



# Scrum a la colombiana: breve acercamiento al scrum hecho en Colombia

## Colombian Scrum: Brief Approach to the Scrum Made in Colombia

Carlos Andrés Piza Martínez<sup>1</sup>

**Para citar este artículo:** Piza C. A. (2016). Scrum a la colombiana: breve acercamiento al scrum hecho en Colombia. *TIA*, 4(2), pp. 88-92.

### ARTÍCULO CORTO

**Fecha de recepción:**  
22-11-2015

**Fecha de aceptación:**  
19-10-2016

ISSN: 2344-8288

Vol. 4 No. 2

Julio - Diciembre 2016

Bogotá-Colombia

### Resumen

El método Scrum o *framework* tiene diferentes puntos de vista dentro de la comunidad, definitivamente es ágil pero gracias a sus resultados y no a su ejecución; este concepto va desde el Scrum team hasta el comercial que vende un proyecto a un cliente. Ahora, ¿cómo empezamos?, ¿el gerente es solo gerente o Scrum Master también?, el arquitecto puede asumir esa responsabilidad, pero ¿quién controla la arquitectura?, el equipo puede hacerlo, ¿están capacitados para tal tarea?, ¿todos son proactivos?, ¿cómo está el equipo de actitud frente al trabajo?, ¿quién es el más comprometido? Motivemos al resto del equipo ¿cómo?, el cliente debe saber qué es Scrum, por lo menos en qué consiste para que prepare su equipo y poder obtener un proyecto exitoso.

**Palabras clave:** arquitecto de software, framework, kaisen, roles scrum.

### Abstract

Scrum method or framework has different points of view within the community, it's definitely agile but thanks to its results and not to its execution. This concept goes from the Scrum team to the commercial who sells to project to a customer. Now, how do we start? The manager is only manager or Scrum Master too? The architect can assume that responsibility, but who controls the architecture? The team can do it, are they qualified for such a task? Are they all proactive? How is the team attitude to work? Who is the most committed? Let's motivate the rest of the team, how? The customer must know what Scrum is, at least what it consists, for him to prepare his team and get a successful project.

**Keywords:** Software architect, framework, kaizen, scrum roles.

<sup>1</sup> Ingeniero de sistemas, Universidad EAN. Correo electrónico: carlosandrespiza@gmail.com

## INTRODUCCIÓN

En el presente artículo se pretende enseñar una realidad desde la cual el lector se identifique con las acciones reales del Scrum en Colombia; también da algunas pautas de lo que en realidad es el Scrum, cómo abordarlo y aclarar algunos términos que los principiantes o algunos experimentados desconocen. Los sacaré de la zona de confort en la que muchas personas están al momento de documentarse en este método sin aplicarlo posteriormente; tenemos que aceptar que somos colombianos, a un lector extranjero le servirá como guía del comportamiento de un trabajador colombiano, las negociaciones a mediano nivel de Colombia, un poco del comportamiento de las empresas colombianas y las cosas que cambian con respecto al desarrollo de software nacional.

## EN REALIDAD QUÉ ES SCRUM

### *Framework o método*

Cuando empezamos a documentarnos nos aseguran que es un *framework* porque debemos apegarnos a los valores y fundamentos que Scrum requiere para hacer Scrum, es el molde al que debemos ajustarnos para que hagamos un Scrum real. En Colombia no es así, hago referencia solo de Colombia porque es donde lo he intentado aplicar. Se habla de Scrum como método porque no se cumple a cabalidad los estrictos fundamentos y valores, me atrevo a comparar a Scrum como un material maleable y que según quién lo use se convertirá en otra cosa, no Scrum. Pienso que cuando lo aplicamos al pie de la letra es una barra de oro; sin embargo, cuando empezamos a cambiar pocas cosas, como el tiempo de ejecución de las celebraciones o reuniones, empezamos a convertirlo en una barra de Plomo; cuando ya el *daily* se pospone, el día del *review* cambia (ya sea más temprano o más tarde) porque el *product owner* tiene otra reunión o sea la excusa que sea,

ya vamos por un trozo de madera, y si el equipo no hace la retrospectiva, eso es una plastilina; tendríamos que hacer un alto en el camino y volver a RUP que es lo tradicional.

Hay empresas de tecnología colombianas que ponen los fundamentos y valores como principios, se le adhiere a la carne, se apropian de Scrum. La discusión de si Scrum es un *framework* o un método también es gran responsabilidad del cliente; en Colombia, Scrum es algo que pocas empresas conocen, ya sea porque es un método solo para construir software (primer gran error, no es solo para el software) o porque las empresas ya funcionan de una manera y no se atreven a cambiar para evolucionar. Es un deber de las empresas de tecnología, dar a conocer en realidad qué es, cómo funciona y cómo aporta valor agregado a las empresas clientes; aclarando las reglas de juego podremos empezar a trabajar, lo importante es que también nuestros clientes se involucren y hagan parte de la construcción de su propio software.

### ¿Qué hace ágil a Scrum?

Scrum hace que el desarrollo de software sea ágil, pero esto no quiere decir que el módulo de contabilidad va a salir en menos tiempo del requerido, que va a estar en producción en la mitad del tiempo que toma con RUP, porque si esto fuera así RUP sería lo más obsoleto del mundo del software. Lo que hace realmente ágil la metodología Scrum es la visualización de la evolución del software que se está gestando; a medida que se observa su evolución, el dueño del software se dará cuenta, tempranamente, de los cambios y podrá solicitar al *team* el nuevo requerimiento; esto a diferencia de una metodología como RUP, donde los cambios se hacen al final, hay que estimar el control de cambio y el equipo de desarrollo se pone los guantes, coge el bisturí y milimétricamente empieza a realizar los cambios solicitados. Con RUP estos cambios pueden ser muy grandes y costará mucho tiempo, Scrum previene estas equivocaciones a grandes escalas haciendo cambios pequeños y fáciles de probar.

Algo muy importante que hace a Scrum una metodología ágil es las personas sobre los procesos y las negociaciones; el protocolo del papeleo quita mucho tiempo porque se divaga demasiado. El software se ve funcionando, no digo que se hagan las cosas sin planear o pensar, sino diciendo y haciendo de tal manera que podamos tener una base más real del comportamiento esperado.

Pasando al tema del tiempo empleado en el desarrollo, es de vital importancia que el equipo esté motivado, relajado pero comprometido y proactivo, es cuestión de suerte llegar a que un equipo se entienda y exista una buena relación laboral dentro de él.

## Los roles

Tenemos un *project manager*, un arquitecto de software y un grupo de desarrolladores, ¿quién se pide Scrum Master?; este tema de definir los roles cuando se está empezando es un poco complejo para el entendimiento de quienes se están desprendiendo de RUP o similares.

El rol de Scrum Master es el más importante, pues es el responsable de que exista armonía dentro del grupo, vela porque los impedimentos no aparezcan, gestiona, pregunta, consigue, llama, se mueve. Las empresas no ven con buenos ojos que una persona no tenga tareas definidas, es un tema complicado de abordar, pues entraríamos a evaluar las estructuras organizacionales, reporte de horas y demás; pero, retomando el tema de quién debería ser el Scrum Master, la experiencia me dice que debe ser una persona que esté el 100% del tiempo con el equipo y se ponga la camiseta del rol que representa.

El *product owner* “es el dueño de la aplicación”, dice la cartilla, pero en realidad no es tan así, las múltiples ocupaciones que una persona identificada como *product owner* tiene, no le permiten desarrollar el rol; una persona valiosa como esta, iniciando en un proyecto, va a estar atado a múltiples tareas porque tiene un valor agregado y es clave, pues llevará el proyecto al

éxito. Es necesario tomar del equipo a una persona que represente al *product owner* (un analista funcional) y se va a encargar de hacer ágil el tema de plasmar los “deseo”, “quiero” y “necesito” en el *product backlog*.

Los Scrum Team son quienes hacen realidad los deseos de los clientes; es muy importante que los integrantes de este equipo, sean personas realmente buenas técnicamente, las curvas de aprendizaje de un miembro dentro de un Team afectan directa o indirectamente a todos. Sería ideal que el nivel técnico sea de un desarrollador que domine buenas prácticas, patrones de diseño básicos, TDD, Mock, SQL, que conozca bien el *framework* (.Net, JEE, AngularJS, etc.) propuesto para trabajar y, obviamente, el *framework* Scrum. Pasando a las habilidades blandas, deben ser personas con sentido de pertenencia, con la motivación de querer desarrollar un software de calidad, no es necesario que sean los más extrovertidos pero sí que haya una persona dentro de cada quien dispuesta a dar una buena respuesta; la proactividad es el resultado del compromiso y la motivación.

Todo este conjunto de roles, para tener éxito, deben comportarse como una familia, las familias son aquellas que se apoyan unos a otros, en las que un miembro puede relevar a otro sin problema; somos personas, y como cualquier ser humano tenemos problemas y nos gobiernan las emociones, no siempre una persona estará dispuesta a trabajar al 100% de su capacidad y no por eso debemos llevarlo al límite, deben apoyarse como hermanos para que el objetivo en común se cumpla y sea la satisfacción del equipo ver el resultado por el cual se esforzaron, ese es el pago para un equipo comprometido.

## Kaizen

Es un término japonés que se divide en Kai (cambio) y Zen (bueno), buena mejora o cambio, ¿por qué kaizen?; es necesario hablar de este término porque se está poniendo de moda y porque todos

entramos creyendo que como ya estudiamos la metodología no vamos a aplicarla; tienen razón, pero hay que hacerlo sin afanes, paso a paso. Para los que están empezando a meterse en este mundo del agilísimo, se debe gatear y dominar el gateo para después caminar y poder correr. Me refiero a que las cosas aprendidas primero se deben dominar en su práctica, haciendo las cosas empezamos a dominarlas y Scrum es así; empezando nos vamos a equivocar, pero la reunión de retrospectiva es la celebración con la cual todo el equipo se sincera y empieza a mejorar su práctica. En los primeros tres *sprints* es donde la gráfica para de subir y empieza a mantenerse constante hasta el final del proyecto.

## Recomendaciones y advertencias

Scrum es un cambio de pensamiento y de forma de trabajo que da resultados diferentes a las metodologías tradicionales; para entender esto debe haber un cambio de pensamiento desde los que hacen la negociación, en ese momento hay que identificar quiénes son los realmente involucrados (usuarios líderes) y a los que les afecta el bolsillo (los ejecutivos), ellos serán los jueces que darán el aval de continuar con la práctica de Scrum, a ellos hay que involucrarlos porque son de “hasta no ver, no creer”. Hay otro riesgo y es la rotación del equipo; las personas que llegan a un equipo que ya tienen varios *sprints* de experiencia, son afortunados porque llegan a acoplarse a un grupo que ya funciona, sin embargo, genera retrasos en el desarrollo por la curva de aprendizaje; a pesar que Scrum no es muy amigo de la documentación, se deben tener instrumentos que permitan a un nuevo integrante, conocer cómo es el modus operandi del desarrollo.

Durante el desarrollo de software, se van generando prácticas que permiten mantener un buen desarrollo, son lineamientos con los que los miembros del equipo trabajan; es ágil realizar esta documentación porque es tiempo que no se le quita al equipo para que el nuevo integrante se acople. La presión no es buena, el equipo es

autogestionable, cada uno sabe qué hay que hacer. Medir el rendimiento de un colaborador es bueno como control, pero al primer bajón no se debe pedir cuentas de por qué no está rindiendo, de las horas que se siente a desarrollar un programador; el 40% es la productividad porque el análisis, aunque sea trabajo, toma tiempo y si hay una complicación es porque hay una razón válida; la presión no es buena, se debe confiar en el equipo.

La definición de las historias de usuario es importante al momento de definir las tareas; si la historia de usuario dice algo como “ver la historia de usuario HU\_0”, fallamos. Las historias de usuario son, en concreto, pedazos del software que no dependen de otras historias posteriores, deben ser tan claras y atómicas que, al momento de desglosarlas en tareas, no haya pendientes o dependencias, eso no es ágil. Otra característica de una historia de usuario es que es una y solo una, no porque una historia de usuario sea muy pequeña, otra historia la adopta; al momento de desarrollar las tareas se crearán confusiones dentro del equipo porque no va a ser lógico o no van a tener coherencia, hasta que se aclare con el *daily*, otra práctica que no es ágil.

El arte de estimar es algo que solo con la experiencia se adquiere, es importante que la contextualización de la historia de usuario sea clara porque de esta manera saldrán las tareas a desarrollar, si una historia está muy grande, por favor, exija que sea dividida en dos o más, las que sean necesarias.

A pesar de que la idea es desarrollar las tareas en el tiempo propuesto, es importante que nunca se sacrifique la calidad, sí, las pruebas de desarrollador y las pruebas unitarias. Es común que al ver que la entrega se aproxima, se dejen de lado las pruebas de calidad, puede que el colaborador sea el más experto de los expertos, pero las pruebas son inalienables o sino esa historia no se entrega, hasta que el equipo no esté seguro de lo que está entregando. Para empezar a desarrollar un *sprint backlog*, es necesario establecer el orden de prioridades del negocio; sí, puede que todo sea

importante, pero hay cosas urgentes también. El equipo debe seguir este orden, ir dejando historias concretas en el desarrollo, todos en conjunto para apoyar el desarrollo de la historia de usuario que toca en el momento; dado el caso que el equipo en el arte de la estimación de tareas se equivoque, pueda el Scrum master negociar el *review* y que el *product owner* no se lleve una mala imagen al momento de la entrega.

## CONCLUSIONES

Es importante pensar en cómo se aplican las diferentes practicas del desarrollo de software; para este caso, Scrum es mal interpretado y no es aplicado de la mejor manera por parte de muchas empresas de desarrollo de software, se cree que es ágil porque necesitamos software funcional de manera inmediata pero no se entiende de verdad cuál es el ciclo de desarrollo de software. El trabajador colombiano es particular en el mundo, no siempre es el más comprometido, aunque cuando quiere puede; hay distintas personalidades y a pesar de que se tiene un fin común, como todos los seres humanos pensamos diferente y es difícil seguir a un líder; en Colombia, la tendencia a ser rebeldes es común y siempre se cuestionan las decisiones que toman las cabezas de grupo. Para aplicar el marco ágil hay que apropiarlo, entenderlo y aceptar el rol que a cada miembro del equipo le corresponde; no hay que creer que, porque se lee bastante acerca de la metodología ya se domina por completo el ejercicio, hay que tomar las cosas con calma y aprender de a pocos, se debe tener en cuenta que en Colombia las metodologías ágiles apenas se están empezando a aplicar y que muchas empresas, donde los desarrolladores de software trabajan, son ajenas a estas nuevas prácticas, que aún son cerradas al cambio y a tratar de avanzar; proponer, muchas veces, no es bien visto.

Scrum funciona, pero hay que mostrar resultados y evangelizar, pues no es que tengamos las cosas en la mitad del tiempo, es saber que las cosas tendrán mucha más calidad, que el costo de soporte por errores será menor y que sí se tendrán desarrollos completos comparado con la metodología RUP, donde no hay espacio al cambio.

## APÉNDICE

Hay prácticas de Scrum que se aplican en Colombia donde se hacen híbridos, ante el cliente se presenta como RUP, pero el equipo hace Scrum; es complicado pero se puede, es difícil de manejar pero el que tiene que hacer el mayor trabajo es el gerente del proyecto o la persona que realiza la gestión ante el tercero; sería bueno que se empezara a vender más la metodología para empezar a desarrollarla plenamente y acogernos a sus bondades; esto permitiría que los nuevos ingenieros de desarrollo de software se enfocaran en prepararse en estas metodologías emergentes y no entrar a improvisar para estrellarse; promover la investigación por lo menos en el área de tecnología.

## REFERENCIAS

- [1] Scrum Institute (2013). *What is Scrum?* Recuperado de: [http://www.scrum-institute.org/What\\_Is\\_Scrum.php](http://www.scrum-institute.org/What_Is_Scrum.php)
- [2] Scrum Institute (2015). *Introduction to Scrum - A Real World Example (Case Study)*. Recuperado de: [http://www.scrum-institute.org/Introduction\\_to\\_Scrum\\_A\\_Real\\_World\\_Example.php](http://www.scrum-institute.org/Introduction_to_Scrum_A_Real_World_Example.php)
- [3] Scrum Institute (2013). *What Makes the Scrum Framework Succeed?* Recuperado de: [http://www.scrum-institute.org/What\\_Makes\\_the\\_Scrum\\_Framework\\_Succeed.php](http://www.scrum-institute.org/What_Makes_the_Scrum_Framework_Succeed.php)