PROPUESTA METODOLÓGICA PARA LA ESTIMACIÓN DE PROYECTOS GESTIONADOS MEDIANTE SCRUM, CON ENFOQUE A LA PEQUEÑA INDUSTRIA DEL SOFTWARE

METHODOLOGICAL PROPOSAL FOR ESTIMATING PROJECTS MANAGED IN SCRUM, FOCUSING ON SMALL BUSINESS SOFTWARE INDUSTRIES

José Guillermo Fierro Mendoza
Tecnológico Nacional de México en Celaya, México
guillermo.fierro@itcelaya.edu.mx

Francisco Gutiérrez Vera
Tecnológico Nacional de México en Celaya, México
francisco.gutierrez@itcelaya.edu.mx

Claudia Cristina Ortega González
Tecnológico Nacional de México en Celaya, México
claudia.ortega@itcelaya.edu.mx

Alejandro Saldaña Contreras
Tecnológico Nacional de México en Celaya, México
16031202@itcelaya.edu.mx

Recepción: 3/noviembre/2019 **Aceptación:** 29/noviembre/2019

Resumen

Típicamente, la estimación del esfuerzo en proyectos de software se ha realizado más basada en la experiencia que en un modelo producto de un análisis exhaustivo que permita desde el inicio del proyecto cuantificar el costo, el tiempo de desarrollo y el tamaño del equipo en la obtención del presupuesto del proyecto de software. Existen modelos maduros de estimación como IFPUG y Cosmic, sin embargo, no son utilizados en las pequeñas industrias dedicadas a la producción de software a la medida. Siendo Scrum un marco que se distingue porque los equipos deben ser auto-gestionados, para el negocio, un reto es la evaluación de la productividad del equipo de trabajo y de sus integrantes. Surgen también algunas dudas acerca de cómo integrar en el modelo ágil tareas relativas a la capacitación o la investigación

para la generación de un punto de historia en el que no se tiene experiencia. En este artículo se elabora una propuesta metodológica para la estimación temprana, una necesidad detectada en las pequeñas empresas dedicadas a la producción de software a la medida. Para la construcción de la propuesta se realizó un estudio de varias empresas y se conjugó con la información documental y la experiencia de los líderes de proyectos de estas empresas, se logró establecer esta propuesta metodológica en la que se plantea una estimación de dos niveles, sin afectar la agilidad del marco de desarrollo. El enfoque contempla la integración de componentes en la estimación de un conjunto de elementos como la velocidad de desarrollo, la estimación de la complejidad de la aplicación y la predictibilidad.
Palabras claves: Estimación de software, Puntos de historia, Scrum, Velocidad de desarrollo.

Abstract

Traditionally, the estimation of the effort in software projects has been made more based on experience than on a model product of an exhaustive analysis that allows from the beginning of the project to quantify the cost, development time and size of the equipment in obtaining of the software project budget. There are mature models of estimation such as IFPUG and Cosmic, however, which are not used in small industries dedicated to the production of custom software. Being Scrum a framework that distinguishes itself because the teams must be self-managed, for the business, a challenge is the evaluation of the productivity of the work team and its members. There are also some doubts about how to integrate tasks related to training or research in the agile model to generate a point of history in which there is no experience. This article elaborates a methodological proposal for early estimation, a need detected in small businesses dedicated to the production of custom software. For the construction of the proposal a study of several companies was carried out and combined with the documentary information and the experience of the project leaders of these companies, it was possible to establish this methodology in which an estimate of two levels is proposed, without affecting the agility of development framework. The approach contemplates the integration of components in the

estimation of a set of elements such as the speed of development, the estimation of the complexity of the application and the predictability.

***Keywords*:** Development velocity, Scrum, S*oftware Estimation,* Story Points,

1. Introducción

Scrum como marco de trabajo para el desarrollo y producción de software, ha venido a revolucionar la manera de organizarse en las empresas dedicadas a ello. Para su implementación hace énfasis en una serie de principios, roles, eventos, artefactos y reglas que en su conjunto conforman, una guía oficial [Schwaber, and Sutherland, 2017] para adecuar Scrum en las organizaciones que se dedican a la producción de software. Sin duda que ha aportado un beneficio al ciclo de vida del desarrollo de software al orientar la construcción del producto por iteraciones, en donde en cada una se obtiene un valor agregado del producto. Sin embargo, a través de la consulta con organizaciones dedicadas a la producción del software se ha encontrado que si se implementa Scrum siguiendo al pie de la letra las recomendaciones del marco, se presentan algunos vacíos, como el caso de la determinación o estimación del costo o tiempo total del producto de software terminado, o la gestión y por supuesto asignación de recursos para la gestión de los bugs detectados después del *sprint* (iteración) correspondiente a su liberación o durante su operación. Es cierto que Scrum pretende ser un marco flexible de tal manera que se puedan incorporar otros marcos ágiles como Kanban o herramientas de gestión. En este trabajo, se han desarrollado e implementado mecanismos para estimar el costo y estimación del esfuerzo para el desarrollo producto, más apegado a la realidad, considerando la posibilidad de correcciones y formulando un esquema para la determinación de la velocidad de desarrollo de los integrantes del equipo.

El primer paso en la estimación y planificación ágil es la creación de la lista del producto, donde se incluirán todas las características del producto a desarrollar. Se puede dividir en objetivos expresados como historias de usuario (*user stories*). Una historia es un requisito de negocio visto desde el punto de vista de un usuario y que aportan valor de negocio incremental e individual. Uno de los atributos de la lista de producto o lista de pendientes (*Backlog*), es la estimación. De acuerdo a la guía

Scrum [Schwaber, and Sutherland, 2017], un elemento de la lista de producto debe tener los atributos de descripción, prioridad (orden en la lista), estimación y valor. Esto es todo lo que la guía establece sobre estimación, no describe o explica que es y cómo realizarla [Nijland, Sjoerd, 2018]. También menciona el mismo autor que, sin embargo, la misma guía en la sección de *Sprint Planning* establece que el trabajo puede ser de tamaño variable o esfuerzo estimado. Desde esta perspectiva, Scrum es flexible respecto al método o técnica para determinación de la estimación. A través del tiempo se han desarrollado varios métodos para la estimación de software, entre los que destacan son los paramétricos como la estimación por puntos de función IFPUG-FPA, desarrollado a partir de funcionalidad entregadas al usuario a partir de aspectos técnicos independientes de la plataforma y etapas del ciclo de vida. Otros métodos similares, son NESMA, MkII, COSMIC y FisMA, sin embargo, como expresa [Quesada, Allue 2009], una característica de los proyectos ágiles consiste en construir una actividad adaptativa en lugar de predictiva. Por lo tanto, la estimación y planificación en un proyecto ágil difieren radicalmente de los proyectos tradicionales.

En un proyecto tradicional, el proceso es relativamente lineal: se estima el producto a desarrollar (generalmente haciendo un desglose por etapas); se planifica el desarrollo (con la consecuente transformación de lo que antes eran estimaciones en compromisos); y luego se procede a ejecutar el plan, que por supuesto debe cumplirse al pie de la letra. Cuando las cosas comienzan a atrasarse, cosa que ocurre generalmente, empiezan las complicaciones. Y esto generalmente se debe a que los equipos tienen diferentes habilidades que se manifiestan en la velocidad de desarrollo y por ende en la productividad. El mismo autor, y otros autores como [Nijland, Sjoerd, 2018], concuerdan que pocas organizaciones desearían invertir en un proyecto sin tener una idea aproximada de cuál será el costo o cuando llevará el desarrollo del proyecto. Por otra parte, [Menzinky and Sutherkand , 2019], manifiestan que quizá ya no hay “productos finales”, sino productos en continua evolución y mejora, lo que nos lleva a establecer el precepto que la gestión de proyectos ágil no se formula sobre la necesidad de anticipación, sino sobre la de adaptación continua, sin embargo es algo que les causa un poco de problemas a

los que venden software porque los clientes finales siguen esperando que se establezca un presupuesto y un compromiso de producto final. Y de ahí la importancia de tener mecanismos claros de control del ejercicio de los recursos, de tal manera que se pueda mantener siempre con claridad la rentabilidad del desarrollo por parte de la empresa desarrolladora y por otro lado el cliente pueda destinar y ejercer un presupuesto durante el tiempo que dure el proyecto. Entonces, bajo estas premisas, ¿Cuáles son las técnicas de estimación apropiadas en Scrum?, las metodologías ágiles implementan muchos conceptos de Lean, el sistema de producción de Toyota, uno de ellos es *small batch sizes*, significando con ello, producir valor en lotes pequeños [Quesada, Allue, 2009], de ahí se deriva el concepto de historia de usuario.

Una de las técnicas más conocidas para estimar en Scrum es, *planning Póker*, que utiliza una medida de tamaño arbitrario para indicar la complejidad en tamaño de una historia de usuario y cuyo valor sólo tiene sentido para el equipo de desarrollo en cuestión. Los puntos de historia (*story points*) no pueden compararse entre diferentes equipos y a veces ni siquiera entre diferentes proyectos, lo que hace difícil la parametrización. Lo único que indican es el tamaño relativo que tiene cada funcionalidad de la lista del producto respecto a las demás. [Pérez, Alejandro, 2017], incorpora tres métodos adicionales, Delphi, PERT y T-Shirt *sizing*. El objetivo principal del método Delphi es llegar a un consenso basado en la discusión entre expertos mediante un proceso iterativo, de hecho, se considera que *planning póke*r está basado en Delphi. El método PERT contempla tres valores, uno para el peor caso o pesimista (P), otro para el mejor caso u optimista (O) y uno para la estimación más probable (M) y aplica una fórmula para determinar el valor de la estimación más real. En el método T-Shirt cada miembro del equipo debe indicar si considera que la tarea a estimar corresponde a un valor del conjunto: XS (Extra Small), S (Small), M (Medium), L (Large), XL (extra-large) o XXL (Double Extra-Large), eliminando necesidad de un modelo numérico, se considera que el equipo puede hacer más fácil la tarea al hacer la estimación de una manera más abstracta.

Entre las consideraciones más importantes para utilizar alguna de estas técnicas, [Garzás Javier, 2015], sugiere tres niveles de planificación utilizados en un proyecto

de software ágil. El primer nivel corresponde al *roadmap* u hoja de ruta, el segundo nivel corresponde a la planificación de un *release* (una versión del producto de software) y el tercer nivel corresponde a la planificación de cada sprint o iteración. Desde su perspectiva sugiere utilizar una técnica de estimación de acuerdo al nivel de planificación, como se muestra en la tabla 1.

Tabla1 Comparativa de los tres tipos de planificación ágiles.

Nivel de planificación	Horizonte temporal	Cómo se especifican los requisitos	Nivel de Abstracción	Método de estimación
Roadmap	Semestres e incluso años	Temas	Estratégico	T-Shirt
Release	Meses	Épicas o historias de usuario	Táctico	Puntos de historia u horas
Sprint	Semanas	Tareas	Operativo	Horas o días ideales

Fuente. Basado en [Garzás Javier, 2015]

Un modelo interesante para estimar utilizando dos niveles diferentes, lo presenta [Cohn, Mike 2018], en su propuesta sugiere que se estime al definir el backlog y posteriormente el sprint backlog.

En este artículo se presenta una propuesta metodológica con los elementos esenciales para la integración de un mecanismo de estimación para el desarrollo de un producto de software, se integran dos niveles en la forma de correlacionar los puntos de historia con respecto a las horas de trabajo reales invertidas, de tal manera que también se puedan determinar métricas de productividad del equipo de desarrollo. La propuesta incluye recomendaciones por parte de expertos en la gestión de desarrollos de software de diez empresas y que viven la implementación de una metodología ágil en su organización. Se han incorporado aspectos que generalmente son omitidos por los modelos existentes, como es el concepto de la predicibilidad y rendimiento, que son significativos al momento de llevar a cabo la construcción del producto real software. Para el cálculo de estas métricas se usarán las expresiones sugeridas por [Fuqua, Andrew, 2013].

2. Métodos

Para elaborar esta propuesta se estudió en detalle los diferentes métodos y técnicas presentadas, posteriormente se realizó un diagnóstico mediante un

instrumento aplicado a diez empresas clasificadas como pequeña industria, dedicadas a la producción de software a la medida. Las empresas estudiadas se encuentran siete en la ciudad de Celaya, Guanajuato, una en la ciudad de Guanajuato capital, otra ubicada en Dearborn, MI, US y otra en la ciudad de Querétaro. El eje de análisis que se tuvo como base para el diagnóstico radica en el proceso de gestión, en particular el marco de desarrollo utilizado y los resultados o impacto de la técnica de estimación utilizada para la elaboración de un producto final. Se hizo un estudio de investigación documental de los métodos y técnicas que se recomiendan actualmente y se realizó un diagnóstico, en las empresas, en base a un instrumento tipo encuesta elaborado para detectar el proceso de desarrollo, las técnicas empleadas y la percepción o retroalimentación relacionada con el grado de certidumbre que les arroja su estimación inicial respecto a la utilización de recursos y tiempo de desarrollo empleado, que al final se refleja el costo de desarrollo del producto final. Mediante un análisis comparativo se determinaron las empresas que mejores resultados tienen y se realizó una entrevista directa para conocer las particularidades de sus métodos. Se construyó la propuesta metodológica, considerando los mejores aspectos utilizados por estas empresas y otros de las referencias teóricas. Finalmente, para validar la propuesta que se plantea aquí, se realizó mediante un ejercicio académico en un curso de ingeniería de software, con un equipo de trabajo colaborativo de cuatro integrantes.

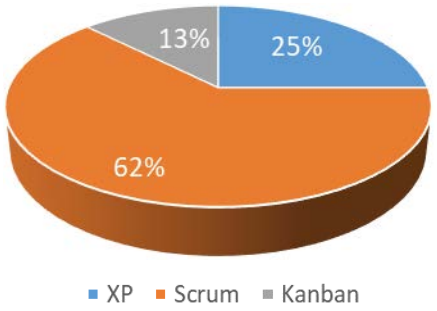
El instrumento tuvo el objetivo de determinar, qué marco o metodología de desarrollo se utiliza actualmente en las empresas para la construcción de software con la finalidad de reconocer la adopción de los métodos ágiles y su forma de implementación al interior de sus equipos de trabajo, así como los métodos utilizados para la estimación y la problemática o resultados que han encontrado en su implementación, con la finalidad de detectar elementos que fortalecieran la propuesta.

3. Resultados

El diagnóstico

El diagnóstico arrojó los siguientes resultados:

a) Respecto al marco de trabajo se obtuvo que el 100% de las organizaciones utilizan un marco de trabajo ágil, ya sea Scrum o XP. Este dato indica que las metodologías ágiles han permeado a la industria del software. La figura 1 muestra los porcentajes de utilización. Incluso algunas usan Scrum y XP y otra usa la integración de Scrum con Kanban.



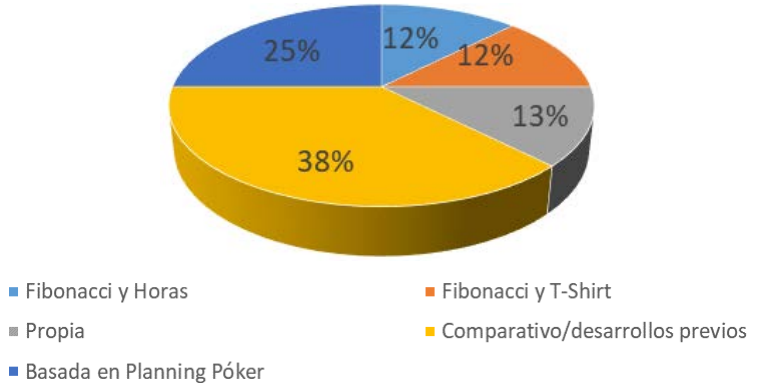
Fuente. Elaboración propia.

Figura 1 Distribución de marcos de trabajo en las empresas estudiadas.

b) En lo referente al método de estimación. De las diez empresas estudiadas sólo dos tienen implementados dos niveles de estimación y corresponden a las que cuentan con un mecanismo de análisis de productividad de sus desarrolladores, lo que les permite la posibilidad de encontrar sus áreas de oportunidad y mejora. En la figura 2 se muestra el grafico con los resultados referentes a este rubro. Una de las empresas la estimación por planning póker para el backlog y para sprint usa estimación por horas basada en la experiencia del programador. Otra empresa estima primero usando Fibonacci, para el *backlog* y en el *sprint* usan T-Shirt (Estimación relativa por categorías, similar a la categorización de tallas de ropa: S, M, L, XL, XXL). La mayor parte usa la comparativa con sistemas anteriores.

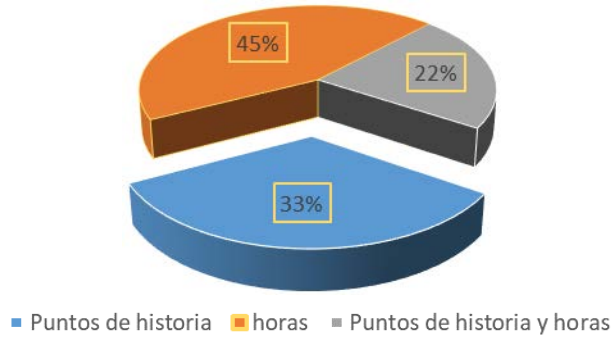
c) Unidades de estimación utilizadas. Como se aprecia en la figura 3, las unidades de estimación que se utilizan en mayor proporción, son los puntos de historia, sin embargo, un par de empresas utilizan una combinación de puntos de historia y horas y esto les ha dado buenos resultados. Un porcentaje menor de empresas estiman sólo en horas. Ante la pregunta ¿Por qué no utilizan una unidad como puntos de función que es estándar y

replicable?, la respuesta estuvo relacionada con la dificultad y el tiempo que demanda el método. Siendo más simple el uso de puntos de historia. Adicionalmente los puntos de historia propician la participación de todo el equipo de desarrollo como establece Scrum.



Fuente. Elaboración propia.

Figura 2 Métodos que utilizan las empresas en su proceso de estimación.

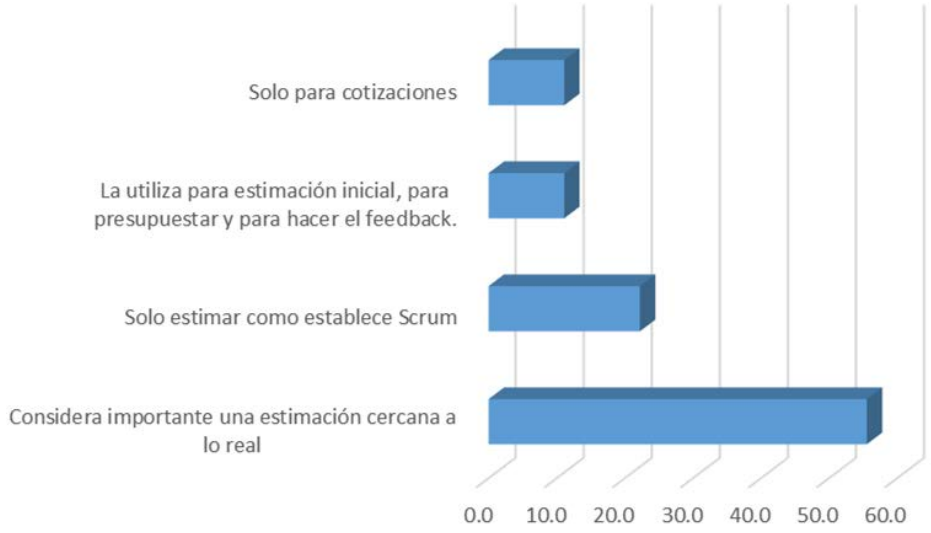


Fuente. Elaboración propia.

Figura 3. Unidades que las empresas usan en sus métodos de estimación.

d) Importancia de la precisión de la estimación. En la figura 4 se muestra la perspectiva que las empresas tienen respecto a la importancia de hacer buenas estimaciones. La mayoría considera que es importante lograr hacer una estimación lo más aproximada a lo real y tratan de alcanzar una certidumbre mayor del 90%. De las empresas estudiadas sólo una tiene un mecanismo formal para realizar las estimaciones y las demás basan su estimación en su personal experto, que en la mayoría de las empresas son pocos. Las que solo estiman de acuerdo Scrum, entienden que el marco no

exige que la estimación sea tan exacta, sino solo un elemento que guíe el desarrollo, entendiendo en este caso que la clave está en negociar con el cliente y que los recursos pueden cambiar, argumentando las características flexibles y adaptables del método, cosa que no siempre es fácil de manejar. A la pregunta ¿su organización cuenta con un mecanismo de feedback que les ayuda a que las estimaciones se vuelvan progresivamente más cercanas a lo real?, el 60% respondió que sería deseable contar con uno. Lo que motivó a continuar con este trabajo.



Fuente. Elaboración propia.

Figura 4 Perspectiva de las empresas respecto al uso e importancia de la estimación.

Elementos de la propuesta.

Para definir la metodología, utilizamos la retroalimentación proporcionada por tres empresas que han implementado un proceso de Scrum muy apegado a los lineamientos del marco y que adicionalmente han incorporado algunas mejoras a través de la experiencia incorporando factores clave de un buen proceso de estimación que garantice la conclusión exitosa de proyectos de acuerdo a lo pactado con el cliente:

a) Establecer niveles de estimación. El primer nivel de estimación se debe hacer para las historias de usuario a nivel del backlog en una reunión preliminar. Es recomendable que este ejercicio se haga antes de la reunión de

planeación. La técnica utilizada puede ser Póker Planning, T-Shirt u ordenamiento, ya que proporcionan buenos resultados respecto a la complejidad y las unidades utilizadas son puntos de historia (phu). El segundo nivel de estimación se deberá hacer al iniciar el Sprint. Se recomienda se usen horas para estimar, ya que se hace por tarea y solamente participa el responsable o se puede hacer con ayuda de un experto para mejorar el resultado.

El tamaño de la historia debe ser lo más pequeño posible. La clave para estimar está aquí. Se pueden establecer parámetros que ayuden para tener una referencia base de estimación, como tomar en cuenta la complejidad ciclomática para métodos, número de componentes en la interface gráfica, complejidad de las consultas en los accesos a datos. En caso de desarrollo web, si son páginas estáticas o dinámicas, el tipo de elementos que se incorpora la página, la facilidad de configuración de los elementos, etc.

b) Hay coincidencia en que la estimación inicial la realicen entre los vendedores, el Scrum Master y el equipo de desarrollo. El vendedor debe tener experiencia y conocer los estándares de la empresa. El Scrum Master es una persona con experiencia y conocimiento de su equipo de desarrollo. Es deseable y lo sugiere el Director de Software de una empresa que tiene certificación CMMI nivel tres y que tiene declarado su proceso de desarrollo basado en Scrum, mantener un catálogo o inventario de *features* (características del producto), esto agilizará bastante el proceso de estimación.

c) Es importante contar en el equipo de trabajo (team development) con una especie de catálogo o relación de *features* (características del producto) en las que se acuerde el nivel de complejidad, a fin de disminuir el tiempo que tome el proceso de estimación. Se pueden tomar en cuenta la complejidad de las interfaces gráficas al usuario (GUIs), de las clases para la implementación del negocio (simples, medianas o complejas), de los métodos (simples, métodos complejos), los controladores, las clases para la gestión de datos y los elementos que hayamos utilizado en nuestros

proyectos históricos. Si tenemos varios equipos de trabajo esto ayudará en las métricas del conocimiento de los equipos y por otro lado adicionará un elemento de certidumbre a las estimaciones ya que como dice la guía de Scrum [Schwaber and Sutherland, 2017, pág. 15], relacionado con el seguimiento hacia el logro del objetivo: “solamente lo que ya ha sucedido puede ser utilizado para en la toma de decisiones”.

d) Considerar la verificación de las estimaciones por un desarrollador par.
   Determinación de la velocidad de desarrollo del equipo e individual, tomando en cuenta la predicibilidad del equipo de trabajo. Para calcularla es necesario hacer el cálculo de historias de usuario comprometidas, historias de usuario que se entregaron, historias de usuario que se no entregaron y el número de integrantes.
e) Determinación del costo estimado. Basado en la cantidad de *sprints* determinado en el *backlog* y los sueldos de los desarrolladores participantes en el equipo de trabajo, se puede hacer una estimación del costo del proyecto.
f) Evaluación de la productividad y mejora. Al finalizar el *sprint* o el proyecto se puede recalcular la velocidad del equipo.
g) Recalibración de la estimación, es deseable al finalizar cada *sprint* recalibrar las estimaciones.

Validación de la propuesta

La validación de esta propuesta se hizo con un equipo de estudiantes de sexto, séptimo y octavo semestres con diferentes niveles de programación. El equipo destinó dos horas de trabajo diario para el proyecto e inicialmente estimó un sprint de una semana para el desarrollo. Se hizo el ejercicio suponiendo que los programadores recibirían el pago, ajustando el sueldo con un factor de 0.25 equivalente al número de horas destinadas al proyecto. El costo por día del equipo se calculó dividiendo el total del mes por 20 días hábiles, resultando 637.5 pesos. La configuración de las características del equipo se muestra en la tabla 2. Es

importante tomar en cuenta que el presupuesto del proyecto cambia de acuerdo al número de integrantes del equipo.

El equipo se dio a la tarea de determinar las historias de usuario y estimar la velocidad, así como el tamaño del sprint.

Tabla 2 Características del equipo de trabajo Scrum.

Usuario	Experiencia	Sueldo proyecto (pesos)	sueldo mensual (pesos)	Velocidad estimada (phu /Sprint)
Developer 1	Alta	4500	18000	10
Developer 2	Media	3250	13000	7
Developer 3	Alta	2500	10000	10
Developer 4	Media	2500	10000	8
	Por día	**637.5**		**4.375**
	del equipo	**12750**		**35**

Fuente: elaboración propia.

Entre más integrantes tenga un equipo mayor será capacidad de entrega de historias de usuario y el proyecto se podrá entregar en menor tiempo. El equipo estimó un *sprint* para concluir las historias de usuario con un costo estimado de 6375.0 peso como se indica en la tabla 3.

Sabiendo que de la estimación inicial resultó un sprint, realizaron la estimación en horas para las tareas del *sprint*, como se muestra en la tabla 4.

Tabla 3 Estimación inicial del backlog.

Historia de usuario	Estimación (phu)
Como usuario deseo visualizar un listado de problemas	1
Filtrar los problemas por categoría	2
Visualizar los códigos de los problemas	3
Probar la solución de un problema seleccionado	8
Mostrar los casos de prueba de cada problema	5
Total de phu	**19**
Días del Sprint	**10**
Costo estimado inicial	**637.5**
Tiempo estimado para hacer el proyecto (sprint)	**1**

Fuente: elaboración propia.

Tabla 4 Estimación del sprint 1 y utilización real.

Historia de usuario	Phu	Horas		
Como usuario deseo visualizar un listado de problemas	1	Estimación	Real	
Diseño de interfaz (GUI)		0.25	**0.5**	
Codificación		0.5	**1.25**	
Pruebas		0.25	**0.25**	
Total de hrs		**1.5**	**2.5**	Total
Filtrar los problemas por categoría	2	Estimación	Real	
Análisis		0.25	**0.25**	
Diseño de interfaz (GUI)		0.25	**0.15**	
Codificación		0.25	**0.35**	
Pruebas		0.25	**0.25**	
Total de hrs		**1**	**1**	Total
Visualizar los códigos de los problemas	3	Estimación	Real	
Análisis		0.25	**0.25**	
Diseño de interfaz (GUI)		0.5	**0.75**	
Codificación		1	**1.5**	
Pruebas		0.25	**0.5**	
Total de hrs		**2**	**3**	Total
Probar la solución de un problema seleccionado	8	Estimación	Real	
Análisis		1	**1.25**	
Diseño de interfaz (GUI)		0.5	**1**	
Diseño de la lógica de negocios (clases)		0.5	**1.25**	
Codificación		1.5	**2.5**	
Pruebas		1	**1**	
Total de hrs		**4.5**	**7**	Total
Mostrar los casos de prueba de cada problema	5	Estimación	Real	
Análisis		0.5	**0.5**	
Diseño de interfaz (GUI)		0.25	**0.5**	
Diseño de la lógica de negocios (clases)		0.25	**0.25**	
Codificación		0.75	**1.25**	
Pruebas		0.25	**0.5**	
Total de hrs		**2**	**3**	Total
Total	**19**	**11**	**16.5**	
Días del Sprint	**10**			
Costo estimado inicial	**0**			

Al inicio el equipo estimó su velocidad ya que no habían hecho anteriormente un ejercicio para determinarla. Este factor puede producir incertidumbre en el cálculo para la cotización del presupuesto del proyecto, ya que al finalizar el primer *sprint* se quedaron 2 tareas por realizar y en el *testing* de aceptación, realizado por otro grupo de estudiantes con conocimiento sobre el problema, resultaron 2 *issues*, para las historias 4 y 5, por lo que se ajustó la planeación a otro sprint. Si se hubiera

cotizado el proyecto con la estimación inicial, hubiera resultado en una pérdida como se muestra en la tabla 5. Por esto es importante introducir en el modelo el factor de predicibilidad. Este factor de predicibilidad para el cálculo de estimación del costo de desarrollo puede obtenerse por las razones de proporción siguientes:

$$Proporción\ de\ historias = comprometidas/entregadas$$

$$Proporción\ de\ puntos\ de\ historia = comprometidos/entregados$$

$$Predicibilidad = Proporción\ de\ historias/proporción\ de\ puntos\ de\ historia$$

Se obtuvieron los resultados mostrados en la tabla 5.

Tabla 5 Determinación de la predicibilidad.

Historias comprometidas	Historias entregadas	relación
5	3	1.67
puntos de historia comprometidos	punto de historia entregadas	relación
19	16	1.1875
	Factor de Predicibilidad	1.40

Fuente: elaboración propia.

Entonces, cuando se conoce este factor se aplica en la determinación del esfuerzo en *sprints*. Por lo que para este equipo de trabajo puede reconsiderarse que el número de sprint de desarrollo será de dos, lo que afecta el costo del proyecto, pero es más certero, ya que efectivamente después de dos sprint el proyecto quedó terminado. Para los siguientes proyectos la velocidad de desarrollo del equipo para los siguientes proyectos se usará el promedio de la velocidad obtenida.

El mismo proyecto se realizó con otros equipos de trabajo, en condiciones similares, encontrando que, efectivamente, cada equipo tiene su propio punto de referencia, respecto a la estimación de los puntos de historia, Por lo tanto, tendrán una estimación diferente con puntos de historia, un equipo podrá estimar 30 puntos de historia, mientras que otro 38. Sin embargo, la planificación real deberá apuntar a un tiempo de entrega y la productividad podrá evaluarse por el número de historias de usuario que entrega cada equipo en un determinado *sprint*. Para tener estimaciones consolidadas, es decir, que el equipo entregue las historias de usuario como se estimaron inicialmente, el registro histórico de datos y su análisis.

4. Discusión

Los resultados arrojan que sólo dos empresas de las estudiadas cuentan con un mecanismo que les permita estimar el presupuesto y esfuerzo de un proyecto de software. La mayoría basa la estimación en la experiencia y comparativo con otros productos de software, pero sin dejar registros históricos o evidencia que les permita aprender de los proyectos anteriores. Estandarizar la forma de medir la complejidad del software para predecir un presupuesto con una certidumbre mayor a 90 porciento, no es sencillo ya que cada proyecto tiene características diferentes. En los marcos ágiles no es una preocupación fundamental, pero en la vida real, sigue siendo una exigencia tener que dar un plazo de entrega y un presupuesto inicial, por lo que mayoría desearía una herramienta de estimación inicial que le proporcione mayor certeza en sus estimaciones y además ayude en el autoconocimiento de los integrantes del equipo con la finalidad de tener elementos de mejora, considerando la velocidad de desarrollo.

La metodología planteada toma lo mejor de las experiencias compartidas las empresas entrevistadas y estrategias documentadas, así como la retroalimentación obtenidas en el trabajo académico, integrando una propuesta que puede funcionar bien en otras empresas y que permite que sus equipos de trabajo conozcan y mejoren cada vez más sus habilidades de desarrollo. Esto permitirá a las empresas ir incrementando el personal capacitado para realizar buenas estimaciones. La incorporación de los componentes planteados no quita agilidad al marco de trabajo y al contrario refuerza el auto conocimiento tanto de los integrantes como del equipo mismo ya que el equipo va mejorando y haciendo más precisas cada vez sus estimaciones, reduciendo el optimismo o pesimismo e incrementando sus métricas de desarrollo y su predicibilidad del esfuerzo y costo de un proyecto. Esto favorece estimación que se pueda dar a un cliente del costo de un producto.

De los resultados de la investigación pudimos detectar que hay pocas referencias aún sobre el tema de la estimación por niveles, sin embargo, siempre ha sido una necesidad de la industria del software, realizar estimaciones de esfuerzo y costo tempranas y eso lo pudimos corroborar con los resultados de las entrevistas con las

empresas, la mayoría de los clientes necesita un presupuesto con una certeza mayor al 90% para tomar su decisión.

Entre los factores clave que fueron utilizados para la implementación de la propuesta se debe considerar si el equipo de trabajo es nuevo ya que es altamente recomendable entrar en un proceso de entrenamiento donde exclusivamente se trabaje en ganar habilidad en las técnicas de estimación.

Entre los trabajos futuros, consideramos importante realizar un estudio con más empresas de diversas ciudades a fin de regionalizar el modelo y por otra parte, automatizar el flujo del modelo manteniendo un catálogo de historias de usuario estandarizado a los equipos de la empresa, contemplando registro histórico de las estimaciones y los reales, y la evolución de la velocidad de los desarrolladores. Un trabajo de laboratorio interesante y un análisis comparativo entre las diferentes técnicas empleadas y los resultados de los recursos empleados.

5. Bibliografía y Referencias

[1] Abualkishik, Abedallah & Lavazza, Luigi. (2018). IFPUG Function Points to COSMIC Function Points convertibility: A fine-grained statistical approach. Information and Software. Information & Software Technology. Vol 97, Mayo de 2018, páginas 179-191.

[2] Cohn, Mike, (2018). Why Agile Teams Should Estimate at Two Different Levels, septiembre de 2018: https://www.mountaingoat software.com/blog/why-agile-teams-should-estimate-at-two-different-levels.

[3] Fuqua Andrew. (2013). Three Approaches to Estimating the Impact of Holidays and Time Off on Velocity. 2019, noviembre 10: https://www.leadingagile.com /2013/07/agile-health-metrics-for-predictability/.

[4] Garzás Javier, (2017). Peopleware y Equipos Ágiles con prácticas de management 3.0. Editorial 233 grados de TI, 2017. ISBN: 978-84-697-7450-2.

[5] Cohn, Mike, (2019). Three Approaches to Estimating the Impact of Holidays and Time Off on Velocity. 2019, noviembre 10: https://www.mountaingoat software.com/blog/three-approaches-to-estimating-impact-of-holidays-and-time-off-on-velocity.

[6] Garzás Javier, (2015). Como sobrevivir … a la planificación de un proyecto ágil. Biblioteca de supervivencia tecnológica. Editorial 233 grados de TI.

[7] Menzinsky, A., López, G., Palacio, J. (2019). Scrum Manager: Temario Troncal I Versión 2.6.1 – enero 2019: https://www.scrummanager.net/files/scrum_manager.pdf.

[8] Nijland, Sjoerd. (2018). A deeper Understanding on estimation. Retrieved 1 november, 2019: https://medium.com/serious-scrum/estimation-103de626551e.

[9] Pérez, A. (2017). 4 Técnicas para estimar: PERT,Delphi,Planning Poker,T-shirt. 1 November, 2019, URL: http://www.ceolevel.com/4-tecnicas-para-estimar-pert-delphi-planning-poker-tshirt.

[10] Quesada, Allue Xavier. (2009). Introducción a la estimación y planificación ágil. 1 November, 2019, URL: https://proyectosagiles.org/2009/06/08/introduccion-estimacion-planificacion-agil/.

[11] Schwaber, Ken and Sutherland, Jeff (2017). The Scrum Guide TM the definitive guide: the rules of the game, november 2017: https://www.scrumguides.org /docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf.

[12] Sidky, A. & Gaafar, A. (2014). The mindset behind estimating and planning for agile. Paper presented at PMI® Global Congress 2014—EMEA, Dubai, United Arab Emirates. Newtown Square, PA: Project Management Institute.

[13] Valdes Souto, Francisco. (2012). Estimación de proyectos de Software, un problema una Solución. Revista Software Guru. No. 32: https://sg.com.mx /revista/32/estimacion-proyectos-software.

[14] Valdes Souto, Francisco. (2012b). Modelo EPEI para la estimación de proyectos de Sofware (parte 2). Revista Software Guru No. 32: https://sg.com.mx /revista/33/estimacion-proyectos-software-parte-2.

[15] Vila Grau, Juan Luis. (2019, 19 de octubre). Cómo estimar el backlog de un producto de software: https://managementplaza.es/blog/como-estimar-el-backlog-de-producto/.