Universidad Autónoma de Baja California

Maestría y Doctorado en Ciencias e Ingeniería

"Estimación de Historias de Usuario en Scrum a través de un Modelo basado en Descomposición de la Complejidad "

Tesis

Para obtener el grado de:

Maestro en Ciencias

Presenta: Janeth Patricia López Martínez

Director de Tesis: Dr. Guillermo Licea Sandoval

Co-Director: Dr. J. Reyes Juárez Ramírez

Noviembre, 2017

Universidad Autónoma de Baja California

FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

COORDINACIÓN DE POSGRADO E INVESTIGACIÓN

FOLIO No. 224

Tijuana, B. C., a 26 de octubre de 2017

C. Janeth Patricia López Martínez
Pasante de: Maestro en Ciencias
Presente

El tema de trabajo y/o tesis para su examen profesional, en la Opción TESIS

Es propuesto, por los C. Dres. Guillermo Licea Sandoval y J. Reyes Juárez Ramírez

Quienes serán los responsables de la calidad del trabajo que usted presente, referido al tema: "Estimación de Historias de Usuario en Scrum a través de un Modelo basado en Descomposición de la Complejidad"

el cual deberá usted desarrollar, de acuerdo con el siguiente orden:

I.- INTRODUCCIÓN
II.- ESTADO DEL ARTE
III.- FACTORES DE LA ESTIMACION Y MODELO PROPUESTO
IV.- EXPERIMENTO
V.- CONCLUSIONES, APORTACIONES Y TRABAJO FUTURO

UNIVERSIDAD AUTONOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS QUIMICAS E INGENIERIA

Dr. Guillermo Licea Sandoval
Director de Tesis

Dr. José Luis González Vázquez
Sub-Director Secretario

Dr. J. Reyes Juárez Ramírez
Co-Director de Tesis

Dr. Luis Enrique Palafox Maestre
Director

Dedicatoria

A Mis Padres: Trinidad y Benjamin

Por haberme dado la vida, las ganas de ser mejos cada día brincando

cualquier obstaculo que se presentaba y por la motivación

y apoyo que han brindado siempre.

A Mi Esposo: Alan David

Quien me apoyó para retomar esta carrera de aprendizaje

a no desistir aún cuando las fuerzas se agotaban

y se complicaban las cosas estando lejos de nuestra casa.

A Mis Hijos: Leonardo y Jonathan

Quienes son mi razón de existir y quienes dan sentido a mi vida

Agradecimientos

Me gustaria incluir en estos agradecimientos primeramente a Dios que me ha bendecido toda mi vida así mismo todas las personas que me dieron animos a la realización de este proyecto y que día con día han estado motivandome a seguir adelante y concluirlo.

En especial a mi familia, mi madre, padre y hermanos quienes siempre me han apoyado en mis decisiones, a mi esposo Alan David por darme el apoyo incondicional cada día y su ayuda cuando se me bloqueaban las ideas y encender esa luz para que se iluminaran nuevamente, a mis hijos Leonardo y Jonathan por ser mi razón de existir ¡Gracias por ser parte de mi vida !

A la Universidad Autonoma de Baja California por abrirme sus puertas y darme la oportunidad de retomar mi carrera, al Consejo Nacional de Ciencia y Tecnología (CONACYT), por el financiamiento brindado durante la realización de esta maestría. ¡Gracias!

A los miembros del Comité, Director de tesis Dr. Guillermo Licea Sandoval Co-Director Dr. J.Reyes Juárez Ramírez y los sinodales Dr. Leocundo Aguilar, Dr. Luis Guillermo Gracias por sus valiosas observaciones y comentarios sobre este documento.

Dr J. Reyes gracias por su apoyo para sacudir un poco la "polillaz el apoyo brindado durante la estancia en la cuidad así como tambien el apoyo para asistir a los congresos nacionales e internacionales.

Al grupo de estudiantes que cursaron la materia "Aseguramiento de la Calidad de Software" e Ïngenieria de Requerimientos", por su apoyo en la fase de experimentación realizada en nuestra propuesta de tesis y estar día con día en nuestras Daily Meeting y Sprint Planning a veces sin comer especialmente Arvizu, Roberto, Leonardo y a Ulises Sandoval quien esta apoyando en la realización del software.

Resumen

Planning Poker es una técnica de estimación mediante cartas basadas en la secuencia Fibonacci. Esta técnica ofrece muchas ventajas, sin embargo, no siempre es eficiente ya que las estimaciones son realizadas en base a criterio de expertos, lo que resulta confuso en cuanto a los factores que se consideran para realizar las estimaciones.

Este trabajo propone un modelo de conocimiento para determinar dos de los aspectos más importantes de la estimación, la complejidad y la importancia de las historias de usuarios basadas en el contexto de Planning Poker en Scrum. El objetivo de este trabajo es modelar la naturaleza compleja de la estimación de la historia del usuario para facilitar esta tarea a los desarrolladores principiantes.

Se construyó una red Bayesiana basada en el modelo propuesto que considera la complejidad e importancia de una historia de usuario.

Las Estimaciones de desarrolladores novatos y profesionales fueron sometidas a pruebas de correlación para validar la aplicabilidad del modelo propuesto. Con base en los resultados, el modelo propuesto alcanza un mayor grado de correlación con la estimación de profesionales que de desarrolladores novatos, lo que significa que el modelo incluye factores considerados por profesionales de empresas desarrollo de software.

Esta propuesta es útil para guiar a los desarrolladores principiantes a evaluar la complejidad y la importancia de las historias de los usuarios a través de preguntas. Los desarrolladores principiantes podrían usar la propuesta para estimar mejor que al utilizar tradicional Planning Poker.

Abstract

Planning Poker is a complexity estimation technique for user stories through cards. This technique offers many advantages; however, it is not efficient enough as estimations are based on experts criteria, which is fuzzy regarding what factors are considered for estimation. This paper proposes a knowledge model to determine two of the most important aspects of estimation, the complexity, and importance of user stories based on Planning Poker in Scrum context. The goal of this work is to model the complex nature of User Story estimation to facilitate this task to novice developers. A Bayesian network was built based on the proposed model that considers the complexity and importance of a user story. Students and professionals submitted their estimates to correlation tests to validate the applicability of the proposed model. Based on the results, the proposed model achieves a greater degree of correlation with the estimation from professionals than students, which means that the model includes factors considered in real world applications. This proposal could be useful for guiding novice developers to evaluate the complexity and importance of user stories through questions. Students could use the proposal to estimate rather than the traditional Planning Poker.

Índice general

Índice de figuras

Índice de tablas

Lista de acrónimos

No.	Acronimo	Significado
1	AIE	Número de archivos de interfaz externos
2	ALI	Número de archivos lógicos internos
3	CE	Número de salidas externas
4	COCOMO	Modelo constructivo de costos
5	EE	Número de entradas externas
6	FIS	Modelo de Inferencia Difusa
7	HU	Historia de Usuario
8	IS	Ingeniería de Software
9	MCD	Mapas Cognitivos Difusos
10	MPS	Mejora de Procesos de Software
11	PP	Planning Poker
12	PO	Product Owner
13	PF	Puntos Función
14	RB	Red Bayesiana
15	ROI	Retorno de Inversión
16	SM	Scrum Master
17	SP	Puntos de Historia
18	TCP	Tablas de Probabilidad Condicional
19	UCP	Modelo de Casos de Uso

Capítulo 1

Introducción

Este capítulo explica los antecedentes del problema que se resuelve en este proyecto; además, se plantean los objetivos y las preguntas de investigación que se derivan. Al finalizar se describen los beneficios de esta investigación.

1.1. Antecedentes

El Software se ha convertido en la actualidad en uno de los principales objetivos estratégicos de las organizaciones debido a que los procesos más importantes dependen del funcionamiento del software (Cendejas Valdéz et al., 2014). Por tal motivo, la industria y los investigadores interesados en la Ingeniería de Software (IS) han expresado especial interés en resolver los problemas con la Mejora de Procesos de Software (MPS).

Ante la necesidad de mejorar el proceso de desarrollo de software, se importaron fundamentos de metodologías existentes en otras áreas y se adaptaron a la IS (Rueda Gutierrez, 2010). Así, diferentes modelos y metodologías surgieron como herramientas de apoyo para el desarrollo de software (Cendejas Valdéz et al., 2014). El primer modelo de desarrollo de software fue el modelo de cascada; posteriormente se han presentado muchos modelos de desarrollo de software tales como: el modelo de prototipos, el modelo incremental, el modelo en espiral, los modelos formales, los modelos basados en

componentes, el modelo RUP, los modelos ágiles, entre otros (Gao, 2010).

El desarrollo de software ágil se refiere a un grupo de metodologías basadas en un proceso iterativo donde los requerimientos y soluciones evolucionan a través de la colaboración en equipos multifuncionales auto-organizados (Cendejas Valdéz et al., 2014). Como todas las disciplinas de la ingeniería, la IS continúa en evolución, esta se puede adaptar con facilidad para enfrentar los retos que implica la exigencia de agilidad. Las metodologías ágiles resuelven los problemas surgidos, posteriormente a la masificación del uso de la computadora personal, dado que las expectativas y necesidades por parte de los usuarios se hicieron mas urgentes y frecuentes (Uribe and Ayala, 2007).

En la actualidad Scrum es la metodología más popular y sigue ganando importancia. Esta metodología es un proceso diseñado para añadir energía, enfoque, claridad y transparencia en las actividades y productos a los miembros del equipo de desarrollo de software. Con una implementación adecuada de Scrum, se puede aumentar la velocidad de desarrollo, alinear objetivos individuales y organizacionales, crear una cultura dirigida por ejecución, crear valores, lograr una comunicación estable y consistente del funcionamiento en todos los niveles, mejorar el desarrollo individual y la calidad de vida. Esta metodología se basa en las teorías actuales de control de proceso y específicamente tiene como objetivo producir el mejor resultado final, con los recursos actuales y el tiempo disponible (Rueda Gutierrez, 2010).

Sin embargo, la adopción de los principios de calidad se ha dificultado en la comunidad del software; debido a que los métodos son difíciles de introducir, se han identificado causas de fracaso o retraso en los proyectos de software tales como (Akif and Majeed, 2012; Bermejo et al., 2014; Eloranta et al., 2013; Gandomani and Nafchi, 2015; Romero et al., 2009): Falta de comprensión de las metodologías por parte de los miembros de los equipo de desarrollo, resistencia general al cambio, falta de colaboración y comunicación con el cliente, estimaciones inexactas, falta de comunicación sobre el estado del proyecto, la falta de información histórica, carencia de destrezas y habilidades de los profesionistas sin experiencia en el área, entre otras.

1.2. Planteamiento del problema

Las estimación de esfuerzo es un factor esencial en la planificación de proyectos de software para evitar excesos de presupuesto, retraso de las fechas de entrega o falta de tiempo; esto a menudo resulta en software de baja calidad (Raith et al., 2013; Rubin, 2012). Scrum no cuenta con una técnica propia para realizar la estimación; sin embargo, la técnica más empleada es Planning Poker (PP) (Mahnic and Rozanc, 2012; Raith et al., 2013).

PP es una técnica de estimación basada en el consenso para estimar el tamaño de historias de usuario. Una historia de usuario (HU) se puede definir como una pieza corta de la funcionalidad que proporciona un cliente o un usuario de un sistema con un valor, representa las necesidades de los usuarios. Para medir estas historias de usuario se utilizan puntos de historia (story points), los cuales representan un valor relativo el cual es usado para comparar el esfuerzo para implementar los requerimientos (Torrecilla-Salinas et al., 2015).

PP tiene muchos beneficios, sin embargo, este método no siempre es eficiente debido a que el resultado es siempre basado en la observación de expertos y su experiencia. El puntaje asignado a las historias de usuario es un valor que no puede ser convertido fácilmente en tiempo (Mahnič and Hovelja, 2012; Popli and Chauhan, 2014a; Raith et al., 2013) además la decisión de los miembros del equipo es poco clara, debido a que toman en cuenta la complejidad y la importancia de la tarea en forma general, sin desglosar sus factores. Por tal razón es necesario desglosar estas variables en sus elementos para definir claramente cómo un miembro del equipo llega a su decisión de seleccionar el puntaje de estimación.

El objetivo de este trabajo es facilitar la toma de decisiones de los desarrolladores principiantes cuando estimen una historia de usuario. La decisión de cada miembro sería más clara al descomponer una decisión compleja en factores más simples y más precisos.

1.3. Objetivos

1.3.1. Objetivo general

- Proporcionar a los miembros novatos de un equipo de desarrollo o empresa sin datos estadísticos, un método de estimación de historias de usuario que mejore la estimación, tomando en cuenta la poca experiencia del desarrollador al iniciar su práctica con Scrum, a través de la descomposición de la complejidad en factores que permitan identificar los aspectos importantes al estimar una historia de usuario.

1.3.2. Objetivos específicos

- Identificar los factores más importantes para la estimación de historias de usuario en Scrum.
- Validar los factores encontrados en la literatura con desarrolladores principiantes y con expertos de la industria del software.
- Definir una técnica para representar el conocimiento dadas las características de nuestro dominio.
- Diseñar un modelo de representación de conocimiento con los factores identificados y el conocimiento del experto.
- Validar con desarrolladores principiantes y expertos el modelo propuesto.
- Probar en un ambiente real el funcionamiento del modelo propuesto.

1.4. Preguntas de investigación

Esta sección describe una serie de preguntas que necesitan ser contestadas para concluir la investigación, cada una tiene una justificación para ser planteada, a continuación se describen cada una de ellas.

1. ¿Cuáles son los factores más importante para asignar complejidad e importancia en la estimación de HU en Scrum? Es necesario conocer los factores considerados al realizar las estimaciones debido a que no se conoce claramente como el integrante con experiencia del equipo de Scrum llega a su estimación. Esto dificulta las estimaciones de los desarrolladores sin experiencia, ocasionando falta de consenso entre los miembros al momento de estimar las historias de usuario.

2. ¿Cuál es la opinión de los desarrolladores principiantes y profesionales acerca de los factores propuestos para la estimación de HU en Scrum? Es necesario tomar en cuenta la opinión de los desarrolladores principiantes debido a que el modelo propuesto servirá para ayudar a ellos mismos. Así mismo, la opinión de los profesionales sirve al proyecto porque ellos cuentan con la experiencia aportan su conocimiento.

3. ¿Cuál es la mejor técnica para representar el conocimiento dadas las características de nuestro dominio? Existen muchas estructuras para representar el conocimiento, sin embargo, debido a las características de nuestro dominio tales como: La falta de datos estadísticos y la incertidumbre, es necesario seleccionar la técnica que se apegue a dichas características.

4. ¿Cómo será diseñado el modelo de representación del conocimiento? Identificar los factores será el primer pasos, por otra parte, ocupamos combinarlos para formar un modelo con la técnica de representación de conocimiento elegida.

5. ¿Cómo será validado el modelo propuesto? El modelo desarrollado en forma teórica es el primer aspecto a cumplir, sin embargo, el modelo estaría incompleto sin las pruebas suficientes para comprobar que funciona, por tanto, primeramente debemos diseñar un experimento para validar el modelo con los profesionales antes de someterlo a un ambiente real.

6. ¿Cómo será probado el modelo en un ambiente real? La segunda parte de validación de la propuesta es definir un experimento ante un proyecto real para para probar la efectividad de la propuesta, así poder concluir los alcances de la investigación.

1.5. Beneficio esperado

Este estudio presenta una propuesta para asignar complejidad a las historias de usuario, en lugar de tener en cuenta la complejidad de la estimación como un valor único, esta se dividió en tres variables. Por lo tanto, la atención se pone en cada variable a la vez, obteniendo más precisión en la toma de decisiones de los miembros del equipo de Scrum. Además, teniendo en cuenta la prioridad y el valor de la historia de usuario, se consideran más detalles para enfocarse en las historias esenciales del proyecto.

Por otra parte, los desarrolladores que inician en la estimación con Scrum pueden apoyarse en la propuesta de este artículo para hacer estimaciones a través del modelo basado en la descomposición de la complejidad. Cuando los desarrolladores principiantes estiman de forma cercana a las estimaciones realizadas a través de BN, podemos afirmar que empiezan a aprender a estimar, por tanto sus estimaciones serán más confiables.

Este proyecto está enfocando en un modelo basado en el conocimiento de expertos, por tanto, no es necesario la recopilación histórica de los proyectos, esto brinda que empresas que inicia en el desarrollo de software empleando PP puedan tener un soporte para estimar mientras generan sus propias estadísticas.

1.6. Organización del documento de tesis

Esta sección describe de forma resumida la organización de este documento.

El capítulo 2 presenta el estado del arte, este se enfocan en los aspectos más importantes sobre Scrum y PP. Por otra parte, se analizan los fundamentos de las redes Bayesianas, la cual es la técnica considerada adecuada dada las características de nuestro problema. Finalmente, esta sección describe los trabajos relacionados al tema.

El capítulo 3 describe cómo se llevó a cabo la validación de los factores propuestos en el nuevo modelo y muestra cómo se construyó el modelo propuesto considerando los factores identificados. Ademas, este apartado explica como se construyó la parte cualitativa y cuantitativa de la red Bayesiana.

El capítulo 4 describe los experimentos con estudiantes y con personal de la industrial de software para validar el modelo construido.

En el capítulo 5 se presentan las conclusiones generales sobre el trabajo de investigación además aportaciones y trabajo futuro.

Las últimas secciones se refieren a las referencias que soportan esta investigación y los anexos.

Capítulo 2

Estado del arte

Esta sección define los conceptos básicos que rodean la investigación presente, así como los trabajos relacionados.

2.1. Conceptos básicos

2.1.1. Scrum

Es un método de desarrollo ágil que se define como un proceso iterativo, incremental y empírico para administrar y controlar el trabajo de desarrollo, es considerado como un marco de trabajo que presenta un conjunto de prácticas las cuales tienen por objetivo mantener la visibilidad, inspección y adaptación de proyectos de desarrollo de software. La implantación de este tipo de prácticas, obedece a una inquietud de las organizaciones o equipos de desarrollo que han adquirido experiencia, tienen una visión más amplia del área de negocio, conocen como aplicar la tecnología para beneficio de la misma y por consecuencia sus procesos van madurando. Esto significa que conocen mejor las actividades que se tienen que realizar para desarrollar software, planean, desempeñan y monitorean de forma más controlada sus proyectos. Por esto, demandan conocer y ejecutar prácticas ágiles, en las que los integrantes de los equipos de trabajo se auto-organizan y por lo tanto no se requiere de agentes externos

que controlen las actividades

Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada. Por lo tanto, Scrum adopta una aproximación pragmática, aceptando que el problema no puede ser completamente entendido o definido y centrándose en maximizar la capacidad del equipo de entregar rápidamente y responder a requisitos emergentes (Pauly and Basten, 2015)

2.1.1.1. Funcionamiento

Scrum se estructura en ciclos de trabajo llamados *Sprint*, los cuales son iteraciones con una duración de 2 a 4 semanas, y se suceden una detrás de otra. Al comienzo de cada *Sprint*, el equipo selecciona de la lista priorizada (*Product Backlog*) los elementos a realizar durante el Sprint, así mismo se comprometen a terminar las tareas al final del *Sprint* durante este ciclo no se pueden cambiar los elementos elegidos. Al finalizar el Sprint, el equipo lo revisa con los interesados en el proyecto, y les enseña lo que han construido (Schwaber and Sutherland, 2011; Sonia and Pedro, 2014).

Scrum, tiene tres diferentes niveles de planificación: *Release planning*, *Sprint planning*, y el *Daily Scrum*. Durante la reunión de *Release planning*, se discuten los aspectos estratégicos básicos como los costos generales o funcionalidad de un proyecto de desarrollo. Los detalles operacionales en cambio se planifican de un *Sprint* a otro durante el desarrollo del proyecto. El nivel más detallado de la planificación se lleva a cabo en las reuniones *Daily Scrum* las cuales tienen una duración de 15 minutos y es donde los miembros del equipo informan de su progreso en el desarrollo actual y las nuevas tareas son asignadas, el proceso se muestra en la figura 2.1. En comparación con las metodologías tradicionales, Scrum se basa en una estructura de equipo de auto-organización, esto no sólo hace que los proyectos Scrum más transparente y flexible, también presume que los miembros del equipo tienen un alto compromiso y sentido de responsabilidad. Además de la colaboración en equipo, la colaboración con el cliente también juega un papel importante en los proyectos Scrum. Como

propietario del producto, los clientes tienen su propio papel en proyectos Scrum y se integran en varias etapas del proyecto. Por ejemplo, deben participar en las reuniones de *Sprint* y el *Daily Scrum* para que estén siempre al tanto del estado de desarrollo actual (Overhage and Schlauderer, 2012).

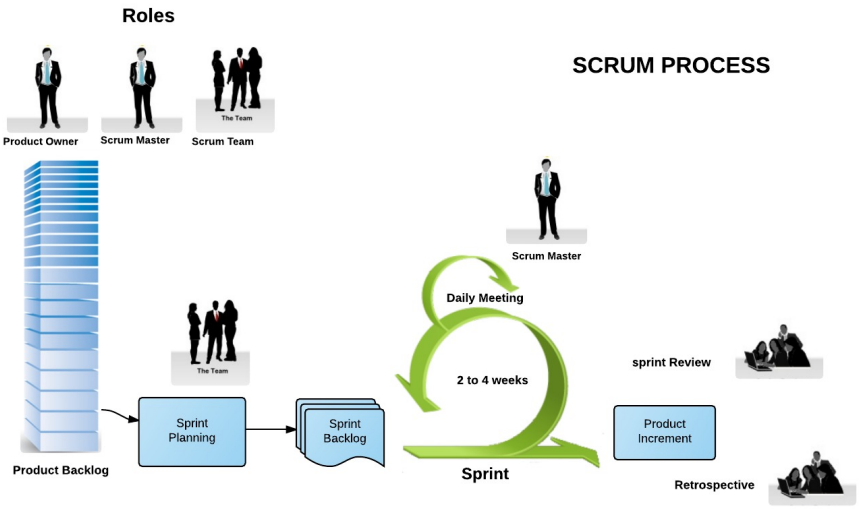

Figura 2.1: Proceso Scrum.

2.1.1.2. Valores de Scrum

- **Apertura**: Poner a disposición el Product Backlog hace visible el trabajo y prioridades. Las reuniones diarias hacen visible el compromiso en su conjunto y el estatus individual.

- **Enfoque**: El equipo tiene que enfocarse en la meta definida en un Sprint sin distracciones, mientras que el Scrum Master se enfoca en remover los obstáculos y evitar interrupciones en el trabajo de Equipo.

- **Compromiso**: El equipo se compromete con una meta definida en un Sprint y tiene la autonomía para decidir por sí mismo qué es lo mejor para cumplirla, para ello el Scrum Master se compromete a remover los obstáculos que reportan los integrantes del Equipo y evitar que factores externos interrumpan o distraigan el esfuerzo de desarrollo; Así mismo el Product Owner se

compromete a definir y priorizar los requerimientos en el Product Backlog, guiar la selección de la meta del Sprint, revisar y proporcionar la retroalimentación de los resultados de cada Sprint.

- **Respeto**: Los integrantes individuales del equipo son respetados por sus fortalezas y debilidades y no son culpados por las fallas del Sprint; Mediante la auto-organización el equipo adopta la actitud de resolver problemas a través de la exploración de soluciones en grupo.

2.1.1.3. Roles de Scrum

Scrum cuenta con tres roles principales: Product Owner, Scrum Master y el Scrum Team.

Product Owner: Representa a quien tiene un interés en el proyecto y en el producto resultante, es el único responsable de definir los requerimientos del producto a desarrollar durante el proyecto y conseguir el financiamiento inicial y continuo para el proyecto, también es responsable de definir los objetivos del retorno de inversión (ROI) y planear cómo se va a liberar el producto, debe asegurarse que la funcionalidad más valiosa se produce primero, priorizando los requerimientos de acuerdo al mercado o al área de negocio.

Entre sus principales funciones encontramos las siguientes:

- Ajustar los requerimientos y sus prioridades a lo largo de todo el proyecto.
- Analizar el contenido del producto de software.
- Decidir sobre la fecha de liberación del producto de software.
- Organizar la reunión *Scrum Planning*.
- Canalizar las necesidades del negocio, sabiendo escuchar a las partes interesadas en el producto y transmitirlas en objetivos de valor para el producto", al equipo de Scrum.
- Maximizar el valor para el negocio con respecto al retorno de inversión (ROI), abogando por los intereses del negocio.

- Revisar el producto e ir adaptándole sus funcionalidades, analizando las mejoras que éstas puedan otorgar un mayor valor para el negocio.

Scrum Master: Es responsable de que el proceso de Scrum funcione adecuadamente, de enseñarlo a cada uno de los involucrados en el proyecto, si es necesario. Es considerado como un facilitador para el equipo de desarrollo de software, debe asegurase de que cada uno sigue las reglas y prácticas. Hace que Scrum sea parte de la cultura de la organización mostrando sus beneficios. Trabaja muy de cerca con el Product Owner (Bass, 2014)

Entre sus principales funciones podemos encontrar las siguientes:

- Guiar la reunión *Daily Standup Meeting*.
- Conocer las tareas completas, qué tareas han comenzado, las nuevas y los estimados qué han sido cambiados.
- Observar cuidadosamente el número de tareas que aún están pendientes.
- Identificar dependencias y barreras que representen impedimentos para Scrum. Los debe priorizar y darles seguimiento implementando un plan de acción para resolverlos de acuerdo al orden de prioridad.
- Observar problemas o conflictos personales que deben resolverse. Se requiere que *Scrum Master* apoye para que el equipo los resuelva mediante el diálogo o bien, puede solicitar ayuda de la administración o de recursos humanos.

Scrum Team: Son los encargados de conocer cómo convertir los requerimientos en un incremento de la funcionalidad y así como de realizar el trabajo necesario para desarrollar dicho incremento. Además son responsables del éxito de cada iteración y del proyecto en su conjunto. Así mismo de seleccionar las actividades que se realizan en cada Sprint; cada miembro del equipo selecciona las tareas que se compromete a realizar durante el Sprint (Sutherland et al., 2014). Entre sus características encontramos las siguientes:

- Ser auto-administrable y auto-organizado.

- Selecciona la meta del *Sprint*, que es el conjunto de requerimientos que puede convertir en un incremento de la funcionalidad para el siguiente *Sprint*.

- Auto-administrarse para alcanzar la meta del *Sprint*, dentro de los límites del proyecto.

- Entregar los resultados del trabajo al *Product Owner*.

2.1.2. Planificación del proyecto

La planificación ágil parte de la idea de planificar en función de objetivos de negocio en lugar de tareas, priorizando los que aportan más valor, y esperando a dar detalle a objetivos y tareas conforme se va acercando el momento de construcción de estos objetivos, cuando la indeterminación se va reduciendo, de manera que se amortiza el esfuerzo de planificar de manera detallada.

Debido a que los requisitos para un proyecto ágil se definen mediante un conjunto de escenarios de usuario, es decir historias de usuario en el caso de Scrum, es posible desarrollar un enfoque de estimación que sea informal, razonablemente disciplinado y significativo dentro del contexto de la planificación del proyecto para cada incremento de software (Roger S. Pressman, 2010).

La estimación para proyectos ágiles usa un enfoque de descomposición que abarca los siguientes pasos:

- Cada historia de usuario se considera por separado para propósitos de estimación

- La historia de usuario se descompone en el conjunto de tareas de Ingeniería de software que sera necesario desarrollar.

- El esfuerzo requerido por cada tarea se estima por separado. La estimación puede basarse en datos históricos, un modelo empico o la experiencia.

- Las estimaciones para cada tarea se suman a fin de crear una estimación para la historia de usuario.

- Las estimaciones de esfuerzo para todos los escenarios se suman a fin de desarrollar la estimación de esfuerzo para el incremento.

Debido a que la duración del proyecto requerido para el desarrollo de un incremento de software es muy corta, este enfoque de estimación tiene dos propósitos:

1) Asegurarse de que el número de escenarios que se van a incluir en el incremento se ajusta a los recursos disponibles

2) Establecer una base para asignar esfuerzo conforme se desarrolla el incremento (Roger S. Pressman, 2010)

En Scrum la reunión de planificación se realiza en dos partes

- La primera parte puede tener una duración de 4 horas, en ella se deciden los elementos del product backlog que se van a desarrollar durante el Sprint: Está constituida por la Lista de Producto, el último Incremento de producto, la capacidad proyectada del Equipo de Desarrollo para el Sprint, y el rendimiento pasado del Equipo de Desarrollo. El número de elementos de la Lista de Producto seleccionados para el Sprint depende únicamente del equipo de desarrollo. Solo el Equipo de Desarrollo puede evaluar qué es capaz de lograr durante el Sprint que comienza. En esta reunión el Product owner y el equipo Scrum pueden desarrollar un objetivo del Sprint. El Objetivo del Sprint también proporciona al equipo flexibilidad sobre el alcance que finalmente entregarán, ya que aunque puedan tener que eliminar algunos elementos (dado que el Sprint está acotado en el tiempo), tendrían de todas formas que comprometerse a entregar algo tangible y "terminado" que esté en consonancia con el espíritu del Objetivo del Sprint.

- En la segunda se desglosan los elementos para determinar las tareas necesarias, estimar el esfuerzo requerido para cada tarea y son auto-asignadas a los miembros del equipo de desarrollo, Scrum no define exactamente cómo realizar la Segunda Parte de la Planificación de Sprint. Algunos equipos

emplean la velocidad de los últimos Sprints como guía de cuánto deberían poder completar. Otros equipos emplean aproximaciones más detalladas para calcular en primer lugar su capacidad.

Esta reunión no debe tardar más de un día

2.1.2.1. Importancia

Saber estimar y planificar es fundamental a la hora de enfrentar proyectos donde el producto requiere un grado importante de creatividad y/o innovación, como los de desarrollo de software

El esfuerzo invertido en un proyecto es una de las variables más importantes y más analizadas; Esto es debido a que la predicción de este valor mientras comenzamos los proyectos de software, ayuda a planificar las actividades futuras de manera adecuada, Sin embargo la estimación del esfuerzo con un gran valor de fiabilidad es un problema que aún no se ha logrado resolver (Suri and Ranjan, 2012)

2.1.2.2. Problemas en la planificación del proyecto

La estimación del proyecto de software es una forma de resolución de problemas y, en la mayoría de los casos, el problema por resolver; es decir, desarrollar una estimación de costo y esfuerzo para un proyecto de software es muy complejo como para considerarse en una sola pieza (Roger S. Pressman, 2010).

La complejidad del proyecto tiene un fuerte efecto sobre la incertidumbre inherente a la planificación. Sin embargo, la complejidad es una medida relativa que es afectada por la familiaridad con el esfuerzo pasado (Roger S. Pressman, 2010).

El tamaño del proyecto es otro factor importante que puede afectar en la precisión y la eficacia de las estimaciones

La disponibilidad de información histórica tiene una fuerte influencia sobre el riesgo de estimación; Al mirar hacia atrás, puede emular las cosas que funcionan y mejorar las áreas donde surgieron problemas

¿Qué ocurre cuando es pobre la concordancia entre las estimaciones? La respuesta a esta pregunta requiere una reevaluación de la información usada para hacer las estimaciones. Las estimaciones ampliamente divergentes con frecuencia pueden tener una de dos causas:

1) El ámbito del proyecto no se entiende adecuadamente o el planificador lo malinterpretó

2) Los datos de productividad usados por las técnicas de estimación basadas en problema son inadecuadas para la aplicación, obsoletos (ya no reflejan con precisión la organización de ingeniería de software) o se aplicaron mal.

Debe determinar la causa de la divergencia y luego reconciliar las estimaciones.

(Roger S. Pressman, 2010).

2.1.3. Planning Poker

Scrum no cuenta con una técnica propia para la estimación sin embargo la mas utilizada es PP. Esta técnica fue creada por James Grenning en el año 2002 y popularizada posteriormente por Mike Cohn; es una técnica basada en juicio de expertos que consiste en asignar a cada historia de usuario una puntuación que representa el esfuerzo requerido para su implementación (Mahnič and Hovelja, 2012).

Definición de historia de Usuario: Es la forma como son redactados los requerimientos, básicamente contiene cuatro características, una descripción corta de la funcionalidad a desarrollar, un tamaño relativo, un valor de negocio que define qué tan importante es esta HS para el negocio y uno o más criterios de validación para definir si está terminada, para definir el tamaño de una HU la métrica que se utiliza son los puntos de historia (Mitre-Hernández Hugo et al., 2014)

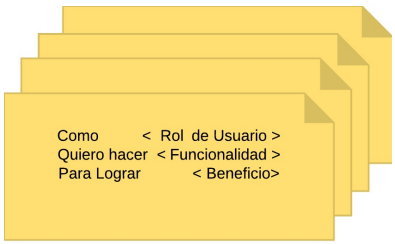

Figura 2.2: Formato de una historia de usuario.

Punto de historia (SP del ingles story point): usualmente corresponde a un día ideal de trabajo (aunque no necesariamente es así) a menudo se utiliza sólo un conjunto predefinido de valores posibles como 0.5, 1, 2, 3, 5, 8, 13, 20, 40 y 100 los cuales son una adaptación de la serie de Fibonacci propuesta por Cohn 2004 (Mahnič and Hovelja, 2012)

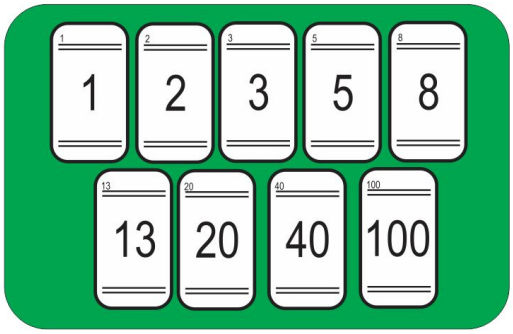

Figura 2.3: Cartas de Planning Poker.

PP provee mejores estimaciones que expertos individuales, ya que asegura la participación de todo el equipo equitativamente en el proceso de estimación independientemente si se encuentran entre las personas más expertas o influyentes, la participación de personas con diferente contextos ayuda a reducir el sobre-optimismo de las estimaciones basadas en juicio de experto identificando problemas que afectan su implementación (Mahnič and Hovelja, 2012).

2.1.3.1. Procedimiento

- Las historias de usuario son estimadas por miembros del equipo de desarrollo responsables de la implementación.
- Para cada historia el Product Owner inicia explicando sus requerimientos.
- Posteriormente el equipo discute el trabajo que estas requerirán haciendo preguntas al Product Owner sobre sus necesidades.
- Cada integrante le asigna de manera privada el esfuerzo requerido en dicha historia seleccionando una carta con el número de puntos que asignó.

- Todas las cartas son mostradas simultáneamente para asegurar la independencia de su estimación
- Si las estimaciones difieren mucho cada estimador explica por qué asignó ese valor, quien dio la estimación más alta y la más baja deberán explicar principalmente sus razones.
- Se deberá repetir el proceso no más de tres veces para llegar al consenso en la estimación.

Este proceso no debe ser democrático sino consensual (Mahnič and Hovelja, 2012; Tamrakar and Jørgensen, 2012).

2.1.3.2. Problemas en la estimación de Planning Poker

Aunque PP cuenta con muchos beneficios; se ha observado que el método de estimación actual en Scrum en su mayoría se basan en datos históricos de proyectos anteriores y la opinión de expertos, pero en ausencia de datos históricos y expertos en estos métodos no son eficientes. La ignorancia de los métodos de estimación puede causar efectos graves como exceder el presupuesto, no entregar a tiempo, mala calidad o no producto adecuado (Popli and Chauhan, 2013).

De Acuerdo con Popli en (Popli and Chauhan, 2013) existen varios problemas en la implementación de métodos de estimación que utiliza Scrum tales como la estimación de Esfuerzo, la Fecha estimada de lanzamiento y que los puntos de historia no se relacionan fácilmente con el tiempo porque estos representan la cantidad de trabajo y la velocidad puede ser diferente de un equipo a otro. Otro problema es que un punto de historia es un valor relativo, ya que el equipo determina cual es la referencia que se tendrá para poder medir con puntos de historia por lo que no se puede comparar los puntos de historia medidos por otros equipos, además no se puede hacer comparaciones de diferentes equipos respecto a la velocidad con que cada equipo desarrolla las historias de usuario.

Diferentes trabajos han enfrentado este problema (Moløkken-Østvold et al., 2008; Raith et al., 2013; Zahraoui et al., 2015) de diferentes maneras, pero sin considerar la incertidumbre proporcionada por la subjetividad de la persona. Una teoría para el manejo de la incertidumbre es la red Bayesiana (BN).

2.1.4. Redes Bayesianas

Las redes Bayesianas juegan diversos papeles importantes dentro de la Inteligencia Artificial. Uno de ellos es su actuación dentro del manejo de incertidumbre en los sistemas expertos. Otro papel importante lo tienen en lo que se conoce como descubrimiento de conocimiento en bases de datos; las redes Bayesianas permiten encontrar, de una manera consistente, relaciones probabilistas entre variables. Esta estructura es un formalismo que en los últimos años ha demostrado su potencialidad como modelo de representación de conocimiento con incertidumbre. El hecho de utilizar una representación gráfica para la explicación del modelo hace de las Redes Bayesianas sean una herramienta realmente muy atractiva en su uso, como representación del conocimiento. No sólo modelan de forma cualitativa el conocimiento sino que además expresan de forma numérica la fuerza de las relaciones entre las variables. Esta parte cuantitativa del modelo suele especificarse mediante distribuciones de probabilidad como una medida de la creencia que tenemos sobre las relaciones entre variables de modelo (César, 2006).

Las redes Bayesianas o probabilísticas se fundamentan en la teoría de la probabilidad y combinan la potencia del Teorema de Bayes con la expresividad semántica de los grafos dirigidos; las mismas permiten representar un modelo causal por medio de una representación gráfica de las independencias / dependencias entre las variables que forman parte del dominio de aplicación. Este enfoque es definido como un grafo acíclico dirigido, donde las uniones entre los nodos tienen definidas una dirección en el que los nodos representan variables aleatorias y las flechas representan influencias causales; el que un nodo sea padre de otro implica que es causa directa del mismo (Felgaer, 2004).

2.1.4.1. Teorema de Bayes

Fue publicado por primera vez en 1763, dos años después de la muerte de su creador: el matemático y teólogo inglés Thomas Bayes. (Rincón, 2014)

El teorema de Bayes es un procedimiento para obtener probabilidades condicionales (probabilidades de ocurrencia de acontecimientos condicionadas a la ocurrencia de otros acontecimientos).

La probabilidad condicional de un evento es una probabilidad obtenida con la información adicional de que algún otro evento ya ha ocurrido. Usamos $P(B|A)$ para denotar la probabilidad condicional del evento B que ocurre, dado que el evento A ya ha ocurrido. La siguiente fórmula (1) se utiliza para encontrar $P(B|A)$ (ecuación 2.1):

$$P(B|A) = \frac{P(A \wedge B)}{P(A)} \tag{2.1}$$

La explicación intuitiva para encontrar una probabilidad condicional a partir de la fórmula anterior sería: la probabilidad condicional de B dada A se puede encontrar asumiendo que el evento A ha ocurrido y, trabajando bajo esa suposición, calcular la probabilidad de que ocurra el evento B. Por lo tanto, la aplicación del teorema de Bayes es adecuada para revisar un valor de probabilidad basado en información adicional que se obtiene posteriormente. La clave para entender la esencia del teorema de Bayes es percibir que estamos tratando con sucesos secuenciales, donde se obtiene nueva información adicional para un evento posterior. Esta nueva información se utiliza para revisar la probabilidad del evento inicial. En este contexto, los términos probabilidad previa y probabilidad posterior son de uso común y por lo tanto es necesario definirlos. Una probabilidad previa es un valor de probabilidad inicial obtenido originalmente antes de obtener cualquier información adicional mientras que una probabilidad posterior es un valor de probabilidad que ha sido revisado usando información adicional que se obtiene posteriormente.

2.1.4.2. Formalismo de las redes Bayesianas

Una red Bayesiana es un grafo acíclico dirigido en el que los nodos representan variables aleatorias que pueden ser continuas o discretas; en las siguientes definiciones se utilizarán letras mayúsculas para denotar los nodos (X) y las correspondientes letras minúsculas para designar sus posibles estados (x_i).

Los estados que puede tener una variable deben cumplir con dos propiedades:

1. Ser mutuamente excluyentes, es decir, un nodo sólo puede encontrarse en uno de sus estados en

un momento dado.

2. Ser un conjunto exhaustivo, es decir, un nodo no puede tener ningún valor fuera de ese conjunto.

Describimos a continuación algunas definiciones y notaciones de la terminología de las redes Bayesinas:

- **Nodo**: Un nodo X es una variable aleatoria que puede tener varios estados x_i . La probabilidad de que el nodo X este en el estado x se denotará como $P(x) = P(X{=}x)$.

- **Arco**: Es la unión entre dos nodos y representa la dependencia entre dos variables del modelo. Un arco queda definido por un par ordenado de nodos (X, Y).

- **Padre**: El nodo X es un padre del nodo Y , si existe un arco (Y, X) entre los dos nodos.

- **Hijo**: El nodo Y es un hijo del nodo X , si existe un arco (Y, X) entre los dos nodos.

- **Probabilidad conjunta**: Dado un conjunto de variables X, Y,..... Z,, la probabilidad conjunta especifica la probabilidad de cada combinación posible de estados de cada variable

  P(x_i,y_j,....,z_k) $\forall$,j,...,k, de manera que se cumple que (ecuación 2.2):

$$\sum_{i,j,...k} P(x_i, y_j, ..., z_k) = 1 \tag{2.2}$$

- **Probabilidad condicional**: Dadas dos variables X e Y, la probabilidad de que ocurra y_j dado que ocurrió el evento x_i es la probabilidad condicional de Y dado X y se denota como P(y_j| x_i). La probabilidad condicional por definición es (ecuación 2.3):

$$P(y_j|x_i) = \frac{P(y_j, x_i)}{P(x_i)}, dado P(x_i) > 0 \tag{2.3}$$

Análogamente, si se intercambia el orden de las variables (ecuación 2.4):

$$P(x_i|y_j) = \frac{P(y_j,x_i)}{P(y_j)} \tag{2.4}$$

A partir de las dos formulas anteriores se obtiene (ecuación 2.5):

$$P(y_j|x_i) = \frac{P(y_j)}{P(x_i,y_j)}P(x_i) \tag{2.5}$$

Esta expresión se conoce como el Teorema de Bayes que en su forma general tenemos (ecuación 2.6):

$$P(y_j|x_i) = \frac{P(y_j,)P(x_i|y_j)}{\sum_j P(x_i|y_j)P(y_j)} \tag{2.6}$$

Al denominador se le conoce como el Teorema de la Probabilidad Total. En las redes Bayesianas el conjunto de valores que componen la probabilidad condicional de un hijo dados sus padres, se representa en las llamadas tablas de probabilidad condicional.

- **Independencia**: Dos variables X e Y son independientes si la ocurrencia de una no tiene que ver con la ocurrencia de la otra. Por definición se cumple que Y es independiente de X si y sólo si (ecuación 2.7):

$$P(y_j|x_i) = P(y_j,)P(x_i)\forall_i, j \tag{2.7}$$

Esto implica que (ecuación 2.8 y 2.9):

$$P(y_j|x_i) = P(y_j,)\forall_i, j \tag{2.8}$$

$$P(x_i|y_j) = P(x_i)\forall_i, j \tag{2.9}$$

- **Observación**: Es la determinación del estado de un nodo ($X = x$) a partir de un dato obtenido en

el exterior del modelo.

- **Evidencia**: Es el conjunto de observaciones e= $\{X{=}x, Y{=}y, ...Z = z\}$ en un momento dado.

- **Probabilidad a Priori** : Es la probabilidad de una variable en ausencia de evidencia.

- **Probabilidad a Posteriori**: Es la probabilidad de una variable condicionada a la existencia de una determinada evidencia ; la probabilidad a posteriori de X cuando se dispone de la evidencia e se calcula como P($X|e$).

2.1.4.3. Redes Bayesianas en la Ingeniería de Software

Muchas de las actividades en la Ingeniería del software, como por ejemplo, la estimación de costes o esfuerzo, evaluación de riesgos o fiabilidad tratan con valores inciertos o probabilísticas. Las redes Bayesianas son cada vez más populares en ingeniería, inteligencia artificial y estadística. (Rodríguez and Dolado, 2007).

Las redes Bayesianas son especialmente adecuadas para modelar la estimación del esfuerzo y pueden contribuir significativamente a la gestión de proyectos de software por lo que la aplicación de redes Bayesianas en la estimación del esfuerzo en el desarrollo de software está ganando popularidad especialmente en los últimos tiempos. Las redes Bayesianas se han utilizado ampliamente para predecir el esfuerzo de software. Las razones fundamentales para el uso de este enfoque es que permite a los analistas incorporar factores causales del proceso, así como combinar medidas cualitativas y cuantitativas, superando así las conocidas limitaciones de los métodos tradicionales de métricas de software (Karna and Gotovac, 2015).

Las redes Bayesianas tienen un número de características que hacen que sean apropiadas para la ingeniería del software (Rodríguez and Dolado, 2007).:

- **Representación gráfica:** Proveen una representación gráfica de las relaciones explícitas de dependencia del dominio. Generalmente las variables en la ingeniería del software como por ejemplo

esfuerzo o coste, están influenciados por muchos factores. Las redes Bayesianas nos permiten modelar sistemas complejos permitiéndonos entender las relaciones causales visualizándolas por medio del grafo.

- **Modelado cualitativo y cuantitativo:** Están formadas por un componente cualitativo, el grafo, y una parte cuantitativa, las tablas de probabilidades, que permiten utilizar criterios objetivos y subjetivos.

- **Inferencia bidireccional:** Pueden hacer inferencia en ambos sentidos, es decir, las variables de entrada pueden ser usadas para predecir las variables de salida y viceversa. Fijando las variables de salida con los valores deseados, es posible predecir qué valores de las variables de entrada permiten dicha salida.

- **Análisis de sensibilidad:** Dado un conjunto de evidencias, las redes Bayesianas permiten fácilmente calcular la sensibilidad de ciertas variables, simplemente modificando las evidencias.

- **Incertidumbre:** Pueden modelar grados de certidumbre, en vez de valores exactos. Por tanto, permiten modelar la incertidumbre de manera efectiva y explícitamente, por lo que pueden realizar buenas predicciones con información incompleta.

- **Valores de confianza:** La salida de una red Bayesiana es una probabilidad de distribución en vez de valores únicos. Este tipo de información se puede usar para medir la confianza que podemos depositar en la salida de la red Bayesiana, lo cual es esencial si el modelo va a ser usado en la toma de decisiones.

2.2. Soluciones existentes

2.2.1. Modelos de estimación en Ingeniería de Software

Muchos modelos se han propuesto durante los últimos 40 años para hacer frente a este problema. Estos modelos pueden clasificarse en Modelos Algorítmicos, Juicio de Experto y Estimación por Analogía (ver Figura 2.4) (Torrecilla-Salinas et al., 2015).

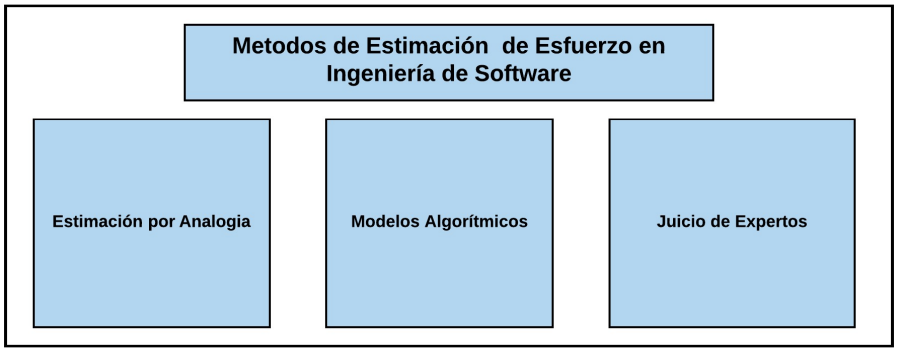

Figura 2.4: Métodos de estimación.

Los modelos basados en algoritmos utilizan enfoques matemáticos para calcular el esfuerzo del proyecto en función de sus principales factores de costo. Algunos de los modelos basados en algoritmos más relevantes se describen a continuación:

- COCOMO: Barry Boehm introdujo una jerarquía de modelos de estimación de software que llevan el nombre COCOMO, por COnstructive COst MOdel: modelo constructivo de costos. El modelo COCOMO original se convirtió en uno de los modelos de estimación de costo más ampliamente utilizados y estudiados en la industria. Evolucionó hacia un modelo de estimación más exhaustivo, llamado COCOMO II. Como su predecesor, COCOMO II en realidad es una jerarquía de modelos de estimación que aborda las áreas siguientes:

    - Modelo de composición de aplicación: Se usa durante las primeras etapas de la ingeniería de software, cuando son primordiales la elaboración de prototipos de las interfaces de usuario,

la consideración de la interacción del software y el sistema, la valoración del rendimiento y la evaluación de la madurez de la tecnología.

- Modelo de etapa temprana de diseño: Se usa una vez estabilizados los requisitos y establecida la arquitectura básica del software.
- Modelo de etapa postarquitectónica: Se usa durante la construcción del software.

Como todos los modelos de estimación para software, los modelos COCOMO II requieren información sobre dimensionamiento. Como parte de la jerarquía del modelo, están disponibles tres diferentes opciones de dimensionamiento: puntos objeto, puntos de función y líneas de código fuente (Roger S. Pressman, 2010).

- Puntos Función: Puede usarse de manera efectiva como medio para medir la funcionalidad que entra a un sistema. Al usar datos históricos, los PF pueden usarse para estimar el costo o el esfuerzo requerido para diseñar, codificar y probar el software, para predecir el numero de errores que se encontraran durante las pruebas y para prever el numero de componentes y/o lineas fuente proyectadas en el sistema implementado. Los puntos función se derivan usando una relación empírica basada en medidas contables del dominio de información del software y en valoraciones cualitativas de la complejidad del software. Los valores de dominio de información son:

  1. Número de entradas externas (EE)
  2. Número de salidas externas (CE)
  3. Número de archivos lógicos internos (ALI)
  4. Número de archivos de interfaz externos (AIE)

  Las organizaciones que usan los métodos de punto de función desarrollan criterios para determinar si una entrada particular es simple, promedio o compleja. No obstante, la determinación de complejidad es un tanto subjetiva.

- Técnica Puntos de Caso de Uso (Use Case Point technique): Los casos de uso brindan a un equipo de software comprensión acerca del ámbito y los requisitos del software. Sin embargo, desarrollar un enfoque de estimación con casos de uso es problemático por las siguientes causas:

    1. Los casos de uso se describen usando muchos formatos y estilos diferentes; no existe una forma estándar
    2. Los casos de uso representan una visión externa del software y por consiguiente, pueden escribirse en muchos niveles de abstracción diferentes.
    3. Los casos de uso no abordan la complejidad de las funciones y de las características que se describen.
    4. Los casos de uso pueden describir comportamiento complejo que involucran muchas funciones y características.

En los años 1990s, modelos no algorítmicos fueron descubiertos y se han utilizado en la estimación de esfuerzo y costo de proyectos; a continuación, se describen algunos de los principales modelos no basados en algoritmos:

- **Estimación por Analogía:** Utilizando esta técnica se crean estimaciones para nuevos proyectos comparando los nuevos proyectos con proyectos similares del pasado, Sin embargo requiere una cantidad considerable de cálculo. Este proceso es muy sencillo. Pero no todas las organizaciones tienen datos históricos para utilizar satisfactoriamente la analogía como medio de estimación (Suri and Ranjan, 2012).
- **Técnica Delphi:** En esta técnica se realiza una reunión con los expertos y un moderador, no implica discusiones cara a cara, sino interacción experta anónima a través de varias iteracioes supervisada por un moderador. Ellos darán sus sugerencias sin discutirlo con otros. El moderador reunirá todos los formularios de los expertos y los resumirá, el proceso se repetirá hasta que el resultado sea aprobado por todos. Además del anonimato, el método debe incluir iteraciones,

retro-alimentación controlada y agregación estadística de respuestas para que se implemente correctamente (Moløkken-Østvold et al., 2008; Kaur and Singh, 2015)

- **Juicio de Expertos:** Este método aprovecha la experiencia y la comprensión de un grupo de expertos para calcular un costo estimado. Esta técnica se utiliza junto con la técnica Delphi para mejorar y sistematizar la opinión de los expertos consultados. Es la técnica mas frecuentemente utilizada para estimación de esfuerzo en los proyectos de software. Este modelo se utiliza cuando hay limitación en la búsqueda y recolección de información precisa. Como se basa en expertos, deben ser experimentados y hay probabilidad de sesgo en los resultados. El proceso se llevará a cabo tomando las sugerencias de los expertos que tuvieron experiencia en proyectos similares (Kaur and Singh, 2015)

2.2.2. Trabajos relacionados

En esta sección se describen algunos trabajos relacionados con la estimación, con un método similar a nuestra propuesta aunque algunos consideran redes Bayesianas. La Tabla 2.1 muestra algunas características de los trabajos relacionados, en ella se describe el año de publicación, si consideran la complejidad dentro de las estimaciones y si esta desglosada, el método utilizado y el objetivos de la estimación.

La complejidad debe ser considerada con para reducir la dificultad de estimar. Diversos autores han propuesto diferentes métodos para la estimación de esfuerzo, entre los que podemos mencionar algunos como (Mendes, 2011) quien en su estudio propone un modelo Bayesiano para estimación de proyectos Web usando conocimiento de un dominio de expertos. Su modelo contiene 36 factores identificados como influyentes en estimación de esfuerzo en software y manejo de riesgos; ademas, 48 relaciones asociadas a estos factores. La red Bayesiana presentada en este trabajo fue construida y validado mediante una adaptación de la ingeniería del conocimiento de las redes Bayesianas

(Nassif et al., 2011) desarrolló un modelo de regresión para la estimación del esfuerzo de software

Tabla 2.1: Trabajos Relacionados.

		Complejidad				
		No	Si			
Art.	**Año**		**No desglozada**	**Desglozada**	**Metodo Utilizado**	**Objetivo**
(Mendes, 2011)	2011		x		Redes Bayesianas	Estimacion de Esfuerrzo de proyectos Web
(Nassif et al., 2011)	2011			x	Modelo de Regresion con SugenoFIS	Estimación de Esfuezo y Tamaño
(Popli and Chauhan, 2014a)	2014		x		Algoritmo Propio	Costo, Esfuerzo y duración del proyecto
(Zahraoui et al., 2015)	2015		x		Algoritmo Propio	Calculo de puntos de historia ajustados
(Karna and Gotovac, 2015)	2015		x		Redes Bayesianas	Estimación de Esfuerzo
(Zare et al., 2016)	2016		x		Redes Bayesianas	Estimacion de Esfuerzo (COCOMO)
Owais (2016)	2016			x	Algoritmo Propio	Estimación de Esfuerzo, duración y costo
(Dragicevic et al., 2017)	2017			x	Redes Bayesianas	Estimación de Esfuerzo

basado en el Modelo de Casos de Uso (UCP por sus siglas en ingles). Los autores emplearon un Sistema de Inferencia Difusa (FIS) para mejorar la estimación. La principal ventaja del modelo UCP es que se puede utilizar en las primeras etapas del ciclo de vida del software, cuando el diagrama de casos de uso está disponible. Manejaban la complejidad como un sinónimo de factor técnico, dividiéndolo en trece aspectos y asignando peso a cada uno de ellos.

(Popli and Chauhan, 2014a) propusieron un método de estimación algorítmica. Este enfoque consideró varios factores, estimando así la fecha de lanzamiento más precisa, costo, esfuerzo y duración del proyecto. La eficacia y la viabilidad del algoritmo propuesto se han demostrado considerando tres casos en los que se tomaron en cuenta y compararon diferentes niveles de factores. El método no descompone la complejidad en más variables.

Con el fin de producir estimaciones de esfuerzo y tiempo más precisas, (Zahraoui et al., 2015) calcularon los puntos de historia usando tres factores: Prioridad, tamaño y complejidad. Para calcular el esfuerzo total de un proyecto de Scrum, propusieron usar una nueva medida de punto de historia ajustada en lugar de una historia que apunte a uno. Además, dieron una manera de usar el punto de la historia ajustada propuesta en la velocidad ajustada; sin embargo, no descomponían la complejidad en factores.

De la misma manera (Karna and Gotovac, 2015) consideraron la complejidad como un variable

simple, pero esta variable es necesario desglosarla en otros factores. En su estudio ellos presentan un modelo Bayesiano incluyendo entidades relevantes que participan en la estimación de esfuerzo, en este estudio se consideran principalmente tres entidades: Proyecto, elementos de trabajo y estimadores. Esta propuesto es destacada, sin embargo, es modelo teórico que necesita ser probado, así mismo la complejidad necesita desglosarse para reducir la dificultad de estimar.

(Zare et al., 2016) presentaron una red Bayesiana de tres niveles basada en los componentes del modelo COCOMO para estimar el esfuerzo necesario para el desarrollo de software, de modo que el esfuerzo estimado se modifica utilizando un coeficiente óptimo resultante del control óptimo diseñado por el algoritmo genético. La complejidad en este estudio se define como una sola variable.

(Owais, 2016) desarrolló un algoritmo para el esfuerzo, duración y estimación de costos para el desarrollo de software ágil, considerando tres tipos de complejidad: Tecnológica, integración y equipo. Sin embargo, los puntos de la historia no se calculan, esta métrica es una entrada del modelo; nuestro trabajo considera esto como el principal problema: el cálculo de los puntos de la historia.

(Dragicevic et al., 2017) propusieron un modelo de red Bayesiana de cuatro niveles para la predicción del esfuerzo de trabajo en proyectos de desarrollo de software ágil. Se definió un conjunto de elementos tales como: Horas de trabajo, requisitos de complejidad, habilidades del desarrollador, tareas, formulario de complejidad, función de complejidad, informe de complejidad y especificación de calidad. La estructura de el modelo fue definido por los autores, mientras los parámetros de estimación son automáticamente aprendido de un conjunto de datos.

Todos los documentos presentados en la Tabla 2.1 consideran la complejidad; esto muestra su importancia en el concepto en las estimaciones. La complejidad necesita desglosarse para facilitar las estimaciones y aunque algunos autores la desglosan (Nassif et al., 2011; Owais, 2016; Dragicevic et al., 2017) otros no lo hacen (Mendes, 2011; Popli and Chauhan, 2014a; Zahraoui et al., 2015; Karna and Gotovac, 2015; Zare et al., 2016). Nuestra propuesta la considera en forma desglosada lo que hace este trabajo diferente.

Las redes Bayesianas han sido ampliamente utilizados en esta área, aunque los autores también han

utilizado la lógica difusa (Mendes, 2011; Karna and Gotovac, 2015; Zare et al., 2016; Dragicevic et al., 2017) y sus propios métodos (Popli and Chauhan, 2014a; Zahraoui et al., 2015; Owais, 2016).

Nuestro trabajo propone un modelo basado en el conocimiento de los expertos. Mediante la obtención de pesos de relaciones entre variables es posible construir la Tabla de probabilidad condicional para la inferencia estadística. Además, el enfoque se basa en Scrum y PP.

El objetivo de los trabajos relacionados es estimar el esfuerzo (todos los trabajos relacionados), el costo (Popli and Chauhan, 2014a; Owais, 2016), la duración (Popli and Chauhan, 2014a; Owais, 2016) y el tamaño (Nassif et al., 2011) de los proyectos. Ninguna de las obras citadas se centra en la educación, nuestra investigación se centra en ofrecer una visión diferente de la estimación a desarrolladores principiantes. Esto a través de un proceso de estimación basado en la descomposición de la complejidad cuando se utiliza PP. Nuestro método se emplea a través de preguntas que representan variables en una red Bayesiana. De esta manera el estudiante percibe la complejidad ya que los elementos descompuestos facilitan su estimación.

Capítulo 3

Factores de la estimación y modelo propuesto

Para hacer esta investigación se realizó una revisión de literatura para identificar los factores que se emplean en la estimación de historias de usuario. Además, con estos factores se desarrolló el modelo para representar el conocimiento de expertos. A continuación se describen a detalle estos aspectos.

3.1. Factores en la literatura

Algunos autores como (Raith et al., 2013; Martel, 2014; Zahraoui et al., 2015; Popli and Chauhan, 2014a) definen atributos que son considerados por los equipos de desarrollo en la estimación, estos atributos se utilizan para definir la complejidad y la importancia de las tareas en Scrum . Estos factores son llamados de diferente forma pero conservan su significado; los factores más significantes se describen a continuación:

Los principales factores para definir la complejidad de una tarea son (Raith et al., 2013; Martel, 2014; Popli and Chauhan, 2014a; Grimstad and Jørgensen, 2006; Moløkken-Østvold et al., 2008):

- Tiempo: Se refiere al tiempo que se invierte al realizar una HU; aun cuando esta sea fácil de desarrollar, puede ser una HU sencilla pero con muchos detalles que hacen que sea muy laboriosa (Popli and Chauhan, 2014b).

- Esfuerzo mental: Se refiere al esfuerzo cognitivo de análisis que se invierte para diseñar y realizar una HU. Por ejemplo, puede involucrar el uso de algoritmos con alta lógica computacional, por tanto la creación o uso de esta clase de algoritmo requiere alto grado de concentración.

- Experiencia: Se refiere al tiempo realizando trabajos o proyectos similares en naturaleza y complejidad. A mayor experiencia puede percibirse menos compleja una HU, el desarrollo de trabajos previos similares aporta calidad y rapidez al término de una HU. Así como la asignación de complejidad de la HU será más precisa

Los principales factores para definir la importancia de una HU son (Martel, 2014; Zahraoui et al., 2015):

- Prioridad: Se refiere a la importancia de la HU debido a que otras tareas dependen de esta. Las más importantes deben ir en los primeros sprint (Zahraoui et al., 2015; Grimstad and Jørgensen, 2006).

- Valor de la tarea: Se refiere a la contribución de la HU respecto a la completud del proyecto; también puede referirse a qué tan completo queda el sistema con o sin esa funcionalidad implementada con la HU en cuestión, además a la cantidad de ingresos que se podrían ganar o perder por desarrollar o no la HU (Zahraoui et al., 2015).

3.1.1. Relaciones entre factores

La Figura 3.1 representa la relación entre los factores. Se establecieron grupos de factores de acuerdo al tipo de relación. El nodo complejidad es definido por el equipo de desarrollo mientras que el nodo importancia lo define el Product Owner.

La variable $complejidad$ se divide en $esfuerzo$, $tiempo$ y $experiencia$. Las dos primeras variables se organizan en un nodo llamado $tiempo - esfuerzo$. Por lo tanto, la variable $experiencia$ tiene más peso sobre $esfuerzo$ y $tiempo$. La experiencia es considerada como un factor importante para obtener calidad y precisión, por eso se considera más importante que otra.

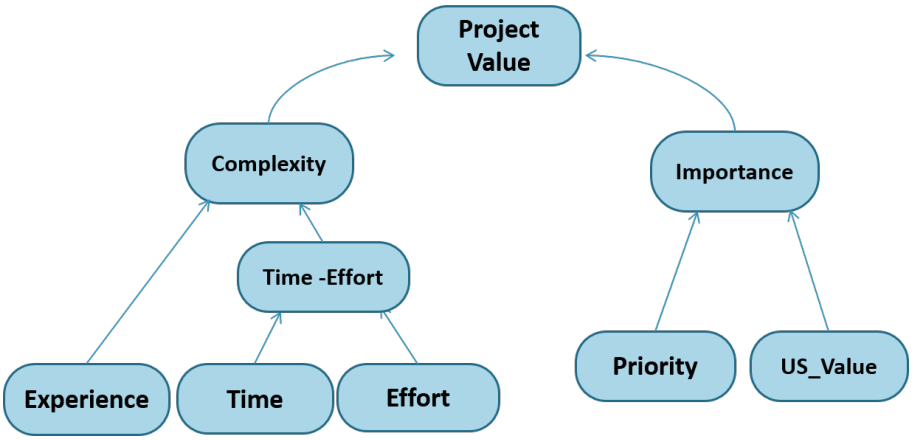

Figura 3.1: Modelo Propuesto.

Las variables *complejidad* e *importancia* están relacionadas por *valor_en_proyecto*. Esta última variable establece la importancia de una tarea en el proyecto. Por lo tanto, es necesario prestar especial atención a la tarea con valores más altos. El cuidado de estas tareas definirá la calidad del proyecto final.

3.1.2. Validación de las relaciones entre factores

Los factores fueron validados a través de profesionales en la industria del desarrollo de software en México. Fueron 16 ingenieros de software de 4 empresas con diferentes funciones, tales como: Scrum masters, product owner, testers, administrador de desarrolladores de software y arquitectos de software. Ellos trabajan con Scrum y usan Planning Poker para asignar complejidad a las historias de usuario, por lo tanto, tienen experiencia en esta área.

Cada factor fue evaluado considerando la sentencia *Empleo el factor X para asignar complejidad o importancia a una historia de usuario en Scrum"*, donde la variable *X* representa cada factor propuesto. La encuesta utiliza una escala Likert de 7 puntos (totalmente en desacuerdo a totalmente de acuerdo) para medir el grado de uso de cada factor (para más detalles de la encuesta ver el Apéndice A.1).

Los resultados de esta validación se muestran en la Tabla 3.1. Considerando que 1, 2 y 3 de la escala son aspectos negativos y 5, 6 y 7 aspectos positivos. El punto neutral no se toma en cuenta hacia

ninguno de los puntos; Los porcentajes de aceptación de los factores se encuentran en el lado positivo en cada caso: experiencia (100 %), tiempo (87.5 %), esfuezo(100 %), prioridad (75 %), y Valor de la HU (81 %).

El grado de aceptación de las variables en grupo fue obtenido tomando la escala Likert de 1-7. Las variables fueron clasificadas en aceptadas (5-7) o no aceptadas (1-3), se agrega cada opinión de los encuestados para obtener el total. Tenemos un grupo de encuestados que aceptan variables (21-35 puntos) y otro grupo que no lo hace (5-20 puntos). Hubo 100 % de opiniones que aceptaba las variables.

Se concluye que los profesionales aceptaron los factores tanto individualmente como en grupo.

Tabla 3.1: Validación de Factores por Profesionales.

	Escala Likert						
	1	2	3	4	5	6	7
Factors	Totalmente en desacuerdo			Neutral			Totalmente de acuerdo
Tiempo	6.25 %	0.00 %	0.00 %	6.25 %	6.25 %	31.25 %	50.00 %
Esfuerzo	0.00 %	0.00 %	0.00 %	0.00 %	31.00 %	25.00 %	44.00 %
Experiencia	0.00 %	0.00 %	0.00 %	0.00 %	6.00 %	44.00 %	50.00 %
Prioridad	0.00 %	6.25 %	0.00 %	18.75 %	18.75 %	12.50 %	43.75 %
Valor de HU	0.00 %	0.00 %	19.00 %	0.00 %	31.00 %	25.00 %	25.00 %

3.2. Técnicas para representar el modelo

Para desarrollar un modelo con los factores de la sección anterior es necesario una técnica de representación del conocimiento. El modelo debe considerar factores que permitan representar el conocimiento del experto en el área de la estimación. Se encontraron en la literatura técnicas tales como: redes Bayesianas, mapas cognitivos difusos, mapas de conocimiento, grafos, mapas de conceptos, redes semánticas y mapas de Memoria. Todas estas técnicas trabajan a través de nodos y relaciones, esto permite modelar el problema basado en los factores detectados y enlaces entre estos.

Los enfoques encontrados fueron evaluados a través de los siguientes aspectos:

- Información cualitativa: Se refiere a la estructura jerárquica útil para representar variables y sus relaciones.

Tabla 3.2: Comparativa entre técnicas.

Enfoque	Inf. Cualitativa	Inf. Cuantitativa	Teoría solida
Red Bayesiana	X	X	X
Mapas cognitivos difusos	X	X	X
Mapas de conocimiento	X	X	
Grafos	X		X
Mapa de conceptos	X		
Red semántica	X		X
Mapa de memoria	X		

- Información cuantitativa: Incorporada dentro de la estructura jerárquica para lograr inferencia y grados de posesión del conocimiento.

- Teoría solida: Técnicas con bases bien fundamentas y con años de trabajo en diversas áreas y estudios.

La Tabla 3.2 representa los resultados de la evaluación. La columna referente a la información cualitativa indica que cualquier técnica aquí presentada es candidata para este tipo de organización de la información. En la siguiente columna (información cuantitativa), los enfoques: redes Bayesianas, los MCD y los mapas de conocimiento cumplen con este punto. Finalmente las RBs y MCD son una técnica con bases bien fundamentas y con años de trabajo en diversas áreas y estudios. En resumen las RBs y los MCD son las que cumplen con los tres criterios especificados.

Las RBs y MCD comparten bastantes elementos y están diseñados para objetivos muy similares, sin embargo, manejan algunas diferencias entre ellos. Un estudio comparativo elaborado por Cheah (Cheah et al., 2008) posiciona en forma general mejor a las RBs frente a los MCD (ver tabla 3.3).

Los MCD al ser la combinación de dos enfoques requieren un curva de aprendizaje mayor al tener que conocer la teoría de la lógica difusa y de los mapas cognitivos, además estos no cuentan con variedad de herramientas de soporte, a diferencia de las RBs que tienen muy buen soporte en este aspecto, a pesar de esto y acorde al trabajo de Cheah este enfoque tiene mayor usabilidad que las RBs, debido a que es más amigable y su representación es más directa, intuitiva y fácil de manejar. Uno de las grandes ventajas es el manejo de ciclos carentes en la redes Bayesianas.

Tabla 3.3: Comparativa entre RBs y MCD.

Enfoque	RB	MCD
Usabilidad		x
Manejo de ciclos		x
Encadenamiento hacia adelante y atrás	x	x
Eficiente mecanismo de propagación	x	
Herramientas de soporte	x	
Aplicación en distintas áreas	x	

La comparativa elaborada por Cheah obtiene puntos a favor de las redes Bayesianas sobre los MCD donde destaca:

- Encadenamiento hacia adelante y hacia atrás.
- Eficientes mecanismos de propagación.
- Suficientes herramientas de soporte e implementación.
- Uso de RBs en distintas áreas.
- Correcta teoría matemáticas bien formada definida sobre axiomas básicos.
- Correctitud de sus mecanismos de inferencia es probable.

Varios trabajos (Moløkken-Østvold et al., 2008; Raith et al., 2013; Zahraoui et al., 2015) han enfrentado el problema de esta investigación sin considerar la incertidumbre introducida por la subjetividad de la persona. Por otro lado, Owais (2016); Dragicevic et al. (2017) consideraron la incertidumbre de la estimación a través de RBs. Las RBs son una de las técnicas más precisas para la estimación de esfuerzo y costo para el desarrollo de software ágil mediante técnicas de computación blanda (Bilgaiyan et al., 2016).

3.3. Construcción del modelo: red Bayesiana

Esta sección muestra cómo se construyó la Red Bayesiana. Primeramente, la parte cualitativa de la RB fue diseñada, esta parte representa la estructura formada por nodos y relaciones, esta parte se desarrolló basándose en el grupo de variables definidas la Figura 3.1. Posteriormente la parte se definió la parte cuantitativa, esta parte representa a las TPCs, la construcción de estas tablas se construyó con la opinión de los profesionales.

3.3.1. Parte cualitativa de la red Bayesiana

La RB forma una estructura jerárquica con variables organizadas por niveles, estas variables se describen a continuación:

- **Variables de primer nivel:** Estos nodos se encuentran en los extremos de la red, son recolectores de evidencia. Para simplificar y evitar muchas opciones en las variables, se definieron cinco estados para los factores experiencia, tiempo y esfuerzo, estas variables se miden en la siguientes escala: Muy poco, poco, medio, mucho y demasiado; además tres estados fueron considerados para los factores prioridad y valor de la HU, estas variables se miden en la siguiente escala: bajo, medio y alto
- **Variables de segundo nivel:** La red cuenta con nodos que agrupan las variables de primer nivel además ayudan a organizar la información jerarquicamente.
- **Variable valor en el proyecto:** Esta variable muestra el valor final para medir el grado de complejidad e importancia de la HU.

Todas las variables de primer nivel, excepto la experiencia, incrementan el valor de la tarea en el proyecto (variable valor del proyecto) cuando estas variables aumentan. Esto significa que si una variable tiene su estado en *alto* entonces subirá la probabilidad del valor final. Sin embargo, el caso de la experiencia es contrario debido a que esta resta complejidad en la HU.

3.3.2. Parte cuantitativa de la red Bayesiana

Este estudio definió el peso de las relaciones entre variables con la opinión de los profesionales de Scrum. En López-martínez et al. (2017) se establecieron las relaciones de variables basándose en la opinión de estudiantes. Sin embargo, los expertos del dominio tienen más conocimiento y experiencia que estudiantes. La construcción de la parte cuantitativa de RB necesita dos aspectos estadísticos: valores de relaciones entre variables y valores a priori de variables de primer nivel.

- **Valor de la relación entre variables:** Este valor fue obtenido de la experiencia y conocimiento de los miembros del equipo de Scrum. Cada variable se validó considerando qué tan de acuerdo estaban los expertos con el uso de los factores mencionados para evaluar la complejidad y la importancia de una historia de usuario.
- **Valores a priori para las variables de primer nivel:** Se refiere a cómo han sido evaluadas otras historias de usuario finalizadas.

3.3.3. Valores de la relación entre variables

Las siguientes ecuaciones y definiciones calculan los pesos entre las variables:

Definición 3.1. *Sea* $V = \{v_1, ..., v_n\}$ *el conjunto que denota la opinión de los expertos. Donde el elemento* $v = (weight, frecuency)$ *representa el peso dado a un punto de la escala de Likert (weight) y la frecuencia de la escala (*$frecuency$*). La escala de cinco puntos de Likert define la variable n en cinco debido al número de puntos. Con base en la ecuación 1, la Ecuación 3.1 establece la relación no normalizada de una variable p (Definición 3.2).*

$$\underset{p \in P}{p} = \frac{\sum_{v \in V} \left(v_{weight} v_{frecuency}\right)}{\sum_{v \in V} v_{frecuency}} \tag{3.1}$$

Definición 3.2. *Sea* $P = \{p_1, ..., p_m\}$ *el conjunto de valores que representan a los nodos padres, estos nodos tienen una influencia causal sobre el mismo nodo hijo. La Ecuación 3.2 obtiene las relaciones*

normalizadas considerando el conjunto de padres y sus valores calculados con la Ecuación 3.1.

$$\underset{p\in P}{p_{norm}} = \frac{p}{\sum_{p\in P} p} \tag{3.2}$$

El peso de cada relación se muestra en la Figura 3.2, esto ilustra la opinión de los profesionales.

Definición 3.3. *Sea* $A = \{a_1, ..., a_q\}$ *el conjunto de variables de segundo nivel; complejidad, importancia, y tiempo − esfuerzo, Estas variables tiene dos estados: bajo, and alto. Sus valores son calculados mediante la Ecuación 3.3.*

$$\underset{a\in A}{a} = \frac{\sum_{p\in P} p}{|P|}. \tag{3.3}$$

Donde el conjunto P representa a los padres de la variable a, el total de variables que tienen influencia sobre otra está representado por $|P|$*, y el valor no ponderado se representa por p. El valor normalizado se calcula con la Ecuación 3.2, pero tomando en cuenta la variable a en lugar de p.*

3.3.3.1. Aplicación de las ecuaciones

En esta sección explica como se obtuvieron los pesos de la RB para construir las TPCs. La Tabla 3.4 muestra las frecuencias organizadas por factor, así como los pesos asignados a cada punto en la escala Likert, la última fila representa el peso del factor en la RB.

Tabla 3.4: Frecuencias y Pesos de las variables de primer nivel.

		Frecuencia				
Pesos	**Escala Likert**	**Experiencia**	**Tiempo**	**Esfuerzo**	**Prioridad**	**Valor de US**
-1	1	0	1	0	0	0
-0.66	2	0	0	0	1	0
-0.33	3	0	0	0	0	3
0	4	0	1	0	3	0
0.33	5	1	1	5	3	5
0.66	6	7	5	4	2	4
1	7	8	8	7	7	4
	Peso de Variable	0.561	0.485	0.515	0.542	0.458

Para obtener los pesos normalizados de la variable *tiempo*, se utilizó la Ecuación 3.1 en base a las

frecuencias y pesos de la Tabla 3.4 (Ecuación 3.4):

$$time = \frac{(-1 \times 1) + (-0.66 \times 0) + ... + (1 \times 8)}{16} = 0.664 \tag{3.4}$$

La Ecuación 3.2 es usada para obtener los pesos de la variable *tiempo*. Es necesario considerar nodos que sean padres del mismo nodo hijo; la ecuación considera el valor no normalizado de $esfuerzo$ (0.706) (Ecuación 3.5).

$$time_{norm} = \frac{0.664}{0.664 + 0.706} = 0.485 \tag{3.5}$$

Los valores de las variables de primer nivel se representan en la tabla 3.4. Las variables de segundo nivel obtienen su peso basándose en la Ecuación 3.3 (Ecuación 3.6).

$$time - effort = \frac{0.664 + 0.706}{2} = 0.685 \tag{3.6}$$

Para obtener el peso normalizado de la variable $tiempo - esfuerzo$, se consideró la Ecuación 3.2 y el otro nodo padre que influyen en la misma variable, en este caso la experiencia, el cuál debe tener su peso no normalizado(0.809) (Ecuación 3.7).

$$time - effort_{norm} = \frac{0.685}{0.809 + 0.685} = 0.458 \tag{3.7}$$

Calculando las ecuaciones para cada variable de primer y segundo nivel, se obtienen los resultados mostrados en la Figura 3.2.

3.3.4. Construcción de tablas de probabilidad condicional

El último paso es asignar una TPC a cada nodo de la estructura. Esta parte construye una TPC para un nodo secundario dado, para cada nodo secundario se repite el proceso. Una variable hija es aquella que tiene influencia de otras variables padres *P*.

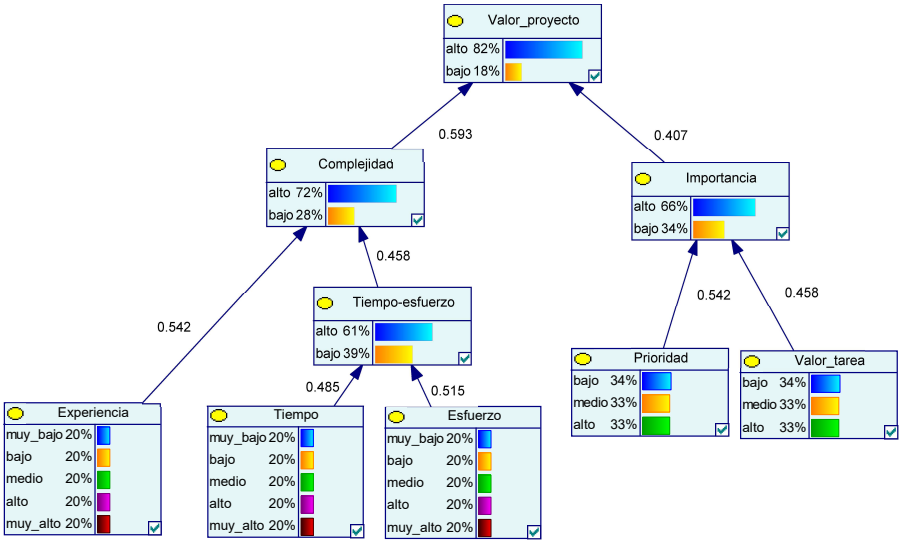

Figura 3.2: Red Bayesiana resultante.

Una vez que cuenta con la estructura de la red y los pesos de sus relaciones; el siguiente paso es asignar la TPC a cada nodo.

Para ello se definen un conjunto de escalas para las variables de primer nivel representadas por $S = \{s_1, s_2, ..., s_m\}$. Donde se colocaron los estados ponderados como: *bajo* (1), *medio* (2), y *alto* (3). Los valores ponderados fueron 0.17, 0.33, y 0.50.

Los valores ponderados para las variables con cinco estados fueron 0.067, 0.133, 0.200, 0.267, y 0.333 para *Muy_Bajo*, *Bajo*, *Medio*, *Alto*, y *Muy_Alto* respectivamente. Los valores ponderados para las variables con tres estados fueron 0.17, 0.33, y 0.50 para *bajo*, *medio*, y *alto* respectivamente.

La matriz en la Ecuación 3.8 muestra el proceso para crear la TPC. El valor de la variable se multiplica por el valor de la escala para encontrar una ponderación ($p_1 s_1$, $p \in P$, $s \in S$). Posteriormente, los resultados en el primer paso se combinan, cada combinación obtiene un valor único que añade cada elemento de la combinación $((p_1 s_1) + (p_2 s_1) + \cdots + (p_n s_1))$.

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{bmatrix} \tag{3.8}$$

Donde:

$$
\begin{aligned}
w_{11} &= ((p_1 * s_1) + (p_2 * s_1) + \cdots + (p_n * s_1)) \\
w_{12} &= ((p_1 * s_1) + (p_2 * s_1) + \cdots + (p_n * s_2)) \\
w_{1m} &= ((p_1 * s_1) + (p_2 * s_1) + \cdots + (p_n * s_m)) \\
w_{21} &= ((p_1 * s_2) + (p_2 * s_1) + \cdots + (p_n * s_1)) \\
w_{22} &= ((p_1 * s_2) + (p_2 * s_1) + \cdots + (p_n * s_2)) \\
w_{2m} &= ((p_1 * s_2) + (p_2 * s_1) + \cdots + (p_n * s_m)) \\
w_{n1} &= ((p_1 * s_m) + (p_2 * s_m) + \cdots + (p_n * s_1)) \\
w_{n2} &= ((p_1 * s_m) + (p_2 * s_m) + \cdots + (p_n * s_2)) \\
w_{nm} &= ((p_1 * s_m) + (p_2 * s_m) + \cdots + (p_n * s_m))
\end{aligned}
$$

Después, la variable *max* representa el valor máximo de la matriz, esta variable se utiliza para obtener los valores finales de forma proporcional. El valor máximo divide la matriz en la Ecuación 3.8, en la forma $W = W/max$, para obtener los valores finales.

Estos valores representan el estado $alto$ de las variables segundo nivel y la variable de valor del proyecto, el estado $bajo$ es el valor complementario del estado $alto$.

$$
W/max = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{bmatrix} /max \tag{3.9}
$$

Finalmente, una TPC se obtiene a una variable, el proceso se repite para cada variable. Una representación de estructura de red adecuada es necesaria, con el fin de controlar el crecimiento exponencial. Esto significa que muchas relaciones o muchos estados en una variable provocan un alto grado de complejidad computacional. Por lo tanto, necesitamos una atención especial a este punto.

Capítulo 4

Experimento

4.1. Pruebas del modelo

El experimento consistió en comparar las estimaciones tradicionales de PP con las estimaciones de la RB de desarrolladores principiantes en proyectos reales y profesionales de empresas de desarrollo de software.

Las estimaciones de los profesionales deben ser más exactas. Por lo tanto, las estimaciones de la RB deben ser similares a las estimaciones profesionales tradicionales. Esto demuestra que la descomposición de la complejidad a través de la RB se fundamenta en sólidas bases teóricas y empíricas.

Ambos tipos de participantes, estudiantes y profesionales, siguieron el mismo proceso, el cual consistió en los siguientes pasos:

1. El equipo se reunió para estimar la complejidad de las HU.

2. Los miembros del equipo Scrum recibieron las HU del Sprint en curso.

3. Los miembros del equipo determinaron la complejidad de la HU con la forma tradicional de PP.

4. Cada miembro del equipo de manera independiente, respondió un conjunto de preguntas relacionadas con los factores de la RB.

5. La información proporcionada por los miembros del equipo se introdujo en la RB para calcular la estimación.

6. Después el Sprint siguió su curso de manera normal.

Finalmente, sólo los desarrolladores principiantes hicieron una estimación extra al final del Sprint, esta última estimación debe ser más precisa que la primera estimación tradicional, debido a que los desarrolladores principiantes adquieren experiencia conforme avanza el Sprint.

Las HU fueron descritas a detalle para estimar su complejidad con respecto a los factores propuestos (ver Apéndice B.1 y B.2). Por ejemplo, para medir la experiencia se utilizó la siguiente pregunta: ¿Cuál es tu EXPERIENCIA para determinar el nivel de complejidad de esta HU?, las preguntas están explicadas con más detalle en el Apéndice A.2.

Este estudio consideró los estados de las variables como respuestas a preguntas. El sistema de la RB recibió respuestas de los profesionales y estudiantes.

Este trabajo contempla la serie Fibonacci modificada para evaluar la complejidad de HUs; esta serie consiste en los valores 1, 2, 3, 5, 8 y 13. En estudio previo (López-martínez et al., 2017) se consideraban 9 cartas para estimar como se sugieren en la técnica de PP; sin embargo, la empresa en el segundo experimento realizado en esta investigación solamente emplea 6 cartas. Ellos argumentan que 6 cartas son suficientes para realizar las estimaciones; esto debido a que una carta con un puntaje arriba de 20 SP en la escala de 9 cartas representan demasiada complejidad y dicha HU deberá ser desglosada en más historias.

La prueba de Spearman se utilizó para medir la correlación entre la RB de desarrolladores principiantes y de profesionales

4.1.1. Estimaciones realizadas por desarrolladores principiantes

La simulación de la propuesta se llevó en primera instancia con la participación de alumnos de Ingeniería en Computación en la Universidad Autónoma de Baja California los cuales cursaban el 7mo y 8vo semestre. El equipo estaba formado por seis integrantes de la materia aseguramiento de la calidad de software.

El experimento consideró un proyecto corto basado en 2 sprint con 9 y 5 HUs (ver Apéndice B.1 y B.2).

4.1.1.1. Resultados de las estimaciones por desarrolladores principiantes

Los desarrolladores principiantes realizaron sus estimaciones usando PP de la manera tradicional con el mazo de cartas, adicionalmente usando la RB utilizando el cuestionario para estimar del Apéndice A.2. Para la estimación a través de la RB, los estudiantes respondieron cinco preguntas para cada variable de primer nivel hubo cinco preguntas para cada HU, La RB recibió como entrada las respuestas de los estudiantes.

La información de cada HU estimada por los desarrolladores principiantes en el Sprint 1 se muestra en la Tabla 4.1 y los resultados del Sprint 2 se presentan en la Tabla 4.2. Ambas tablas representan la salida de la RB para cada sprint. Las variables de segundo nivel y el valor de proyecto variable se representan por su nombre en columnas de tabla. La Columna $complex_{stu}$ representa las estimaciones realizadas por los estudiantes de la manera tradicional. Los valores de la $complexity$ se convierten a la escala de Fibonacci modificado utilizando la Ecuación 4.1, los resultados se muestran en la columna $complex_{stu\_RB}$. La columna $complex_{stu2}$ muestra las estimaciones de los desarrolladores principiantes una vez concluida la HU.

$$Y = Y1 + [\left(\frac{X - X_1}{X_2 - X_1}\right)(Y_2 - Y_1)] \tag{4.1}$$

Donde:

- X representa el valor de la complejidad de la RB.
- Y representa la carta de PP a seleccionar.
- X_1 representa el menor valor de complejidad de la RB.
- X_2 representa el máximo valor de complejidad de la RB.
- Y_1 representa la menor posición de las cartas de PP.
- Y_2 representa la mayor posición de las cartas de PP.

El resultado de la Ecuación 4.1 es el número de la carta en lugar del valor Fibonacci. Es decir, el puntaje de Fibonacci 1 corresponde a la carta 1, puntaje Fibonacci 2 corresponde a la carta 2, puntaje Fibonacci 13 corresponde a la carta 6, esto proporciona la posibilidad de adaptarse a cualquier valoración posible.

4.1.1.2. Pruebas de correlación en la RB (desarrolladores principiantes)

Se utilizaron tres variables:$complex_{stu}$, $complexity_{stu\_BN}$, y $complex_{stu2}$:

- La variable $complex_{stu}$ representa el valor asignado a la complejidad de una HU realizado con la técnica PP de la manera tradicional por los desarrolladores principiantes.
- La variable $complex_{stu\_BN}$ representa las probabilidades resultantes con la RB para asignar la complejidad a una HU en base a las respuestas proporcionadas por los desarrolladores principiantes al contestar las preguntas propuestas en nuestro modelo.
- La variable $complex_{stu2}$ representa el valor asignado a la complejidad de una HU realizado con la técnica PP de la manera tradicional por parte de los desarrolladores principiantes una vez completada la HU.

Tabla 4.1: Resultados de la evaluación de HU por desarrolladores principiantes (sprint uno).

Caso	HU	Complejidad	Importancia	Valor en el Proyecto	$complex_{stu\_BN}$	$complex_{stu}$	$complex_{stu2}$
1	HU1	55%	84%	79%	1	5	2
2	HU1	65%	100%	86%	2	2	2
3	HU1	52%	84%	78%	1	2	3
4	HU1	59%	66%	77%	2	8	1
5	HU1	62%	66%	78%	2	2	1
6	HU2	70%	100%	88%	3	3	2
7	HU2	73%	100%	90%	3	2	2
8	HU2	62%	66%	78%	2	2	2
9	HU2	69%	66%	81%	3	3	2
10	HU2	72%	66%	80%	3	3	2
11	HU3	70%	100%	88%	3	3	2
12	HU3	65%	82%	83%	2	3	2
13	HU3	67%	100%	87%	3	1	2
14	HU3	69%	66%	81%	3	3	2
15	HU3	72%	66%	82%	3	1	2
16	HU4	62%	82%	82%	2	1	2
17	HU4	70%	100%	88%	3	1	2
18	HU4	59%	100%	84%	2	1	2
19	HU4	58%	66%	77%	2	2	1
20	HU4	58%	66%	77%	2	2	1
21	HU5	52%	82%	78%	1	2	1
22	HU5	59%	82%	80%	2	1	1
23	HU5	59%	100%	84%	2	3	1
24	HU5	69%	66%	81%	3	3	2
25	HU5	58%	66%	77%	2	3	2
26	HU6	84%	100%	94%	5	8	5
27	HU6	84%	100%	94%	5	13	5
28	HU6	69%	100%	88%	3	5	5
29	HU6	94%	82%	94%	8	8	8
30	HU6	86%	82%	91%	5	8	5
31	HU7	52%	82%	78%	1	1	1
32	HU7	52%	82%	78%	1	2	1
33	HU7	55%	100%	83%	1	3	1
34	HU7	55%	49%	72%	1	2	1
35	HU7	58%	66%	77%	2	2	1
36	HU8	66%	100%	87%	2	5	1
37	HU8	62%	82%	82%	2	2	2
38	HU8	70%	100%	88%	3	3	2
39	HU8	86%	100%	95%	5	3	3
40	HU8	86%	100%	95%	5	3	1
41	HU9	58%	66%	77%	2	3	3
42	HU9	65%	82%	83%	2	1	1
43	HU9	59%	49%	73%	2	3	2
44	HU9	75%	100%	90%	3	2	2
45	HU9	78%	100%	92%	5	5	3

Las pruebas de correlación de Spearman consideran las relaciones: $complex_{stu}$-$complex_{stu\_BN}$ para determinar la correlación de los desarrolladores principiantes antes de empezar la HU, y $complex_{stu}2$-$complex_{stu\_BN}$ para calcular la correlación una vez que se concluyó la HU. Los resultados se muestran en la Tabla 4.3, donde la columna Variables representa las relaciones antes mencionadas. Las pruebas fueron divididas por Sprint considerando el p-valor y la correlación.

Tabla 4.2: Resultados de la evaluación de HU por desarrolladores principiantes (sprint dos).

Caso	HU	Complejidad	Importancia	Valor en el Proyecto	$complex_{stu\_BN}$	$complex_{stu}$	$complex_{stu2}$
1	HU1	62%	100%	85%	2	2	2
2	HU1	65%	82%	83%	2	2	2
3	HU1	62%	100%	85%	2	2	2
4	HU1	65%	100%	86%	2	1	2
5	HU1	69%	100%	88%	3	3	2
6	HU2	88%	84%	92%	5	5	5
7	HU2	84%	100%	94%	5	5	5
8	HU2	77%	100%	91%	3	5	5
9	HU2	94%	100%	98%	8	8	5
10	HU2	94%	66%	91%	8	5	5
11	HU3	73%	100%	90%	3	1	3
12	HU3	62%	82%	82%	2	2	3
13	HU3	59%	100%	84%	2	2	3
14	HU3	65%	100%	86%	2	3	3
15	HU3	55%	100%	83%	1	1	2
16	HU4	65%	66%	79%	2	3	2
17	HU4	65%	82%	83%	2	1	2
18	HU4	69%	100%	88%	3	2	3
19	HU4	65%	82%	83%	2	2	2
20	HU4	72%	100%	89%	3	3	2
21	HU5	52%	66%	74%	1	2	2
22	HU5	77%	66%	84%	3	3	2
23	HU5	52%	49%	71%	1	3	2
24	HU5	52%	82%	78%	1	1	1
25	HU5	52%	66%	74%	1	2	1

Tabla 4.3: Correlación de Spearman de las estimaciones de desarrolladores principiantes y la RB.

	Sprint Uno		**Sprint Dos**	
Variables	**P-valor**	**Correlación**	**P-valor**	**Correlación**
$Complex_{stu}$-$complex_{stu\_BN}$	0.004	0.418	0.001	0.643
$Complex_{stu2}$-$complex_{stu\_BN}$	0.000	0.605	0.000	0.741

4.1.2. Estimaciones realizadas por profesionales

El segundo experimento se llevó a cabo con apoyo de profesionales del desarrollo de software, estos son parte de una empresa mexicana del campo de las TI. La compañía ofrece desarrollo de software personalizado, aplicaciones móviles y web, soluciones empresariales y desarrollo de bases de datos en diversas plataformas y sistemas operativos.

4.1.2.1. Resultados obtenidos por los profesionales

Los profesionales también hicieron su estimación usando PP de la manera tradicional y con el modelo propuesto con RB.

Para realizar las estimaciones a través de la RB, los profesionales respondieron las cinco preguntas para cada variable de primer nivel (Apéndice A.2), 5 preguntas por cada HU, la RB recibió como entrada las respuestas de los profesionales.

La información proporcionada por los profesionales para cada HU (Apéndice B.3) se despliega en la Tabla 4.4, en esta tabla se muestran las salidas de de la RB. Las variables de segundo nivel y la variable valor en el proyecto están representadas por su nombre en las columnas de la tabla. La columna $complex_{prof}$ representa la estimación realizada por los profesionales de la forma tradicional con PP. Los valores en la columna *complejidad* son convertidas a la escala de Fibonacci modificado en base a la Ecuación 4.1, los resultados se muestran en la columna $complex_{prof\_BN}$.

Tabla 4.4: Resultados de la evaluación de HU por profesionales.

Caso	HU	Complejidad	Importancia	Valor en el proyecto	$complex_{prof\_BN}$	$complex_{prof}$
1	HU1	100%	64%	92%	13	13
2	HU1	97%	82%	95%	8	13
3	HU1	97%	64%	91%	8	13
4	HU1	97%	64%	91%	8	13
5	HU2	97%	49%	88%	8	8
6	HU2	97%	100%	99%	8	8
7	HU2	97%	49%	88%	8	8
8	HU2	97%	66%	92%	8	8
9	HU3	100%	82%	96%	13	8
10	HU3	97%	100%	99%	8	8
11	HU3	94%	82%	94%	8	13
12	HU3	94%	82%	94%	8	8
13	HU4	87%	34%	81%	5	3
14	HU4	87%	70%	89%	5	5
15	HU4	87%	34%	81%	5	5
16	HU4	84%	66%	87%	5	5
17	HU5	84%	34%	80%	5	3
18	HU5	80%	34%	78%	5	3
19	HU5	84%	34%	80%	5	3
20	HU5	88%	34%	81%	5	3
21	HU6	73%	49%	79%	3	3
22	HU6	80%	34%	78%	5	3
23	HU6	73%	34%	76%	3	3
24	HU6	65%	34%	73%	2	3

4.1.2.2. Pruebas de hipótesis de la RB de profesionales

Fueron utilizadas dos variables principales:

- La variable $complex_{prof\_BN}$ representa las probabilidades que genera la RB para asignar la complejidad a una HU en base a las respuestas proporcionadas por los profesionales.

- La variable $complex_{prof}$ representa los valores asignados por los profesionales para asignar la complejidad a la HU por el método normal de PP.

La Prueba de Correlación de Spearman fue aplicada, se obtuvieron los resultados de la Tabla 4.4 con la cual se analiza estadísticamente las estimaciones de los profesionales.

Las hipótesis son:

- H_0: Las variables $complex_{prof}$ y $complex_{prof\_BN}$ no tienen correlación.
- H_1: Las variables $complex_{prof}$ y $complex_{prof\_BN}$ tienen correlación.

La prueba de Spearman proporcionó una correlación de 0.873 con un nivel de confianza del 95 %. El $p-valor$ fue menor al nivel de confianza $0.000 < 0.05$; por lo tanto, las variables $complex_{prof}$ y $complex_{prof\_BN}$ tienen correlación, aceptándose la hipótesis H_1.

4.2. Discusión

Este estudio propone descomponer la estimación de la complejidad en factores; tales como: experiencia, tiempo y esfuerzo, con la finalidad de reducir la subjetividad de la evaluación con PP. Este proceso es subjetivo por naturaleza, pero la inexperiencia de los equipos hace que sea más complejo el proceso de estimación. Se ha planteado la hipótesis de que la estimación de la complejidad de la BN basada en factores como el tiempo, el esfuerzo y la experiencia es similar a la estimación hecha por un experto. Los resultados de este estudio apoyan esta hipótesis.

Previamente en López-martínez et al. (2017), se analizaron las estimaciones de los desarrolladores principiantes y los resultados de la RB. La RB se basó en el conocimiento del desarrolladores principiantes, los resultados mostraron que ambas estimaciones tenían una correlación a través de la prueba de Spearman de 0.446, esto es una correlación moderada. Por otro lado, el experimento se repitió con diferentes desarrolladores principiantes y las relaciones de la RB se basaron en el conocimiento de expertos. La estimación de los desarrolladores principiantes y la RB obtuvieron una correlación de Spearman moderada de 0.418. En ambos experimentos de desarrolladores principiantes, las estimaciones fueron similares. Esto ocurre debido a que los desarrolladores principiantes no cuentan son

experiencia suficiente en el desarrollo de proyectos y sus estimaciones, por este motivo sus estimaciones se consideran menos confiables.

Es un hecho que una estimación de una HU es más exacta al concluir la HU que cuando aún no se ha iniciado. Con base en esto, la estimación realizada por los desarrolladores principiantes y por la RB antes de iniciar la HU obtuvieron una correlación de 0.418; las estimaciones realizadas por desarrolladores principiantes una vez concluida la HU obtuvo una correlación de 0.605 respecto a la estimación de la RB. Para el segundo Sprint, en la estimación de la RB se obtuvo una correlación de 0.643 con la estimación inicial de la HU y una correlación de 0.741 con la estimación final al concluir la HU (estos resultados se muestran en la Tabla 4.3). Se puede considerar que la estimación de los desarrolladores principiantes al concluir una HU es más precisa debido a la experiencia que obtienen al desarrollar esta HU. Los resultados muestran un aumento en la correlación al concluir la HU con relación a la estimación de la RB. Este aumento indica que la estimación realizada a través de la RB es más cercana a la estimación más precisa. En este punto, la estimación de la RB es mejor que la primera estimación de la complejidad de los desarrolladores principiantes.

Por otra parte, los profesionales con más experiencia estimando HU usando PP muestran mejor correlación que los desarrolladores principiantes. La mayoría de los resultados de la estimación de la RB fueron los mismos de la evaluación profesional. Al aplicar las pruebas de correlación de Spearman se obtuvo una correlación de 0.873, la RB y la estimación de los profesionales estuvieron fuertemente correlacionadas. Los factores propuestos en la RB están en la mente del experto inconscientemente cuando hacen la estimación de la complejidad. Un profesional con mucha experiencia en el desarrollo de proyectos de software estima con mayor precisión una HU porque la desarrolló previamente y sabía lo difícil o complejo que es. De la misma manera, la experiencia proporciona precaución al considerar la complejidad de una HU desconocida.

Al momento, los resultados nos llevan a establecer que las estimaciones de la RB son similares a las de los expertos profesionales. Estos hallazgos sugieren que podría ser útil para los miembros inexpertos cuando necesitan estimar la complejidad, esto ofrece la posibilidad de usar la salida de la red

para confiar en la complejidad de una HU. A través de este proceso, los estudiantes no mostrarán una tarjeta de PP con la serie Fibonacci; en lugar de eso, los desarrolladores principiantes darán respuestas a las preguntas que alimentan la red. La propuesta es un proceso de capacitación mientras que los desarrolladores novatos adquieren la experiencia y las habilidades para estimar las HUs de un proyecto.

La red maneja dos aspectos significativos, la complejidad de una HU y el valor de la HU para el proyecto. El nodo principal de la RB es el valor de la HU del proyecto, considerando las estimaciones de los profesionales que tienen más experiencia que los desarrolladores principiantes, se ha obtenido la HU que aporta más valor al proyecto. Estos fueron elegidos para considerar la complejidad y la importancia de la tarea. La $HU3$ es considerada la historia más fundamental con un 96 % probabilidad, la $HU1$ es el siguiente con 92 %. Estas dos HU son los más esenciales del proyecto, ya que le dará identidad al proyecto, por lo tanto, tenemos que poner especial atención a estas HU. Por otra parte, la $HU6$ es la menos necesaria; esto no significa que no se debe desarrollar, esta HU trae menos identidad al proyecto, podemos concentrar más recursos a otras HU antes de esta.

Capítulo 5

Conclusiones, aportaciones y trabajo futuro

5.1. Conclusiones

Este estudio presentó una estructura de conocimiento para asignar complejidad e importancia a las HU con base en RBs y PP. En lugar de tener en cuenta la complejidad como un valor único, se dividió en tres variables (experiencia, tiempo y esfuerzo). Así, la atención se da en cada variable a la vez, obteniendo una mayor precisión en la toma de decisiones de estimación por parte de los desarrolladores de software.

Teniendo en cuenta la complejidad e importancia de una HU, se pueden definir las HU más importantes del proyecto, la cuales dan identidad al proyecto, por lo que deben tener tratamiento especial para desarrollarlos adecuadamente.

Este proyecto desarrolló una estructura de conocimiento que ayuda a los desarrolladores principiantes o empresas sin datos históricos a hacer estimaciones a través de la descomposición de la complejidad. El aprendizaje se reflejará cuando los desarrolladores principiantes comiencen a correlacionarse mejor con las estimaciones hechas a través de RB, por lo que sus estimaciones serán más confiables; podemos afirmar que los desarrolladores principiantes aprendieron a estimar a través de PP. Hasta el momento, se tienen resultados prometedores que indican una alta correlación de las estimaciones profesionales en el

desarrollo de software, por lo que los principiantes puede confiar en los resultados que muestra la RB.

5.2. Aportaciones

A continuación se describen los puntos más destacados de la investigación:

- Un revisión de literatura sobre los factores más destacados en la estimación de HU con PP.
- El diseño y validación de una estructura para representar el conocimiento de personas con experiencia en la estimación de HU.
- Una modificación al método actual de estimación de PP, usando una RB como intermediario.
- Un nuevo método para estimar HU sin necesidad de información estadística.

5.3. Trabajo Futuro

Esta investigación logró varios aportaciones, sin embargo, aun quedan trabajo por ser completado con el objetivo de mejorar este proyecto:

- Definir un proyecto para desarrollarlo de principio a fin con la propuesta de este trabajo.
- El desarrollo de una aplicación móvil que pueda dar soporte a la estimaciones a través del método propuesto.
- Validar que las HU estimadas por las RB son realmente esenciales para el proyecto.

Bibliografía

R Akif and H Majeed. Issues and Challenges in Scrum Implementation. *International Journal of Scientific & Engineering Research*, 3(8):1–4, 2012. doi: 2229-5518.

Julian M Bass. How product owner teams scale agile methods to large distributed enterprises. 2014. doi: 10.1007/s10664-014-9322-z.

Paulo Henrique De Souza Bermejo, André Luiz Zambalde, Adriano Olímpio Tonelli, Samara Alyne Souza, Larissa Avelino Zuppo, and Priscila Luiz Rosa. Agile Principles and Achievement of Success in Software Development: A Quantitative Study in Brazilian Organizations. *Procedia Technology*, 16:718–727, 2014. ISSN 22120173. doi: 10.1016/j.protcy.2014.10.021. URL `http://linkinghub.elsevier.com/retrieve/pii/S2212017314002485`.

Saurabh Bilgaiyan, Samaresh Mishra, and Madhabananda Das. A Review of Software Cost Estimation in Agile Software Development Using Soft Computing Techniques. *2016 2nd International Conference on Computational Intelligence and Networks (CINE)*, pages 112–117, 2016.

Luis José Cendejas Valdéz, Carlos Arturo Vega Lebrún, and Anayansi Careta Isordia. Diseño del modelo integral colaborativo para el desarrollo ágil de software en las empresas de la zona centro-occidente en México. *Revista Electrónica Nova Scientia*, 7:133–148, 2014.

Matilde Inés Césari. *Trabajo Final de Especialidad en Tecnologías de Explotación de Información DE UNA RED BAYESIANA*. PhD thesis, 2006.

Wooi-Ping Cheah, Kyoung-Yun Kim, Hyung-Jeong Yang, Soo-Hyung Kim, and Jeong-Sik Kim. Fuzzy Cognitive Map and Bayesian Belief Network for Causal Knowledge Engineering: A Comparative Study. *The KIPS Transactions: PartB*, 15(2):147–158, 2008.

Srdjana Dragicevic, Stipe Celar, and Mili Turic. Bayesian network model for task effort estimation in agile software development. *Journal of Systems and Software*, 127:109–119, 2017. ISSN 01641212.

Veli-Pekka Eloranta, Kai Koskimies, Tommi Mikkonen, and Jyri Vuorinen. Scrum Anti-Patterns – An Empirical Study. *2013 20th Asia-Pacific Software Engineering Conference (APSEC)*, 1:503–510, 2013. ISSN 1530-1362.

Pablo Felgaer. *Optimización de Redes Bayesianas basado en Técnicas de Aprendizaje por Inducción*. PhD thesis, 2004.

Taghi Javdani Gandomani and Mina Ziaei Nafchi. An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach. *Journal of Systems and Software*, 107:204–219, 2015. ISSN 01641212. doi: 10.1016/j.jss.2015.06.006. URL `http://linkinghub.elsevier.com/retrieve/pii/S0164121215001223`.

Yu Gao. Research on the rule of evolution of software development process model. *2010 2nd IEEE International Conference on Information Management and Engineering*, pages 466–470, 2010. doi: 10.1109/ICIME.2010.5477884. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5477884`.

Stein Grimstad and Magne Jørgensen. A framework for the analysis of software cost estimation accuracy. *Proceedings of the 2006 ACM/IEEE international symposium on International symposium on empirical software engineering - ISESE '06*, page 58, 2006. doi: 10.1145/1159733.1159745.

H. Karna and S. Gotovac. Estimating software development effort using Bayesian networks. pages 229–233, 2015.

Pawandeep Kaur and Rupinder Singh. A Proposed Framework for Software Effort Estimation Using the Combinational Approach of Fuzzy Logic and Neural Networks. *International Journal of Hybrid Information Technology*, 8(10):73–80, 2015. ISSN 1738-9968. doi: 10.14257/ijhit.2015.8.10.07.

Janeth López-martínez, Alan Ramírez, Reyes Juárez-ramírez, Guillermo Licea, and Samantha Jiménez. User stories complexity estimation using Bayesian networks for inexperienced developers. *Cluster Computing*, 2017. ISSN 1573-7543. doi: 10.1007/s10586-017-0996-z.

V. Mahnic and I. Rozanc. Students' perceptions of Scrum practices. pages 1178–1183, 2012.

Viljan Mahnič and Tomaž Hovelja. On using planning poker for estimating user stories. *Journal of Systems and Software*, 85(9):2086–2095, 2012. ISSN 01641212. doi: 10.1016/j.jss.2012.04.005.

Antonio Martel. *Gestión práctica de proyectos con Scrum: Desarrollo de software ágil para el Scrum Master*. 3rd. edition, 2014. ISBN 978-1517192365.

Emilia Mendes. Knowledge representation using Bayesian networks — A case study in Web effort estimation. *Proceedings of the World Congress on Information and Communication Technologies*, pages 612–617, 2011. doi: 10.1109/WICT.2011.6141315.

A. Mitre-Hernández Hugo, Ortega-Martínez Edgar, and Lemus-Olalde Cuauhtémoc. Estimación y control de costos en métodos ágiles para desarrollo de software: un caso de estudio. *Ingeniería, Investigación y Tecnología*, 15(3):403–418, 2014. ISSN 14057743. doi: 10.1016/S1405-7743(14)70350-6. URL `http://www.sciencedirect.com/science/article/pii/S1405774314703506`.

Kjetil Moløkken-Østvold, Nils Christian Haugen, and Hans Christian Benestad. Using planning poker for combining expert estimates in software projects. *Journal of Systems and Software*, 81(12): 2106–2117, 2008. ISSN 01641212. doi: 10.1016/j.jss.2008.03.058.

a B Nassif, L F Capretz, and D Ho. Estimating Software Effort Based on Use Case Point Model

Using Sugeno Fuzzy Inference System. *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*, pages 393–398, 2011. ISSN 1082-3409. doi: 10.1109/ICTAI.2011.64.

Sven Overhage and Sebastian Schlauderer. Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects. In *2012 45th Hawaii International Conference on System Sciences*, pages 5452–5461. IEEE, jan 2012. ISBN 978-1-4577-1925-7. doi: 10.1109/HICSS.2012.387.

Mohd Owais. Effort , Duration and Cost Estimation in Agile Software Development. pages 1–5, 2016. doi: DOI:10.1109/IC3.2016.7880216.

Daniel Pauly and Dirk Basten. Do Daily Scrums Have to Take Place Each Day? A Case Study of Customized Scrum Principles at an E-Commerce Company. *Hawaii International Conference on System Sciences*, pages 5074–5083, 2015. doi: 10.1109/HICSS.2015.601.

R Popli and N Chauhan. Cost and effort estimation in agile software development. *Optimization, Reliabilty, and Information Technology (ICROIT), 2014 International Conference on*, pages 57–61, 2014a. doi: 10.1109/ICROIT.2014.6798284.

R Popli and N Chauhan. Agile estimation using people and project related factors. *Computing for Sustainable Global Development (INDIACom), 2014 International Conference on*, pages 564–569, 2014b. doi: 10.1109/IndiaCom.2014.6828023.

Rashmi Popli and Naresh Chauhan. A sprint-point based estimation technique in scrum. *Proceedings of the 2013 International Conference on Information Systems and Computer Networks, ISCON 2013*, pages 98–103, 2013. doi: 10.1109/ICISCON.2013.6524182.

Florian Raith, Ingo Richter, Robert Lindermeier, and Gudrun Klinker. Identification of Inaccurate Effort Estimates in Agile Software Development. *2013 20th Asia-Pacific Software Engineering Conference (APSEC)*, pages 67–72, 2013. doi: 10.1109/APSEC.2013.114.

Luis Rincón. *INTRODUCCIÓN A LA PROBABILIDAD*. 2014.

D Rodríguez and Javier Dolado. Redes Bayesianas en la Ingeniería del Software. *Cc.Uah.Es*, pages 1–21, 2007. URL http://www.cc.uah.es/drg/b/RodriguezDolado.BBN.2007.pdf.

Roger S. Pressman. *Ingenieria Del Software*. México, D. F, 7 edicion edition, 2010. ISBN 9786071503145. doi: 10.1017/CBO9781107415324.004.

M. Romero, A. Vizcaino, and M. Piattini. Teaching Requirements Elicitation within the Context of Global Software Development. In *Computer Science (ENC), 2009 Mexican International Conference on*, pages 232 – 239, México City, 2009. IEEE. ISBN 978-1-4244-5258-3. doi: 10.1109/ENC.2009.29.

Kenneth S. Rubin. *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley, Ann Arbor, MI., 1st edition, 2012. ISBN 9780137043293.

Allan Rueda Gutierrez. *PROPUESTA DE UNA GUÍA PARA INTERPRETAR LOS PROCESOS DE MOPROSOFT DE LA CATEGORÍA DE OPERACIÓN USANDO UNA COMBINACION DE METODOS AGILES*. PhD thesis, Instituto Politécnico Nacional, 2010. URL http://www.cic.ipn.mx/sitioCIC/images/sources/cic/tesis/B020892.pdf.

K Schwaber and Jeff Sutherland. The scrum guide. *Scrum. org, October*, 2(July):17, 2011. ISSN 00195847. doi: 10.1053/j.jrn.2009.08.012.

I Sonia and L Pedro. Implementación de SCRUM en el diseño del proyecto del Trabajo Final de Aplicación Implementing SCRUM in design of the Trabajo Final de Aplicación. *Scientia Et Technica*, 19:413–418, 2014.

P K Suri and Pallavi Ranjan. Comparative Analysis of Software Effort Estimation Techniques. *International Journal of Computer Applications*, 48(21):975–8887, 2012.

Jeff Sutherland, Neil Harrison, and Joel Riddle. Teams That Finish Early Accelerate Faster: A Pattern Language for High Performing Scrum Teams. *2014 47th Hawaii International Conference on System Sciences*, pages 4722–4728, 2014. ISSN 15301605. doi: 10.1109/HICSS.2014.580.

R. Tamrakar and M. Jørgensen. Does the use of Fibonacci numbers in Planning Poker affect effort estimates? In *16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012)*, pages 228–232, 2012. ISBN 978-1-84919-541-6. doi: 10.1049/ic.2012.0030. URL http://digital-library.theiet.org/content/conferences/10.1049/ic.2012.0030.

C. J. Torrecilla-Salinas, J. Sedeño, M. J. Escalona, and M. Mejías. Estimating, planning and managing Agile Web development projects under a value-based perspective. *Information and Software Technology*, 61:124–144, 2015. ISSN 09505849. doi: 10.1016/j.infsof.2015.01.006. URL http://dx.doi.org/10.1016/j.infsof.2015.01.006.

E.H. Uribe and L.E.V. Ayala. Del Manifiesto Ágil, sus valores y sus principios. *Redalyc.Uaemex.Mx*, (34):381–385, 2007. URL http://redalyc.uaemex.mx/redalyc/html/849/84934064/84934064{_}1.html.

Hind Zahraoui, Mohammed Abdou, and Janati Idrissi. Adjusting Story Points Calculation in Scrum Effort & Time Estimation. In *Intelligent Systems: Theories and Applications (SITA), 2015 10th International Conference on*, pages 1–8, Rabat, Morocco, 2015. IEEE. ISBN 978-1-5090-0220-7. doi: 10.1109/SITA.2015.7358400.

Fatemeh Zare, Hasan Khademi Zare, and Mohammad Saber Fallahnezhad. Software effort estimation based on the optimal Bayesian belief network. *Applied Soft Computing*, 2016. ISSN 15684946. doi: 10.1016/j.asoc.2016.08.004.

Apéndices

Apéndice A

Encuestas

A.1. Encuesta para validar factores

- **Q1. ¿Qué tan de acuerdo está usted con que la EXPERIENCIA del desarrollador es un factor importante para asignar complejidad a una HU en Scrum?**

  Experiencia del equipo: El desarrollo de trabajos previos similares aporta calidad y rapidez al término de una HU. Así como la asignación de complejidad de la tarea será más precisa.

1	2	3	4	5	6	7
Totalmente en desacuerdo						Totalmente de Acuerdo

- **Q2. ¿Qué tan de acuerdo está usted con que el TIEMPO es un factor importante para asignar complejidad a una HU en Scrum?**

  Tiempo: Se refiere que tan laboriosa puede ser la tarea, aun cuando esta sea fácil de desarrollar. Puede ser una HU sencilla pero con muchos detalles que hacen que se dedique mucho tiempo.

1	2	3	4	5	6	7
Totalmente en desacuerdo						Totalmente de Acuerdo

- **Q3. ¿Qué tan de acuerdo está usted con que el ESFUERZO es un factor importante para asignar complejidad a una HU en Scrum?**

Esfuerzo mental: Se refiere al uso de algoritmos que requieren lógica computacional elevada. Por tanto la creación o uso de esta clase de algoritmo requiere mucho grado de concentración.

1	2	3	4	5	6	7
Totalmente en desacuerdo						Totalmente de Acuerdo

- **Q4. ¿Qué tan de acuerdo está usted con que la PRIORIDAD es un factor significativo para asignar importancia a una HU en Scrum?**

  Prioridad: Se refiere a la importancia de la tarea debido a que otras tareas dependen de esta. Las más importantes deben ir en los primeros sprint.

1	2	3	4	5	6	7
Totalmente en desacuerdo						Totalmente de Acuerdo

- **Q5. ¿Qué tan de acuerdo está usted con que el VALOR DE LA HU es un factor significativo para asignar importancia a una HU en Scrum?**

  Valor de la HU: Se refiere a la cantidad de ingresos que se podrían ganar o perder por desarrollar o no la historia de Usuario. (También puede referirse a que tan completo queda el sistema con o sin esa funcionalidad implementada con la tarea en cuestión).

1	2	3	4	5	6	7
Totalmente en desacuerdo						Totalmente de Acuerdo

- **Q6. La COMPLEJIDAD y la IMPORTANCIA hacen que la HISTORIA DE USUARIO sea ESENCIAL PARA EL PROYECTO**

  Esencia de la HU en el proyecto. Aquellas tareas que tienen alta complejidad y alta importancia representan en mayor medida la calidad que podría obtener el proyecto, por tal razón se les debe prestar mayor atención.

1	2	3	4	5	6	7
Totalmente en desacuerdo						Totalmente de Acuerdo

A.2. Preguntas para estimar a través de la RB

- ¿Cuál es tu EXPERIENCIA para determinar el nivel de complejidad de esta tarea?

  Experiencia del equipo: El desarrollo de trabajos previos similares aporta calidad y rapidez al término de una HU. Así como la asignación de complejidad de la tarea será más precisa.

Muy Mala	Nunca he desarrollado un historia similar
Mala	He desarrollado alguna vez una historia similar, sin embargo, no recientemente
Regular	he desarrollado una historia similar recientemente
Buena	he desarrollado dos o más historias similares
Muy Buena	Desarrollo este tipo de historias continuamente

- ¿Cuánto TIEMPO consideras para esta historia de usuario?

  Tiempo: Se refiere que tan laboriosa puede ser la tarea, aún cuando esta sea fácil de desarrollar. Puede ser una HU sencilla pero con muchos detalles que ocasiona se dedique mucho tiempo.

Muy Poco	Menos de 3 horas
Poco	De 3 a 6 horas
Ni mucho, ni poco	De 6 a 9 horas
Mucho	De 9 a 12 horas
Demasiado	Más de 12 horas

- ¿Cuánto ESFUERZO consideras para esta historia de usuario?

  Esfuerzo mental: Se refiere al uso de algoritmos que requieren lógica computacional elevada. Por tanto la creación o uso de esta clase de algoritmo requiere mucho grado de concentración.

Muy Poco	Algoritmos sencillas que se solucionan con herramientas de terceros
Poco	Algoritmos simples con poca lógica y razonamiento
Ni mucho, ni poco	Algoritmos de mediana complejidad
Mucho	Algoritmos complejos con mucha lógica y razonamiento
Demasiado	Algoritmos complejos con técnicas que necesitan ser aprendidas

- ¿Cuánta PRIORIDAD tiene la historia de usuario para determinar la importancia en el proyecto?

  Prioridad: Se refiere a la importancia de la tarea debido a que otras tareas dependen de esta. Las más importantes deben ir en los primeros sprint.

Poco	Historia de usuario independiente
Regular	Esta historia de usuario es base para una o dos historias mas
Mucho	Esta historia de usuario es base para tres o más historias

- ¿Qué tanto VALOR DE TAREA tiene la historia de usuario para determinar la importancia en el proyecto?

  Valor de la HU: Se refiere a la cantidad de ingresos que se podrían ganar o perder por desarrollar o no la historia de Usuario. (También puede referirse a que tan completo queda el sistema con o sin esa funcionalidad implementada con la tarea en cuestión).

Poco	Historia con poco impacto en el valor del productos
Regular	Historia no esencial pero aporta un valor agregado significante
Mucho	Tareas esenciales del proyecto, brindan identidad al proyecto

Apéndice B

Historias de usuario

B.1. Historias de usuario de estudiantes Sprint 1

ID HU	Descripción	Alias
HU1	Quiero visualizar los encabezados del archivo a convertir, con la finalidad de realizar operaciones sobre ellos	Visualizar encabezados
HU2	Quiero decidir si una columna va a ser usada o no y el tipo de campo que será con el fin utilizarla en el resultado final	Clasificación opcional
HU3	Necesito obtener una clasificación de las variables detectadas por el sistema ya sean categóricas o continuas, con la finalidad de poder realizar futuras operaciones sobre ellas	Clasificación automática de columnas
HU4	Necesito visualizar los encabezados de una tabla, aquellos, contendrán una palabra en específico (ID), con la finalidad de organizar los datos del conjunto de archivos que importaré	Detectar ID
HU5	Necesito que muestre campos por default para cada uno de los datos que voy a utilizar, pero que me sea posible modificarlos, con el fin de ofrecer una sugerencia de clasificación	Modificar manualmente tipos de datos
HU6	Necesito poder adjuntar archivos .csv, con la finalidad de anexar datos a mi archivo final	Anexar CSV
HU7	Quiero poder normalizar los valores de una columna con la escala que yo decida, con la finalidad de que los datos sean más fáciles de visualizar	Normalizar valores
HU8	Quiero poder designar una sola variable objetivo, que no se binariza, la cual sera la ultima que haya seleccionado (agregar)	Variable Objetivo
HU9	Quiero que los valores nulos de mis datos continuos sean reemplazados por el promedio de los valores del conjunto	Reemplazar nulos continuos

B.2. Historias de usuario de estudiantes Sprint 2

ID Historia de Usuario	Descripción	Alias
HU1	Necesito obtener una clasificación de las variables detectadas por el sistema ya sean categóricas o continuas, con la finalidad de poder realizar futuras operaciones sobre ellas	Clasificación automática de columnas
HU2	Necesito poder adjuntar archivos .csv, con la finalidad de anexar datos a mi archivo final	Anexar CSV
HU3	Quiero poder designar una sola variable objetivo, que no se binariza, la cual sera la ultima que haya seleccionado (agregar)	Variable objetivo
HU4	Quiero que los valores nulos de mis datos continuos sean reemplazados por el promedio de los valores del conjunto	Reemplazar nulos continuos
HU5	Quiero visualizar las estadísticas de los datos numéricos, como lo es la media, mínimo, máximo y desviación estándar	Datos Estadísticos

B.3. Historias de usuario de profesionales

ID Historia de Usuario	Descripción	Alias
HU1	Como Product Owner necesito que se hagan los métodos faltantes para cuda para optimizar el reconocimiento de gestos	Desarrollo en Cuda de métodos faltantes
HU2	como Product Owner necesito que se haga refactor del modulo de detección de voz para poder actuar mas rápido en caso de una incidencia	Mejora detección de voz
HU3	Como desarrollador necesito consumir el API de reconocimiento de voz del interlocutor para poder identificar personas de interés	Consumo de API de reconocimiento
HU4	Como encargado de seguridad necesito que se implemente la detección de armas de fuego para poder actuar en tiempo en caso de un incidente	Detección de armas de fuego
HU5	Como Product Owner necesito que se implemente la arquitectura de retransmisión de vídeo por medio de RTSP para poder ver las señales de las diferentes cámaras IP desde web	Implementación inicial de retransmisión de vídeo

Apéndice C

Publicaciones

C.1. Publicaciones JCR

- User Stories Complexity Estimation using Bayesian Networks for Inexperienced Developers

  Autores: Janeth López-Martínez, Alan Ramírez-Noriega, Reyes Juárez-Ramírez, Guillermo Licea, Samantha Jiménez

  Jounal (JCR) : Cluster Computing. Factor de Impacto: 1.514 Q2

  Special Issue "SI: Parallel and Distributed Knowledge Discovery (PDKD)"

C.2. Publicaciones en Libros

- Estimating User Stories' complexity and importance in Scrum with Bayesian Networks

  Autores :Janeth López-Martínez, Alan Ramírez-Noriega, Reyes Juárez-Ramírez, Guillermo Licea, Raúl Navarro-Almanza

  Libro: Recent Advances in Information Systems and Technologies,

  Advances in Intelligent Systems and Computing 569

- Towards a Compilation of Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review.

  Autores: Janeth López-Martínez, Reyes Juárez-Ramírez, Carlos Huertas, Samantha Jiménez.

  Libro : Software Engineering: Methods, Modeling, and Teaching, vol. 4

C.3. Publicaciones en conferencia

- Analysis of Planning Poker Factors between University and Enterprise

  Autores: Janeth López-Martínez, Reyes Juárez-Ramíırez, Alan Ramírez-Noriega, Guillermo Licea

  Conferencia: 5th International Conference in Software Engineering Research and Innovation CONISOFT 2017

- Problems in the Adoption of Agile-Scrum Methodologies A Systematic Literature Review.

  Autores: Janeth López-Martínez, Reyes Juárez-Ramírez, Carlos Huertas, Samantha Jiménez.

  Conferencia: 4th International Conference in Software Engineering Research and Innovation CONISOFT 2016

- Using Bayesian Networks to Obtain the Task ' s Parameters for Schedule Planning in Scrum.

  Autores: Alan Ramírez-Noriega, Reyes Juarez-Ramirez, Raul Navarro, Janeth López-Martínez.

  Conferencia: 4th International Conference in Software Engineering Research and Innovation CONISOFT 2016

- An Ontology of the Object Orientation for Intelligent Tutoring Systems.

  Autores: Ramírez-Noriega, A., Juárez-Ramírez, R., Jiménez, S., Inzunza, S., Navarro, R., & López-Martínez, J. (2017).

Conferencia: 2017 5th International Conference in Software Engineering Research and Innovation (pp. 1–10). Merida, Yucatan, México: IEEE.