SCRUM: Una revisión de la literatura

Gabriela Patricia Tomaselli; Cesar J. Acuña; Marcelo Estayno; Cintia Lenkovich
Universidad Tecnológica Nacional, Facultad Regional Resistencia

Abstract
En la actualidad, el desarrollo de software a través de metodologías agiles, particularmente SCRUM, ha ganado terreno dentro de las empresas que se dedican a la producción de software. En este tipo de prácticas, la participación e influencia del equipo en la definición de su propia metodología. Sin embargo es esto mismo lo que, en oportunidades hace que dichas metodologías se utilicen de manera "anárquica", es decir, sin guías claras acerca de los procesos, artefactos, etc. necesario para que el proceso de desarrollo alcance un nivel de calidad aceptable. Para resolver este problema, el grupo GICS de la Facultad Regional Resistencia (UTN) como un punto de inicio en este trabajo se analizan las diferentes propuestas de desarrollo ágil, con el objetivo de establecer las diferencias y semejanzas entre las mismas, y que en conjunto permitan definir una guía uniforme en métodos, artefactos, actividades y roles que caracterizan un proceso de desarrollo ágil.

Palabras Clave
Metodologías Agiles, Scrum, Estado del Arte.

I. Introducción

Desde que se introdujo el concepto de desarrollo de software a través de metodologías agiles, particularmente SCRUM, este se ha sido no solo ampliamente aceptado, sino que se ha convertido un tema de gran controversia en el mundo del software.

Se han encontrado diferentes fuentes y modelos que intentan determinar las características fundamentales que toda metodología ágil debe tener. La literatura destaca la participación e influencia del equipo en la definición de su propia metodología.

La comunidad del software que utiliza estas metodologías agiles se ha distinguido en los últimos tiempos por su falta de interés en la documentación de procesos de desarrollo. Esto trae como consecuencia la ausencia de guías claras u homogéneas de cómo implementar los procesos necesarios en caso de querer adoptar una metodología de desarrollo ágil. Es importante destacar que cuanto más cantidades de definiciones, menos aplicación a diferentes realidades tendrá.

Por ello, se plantea la necesidad de poder evaluar la calidad con la que los procesos agiles son llevados a cabo en las diferentes empresas que se dedican al desarrollo de Software con varios objetivos, pero principalmente con la finalidad de recibir una retroalimentación que permita mejorar o corregir desviaciones en cuanto al uso de las buenas prácticas que las metodologías agiles proponen. Una de las líneas de investigación del grupo GICS es poder diseñar un modelo de referencia para las metodologías agiles, que permita determinar el grado de adherencias de las Pymes de la Región del NEA con respecto a las propuestas Ágiles que existen.

En este sentido y como un punto de inicio de esta línea de trabajo este trabajo se analizan las diferentes propuestas de desarrollo ágil, con el objetivo de establecer las diferencias y semejanzas entre las mismas, y que en conjunto permitan definir una guía uniforme en métodos, artefactos, actividades y roles que caracterizan un proceso de desarrollo ágil.

El presente artículo se estructura de la siguiente manera: en la sección II se presentan las diferentes fuentes que han sido tomadas como referencia para este trabajo y se definen dimensiones comunes que permitan analizar cada una de las propuestas. En la sección III se analizan y comparan las diferentes fuentes, teniendo en cuenta las dimensiones de análisis definidas en la sección II. Finalmente en la sección IV se presentan las principales conclusiones y trabajos futuros.

II. Presentación y Comparación de las Fuentes Analizadas

En esta sección se describen las principales propuestas de desarrollo ágil y se definen dimensiones de análisis comunes para cada una de ellas.

A. Fuentes analizadas

Para alcanzar el objetivo de este trabajo, se ha tomado como referencia 5 de las principales propuestas agiles que se pueden encontrar en la literatura.

La primer fuente que se considera para llevar a cabo este estudio es la desarrolla y mantenida por los autores Ken Schwaber y Jeff Sutherland, quienes son en realidad los inventores del proceso ágil de desarrollo de software conocido como SCRUM. Es su obra SCRUM GUIDE [1] plasman los conceptos principales y fundamentales de la metodología en cuestión, es decir lo que no debería cambiar según se adapte el modelo a las distintas realidades.

Una segunda fuente de interés es la publicada por Scrum Sense, escrita por Peter Hundermark, quien es coach y entrenador certificado en SCRUM y redacta en su obra UN MEJOR SCRUM [2], 12 principios del manifiesto ágil y describe la metodología junto con sus principios básicos.

La tercera fuente analizada es la titulada SCRUM Y XP DESDE LAS TRINCHERAS [3], escrito por Henrik Kniberg, coach en ágil de los directores de la Agile Alliance. Esta fuente manifiesta la experiencia de Kniberg durante la implementación de la metodología en la empresa en la que se encontraba trabajando. Además presenta sus ideas de cómo usar el manifiesto ágil.

La cuarta fuente que se toma para analizar es la publicada por ScrumSense, titulada TODO LO QUE UN PRODUCT OWNER DEBERIA SABER [4], la cual presenta los conceptos fundamentales de la metodología.

La quinta y última fuente considerada para este estudio es la escrita por Peter Demeer, cuyo título es INFORMACION BASICA DE SCRUM [5]. Como su título indica, esta obra presenta descripciones concisas sobre la metodología y sirve como apoyo para literatura escrita por Ken Schwaber y Jeff Sutherland.

B. Dimensiones de análisis

Dado que el objetivo de este trabajo es comparar las distintas propuestas y proponer una guía unificada, no es suficiente solo con describir brevemente la metodología, sino que además es necesario definir las dimensiones de análisis. Para este trabajo se han elegido 3 dimensiones, que permiten analizar las fuentes propuestas transversalmente. Estas dimensiones son: roles, bloques de tiempo y artefactos.

i) Roles

Las actividades del proceso de desarrollo ágil son llevadas a cabo según el rol asignado, en la metodología SCRUM, se pueden aprecias tres roles: Product Owner, ScrumMaster y un equipo de desarrollo, más conocido como Team.

Es importante destacar que el objetivo de la conformación del equipo de desarrollo es optimizar la productividad y la flexibilidad del proceso de desarrollo [1]. Este objetivo puede ser alcanzado gracias a la auto-organización de los miembros de los equipos. Esta auto-gestión no sigue un enfoque del tipo laissezfaire (liberalismo), sino todo lo contrario, son equipos muy disciplinados. Gracias a la autonomía que poseen los miembros, se espera de ellos la responsabilidad y compromiso necesarios para cumplir con lo pactado [2].

El discovery team está compuesto por los siguientes roles: Product Owner, arquitecto y un user interaction.

El delivery team está compuesto por los desarrolladores y por los encargados de QA.

De los roles mencionados anteriormente es preciso comprender las responsabilidades y características de los siguientes:

Product Owner (PO): El PO es el responsable de gestionar uno de los artefactos de esta metodología conocido como Product Backlog y de maximizar el valor del trabajo realizado por el equipo. Se encarga de mantener el Product Backlog y asegurar que sea visible para todos. Para que el PO tenga éxito se deben respetar sus decisiones, estas se hacen visibles en el contenido y priorización del Product Backlog [1]. Se puede decir que el PO es como un CEO, ya que debe tratar de obtener una visión compartida y maximizar el retorno de inversión del proyecto (ROI) [2]. No es responsable del ROI en el mismo sentido de un producto comercial (que dará beneficio), pero es responsable de maximizar el ROI en el sentido de elegir - en cada Sprint – los elementos de más valor de negocio y menos coste [5]. El PO es una persona, no un comité [1] y su principal foco de comunicación son los clientes y stakeholders que estén involucrados en el producto [4].

Team: El equipo está formado por desarrolladores con las capacidades necesarias para transformar los requerimientos del PO, es decir, convertir el Product Backlog en incrementos de funcionalidad entregable en cada Sprint. El equipo debe contar con habilidades especializadas como la programación, QA, arquitectura, diseño de interfaces y de base de datos, y con habilidades compartidas como la comprensión de los requerimientos para poder llevar a cabo los mismos [1]. Las responsabilidades que se pueden destacar son la estimación del tamaño de los ítems del Product Backlog, seguimiento de su propio avance y entrega de incrementos de software [2]. El equipo en Scrum consta de siete personas más menos dos, a los equipos estables se les asocia con una productividad más alta, así que evita cambiar miembros del equipo. A los grupos de desarrollo de aplicaciones con mucha gente se les organiza en varios equipos Scrum, cada uno centrado en diferentes funcionalidades del producto, coordinando sus esfuerzos muy de cerca [5]. Posee una comunicación directa con la comunidad de usuarios o con el mismo cliente. El PO no debe ser el nexo de comunicación, sino que debe haber una comunicación directa para evitar malos entendidos [4].

ScrumMaster: Es el responsable de asegurar que el proceso es comprendido y seguido. El ScrumMaster es responsable de asegurar que el equipo Scrum se adhiere a los valores, prácticas y normas Scrum, y de ayudar a que el Equipo y la organización adopten Scrum. El ScrumMaster enseña al Equipo Scrum mediante entrenamiento y liderándolo para que sea más productivo y a construya productos de mayor calidad. El ScrumMaster ayuda a que el Equipo Scrum comprenda y utilice la auto-gestión y a ser multidisciplinar. El ScrumMaster también debe ayudar al Equipo Scrum a entregar lo mejor de sí mismo, en un entorno de organización que posiblemente aún no esté optimizado para el desarrollo de productos complejos. Cuando el ScrumMaster ayuda a realizar estos cambios, esto recibe el nombre de "eliminar impedimentos." El papel del ScrumMaster es el de servidor y líder del equipo de Scrum. Sin embargo, el ScrumMaster no gestiona al Equipo Scrum; el equipo Scrum es auto-gestionado [1]. El ScrumMaster es un facilitador, coach, líder, mentor y aplanadora [3]. El ScrumMaster no es el jefe del equipo o jefe de proyecto; el ScrumMaster sirve al equipo, le protege de interferencias del exterior, y enseña y guía al DP y al equipo en el uso fructífero de Scrum. Los equipos de Scrum deberían tener un ScrumMaster a tiempo completo, aunque en un equipo más pequeño podría ser un miembro del equipo (llevando una carga de trabajo más ligera). El ScrumMaster y el Dueño de Producto no pueden ser la misma persona; a veces el ScrumMaster necesitará parar los pies al DP (por ejemplo si intenta meter nuevas funcionalidades en mitad de un Sprint). Y al contrario de un jefe de proyecto, el ScrumMaster no le dice a gente las tareas que tienen asignadas. No existe el rol de

jefe de proyecto en Scrum. A veces un (ex-)jefe de proyecto pasa a ser ScrumMaster [5].

En la tabla 1 se presentan los roles que definen las fuentes elegidas para el análisis.

Tabla 1: Roles definidos por las fuentes.

	Fuente 1	Fuente 2	Fuente 3	Fuente 4	Fuente 5
PO	√	√	√	√	√
Desarro-lladores	√	√	√	√	√
Scrum-Master	√	√	√	√	√

ii) Artefactos

Antes detallar esta dimensión es necesario dejar en claro que es un artefacto. Un artefacto es un producto tangible resultante del proceso de desarrollo de software. Ayuda a la descripción de la función, la arquitectura o el diseño del software. En ocasiones un artefacto puede referirse a un producto terminado, además, pueden variar en su necesidad de mantenimiento y actualización.

SCRUM emplea los siguientes artefactos principales:

Product Backlog: es una lista priorizada de todo lo que podría ser necesario en el producto. Los requisitos para el producto están listados en el Product Backlog. El PO es responsable del Product Backlog, de su contenido, disponibilidad y priorización. El Product Backlog nunca está completo. La primera versión para el desarrollo, tan sólo establece los requisitos inicialmente conocidos, y que son entendidos mejor. El Product Backlog evoluciona a medida que el producto y el entorno en el que se utilizará evoluciona. Los elementos del Product Backlog deben tener los siguientes atributos: una descripción, una prioridad, una estimación [1], un ID y como probarlo (Aceptance Criteria). Además se pueden agregar los siguientes campos adicionales: categoría, componentes, solicitante y bug tracking ID [3]. La prioridad está guiada por el riesgo, el valor y la necesidad. Hay muchas técnicas para evaluar estos atributos. La parte más prioritaria del Product Backlog está más clara, y tiene información más detallada que el Product Backlog de menor prioridad. Recibe mejores estimaciones, basadas en la mayor claridad y el mayor detalle [1]. Con estas dos estimaciones (esfuerzo y valor) y quizás con estimaciones adicionales de riesgo, el PO prioriza el Backlog para maximizar el ROI o secundariamente, para reducir algún riesgo importante. Estas estimaciones del valor y esfuerzo se pueden actualizar en cada Sprint a medida que se aprenden cosas nuevas; por tanto, es una actividad continua de re-priorización de la Product Backlog que evoluciona constantemente [5]. A menudo, varios equipos Scrum trabajan juntos en el mismo producto. En ese caso, se utiliza un solo Product Backlog para describir el trabajo futuro en el producto. Se utiliza entonces un atributo del Product Backlog que agrupa a los elementos. La agrupación puede hacerse por conjunto de características, por tecnología o por arquitectura, y es utilizada a menudo por el Equipo Scrum como una forma de organizar el trabajo [1]. Los requerimientos son *emergentes*, lo que significa que no se conocen ni se pueden conocer de antemano todas y cada una de las características que debe contener el producto. Es por ello que Product Backlog es un documento viviente, que requiere una constante *preparación* (*grooming*) para mantenerlo actualizado y útil con el paso del tiempo. Se agregarán nuevos ítems; ítems existentes serán desagregados en múltiples ítems más pequeños; algunos ítems serán removidos al darnos cuenta que ya no son necesarios. Más aún, los ítems deben ser estimados para conocer la relación costo beneficio de los mismos, lo que influirá de manera directa en la ubicación que el mismo tendrá en el Backlog. En prácticamente todos los casos es suficiente, y generalmente preferible, crear y mantener un Product Backlog compuesto exclusivamente de un conjunto de historias de usuario escritas sobre fichas de 150 x 100 mm [2]. Se mantiene esta ficha en un documento Excel

que puede ser editable por todos los miembros del equipo. Oficialmente, el PO es el propietario del documento, sin embargo, muchas veces un desarrollador necesita abrir el documento para clarificar algo o cambiar una estimación. Por la misma razón, no se coloca este documento en el repositorio de control de versiones; en vez de eso, lo almacenamos en una unidad de red compartida [3].

Sprint Backlog: es una lista de tareas para convertir el Product Backlog correspondiente a un Sprint, en un incremento del producto potencialmente entregable. Se compone de las tareas que el Equipo realiza para convertir los elementos del Product Backlog en un incremento "hecho". Muchas de ellas se desarrollan durante la Reunión de Planificación del Sprint. Constituyen todo el trabajo que el equipo identifica como necesario para cumplir con el Objetivo del Sprint. Los elementos del Sprint Backlog deben descomponerse. La descomposición debe ser suficiente para que los cambios en curso puedan ser entendidos en el Scrum Diario. El tamaño normal para un elemento del Sprint Backlog en el que se está trabajando es de un día o menos. El equipo modifica el Sprint Backlog a lo largo de todo el Sprint, así como la parte de Sprint Backlog adicional que surja durante el Sprint. A medida que se van concretando tareas individuales, se puede descubrir que se necesitan más o menos tareas, o que una determinada tarea llevará más o menos tiempo de lo previsto. A medida que es necesario nuevo trabajo, el Equipo lo añade al Sprint Backlog. A medida que se va trabajando o se terminan tareas, se actualizan las horas de trabajo estimado restante para cada tarea. Cuando alguna tarea se revela como innecesaria, es eliminada. Sólo el Equipo puede cambiar su Sprint Backlog durante un Sprint. Sólo el Equipo puede cambiar el contenido, o las estimaciones. El Sprint Backlog es una instantánea muy visible y en tiempo real del trabajo que el equipo tiene previsto realizar durante el Sprint, y pertenece exclusivamente al Equipo [1]. También se conoce al *sprint backlog* como el *tablero de tareas*, que es simplemente una representación física del trabajo al que se han comprometido para lo que resta del sprint. El tablero de tareas es un ejemplo de un *kanban*, una palabra japonesa que significa ―señal visual‖. El mismo comunica al equipo y a cualquiera que desee saberlo qué tareas planificó el equipo y su estado actual [2]. El equipo decide cuántas historias incluirá en el Sprint, no el PO ni nadie más. Para poder determinarlo utiliza dos técnicas: *a ojo de buen cubero* y *cálculos de velocidad, tiempo que hizo ayer o cálculo día/hombre* [3].

*Burndown de Versión***:** mide el Product Backlog restante durante el tiempo correspondiente a una liberación de una versión. El gráfico de Burndown de la Entrega o de Versión registra la suma del esfuerzo restante estimado del Product Backlog a lo largo del tiempo. El esfuerzo se estima en cualquier unidad de trabajo que el Equipo Scrum, y la organización, hayan decidido. La unidad de tiempo que se utiliza generalmente es el Sprint. Durante la Planificación de la Entrega se calculan inicialmente las estimaciones de los elementos del Product Backlog y, también posteriormente, a medida que los elementos son creados. Durante la preparación del Product Backlog las estimaciones son examinadas y revisadas. Sin embargo, pueden ser actualizadas en cualquier momento. El equipo es responsable de todas las estimaciones [1]. Mide el ritmo de entrega de funcionalidades testeadas a lo largo del tiempo. Este ritmo es conocido como la *velocidad del equipo*. Dado que las funcionalidades varían en complejidad y, por ende, en esfuerzo y tiempo, usamos una escala para poder comparar su tamaño. La unidad más conocida es la de *puntos de historia*. Una vez de los miembros del equipo han trabajado juntos por algunos sprints, generalmente puede decirse que han establecido una velocidad que se encuentra

dentro de un cierto rango. Los PO utilizan esta velocidad para predecir el ritmo con el cual el equipo va a entregar funcionalidad en el futuro, lo que lleva a tener planes de entrega cada vez más predecibles. Utilizando este gráfico los Product Owners podrán reportar progreso, determinar fechas de release y predecir el alcance del mismo. [2].

Sprint Burndown: mide los elementos restantes del Sprint Backlog en el transcurso de un Sprint [1]. El gráfico de burndown de tareas del sprint está diseñado para ayudar al equipo en la monitorización de su progreso y para ser el indicador principal que informará sobre sus posibilidades de alcanzar su compromiso al finalizar el sprint. El formato clásico requiere que el equipo estime la duración de cada tarea en horas de forma diaria. El burndown deberá completarse de forma tal que grafique cuántas horas de trabajo restan para concluir el sprint [2]. Idealmente es una gráfica de pendiente *descendiente* que está en la trayectoria para llegar a ―no queda nada de esfuerzo‖ el último día del Sprint. Y aunque a veces tiene buena pinta, frecuentemente no la tiene; es la realidad del desarrollo de productos. Lo importante es que muestre al equipo el progreso hacia su objetivo, no en términos de cuanto *hicimos* en el pasado (un hecho irrelevante en término de *progreso*), sino en término de cuanto trabajo *queda en el futuro* –lo que separa al equipo de su objetivo. Si la línea de trabajo restante no está bajando hasta la finalización del trabajo cuando se acerca el final del Sprint, entonces el equipo necesita hacer ajustes, por ejemplo reducir el alcance del trabajo o encontrar una forma de trabajar más eficientemente al tiempo que mantienen un ritmo sostenible. Aunque se puede usar una hoja de cálculo para crear y mostrar la Gráfica de Trabajo Restante, muchos equipos encuentran más efectivo tenerlo en papel en una pared, con actualizaciones en rotulador; esta solución de ―poca tecnología y mucho contacto‖ es rápida, simple y frecuentemente más visible que una gráfica de ordenador [5].

Backlog de Impedimentos: El backlog de impedimentos es simplemente la lista de situaciones que están impidiendo que el equipo progrese. Estas son situaciones que el ScrumMaster debe quitar del camino en su interminable gesta a través del cual ayudará al equipo para que trabaje de la mejor manera posible [2].

Bug Backlog.

Retrospective Issue y *Action to be taken.*

Mejoras.

En la tabla 2 se muestran los artefactos que definen las Fuentes seleccionadas:

Tabla 2: Artefactos definidos por las fuentes.

	Fuente 1	**Fuente 2**	**Fuente 3**	**Fuente 4**	**Fuente 5**
Product Backlog	√	√	√	√	√
BurnDown de Versión	√	√		√	
Sprint Backlog	√	√	√		√
Sprint BurnDown	√	√			
Backlog de impedimientos		√			
Bug backlog					
Restrospective Issues					
Action to be taken					
Mejoras.					

iii) Bloques de Tiempo

Los bloques de tiempo se refieren específicamente a las reuniones que tienen los integrantes de esta metodología ágil. A continuación se detallan las reuniones existentes en SCRUM:

Sprint: es una iteración durante la cual, el ScrumMaster asegura que no se realizan cambios que afecten al Objetivo del Sprint. Tanto la composición del Equipo como los objetivos de calidad se mantienen constantes durante todo el Sprint, y ocurren uno tras otro, sin tiempo entre ellos. Un Sprint puede ser cancelado antes de que el bloque de tiempo del Sprint se haya

terminado. Sólo el Propietario del Producto tiene la autoridad para cancelar el Sprint, aunque puede hacerlo bajo la influencia de los interesados, del Equipo, o del ScrumMaster. Cuando un Sprint se cancela, cualquier elemento del Product Backlog que haya sido completado y "hecho", es revisado. Estos elementos son aceptados si representan un incremento potencialmente entregable. Si no es así, el elemento se vuelve a colocar en el Product Backlog con sus estimaciones iniciales. Cualquier trabajo realizado en él se asume como perdido. La cancelación de un Sprint consume recursos, ya que todo el mundo tiene que reagruparse en otra Reunión de Planificación de Sprint para iniciar otro Sprint. Los Sprints se componen de: la Reunión de Planificación de Sprint, el trabajo de desarrollo, la Revisión del Sprint, y la Retrospectiva del Sprint [1].

Reunión de Planificación de la entrega: establece el plan y las metas que los Equipos Scrum y el resto de las organizaciones puedan entender y comunicar. El plan de entrega establece el objetivo de la entrega, el Product Backlog de mayor prioridad, los principales riesgos, y las características generales y la funcionalidad que va a contener la entrega. También establece una fecha probable de entrega, y el coste, que debería mantenerse si no cambia nada. La planificación de entrega es completamente opcional. Si los Equipos Scrum empiezan a trabajar sin esta reunión, la ausencia de los artefactos generados en ella se revelará en la forma de impedimentos que hay que resolver. El trabajo necesario para resolver cada impedimento se convertirá en un elemento del Product Backlog. Esta planificación de entrega por lo general no requiere más de un 15-20% del tiempo consumido por una organización para construir un plan de entrega tradicional. Sin embargo, durante el trabajo en una entrega con Scrum, se realizará planificación sobre la marcha en cada Revisión del Sprint y Reunión de Planificación del Sprint, así como diariamente en cada reunión diaria de Scrum. En total, los esfuerzos de planificación de una entrega de Scrum probablemente consumen muy poco esfuerzo más que los de la planificación de entrega tradicional [1].

Reunión de Planificación del Sprint: Durante Reunión de Planificación del Sprint la iteración es planificada. La reunión se restringe a un bloque de tiempo de ocho horas para un Sprint de un mes. Para Sprints más cortos, se debería reservar para esta reunión un tiempo proporcionalmente menor, aproximadamente el 5% de la longitud total del Sprint. La Reunión de Planificación del Sprint y consta de dos partes. La primera parte, un bloque de cuatro horas, es cuando se decide lo que se hará durante el Sprint [1]. El Product Owner presenta la serie de funcionalidades que desea que sean implementadas y el equipo realiza las preguntas necesarias para comprender los requerimientos con el detalle suficiente que le permita comprometerse a entregar dichas funcionalidades al final del sprint. El equipo decide por si mismo cuánto puede entregar, considerando la duración del sprint, el tamaño del equipo y las habilidades y disponibilidad de sus miembros, la definición de LISTO y cualquier acción a tomar decidida durante la retrospectiva que precedió a esta reunión. El equipo se compromete ante el Product Owner a desarrollar aquello que consideran podrá ser entregado testeado y funcionando al finalizar el sprint. Un equipo experimentado podrá usar su velocidad histórica para predecir su capacidad [2]. La segunda parte (otro bloque de cuatro horas para un Sprint de un mes), es cuando el equipo determina cómo se va a convertir esta funcionalidad en un incremento del producto durante el Sprint [1]. Durante esta sesión el equipo colabora de forma tal de crear un diseño a alto nivel de los ítems a los que se ha comprometido. Un resultado de la reunión será el backlog del sprint, o la lista de tareas que el equipo debe ejecutar

de manera colectiva para poder entregar en forma de funcionalidades testeadas [2].

Revisión del Sprint: Esta es una reunión restringida a un bloque de tiempo de cuatro horas para un Sprint de un mes. Para Sprints de menor duración, hay que asignar proporcionalmente menos tiempo de la longitud total para esta reunión. Esta reunión no debe consumir más de 5% del total del Sprint. Durante la Revisión de Sprint, el Equipo Scrum y las partes interesadas debaten sobre lo que se acaba de hacer. En base a eso, y a los cambios en el Product Backlog que se hayan hecho durante el Sprint, colaboran para determinar las próximas cosas que se podrían hacer. Se trata de una reunión informal que incluye por lo menos lo siguiente: El Propietario del Producto identifica lo que se ha hecho y lo que no se ha hecho. El Equipo analiza lo que salió bien durante el Sprint y cuáles son los problemas que encontró, y cómo resolvió estos problemas. El Equipo entonces muestra el trabajo que ha sido completado y responde preguntas. El Propietario del Producto a continuación, analiza el Product Backlog en su estado actual. Proyecta las fechas probables de finalización con distintos supuestos de velocidad. Todo el grupo colabora entonces sobre lo que ha visto y lo que esto significa en cuanto a qué hacer a continuación. La Revisión de Sprint ofrece una valiosa aportación a la siguiente Reunión de Planificación de Sprint [1]. Es a veces denominada de forma incorrecta como *demo*. Si bien es cierto que se incluye en ella una demostración de las nuevas funcionalidades que el equipo desarrolló durante el sprint, su principal objetivo es *inspeccionar* lo entregado por el equipo y obtener feedback de los asistentes a la reunión para poder *adaptar* el plan para sprints subsiguientes. Es por ende mucho más que una mera demo [2].

Retrospectiva del Sprint: Después de la Revisión del Sprint, y antes de la próxima Reunión de Planificación de Sprint, el Equipo Scrum mantiene una reunión Retrospectiva del Sprint. Es una reunión restringida a un bloque de tiempo de tres horas para Sprints de un mes. En esta reunión, el ScrumMaster alienta al Equipo Scrum a revisar, en el marco de proceso y prácticas de Scrum, su proceso de desarrollo (composición del equipo Scrum, la organización de las reuniones, las herramientas, la definición de "hecho", los métodos de comunicación, y los procesos para convertir los elementos del Product Backlog en algo "hecho"), para que sea más eficaz y agradable para el próximo Sprint. Al final de la Retrospectiva del Sprint, el Equipo Scrum debería haber identificado acciones concretas de mejora que se implementarán en el próximo Sprint. Estos cambios se convierten en la adaptación derivada de la inspección empírica [1]. Mientras que la revisión está centrada en el producto, la retrospectiva se encuentra enfocada en el *proceso*. Si bien la revisión se encuentra abierta a quien desee asistir, la retrospectiva se restringe a los miembros del equipo Scrum—esto es el Product Owner, los miembros del equipo de desarrollo y el ScrumMaster [2].

Scrum Diario: Cada Equipo se reúne todos los días 15 minutos en una reunión de inspección y adaptación llamada Scrum Diario. El Scrum Diario se lleva a cabo a la misma hora y en el mismo lugar a lo largo de todos los Sprints. Durante la reunión, cada miembro del equipo, explica:

1. Lo que ha conseguido hacer desde la última reunión;
2. Lo que va a hacer hasta la próxima reunión, y
3. Qué obstáculos tiene en su camino [1].

La reunión diaria NO tiene como objetivo reportar progreso al ScrumMaster, Product Owner o cualquier otro stakeholder. El Product Owner podrá participar de la misma siempre y cuando se mantenga en posición pasiva, hablando únicamente cuando se le realicen preguntas [2].

Reunión de Estimación: para estimar el costo de nuevos ítems del backlog o recalcular el tamaño de ítems de gran costo que deberán ser subdivididos en otros más pequeños, de forma tal que puedan ser desarrollados en los próximos sprints [2].

Backlog Grooming: el equipo debe reunirse con el PO a lo sumo una vez por sprint para planificar los sprints futuros. Además del PO y del Team, pueden asistir a la reunión miembros de la comunidad de usuarios. Entre todos los participantes de la reunión deben escribir las stories para el sprint futuro.

En la tabla 3 se presenta una comparación de las diferentes reuniones propuestas por las fuentes analizadas.

Tabla *3: Bloques de tiempo definidos por las fuentes*

	Fuente 1	**Fuente 2**	**Fuente 3**	**Fuente 4**	**Fuente 5**
Sprint	√	√	√		√
Reunión de Planificación de la Entrega	√		√		√
Reunión de Planificación del Sprint	√	√	√		√
Revisión del Sprint	√	√	√	√	√
Restrospectiva del Sprint	√	√	√	√	√
Scrum Diario	√	√	√		√
Reunión de Estimación		√			
Backlog Grooming				√	

C. Buenas Prácticas

Con las comparaciones de las diferentes propuestas hechas previamente hemos establecido los aspectos básicos necesarios a tener en cuenta para lograr el desarrollo de una metodología ágil obteniendo los mejores resultados. Además de dichos aspectos principales, las diferentes fuentes utilizadas para realizar este trabajo describen un conjunto de ―tips‖ o ―buenas prácticas‖, las cuales algunas no son indispensables para el desarrollo ágil, pero sí se manifiesta que son aconsejables aplicarlas para obtener mejoras que probablemente resulten significantes. Algunas de las buenas prácticas más importantes que se proponen en las fuentes analizadas son: definición clara de ―hecho‖, ubicación física de los equipos, definición de reglas, gestión visual de tareas y utilización de métricas.

i) Definición Clara de "Hecho"

Durante el desarrollo de un Sprint, diferentes personas del equipo pueden considerar terminado un elemento según distintos criterios como por ejemplo si el código se ha construido completamente, o si el producto se ha sometido a pruebas unitarias y de aceptación. Es necesario, por lo tanto, que el PO junto con el equipo establezca la definición clara de ―hecho‖ que los elementos deben cumplir. Dichos elementos o historias no pueden ser tratados de igual forma. Una historia llamada ―Formulario de consulta de usuarios‖ se tratará de forma muy diferente a otra llamada ―Manual de operaciones‖ En el último caso, la definición de terminado puede significar simplemente ―Aceptado por el equipo de operaciones‖[3]. Por lo tanto las definiciones de ―hecho‖ deben tratarse por cada elemento y es imprescindible que estas sean conocidas y comprendidas de forma univoca y clara por todos los participantes del proyecto, evitando ambigüedades. Más que una simple buena práctica, la definición clara de ―hecho‖, consiste en algo deseable a alcanzar cuanto mayor sea posible.

ii) Ubicación Física de los Equipos

El manifiesto ágil dice que desarrolladores y representantes del negocio deben trabajar juntos de manera cotidiana y que las conversaciones cara a cara son la mejor manera de transferir información[2]. Un documento escrito es una representación de una idea que puede llevar a diferentes interpretaciones. Por lo tanto cuando se utiliza la escritura como medio de comunicación se pueden generar

malentendidos que se irán acumulando sucesivamente.

Compartir el espacio físico es un elemento importante para que un equipo pueda trabajar colaborando de forma efectiva. Un estudio sobre equipos de desarrollo de software con equipos que compartían el espacio físico [Teasley, Covi, Krishnan, & Olson, 2000] mostró que fueron el doble de productivos que otros equipos cuyos miembros se encontraban separados entre sí [2].

iii) Definición de Reglas

Las reglas sirven de unión para los bloques de tiempo, los roles y los artefactos de Scrum[1]. Es conveniente establecer determinadas reglas que van a regir el desarrollo del proyecto. Una regla de Scrum puede ser que sólo los miembros del equipo pueden hablar durante un Scrum Diario[1]. Cuando no existen reglas establecidas, los usuarios de Scrum deben deducir qué hacer. El ScrumMaster es el responsable de enseñar al Equipo a trabajar con Scrum aplicando las reglas.

iv) Gestión Visual de Tareas

En Scrum las tareas no se asignan a personas según determinados ―puestos de trabajo‖ sino más bien se busca tener trabajadores multi-funcionales donde los miembros del equipo trabajan conjuntamente ayudando en las tareas pendientes que existen en cada momento. Considerando esto, una buena práctica es usar una herramienta visual de seguimiento de tareas en forma de un gran tablero de tareas en la pared donde las tareas se cambian durante el Sprint entre columnas etiquetadas con ―No empezado‖, ―En progreso‖ y ―Completado‖ [5]. Este tablero debe ser inspeccionado diariamente por el equipo desarrollador de manera de poder informarse en forma rápida y fácil del estado actual de las tareas y mediante eso definir lo que cada uno llevara a cabo en el Sprint Diario.

v) Utilización de Métricas

Es razonable que cualquier gerente desee, dentro del rango de lo posible, poder medir los resultados de la transición de un equipo u organización hacia métodos agiles[2]. Esta medición resulta de gran ayuda para poder evaluar la eficiencia del equipo y corregir aspectos que sean necesarios.

Existen métricas que pueden implementarse de manera simple y rápida. Algunas de las métricas recomendables para utilizar las al comienzo de una implementación son: Encuestas de satisfacción de equipo y clientes, gráfico de velocidad, gráfico de burndown, cuántos test están automatizados, deuda técnica, trabajo en proceso y tiempo promedio de finalización de historias. Una vez que se ha podido establecer algún tipo de medida concreta al valor de negocios, se pueden agregar otras métricas como: costo por Sprint o punto de historia, verdadero valor agregado, retorno de inversión [2].

En la tabla 4 se presenta una comparación de las diferentes buenas prácticas o tips propuestas por cada una de las fuentes analizadas.

Tabla 4: Buenas prácticas definidas por las fuentes

	Fuente 1	**Fuente 2**	**Fuente 3**	**Fuente 4**	**Fuente 5**
Definición Clara de ―Hecho‖	√	√	√	√	√
Ubicación Física de los Equipos		√			
Definición de Reglas	√				
Gestión Visual de Tareas					√
Utilización de Métricas					√

III. Conclusiones

En este artículo se presentan los resultados finales de un trabajo de recopilación bibliográfica llevado a cabo en el seno del grupo GICS de la Facultad Regional Resistencia (UTN). Este trabajo se encuadra dentro de la línea de investigación que pretende diseñar un modelo de referencia

para las metodologías agiles, que permita determinar el grado de adherencias de las Pymes de la Región del NEA con respecto a las propuestas Ágiles que existen.

En este sentido y como un punto de inicio de esta línea de investigación, se analizaron diferentes propuestas de desarrollo ágil, con el objetivo de establecer las diferencias y semejanzas entre las mismas, y que en conjunto permitan definir una guía uniforme en métodos, artefactos, actividades y roles que caracterizan un proceso de desarrollo ágil.

El siguiente paso en este trabajo de investigación, consiste en elaborar un modelo de referencia utilizando los resultados de este artículo, este modelo establecerá cuales las características mínimas y máximas que debe tener un proceso ágil para ser considerado como tal. Actualmente se encuentra en proceso la elaboración de encuestas que permitan determinar el conjunto de artefactos utilizados por las diferentes empresas de desarrollo de Software de NEA y poder compararlas con el modelo de referencia.

Referencias

[1] Ken Schwaber, Jeff Sutherland (2010) The Scrum Guide.

[2] Peter Hundermark (2011) Un mejor Scrum.

[3] Henrik Kniberg (2007) Scrum y XP desde las Trincheras.

[4] (2001) Todo lo que un PO debería saber.

[5] Peter Demeer (2009) Información básica sobre Scrum.

Datos de Contacto:

Gabriela Patricia Tomaselli. Universidad Tecnológica Nacional – Facultad Regional Resistencia. French 414. E-mail: gabriela.tomaselli@gmail.com

Cesar J. Acuña. Universidad Tecnológica Nacional – Facultad Regional Resistencia. French 414. E-mail:csr.acn@frre.utn.edu.ar

Marcelo Estayno. Universidad Nacional de Lomas de Zamora – Fcaultad de Ingeniería. E-mail: mestayno@fibertel.com.ar

Cintia Lenkovich. . Universidad Tecnológica Nacional – Facultad Regional Resistencia. French 414. E-mail: cintialenkovich@gmail.com