile. La figura 9 muestra el modelo conceptual generado.

Una vez finalizada la generación del modelo básico, el equipo de analistas podría modificarlo con el objetivo de mejorarlo y enriquecerlo. Tras esto, el equipo debe volver a ejecutar NDT-Quality para controlar que la trazabilidad se mantiene. Este proceso se puede ir ejecutando en el paso de cada fase.

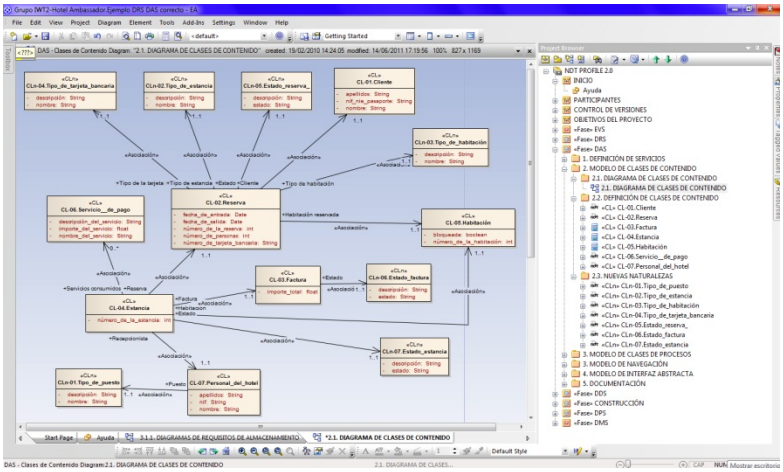

Figura 9. Visualización en NDT-Profile del modelo conceptual de la fase de Análisis

5. La experiencia práctica

Desde sus inicios, NDT ha sido una propuesta ampliamente relacionada con el mundo empresarial y gracias a ello, la metodología ha ido evolucionando y se ha nutrido constantemente del feedback que los proyectos reales les han ido ofreciendo. En [1] se presenta la evolución práctica de la propuesta.

Esta aplicabilidad ha aumentado en los últimos años. Desde hace dos años, varias entidades han asumido NDT como su entorno de trabajo, primero en la fase de requisitos y análisis y luego en la propuesta extendida. En la actualidad, la Consejería de Cultura de la Junta de Andalucía[3], el Servicio Andaluz de Salud[4], la empresa municipal de aguas Emasesa[5] y varias empresas privadas, hacen uso de la suite de NDT.

De hecho, el nacimiento de NDT-Suite ha surgido de las demandas de los proyectos.

NDT-Suite comenzó con la aplicación de NDT al proyecto Diraya. La magnitud de este proyecto: un equipo de desarrollo de más de 80 personas, seis empresas trabajando, un complejo entorno tecnológico y funcional, exigían una herramienta potente y adaptable. Como NDT-Tool no ofrecía estos aspectos se comenzó a trabajar con la Oficina Técnica del Servicio Andaluz de Salud y, tras varios estudios, se comenzó el uso de Enterprise. En el marco de este proyecto se desarrolló en colaboración con la empresa Deiser, distribuidora de EA, un primer profile para NDT en Enterprise que comenzó a usarse con éxito en el proyecto: las empresas podían trabajar de una manera flexible pero homogénea, concurrente, puesto que se disponía de un servidor único donde estaba el fichero EA, etc.

Este proyecto nos motivó a realizar una ampliación del profile y a mejorar muchos aspectos inicialmente no definidos. Se extendió a nuevas fases y se fueron incorporando nuevos artefactos y utilidades. El profile se autodocumentó y se le incluyeron utilidades de ayudas y así surgió NDT-Profile. Este trabajo se realizaba en paralelo con la implantación de NDT en la Consejería de Cultura. Así obteniendo comentarios y opiniones de los equipos de trabajo se elaboró el que actualmente es el perfil en uso.

[3] www.juntadeandalucia.es/cultura
[4] www.juntadeandalucia.es/servicioandaluzdesalud
[5] www.aguasdesevilla.com

NDT-Profile es utilizado por un gran número de proyectos, tanto pequeños (unos 20 o 25 requisitos) hasta proyectos muy grandes (más de 2000 requisitos). Proyectos abordados por equipos heterogéneos o simples y se encuentra en continua mejora.

A medida que se extendía el uso de NDT-Profile y que recibíamos y analizábamos los problemas con la se encontraban los equipos, se comenzó a abordar el desarrollo de las otras herramientas. NDT-Quality resultó un hito importante puesto que permitía, de una manera sencilla, tanto al equipo de desarrollo como a los jefes de equipo controlar si NDT y, por supuesto el profile, se estaban usando de manera correcta.

Aunque no es interés de este trabajo mostrar estadísticas numéricas, sirva de ejemplo que en los inicios de Diraya Especializada, por ejemplo, revisar la calidad de los trabajos realizados por una de sus seis empresas podría tomar más de 1 semana. Actualmente, con NDT-Quality, no se tarda más de 2 minutos. Además, como los equipos disponen de la herramienta pueden validar ellos mismos sus trabajos.

La siguiente herramienta fue NDT-Driver y esta ha sido sin duda una de las más aceptadas en el mundo empresarial. A pesar de que desde el mundo académico se haya demostrado la necesidad de los análisis, la documentación, etc. para los equipos de desarrollo la generación de la documentación sigue siendo una tarea poco valorada y que suele entenderse más como “algo que hay cumplir y que quita tiempo”. NDT-Driver permite automatizar en gran medida todo este trabajo. Así, obtener una primera propuesta de análisis desde una ingeniería de requisitos es tan rápido como un clic de ratón.

A estas herramientas también le han llegado muchas críticas. Críticas que intentamos analizar y solventar. Por ejemplo, una de las más importantes ha llegado desde la Consejería de Cultura y la Consejería de Innovación, Ciencia y Empresa en un proyecto conjunto. Ellos siguen un ciclo de vida iterativo e incremental, en lugar de secuencial. Así, por ejemplo, si en la primera iteración se definían los requisitos, usando NDT-Driver se generaba el análisis básico y ellos evolucionaban hacia un análisis final, cuando en la siguiente iteración incrementaban los requisitos y volvían a usar NDT-Driver, este destruía el trabajo de análisis realizado en la fase anterior.

Para salvar este problema, se ha enriquecido a NDT-Driver con una opción de generación en bloques adaptado especialmente para este tipo de ciclo de vida.

En la actualidad, para todos los proyectos en los que colaboramos como equipo, se llevan una serie de indicadores e inspectores que nos permiten medir o evaluar el uso de todo este entorno. Además, nos ofrecen mucha información relevante para trabajos futuros.

Resulta muy interesante ver cómo, intentando satisfacer a los equipos empresariales de desarrollo con entornos adecuados para ellos que cubran sus necesidades existe una aceptación y una mejor aplicación de las metodologías de trabajo.

6. Conclusiones y trabajos futuros

Este trabajo ha analizado la evolución práctica de NDT y cómo la metodología ha tenido que adaptarse ofreciendo un entorno de herramientas, llamado NDT-Suite, que ofrece una manera sistemática y que vela por la calidad de los resultados.

El trabajo ha presentado una visión general de NDT y una presentación de las herramientas que componen el entorno NDT-Suite. Con un ejemplo práctico se ha tratado de ilustrar cómo se utilizarían estas herramientas y también se ha justificado con un breve recorrido de las experiencias prácticas cómo ha ido surgiendo la necesidad de su existencia.

Respecto a los trabajos futuros sin duda son muchos. La mejora continua de estas herramientas que nos llegan desde los proyectos reales en los que se están utilizando es una fuente de modificaciones, adaptaciones, evoluciones, etc. constante que nos está llevando a ir depurando cada día nuestras propuestas.

Por otra parte, la fase de implementación que incluye construcción e implantación, queda aislada. En este sentido se ha realizado un trabajo en colaboración con la empresa Sun Microsystem y que, actualmente sólo se está usando en uno de los proyectos, el proyecto Diraya Especializada. Esta herramienta, basándose en el análisis del código Java y de la base de datos de Enterprise en la fase de diseño, se encarga de validar que el diseño se corresponde con el código. Esta herramienta, que ha recibido el nombre de CADI es otro de los puntos futuros de trabajo, y que nos están abriendo un trabajo importante para asegurar la calidad y la trazabilidad de los proyectos.

La búsqueda de inspectores y métricas que se obtengan de la manera más sistemática posible es otra línea abierta. Ofrecer indicadores que permitan validar la corrección de los entregables, la evaluación de la calidad de lo que se entrega, etc. es un punto esencial que es necesario para los jefes de equipo y los directores. En la actualidad se han definido un conjunto de inspectores que se llevan midiendo sobre los proyectos desde hace más de un año. Sin embargo, es necesario automatizar la consecución de estos inspectores puesto que actualmente es un trabajo demasiado manual. En la actualidad se están buscando alternativas para que NDT-Quality incluya esos medidores y una vez obtenga la evaluación indique esas medidas para los proyectos.

Agradecimiento

En este trabajo de investigación ha sido soportado por el proyecto Tempros (TIN2010-20057-C03-02), por RED CaSA (TIN2010-12312-E) del Ministerio de Ciencia y Educación, España, y por el proyecto NDTQ-Framework de la Junta de Andalucía, España (TIC-5789).

Referencias

[1] Escalona, M.J. & G. Aragón, G. (2008). NDT. A model-driven approach for web requirements. *IEEE Transaction on Software Engineering*, 34(3) 377-390.

[2] Rossi, G. (1996). An Object-Oriented Method for Designing Hypermedia Applications. PHD Thesis, University of PUC-Rio. Rio de Janeiro. Brazil.

[3] Koch, N., Knapp, A. Zhang, A. & Baumeister, H. (2008). UML-Based Web Engineering, in: G. Rossi, O. Pastor, D. Schwabe, L. Olsina (Eds.) *Web Engineering: Modelling and Implementing Web Applications*, Springer, pp. 157-191.

[4] Ceri, S., Fraternali, P. & Bongio, P. (2000). Web Modelling Language (WebML): A Modelling Language for Designing Web Sites. *Conference WWW9/Computer Networks*, 33(1-6) 137-157.

[5] UML, (2005). *Unified Modeling Language: Superstructure. Specification*, OMG. http://www.omg.org/cgi-bin/doc?formal/05-07
[6] Escalona, M.J., Parra, C.L. & Nieto, J. (2007). Diraya Project. The power of metamodels in real experiences with Web Engineering. AEIPRO 2007. Proceedings of XI International Congress on Project Engineering (2007-2010); España pp. 167,
[7] Escalona, M.J., Aragón, G. Gutierrez, J.J., Ortega, J.A. &Ramos, I. (2008). NDT & Metrica v3. An approach for public organization in Spain. International Conference on Web Information Systems and Technologies (WebIST'08). Madeira, Portugal.
[8] EA (Enterprise Architect), 2010. http://www.sparxsystems.com
[9] D. Pan, D. Zhu, K. Johnson. Requirements Engineering Techniques. Internal Report. Department of Computer Science. University of Calgary. Canada. 2001.
[10] Query QVT-Merge Group (2004). Revised submission for MOF 2.0 Query/Views/ Trans-formations RFP. *Object Management Group*, http://www.omg.org/cgi-bin/apps/doc?ad/04-04-01.pdf.

Medición de la Productividad de los Puestos de Trabajo en Ingeniería del Software

Adrián Hernández-López, Ricardo Colomo Palacios, Ángel García Crespo

Dpto. Informática

Universidad Carlos III de Madrid

Madrid - España

adrianhernandezlopez@gmail.com; {ricardo.colomo, angel.garcia}@uc3m.es

***Abstract**: Productivity is a key element in organizational management. Although it can be measured at different levels (country, sector, organization...) this article focuses on productivity at job position level. Additionally, the area of activity of these jobs is Software Engineering (SE), which is a part of computer engineering, and focuses on the development and maintenance of software. The aim of this paper is to provide a vision of the state of the art about productivity measurement, inputs and outputs of the production process used for this measurement at job position level in SE. In addition, this article raises points that should be addressed to develop new productivity measures, along with some general difficulties in carrying out that task.*

***Resumen**: La productividad es un elemento clave en la gestión organizacional. Aunque puede ser medida a diferentes niveles (país, sector, organización...) este artículo se centra en la productividad a nivel de puesto de trabajo. Además, el área de actividad de estos puestos de trabajo es la Ingeniería del Software (IS) que se enmarca dentro de la ingeniería informática, y se centra en el desarrollo y mantenimiento de software. El objetivo de este artículo es disponer de una visión del estado de la cuestión sobre la medición de la productividad, y de las entradas y salidas del proceso productivo utilizadas para dicha medición a nivel de puesto de trabajo en IS. Además, se plantean los puntos que deben ser tratados para elaborar nuevas medidas de productividad, junto con las dificultades generales para llevar a cabo dicha tarea.*

***Keywords**: Software Engineering, Productivity, Measurement, Job*

1. Introducción

La gestión de la productividad continúa siendo un reto dentro de la gestión de proyectos de las Tecnologías de la Información y Comunicación (TIC). Mientras que en la industria de fabricación se han diseñado y probado métodos para determinar la productividad, la industria TIC está un paso por detrás en términos de disponer de métodos para evaluar las salidas y predecir el esfuerzo necesario para completar los proyectos [1]. La dificultad de la gestión de la productividad en este sector se debe principalmente a que el factor clave del mismo son las actividades con gran necesidad de capital humano [2]. Dentro de la industria TIC, la Ingeniería del Software (IS) tiene como principal actividad el desarrollo y mantenimiento de aplicaciones software.

Las medidas de productividad en los proyectos de software están principalmente basadas en ratios entre el tamaño de software entregado y el esfuerzo realizado para obtenerlo [3]. Siguiendo esta orientación, las dos formas más utilizadas de medir el tamaño de software entregado: los Puntos Función (FP, *Function Points*) y las líneas de código (SLOC, *Source Lines of Code*), pese a ser muy utilizadas, son medidas de dudosa fiabilidad ya que el desarrollo general puede ser improductivo incluso si la productividad a nivel de desarrollo crece [4]. Además de estas formas de medir la salida producida, hay otras propuestas recientes tales como la medición basada en puntos función post mórtem [5], la utilización de medidas de tamaño múltiple [6], y métodos específicos para metodologías específicas tales como la Orientación a Objetos [7]. De forma adicional hay que destacar que las medidas basadas en SLOC sólo representan la productividad de parte de las actividades de programación, mientras que otras actividades como la gestión, el análisis, y el diseño se quedan fuera del alcance de estas medidas. De igual forma, las medidas basadas en FP sólo representan la funcionalidad ofrecida por el software desarrollado pero no tiene en cuenta la parte no funcional del desarrollo, ni otros elementos tales como la reutilización o la accesibilidad.

Los factores que afectan a la productividad TIC están aceptados y reconocidos por la mayoría de las organizaciones: el incremento de las restricciones de almacenamiento, las limitaciones de tiempo, los requisitos de fiabilidad, los lenguajes de alto nivel, el tamaño del equipo de desarrollo, la volatilidad de los requisitos, las habilidades con herramientas de personal, la disponibilidad de personal, la participación de los clientes, y la duración del proyecto (ver por ejemplo [8]). A pesar de su aceptación y reconocimiento, la influencia de algunos de estos factores, tanto de forma positiva, como negativa, en la productividad TIC no es clara y puede variar en función de otros factores externos tales como el sector de negocio [8] y el ratio de *outsourcing* [9].

Finalmente, la medición de la productividad está relacionada con las prácticas de gestión, y más en concreto con la cultura organizacional y su estructura. En organizaciones en las cuales la cultura de informar y comparar está implantada, las medidas de productividad pueden ser consideras como un posible indicador útil para medir el efecto de cualquier mejora en la productividad. Por un lado, en organizaciones cuya gestión esté basada en estilo de mando directivo, donde las ordenes vienen de arriba hacia abajo, las medidas de productividad serán aceptadas como una forma de comunicar a los niveles superiores cómo de bien se están ejecutando los proyectos. Por otro lado, en organizaciones con una cultura de innovación y creatividad, las medidas de productividad podrían ser vistas como una herramienta de control que no sirve para la consecución de los objetivos. Adicionalmente, la unión entre los objetivos organizacionales y la productividad podría no estar clara en estas organizaciones, y esto puede llevar a una falta de comprensión sobre el qué, el cómo y porqué es importante evaluar, y por tanto, medir la productividad.

En este escenario, la definición de nuevos modelos, medidas y métodos de evaluación pueden dilucidar la productividad en los proyectos TIC, especialmente en los proyectos de desarrollo de software. A su vez, las organizaciones necesitan establecer sus propias medidas de productividad, además de las medidas del sector, para realizar un *benchmarking* (evaluación comparativa) de sus datos [8]. Antes de definir nuevos modelos, medidas y métodos para evaluar la productividad en los proyectos de desarrollo de software, es necesario disponer del estado de la cuestión para tener una visión de la situación actual. Así pues, el presente artículo presenta el estado de la cuestión sobre la medición de la productividad a nivel de puesto de trabajo en la IS.

El resto de este artículo se organiza de la siguiente manera: en primer lugar, se presenta un resumen de la productividad en IS desde el punto de vista de su medición; en segundo lugar, se introducen los elementos necesarios para medir la productividad; en tercer lugar, se comenta la definición de puesto de trabajo en IS; en cuarto lugar, se resume el estado actual de la medición de la productividad a nivel de puesto de trabajo en IS; finalmente, se presentan las conclusiones y algunas de las líneas de investigación posibles en esta área de conocimiento.

2. Productividad en Ingeniería del Software

Los orígenes del término “productividad” se remontan al siglo XVIII [10]; sin embargo, hasta mediados del siglo pasado, las definiciones eran confusas. Tradicionalmente, la productividad ha sido definida como el ratio de salidas producidas por unidad de entrada; es decir, como sinónimo de la eficiencia en el uso de la entrada para producir unas salidas [11]. Esta definición encaja bien en el paradigma de la fabricación porque está basado en cantidades de unidades de medida estandarizadas y claramente identificadas, pero no encaja en nuevos

entornos tales como la IS, en los cuales hay activos intangibles, junto con tangibles. Es por ello que en las industrias de servicios, la productividad debe ser vista como un componente del rendimiento, no como un sinónimo de éste [12]. Por otro lado, la productividad tiene por objeto ser comparada con anteriores medidas de productividad, y no como una unidad de resultado, es decir, las medidas de productividad deben ser utilizadas para compararlas sobre el tiempo, mientras que el rendimiento representa una medida temporal. Siguiendo esta puntualización, el valor de las medidas de productividad se encuentra en la capacidad de gestionar y controlar para alcanzar un uso más eficiente de los recursos [13]. Además el objetivo principal de la medición de la productividad es la mejora de la productividad [14]. Tal y como Anselmo y Ledgard indicaron [15] siguiendo la afirmación de Lord Kelvin - *"When you can measure what you are speaking about, … you know something about it; but when you cannot measure it, … your knowledge is of a meager and unsatisfactory kind…".* Por ello, la mejora de la productividad en IS no puede ser realizada sin una medición de la productividad. De este modo, una medida de productividad apropiada aporta una herramienta de pronóstico adecuada para lograr esa mejora de la productividad [14].

En IS, el interés científico por la medición de la productividad puede decirse que comenzó a finales de los años 80. En sus orígenes, los objetos de estudio fueron principalmente las actividades de programación [16] y los proyectos de software [17]. Además, en esa época se empezaron a considerar los factores que afectan a la productividad [18-19]. Con el paso del tiempo, y dado que la medición de la productividad tiene como uno de sus objetivos comparar productividades, fueron apareciendo investigaciones que realizaban comparaciones a diferentes niveles [20-21]. De igual forma, el estudio de los factores continuó en las siguientes décadas [22]. Por otro lado, se fueron incluyendo elementos a la medición de la productividad tales como la reutilización [23-24] o la orientación a objetos [25]. No obstante, la utilización de estos elementos no parece haber sido aceptada ampliamente ya que las medidas de productividad continúan utilizando los estudios de finales de los años 80 como marco de trabajo [26-28]. La finalidad que persiguen estas medidas es medir la eficiencia con la que se termina un proyecto software, bien mediante la medición de las líneas de código desarrolladas (SLO) o de la funcionalidad entregada (FP). Esta utilización además permite emplear la medición para estimar futuros proyectos software utilizando alguno de los métodos disponibles [29]. Así pues, y dado que uno de los retos clásicos de la industria del software es la entrega en tiempo de los proyectos [30], la utilización de medidas de productividad que tengan por objeto medir la eficiencia de entrega no ha de resultar extraño [31]. Sin embargo, estas medidas de productividad no miden todas las actividades de IS, y, por lo tanto, no miden la productividad a nivel de puesto de trabajo. De este modo, es necesario elaborar medidas de productividad para dicho nivel.

Antes de cualquier intento de medir la productividad, es necesario identificar qué debe ser capturado en dicha medida [32]. Así pues, teniendo en cuenta estas contribuciones, y con el objetivo de crear una medida de productividad en IS, es necesario distinguir los factores, entradas y salidas susceptibles de ser medidas. En primer lugar, los factores que afectan a la productividad son de diversa tipología y pueden organizarse en: factores técnicos (producto, proceso, entorno de desarrollo) y factores *soft* (cultura organizacional, cultura de equipo, capacidades y experiencias, entorno, proyecto) [33]. Aunque muchos de estos factores son ampliamente conocidos desde hace años - por ejemplo, la influencia de la experiencia de los trabajadores [34-35] y el lenguaje de programación (entorno de desarrollo) - no está claro que su importancia realmente sea la que hace años se le adjudicó ya que las prácticas y procesos de trabajo así como las herramientas han

evolucionado considerablemente [36]. Además, evaluar el grado de influencia y la valencia del factor es difícil dado que no siempre será la misma y dependerá en gran medida de las características del proyecto y del entorno de trabajo [37]. Lo que sí parece estar claro es la existencia de dos grandes grupos de factores: "factores de personas" (*people factors*) y "factores tecnológicos" (*technical factors*).

En segundo lugar, las entradas del proceso productivo susceptibles de ser medidas utilizadas hasta ahora se centran en unidades de tiempo, principalmente horas y días [38]. La utilización del tiempo, como única entrada, se puede deber a dos intereses: por un lado, la entrega de un proyecto en el tiempo planificado, así como la puesta en el mercado del mismo, elementos que son un factor importante para el éxito de un proyecto; y, por otro lado, el coste o esfuerzo de personal es el mayor coste en las organizaciones productoras de software por lo que el tiempo que se dedica a desarrollar software es el principal coste [39]. Ambos intereses en emplear esta entrada son de índole económica, sin embargo la productividad no es sólo un indicador económico. Además, el empleo de esta entrada plantea algunas cuestiones sin resolver: ¿realmente sólo se necesita tiempo para producir software?, y si sólo se mide el tiempo ¿qué tiempo debería medirse: el empleado por el trabajador para llevar a cabo sus tareas o el pagado por el empleador? De forma paralela y con el avance del tiempo, el coste de los equipos hardware se ha ido reduciendo y el coste de estos recursos es casi irrelevante en proyectos de gran tamaño. Así pues, cualquier recurso de interés para la medición de la productividad no estará relacionado con este recurso sino que estará directa o indirectamente relacionado con la mano de obra empleada. El tiempo es, sin lugar a duda, una de ellas ya que a mayor tiempo mayor empleo de mano de obra, sin embargo no parece ser la única entrada cuando todavía se piensa en la IS como una actividad artesanal [30].

En tercer lugar, las salidas del proceso productivo han seguido la misma filosofía que las entradas de entrega en tiempo por lo que se han medido principalmente en SLOC y/o funcionalidad [40]. Estas medidas de la salida, pese a sus limitaciones, tienen gran aceptación en las organizaciones. Por un lado, los FP son una forma de medir la funcionalidad a desarrollar y, por lo tanto, una forma de medir qué se entrega al cliente midiendo la funcionalidad como salida. Por otro lado, las SLOC son las instrucciones que codifican la funcionalidad desarrollada, de modo que son una medida de bajo nivel por lo que es una medida más tangible. No obstante, estos dos tipos de salida no son los únicos que se producen en la IS. Por ejemplo, una salida importante, que queda fuera de las salidas habitualmente consideradas, es la calidad, la cual afecta a la salida producida (eficacia) y al proceso productivo en sí mismo (eficiencia) [41]. Pero hay otras salidas que se producen, entre ellas existen salidas de tipo organización del trabajo o de tipo, que no siempre son medidas cuando se habla de productividad. Además, hay características de las salidas que pueden afectar a la productividad, tanto por su creación inicial como por su posterior uso, entre ellas destacan la reutilización [15] y la documentación [42].

De forma adicional, el nivel de análisis debe ser tenido en cuenta cuando se mide la productividad en IS ya que no todas las medidas son útiles en todos los niveles. Así pues, debe haber una clara especificación del objetivo de la medida y las audiencias interesadas en los resultados de dichas medidas; esta información debe estar definida antes que cualquier otra información sobre productividad [12]. A nivel de sector y organización, la granularidad de los factores, entradas y salidas envueltas no es suficiente para una medida precisa que pueda ser utilizada como fuente de información en la mejora de ésta; de modo que las medidas a este nivel constituyen sólo un elemento para comparar entre periodos. A niveles más bajos de medición es necesario un amplio análisis de los elementos que deben ser medidos, lo que requiere un proceso establecido (ver por

ejemplo [12]). Llegado este punto, es necesario preguntarse si las prácticas actuales de medición de productividad en IS siguen este tipo de procesos; y, si los factores, las entradas y las salidas están claramente identificados en cada uno de los niveles de medición. Y por otro lado, si factores ampliamente aceptados como la reutilización o la calidad están incluidos (de forma directa o indirecta) en las medidas de productividad empleadas.

3. Elementos necesarios para la medición de la productividad: factores, entradas y salidas

Antes de elaborar una medida de productividad es necesario identificar qué va a ser medido y qué influye en lo que va a ser medido. En concreto, es necesario identificar las entradas y salidas (qué va a ser medido) y los factores (qué influye). Es conocido que en IS, la calidad de las salidas y las entradas del proceso productivo pueden afectar la productividad, por lo tanto, la calidad y la productividad deben ser medidas como conceptos interrelacionados [43]. Tal y como Gummesson [44] observó, la calidad, la productividad y la rentabilidad forman una tripleta en la que todas las partes están relacionadas con el mismo fenómeno: el resultado económico de la organización. La calidad de los productos puede ser significativa y positivamente afectada por la capacidad del personal, los factores del proceso de desarrollo software, y los recursos de despliegue en las etapas iniciales de desarrollo del producto, especialmente en el diseño [45].

La industria IS en gran medida es un sistema abierto en el cual las partes interesadas, los clientes y los usuarios finales afectan a las entradas y salidas, producen una contribución a la eficiencia interna y externa, y por lo tanto a la productividad del proceso productivo (ver por ejemplo [46]). Por consiguiente, es necesario un enfoque totalmente diferente de la productividad para obtener una medida global que establezca como de bien utilizan las organizaciones IS los recursos para crear salidas con la calidad percibida necesaria y el valor para el cliente requerido [43]. Por lo tanto, las medidas de las entradas y las salidas deben tener en cuenta la cantidad y calidad. Esta importancia se refleja en la premisa que Grönroos y Ojasalo [43] establecieron: “Cuanto mejor es la calidad percibida producida utilizando una cantidad determinada de entradas (entradas de los proveedores de servicios y de los clientes), mejor es la eficiencia externa, dando por resultado una mejora de la productividad del servicio” y “Cuanto más eficientemente la organización de servicios utiliza sus propios recursos como entrada en los procesos y cuanto más la organización puede educar y guiar a los clientes hacia entradas de procesos de apoyo para producir una cantidad determinada de salidas, mejor es la [eficiencia] interna”.

3.1. Factores

Los factores que afectan en la medición de la productividad han sido ampliamente estudiados. Por ejemplo, Anselmo y Ledgard [15] indican algunos de estos factores: independencia de los módulos, comprensibilidad del código y la arquitectura, flexibilidad del proceso de desarrollo software, visibilidad de la arquitectura, y la abstracción del proceso de desarrollo software para ser examinado experimentalmente. Además, la importancia de la arquitectura y el diseño en la productividad del desarrollo de software ha sido contrastada [15]. Éstas son propiedades inherentes al entorno de desarrollo de software, y pueden aumentar o disminuir con el diseño. De igual forma, otros factores afectan a la productividad: restricciones de recursos, volatilidad de los requisitos, utilización de herramientas software, complejidad del programa, lenguajes de programación,

involucración del cliente y el usuario, experiencia y capacidades del personal, estabilidad del personal, tamaño del equipo, ratio de *outsourcing*… [8, 47]. Por otro lado, Scacchi [22] dividió estos factores en tres listas de atributos: entorno de desarrollo software, producto del sistema software, y atributos del personal del proyecto. Además, existen revisiones sistemáticas de la literatura sobre este objeto de estudio [33, 36].

Desde el punto de vista de los autores del presente artículo, los factores de personal requieren una especial atención (ver por ejemplo [48]). Las actividades de IS son intensivas en capital humano, por lo que el factor humano tiene que ser analizado en cualquier práctica de gestión con el objetivo de obtener resultados más ajustados a la realidad. En el contexto de la medición de la productividad, es conocido que factores relacionados con el personal tales como las capacidades técnicas y no técnicas, las habilidades, y la experiencia (en diversas áreas: programación, lenguajes de programación, diseño, análisis…) afectan directamente a la productividad pese a que no existe una literatura que trate este asunto [28]. Además de estos factores, y considerando la falta de literatura relacionada con esta área, se propone que otros factores como la motivación [49], la prácticas vinculadas con los resultados y la evaluación dentro de la gestión del desempeño, los sistemas de compensación y beneficios, el clima organizacional y la felicidad de los trabajadores podrían influir en los resultados de productividad; pero no está claro como pueden influenciar y como pueden .ser introducidos en la medición de la productividad. Por lo tanto, se presenta un amplio abanico de posibilidades de investigación al combinar el conocimiento de la gestión de recursos humanos con la gestión de la productividad; de modo que esta combinación, puede llevar a una transferencia de conocimiento con un propósito común de investigación [50].

3.2. Entradas y salidas

En IS las fronteras entre las entradas y las salidas son difusas [51]; por ejemplo, el conocimiento utilizado en los procesos de producción de software puede ser considerado como entrada y como salida, y puede sufrir una transformación durante estos. Un ejemplo de estas fronteras difusas sucede durante la fase de mantenimiento, en la que todos los productos de los procesos de producción de software son utilizados como entradas y son transformados para adaptarlos a las necesidades del cliente, o para solucionar los *bugs* (errores) encontrados. Así pues, debería tenerse en cuenta la realización de un análisis profundo de las entradas ya que el uso de entradas *proxies* (medidas indirectas) introduce sesgos en la medición de la productividad. Además, esta fuente de error es una de las más difíciles de superar, y los posibles métodos para tratar con ello no son todos obvios [52].

Además, definir “una unidad de ingeniería del software” es una tarea difícil, aunque hay ciertas medidas universales de entradas y salidas [29]. Desde el punto de vista de las entradas, algunas medidas de productividad utilizan las horas-hombre como una medida de esfuerzo, y las capacidades y competencias técnicas y no técnicas del personal como un factor correctivo para el esfuerzo. Desde el punto de vista de las salidas, algunas medidas de productividad utilizan las SLOC como medida de cuantificación de la salida y el ratio de errores como un factor correctivo. Otras medidas ampliamente utilizadas son los FP completados y los productos terminados. Estos parámetros son medidas físicas, que pueden ser transformados en unidades monetarias para obtener medidas financieras de productividad. Sin embargo, se debe tener en cuenta que hay algunos problemas con las medidas financieras cuando se utilizan. Por ejemplo, los ingresos no son siempre

una buena medida de la salida, ya que los precios no siempre reflejan la calidad percibida del servicio ofrecido [43].

Una vez que las entradas, salidas, y factores que afectan a la medición de la productividad han sido definidos, es posible formular una medida de productividad. Desde el punto de vista de los autores de este artículo, es sorprendente que a pesar de la controversia existente sobre la validez de algunas medidas de entradas y salidas para medir la productividad en IS se continúen utilizando dichas medidas. Ejemplo de ello son las medidas derivadas de la utilización de las SLOC como salida y de las Horas-Hombre como entrada [42], o la utilización de los FP como salida. Estas medidas continúan en uso en la actualidad en numerosas organizaciones pese a dicha controversia; probablemente bajo la ley que DeMarco y Lister [53] versionan de los principios de la capacidad de ser medido de Gilbs: "Todo lo que necesitas cuantificar puede ser medido de alguna forma mejor que no medirla de ninguna forma".

4. Puestos de trabajo en Ingeniería del Software

La estructura de cualquier organización, independientemente de su forma, necesita de la definición de los puestos de trabajo que forman dicha estructura [54]. Estos puestos de trabajo son los engranajes que deben dar como resultado el cumplimiento de los objetivos y misión de la organización. Por ello, disponer de la definición de los puestos de trabajo existentes en la organización, junto con su valoración, es una necesidad si se desea realizar cualquier cambio organizacional o si se está creando una nueva organización [55]. En concreto, la definición del puesto de trabajo puede contener la siguiente información: la función del puesto, la situación del mismo en el organigrama organizacional, las tareas que se realizan en el mismo (qué, cómo, con qué, para qué (lo hace), frecuencia, tiempo, autonomía, relaciones con otros puestos…), flujo de trabajo, esfuerzos (físicos, intelectuales), riesgos, condiciones de trabajo, supervisión, conocimientos y competencias necesarios, valores, acceso al puesto y destinos [54]. Como se puede apreciar, el puesto de trabajo no es el sitio físico en el que se realizan las actividades de un trabajador, sino la definición integral de una necesidad organizacional. Además, hay que tener en cuenta que una misma persona puede desempeñar más de un puesto de trabajo en la misma organización; y que varias personas pueden desempeñar el mismo puesto de trabajo.

En IS, existen varios puestos de trabajo más o menos reconocidos universalmente, aunque su definición varía en cada organización y están en constante actualización [56]. Por ejemplo, puestos tales como jefe de proyecto, programador o analista son puestos referenciados tanto en la academia como en la industria desde hace años [57] pero su definición es difusa [58]. Una fuente de consulta de estos puestos de trabajo es el proyecto O*Net creado por el departamento de trabajo de los EE.UU (http://www.onetonline.org/). En él es posible consultar descripciones genéricas de puestos de trabajo muy útiles a la hora de crear nuevas o actualizar definiciones de puestos de trabajo existentes. Pese a esta variedad, los puestos de trabajo en IS se pueden agrupar entorno a lo que se denomina trabajadores de cuello blanco (*white-collar worker*), también llamados trabajadores del conocimiento [2, 59]. Estos puestos de trabajo se caracterizan principalmente por la utilización de capital humano, intelectual y relacional para desempeñar las tareas y cumplir los objetivos de los puestos de trabajo [60]. Estas características los diferencian considerablemente de los puestos de trabajos de otras industrias en los que la presencia de la mano de obra desde un punto de vista físico, en lugar de intelectual, es el principal recurso necesario. Además, los recursos que necesitan los puestos de trabajo en IS

son, en gran medida, intangibles (por ejemplo, conocimiento y experiencia) frente a los recursos tangibles de las industrias clásicas [61] . Adicionalmente, la configuración del entorno de trabajo tiene diversas vinculaciones con el puesto de trabajo. Por ejemplo, gracias a la combinación de espacios individuales, para el trabajo técnico con el objetivo de evitar distracciones, con espacios compartidos, para comunicarse y realizar reuniones, es posible aumentar la productividad de las actividades de desarrollo de software [37], pero ¿cuál es la configuración ideal del espacio de trabajo para aumentar la productividad? [62]. Esto hace que la configuración del entorno de trabajo sea un aspecto a tener en cuenta desde la definición de los puestos de trabajo.

Finalmente, y a diferencia de otros sectores, los puestos de trabajo en IS están muy influenciados por la forma en la que el trabajo se estructura. En la mayoría de las organizaciones IS, la organización del trabajo en equipos de trabajo es una realidad [63-65], lo que está contrapuesto con la organización del trabajo en puestos aislados en los que la interacción con otro puesto es recibir una entrada para dar una salida. El trabajo en equipo requiere habilidades y competencias propias de dicha forma de trabajo por lo que su utilización requiere que las definiciones de los puestos de trabajo estén actualizadas de acuerdo con estas nuevas necesidades. Un ejemplo de esta forma de organización del trabajo son las metodologías de desarrollo ágiles [66], cuya utilización va en aumento [67]. Otra de las diferencias es la utilización de una estructura organizacional distribuida a lo largo del planeta, denominada *Global Software Development* (GSD) [68-69]; esta estructura de trabajo tiene como objetivos desarrollar software de forma ininterrumpida y dar mantenimiento continuado. Desde el punto de vista del puesto, esta forma de organización requiere una serie de cambios en los puestos de trabajo: multiculturalidad [70], idiomas [71], gestión de la confianza [72]... Estos cambios hacen que los puestos de trabajo en IS difieran entre cada organización y sea necesario disponer de una definición actualizada de cada puesto de trabajo, máxime cuando se quiere medir la productividad a nivel de puesto de trabajo.

5. Medición de la productividad a nivel de puesto de trabajo en Ingeniería del Software

La medición de la productividad a nivel de puesto de trabajo en IS tiene como origen los trabajos publicados a finales de los años 70 y comienzos de los 80 [19, 35, 73]. En sus comienzos, la medición de la productividad, al igual que otros tipos de medidas, se centraba en la actividad de programación [16]. Por otro lado, en esos años, otros autores indicaban la importancia del factor humano en el desarrollo software [74-75]. En aquel contexto se crearon medidas de productividad basadas en la filosofía ingenieril que utiliza la productividad como sinónimo de eficiencia de recursos. De este modo, se utilizaron medidas basadas en SLOC y posteriormente, tras su definición y creación [40], basadas en FP. Estas medidas no reflejan toda la actividad desarrollada por cada puesto de trabajo, sino que son medidas de productividad que tienen por objetivo medir la eficiencia de la entrega del proyecto.

Con respecto al nivel de medición, la investigación se ha centrado en el sector (por ejemplo, a nivel de país [9]), la organización [76] y proyecto [6] dentro de las mismas de modo que las medidas de productividad a niveles inferiores (equipos de trabajo, puestos de trabajo) no han tenido el mismo interés, quizá por la dificultad de medir a dicho nivel [2]. No obstante, hay que destacar que las medidas empleadas a nivel de puesto de trabajo son las mismas que las empleadas a niveles superiores de medición [77]. Esto representa un problema ya que

los recursos empleados (entradas), los productos y/o servicios generados (salidas) y los factores que afectan a la productividad a este nivel no son los mismos que a niveles superiores, de modo que las medidas empleadas son de dudosa validez para este nivel de medición [78]. Por ello es necesario elaborar nuevas medidas que tengan validez para medir la productividad a nivel de puesto de trabajo. Con el objetivo de establecer los puntos de partida para elaborar estas nuevas medidas se presentan a continuación los puntos clave a tener en cuenta en dicha tarea desde el punto de vista de los factores, de las entradas y las salidas, y de las medidas.

5.1. Factores que afectan a la productividad a nivel de puesto de trabajo en IS

A nivel de puesto de trabajo, los factores que afectan directamente a la productividad, siguiendo la clasificación presentada por Santillo [79] son los referentes al personal del proyecto junto con los relativos a las condiciones de trabajo definidas en el puesto de trabajo [54]. No obstante, de forma indirecta los factores de proceso, producto, y tecnológicos presentados por Santillo deben ser considerados ya que en el caso de disponer de una correcta definición de puesto de trabajo, como la planteada en [54], el proceso corresponde al cómo lo hace, el producto a qué hace, y los tecnológicos al con qué lo hace. Así pues, se observa claramente que la definición de los puestos de trabajo está vinculada con la medición de la productividad ya que aporta la información de partida para plantear una posible medida.

Por otro lado, tal y como se ha planteado anteriormente, los puestos de trabajo en IS son puestos con una alto uso de capital humano por lo que los factores que más pueden influir en la productividad deberían ser los relativos al capital humano. De este modo, factores tales como la motivación, tratada inicialmente en los años 20 por Elton Mayo, y en IS tenida en cuenta desde hace años [49, 80] influyen en la productividad. Sin embargo, no está claro qué motiva a los ingenieros software, cómo están motivados, ni las salidas y beneficios de dicha motivación [49]. Pese a ello, la conclusión general es disponer de trabajadores motivados como fuente de buenos resultados para los proyectos de desarrollo software [81]. Además, existen factores, incluidos también en otras medidas como las relacionadas con la estimación, que influyen en la productividad; por ejemplo, la experiencia en las tareas (análisis, diseño, programación…) o en los lenguajes de programación empleados [34]. Pese a que existe un amplio número de estudios que tratan los factores que afectan a este tipo de puestos de trabajo, el signo y el grado con el que influyen en la productividad, y cómo se afectan entre sí, es una tarea difícil pero necesaria [33].

5.2. Entradas y salidas a nivel de puesto de trabajo en IS

De igual forma que para los factores, las entradas que se utilizan y las salidas que se producen a nivel de puesto de trabajo dependen de cada puesto de trabajo, y deben estar de explicitas en la definición de cada puesto [54]. Si por ejemplo se consideran las salidas clásicas utilizadas (SLOC, FP), y los puestos de trabajo perfilados en Métrica 3 (directivo, jefe de proyecto, consultor, analista, programador) [57], faltarían por definir salidas, generalmente productos y servicios intermedios antes de la puesta en servicio del software. Poniendo el caso del puesto de trabajo analista, para el cual Métrica 3 define su responsabilidad como *"[…] elaborar un catálogo detallado de requisitos que permita describir con precisión el sistema de información, para lo cual mantendrán entrevistas y sesiones de trabajo con los responsables de la organización y usuarios, actuando de el interlocutor entre éstos y el equipo de proyecto en lo que a requerimientos se refiere. Estos requisitos permiten a los analistas elaborar los distintos modelos que sirven de base para el diseño, obteniendo los modelos de datos y de procesos en el caso del análisis estructurado y los modelos de clases e interacción de objetos en análisis*

orientado a objeto. Así mismo realizan la especificación de las interfaces entre el sistema y el usuario." Teniendo en cuenta esta definición, puede decirse que las salidas del analista son catálogos de requisitos, modelos de datos y procesos o de clases e interacción, y especificaciones de interfaces; por otro lado, las entradas son el conocimiento de los usuarios y organización para la cual se construye el proyecto software y los catálogos de requisitos. Este ejemplo ilustra a la perfección la problemática ya que ni genera las salidas clásicamente utilizadas para medir la productividad (SLOC o FP), la frontera entre las entradas y salidas es difusa (genera un catálogo de requisitos que a su vez sirve como entrada para realizar los modelos) [51], y como entradas no sólo utiliza el tiempo sino que emplea conocimiento y el catálogo de requisitos. Además, para generar las salidas interactúa con otros puestos de trabajo y con personas externas a la organización lo que puede ser visto como otra entrada [54]. Además, otro elemento a tener en cuenta debe ser la calidad (tanto de las entradas utilizadas, como de las salidas generadas) [82]. Así pues, parece claro que las entradas y salidas utilizadas en la medición de la productividad a nivel de puesto de trabajo en IS deben estar ligadas al puesto de trabajo y no a la salida final entregada al cliente.

5.3. Medidas de productividad a nivel de puesto de trabajo en IS

Una vez han sido tratados los factores, y las entradas y salidas, es posible abordar las medidas de productividad a este nivel de medición. Teniendo en cuenta la problemática descrita en los apartados anteriores, tiene sentido plantearse si las medidas de productividad empleadas actualmente en IS son de utilidad para medir la productividad de cada uno de los puestos de trabajo. La respuesta será afirmativa si la medida mide lo que el puesto de trabajo debería aportar y lo que debería utilizar para realizar dicho aporte [83]. De este modo, las medidas clásicas (SLOC/t o FP/t) no son de utilidad para todos los puestos de trabajo en IS. Puede que sean de gran utilidad en puestos de generación de código (programador) pero actualmente no todos los desarrollos emplean código nuevo [84], sino que se reutiliza código y por lo tanto emplear medidas basadas en las SLOC producidas puede carecer de utilidad incluso para dichos puestos [85]. Por otro lado, y utilizando el mismo puesto de trabajo (programador), hay que tener en cuenta las tareas de mantenimiento de software en las que las líneas de código generadas no es tanto la finalidad del puesto sino la solución de errores o la inclusión de nueva funcionalidad [86].

Considerando las observaciones anteriores, y volviendo al ejemplo del analista definido en Métrica 3, la medida de productividad a emplear para dicho puesto sería del tipo:

$p_1 = f\ (s_1(\text{catálogo de requisitos}), e_1(\text{conocimiento}), e_2(\text{usuarios}), \text{factores})$

$p_2 = f(s_2(\text{modelos}), e_3(\text{catálogo de requisitos}), e_4(\text{usuarios}), e_5(\text{otros puestos}), \text{factores})$

$pp = g(p_1, p_2, \text{factores})$

donde

f es una medida que emplea las entradas y salidas para dar un resultado de productividad,

p_x es una función que emplea como medidas de entradas (e_x) y salidas (s_x) y devuelve un resultado de productividad para una tarea del puesto de trabajo (p_x),

g es una función que emplea como entradas las productividades de cada actividad del puesto y devuelve un resultado de productividad para el puesto de trabajo (pp),

s_x es una medida de una salida,

e_x es una medida de una entrada,

factores son todos los factores que afectan a la productividad en dicha función.

Pese a que la definición textual es sencilla, la dificultad radica en establecer unidades de medición para cada elemento medido y en definir una unidad de productividad que permita incluir las medidas de cada elemento en cada función de productividad. Por otro los factores, incluida la calidad, deben ser evaluados para ver su importancia y signo en la función de productividad.

6. Conclusiones y líneas futuras de investigación

Los proyectos de IS están creciendo en tamaño y está siendo cada vez más y más complicado cumplir con los requisitos, mientras que la entrega es cada vez más urgente [87]. Por lo tanto, la necesidad de disponer de indicadores sobre la eficiencia y eficacia a diversos niveles es una necesidad diaria en las organizaciones del sector TIC. En esta línea, las medidas de productividad representan un indicador de cuán eficiente es el proceso productivo ejecutado a un nivel específico de análisis en una organización. Por otra parte, la investigación sobre medidas de productividad en IS está principalmente centrada en la productividad a nivel de proyecto o a niveles superiores (organización, sector, industria…), y existe poca literatura sobre medidas de productividad a niveles más bajos (a nivel equipos o a individual) [88]. Existen algunos estudios que reflejan la importancia del diseño en los procesos de producción de software (por ejemplo, [15]), pero continua sin estar claro cómo medir la productividad de los diseñadores de software e incluso las tareas de diseño de software. De este modo, las medidas de productividad a nivel de puesto de trabajo en IS es una necesidad tangible [2].

Por un lado, hay que tener en cuenta que este artículo es un breve estado de la cuestión realizado utilizando una metodología clásica de búsqueda revisión de literatura. Por el contrario, otra forma de construirlo es realizar una revisión sistemática de la literatura [89]. Con este tipo de revisión se consigue que el estudio pueda ser replicado ya que se informa de todos los pasos realizados para construirlo; todo lo contrario que con una revisión literaria en la que no hay un proceso de trabajo ordenado y documentado. Así pues, la realización de una revisión sistemática de la literatura sobre medición de productividad a nivel de puesto de trabajo es una línea futura de investigación.

Por otro lado, este artículo no presenta ningún resultado basado en un estudio al tratarse de una revisión de la literatura. De este modo, y con el objetivo de aumentar el conocimiento en el objeto de estudio, es necesario realizar investigaciones y publicar los resultados obtenidos. Considerando la casuística del objeto de estudio, la utilización de metodologías que exploratorias toma ventaja frente a otras metodologías. La principal ventaja de estas metodologías es que la recogida de información está abierta a cualquier nueva entrada frente a las recogidas de información cerradas [90-91]. Una posible línea de investigación es la realización de un estudio cualitativo mediante entrevistas a trabajadores para ver la visión de la medición de la productividad desde los propios trabajadores en IS.

Finalmente, la creación de nuevas medidas de productividad a nivel de puesto en IS permitirá a investigadores de otras áreas de conocimiento elaborar medidas para puestos de otros sectores, principalmente de puestos del conocimiento. Esta tarea continua siendo un reto desde hace una década [59], y continuará siéndolo a no ser que se realicen esfuerzos desde la academia y la industria para elaborar nuevas medidas basadas en las definiciones de los puestos de trabajo y no partiendo de medidas utilizadas a niveles mayores de medición.

Referencias

[1] Dalcher, D. (2006). Supporting software development: enhancing productivity, management and control. *Software Process: Improvement and Practice* 11(6) 557-559.

[2] Ramirez, Y.W. & Nembhard, D.A. (2004). Measuring knowledge worker productivity: A taxonomy. *Journal of Intellectual Capital* 5(4) 602-628.

[3] MacCormack, A., Kemerer, C.F., Cusumano, M. & Crandall, B. (2003). Trade-offs between Productivity and Quality in Selecting Software Development Practices, *IEEE Software* 20(5)78-85.

[4] Lee, S. & Schmidt, R.C. (1997). Improving application development productivity in Hong Kong. In: *Information technology and challenge for Hong Kong*, B. J.M. & M. B.M. (eds.), Hong Kong University Press.

[5] Asmild, M., Paradi, J.C. & Kulkarni, A. (2006). Using data envelopment analysis in software development productivity measurement. *Software Process Improvement and Practice* 11(6) 561-572.

[6] Kitchenham B.A. & Mendes, E. (2004). Software Productivity Measurement Using Multiple Size Measures. *IEEE Transactions on Software Engineering* 30(12) 1023-1035.

[7] Pendharkar, P.C. (2006). Scale economies and production function estimation for object-oriented software component and source code documentation size," *European Journal of Operational Research* 172(3) 1040-1050.

[8] Maxwell K.D. & Forselius, P. (2000). Benchmarking Software-Development Productivity. *IEEE Software* 17(1) 80-88.

[9] Tsunoda, M., Monden, A., Yadohisa, H., Kikuchi, N. & Matsumoto, K. (2009). Software development productivity of Japanese enterprise applications. *Information Technology and Management* 10(4) 193-205.

[10] Quesnay, F. (1766). Analyse de la formule arithmétique du tableau économique de la distribution des dépenses annuelles d'une nation agricole. *Journal de l'Agriculture, du Commerce & des Finances* 11-41.

[11] Jefferys, J., Hausberger, S. & Lindblad, G. (1954) *Productivity in the distributive trade in Europe: wholesale and retail aspects*, Organisation for European Economic Co-operation.

[12] Sink, D.S., Tuttle, T.C. & DeVries, S.J. (1984). Productivity measurement and evaluation: what is available? *National Productivity Review* 3(3) 265-287.

[13] Fitzgerald L. & Moon, P. (1996). *Performance measurement in service industries: making it work*, Cima.

[14] Nachum, L. (1999). Measurement of productivity of professional services. *International Journal of Operations and Production Management* 9(9/10) 922-950.

[15] Anselmo, D. & Ledgard, H. (2003). Measuring productivity in the software industry. *Communications of the ACM* 46(11) 121-125.

[16] Chrysler, E. (1978). Some basic determinants of computer programming productivity. *Communications of the ACM* 21(6) 472-483.

[17] Albrecht, A.J. (1979). Measuring application development productivity. *Proc. Joint SHARE/GUIDE/IBM Application Development Symposium*, 83-92.

[18] Vessey, I. & Weber, R. (1983). Some factors affecting program repair maintenance: an empirical study. *Communications of the ACM* 26(2) 128-134.

[19] Thadhani, A.J. (1984). Factors affecting programmer productivity during application development. *IBM Systems Journal* 23(1) 19-35.

[20] Cusumano, M.A. & Kemerer, C.F. (1990). A quantitative analysis of U.S. and Japanese practice and performance in software development. *Management Science* 36(11) 1384-1406.

[21] Maxwell, K.D., Wassenhove L.V., & Dutta, S. (1996). Software Development Productivity of European Space, Military, and Industrial Applications. *IEEE Transactions on Software Engineering* 22(10) 706-718.

[22] Scacchi, W. (1994). Understanding Software Productivity. In: *Software Engineering and Knowledge Engineering: Trends for the Next Decade*, Software Engineering and Knowledge Engineering 3, W. D. Hurley, (ed.), pp. 293-321.

[23] Gaffney, J. (1989). Software reuse--key to enhanced productivity: some quantitative models. *Information and Software Technology* 31(5) 258-267.
[24] Banker, R.D. & Kauffman, R.J. (1991). Reuse and Productivity in Integrated Computer-Aided Software Engineering: An Empirical Study. *MIS Quarterly* 15(3) 375-401.
[25] Purao, S. & Vaishnavi, V. (2003). Product metrics for object-oriented systems. *ACM Computing Surveys* 35(2) 191-221.
[26] Kitchenham, B.A. (2010). What's up with software metrics? - A preliminary mapping study. *Journal of Systems and Software* 83(1) 37-51.
[27] Bellini, C., Pereira, R. & Becker, J. (2008). Measurement in software engineering: from the roadmap to the crossroads. *International Journal of Software Engineering and Knowledge Engineering* 18(1) 37-64.
[28] Pfleeger, S.L. (2008). Software Metrics: Progress after 25 Years?. *IEEE Software* 25(6) 32-34.
[29] Boehm, B.W., Abts, C. & Chulani, S. (2000). Software development cost estimation approaches - A survey. *Annals of Software Engineering* 10(1) 177-205.
[30] Boehm, B.W. & Ross, R. (1989). Theory-W Software Project Management Principles and Examples. *IEEE Transactions on Software Engineering* 15(7) 902-916.
[31] Dale, C.J. & van der Zee, H. (1992). Software productivity metrics: who needs them? *Information and Software Technology* 34(11) 731-738.
[32] Gummesson, E. (1992). Quality dimensions: what to measure in service organizations. In: *Advances in services marketing and management*, T. A. Swartz, et al., (eds.), JAI Press, pp. 64-78.
[33] Wagner, S. & Ruhe, M. (2008). A Systematic Review of Productivity Factors in Software Development. *Proc. 2nd International Workshop on Software Productivity Analysis and Cost Estimation (SPACE 2008)*, IEEE Computer Society.
[34] Boehm, B.W. (1981). *Software Engineering Economics*, Prentice Hall PTR.
[35] Walston, C.E. & Felix, C.P. (1977). A method of programming measurement and estimation. *IBM Systems Journal* 16(1) 54-73.
[36] Paiva, E., Barbosa, D., Lima, R. & Albuquerque, A. (2010). Factors that Influence the Productivity of Software Developers in a Developer View. In: *Innovations in Computing Sciences and Software Engineering*, T. Sobh & K. Elleithy (eds.). Springer Netherlands, pp. 99-104.
[37] Sommerville, I. (2010). *Software Engineering*, Addison-Wesley.
[38] Gómez, O., Oktaba, H., Piattini, M. & García, F. (2008). A Systematic Review Measurement in Software Engineering: State-of-the-Art in Measures. *Software and Data Technologies*, Communications in Computer and Information Science 10, J. Filipe, et al. (eds.), Springer Berlin Heidelberg, pp. 165-176.
[39] Jørgensen, M. & Shepperd, M. (2007). A Systematic Review of Software Development Cost Estimation Studies. *IEEE Transactions on Software Engineering* 33(1) 33-53.
[40] Albrecht, A.J. & Gaffney, J.E. (1983). Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation. *IEEE Transaction Software Engineering* 9(6) 639-648.
[41] Al-Darrab, I.A. (2000). Relationships between productivity, efficiency, utilization, and quality. *Work Study* 49(3) 97-104.
[42] Boehm, B.W. (1987). Improving Software Productivity. *Computer* 20(9) 43-57.
[43] Grönroos, C. & Ojasalo, K. (2004). Service productivity: Towards a conceptualization of the transformation of inputs into economic results in services. *Journal of Business Research* 57(4) 414-423.
[44] Gummesson, E. (1995). "Service productivity: a blasphemous approach. In: *Quality, productivity and profitability in service operations (conference papers from the QP&P Research Program 1992–1994)*, E. Gummesson, (ed.), University of Stockholm, Department of Business Administration, pp. 8-22.
[45] Krishnan, M.S., Kriebel, C.H., Kekre, S. & Mukhopadhyay, T. (2000). An Empirical Analysis of Productivity and Quality in Software Products. *Management Science* 46(6) 745-759.
[46] Hsu, J. S.-C., Chen, H.-G., Jiang, J. & Klein, G. (2010). The Role of User Review on Information System Project Outcomes: A Control Theory Perspective. *International Journal of Information Technology Project Management* 1(1) 1-14.
[47] Premraj, R., Shepperd, M., Kitchenham, B.A. & Forselius, P. (2005). An Empirical Analysis of Software Productivity over Time. In: *Book An Empirical Analysis of Software Productivity over Time*, Series An Empirical Analysis of Software Productivity over Time, IEEE Computer Society, pp. 37.

[48] Colomo-Palacios, R., Tovar-Caro, E., García-Crespo, Á. & Gómez-Berbís, J. (2010). Identifying Technical Competences of IT Professionals: The Case of Software Engineers. International Journal of Human Capital and Information Technology Professionals 1(1) 31-43.
[49] Beecham, S., Baddoo, N., Hall, T., Robinson, H. & Sharp, H. (2008). Motivation in Software Engineering: A systematic literature review," Information and Software Technology 50(9-10) 860-878.
[50] Koskinen, K.U. (2008). Boundary brokering as a promoting factor in competence sharing in a project work context. *International Journal of Project Organisation and Management* 1(1) 119-132.
[51] Gupta, A. (1995). Productivity measurement in service operations: a case study from the health-care environment. *Managing Service Quality* 5(5) 31.
[52] Rees, A. (1980). Improving Productivity Measurement. *The American Economic Review* 70(2) 340-342.
[53] DeMarco, T. & Lister, T. (1999). *Peopleware: productive projects and teams*, Dorset House Publishing Co., Inc.
[54] Fernández-Ríos, M. (1995). *Análisis y descripción de puestos de trabajo*, Diaz de Santos, 1995, p. 423.
[55] Schuler, R.S. & Jackson, S.E. (1987). Linking Competitive Strategies with Human Resource Management Practices. The Academy of Management Executive (1987-1989) 1(3) 207-219
[56] Garcia-Crespo, A., Colomo-Palacios, R., Gomez-Berbis, J. M.,& Tovar-Caro, E. (2008). The IT Crowd: Are We Stereotypes?. IT Professional 10(6) 24-27.
[57] *Métrica Versión 3* (2000). Consejo Superior de Informática - Ministerio de Administraciones Públicas.
[58] Alexander, S. (1999). What's in a job title? Less and less, some say. *InfoWorld* 21(21) 1999.
[59] Drucker, P. (1999). Knowledge-Worker Productivity: The Biggest Challenge. *California management review* 41(2) 79-85.
[60] Colomo-Palacios, R., Cabezas-Isla, F., García-Crespo, Á. & Soto-Acosta, P. (2010). Generic Competences for the IT Knowledge Workers: A Study from the Field Knowledge Management, Information Systems, E-Learning, and Sustainability Research. Communications in Computer and Information Science 111, M. D. Lytras, et al. (eds.), Springer Berlin Heidelberg, pp. 1-7.
[61] Rus, I. & Lindvall, M. (2002). Knowledge management in software engineering. *IEEE Software* 19(3) 26-38.
[62] Jones, C. (1995). How Office Space Affects Programming Productivity. *Computer* 28(1) 76.
[63] Carmel, E. (1999). *Global software teams: collaborating across borders and time zones*, Prentice Hall.
[64] Gorla, N. & Lam, Y.W. (2004). Who should work with whom?: building effective software project teams. *Communications of the ACM* 47(6) 79-82.
[65] Sawyer, S. (2004). Software development teams. *Communications of the ACM* 47(12) 95-99.
[66] Cockburn, A. & J. Highsmith, J. (2001). Agile software development, the people factor. *Computer* 34(11) 131-133.
[67] Dyba, T. & Dingsoyr, T. (2009). What Do We Know about Agile Software Development? *IEEE Software* 26(5) 6-9.
[68] Herbsleb, J.D. & Moitra, D. (2001). Global Software Development. *IEEE Software* 18(2) 16-20.
[69] Hernández-López, A., Colomo-Palacios, R., García-Crespo, Á. & Soto-Acosta, P. (2010). Team Software Process in GSD Teams: A study of new work practices and models. *International Journal of Human Capital and Information Technology Professionals* 1(3) 32-53.
[70] Linna, P., Karttunen, E. & Jaakkola, H. (2011). Software engineering companies' multicultural education. In *Proc. 34th International Convention* (*MIPRO, 2011),* pp. 1140-1145.
[71] Abufardeh, S. & Magel, K. (2010). The impact of global software cultural and linguistic aspects on Global Software Development process (GSD): Issues and challenges. In *Proc. 4th International Conference on New Trends in Information Science and Service Science (NISS. 2010)*, pp. 133-138.
[72] Moe, N.B. & Šmite, D. (2008). Understanding a lack of trust in Global Software Teams: a multiple-case study. *Software Process: Improvement and Practice* 13(3) 217-231.
[73] Jones, C. (1981). *Programmer Productivity: Issues for the Eighties*, IEEE Computer Soc. Press.
[74] DeMarco, T. & Lister, T. (1985). Programmer Performance and the Effects of the Workplace. In: *Proc. 8th International Conference on Software Engineering. New York: Institute of Electrical and Electronics Engineers*, pp. 268-272.
[75] DeMarco, T. & Lister, T. (1987). *Peopleware: productive projects and teams*, Dorset House Publishing Co., Inc.
[76] Anda, B.C.D., Sjoberg, D.I.K. & Mockus, A. (2009). Variability and Reproducibility in Software Engineering: A Study of Four Companies that Developed the Same System. *IEEE Transaction Software Engineering* 35(3) 407-429.
[77] Hernández-López, A., Colomo-Palacios, R., García-Crespo, Á. & Cabezas-Isla, F. (2011). Software Engineering Productivity: Concepts, Issues and Challenges. *International Journal of Information Technology Project Management* 2(1) 37-47.

[78] Briand, L. C., Morasca, S. & Basili, V. R. (2002). An Operational Process for Goal-Driven Definition of Measures. *IEEE Transactions on Software Engineering* 28(12) 1106-1125.
[79] Santillo, L. & Moretto, G. (2011). *Software Productivity Factors Taxonomy*, GUFPI-ISMA.
[80] Boehm, B.W. (1981). An Experiment in Small-Scale Application Software Engineering. *IEEE Transactions on Software Engineering* 7(5) 482-493.
[81] Hall, T., Sharp, H., Beecham, S., Baddoo, N. & Robinson, H. (2008). What Do We Know about Developer Motivation? *IEEE Software* 25(4) 92-94.
[82] Procaccino, J.D., Verner, J.M., Shelfer, K.M. & Gefen, D. (2005). What do software practitioners really think about project success: an exploratory study. *Journal of Systems and Software* 78(2) 194-203.
[83] Koch, M.J. & McGrath, R.G. (1996). Improving labor productivity: Human resource management policies do matter. *Strategic Management Journal* 17(5) 335-354.
[84] Mohagheghi, P. & R. Conradi, R. (2007) Quality, productivity and economic benefits of software reuse: a review of industrial studies. *Empirical Software Engineering* 12(5) 471-516.
[85] Banker, R.D., Datar, S.M. & Kemerer, C. (1987). Factors Affecting Software Maintenance Productivity: An Exploratory Study. In: *Proc. Eighth International Conference on Information Systems*, pp. 160-175.
[86] ISO (2006). *ISO/IEC 14764*.
[87] Boehm, B.W. (2006). A view of 20th and 21st century software engineering. In: *Proceedings of the 28th international conference on Software engineering (ICSE'06)*, pp. 12-29.
[88] Erne, R. (2011). What is Productivity in Knowledge Work? - A Cross-industrial View -. *Journal of Universal Computer Science* 17(10) 1367-1389.
[89] Kitchenham, B.A. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering*, Keele University and University of Durham.
[90] Strauss, A. & Corbin, J. (1990). *Basics of qualitative research*, Sage.
[91] Bardin, L. (1986). *El análisis de contenido*, Akal.

Traducción de la nueva versión 4.3.1 del Manual del IFPUG

AEMES es una Asociación sin ánimo de lucro, y entre cuyos fines están el apoyar y promocionar las actividades encaminadas a fomentar los procesos de medición de las Tecnologías de la información, ya que en la Misión de la Asociación figura el contribuir a la difusión de los métodos y técnicas relacionados con la gestión cuantitativa de las Tecnologías de la Información y en particular en aquellos aspectos relacionados con la mejora del proceso de desarrollo y mantenimiento del software y su gestión económica, en las empresas e instituciones que lo desarrollan o lo utilizan, promoviendo el uso de indicadores, métricas y cuadros de mando en las mismas.

AEMES es el Capítulo español del IFPUG

La Asociación Internacional de Usuarios de Puntos Función (IFPUG) es una organización sin ánimo de lucro que se ha propuesto cómo misión el ser un líder reconocido en difundir y fomentar la gestión eficaz de las actividades de desarrollo y mantenimiento de las aplicaciones de software mediante el uso del Análisis de Puntos Función (FPA) y otras técnicas de medición de software.

IFPUG respalda el FPA como su metodología estándar para el dimensionamiento del software. Para soportarla el IFPUG mantiene un Manual de Prácticas de Medición reconocido por la industria como un estándar para el Análisis de Puntos Función.

El IFPUG ofrece formación en esta metodología y una certificación profesional para expertos en FPA, Certified Function Point Specialist (CFPS) basado en el conocimiento del anterior manual, que les habilita para realizar las tareas necesarias para el Análisis de Puntos Función (FPA).

AEMES desde su creación ha considerado que una de las formas más adecuadas para la medición del software, es el uso del tamaño funcional, que cuenta con muchos años de utilización y una amplia experiencia. Y dentro de las técnicas existentes, la más utilizada, y que por tanto, un estándar de facto, es la metodología de Análisis de Puntos Función del IFPUG.

Por consiguiente, desde el principio de su creación, se hizo miembro del IFPUG, pasando posteriormente a ser Capítulo español del IFPUG. Se decidió que para la difusión de esta metodología en el seno de los hispano hablantes, sería más útil traducir al español el Manual que publica la IFPUG, lo que ha venido haciendo desde la versión 4.1.

Revisión histórica de las versiones del Manual de Medición de Puntos Función del IFPUG

El primer documento de partida que tenemos desde que Allan Albretch de IBM, definió los conceptos que facilitan la medida de los productos de desarrollo del software ha sido la publicación *IBM CIS & A Guideline 313, AD/M Productivity Measurement and Estimate Validation*, con fecha 1 de Noviembre de 1984.

Al ir incrementándose el uso de los puntos función, hizo necesario que se publicasen unas guías para interpretar las reglas para los nuevos entornos de software, por lo que en abril de 1988, fue publicada por el International Function Point Users Group (IFPUG), la versión 2.0 del Function Point Counting Practices Manual.

La versión 3.0 del Function Point Counting Practices Manual en abril de 1990 fue un hito en la evolución de la medida del tamaño funcional, haciendo el IFPUG un esfuerzo para presentar una visión consensuada de las reglas de medición de Puntos Función. En este sentido, fue el primer paso para establecer verdaderos estándares para la medición de puntos función que pudieran ser aplicados en distintas organizaciones.

La aparición de la versión 4.0 del Function Point Counting Practices Manual del IFPUG en enero de 1994 fue objeto de polémicas y controversias. Esto se debió a que las reglas de la versión 4.0 tenían el efecto de reducir considerablemente los totales en puntos función en algunas aplicaciones. En esta versión se recogió el uso en fases tempranas de los Puntos Función en el desarrollo de un proyecto para estimar el tamaño del mismo utilizando disciplinas de la ingeniería informática.

En enero de 1999 se publica la versión 4.1 del Function Point Counting Practices Manual, que proporciona aclaraciones sobre las reglas existentes, nuevas o modificadas que trataban situaciones no documentadas con anterioridad y nuevas indicaciones y ejemplos para ayudar a su comprensión.

La evolución de la versión 4.1 respecto a la ver 4.0 fue mucho más suave y menos polémica, ya que los resultados que se obtienen con esta nueva versión son lo suficientemente cercanos a la versión 4.0, como para no tener que rehacer las mediciones.

Según se indica en el Prólogo de la versión 4.1, "*la fiabilidad del análisis de puntos función es suficientemente buena como para que éstos sirvan de base en contratos, en investigación avanzada, en estimaciones de costos y en la realización de "benchmarks" fiables. En tanto no se determine lo contrario, la precisión de los puntos función es igual o superior a la de muchas otras métricas de negocio tales como la tasa interna de rentabilidad, el valor actual neto, o el retorno de la inversión*".

Los principales cambios en ésta versión 4.1 residen en la clarificación de algunas mediciones complejas mediante los ejemplos y en la clarificación de la exposición de la reglas.

Al publicarse ésta versión 4.1 del Function Point Counting Practices Manual, AEMES decidió hacer el esfuerzo de traducirlo, para lo cual creó un Comité de miembros de la Asociación que se brindaron a acometer esta tarea.

Se encargó una primera traducción a un traductor profesional, que cuando la terminó, se comprobó que todavía quedaba una ardua tarea de revisión y homogenización, ya que el traductor no era conocedor del lenguaje técnico empleado.

En el ínterin el IFPUG publicó en abril de 2000 la versión 4.1.1, que, afortunadamente, no hacía cambios sustanciales sobre la versión anterior, por lo que se decidió pasar a traducir ésta versión. Finalmente el Comité, en el que sólo quedaban dos personas, consiguió publicar la versión 4.1.1 en español del Manual para la Medición de Puntos Función, contando con la aprobación del IFPUG.

En enero de 2005 el IFPUG publica la versión 4.2.1 del Manual, que no modifica ninguna de las reglas de la versión anterior, sino que clarifica la interpretación de las dichas reglas, incorporando el contenido de otros documentos publicados como publicaciones independientes. Ésta versión también ha sido traducida al español por AEMES, aprobada por el IFPUG y puesta a disposición de la comunidad hispano hablante.

Mientras tanto, ISO (INTERNATIONAL STANDARDIZATION ORGANIZATION) elaboró la serie de normas ISO/IEC 14143 Software Measurement - Functional size measurement, que varias de ellas han sido traducidas al español y adoptadas como Normas UNE por AENOR. Estas normas especifican los requisitos que deben cumplir una metodología de medida del tamaño funcional del software que permita que se realicen las mediciones de una manera consistente. Estas normas no proporcionan reglas detalladas sobre cómo medir el software usando un método particular, sino que es una base para valorar si un método de medición de software cumple con los requisitos para que sea considerado un Método de Medición Funcional de software.

En consecuencia el IFPUG reestructuró la metodología para que cumpliera con los requisitos de las citadas normas ISO/IEC 14143, excluyendo las 14 Características Generales, consiguiendo que se publicara por ISO la norma ISO/IEC Software measurement - IFPUG functional size measurement method 2009.

En enero de 2010, el IFPUG publica la versión 4.3.1 que contiene modificaciones menores y proporciona nuevos ejemplos, aclaraciones e interpretaciones mejoradas para las reglas vigentes, que tratan de mejorar aún más la coherencia entre mediciones.

AEMES, dentro de su línea de difusión de las metodologías de análisis de la medida funcional del software, ha vuelto organizar un grupo de trabajo compuesto por varios expertos, que se han encargado de traducir al español la versión 4.3.1, que en el mes de septiembre de 2011 ha sido también aprobada por el IFPUG.

Éste documento, al igual que las traducciones anteriores, está disponible sin ningún coste para los miembros de AEMES en la web www.aemes.org.

En el momento actual el Manual para la Medición de Puntos Función del IFPUG está disponible en siete idiomas: inglés, francés, chino, coreano, portugués, español e italiano. Esto es un exponente del grado de aceptación y uso, cada vez más extendido, de ésta metodología.

Madrid, Septiembre de 2011

José Luis Lucero, Vicepresidente de AEMES

IEE, Informáticos Europeos Expertos

RPM 8 (1) (2011) ISSN 1698-2029

Procesos y Métricas en la WWW

En esta sección de la revista se presenta una lista ordenada de sitios web en los que se tratan los temas de interés de los lectores de la misma.

Sitios Web de Asociaciones Nacionales de Medición del Software

Alemania. Asociación Alemana de Medición del Software. DASMA. www.dasma.org
Dinamarca. Asociación Danesa de Métricas del Software. DANMET. www.danmet.dk
Finlandia. Asociación Finlandesa de Métricas del Software. FISMA. www.sttf.fi
Italia. Asociación Italiana de Métricas del Software. GUFPI-ISMA. www.gufpi-isma.org
Holanda. Asociación Holandesa de Métricas del Software. NESMA. www.nesma.nl
Reino Unido. Asociación de Métricas del Software del Reino Unido. UKSMA. www.uksma.co.uk

Sitios Web de Organismos Internacionales de Medición del Software

COmmon Software Measurement International Consortium. COSMIC. www.cosmicon.com
International Function Points Users Group. IFPUG. www.ifpug.com
International Software Benchmarking Standards Group. ISBSG. www.isbsg.org.au

Sitios Web de Laboratorios de Investigación en Medición del Software

Alemania. Laboratorio de Medición del Software. SMLAB. ivs.cs.uni-magdeburg.de/sw-eng/us
Canada. Laboratorio de Investigación en Ingeniería del Software. GELOG. www.gelog.etsmtl.ca
España. Laboratorio de Medición del Software. CuBIT. www.cc.uah.es/cubit

Relación con RPM

Guía para Autores de Artículos de Divulgación

Los artículos de divulgación podrán ser publicados por cualquier persona que pertenezca a una organización miembro de AEMES. Con la pertinente autorización de su organización. Deberán versar sobre algún asunto de interés relacionado con el alcance de AEMES. Los artículos no tendrán revisión por pares pero no podrán ser artículos de información meramente comercial.

Los autores deberán enviar los artículos electrónicamente utilizando la dirección de correo electrónico rpm@aemes.org. Por favor dirigir los artículos al Editor de la Revista de Procesos y Métricas de las Tecnologías de la Información. El artículo debe ser enviado para el proceso de revisión en formato Microsoft Word.

Guía para Autores de Artículos de Investigación

Los artículos de investigación podrán ser publicados por cualquier persona que pertenezca a una organización miembro de AEMES. Deberán versar sobre algún asunto de interés relacionado con el alcance de AEMES.

Los autores deberán enviar los artículos electrónicamente utilizando la dirección de correo electrónico rpm@aemes.org. Por favor dirigir los artículos al Editor de la Revista de Procesos y Métricas de las Tecnologías de la Información. El artículo debe ser enviado para el proceso de revisión en formato Microsoft Word.

El envío de un artículo implica que el trabajo descrito no ha sido publicado previamente (excepto en el caso de una tesis académica), que no se encuentra en ningún otro proceso de revisión, que su publicación es aceptada por todos los autores y por las autoridades responsables de la institución donde se ha llevado a cabo el trabajo y que en el caso de que el artículo sea aceptado para su publicación, el artículo no será publicado en ninguna otra publicación en la misma forma, ni en Español ni en ningún otro idioma, sin el consentimiento de AEMES.

Una vez recibido un artículo se enviará al autor de contacto por correo electrónico un acuse de recibo.

Todos los artículos de investigación recibidos para ser considerados para su publicación serán sometidos a un proceso de revisión. La revisión será realizada por dos o, en su caso, tres expertos independientes. Para asegurar un proceso de revisión lo más correcto posible los nombres de los autores y los revisores permanecerán confidenciales. Una vez revisado un artículo se enviarán por correo electrónico los resultados de la revisión. En el caso de que el artículo haya sido rechazado se adjuntarán las valoraciones de los revisores. El proceso de revisión está libre de costes para los autores.

Una vez que un artículo haya sido aceptado, se solicitará a los autores que transfieran los derechos de autor del artículo a AEMES. Recibida la transferencia, se solicitará a los autores el envío de una versión del artículo lista para publicación que se deberá enviar en formato Microsoft Word.

La publicación de un artículo en la revista está libre de costes para los autores, pero todas las instituciones de origen de todos los firmantes del artículo deberán ser miembros de AEMES.

Guía para la preparación de manuscritos

El texto deberá estar escrito en un correcto castellano (Uso Español) o en Inglés (Uso Británico). Excepto el abstract que deberá estar escrito en un correcto Inglés (Uso Británico).

Abstract y Resumen. Se requiere un abstract en inglés con un máximo de 200 palabras. El abstract deberá reflejar de una forma concisa el propósito de la investigación, los principales y resultados y las conclusiones más importantes. No debe contener citaciones. Se debe presentar a continuación del abstract en inglés una traducción del mismo al castellano bajo el epígrafe Resumen.

Palabras clave. Inmediatamente después del Resumen se proporcionarán un conjunto de 5 palabras clave evitando términos en plural y compuestos, tampoco se deben usar acrónimos o abreviaturas a no ser que sean de un uso ampliamente aceptado en el campo del artículo. Estas palabras claves serán utilizadas a efectos de indexación.

Subdivisión del artículo. Después del Abstract y el Resumen, que no llevarán numeración, se debe dividir el artículo en secciones numeradas, comenzando en 1 y aumentando consecutivamente. Las subsecciones se numerarán 1.1 (1.1.1, 1.1.2, etc.), 1.2, etc. No se deben incluir subdivisiones por debajo del tercer nivel (1.1.1). Cada sección o subsección debe tener un título breve que aparecerá en una línea separada.

Apéndices. Si hay más de un apéndice, se deben identificar como A, B, etc. Las ecuaciones en los apéndices tendrán una numeración separada: (Eq. A.1), (Eq. A.2), etc.
Agradecimientos. Se deben situar antes de las referencias, en una sección separada.
Tablas. Se deben numerar las tablas consecutivamente de acuerdo con su orden de aparición en el texto. Se deben poner títulos a las tablas debajo de las mismas.
Figuras. Se deben numerar las figuras consecutivamente de acuerdo con su orden de aparición en el texto. Se deben poner títulos a las figuras debajo de las mismas.
Referencias. Se debe verificar que cada referencia citada en el texto se encuentra también en la lista de referencias y viceversa. Los trabajos no publicados o en proceso de revisión no pueden ser citados.

- Citaciones en el texto: Un solo autor. El primer apellido del autor, seguido de una coma y la primera inicial, seguida de un punto, a continuación, tras una coma, el año de publicación. Todo entre corchetes. Dos o más autores. Los nombres de los autores, siguiendo el formato de un solo autor, separados por puntos y comas y el año de publicación. Lista. Las listas deberán ser ordenadas, primero de forma alfabética y luego, si fuera necesario, de forma cronológica. Si hay más de una referencia del mismo autor en el mismo año deben ser identificadas por las letras “a”, “b”, etc., situadas después del año de su publicación.
- Referencias. Véase Volumen 1 Número 1 de esta publicación. Apartado 2.8.2.

Formato

Los autores deberán bajar de la página web de RPM en el sitio web de AEMES el artículo de ejemplo y seguir estrictamente el mismo formato.

ASOCIACIÓN ESPAÑOLA DE MÉTRICAS DE SISTEMAS INFORMÁTICOS AEMES

Facultad de Informática de la UPM. Campus de Montegancedo. 28660-Boadilla del Monte (Madrid)
Teléfono: 91 336 66 08. Fax: 91 336 74 12. E-mail: admon@aemes.org

FORMULARIO DE INSCRIPCIÓN A AEMES

Información Personal
Apellidos: Nombre:
Teléfono: e-mail:
Dirección Personal:
Empleo:

Inscripción Institucional o Empresarial
Institución o Empresa: C.I.F.:
Dirección:

Datos Bancarios
Número de Cuenta:
Titular:

Autorizo a AEMES a cargar a la cuenta arriba indicada la cuota Anual de inscripción que asciende a 360 € + la cuota de inscripción que asciende a 150 €

Fecha y Firma

FORMULARIO DE INSCRIPCIÓN A RPM (Revista de Procesos y Métricas)

Información Personal
Apellidos: Nombre:
Teléfono: e-mail:
Dirección Personal:
Empleo:

Inscripción Institucional o Empresarial
Institución o Empresa: C.I.F.:
Dirección:

Datos Bancarios
Número de Cuenta:
Titular:

Autorizo a AEMES a cargar a la cuenta arriba indicada la cuota Anual de subscripción a RPM que asciende a 60 € + Gastos de Envío

Fecha y Firma