Reporte de experiencia: utilización de métricas en Scrum para analizar y mejorar la productividad de un equipo

Vanesa Mola, Luis Mariano Bibbo, Leandro Antonelli

Lifia, Facultad de Informática, UNLP
La Plata, BsAs, Argentina

{vmola, lmbibbo, lanto}@lifia.info.unlp.edu.ar

Abstract. La gestión del proceso de desarrollo de software es una tarea crítica a la hora de cumplir los objetivos del proyecto, entre ellos, la satisfacción del cliente. Las metodologías ágiles ayudan a mejorar los resultados en cuanto a tiempos, calidad y alcance. De todas formas, es necesario analizar y mejorar la productividad del equipo para obtener los resultados esperados. En este trabajo se describe la experiencia de implementar el conjunto de métricas propuestas por Sutherland et al [7] en un entorno de desarrollo Scrum. Se tomaron medidas de tres proyectos en simultáneo durante aproximadamente 10 iteraciones en cada uno. La experiencia requirió la utilización de *story points* como unidades de estimación relativa y la concientización de un patrón de referencia denominado *piedra angular.* Ambos conceptos se pusieron en práctica en dos de las reuniones prescriptas por Scrum: *Planning* y *Daily Meeting*. A partir de la implementación de estas métricas logramos, por un lado, mejorar la previsibilidad de la productividad del equipo y por otro, proveerle una herramienta para analizar su propio desempeño.

keywords: métricas, Scrum, estimaciones, story points.

1 Introducción

La gestión del proceso de desarrollo del software es una tarea crítica por dos motivos. Por un lado, el software es complejo por naturaleza y por otro lado la gestión en sí agrega más complejidad. El software es complejo, porque su esencia es la de modelar la complejidad del contexto en el cual debe ser inmerso [2]. Los dominios están repletos de reglas de negocio que lo caracterizan y si las mismas no son tenidas en cuenta e implementadas en el sistema, él no será útil.

Por otra parte, la gestión de la construcción de software es compleja porque el software es intangible [2]. El mismo está representado a través de líneas de código y de muchos otros modelos: requerimientos, diseños arquitecturales, diseños detallados, casos de prueba, código fuente, etc. Todos estos modelos ayudan a construir una imagen del producto software, pero esta imagen no es

tan completa como la que se tendría al construir un puente o un avión donde el objeto es visible y tangible.
Se han hecho estudios para analizar cuáles eran las principales causas de fracaso en los proyectos [6] [4]. Uno de los principales motivos es la falta o inadecuada planificación y un punto crucial en las planificaciones es la estimación del tamaño del producto. En las etapas iniciales del proyecto, cuando se realiza el plan del mismo, no se cuenta con los detalles necesarios para lograr una estimación con una alta precisión. McConnell [5] define el cono de incertidumbre de las estimaciones donde determina que cuanto más lejos de la fase de construcción se estime el tamaño, mayor será el error cometido. Por lo cual, las estimaciones iniciales de un proyecto se vuelven más un objetivo o compromiso a cumplir, que una previsión certera de cuándo se completaría la construcción.
En este contexto de imprecisiones y de necesidad de cumplir con los clientes, Scrum como desarrollo ágil e iterativo adquiere notoriedad. En los métodos ágiles se desarrollan iteraciones de pocas semanas (generalmente entre 2 y 4) en donde se construye gradualmente el producto y el cliente provee feedback para ajustar el rumbo del proyecto.
En Scrum se suele utilizar *user stories* como unidad de requerimiento y las mismas se pueden estimar a través de una medida de tamaño relativa: *story points*.
A pesar de que Scrum minimiza los desvíos (en comparación con otros procesos de desarrollos clásicos), los mismos de todas formas ocurren, por lo cual es importante mejorar la precisión de las estimaciones con el fin de lograr un mejor rendimiento del equipo. En este reporte describimos la experiencia de haber utilizado un conjunto de métricas que apuntan a analizar y mejorar la productividad de un equipo de desarrollo ágil [7].
El resto del reporte está organizado de la siguiente manera. En la sección 2 describimos el *background* necesario para comprender la experiencia. En la sección 3 describimos la experiencia en sí. Finalmente, presentamos nuestras conclusiones en la sección 4.

2 Background

En esta sección se describirá la esencia de Scrum, como así también las métricas utilizadas en la experiencia que reporta este artículo.
Scrum es un método de desarrollo ágil el cual organiza la construcción a través de iteraciones (*sprints*). Uno de los pilares de los métodos ágiles es el feedback que los clientes brindan al equipo de desarrollo al final de cada *sprint* [1].
Al inicio del *sprint* existe una sesión de *planning* en la cual se determina la funcionalidad a construir durante el mismo. Por otro lado, en cada día del

sprint, se debe realizar una reunión en la que participe todo el equipo. Esta reunión se conoce como *daily scrum meeting* o *stand up meeting*. Cohn [3] recomienda que cada integrante responda a 3 preguntas: (i) ¿qué hice desde la última *daily meeting*?, (ii) ¿qué haré desde ahora hasta la próxima *daily meeting*? y (iii) ¿qué impedimentos tuve o estoy teniendo para lograr mis objetivos?

Los requerimientos se suelen registrar a través de *user stories*, a pesar de que existe un rol específico como es el del *product owner* el cual es el encargado de transmitir oralmente sus necesidades, deseos y expectativas. Las *user stories* se utilizan tanto en la sesión de *planning* para determinar la funcionalidad a construir, como en las *daily meetings* para que el equipo tome conocimiento sobre en que estado se encuentran. Estas *user stories* se suelen estimar en *story points*, una medida de tamaño relativa de tamaño.

Es importante enfatizar que se realiza la medida de tamaño y no tiempo que demanda la construcción. La medición a través de tiempo está condicionada por cierta subjetividad. Y por otro lado, el paso del tiempo, no representa necesariamente progreso en la construcción del producto.

El concepto de las medidas de tamaño tiene que estar presente tanto en la sesiones de planning como en las daily meetings, ya que esta medida de tamaño relativa es la que se tiene que utilizar para estimar el tamaño en el momento de planificar, como así también en el día a día para monitorear el avance del proyecto.

Con el fin de disponer de métricas para tener conocimiento del rendimiento del equipo y lograr mejorarlo, Sutherland et al [7] definieron el siguiente conjunto de métricas:

- Velocity. Determina qué tamaño de software (que satisfaga al *product owner*) el equipo puede conseguir en un *sprint*. Es la suma de los *story points* originalmente estimados para cada *story*, pero considerando solamente las *user stories* que fueron aprobadas por el *product owner*.
- Work capacity. Determina el tamaño del trabajo que puede hacer un equipo en un *sprint*. El equipo puede haber dedicado esfuerzo a *user stories* que no fueron aceptadas por el *product owner*, y *work capacity* es la suma de todo el esfuerzo reportado durante el *sprint*, haya resultado en una *user story* aceptada o no.
- Focus factor. Determina qué porcentaje del esfuerzo invertido termina siendo aceptado por el cliente. Se define como *velocity* / *work capacity*.
- Adopted work. El equipo es el que determina qué *user stories* se compromete a realizar, por lo cual, es necesario conocer si el equipo se compromete a realizar menos de lo que realmente puede. Esta medida se define como el cociente entre el esfuerzo realizado por encima de lo originalmente comprometido y lo originalmente comprometido.

- Found work. Determina el desvío entre el tamaño estimado en la planificación contra el tamaño determinado luego de construirlo (considerando las *user stories* que se estimaron de menos) independientemente de si la *user story* fue aceptada o no. Se define como la diferencia entre lo originalmente estimado y lo finalmente reportado, dividido lo originalmente comprometido.
- Accuracy of estimation. Mide la precisión con que se estimó el tamaño de cada una de las *user stories*. Se define como la suma de todos los *deltas* (tamaño final reportado menos el tamaño original estimado), dividiéndolo por la suma total de lo reportado, para finalmente restar a 1 este valor.
- Accuracy of commitment. Establece el margen de error sobre la porción de trabajo que compromete el equipo. Se define como el cociente entre la suma de todas las estimaciones iniales por un lado, y por otro la suma de las estimaciones iniciales, *adopted work* y *found wor*k.

3 Experiencia

La presente sección describe la experiencia de haber aplicado las métricas descriptas previamente. La sección se encuentra organizada de la siguiente manera. En primer lugar se describe el marco en que se aplicaron las métricas. Luego se describe la piedra angular, un elemento clave en las mediciones. Después se muestran ejemplos reales de las mediciones. Finalmente, se presentan las lecciones aprendidas.

3.1 Contexto de la experiencia

La experiencia reportada en este artículo no es la única experiencia que poseemos en el uso de Scrum y de métricas. Previamente a esta experiencia fuimos recolectando durante más de 50 sprints datos relacionados con métricas basadas en tiempo. A partir de estas mediciones logramos arribar a varias conclusiones y a confirmar algunas otras determinadas por la literatura.
Con el conocimiento de Scrum y de métricas, y motivados por las métricas propuestas por Sutherland et al [7], decidimos hacer la experiencia de aplicar métricas basadas en tamaño relativo con el fin de mejorar la productividad del equipo. La experiencia de utilización de métricas basadas en tamaño se realizó en forma simultánea en 3 proyectos independientes. Cada uno de los proyectos realizó alrededor de 10 sprints.

3.2 Piedra angular

El primer concepto que resulta relevante para poner en práctica las métricas de Sutherland et al [7] es la piedra angular. Dado que la estimación a realizar

es una estimación de tamaño relativa, es necesario definir cuál es el punto de referencia a partir del cual y por comparación se realizan las estimaciones de los distintos productos a construir. Este punto de referencia es la piedra angular (se utiliza este nombre por analogía a los arcos romanos). La misma debe ser tenida en cuenta en las siguientes reuniones de Scrum: *planning* y *daily meetings*.

En la sesión de *planning* el equipo analiza cada una de las *user stories* que construirá, y el mismo equipo es el que debe determinar el tamaño. Este tamaño debe ser indicado en relación a la piedra angular. Es decir, se debe definir una piedra angular que se utilizará a lo largo de todo el proyecto. Esta piedra es una funcionalidad simple y conocida por todos. Entonces, en el momento de estimar cada *user story*, se debe determinar cuánto más (o menos) compleja es la *user story* que se está estimando en relación a la piedra angular.

Por otro lado, en cada una de las reuniones diarias, el equipo se reúne para indicar: (i) qué hizo, (ii) qué hará y (iii) qué problemas se le presentaron. Y también debe reportar el avance en cada *user story*. Este avance también debe estar indicado en forma relativa a la piedra angular.

Por ejemplo, considerando un dominio bancario en donde existen diferentes tipos de cuentas bancarias: cuenta corriente y caja de ahorro, con las operaciones depositar, extraer y consultar el saldo. Podemos considerar que la piedra angular es la operación de extracción en una caja de ahorro y que está estimada en 3 story points.

Al momento de estimar la operación de extraer en una cuenta corriente, podemos considerar que esta operación es más compleja que la extracción de caja de ahorro (porque el banco permite que el saldo de la cuenta corriente pueda estar por debajo de 0, situación no permitida en una caja de ahorro) por lo que podría estar estimada en 5 story points. Por otro lado, podemos considerar que a partir de las nuevas reglamentanciones el banco debe proveer la extracción de dólares desde una caja de ahorro. Si bien esta operación parece similar a la piedra angular, el límite de la extracción en dólares depende de la AFIP, es por ello que estos cálculos hacen que la operación tenga un tamaño de 8 story points.

Ahora consideremos la situación de la *daily meeting*, en donde el equipo reporta el avance de la user story correspondiente a la extracción de dólares de una caja de ahorro. Este avance también debe ser determinado en relación a la piedra angular. Consideremos que luego de un día de trabajo se consigue una funcionalidad similar a la extracción de la caja de ahorro común, por lo cual se debe reportar 3 *story points* (tamaño de la piedra angular). Al segundo día de trabajo, el equipo completa la *user story* y debe reportar nuevamente el avance comparándolo siempre con la piedra angular. Es decir, a pesar de que la diferencia entre el tamaño original estimado de 8 *story points* y el tamaño construido de 3 *story points* del primer día, es de 5 *story points*, el equipo no

debe reportar automáticamente este valor, sino que debe reportar el avance real. Es decir, debe volver a comparar la funcionalidad que desarrolló contra la piedra angular, y reportar lo que resulte de la relación. Por lo cual, si bien la extracción de dólares de una caja de ahorro pudo haber sido estimada en 8 *story points*, luego de su construcción el equipo puede terminar reportando por ejemplo un total de 5 *story points* (sin considera que la construcción del segundo día tiene un tamaño de 2 *story points*).

3.3 Ejemplo de mediciones reales

En la siguiente planilla se muestran las *user stories* (una por fila) de un *sprint* dado, junto a la estimación original realizada en la reunión de *planning*. La planilla también muestra para cada uno de los días el *tracking* que se realizó en las *daily meeting*s.

	Commit Type	Source	Original Estimate	Approval										
				Yes/No	Close?	Mar	Mier	Jue	Vier	Lu	Mar	Mier	Jue	
ar asignación	Planned	New Feature	3	yes	yes	0	1	1	1					
asignación	Planned	New Feature	3	yes	yes	0	1	1	1		0			
ar vinculación	Planned	New Feature	3	no	yes				0	1	1	1		
gestión de los	Planned	New Feature	5	yes	yes	0	2	1	2	0	0			
mente solicitudes	Planned	New Feature	5	yes	yes	0	1	1		1				
un usuario	Planned	New Feature	3	yes	yes					1	1	1		
eso de un	Planned	New Feature	3	yes	yes					1	1		1	
n JMeter	Planned	New Feature	5	yes	yes							1	3	
ts de servicios y	Planned	New Feature	2	yes	yes						0	1	1	
[ACTION] Y	Adopted	New Feature	2	yes	yes							0	2	
culación	Adopted	Bug	1	yes	yes									

Fig. 1. User *stories* con la estimación inicial y *tracking* diario.

El *tracking* diario permite monitorear la evolución del *sprint*. Esto significa que es posible obtener una curva que muestre qué tamaño del producto aún resta por construir. En el siguiente gráfico (que no se corresponde al mismo *sprint* del gráfico previo) se muestran dos curvas: una recta de color rojo con la evolución esperada y en azul la evolución real. En este ejemplo particular se puede observar que en los primeros días del sprint el equipo avanzó más rápido que lo esperado mientras que sobre el final del mismo algunas complicaciones alteraron la evolución que venía dando.

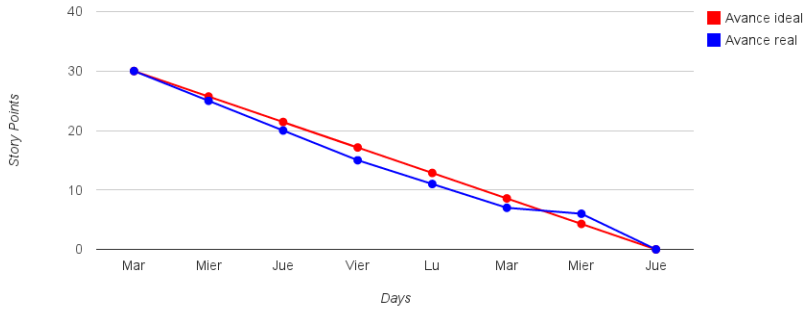

Fig. 2. Evolución día por día del tamaño del producto que resta por construir.

En el siguiente gráfico se muestran ejemplos de medidas.

Team Local Only						
Original Commitment	∑Original Estimates for All Planned Cards	22	SP		Days Available	18
Total Commitment	∑Original Estimates for All Cards, whether Planned or Adopted	27	SP		Predicted Velocity	18
Velocity	∑Original Estimates for all Cards approved by the SPO by the end of the Sprint	24	SP		Velocity average	30.571
Work Capacity	∑All Work Reported by the Team during the Sprint	20	SP		Predicted range	[28-34]
Comparable Between Teams						
Focus Factor	Velocity ÷ Work Capacity			120.0	Should be between 70%-90%.	
Found Work	(∑Total Work Reported per Card - That Card's Original Estimate) ÷ Total Commitment	3	SP	11.1%	Should be less than 20%.	
Adopted Work	(∑Original Estimates for all Adopted Cards) ÷ Original Commitment	5	SP	22.73	Should be less than 20%.	
Accuracy of Estimation	1 - (∑Estimate Deltas ÷ Total Commitment)			51.9%	Should be about 80%.	
Accuracy of Commitment	Velocity ÷ (Total Commitment + Found Work)			80.0%	Should be about 80%.	
	This Sprint would be calculated as a LOSS. (Surprise FAIL. Approve OK.) This is because, although at least 80% of the Total Commitment was approved by the Product Owner, Found + Adopted Work was 33.8% of your Original Commitment. A maximum of 20% is allowable for a win.					

Fig. 3. Medidas para analizar y mejorar la productividad del equipo.

Originalmente el equipo se comprometió con un conjunto de *stories* cuyas estimaciones sumaban 22 *story points*, reflejados en *Original Commitment*. Avanzado el sprint, hubo margen para adoptar algunas otras *stories*, las cuales sumaban 5 *story points*. Esto conforma el numerador del cociente de *Adopted Work* y como consecuencia *Total Commitment* tiene un valor de 27 *story points*. En la reunión de *review*, el *product owner* aprobó solamente 24 puntos de esos 27. Esto hace que *Velocity* haya sido 24. La suma del reporte diario del equipo (*Work Capacity*), considerando la piedra angular, arrojó 20 *story points*. Por otra parte, *Focus Factor* tiene un valor de 120%, que significa que el equipo consiguió más con menos esfuerzo. Luego, *Found Work*, tiene un valor cercano a 11% ya que el desvío entre el tamaño estimado y el tamaño

real luego de construido fue pequeño. *Accuracy of Estimation* que mide qué tan bien el equipo estima el trabajo a realizar, posee un valor bajo de 50% y *Accuracy of Commitment*, que representa qué tan bien el equipo define la cantidad de trabajo a realizar en el sprint, posee un valor alto 80%.
Dentro de los valores que se destacan está el *Adopted Work* con 22,73%. Según la valoración que hace Sutherland se adoptó demasiado trabajo en este *sprint*, ya que ese valor no debería superar el 20%. Esto pudo deberse a que el equipo no comprometió adecuadamente la cantidad de *story points* a realizar en este *sprint*. Recordemos que originalmente se habían comprometido 22 *story points* y luego el compromiso total se llevó a 27 *story points*. Junto con *Accuracy of Estimation*, determinaron que el sprint resultó "perdido" pese a que se produjo más de lo estimado con menos esfuerzo.
Los resultados de los sprints pueden compararse a lo largo del proyecto para analizar su evolución. En el siguiente gráfico se muestra la evolución de: *Velocity* y *Work Capacity* (notar que el *tracking* de trabajo se comenzó a realizar a partir del tercer *sprint*). Por un lado se aprecia, la tendencia de crecimiento de *Velocity*. Y por otro lado la comparación de las dos curvas nos indica que en los primeros *sprints* se necesitaba más *Work Capacity* (trabajo) para obtener determinada *Velocity* y en los *sprints* finales con menos *Work Capacity* se obtuvo mayor *Velocity*.

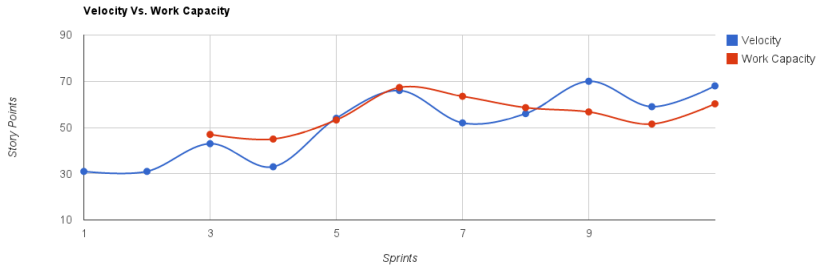

Fig. 4. Comparación de *Velocity* y *Work Capacity* a lo largo del proyecto.

3.4 Lecciones aprendidas

Las lecciones aprendidas de la experiencia de implantación de las mediciones basadas en tamaño con el fin de utilizar las métricas descriptas son dos: (i) la importancia de la piedra angular para registrar el avance y (ii) cómo registrar el avance cuando no hay avance.
La piedra angular es vital para implementar estas mediciones, puesto que es la referencia que se toma para realizar las estimaciones de todas las *user stories* como así también reportar el avance. Esta piedra angular no puede cambiar a lo largo del proyecto, porque desvirtuaría las mediciones realizadas. Por otro lado es claro que en las estimaciones durante el planning se debe utilizar la

referencia de la piedra angular, pero es importante destacar que en las *daily meetings* también hay que considerar la piedra angular y no reportar el avance en función de la estimación original. Y este ajuste que se realiza en el tamaño (porque se deduce un nuevo valor diferente al estimado) es parte del análisis que realizan las métricas adoptadas.

Consideremos ahora la situación cuando no hay avance (es decir, se avanza menos de lo previsto). En este punto hay que analizar el motivo. Si una persona estuvo trabajando en resolver algún impedimento para avanzar con una *user story*, pero que no es una tarea inherente a la misma, *s*e puede reportar 0, lo cual indicaría que la *story* no tuvo avance. Por otro lado, puede suceder que no se haya avanzado según lo previsto porque se descubrió funcionalidad no prevista en el *planning* pero que es funcionalidad inherente a la *story*. En este caso, dado que la funcionalidad es competencia de la *user story*, debe reportarse el tamaño trabajado (siempre en relación a la piedra angular). Esto es necesario reportar, ya que seguramente, el tamaño final de la *user story*, será diferente de lo estimado inicialmente, por lo cual, es necesario registrar el desvío que hubo entre lo estimado y lo real.

Como conclusión, si hay complejidad adicional en la story no prevista en la sesión de *planning*, entonces sí se debería reportar un valor mayor a cero (siempre relativo a la piedra angular). Si, en cambio, simplemente la resolución está llevando más de lo previsto (pero no existe nueva complejidad involucrada) entonces se debería reportar 0.

4 Conclusiones

Utilizar el conjunto de métricas propuesto por Sutherland et al [7] fue una experiencia altamente positiva. Tuvimos que aprender como resolver algunas situaciones que comúnmente se presentan y no estaban prescriptas por las métricas para poder aplicarlas.

Este enfoque de medición por tamaño ha sido positivo para el equipo de desarrollo desde diferentes puntos de vista. Por un lado se tiene una mayor visualización del avance de la construcción, ya que día a día se cuenta con una medida concreta de la evolución, tan necesaria para un producto intangible como es el software.

Por otro lado, con el paso de los distintos *sprints* tomamos conciencia de la importancia de utilizar la piedra angular tanto en la sesión de *planning* como en las *daily meetings*, ya que esto es un punto de referencia para medir el tamaño de las piezas de software a construir y construidas. Asimismo, el hecho de mantener una misma piedra angular a lo largo del proyecto nos sirvió para realizar comparaciones entre los distintos *sprints*. Pudimos notar que más trabajo (*Work Capacity*) no significa necesariamente mayor producción (*Velocity*).

Otro punto a destacar es que con el avance del proyecto fuimos ganando previsibilidad de la capacidad de desarrollar funcionalidad. Esto mejora la relación con el cliente ya que se pueden establecer acuerdos realistas. De todos modos, hay que tener en cuenta que el rendimiento de las personas no es constante a lo largo del proyecto. Es por ello que las métricas de *Velocity* deben ser tomadas como una referencia y no como un objetivo.
Por otro lado, estas métricas son de importancia también en las reuniones de retrospectiva (reunión que se realiza al final del *sprint* para hacer un balance, aprender del mismo y decidir mejoras para el próximo). Las métricas constituyen un disparador para analizar la productividad del equipo con el fin de detectar falencias y aciertos.
Si bien las métricas fueron útiles para ganar previsibilidad del proyecto de desarrollo y mejorar el desempeño del equipo, las mismas no sólo son útiles para un equipo dado, sino que además podrían ser explotadas desde otras perspectivas de la organización. Esto es, realizando análisis transversales entre proyectos.

5 Referencias

1. Agile Manifesto (2012) disponible en: http://agilemanifesto.org/, accedido en Mayo de 2012.
2. Brooks F (1995) The Mythical Man-Month: Essays on Software Engineering, Addison-Wesley Professional, 2 edition.
3. Cohn, M (2009) Succeeding with agile: software development using scrum, Pearson Education, ISBN 978-0-321-57936-2.
4. Keil M, Cule PE, Lyytinen, K, Schmidt RC (1998) A framework for identifying software project risks, in Communication of the ACM, volume 41, Issue 11, nov.
5. McConnell S (2006) Software Estimation: Demystifying the Black Art, Microsoft Press, ISBN: 0735605351.
6. Standish Group (1995), The Chaos Report, disponible en: http://www.standishgroup.com/chaos_resources/index.php
7. Sutherland J, Downey S (2010) “Scrum metrics for hyper productive teams”. Agile 2010 Conference. Orlando, Florida. Disponible en: http://www.agile2010.org/