CASO DE ESTUDIO SOBRE APROPIACIÓN DE *SCRUM* EN EMPRESAS QUE HAN ADOPTADO *CMMI*

Silvia Isabel Lozano Argel

Universidad EAFIT
Escuela de Ingeniería
2013

UNIVERSIDAD EAFIT

PROYECTO DE GRADO PARA OPTAR AL TÍTULO DE MAGÍSTER EN INGENIERÍA

CASO DE ESTUDIO SOBRE APROPIACIÓN DE *SCRUM* EN EMPRESAS QUE HAN ADOPTADO *CMMI*

Autor:
Silvia Isabel LOZANO ARGEL

Supervisor:
Dra. Raquel ANAYA

Departamento de Informática y Sistemas
Escuela de Ingeniería
Universidad EAFIT
Noviembre 2013

Nota de aceptación

_____

_____

Presidente del jurado

_____ Rocío Arango _____

Jurado

_____Rocío Arango_____

Jurado

_____

Medellín (27, Noviembre, 2013)

Agradecimientos

En primer lugar deseo agradecer a Dios por darme la vida y salud cada día, permitiéndome cumplir los deseos de mi corazón.

Agradezco a mis padres, que me criaron en un ambiente amoroso y me dieron la mejor herencia posible: la educación. A pesar de la distancia que nos separa, ellos siempre me han dado coraje y apoyo durante mi vida. A mis hermanos que son mi roca firme durante las tormentas de mi vida.

Agradezco a mi esposo y eterna alma gemela Hernán Sánchez quien me apoyó con incondicional amor para terminar mi maestría. El y mi hijo Alejandro me han dado una nueva dimensión de la vida. Sin su amor y apoyo moral, este estudio no hubiera alcanzado sus frutos.

Agradezco a mi asesora, Dra. Raquel Anaya, su valiosa guía, sugerencias y consejos en lo profesional y lo personal.

Agradezco a la universidad EAFIT y sus profesores que han aportado tanto a mi formación en ingeniería.

Agradezco a todas las empresas de desarrollo y consultoría que me colaboraron con el desarrollo de mi investigación; así como a todos los profesionales y colegas que compartieron sus experiencias, ideas y valioso tiempo.

Finalmente, agradezco a Ruta N, por ser un participe clave para realizar este estudio experimental en las empresas de Medellín.

Contenido

Índice de Figuras

Índice de Tablas

Lista de Abreviaturas

AUP Agile Unified Process

ASD Adaptive Software Development

CMMI Capability Maturity Model Integration

DAD Distributed Agile Development

DSDM Dynamic Systems Development Method

FDD Feature-Driven Development

GQM Goal-Question-Metric

GT Grounded Theory

P-CMMI People Capability Maturity Model

QIP Quality Improvement Paradigm

TDD Test-Driven Development

XP EXtreme Programming

Resumen

Actualmente la industria de software local se enfrenta a una situación interesante: De una parte se evidencia un auge en la adopción de las prácticas ágiles, que buscan encontrar mejores formas de trabajo de los equipos de desarrollo de software y, de otra parte, algunas de estas industrias, ya han realizado proyectos de mejora de procesos utilizando CMMI como el modelo referente, e incluso algunas de estas compañías cuentan con valoraciones oficiales de madurez en niveles 2, 3 y superiores.

El presente trabajo analiza esta situación con el propósito de conocer como ha sido la adopción de estas prácticas ágiles en empresas de desarrollo que anteriormente han adoptado prácticas orientadas a planes como CMMI o PSP/TSP.

En algunos casos, las empresas buscan convivir con ambos enfoques entendiendo que se debe elegir la práctica dependiendo del contexto o teniendo la apertura de realizar una mezcla de prácticas tradicionales y ágiles, por otro lado también hay empresas que buscan adoptar solo las prácticas ágiles de manera estricta para cualquier tipo de proyecto quizá buscando agilidad.

El objetivo de este trabajo es acumular, estructurar y divulgar información veraz sobre los retos y logros que las empresas de desarrollo de software han tenido al adoptar prácticas ágiles como *Scrum*, dentro del marco de un proyecto liderado por Ruta N (organismo que promueve la transferencia tecnológica en la ciudad de Medellín) en el que participaron un total de 10 empresas de la ciudad. Con el apoyo de un método para estudio experimental de tipo cualitativo (Grounded Theory), se busca identificar una teoría alrededor de los datos recabados para identificar cuáles han sido los retos iníciales de la adopción. Estos resultados se complementan y validan con las respuestas a una encuesta realizada meses después sobre la evolución de la adopción de la práctica ágil *Scrum* en las empresas participantes del estudio. De este modo podemos identificar un compendio de retos,

lecciones aprendidas y beneficios encontrados en este tipo de adopciones de prácticas que a diferencia de ser opuestas se complementan entre sí.

Dentro de los resultados alcanzados en este trabajo, se destacan los siguientes:
Los retos y dudas criticas generadas durante el proceso de apropiación de prácticas ágiles (*Scrum)* en empresas de desarrollo de software que han adoptado CMMI, resultaron organizados alrededor de 3 categorías principales: los valores establecidos en el manifiesto ágil, los elementos de *Scrum* y las prácticas complementarias a *Scrum*.

El estudio también proporciona algunas recomendaciones que pueden ayudar a las organizaciones a evitar obstáculos en la adopción de *Scrum* dentro de sus prácticas de desarrollo de software al integrarlo con prácticas orientadas a planes. Además, la descripción detallada de la manera como se realizó un estudio In-Vivo aplicando Grounded Theory, sirve de referencia para aquellos lectores que deseen aplicar este método en estudios de tipo cualitativo.

PRACTICAS ÁGILES, CMMI, *SCRUM*, ADOPCIÓN, RETOS, LECCIONES APRENDIDAS.

Capítulo 1. Introducción

1.1. Motivación

La productividad y la competitividad que demanda actualmente el mercado a las organizaciones de software, les generan grandes retos. Estos retos son una de las causas por la que los procesos de desarrollo de software deben también estar en constante evolución y adopción de nuevas prácticas.

Tanto las prácticas ágiles como las prácticas orientadas a planes hacen frente a los desafíos que tienen las empresas en el manejo de la complejidad, la innovación tecnológica y el cambio de requisitos pero con diferentes enfoques y perspectivas en los métodos aplicados [16].

Las prácticas orientadas a planes han venido aplicándose tradicionalmente en las empresas del sector, buscando manejar la complejidad del software a través de la disciplina en los procesos, planeación a largo plazo y énfasis en prácticas de arquitectura para fortalecer la escalabilidad, mantenibilidad y confiabilidad. Buscando flexibilidad y adaptabilidad, durante los últimos años, las prácticas ágiles de desarrollo de software empiezan a tener un gran auge y se convierten cada vez más en una alternativa importante para la industria al enfrentar la evolución dinámica de los requisitos, promover la autogestión del equipo de desarrollo y lograr una mejor colaboración con el cliente durante el proyecto [11,15].

La mejora de los procesos organizacionales se realiza a través de proyectos de mejora los cuales suponen gran inversión en dinero, tiempo y recursos para las empresas. La mayoría de estas empresas no están preparadas para asumir solas estas iniciativas de mejora de proceso [10], por esto se hace necesario el apoyo de las asociaciones y entidades de gobierno tanto a nivel nacional como regional que reconocen la importancia de las TIC como un renglón competitivo de la economía.

Como ejemplos de estos programas de apoyo podemos mencionar:

Las tres fases del proyecto para la adopción del modelo CMMI en Pymes del sector software y servicios conexos que fueron impulsados por Colciencias, SENA, PROEXPORT desde el año 2005 y cuya última fase (2008-2010) presentó un ejercicio de articulación universidad-empresa-estado [19].
El programa promovido por Ruta N [http://www.rutanmedellin.org] para la transferencia de conocimiento y apropiación de *Scrum* en un grupo de empresas de desarrollo de software de Medellín.
La población para este estudio fue tomada del conjunto de empresas participantes en los programas de apoyo anteriormente presentados ya que dentro de los requisitos de postulación que solicitó Ruta N, se encuentra la disposición para aplicar la metodología en el desarrollo de un caso práctico o proyecto informático [22].

Los estudios experimentales son una estrategia clave en el campo de ingeniería de software, para obtener evidencias de aplicación en contextos reales de nuevas prácticas. Existen diversas investigaciones que han tratado el reto de la convivencia entre las prácticas orientadas a planes como CMMI y las prácticas ágiles como *Scrum* pero de acuerdo con la bibliografía y trabajos relacionados consultados para este estudio ninguna investigación de esta clase se ha realizado en el ámbito nacional.

Desarrollar este tipo de estudios experimentales en la ciudad de Medellín nos permitirá acumular y divulgar información veraz sobre los retos y logros que las empresas de desarrollo de software han tenido al adoptar prácticas ágiles como *Scrum* y del impacto que pueden tener programas como el de Ruta N en dicha adopción, buscando anticiparse y acortar el camino de apropiación de las prácticas ágiles en un entorno en el que anteriormente han trabajado con prácticas orientadas a planes.

A través de este trabajo se espera contribuir al cuerpo de conocimientos existente en la industria sobre la adopción de *Scrum*, en empresas de desarrollo de software. Las empresas que fueron seleccionadas en el programa de Ruta N, fueron a la vez empresas que

anteriormente se habían beneficiado de programas de mejora para adopción de CMMI y por lo tanto surge la connotación especial del estudio en el contexto de empresas que ya han adoptado modelos de madurez como CMMI.

La principal motivación de este estudio es identificar y mostrar los resultados de la transferencia de conocimiento y apropiación de *Scrum* realizado por las empresas vinculadas al programa liderado por Ruta N.

1.2. Objetivos

El objetivo de este trabajo es identificar los retos y lecciones aprendidas que emergen durante el proceso de apropiación de prácticas ágiles y de la integración de estas prácticas ágiles con prácticas orientadas a planes ya institucionalizadas.

Definiendo formalmente el objetivo de acuerdo a [36] se tendría lo siguiente:
El **objetivo de este estudio** es analizar el proceso de apropiación de prácticas ágiles específicamente *Scrum* en empresas de desarrollo de software o servicios relacionados. **Con el propósito de** caracterizar los retos, logros y lecciones aprendidas, **con respecto a**l uso de prácticas ágiles y su relación con las prácticas orientadas a planes (CMMI) ya adoptadas, **desde el punto de vista de** los investigadores, patrocinadores, consultores y miembros de los equipos de los proyectos bajo observación **en el contexto de** las empresas participantes en el programa piloto de transferencia del conocimiento y apropiación de la metodología ágil *Scrum*.

1.3. Preguntas de investigación

- ¿Cuáles son los retos durante la apropiación de prácticas ágiles (*Scrum*) en empresas de desarrollo de software que han adoptado CMMI?

- ¿Cuáles fueron las prácticas ágiles adoptadas por los diferentes equipos de trabajo en las empresas? ¿Cuál es el nivel de adopción de las prácticas?
- ¿Los retos identificados y caracterizados durante la apropiación de las prácticas ágiles son similares a los retos encontrados en la literatura?
- ¿Cuáles son los principales beneficios y lecciones aprendidas percibidas por los profesionales de las empresas participantes del estudio?

1.4. Metodología de trabajo

Para responder a las preguntas de investigación, se condujeron dos estudios, un estudio cualitativo aplicando algunos elementos del enfoque Grounded Theory que cubrió un periodo de 5 meses de observación (Noviembre 2012 – Abril 2013), recopilando notas e información del trabajo de campo durante las tres fases del programa piloto de Ruta N, en 8 empresas dedicadas al desarrollo de software en Medellín - Colombia.

Luego se realizó otro estudio cualitativo durante los meses de mayo y junio de 2013 basado en una encuesta que fue diligenciada por empleados de las empresas pertenecientes a la población objetivo. Esta encuesta contenía preguntas relacionadas con la adopción de prácticas ágiles, especialmente *Scrum*, cuyas respuestas complementan los hallazgos del estudio.

A continuación se procedió a realizar el análisis de los datos, con la técnica de comparación de conceptos en la cual se basa el Grounded Theory, intentando crear una teoría alrededor de las dudas, retos y lecciones aprendidas durante el proceso de adopción de *Scrum* y de esta manera responder las preguntas de investigación formuladas anteriormente.

Por último se realizó la consolidación de las conclusiones encontradas en el estudio.

1.5. Estructura del documento

Además del capítulo de Introducción, este documento está organizado en otros cinco capítulos, de la siguiente manera:

Capítulo 2. Marco teórico: Se presentan los trabajos relacionados con la adopción de prácticas ágiles en empresas de desarrollo de software.

Capítulo 3. Metodología de investigación: se describe el diseño de la investigación, el método de investigación cualitativa Grounded Theory (GT) y como aplica a la ingeniería del software, la recolección y el análisis de datos.

Capítulo 4. Análisis de datos recolectados: Se detalla cómo se analizan los datos recolectados durante la observación y posterior a la observación.

Capítulo 5. Conclusiones y Trabajos futuros: se presentan los hallazgos, seguidos de una discusión sobre los mismos.

Capítulo 2. Marco teórico

Debido a que los grupos objetivo de este trabajo de maestría son los equipos de *Scrum* y los equipos de desarrollo que usan prácticas orientadas a planes, este capítulo presentará una descripción de los dos temas principales discutidos a través del trabajo de maestría; *Scrum* y CMMI.

Esta sección también presenta algunos aspectos de la naturaleza del proceso software, el papel de las prácticas, las perspectivas y tendencias en la adopción de prácticas para la mejora del mismo, algunos enfoques de desarrollo ágil. Así como la explicación del método de investigación principal usado para generar los resultados y conclusiones del estudio.

2.1. La naturaleza del proceso software

El proceso de desarrollo del software se define según la IEEE 610 [25] como “el proceso por el cual las necesidades de usuario son trasladadas a un producto software”.

Por su parte, el Instituto de Ingeniería del Software (Software Engineering Institute - SEI) de la Universidad Carnegie Mellon define el proceso de desarrollo del software como un “conjunto de actividades, métodos, prácticas y transformaciones que las personas usan para desarrollar y mantener software y los productos de trabajo asociados (planes de proyecto, diseño de documentos, código, pruebas y manuales de usuario)” [44].

Teniendo en cuenta las anteriores definiciones, el reto está en analizar y decidir adecuadamente, cuáles son las prácticas y actividades que deben ser parte de un proceso

software, de tal manera que permita entregar un producto con alta calidad, generando satisfacción en los clientes y con beneficios para el proveedor.

Es bien conocida la premisa de la gestión de procesos, “la calidad de un sistema o producto está muy influenciada por la calidad del proceso empleado para desarrollarlo y mantenerlo” [34] y recogiendo esta premisa, una de las principales líneas de trabajo para la mejora de la calidad de los productos software es el estudio y mejora de los procesos mediante los cuales el software es desarrollado y mantenido [31].

Esta sección busca analizar las principales características del proceso software con la intención de entender su naturaleza especial y poder identificar la mejor manera de adoptar prácticas como parte de su mejora.

La complejidad del proceso software. Según la propuesta de Hanakawa [27], la complejidad del proceso software se ve influenciado durante el progreso de un proyecto debido a los procesos adicionales necesarios para desarrollar las solicitudes repentinas o urgentes del cliente. La suma total de estos procesos adicionales es una manera de calcular la complejidad de un proceso software. Y este valor de complejidad permite que los gerentes puedan decidir cuándo hacer el “refactoring” de un proceso, adicionando u omitiendo actividades en su plan de proyecto [27], esto significa que más que seguir una metodología rígida, el proceso de software debe ser flexible o adaptable a las condiciones del contexto.

El desarrollo de software no es un proceso de producción típico. El proceso software no es un proceso de producción típico, debido a que está dirigido por excepciones, se ve muy determinado por circunstancias impredecibles y tiene peculiaridades que distinguen un proceso de los demás. En el caso del software a la medida las condiciones del contexto para cada cliente guían el proceso de producción del software lo cual hace difícil pensar en una línea de producción.

El proceso de desarrollo no es completamente un proceso creativo ni de ingeniería "pura". El proceso software no es completamente creativo, pues tiene algunas partes que pueden ser descritas con detalle y algunos procedimientos han sido impuestos previamente, por otro lado tampoco es un proceso de ingeniería "pura", porque aún no usa totalmente la ciencia experimental para apoyar sus decisiones, dependen demasiado de mucha gente, el diseño y la producción no están claramente diferenciados, y los presupuestos, calendarios y la calidad no pueden ser planificados de forma suficientemente confiable [31].

Se podría quizás afirmar que esta doble naturaleza del proceso software como arte y ciencia podría influenciar la selección del conjunto de prácticas para el proceso y que sea necesario que se componga de prácticas ágiles que permitan la adaptación a los cambios, así como de prácticas robustas que permitan definir planes a largo plazo y lidiar con la complejidad de grandes equipos.

Boehm y Turner en [47] sugieren una marcada diferencia entre enfoques orientados a planes y enfoques ágiles que es preciso balancear y plantean 5 dimensiones críticas que pueden ser usadas para describir una organización o un proyecto en términos de sus características que luego permitirá definir si tendrá más éxito con prácticas disciplinadas o ágiles. Se plantea que las prácticas orientadas a planes son enfoques disciplinados, más exitosos en contextos donde el tamaño del equipo y el proyecto son grandes, la criticidad es alta, la cultura organizacional está acostumbrada al orden a través de políticas y procedimientos, se presentan bajas tasas de cambio y los equipos están formados por profesionales con bajos o altos niveles de habilidad mientras que las practicas ágiles están al otro extremo de lo disciplinado y se usan más exitosamente en equipos y proyectos pequeños, baja criticidad, en culturas que maneja varios grados de libertad y se alimentan del caos y con equipos de profesionales con altos niveles de habilidad. La propuesta de [47] es balancear agilidad y disciplina, logrando que la disciplina cree memorias bien organizadas, manejo de la historia y experiencia de la organización que luego la agilidad usará para ajustarse y adaptarse a los cambios.

Sin embargo, se percibe al conocer en profundidad las prácticas ágiles que para una adopción exitosa de las mismas es preciso hablar también de disciplina en el proceso, pues se necesita disciplina con aspectos de desempeño del equipo como la planeación y seguimiento constante, retroalimentación continua, interacción permanente con el cliente, prácticas de prueba, integración y liberaciones continuas y manejo de la propiedad colectiva, así como con el seguimiento de los valores del manifiesto ágil [14,48].

El proceso de desarrollo depende de la comunicación, coordinación y cooperación. El proceso de desarrollo está basado en descubrimientos que dependen de la comunicación, coordinación y cooperación de los equipos dentro de marcos de trabajo predefinidos.

El desarrollo de software es un proceso basado en el conocimiento y su éxito depende de la colaboración con el usuario y de la coordinación de muchos roles. Se percibe que las prácticas tradicionales u orientadas a planes comparten el conocimiento principalmente a través de documentos y que sus equipos de trabajo por lo general están conformados por personas con roles muy especializados, cada persona o equipo es experta(o) en alguna disciplina y las tareas se planifican cuidadosamente para permitir el apoyo transversal a diferentes proyectos. Esta forma de trabajo cambia el foco desde los individuos y su creatividad hacia los procesos. Esto puede crear islas de conocimiento y dificulta compartirlo [35].

Por otro lado, las prácticas ágiles enfatizan el tener un equipo "*cross-functional*", donde las personas no se especializan en una disciplina, sino que desempeñan todos los roles definidos, permitiendo rotar los roles de una persona a otra y auto-asignarse tareas no de acuerdo a su experiencia sino a la necesidad del momento. Se enfatiza en trabajar con un objetivo común, en vez de solo preocuparse por las tareas de su rol. Sin embargo, es posible tener expertos especializados que se rotan entre los diferentes equipos de la organización.
Este tipo de equipos valoran más la interacción entre los individuos que los procesos y herramientas, reemplazando mucha de la documentación escrita por comunicaciones

informales [35]. Dentro de las prácticas usadas para transferir el conocimiento se encuentra el uso de estándares de codificación que permitan crear un código bastante entendible por todo el equipo de trabajo, siempre se busca tener la documentación suficiente y necesaria. Una ventaja de los equipos multifuncionales es que facilitan la transferencia de conocimiento dentro del equipo. Sin embargo, es posible que si los miembros de equipos saben solo un poco de todo, no se profundice en mejores técnicas dentro de las disciplinas para lograr productos de trabajo de mayor calidad o en menor tiempo.

2.2. El papel de las prácticas

En el contexto de la Ingeniería de Software, la práctica puede tener diferentes connotaciones. Según la definición de la IEEE [25], la práctica es un requisito empleado para prescribir un enfoque uniforme disciplinado al proceso de desarrollo de software, la práctica se entiende entonces como un elemento que indica cómo se realiza el trabajo.

Según CMMI la práctica representa un lineamiento general (qué) sin precisar la manera como éste se realiza (cómo) y por lo tanto se dice que son prácticas de tipo descriptivo. Existe una distinción entre las prácticas específicas y genéricas de un área de proceso; las prácticas específicas son componentes esperados del modelo que describen las actividades que son importantes para lograr una meta de un área de proceso específica, las prácticas genéricas son aquellas que se aplica a múltiples área de proceso [34].
Son las prácticas genéricas las que contribuyen a la institucionalización del proceso, es decir las que permiten engranar las prácticas en la manera como las organizaciones hacen su negocio. Por tanto se espera que los proyectos en una organización desarrollen actividades que además de lograr las prácticas específicas, logren cada una de las prácticas genéricas para asegurar la interiorización de los procesos.

Algunos autores como Wang y King establecen una clasificación en niveles de los procesos de ingeniería del software. La siguiente taxonomía es una de esta clasificaciones (en orden de menor a mayor jerarquía) [31].

- *Práctica*: Una actividad o un estado en un proceso software para llevar a cabo una tarea específica. Es la unidad mínima que puede ser modelada.
- *Proceso*: Es un conjunto de prácticas funcionalmente coherentes y reutilizables para un proyecto software.
- *Categoría*: Es un conjunto de procesos funcionalmente coherentes y reutilizables en algún aspecto de la ingeniería del software.
- *Subsistema*: Es un conjunto de categorías funcionalmente coherentes y reutilizables en alguna parte principal de la ingeniería del software.
- *Marcos metodológicos*: Conjunto completo de procesos software estructurados.

Entonces, al ser la práctica la unidad mínima que puede ser modelada en un proceso, es importante enfocarse en estudiar y mejorar las prácticas de ingeniería del software y las organizacionales para así lograr una mejora del proceso software.

Según Scott Ambler, uno de los precursores del enfoque ágil, la práctica es solo un elemento de una metodología. Toda práctica está fundamentada en valores y principios; por ejemplo AM (Agile Modeling) es una metodología basada en prácticas para modelar y documentar sistemas basados en software. Se puede apreciar en el sitio web de AM [29] que una metodología no es un proceso prescriptivo, en otras palabras, no define procedimientos detallados de cómo realizar las tareas, sino que proporciona consejos de cómo ser más efectivo. Las metodologías no son procesos de software completos, por tanto necesitan ser usadas como complemento de un proceso software base; esto permite personalizar el proceso para que refleje necesidades únicas.

2.3. Perspectivas y tendencias en la mejora del proceso software

El desarrollo de software es una actividad intensiva en conocimiento que demanda procesos efectivos para soportar la organización del trabajo; siendo el negocio del software altamente competitivo, y el contexto en el que se aplica las soluciones altamente dinámico, es necesario mejorar continuamente sus prácticas por lo que la mejora del proceso software (Software Process Improvement SPI) es un área de oportunidad para mejorar las capacidades de la organización.

Actualmente la mejora de procesos software (SPI) ofrece diferentes perspectivas sobre su objetivo de intervención, dependiendo de hacia dónde se debe enfocar la atención y que tipos de medios son útiles. Dentro de SPI el enfoque puede estar sobre los procesos de la organización, sobre las competencias de las personas responsables de desarrollar el producto o sobre el contexto que soporte las actividades de la ingeniería del software [28].

A continuación se presentan las perspectivas en la mejora del proceso software, agrupadas en dos categorías: Mejoras orientadas a los procesos y mejoras orientadas a las personas.

2.3.1. Mejoras orientadas a los procesos

El SEI, ha identificado varias dimensiones en las que una organización puede centrarse para mejorar su actividad: las personas, los métodos y procedimientos, y las herramientas y equipos.

Lo que mantiene todo unido son los procesos usados en las organizaciones, los cuales permiten alinear el modo de trabajar, ser escalables y proporcionan una forma de incorporar el conocimiento de cómo hacer mejor las cosas. Los procesos permiten explotar mejor los recursos y analizar las tendencias de la actividad organizacional. Aunque las personas y la tecnología son importantes, un enfoque de mejora orientada al proceso tiene como meta proporcionar la infraestructura y la estabilidad necesarias para hacer frente a un mundo cambiante y maximizar la productividad de las personas y el uso de la tecnología para ser competitivos [34].

El proceso software influencia el trabajo diario de las personas y su colaboración con otros, mejora el desempeño y puede llegar a convertirse en una herramienta de mercadeo de los productos y servicios.
Dentro de la categoría de mejoras orientadas al proceso, se encuentran las siguientes tendencias:

Analytical SPI

Para esta tendencia no hay un patrón común para el desempeño o la conformidad. La organización de software de manera individual es el punto de origen para la mejora. Asume que las metas individuales, características, atributos de producto y experiencias de la organización deben guiar el proceso de cambio. El cambio es definido por un dominio local en vez de un conjunto universal de prácticas.

La mejora de procesos se convierte en un esfuerzo colaborativo, los usuarios del proceso de manera colectiva diseñan el proceso software a través de la facilitación, la reflexión y la improvisación.

Algunos sinónimos de esta tendencia podrían ser:

- “Bottom-Up”
- “Receipt-based”
- Problem-driven

Un ejemplo de SPI con el enfoque Analytical es el paradigma de mejora de la calidad (QIP), el cual usa el concepto de “lean empresarial” para concentrar la producción y recursos en las actividades que añaden valor al negocio y que representan los procesos de negocio críticos de la organización. Su objetivo es crear una organización al usuario y al aprendizaje.

Dentro de sus características claves se pueden mencionar las siguientes:

- Obliga a un punto local de inicio para la mejora, basado en las particularidades de la organización.
- Debe ser incremental y basado en ciclos de aprendizaje produciendo “paquetes de experiencia”.
- Guiado por el método Meta-Pregunta-Métrica (Goal-Question-Metric GQM).
- Fábrica de experiencia: infraestructura organizacional que permite el aprendizaje.

Algunos hallazgos y críticas relacionados con QIP incluyen: [38]

- El impacto sobe la productividad por el aprendizaje de experiencias es significativo. Pero se diferencia en niveles de análisis (individuos, grupos, unidad organizacional).
- La mejora de proceso software es una tarea que depende del compromiso organizacional.

Agile SPI

Es una tendencia emergente, pragmática y orientada a la práctica. Se basa en los principios y métodos de desarrollo ágiles. Es un enfoque flexible que se adapta con el entendimiento colectivo de las necesidades contextuales para proporcionar respuesta rápida a los cambios, incremento de la satisfacción del cliente y bajas tasas de defectos. [38]

Dentro de sus características claves se pueden mencionar las siguientes:

- Guiado por la retroalimentación iterativa.
- Es un híbrido entre proceso software y SPI analítico.

Un ejemplo de SPI con el enfoque ágil es “Tácticas de guerrilla” de Ericsson la cual tiene dos características importantes:

- Estrategia de grupos de difusión para la mejora de proceso usando proyectos pilotos para crear masa crítica y amplia adopción en la organización.

- Los proyectos pilotos actúan como agentes de cambios y la masa crítica permite cruzar el abismo hacia una mayoría que adopta las nuevas prácticas.

Algunos hallazgos y críticas relacionados con Agile SPI incluyen: [38]

- Muy temprano para concluir pues aún no ha sido evaluado por otros.
- SPI orientado a la práctica puede mostrar un ROI más rápido que por ejemplo CMMI.

Benchmarking SPI

Esta tendencia es conforme con la idea de adoptar una norma existente para mejorar los procesos software, es un enfoque basado en la norma para la mejora de los procesos software y proporciona la base para realizar evaluaciones de capacidad de los procesos existentes de acuerdo a un estándar profesional que indica una configuración ideal de procesos y para la cual se formulan estrategias de ayuda para disminuir la brecha entre la norma y la practica [28].

Se basa en modelos estables, idealizados y definidos a priori de normativas de ingeniería de software. Dentro de sus características claves se pueden mencionar las siguientes:

- Conjunto predefinido de mejores prácticas y medición común de éxito.
- Enfocado en la mejora del proceso.
- Meta generalizada
- Independiente del dominio

Algunos sinónimos de esta tendencia podrían ser:

- “Top-down”
- Blueprints
- “Best practice”
- Norm-driven

Un ejemplo de SPI con el enfoque de Benchmarking es CMMI y su modelo de mejora IDEAL^{SM}.

El modelo de madurez y capacidad (CMM) ha existido desde 1991. Este modelo describe un método basado en la evolución para mejorar una organización desde un nivel ad hoc e inmaduro a uno que es disciplinado y maduro. CMM es mundialmente reconocido y fue desarrollado por el Instituto de Ingeniería del Software (Software Engineering Institute - SEI) de la Universidad Carnegie Mellon de Pittsburgh, USA.
En el 2002, se anunció una versión extendida llamada CMMI, la "I" viene de la palabra Integración. CMMI-DEV define 22 áreas de proceso a ser implementadas. Cada área de proceso tiene definidas unas metas requeridas, unas prácticas esperadas y unas sub-practicas recomendadas. Además se tienen un conjunto de prácticas genéricas que deben ser aplicadas para todos los procesos [29,34].
La experiencia con CMMI, demuestra que organizaciones evaluadas con altos niveles de madurez CMMI mejoran la habilidad de entregar a tiempo, dentro del presupuesto y con la calidad acordada [30].
Un gran número de organizaciones gubernamentales en el mundo, han establecido requisitos de madurez CMMI para la contratación de sus proveedores de software [16].

Para soportar la constitución e implementación de esquemas y proyectos de mejora de procesos software, el Instituto de Ingeniería de Software (SEI) propone un marco de trabajo llamado IDEAL^{SM} por sus siglas en inglés (Initiating, Diagnosing, Establishing, Acting, and Learning), que consta de cinco fases y que proporciona un estructurado enfoque de mejora continua. Las fases del modelo son: Iniciación, Diagnostico, Establecimiento, Actuación, Aprendizaje [37].

Algunos hallazgos y críticas relacionados con programas de mejora de procesos soportados en CMMI incluyen: [38]

- Mayoría de evidencia empírica de CMMI.

- Muchas historias de éxito de casos no representativos, por ejemplo: organizaciones muy grandes.
- Resultados aunque numerosos no concluyentes.
- Las fallas son raramente reportadas.
- Niveles altos de madurez altos también reflejan un aumento de la burocratización y amplía la barrera entre los modelos de procesos y la práctica de software.
- El "patrón" común impulsa el cambio - no la experiencia o los objetivos de la organización.
- Demasiado costoso, no es apropiado para las organizaciones más pequeñas.

Las organizaciones más pequeñas se adaptan a través de la exploración en lugar de la optimización.

2.3.2. Mejoras orientadas a las personas y los equipos de trabajo

Aun los mejores métodos y herramientas requieren personas competentes para hacer uso de ellos, las personas competentes son un ingrediente de cualquier proceso software que esté funcionando bien. Varios autores reconocen este factor, por ejemplo, Boehm identifica a las personas como el mayor riesgo para el desarrollo de software y Humphrey argumenta que las personas talentosas son el elemento más importante en cualquier organización de software. Como las condiciones varían de proyecto a proyecto se requiere de personas comprometidas para seguir un proceso software establecido, pero también es necesario que sean competentes en sus labores.

La manera como la organización desarrolla las competencias de sus empleados es una parte esencial del éxito de proyectos de mejora de procesos. Algunos modelos de desarrollo de competencias han sido desarrollados a diferentes niveles (individual, proyecto y organizacional) para apoyar los esfuerzos de las compañías en lograr habilidades y responsabilidades apropiadas entre los ingenieros de software, que complementen infraestructura de apoyo con métodos y herramientas [28].

Un componente crítico, para ser una compañía ágil, es la fuerza de trabajo con el conocimiento, las habilidades para hacer ajustes rápidos y la voluntad de adquirir nuevas competencias, es por esto que la retención de empleados experimentados llega a ser un factor esencial a la hora de mejorar la productividad y el tiempo de entrega [39].

Algunas de las técnicas que las organizaciones han tratado de aplicar en sus esfuerzos para ir más allá de una gestión estratégica del capital humano incluyen:

- Combinar reducción con reestructuración
- Mejorar el intercambio de información
- Comunicar con claridad la misión de la organización
- Desarrollar programas de participación para los empleados
- Establecer procedimientos formales de resolución de quejas
- Desarrollar planes de incentivos
- Enfatizar la importancia del entrenamiento y la capacitación
- Formalizar la gestión del desempeño y procesos de retroalimentación
- Desarrollar análisis y diseños de trabajos
- Rotación de puestos de apoyo
- Programas formales de tutoría

Sin embargo, muchas organizaciones no tienen un marco de trabajo (framework) para implementar estas prácticas avanzadas. Un ejemplo de framework para desarrollar las competencias de los empleados, a través de un programa de desarrollo continuo de la fuerza de trabajo, lo encontramos en el People Capability Maturity Model® (People CMM®) del SEI [39].

Este modelo de mejores prácticas para gestionar y desarrollar la fuerza de trabajo de una organización ayuda a las empresas a caracterizar la madurez de sus prácticas en la gestión del talento humano, colocar prioridades para las acciones de mejoras y establecer una cultura de excelencia. Aunque su liberación fue en 1995, hoy en día aparecen solo 21 organizaciones en el mundo que se han evaluado con este modelo, entre las más conocidas

están Accenture y Tata Consultancy Services [9].Se percibe además que este modelo no es aplicado ni conocido en las empresas de desarrollo de software colombianas.

Este modelo consiste de cinco niveles de madurez que establecen bases sucesivas para mejorar de manera continua las competencias individuales, desarrollar equipos efectivos, motivar la mejora del desempeño y moldear el talento humano a las necesidades de la organización [39].

Otra tendencia en las mejoras orientadas a las personas es la incorporación de los valores y principios ágiles como motor del cambio en los equipos de trabajo; con esto se busca mejorar los resultados del proceso de desarrollo software potenciando en las personas valores como:

- Comunicación
- Retroalimentación
- Simplicidad
- Coraje
- Respeto

Este tipo de ambientes altamente colaborativos permite mejorar el proceso de desarrollo debido a que este es un proceso muy social y al desarrollar estos valores en los equipos a nivel individual y organizacional y también entre el cliente y los equipos se espera aumentar la confianza. Cuando se adoptan practicas ágiles como la propiedad colectiva del código, reuniones diarias, cliente en sitio o programación en pares se está promoviendo también esa confianza mutua [35]. La confianza en otras personas y en su trabajo facilita la generación de conocimiento y poder compartirlo. La creación de una cultura de conocimiento permite a su vez, hacer sostenible el proceso de desarrollo disminuyendo el caos y haciéndolo cada vez más eficiente. Esto se explica fácilmente porque el flujo de información se hace más dinámico y permite disminuir la entropía en los equipos de trabajo.

2.4. Enfoques de desarrollo ágil

Los enfoques ágiles fueron concebidos como una nueva perspectiva para desarrollar software y mejorar la baja tasa de proyectos exitosos alcanzados con prácticas orientadas a planes [14].

En el 2001, se reunieron importantes representantes de estos enfoques ágiles (XP, *Scrum*, DSDM, ASD, Crystal, FDD, Pragmatic Programming, etc) para tratar de encontrar bases comunes y buscar una alternativa a los procesos de desarrollos de software robustos y guiados por la documentación. De esta reunión emergió el Manifiesto por el Desarrollo Ágil de Software en el cual se plasma el consenso alrededor de cuatro valores principales.

Los 4 valores que define el manifiesto ágil son los siguientes: [18].

"Individuos e interacciones sobre procesos y herramientas
Software funcionando sobre documentación extensiva
Colaboración con el cliente sobre negociación contractual
Respuesta ante el cambio sobre el seguir un plan

Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda."

Este grupo de representantes que se reunieron en el 2001 se autonombraron "The Agile Alliance". Una propuesta del Agile Alliance [45] consiste en categorizar las practicas ágiles existentes en "tribus" o áreas de interés que se pueden entrecruzar dando flexibilidad al proceso definido como se aprecia en la figura 2.1.

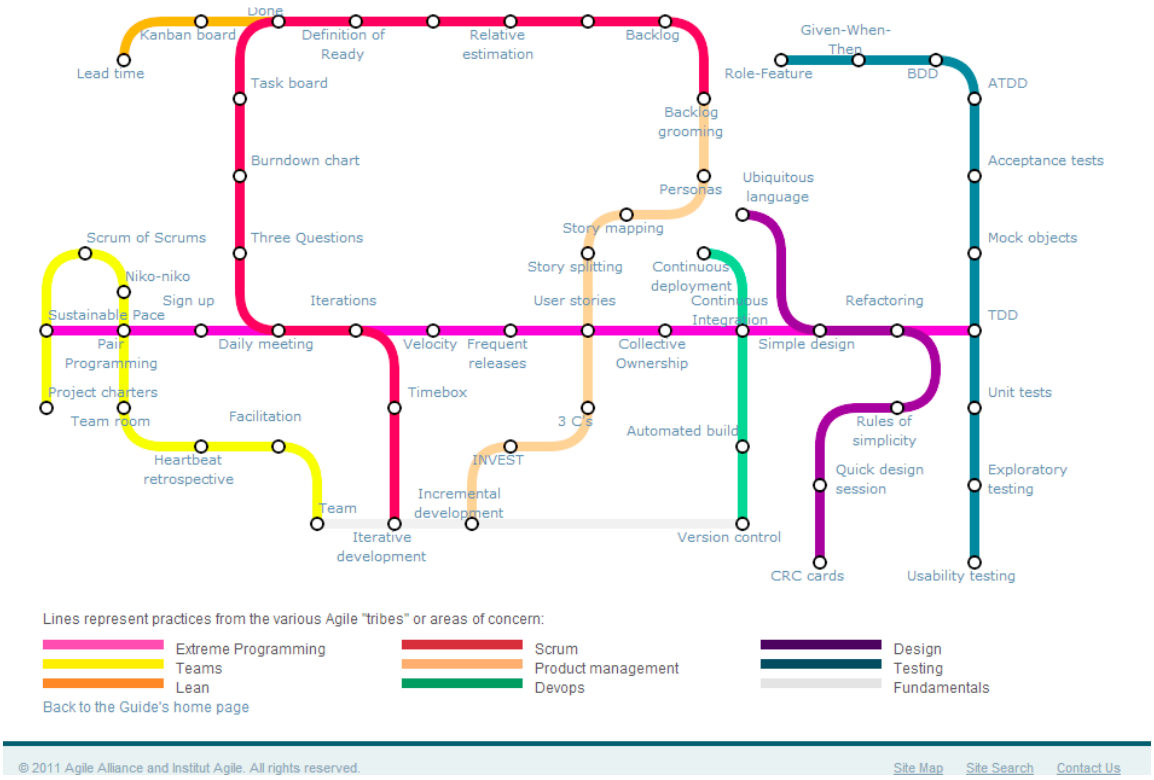

Figura 2.1: Tribus de prácticas ágiles. Tomado de [45].

Existen diferentes propuestas del enfoque ágil que es preciso comparar. Hay enfoques ágiles como el desarrollo ágil distribuido (DAD) o el proceso unificado ágil (AUP) que promueven el uso de modelos. En DAD se busca que los retos causados por la distribución de los equipos, como por ejemplo, los problemas de comunicación, falta de proximidad física, falta de contexto y conocimiento compartido y poca disponibilidad de los miembros del equipo, sean solucionados usando prácticas de soporte distribuidas como las reuniones diarias distribuidas y se soporten en el uso de modelos.

Por su parte, otros enfoques se centran en la gestión de proyectos y practicas colaborativas promoviendo el trabajo de los equipos, entre ellos *SCRUM*, ASD, Lean Development, DSDM, etc. Y otros enfoques están orientados hacia las prácticas de implementación del software o el código como XP, AM (Agile Modeling) y FDD.

2.4.1. Scrum

Según Schwaber [2], *Scrum* es un proceso para manejar proyectos complejos. Los problemas complejos son aquellos que tienen un comportamiento impredecible.
Scrum no es un proceso prescriptivo, no describe lo que se debe hacer en cada situación. *Scrum* ofrece un marco de trabajo y un conjunto de prácticas que orientan el trabajo del equipo, con el propósito de promover su autogestión.
Scrum es un marco de trabajo (framework) de proceso compuesto de un conjunto de prácticas que puede ser usado para administrar y controlar proyectos complejos. *Scrum* maneja la complejidad de los proyectos de desarrollo de software implementando los pilares de inspección, adaptación y visibilidad de un proceso de control empírico, como se explicará a continuación [2].

Existen procesos tan complicados que no pueden producir de manera repetible salidas de calidad, es decir, no pueden ser descritos como procesos definidos y mucho menos ser controlados como procesos definidos; para este tipo de procesos se emplean los procesos empíricos. Estos procesos empíricos se apoyan en tres elementos: inspección, adaptación y visibilidad. Cuando se habla de inspección se refiere a que el proceso o el producto son inspeccionados tan frecuentemente que cualquier variación no aceptable es detectada. Al determinar a través de la inspección que algo está fuera de límites aceptables, se debe ajustar el proceso o el producto inspeccionado; a esto se refiere la adaptación. Por último, la visibilidad se refiere a mostrar con transparencia y coraje a todos los interesados los problemas que afectan la salida; este último elemento da fuerza a los dos primeros.

Valores y principios básicos de *Scrum*

En [7] se resumen algunos principios básicos de *Scrum*:

- Pequeños equipos de trabajo auto-organizados, buscando maximizar la comunicación y el intercambio de conocimiento tácito e informal, y minimizar el sobreesfuerzo.

- Adaptación a las solicitudes de cambios técnicos o cliente / servidor, lo que garantiza la entrega del mejor software posible.
- Entregas frecuentes de versiones que pueden ser probadas, ajustadas, implementadas, documentadas y publicadas para la producción.
- La división del trabajo y las responsabilidades de equipo del proyecto para pequeñas entregas.
- Capacidad para ofrecer software listo cuando lo necesite el cliente o negocio.

Estos principios están alineados con los 4 valores que define el manifiesto ágil [18].

Información general

La figura 2.2 muestra el proceso básico de *Scrum*, el cual se explica más adelante. A continuación se presenta una terminología básica:

El *Product Backlog* es una lista priorizada de todos los requisitos del producto, incluyendo nuevas características, funciones, tecnologías, mejoras y corrección de defectos. El contenido puede venir de cualquier lado; usuarios, clientes, ventas, mercadeo, servicio al cliente e ingeniería pueden enviar elementos al *backlog*. Sin embargo solo el *Product Owner* tiene permisos de priorizar la lista y decide cual elemento será implementado. El primer *Product Backlog* puede ser una lista de requisitos desde un documento de visión y luego emerge de acuerdo a cómo evoluciona el entendimiento del cliente.

El *equipo de desarrollo* toma los elementos de más prioridad del *Product Backlog* (solicitudes de cambio) de acuerdo a lo que ellos piensen puedan manejar durante la siguiente iteración llamada *Sprint*. Durante cada *Sprint*, el equipo mantiene una lista de las tareas a ser desarrolladas, esto se llama el *Sprint Backlog*. Cada *Sprint* debe resultar en una versión de producto completamente funcional y potencialmente distribuible.

Roles de *Scrum*

En *Scrum*, sólo hay tres roles principales: el *Team*, el *Product Owner* y el *Scrum Master*.

Team

El Team corresponde a un equipo de desarrollo pequeño (3-9 personas), auto-organizado y multifuncional, con diferentes habilidades y conocimientos técnicos, el cual es apoyado por los otros dos roles: el Product Owner y el *Scrum* Master.

El equipo tiene autoridad completa para hacer lo necesario para alcanzar la meta del sprint. Está restringido solo por los estándares y convenciones de la organización. No hay roles o títulos dentro del equipo y los miembros no tienen una descripción del trabajo diferente a hacer el mejor trabajo posible.

Product Owner

El *Product Owner* representa los intereses de todos los que directa o indirectamente serán beneficiados con el éxito del proyecto [7].

Scrum Master

El *Scrum Master* es quien vela por la productividad del equipo y es responsable por mantener el proceso *Scrum*, por enseñarlo a cualquiera involucrado en el proyecto y asegurar que todos siguen las reglas y prácticas de *Scrum* [2].

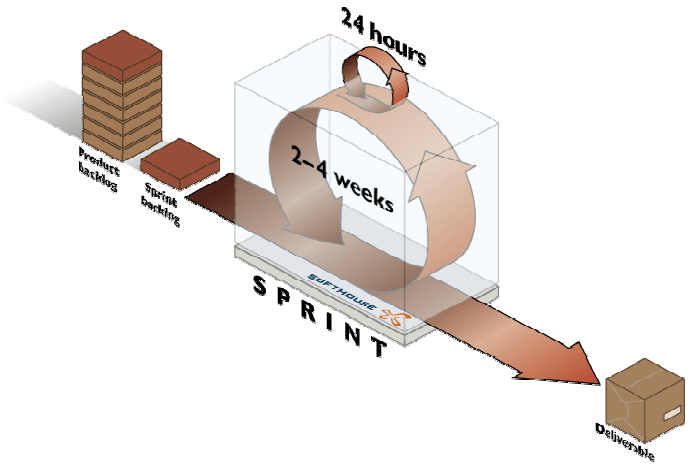

Figura 2.2: El proceso Scrum (imagen tomada de Softhouse).

El proceso

El progreso de los proyectos que utilizan *Scrum* se realiza y verifica es una serie de iteraciones llamadas *sprints*. Estos *sprints* tienen una duración fija preestablecida de no más de un mes. Al comienzo de cada *sprint* el equipo realiza, junto con el *Product Owner*, un compromiso de entrega de una serie de funcionalidades o características del producto.
Los equipos trabajan orientados a las funcionalidades y actividades definidas al inicio de cada *sprint*. Existen cuatro reuniones en *Scrum*, la reunión de planeación, la reunión diaria, la reunión de revisión y la reunión de retrospectiva.

Reunión de planeación del sprint

En las reuniones de planeación los clientes, usuarios, gestión, el *Product Owner* y el *Team* determinan las metas y funcionalidades que serán implementadas en el próximo sprint. Esta reunión consiste de dos reuniones separadas. Primero el equipo reunido con el *Product Owner*, gestión y usuarios define que funcionalidad construir durante el próximo sprint. En la segunda reunión, el equipo trabaja solo para definir como construirán esta funcionalidad durante el sprint, es decir se identifican las tareas o plan de trabajo a ser seguido.

Reuniones diarias

Las reuniones diarias, tienen una duración de 15 minutos y sirven para la alineación y visibilidad del equipo. El *Scrum Master* es responsable por conducir de manera exitosa la reunión diaria, manteniéndola corta y asegurándose que los miembros del equipo hablan brevemente. Otros interesados pueden asistir a la reunión, pero solo como invitados y no está permitido que interfieran de alguna manera.
Cada persona del equipo, uno a la vez, brevemente responde tres preguntas [49]:

- ¿Qué hice ayer que ayudó al Equipo de Desarrollo a lograr el Objetivo del Sprint?
- ¿Qué haré hoy para ayudar al Equipo de Desarrollo a lograr el Objetivo del Sprint?
- ¿Veo algún impedimento que evite que el Equipo de Desarrollo o yo logremos el Objetivo del Sprint?

Las primeras dos preguntas da a los asistentes un breve reporte del progreso y permite al equipo sincronizar su trabajo. Si un miembro del equipo identifica algo que está obstaculizando su trabajo, el *Scrum Master* tiene como prioridad remover este impedimento.

Reunión de revisión del sprint

Una reunión de revisión del sprint tiene como objetivo presentar a los clientes los desarrollos e incrementos de cada sprint al momento de la entrega. Esta reunión permite que todos tengan un entendimiento del incremento del producto, ya que es el conocimiento que ellos necesitarán para la próxima reunión de planeación.

Reunión de retrospectiva

En la reunión de retrospectiva, son evaluados los errores y aciertos en el desarrollo del sprint y lo que puede ser mejorado para aumentar la productividad del siguiente *Sprint* [7].

2.4.2. XP

Extreme Programming (XP) fue concebido y desarrollado para apoyar necesidades específicas de desarrollo de software realizados por equipos pequeños, en el contexto de requisitos vagos e indefinidos. Este nuevo método encontró retos, incluido el supuesto de que el costo de cambiar una pieza de software se incrementa dramáticamente con el tiempo durante el ciclo de vida. XP reconoce que se debe trabajar para alcanzar esta reducción en costo y explotar esos ahorros una vez hayan sido ganados [41].

Las bases de XP incluyen:

- Distinguir entre las decisiones a ser tomadas por los intereses del negocio y las que realizan los stakeholders del proyecto.
- Escribir pruebas unitarias antes de programar y mantener todas las pruebas corriendo correctamente en todo momento.
- Integrar y probar todo el sistema – varias veces al día.

- Producir todo el software en parejas, dos programadores en una pantalla.
- Iniciar proyectos con un diseño simple que constantemente evoluciona para añadir flexibilidad y eliminar la complejidad innecesaria.
- Liberar lo mínimo de un sistema en producción rápidamente y crecer el software en la dirección que pruebe ser de más valor para el negocio.

¿Por qué XP es tan controversial? A continuación se exponen algunas posibles razones [41]:

- No obliga a los miembros del equipo a especializarse y convertirse en analistas, arquitectos, programadores, evaluadores o integradores - cada programador en XP participa en todas estas actividades esenciales cada día.
- No conduce a un análisis y diseño inicial completo (Big up-front design).
- Desarrolla la infraestructura y los marcos de trabajo como usted desarrolla su aplicación, no por adelantado – entregar valor del negocio es el corazón que impulsa los proyectos XP.
- No se escribe o mantiene la documentación de la aplicación- La comunicación en proyectos XP ocurre cara a cara, o por medio de pruebas eficientes y código cuidadosamente escrito.

2.4.3. Otras prácticas de apoyo

Kanban: Kanban es un enfoque lean para el desarrollo ágil de software. Literalmente, Kanban es una palabra japonesa que significa “Tarjeta visual” [26]. Su significado dentro del contexto ágil fue tomado del sistema de producción de Toyota, donde designaba un sistema para controlar los niveles de inventario de varias partes, En el 2004, David Anderson presentó una implementación más directa del pensamiento Lean y de la teoría de restricciones para el desarrollo de software. Bajo la guía de expertos como Don Reinertsen, hubo una evolución que Anderson llamó un “sistema Kanban para el desarrollo de

software", así mientras el uso de Kanban en el desarrollo de software es relativamente nuevo, en la producción Lean tiene alrededor de medio siglo de antigüedad [42].

El núcleo de Kanban se basa en las siguientes características:
Visualizar el flujo de trabajo

- Dividir el trabajo en piezas, escribir cada ítem en una tarjeta y colocarlo en la pared
- Usar columnas etiquetadas para ilustrar donde está cada ítem en el flujo

Limitar el trabajo en progreso (Work In Progress WIP)

- Asignar límites explícitos a cuantos ítems pueden estar en progreso en cada estado del flujo de trabajo.

Medir el tiempo de ejecución (tiempo promedio para completar un ítem, algunas veces llamado "tiempo del ciclo")

- Optimizar el proceso para hacer el tiempo de ejecución tan pequeño y predecible como sea posible.

TDD: Test Driven Development (TDD), también como prueba antes del código o diseño guiado por las pruebas, es una práctica donde el programador debe escribir el código de producción solo después de escribir un caso de prueba automatizado fallido.
Kent Beck acuñó el término Test Driven Development (TDD), en su *libro Extreme Programming Explained: Embrace Change* en 1999. Beck inició el uso de la práctica a principio de los 90´s mientras desarrollada en SmallTalk. Sin embargo, como el mismo Beck anota, la idea de escribir código de pruebas antes del código de producción no es su idea original. Una de las referencias de equipos usando un enfoque al estilo TDD la podemos encontrar en el proyecto Mercurio de la NASA a principio de los 60´s.

William Wake adaptó una metáfora de semáforo para dar a los desarrolladores una idea de cómo la practica debe trabajar. Se inicia escribiendo la prueba y esto corresponde al color verde, cuando la prueba falla al no compilar correctamente pues no hay código de producción disponible, el semáforo pasa a la luz amarilla. Una vez que un "stub" se escribe

en el código de producción, el código podría fallar porque el “stub” no hace nada, resultando en una luz roja.
Una vez que el código de producción ha sido escrito y el caso de prueba falla, el desarrollador podría retornar a un estado de luz verde [43].

Este semáforo fue recortado a Rojo, Verde, Refactor. La luz roja representa la falla, o posible no compilación del código de prueba. La luz verde es el resultado final de escribir la mínima cantidad de código para que la prueba pase. El refactoring es usado para eliminar cualquier código duplicado o malas técnicas de programación que fueron introducidas al tratar de pasar al verde.

Después de su libro introductorio de TDD en el 2001, Beck escribió un segundo libro para proporcionar mayor detalle de la práctica, resumiéndola en 5 pasos:

- Escribir un nuevo caso de prueba
- Correr todos los casos de prueba y ver la falla del nuevo
- Escribir código suficiente para hacer que la prueba falle
- Volver a correr los casos de prueba y ver que todos pasan
- Refactorizar el código para remover la duplicación

TDD es una práctica fundamental en XP para desarrollar software, pero puede ser usada de forma separada. Se presenta a continuación el resultado de varias encuestas realizadas sobre prácticas ágiles, en el 2010 se realizó una encuesta de autodenominados practicantes ágiles y 53% de los 293 encuestados usó TDD para validar software. Kent Beck encuestó a los asistentes al webinar “Líderes ágiles 2010” preguntando por el uso de TDD y aproximadamente 50% de 200 encuestados indicaron que ellos usaron TDD, sin embargo en otra encuesta más amplia, Forrester reportó que solo 3.4% de 1298 profesionales de TI usaron un enfoque TDD para desarrollar. A pesar de la diferencia, los números indican suficiente interés en TDD como para tratarla como una técnica seria [43].

2.5. La importancia de los estudios experimentales en Ingeniería de Software

Los sistemas software forman la base de la moderna sociedad de la información y muchos de esos sistemas están entre las cosas más complejas que se han creado. La Ingeniería de Software, por su parte, se relaciona con el desarrollo, mantenimiento y gestión de sistemas software de alta calidad de una manera predecible y efectiva en costo.

Las investigaciones en Ingeniería de Software (IS) estudian fenómenos del mundo real relacionadas con la IS y problemas relacionados con el desarrollo de nuevas tecnologías (modelos de procesos, métodos, técnicas, herramientas o lenguajes) o modificaciones de las existentes para soportar las actividades de IS, así como con la evaluación de los efectos de usar estas tecnologías en entornos complejos. Las ciencias que estudian los fenómenos del mundo real como las ciencias empíricas necesitan de métodos empíricos, los cuales obtienen información de observaciones y experimentos sistemáticos [33].

Victor R, Basili en [4] muestra la evolución de la ingeniería del software experimental durante 40 años, desde el pasado hasta el futuro, desde una perspectiva personal. Y concuerda parcialmente con Dag I.K. Sjøberg et al. En [33] en que la investigación en Ingeniería de Software tuvo una fase en donde se enfocaba en especificar los efectos de las tecnologías e identificar experimentalmente los límites de las técnicas, pero que hoy en día busca también desarrollar teorías, hipótesis y guías que permitan caracterizar, evaluar, predecir y mejorar como un ambiente (hardware, software, humano) afecta el desarrollo de software. Algunos de los métodos usados en los estudios experimentales incluyen experimentos controlados, estudios observacionales, estudios de casos, encuestas y entrevistas [4].

De acuerdo a Travassos et al. [36], se pueden mencionar los siguientes paradigmas de investigación:

- **Paradigma analítico**: que busca proponer una teoría formal o conjunto de axiomas, desarrolla una teoría, deriva resultados y si es posible, verifica los resultados con observaciones empíricas.
- **Paradigma experimental:** que observa el mundo, propone un modelo o teoría de comportamiento, mide, analiza y luego valida las hipótesis del modelo o teoría y repite el procedimiento evolucionando la base de conocimiento.

El paradigma experimental involucra varias etapas: el diseño experimental, la observación, un análisis cualitativo o cuantitativo, la recolección de datos y la validación sobre el proceso o producto que está siendo estudiando.

El análisis de tipo cualitativo, que es donde se enmarca este trabajo, utiliza métodos que involucran observaciones no controladas, tiene un carácter subjetivo y está orientado al descubrimiento más que a verificar. Por ejemplo, un estudio de tipo observacional es guiado por el entendimiento, domina el análisis cualitativo y dentro de las técnicas de investigación más usadas están la encuesta y la entrevista. Estas técnicas son realizadas después de los hechos, permiten obtener evidencia pero no permite tener control sobre las variables.

La investigación cualitativa es usada para estudiar actividades y fenómenos sociales como los sentimientos, procesos de pensamientos y emociones, los cuales son difíciles de estudiar a través de métodos cuantitativos [1].

De acuerdo a lo anterior y teniendo en cuenta que el desarrollo del software y la mejora de sus procesos es por naturaleza una actividad social, resulta natural que para el estudio sobre la adopción de prácticas ágiles como *Scrum* se utilice una metodología de investigación cualitativa.

Una de las metodologías que pertenece a las investigaciones cualitativas, y que resulta muy adecuada para estudiar el comportamiento humano y las culturas organizacionales es la de Grounded Theory (GT) [32].

2.6. Principios de Grounded Theory (teoría fundamentada)

A pesar de que Grounded Theory se originó a finales de los años 60s y que ha sido aplicada de manera extensa en las áreas de ciencias sociales [13], el uso de GT en estudios de desarrollo de software no es muy común, sin embargo se resaltan los trabajos de Orlikowski, Coleman and O'Connor, Nasirin et al y Crabtree et al. [32] que estudiaron la experiencia de organizaciones con la adopción y uso de herramientas, uso de prácticas de proceso en contextos particulares, factores críticos de éxito en la implementación del software y aspectos de comportamiento relacionados con la descripción de procesos.

El estudio de Montoni et al. En [32] concluye que la aplicabilidad de Grounded Theory para soportar las investigaciones sobre implementación de mejoras de procesos software está demostrada y desarrolla un marco de trabajo que explica los factores contextuales que tienen influencia positiva o negativa en el éxito de las iniciativas de mejora y en tomar acciones estratégicas para la mejora de procesos.

El método de Grounded Theory descrito en 1967 por Glasser y Strauss fue la primera síntesis de los dos puntos de vista: Glaser, que venía de Columbia con una herencia de pensamiento "científico" en sociología, complementó el estilo-Chicago de Strauss con un énfasis en el entendimiento y explicación de los fenómenos sociales. La salida fue una técnica inductiva fundamentada en los datos. Una década después Strauss, con uno de sus estudiantes publicó un procedimiento detallado de cómo guiar el método GT. Glasser, sin embargo se opuso a este procedimiento propagado por Strauss y Corbin pues afirmaba que inhibía que la teoría emergiera [40].

El objetivo de GT es desarrollar una teoría desde los datos en vez de obtener datos para probar una teoría o hipótesis. La teoría es fundamentada en la realidad como lo representan los datos [1].

En contraste con los métodos cuantitativos de investigación, en la cual la teoría es generada por deducción lógica desde supuestos a priori [3], en los métodos cualitativos el problema

de investigación no se declara en términos de variables dependientes o independientes sino que el enunciado del problema o pregunta de investigación es declarada de una manera que proporciona flexibilidad y libertad para explorar el fenómeno en profundidad [1].
La objetividad en la investigación cualitativa no significa controlar las variables sino tener una actitud abierta y tener la voluntad de escuchar y dar voz a los encuestados. Es entendido y aceptado que el entendimiento del investigador está basado en sus valores, cultura y las experiencias que el lleva a la situación investigada y que podría ser diferente de los participantes en la investigación [1].

Se escogió GT como método de investigación por diferentes razones: Primero, las prácticas y métodos ágiles se centran en las personas y sus interacciones, y GT es una metodología de investigación cualitativa rigurosa usada para investigar fenómenos como sentimientos, comportamientos y emociones, los cuales son difíciles de estudiar a través de métodos cuantitativos [1]. Segundo, GT es indicado para áreas de investigación las cuales no han sido exploradas en gran detalle antes, y la literatura de investigación sobre la adopción de prácticas agiles en empresas colombianas es escasa.

2.7.1 Procedimientos para crear una teoría fundamentada

La codificación es el método central en la transformación de los datos a la teoría. La codificación es definida como el proceso analítico a través del cual los datos son fracturados, conceptualizados e integrados para formar una teoría.
El objetivo de GT es identificar, desarrollar y relacionar los conceptos que son los componentes básicos de la teoría. Los autores, Corbin y Strauss identifican tres diferentes tipos de codificación para transformar los datos en una teoría fundamentada en la realidad: codificación abierta, codificación axial y codificación selectiva, que serán discutidas en los párrafos siguientes. Los diferentes tipos de codificación se realizan al mismo tiempo y la división entre ellos es una forma artificial de explicar el proceso [1].

Codificación abierta

La codificación abierta es el proceso analítico a través del cual se identifican los conceptos y sus propiedades y dimensiones son descubiertas en los datos. Para ser capaz de identificar los conceptos (fenómenos etiquetados) se debe abrir el texto y exponer los pensamientos, ideas y significados que figuran en él. Un concepto debe ser visto como una representación abstracta de un evento, objeto, o la acción/interacción que un investigador identifica como significativos en los datos. Los conceptos se comparan entre sí. Figura 2.3.

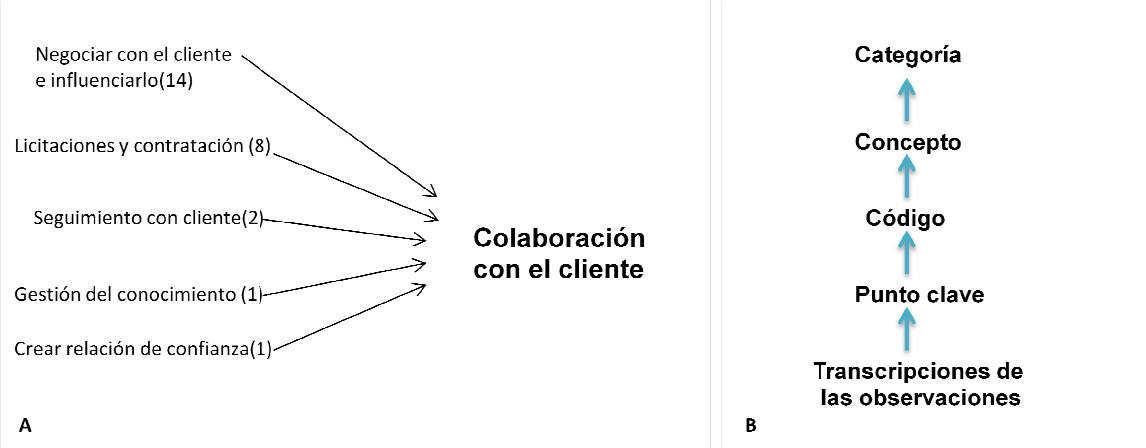

Figura 2.3: A: Como emerge la categoría *Colaboración con cliente* desde los conceptos subyacentes. B: Niveles de abstracción en Grounded Theory.

Las categorías surgen de conceptos similares que tienen propiedades similares. Las propiedades son características que son comunes a todos los conceptos de la categoría. Las propiedades de la "flor" pueden ser tamaño, duración, color, forma, etc.

La categorización de los conceptos es un proceso de abstracción. El investigador puede dar los nombres de las categorías pero también puede venir de las palabras de los entrevistados. Sin embargo, debería ser un descriptor lógico de lo que está pasando.

La siguiente cita de Strauss y Corbin sobre la identificación de las categorías sintetiza el espíritu de este trabajo de maestría: "Queremos ver nuevas posibilidades en los fenómenos y clasificarlos en formas que otros no podrían haber pensado antes (o, si se consideró

anteriormente, no fueron desarrolladas sistemáticamente en términos de sus propiedades y dimensiones).”

Las categorías deben estar fundamentadas. Esto implica que se forman a partir de la evidencia en la investigación. La literatura puede ser usada para añadir nuevas categorías, pero esas categorías serán etiquetadas como preliminares hasta que puedan ser verificadas por los datos, eso significa estar basado en la realidad. La interpretación de los hechos por el investigador influye en la denominación de las categorías.

Las dimensiones representan la ubicación de una propiedad a lo largo de un rango. Las dimensiones de la propiedad tamaño de la categoría flor puede ser de pequeño a grande, y de la propiedad color puede ser de diferentes colores o intensidades. Las categorías nos dan un método para comparar los diferentes incidentes. Los incidentes se comparan en términos de propiedades y dimensiones. Las categorías pueden ser definidas en subcategorías que responden a preguntas como qué, cuándo, dónde, quién, cómo y con qué consecuencias sobre las categorías. Las relaciones entre las categorías se estudian para formar las teorías.

Codificación Axial

La codificación axial se define como "el proceso de relacionar las categorías a sus subcategorías, se denomina "axial" porque la codificación se produce alrededor del eje de una categoría, que une categorías a nivel de propiedades y dimensiones. El propósito de la codificación axial es volver a montar los datos que se fracturaron durante la codificación abierta. Al refinar la información sobre cada categoría y sus subcategorías se logra lo anterior. Las condiciones, acciones/interacciones y consecuencias asociadas con el fenómeno (o categoría) se identifican para describir el contexto (estructura) y el proceso de un fenómeno. Dado que una categoría es una forma codificada de un fenómeno, puede ser vista como una representación de un patrón de acontecimientos, eventos o acciones / interacciones, que pueden ser descritas por las condiciones, acciones/interacciones y consecuencias.

Las condiciones explican la situación o contexto en el que se produce un fenómeno. Las condiciones pueden ser causales, de intervención y contextuales. Las condiciones contextuales son los conjuntos específicos de condiciones (causales y de intervención) que

se cruzan dimensionalmente en este momento y lugar para crear el conjunto de circunstancias o problemas a los que responden las personas a través de acciones / interacciones.

Las acciones/interacciones son las respuestas a la pregunta de cómo las personas manejan las condiciones. Las acciones estratégicas son acciones que tienen un propósito en la solución de un problema y de ese modo dan forma al fenómeno. Las acciones de rutina representan las respuestas a las situaciones de cada día. Estos dos tipos de acciones deben ser investigados para comprender por completo cómo la gente mantiene el orden social.

Independientemente de que se tomen medidas en respuesta a una situación problemática o no, siempre hay consecuencias.

Las consecuencias del fenómeno necesitan ser descritas con el fin de entender un fenómeno completamente. Las consecuencias tienen propiedades tales como la duración, visibilidad, impacto, previsibilidad y alcance.

La codificación axial se trata de encontrar relaciones entre las categorías y las subcategorías. Estos representan los vínculos entre conceptos en las situaciones de investigación. Las predicciones o "hipótesis" de los investigadores acerca de cómo los conceptos se vinculan son declaradas. Estas relaciones deben ahora ser validadas contra los datos de los incidentes reales para determinar si pueden ser basadas en la realidad. Las contradicciones entre la realidad y la hipótesis ayudan a refinar la descripción de la categoría mediante el perfeccionamiento de las condiciones, acciones / interacciones y consecuencias de los fenómenos.

Codificación selectiva

La codificación selectiva es el proceso de refinar la teoría. Las categorías son sólo las descripciones de datos y no son aún una teoría. Las distintas categorías deben integrarse para formar una teoría. El primer paso es decidir una categoría central que represente el tema principal de la investigación. Se tiene que encontrar una intersección entre todas las categorías importantes de la investigación. Strauss y Corbin dan los siguientes criterios para elegir una categoría central:

- Debe ser central, es decir, todas las otras categorías principales pueden estar relacionadas a la misma.
- Tiene que aparecen con frecuencia en los datos. Esto significa que dentro de todos o casi todos los casos, hay indicadores que apunta a ese concepto.
- La explicación que evoluciona al relacionar las categorías es lógica y coherente. No se fuerzan los datos.
- El nombre o frase utilizada para describir la categoría central deben ser lo suficientemente abstracta para que se puede utilizar para realizar investigaciones en otras áreas sustantivas, lo que lleva al desarrollo de una teoría más general.
- A medida que el concepto se refina analíticamente a través de la integración con otros conceptos, la teoría crece en profundidad y poder explicativo.
- El concepto es capaz de explicar la variación, así como el punto principal realizada por los datos, es decir, cuando las condiciones variar las explicaciones aún se mantiene, a pesar de la forma en que se expresa un fenómeno podría ser algo diferente. Se debe también ser capaz de explicar casos contradictorios o alternativos en términos de esa idea central.

Existen varias técnicas para determinar la categoría central. Estos incluyen escribir la historia, haciendo uso de diagramas, y revisar y clasificar las notas ya sea a mano o por software. Cuando la categoría central es descrita, el perfeccionamiento de la codificación de las principales categorías debe llenar estos vacíos. El objetivo es escribir una historia a la que los incidentes en los datos se pueden instalar. Los casos que no se ajustan al argumento deben ser explicados en términos de las condiciones de intervención. El descubrimiento de casos periféricos y la creación de explicaciones en la teoría, aumenta su generalización y el poder explicativo.

Muestreo

A continuación se presenta una breve discusión sobre los procedimientos de muestreo. El muestreo teórico es definido por Strauss y Corbin como: la recopilación de datos impulsada por conceptos derivados de la teoría de la evolución y se basa en el concepto de "hacer

comparaciones ', cuyo propósito es ir a lugares, personas o acontecimientos que maximicen las oportunidades para descubrir variaciones entre los conceptos y para densificar categorías en términos de sus propiedades y dimensiones.

Durante la codificación abierta, el muestreo debe hacerse lo más amplio posible para permitir a los investigadores a estar abiertos a descubrir conceptos en la situación. Aunque el muestreo debe hacerse de forma sistemática, el investigador debe ser flexible para codificar cualquier caso de que encuentre interesante para el estudio. Durante el estudio, el investigador debe cuestionar y comparar los datos continuamente. Las respuestas a las preguntas del investigador darán lugar a una mayor toma de muestras y la codificación de más incidentes.

Durante la codificación axial, el muestreo se realiza para definir las dimensiones y las propiedades de las categorías, así como para definir las subcategorías y su relación con las categorías. Durante la codificación de selección, el muestreo se utiliza para fortalecer la teoría. Los incidentes se probaron para ajustarse a la teoría y la teoría se refina hasta que las categorías están saturadas. Esto significa que más codificación no altera la descripción de las categorías.

Cabe señalar que el muestreo no podría ser planificado en detalle antes del inicio del estudio de campo. Es importante para la teoría fundamentada descubrir una teoría en los datos y no poner a prueba una hipótesis previamente escrita mediante la recopilación de los datos pertinentes. No son las personas u organizaciones de las que se toman muestras, sino de los incidentes y eventos. Aunque el muestreo durante el inicio del proyecto es bastante desenfocado, se enfocará a medida que avanza el proyecto. La toma de muestras sólo terminará cuando todas las categorías están saturadas [1].

2.7. Trabajos relacionados

El método Grounded Theory ha sido usado como estrategia de investigación en muchos estudios y en varios contextos. Al investigar la naturaleza social de los equipos ágiles, podemos encontrar ejemplos de estudios usando Grounded Theory, como en [17] donde los

autores aplican esta técnica para investigar los roles de auto-organización en equipos de desarrollo ágil de software. Otro ejemplo puede ser encontrado en [32] donde se presenta a Grounded Theory como base de un estudio para investigar las iniciativas de implementación de mejoras en proceso software desde la perspectiva de consultores expertos en SPI.

A continuación se referencian también otros estudios empíricos sobre enfoques ágiles que han usado diferentes métodos de investigación como por ejemplo en [8] donde se muestra como los desarrolladores perciben *Scrum* a largo plazo analizando los datos con la Teoría de la Difusión de la Innovación. Otro ejemplo se puede observar en [5] donde se describen los resultados de una encuesta aplicada a empresas irlandesas para mostrar el progreso de la madurez de los principios y prácticas en el desarrollo ágil de software.

Capítulo 3. Metodología de investigación

En este capítulo se introduce la metodología de investigación usada para este estudio y como ha guiado la recolección de datos, el análisis y desarrollo de la teoría. En primer lugar, se proporciona una caracterización y contexto del estudio. Seguido de una definición del estudio experimental. El capítulo concluye describiendo las fases de recolección de datos para este estudio, el cual consiste de observaciones al acompañamiento realizado a las empresas seguidas de una encuesta.

3.1. Contexto y caracterización del estudio

El estudio se realizó con base en la información generada en la fase I del programa piloto de transferencia del conocimiento y apropiación de la metodología ágil *Scrum* auspiciada por la corporación Ruta N [22].
Este programa estuvo dirigido a empresas dedicadas al desarrollo de software en la ciudad de Medellín y el área metropolitana, pertenecientes a la Corporación Intersoftware y al Clúster TIC.

El programa se realizó durante el 2012 -2013 y su objetivo principal era fortalecer las competencias en la implementación de la metodología *Scrum* para la formulación de proyectos en personal especializado vinculado a empresas dedicadas al desarrollo de software de la ciudad de Medellín y el área metropolitana; en el marco de las acciones para el cierre de brechas tecnológicas identificadas en el Plan de Ciencia, Tecnología e Innovación de la ciudad 2011-2021, así como la consolidación redes colaborativas universidad empresa entorno a las metodologías ágiles [22].

El grupo objetivo del programa fueron 10 empresas desarrolladoras de software, que presentaron un equipo de máximo tres personas enmarcado en los siguientes perfiles

profesionales: gerente, arquitecto desarrollador, jefe de PMO u oficina de gestión de proyectos. Esto con el fin de asegurar la implementación práctica de la metodología en el mercado.

A continuación se presenta el esquema general del programa en la figura 3.1.

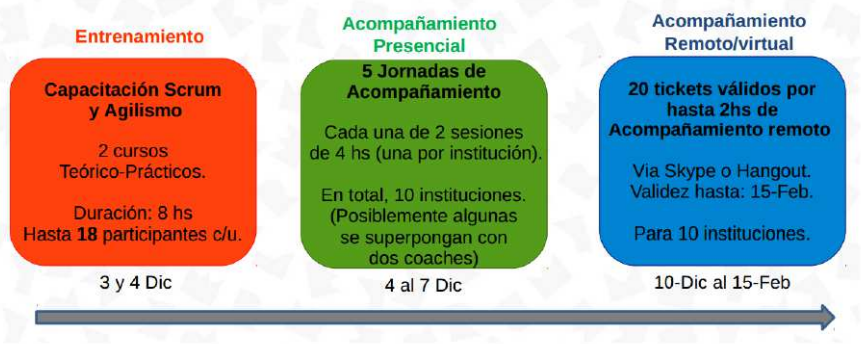

Figura 3.1: Esquema general del Programa para la transferencia y apropiación de Scrum – Ruta N. Tomado de [22].

3.1.1. Tipo de estudio

El tipo de estudio a ser usado en la investigación será un estudio experimental cualitativo primario in vivo basado en la observación usando la técnica de Grounded Theory y apoyado por la técnica de encuesta.

El método de investigación de campo (encuesta) fue escogido con el objetivo de apoyar los datos cualitativos resultantes de la fase de observación directa. La encuesta está conformada por un total de 15 preguntas divididas en 3 secciones: Información general del encuestado y el proyecto, Adopción actual de *Scrum* y prácticas complementarias y la sección de Percepciones generales. El anexo A presenta la encuesta diseñada para este estudio.

3.1.2. Dominio

Ingeniería de software, Mejora de proceso software, Gestión de proyectos.

3.1.3. Idioma

Español.

3.1.4. Participantes

- Universidad EAFIT. Departamento de Informática y Sistemas. Grupo de I+D+I Investigación en TIC, línea en Ingeniera de Software. http://www.eafit.edu.co/investigacion/grupos/escuela-ingenieria/i-d-i-tic/Paginas/i-d-i-tic.aspx
- Ruta N [http://www.rutanmedellin.org]. A través de esta entidad se hizo contacto con las empresas que participaron en el estudio.

3.2. Definición del estudio experimental

3.2.1. Objetivo general

Analizar: El proceso de apropiación de prácticas ágiles (*Scrum*) en empresas de desarrollo de software o servicios relacionados.
Con el propósito de: Caracterizar los logros, retos y lecciones aprendidas
Con respecto a: al uso de prácticas ágiles y su relación con las prácticas orientadas a planes (CMMI) ya adoptadas.
Desde el punto de vista de: Los investigadores, patrocinadores, consultores y miembros de los equipos de los proyectos bajo observación.
En el contexto de: las empresas participantes en el programa piloto de transferencia del conocimiento y apropiación de la metodología ágil *Scrum*.

3.2.2. Objetivos específicos

- Caracterizar a las empresas participantes del programa y el perfil de los miembros de los equipos de trabajo.
- Observar y recopilar la información generada durante el proceso de apropiación de prácticas ágiles en proyectos *Scrum* en las empresas del programa que anteriormente han adoptado prácticas de CMMI.
- Determinar cuáles son los logros, retos y lecciones aprendidas generadas durante la apropiación según los datos proporcionados por las empresas buscando que sea un referente para ejercicios futuros.

3.2.3. Foco de calidad

Desde las prácticas ágiles:

El foco de calidad está en el nivel de adopción de *Scrum* y prácticas ágiles complementarias y su relación con las prácticas orientadas a planes (CMMI) ya adoptadas en las empresas.
Para identificar cuáles son las prácticas ágiles complementarias a *Scrum*, se toma como referente el mapa de prácticas ágiles en [45].

3.2.4. Perspectiva de observación

Desde la perspectiva de observación de los investigadores, patrocinadores, consultores y miembros de los equipos de los proyectos bajo observación.

3.2.5. Selección del contexto

El contexto del estudio experimental está dado por el programa en el cual este se realiza, el cual fue explicado en la sección 3.1.
La entidad promotora del proyecto realizó un proceso de selección de las empresas beneficiadas del programa, las cuales, a su vez, serían las empresas participantes de este estudio. Estas empresas debían tener los siguientes requisitos mínimos [22],

- Mínimo dos años de constituidas en la ciudad de Medellín o el Área Metropolitana.
- Actividades productivas y comerciales vinculadas al desarrollo de software, de contenidos o aplicaciones móviles para los sectores energía, salud y TIC.
- Disposición para aplicar la metodología en el desarrollo de un caso práctico o proyecto informático.
- Conformación de grupos de trabajo de formación a nivel de pre y posgrado en áreas afines a la Ingeniería de Sistemas, Informática, Eléctrica y Mecánica; y disponibilidad manifiesta para la dedicación a las actividades comprendidas en el presente proyecto de un número de entre 1 y 3 personas.

En las observaciones del acompañamiento y en la encuesta participaron profesionales en el área de ingeniería de software con experiencia en el desarrollo de software y vinculados con las empresas del programa y consultores expertos en adopción de prácticas ágiles.

3.2.6. Preguntas de investigación

- ¿Cuáles son los retos durante la apropiación de prácticas ágiles (*Scrum*) en empresas de desarrollo de software que han adoptado CMMI?
- ¿Cuáles fueron las prácticas ágiles adoptadas por los diferentes equipos de trabajo en las empresas? ¿Cuál es el nivel de adopción de las prácticas?
- ¿Los retos identificados y caracterizados durante la apropiación de las prácticas ágiles son similares a los retos encontrados en la literatura?
- Cuáles son los principales beneficios y lecciones aprendidas percibidas por los profesionales de las empresas participantes del estudio

3.2.7. Técnicas o instrumentos para la recolección de datos

Durante el período de observación

La recolección de datos se realizó de manera directa, a través de la observación de los equipos de trabajo de las empresas participantes en el programa, durante las sesiones de acompañamiento establecidas, las cuales fueron de dos tipos: sesiones presenciales y sesiones virtuales.

Las sesiones fueron conducidas por el consultor experto y giraban alrededor del contexto particular de cada empresa. En algunas sesiones presenciales se siguió una estrategia parecida a un Open Space, que es una estrategia de trabajo en grupo generalizada dentro de la comunidad ágil. La dinámica general que siguieron las sesiones presenciales, es la siguiente

a. Se realizó una introducción sobre el método. Se explicó a los participantes cómo funcionaba y como se desarrollaba.
b. Se recopilaban las inquietudes. Cada participante con una inquietud la podía mencionar y se colocaba en una tarjeta. Luego en una ronda de calificaciones se decidía el orden de los temas y los menos votados quedaban al final en prioridad.
c. El consultor comenzaba a dar sugerencias sobre los temas y a responder preguntas en el orden de prioridad en que habían sido votados los temas.
d. Al final en 15 minutos se trataban de responder de manera muy rápida los temas que quedaban faltando en el tablero.

Para las sesiones virtuales, la estrategia giro en torno a preguntas más puntuales sobre la adopción en los proyectos.

En principio, la investigadora tomó un rol solo de observador pero a medida que fue conociendo a los participantes de los equipos, se involucró en algunas sesiones haciendo preguntas en contextos que eran conocidos para ella.

Algunas sesiones fueron registradas con un grabador de voz digital y los archivos transferidos a un PC para su transcripción. Las sesiones presenciales se realizaron en sitio en las oficinas de las empresas. Los participantes fueron citados y se les pidió que llevaran

la sesión, las preguntas relacionadas con *Scrum* y el contexto de proyectos con el cual trabajaban.

Después de la observación se transcribieron las sesiones y se codificaron inmediatamente, de esta manera los resultados de codificación podrían ayudar a estructurar algunas preguntas para las próximas sesiones. De hecho un valor agregado para las empresas fue el envío de un resumen de las transcripciones de las sesiones realizadas y de información relacionada con la comunidad ágil en Medellín y Colombia.

Cuando se transcribieron las sesiones, los nombres de las empresas y participantes fueron reemplazados con códigos, de P1 a P8 para las empresas y solo se mostró el rol del participante. Para este estudio se usó la hoja de cálculo Microsoft Excel para ayudar a manejar los datos y hallazgos de la observación y un procesador de palabras, Microsoft Word para apoyar el proceso de investigación.

Siguiendo la metodología de Grounded Theory sobre la codificación (ver sección 2.6), el investigador trabajó sobre cada una de las transcripciones y codificó línea por línea para tomar nota de los temas y fenómenos al margen, los puntos claves se crearon como una columna aparte. A través de este ejercicio se escribieron memos para mantener un seguimiento a las ideas y pensamientos más allá del análisis de los datos.

Al final del ejercicio de codificación se creó una matriz que contenía en sus filas cada una de las preguntas y respuestas de los participantes y en sus columnas códigos, conceptos, categorías y memos.

Este listado de códigos fue revisado continuamente mientras más observaciones eran hechas. Los códigos fueron modificados y verificados para soportar el proceso de análisis.

El proceso de codificación facilitó la reflexión sobre códigos y categorías, los cuales fueron capturados escribiendo memos. Estos memos fueron consultados al crear relaciones entre las categorías y se configuró el marco de trabajo teórico inicial.

Posterior a la observación

Una encuesta fue conducida a través de un cuestionario y distribuida después del periodo de acompañamiento con las empresas. La encuesta complementó los hallazgos de las sesiones y ayudó a obtener una mejor idea de las opiniones de los participantes sobre la adopción de *Scrum* en sus empresas. Una ventaja con la encuesta fue recolectar gran cantidad de datos en corto tiempo.

La población objetivo de esta encuesta, fue una muestra de los mismos profesionales que formaban parte de las empresas vinculadas al programa. Los participantes de la encuesta fueron seleccionados de manera directa según el juicio y conocimiento del investigador, teniendo en cuenta que su objetivo era dar mayor fuerza o validación de los hallazgos encontrados en la fase de observación directa.

Se distribuyó el cuestionario a unas 84 personas de las cuales contestaron 34. Se siguieron directrices éticas con el manejo de la información: tratamiento confidencial de los datos, lo cual fue explicado a los participantes. Los participantes tuvieron la libertad de enviar un email después de responder la encuesta si estaban interesados en tomar parte de la rifa de un bono para reclamar un libro.

Se realizó un piloto para probar el cuestionario antes de difundirlo para detectar preguntas ambiguas o confusas, para este piloto se enviaron 5 cuestionarios a amigos y la asesora para verificar que se cumplía con los objetivos del estudio. Algunas de las preguntas y opciones se cambiaron a raíz de la retroalimentación recibida.

El cuestionario fue diseminado online a través de internet, se reconoce que un sesgo del investigador podría haber ocurrido al seleccionar a los participantes, sin embargo muchas de las encuestas se respondieron debido a que la distribución del cuestionario se hizo sobre esta base personal. El número de participantes que no respondieron y sus razones para no tomar parte del estudio no podrían ser conocidos.

La versión online de la encuesta fue publicada en https://www.limeservice.com/. Esta forma de distribución resultó ser muy conveniente para recolectar los datos y generar gráficos de los mismos. El diseño de las preguntas puede ser vista en el anexo A.

Capítulo 4. Análisis de los datos recolectados

En este capítulo se da la discusión final del resultado integrando las dos partes del estudio.

4.1. Análisis de resultados durante el período de observación

Uno de los hallazgos claves que encontramos al realizar este estudio es que los diferentes retos y dudas generadas durante el proceso de apropiación de *Scrum* se organizan alrededor de 3 temas principales: los valores establecidos en el manifiesto ágil, los elementos de *Scrum* y las prácticas complementarias a *Scrum*. Estas categorías "core" junto con los conceptos más relevantes se presentan en la figura 4.1.

El anexo B presenta el detalle de las observaciones que apoyan estos resultados.

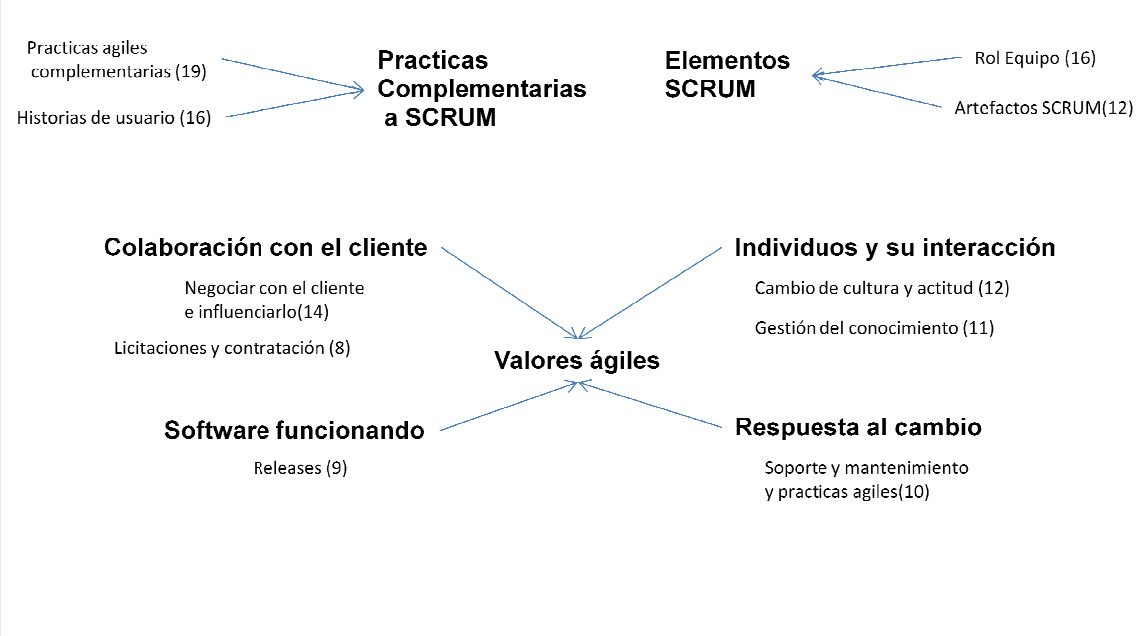

Figura 4.1: Categorías "core" y conceptos relevantes asociados a los retos.

4.1.1. Practicas complementarias a *Scrum*

Se evidencia que el mayor número de dudas que expusieron los participantes en las primeras sesiones de acompañamiento se relacionan con la integración del marco de trabajo *Scrum* y otras prácticas que la complementan en ingeniería.

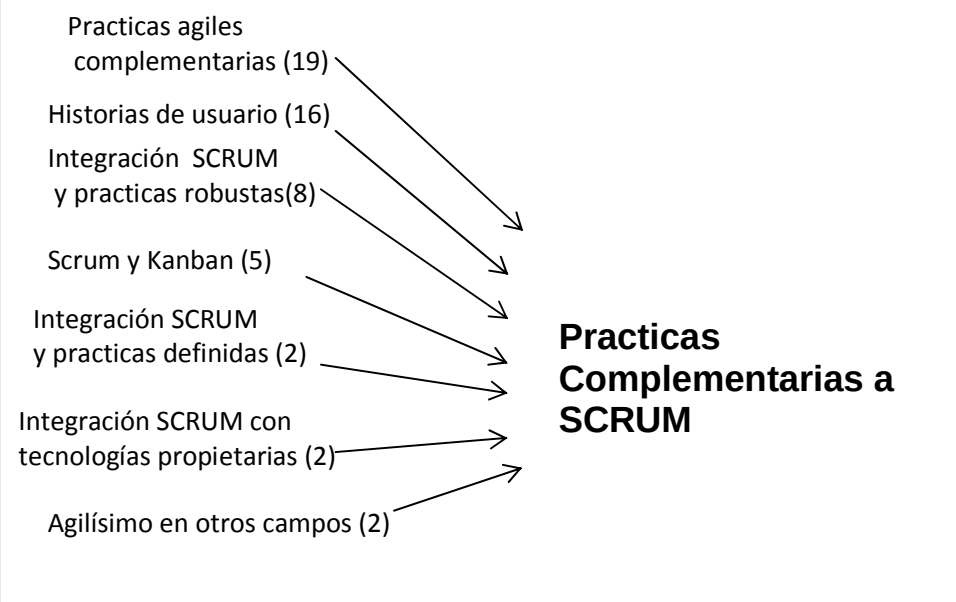

Figura 4.2: Retos asociados a Prácticas Complementarias a *Scrum*.

Entre las prácticas complementarias a *Scrum* mencionadas como retos o dudas se encuentran: refactoring, pruebas automatizadas, pruebas unitarias y de integración, integración continua, reuso de componentes, XP, trabajo en pares, TDD y DSDM.

Algunos participantes perciben la práctica de *refactoring* como una actividad que consume mucho tiempo si se contempla dentro de todos los *sprints*, todo el día y todos los días.

> Los consultores recomendaron esta práctica por la mantenibilidad que se da al producto y porque permite la transferencia de conocimiento desde expertos a novatos, sobre todo si se combina con programación en pares al usar TDD, además permite dejar el código organizado, principalmente si se inicia el proyecto con conocimiento de *refactoring*

Otro hallazgo fue que al inicio de la adopción de *Scrum* no existía el uso de pruebas automatizadas o de integración continua en algunas de las empresas participantes.

También se percibe dificultad en implantar la práctica de programación en pares pues no se conocen datos reales sobre la mejora de la productividad o calidad relacionada con su uso, no se percibe la ventaja de realizar esta práctica pues algunas veces al realizar el trabajo en pares uno de los desarrolladores no participa de forma activa y más bien adopta una posición relajada y que no aporta conocimiento.

Por otro lado, dentro de las dudas iníciales que expusieron las empresas también se encuentran aquellas relacionadas con la conexión entre las historias de usuario y otras prácticas ya consolidadas dentro del proceso de requisitos como casos de uso y prototipos. Encontramos que no es claro para las empresas hasta qué nivel de detalle se manejan las historias de usuario, cómo se documentan las conversaciones adicionales, cuáles son los elementos de una historia de usuario, cómo se manejan los requisitos no funcionales, cómo se realiza una transición entre casos de uso y las historias de usuario, si una técnica reemplaza a la otra o se pueden complementar.

Otro reto observado es la manera como se integra *Scrum* con prácticas orientadas a planes como las del modelo CMMI y PSP/TSP. Se percibió por parte del investigador una mayor resistencia a integrar prácticas ágiles cuando ya anteriormente se había adoptado prácticas de CMMI y PSP/TSP. Uno de los dolores expresados por las empresas que ya tienen implantadas prácticas orientadas a planes como CMMI o PSP/TSP y que han realizado valoraciones para este modelo, es perder la trazabilidad entre los artefactos generados durante el proceso de desarrollo, porque se entiende que el manejo dado por *Scrum* a estos artefactos es realizar la documentación mínima y suficiente. Algunas veces es mal entendido el valor ágil de software funcionando por encima de la documentación, asumiendo que se refiere a una eliminación de productos de trabajo del proceso de desarrollo sin ningún tipo de criterio.

Los retos antes mencionados se relacionan con dudas expuestas sobre cómo integrar las prácticas ya definidas en sus procesos y *Scrum* y cómo usar las prácticas ágiles como *Scrum* sin dejar de lado las prácticas ya definidas por la organización.

Siguiendo el orden de los registros relacionados en la categoría de Prácticas complementarias a *Scrum*, analizaremos los retos asociados al concepto de *Kanban* y *Scrum.*
La propuesta de las empresas con relación a *Kanban* se centra en usar esta técnica japonesa de visualización del trabajo planeado, para gestionar el product backlog de proyectos de soporte y mantenimiento. Unos de los retos observados al usar *Kanban* al gestionar la asignación de tareas en estos proyectos, es cómo especificar el límite del trabajo en progreso para cada estado del procesamiento de una orden de cambio en soporte y mantenimiento. También es importante aprender a identificarla información necesaria en cada tarjeta del tablero *Kanban*, existen dudas sobre cómo se estima cada tarea u orden de trabajo, su prioridad, su fecha de asignación.

Durante las observaciones, dos de las ocho empresas mencionaron tener retos y dudas respecto al uso de TDD, así como con el desarrollo con código propietario y cerrado, pues es difícil la automatización de pruebas. También se tienen dudas referentes a la adopción de *Scrum* en el desarrollo de software usando la solución SIEBEL.

El agilismo se extiende a otras disciplinas diferentes al desarrollo de software dado su naturaleza basada en los individuos y sus interacciones que es un valor presente en todas las disciplinas. Algunas empresas que usan actividades alternas al desarrollo de software a la medida, como por ejemplo consultoría, mercadeo o comunicación se hacían las siguientes preguntas: ¿qué otras áreas pueden beneficiarse de las practicas ágiles?, ¿Es posible aplicarla en administración?, ¿En proyectos de publicidad?

4.1.2. Elementos *Scrum*

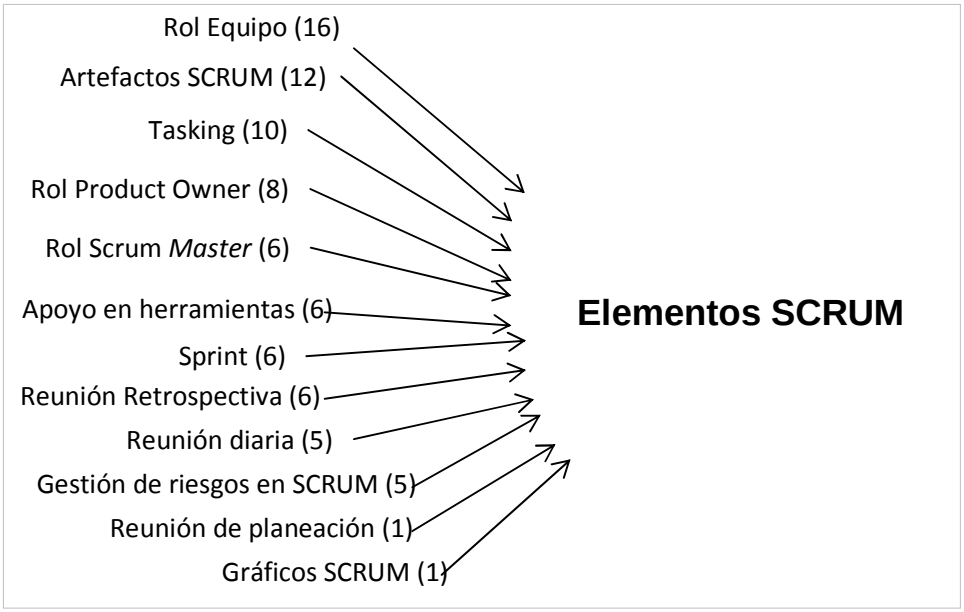

Figura 4.3: Retos asociados a Elementos *Scrum*.

Otra categoría 'core' emergente de los datos analizados es la de ***Elementos Scrum*** y se refiere a un listado de conceptos y prácticas necesarias para usar *Scrum.*

El concepto de **Rol de equipo** se asocia a un equipo de desarrollo *Scrum*, el cual es un grupo de individuos multi – funcionales y auto-organizados, trabajando sobre un proyecto y que tienen la responsabilidad de cumplir las metas del *sprint*, sin embargo se supone que sea autónomo y que tenga control sobre el proceso exacto [6].

La mayoría de los retos que surgieron alrededor del rol de Equipo, se relacionan con el manejo de la multitarea (multitasking). Se observó que algunos miembros de equipos estaban asignados a varios proyectos o tareas por vez debido a las habilidades y conocimientos que poseen o a la etapa en la cual se encuentra el proyecto, por ejemplo en mantenimiento una misma persona puede estar asignada a varios proyectos o clientes debido a la variación en la demanda de cambios en los productos para los cuales se contrató el servicio. La asignación de una persona a varios proyectos o tareas para trabajar en "paralelo" también podría estar relacionada con la percepción de que cuanto antes se inicie un tarea, antes se termina y que una persona es un recurso y no se puede desperdiciar ni un

minuto de su tiempo. La multitarea es uno de los factores que puede influenciar la productividad, calidad y satisfacción del cliente en trabajos basados en el conocimiento y aunque parece que ha aumentado la productividad de las empresas no se percibe que el individuo no está trabajando a su máxima efectividad, debido quizá a la naturaleza incontrolable de la multitarea y su efecto sobre actividades como el pensamiento crítico y de alta calidad, en conclusión, la efectividad de la persona se ve afectada porque cambiar de un proceso a otro toma un tiempo [23].
Por lo anterior, algunos consultores recomiendan enfocarse en una tarea por vez, lo cual permitiría minimizar el tiempo que transcurre entre el inicio de una tarea o asignación hasta que esta se completa (Lead Time). Se sugiere también revisar técnicas de manejo del tiempo como *Pomodoro* para aumentar la efectividad de las personas.

Otras dudas alrededor de la multitarea se centraban en saber cuál es la mejor forma de organizar las reuniones en *Scrum* cuando hay personas que pertenecen a varios equipos, cómo manejar la naturaleza multi-funcional de los equipos en *Scrum*, por ejemplo, dado que en el mismo sprint una persona debe realizar varias tareas como priorizar cual tarea debe realizar primero y qué usar para manejar grupos virtuales.
Para esta última duda los consultores sugieren usar la técnica *Kanban* y herramientas asociadas como *Trello*, el cual es un software de apoyo para asignación de tareas.

Otras consideraciones que emergieron con el concepto de **Rol de Equipo**, fueron las relacionadas con la conformación de estos equipos *Scrum*. Respecto a este tema surgen varios retos, por ejemplo, definir cómo se distribuyen los roles necesarios entre las personas que hacen parte del equipo, si el personal de calidad seguirá apoyando el proceso software como un equipo independiente, ¿cómo se conforma un equipo multi-funcional, cómo se definen los salarios, cuantas personas son necesarias?

El concepto de **Artefactos *Scrum*** se asocia a la información generada durante las actividades del flujo *Scrum*, dentro de estos artefactos se encuentra el *Product Backlog*, que

es un listado evolutivo de las funcionalidades de negocio y técnicas que deben ser desarrolladas [6].

El *product backlog* se observó como un artefacto crucial durante la fase de adopción, se generaron muchas preguntas alrededor de este elemento, por ejemplo, ¿Qué se debe incluir como parte del *product backlog*?, ¿Cuáles roles son necesarios en las reuniones de planeación por parte del cliente: personal de tecnología, personal del área financiera?, ¿Cuáles roles se recomienda deberían asistir a la reunión de planeación por parte del proveedor: técnicos, área de comercial? ¿Qué tipo de técnicas usar para lograr una adecuada priorización del backlog?, ¿cómo identificar lo que tiene valor para el cliente?

Dentro de las sugerencias dadas por el consultor se explica que si asisten personas del área financiera a las reuniones de priorización del backlog es una buena oportunidad para convencerlos de que la práctica de entregas tempranas desde el punto de vista financiero es mucho mejor pues el proyecto se puede pagar solo. Dentro de las estrategias sugeridas para manejar la priorización del *product backlog* se habló del *Visual Story Mapping*, la cual permite identificar cuales opciones son esenciales y de valor para el cliente y además permite generar sinergia dentro del equipo. También se recomienda priorizar usando la técnica de *Pareto* y siempre trabajar en lo que el cliente piensa es más valioso.

Durante el uso de la técnica de *Visual Story Mapping* se tuvieron dudas acerca de cómo realizar una transición entre la información que inicialmente se consolidó en una hoja de cálculo al mapa visual de la técnica.

Otro reto asociado a la priorización del *product backlog*, surgió dentro del contexto de proyectos de soporte y mantenimiento bajo un marco contractual regido por acuerdos de niveles de servicios (ANS). En este caso el *product backlog* es un listado de requerimientos del cliente y se entiende que en todo momento será lo que está asignado o no en la herramienta o mesa de servicios del cliente. Sin embargo la priorización de los requerimientos se define de acuerdo a su urgencia de resolución y ha sucedido que los desarrolladores deben posponer tareas ya asignadas para poder estimar y trabajar en los nuevos requerimientos con acuerdos de niveles de servicios más estrictos.

Solo se observó una duda acerca del artefacto *Impediment backlog*, relacionada con el contenido que debe ser registrado allí, sobre los demás artefactos *Scrum* no se observó duda alguna.

Uno de los resultados de la reunión de planeación del *Sprint* es un plan tentativo para iniciar el *Sprint*, el ejercicio de identificar y asignarlas tareas que harán parte del *Sprint backlog* es lo que conocemos como **Tasking**, y este fue uno de los conceptos emergentes de los datos observados. Se encontró que cuando se están definiendo las tareas, el reto para los equipos es identificar cuáles son las tareas asociadas a una fase (requisitos, diseño, codificación, etc.) y su nivel de detalle.

Las sugerencias dadas al respecto durante la sesión de consultoría fueron las siguientes:

El conocimiento al inicio de una iteración es sobre lo que necesita el usuario y no como se va a realizar, luego durante una segunda fase de la reunión de planeación se hace el ejercicio de desglose en tareas, que pueden ser tareas orientadas a entregables o solo tareas o una mezcla de ellas, de forma tal que les queda como llevar adelante ese desarrollo, es un aprendizaje del equipo y se va ajustando constantemente.

Las tareas asociadas a una fase, por ejemplo a diseño, o arquitectura si van dentro del sprint. En diseño se deben tener en cuenta tareas como diseño de las responsabilidades en clases de negocio, distribución de carga, otros atributos de calidad, patrones de persistencia, llamado de servicios, qué framework usar.

En proyectos de soporte y mantenimiento se presenta un reto con el tamaño de las órdenes de cambio solicitadas por el cliente y que tienen varias asignaciones dentro o con órdenes de cambio tan grandes como para manejarlas con una sola tarjeta. Se sugiere desagregar las tareas por disciplinas del proceso de mantenimiento o asignaciones internas de la orden de cambio, lo cual permite un seguimiento más eficaz.

Dos de las empresas observadas durante la consultoría expusieron dudas relacionadas con nuevas tareas que surgen durante la ejecución de un sprint. ¿Cómo tener la flexibilidad de incluir tareas adicionales o urgentes que llegan de improviso y que suman horas adicionales a la planeación inicial?

El consultor sugiere reaccionar lo más rápido posible y decidir en la daily meeting quien puede tomar estas nuevas tareas dentro del equipo.

¿Qué pasa con las tareas bloqueadas durante un sprint? Cuando haya tareas en progreso y estas resultan bloqueadas por cualquier razón durante el sprint, lo que suele hacerse es descomponerla en tareas que pueda terminar, las que están listas las paso a done, y luego se descarta la tarea original.

El rol de ***Product Owner*** es el responsable de representar los intereses de cualquiera con un interés en el proyecto y su sistema resultante, consigue la financiación inicial y permanente para el proyecto creando los requisitos iniciales del proyecto, los objetivos del retorno a la inversión (ROI) y los planes de liberación del producto [2].
La intervención del *Product Owner* durante las reuniones diarias es un reto pues algunas veces no es entendida la naturaleza de auto organización del equipo y el Product Owner interfiere en la reunión involucrándose demasiado en las decisiones que el mismo equipo debería tomar. Se percibió un Product Owner que tendía a estar "muy preocupado" en dar directrices al equipo.
En otra empresa se observó que la misma dinámica de las reuniones diarias le indicó al *Product Owner* si él estaba generando valor.

Los equipos compartieron su preocupación sobre cómo incrementar la participación activa del cliente y hasta donde se debía involucrar el *Product Owner*, sobre todo cuando este tiene poco conocimiento del negocio.
Una de las empresas observadas percibe que el tema cultural es muy complicado, y que la selección del *Product Owner* se hace con ligereza, es necesario tener criterios más adecuados para la elección de este rol, sobre todo por las responsabilidades que tiene, se le debe dar la autoridad y autonomía para tomar decisiones sobre el producto, muchas veces hay que esperar demasiado por una decisión porque el *Product Owner* debe consultar con otras personas. Se observa que esta misma empresa usa el esquema de tener un *Product Owner* del cliente que trabaja con el equipo y que ha funcionado; sin embargo sugieren

tener un *Product Owner* del lado del proveedor que conozca del negocio y otro del lado del cliente.

El ***Scrum Master*** es un rol de gestión introducido por *Scrum*. Su misión es facilitar el proceso y ayudar a las personas a resolver problemas (incluidos los psicológicos) mientras refuerza las reglas del proceso [6]. Dentro del proceso de adopción, el *Scrum Master* juega un papel crucial y se observó lo siguiente relacionado con este rol:

¿Cuán necesario es el *Scrum Master* en equipos auto organizados?, ¿Cuál es su papel para saldar la deuda técnica del proyecto?, ¿Dentro de un modelo Kanban es válido tener en cuenta este rol?

¿Cuán valida es la siguiente propuesta de una empresa como apoyo a su servicio de soporte y mantenimiento?

Se percibe un reto en el tema cultural pues aún hay roles específicos dentro de los equipos que asignan las tareas en los tableros, en parte las empresas aceptan que no se tiene la cultura de auto asignación y que si no existiese este rol todo sería muy caótico.

Una empresa comenta que el rol de *Scrum Master* no necesariamente es un rol para personas que anteriormente hayan sido gerentes de proyectos, pues si tiene las cualidades y características idóneas y se apoya con un proceso de aprendizaje muy fuerte puede llegar a ser *Scrum Master*.

Durante las sesiones de coaching virtual se generaron dudas relacionadas con el apoyo en herramientas para potenciar la adopción de prácticas ágiles en las empresas.

Por ejemplo, cuáles son las herramientas sugeridas para la automatización de pruebas, para las métricas de *Scrum* como burndown chart.

Se observó que una de las empresas realizó un cambio táctico desde una herramienta virtual como Google Docs a una herramienta física como el tablero Kanban usando post its. Sin embargo el consultor comentó que a veces es necesario usar ambas técnicas para poder tener un historial de las métricas del equipo.

Otra de las empresas expuso como uno de sus retos el no adicionar más complejidad al proceso de desarrollo con más herramientas de apoyo a prácticas agiles y que se esperaba ir evolucionando a herramientas más complejas.
Otro reto identificado dentro del concepto de apoyo en herramientas para prácticas agiles es la integración con herramientas de BI.

Dentro de la categoría de Elementos de *Scrum*, emergió el concepto de *Sprint* y uno de los retos en la adopción fue que no se respetó la duración fija (timeboxing) y el sprint se extendió en vez de negociar el alcance comprometido por el equipo. Se generaron dudas relacionadas con las tareas que no se terminaban en un sprint.

Otras dudas generadas durante las observaciones fueron: ¿Qué tipos de *sprints* existen?, ¿A qué se refiere el *sprint* cero, el *sprint* de planificación?, ¿Puedo planear un *sprint* con tareas para trabajar en una deuda metodológica con CMMI?
Aunque es típico que los tres primeros *sprints* sean problemáticos, lo importante es ser proactivos y tener una estrategia de cómo actuar en esos casos.
Durante las observaciones se generaron dudas sobre la reunión retrospectiva, entre ellas: ¿Cuál es la periodicidad de las reuniones retrospectivas?, ¿Cuáles métodos de retrospectiva existen?, ¿Qué relación existe entre las reuniones retrospectivas y las lecciones aprendidas? ¿En qué documentos se pueden registrar las lecciones aprendidas y mejoras generadas?
El consultor comenta que la información generada en una retrospectiva incluye las acciones concretas de mejora con responsable, fecha de límite de ejecución y estas pueden estar registradas en documentos, papeles, fotos, citas en calendarios, etc.
Las experiencias reportadas por una empresa muestran que las reuniones retrospectivas tienen un impacto positivo dentro de los equipos de trabajo. Como lo ilustra la siguiente reflexión de un gerente de proyecto:
"¿Cómo podríamos enfocar las reuniones de retrospectiva propuestas en Scrum, en el modelo Kanban? Hicimos retrospectivas y todo les ha parecido muy bueno, el tablero… ya se están haciendo más efectivas las reuniones con el seguimiento del tablero. Se van a

manejar indicadores individuales que hacen parte de la cultura y con esto se moviliza al equipo.”–P1, Gerente de proyecto.

Otro concepto relevante dentro de la categoría Elementos *Scrum* es la reunión diaria, al inicio de la adopción se observó que los equipos se dispersaban al realizar esta reunión y que se tomaban parte del tiempo de la reunión diaria para tratar de resolver los impedimentos, lo que generaba que a veces se extendiera la reunión, se observó que la mayoría de los miembros del equipo tienden a dar informe al gerente de proyectos y que aún no se tiene la disciplina de informar al mismo equipo.

La mezcla de las reuniones diarias de *Scrum* usando el tablero de *Kanban* es una de las prácticas que han usado algunos equipos observados, sin embargo se presentan varias dudas al respecto: ¿qué clase de columnas usar? ¿Cómo definirlas?, ¿Cuándo pasar de una columna a otra?, Una de las necesidades expuestas es la de realizar ejercicios prácticos sobre la ejecución de esta reunión diaria.
Esta combinación se sugiere porque se pueden manejar los impedimentos, el aprendizaje y las insatisfacciones del equipo además de la prioridad de las tareas, sin embargo surge la duda sobre si ¿es recomendable y efectivo usar tableros adicionales al de Kanban durante el desarrollo de estas reuniones diarias, por ejemplo un tablero de indicadores?
Es de anotar que las reuniones diarias aportan más valor en un contexto que en otro.

Según los comentarios de los participantes, la gestión de riesgos en *Scrum* es otro reto para manejar en empresas con CMMI, pues la identificación y registro de los riesgos no es algo explicito dentro de las prácticas de *Scrum* sino que es una práctica continua que identifica los riesgos a través de las reuniones diarias y que se gestionan a través de las acciones tomadas para eliminar los impedimentos. El mayor riesgo, afirma, es no cumplir con los principios del manifiesto ágil o sus valores y que los equipos no tengan inyectada la mejora continua que permita llegar a mejorar. Es importante fallar rápido, aprender y después no fallar.

Dentro de las dudas más comunes acerca del manejo de riesgos están ¿Dónde se lleva el registro de los riesgos identificados y mitigados durante cada sprint? Y ¿Cuáles son los riesgos reales para que no funcione *Scrum*? Un equipo *Scrum* observado percibe que la mitigación del riesgo no tiene un dueño, el consultor comenta que en equipos auto-gestionados no hay dueño de nada, sino la sinergia de un equipo.
La reunión de planeación y los gráficos que usa *Scrum* para mostrar la cantidad de trabajo restante durante cualquier momento del sprint, como el *burndown chart*, fueron los conceptos que menos se mencionaron durante las sesiones observadas.

4.1.3. Retos alrededor de los Valores del manifiesto ágil

De manera natural las categorías de retos se fueron agrupando alrededor de los 4 valores del manifiesto ágil. Se podría afirmar que todos estos retos, surgen del cambio de cultura que se debe dar en las organizaciones que ya tienen un proceso maduro establecido.

Retos asociados a Individuos y su interacción

Los conceptos relacionados con este valor ágil que emergieron durante el estudio están definidos en la figura 4.4.

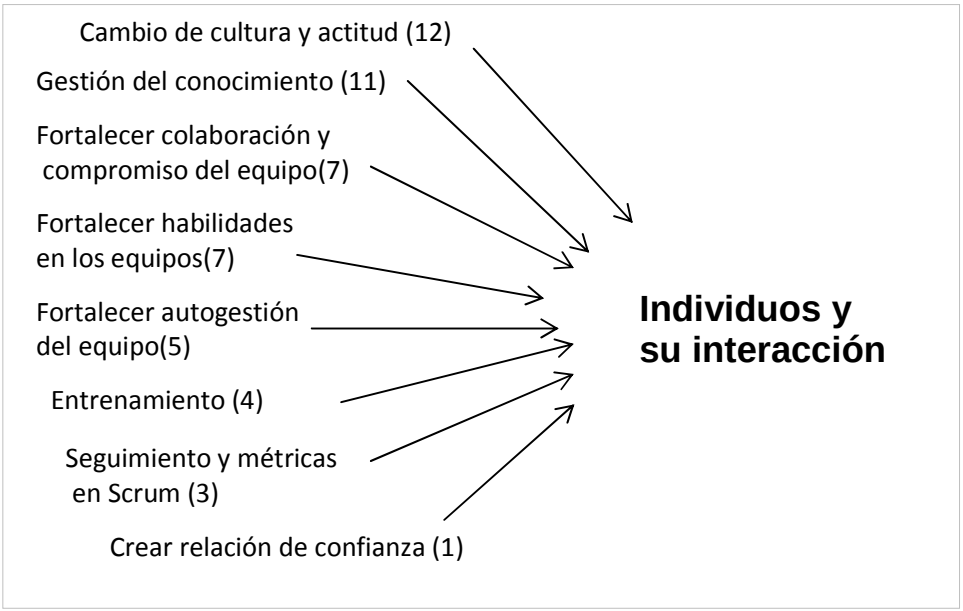

Figura 4.4: Retos asociados a Valores del Manifiesto Ágil – Individuos y su interacción.

Cambio de cultura y actitud

La mayoría de las observaciones mostraron que los equipos perciben como un reto el cambio de cultura y actitud para llegar a ser equipos más auto-organizados y disciplinados.

Los equipos de trabajo de la mayoría de las empresas de desarrollo están conformados por una mezcla de personal experimentado y personal nuevo poco experimentado. Algunas personas más novatas y jóvenes tienen poca experiencia de trabajo en equipo y aun no desarrollan habilidades como la auto-organización, es un gran reto para las empresas la conformación de equipos auto-gestionados con esta mezcla de personal novato y experimentado.

Un hecho observado por los equipos y muy repetitivo fue la falta de disciplina con las reuniones diarias.

Se pueden usar técnicas si no existe la disciplina de llegar a tiempo a las reuniones diarias, por ejemplo, se puede hacer una dinámica en la cual se deba pagar algo por llegar tarde.

Un reto para las empresas es incentivar a los empleados para que trabajen como un equipo auto-organizado, que se sientan comprometidos con el resultado que se va a entregar al cliente. Desde las áreas de gestión humana se tiene el reto de incentivar a los empleados sin premios o castigos sino haciendo a través del reconocimiento y del trabajo en equipo.

El cambio de actitud y cultura también se espera al estimar con juicio experto, esta método juega un papel fundamental al estimar en *Scrum*, y aunque algunas empresas realizan la estimación usando este método, el reto de las empresas es cambiar la manera actual al estimar con juicio experto basándose solo en la experiencia de una sola persona que a veces ni siquiera hará parte del equipo.

Otra pregunta interesante que emergió fue ¿Cómo cambiar la cultura de tener todos los requisitos definidos desde el inicio del proyecto y pensar que nunca más se necesitarán conversaciones con el usuario por que se cree que el detalle del requisito ya contiene toda la información necesaria hasta el final del proyecto?

Otra reto que emergió dentro de la categoría de individuos y sus interacciones es que los equipos aún esperan que alguien les esté guiado o dando órdenes y el equipo no asume su responsabilidad de mejorar o tomar decisiones.

Lo anterior se puede relacionar con el hecho de que las empresas tienen dudas sobre cómo adaptar los roles y cargos existentes en las compañías con los roles definidos en *Scrum*.

Una de las empresas comentó que algunas actividades que se hicieron con los equipos para cambiar el esquema de pensamiento fueron apoyarse de expertos, buscar que todos hablaran un mismo lenguaje pero aún se tiene que mejorar el tema de las habilidades “blandas” en los equipos.

Gestión del conocimiento

Saber conformar el equipo con los conocimientos y experiencias adecuadas es la clave de todo buen equipo *Scrum*, pero tener ese *background* toma tiempo y se realiza a través de la transferencia del conocimiento desde los más expertos a los más novatos. A continuación se expone una estrategia usada por una de las empresas para transferir el conocimiento:

"Se debe conocer muy bien el negocio antes de tirar línea de código, de acuerdo a sus capacidades asígnele [al empleado] responsabilidades, hay veces que toca llegar a eliminar esos vicios. Se deben tomar los papeles de acuerdo a los conocimientos y habilidades, se deben explicar los conocimientos entre los miembros del equipo para que vaya formando a otros. Se debe nivelar el conocimiento porque si no el mismo equipo lo saca, es darle la carga que cada miembro puede soportar. Como alguien que va al gimnasio y quiere musculo que va cargando cada vez más peso." P8, Gerente de proyecto.

Por lo anterior un reto identificado es el manejo de la transferencia de conocimiento entre un novato y un experto que permita dejar fluir la curva de aprendizaje y que sea efectiva. Hay estrategias de la gestión del conocimiento en las cuales se pueden apoyar y consiste en identificar los diferentes niveles de conocimiento y saber entre que niveles es más eficiente la transferencia de conocimiento. ¿Esta transferencia se facilita cuando el equipo se encuentra en sitio pero como manejar la transferencia de conocimiento en los equipos distribuidos o en sitios remotos?

Otro reto que emergió se relaciona con el conocimiento técnico necesario para resolver un problema o definir la solución a ser implementado.
Como equipo se necesita entender y explorar, para lo cual existen técnicas como *spike* que permiten hacer pruebas de concepto y conocer sobre el negocio. También es importante que los equipos trabajen en pares.

Otro de los retos que las empresas tienen relacionado con la gestión del conocimiento es el manejo de las lecciones aprendidas, para modelos como CMMI es necesario tener evidencia de la gestión de las lecciones aprendidas, es preciso definir que la información generada por las retrospectivas quede registrada en algún sitio, y se registra estructurada con elementos básicos y en una plataforma que permite la búsqueda futura por parte de otros equipos, se sugiere que para distintos proyectos participen de distintas reuniones ínter equipos, *Scrum* de *Scrum*. Las lecciones aprendidas se exponen de forma oral.

Colaboración y compromiso del equipo

Se percibió al inicio de la adopción falta de compromiso grupal y se observó que algunas veces los miembros de los equipos no veían viable adelantar tareas de los compañeros aunque ellos hubieran terminado las suyas, el motivo que se expone es la especialización que existe en los roles, es decir, no se tienen las habilidades y conocimiento del compañero como apoyarlo con las tareas, actualmente se auto asignan tareas pero de ellos mismos no de otro compañero. Se evidencia además que no existe una cultura de auto-asignación de tareas pendientes, la colaboración se expresa de formas diferentes pero cuando llega la hora de tener iniciativa con la auto-asignación de tareas pendientes esta no se da y cuando no existe un jefe que asigne las tareas el equipo entra en caos.

Durante el seguimiento es importante aprender a manejar las excusas de algo con lo cual se comprometió el equipo y no se cumplió, se sugiere planear tareas pequeñas para ver más fácilmente el concepto de “hecho”.

Una de las empresas percibe falta de compromiso del área comercial como parte del equipo para lograr lo que prometen al cliente. Este tema contempla el sentimiento de abandono que perciben los equipos por parte del área comercial que realiza acuerdos iniciales con el cliente.

Fortalecer habilidades en los equipos

Este concepto emerge de los datos y se observan dos corrientes, por un lado se mencionó la falta de habilidades técnicas de los equipos de calidad relacionada con prácticas de TDD, pruebas automatizadas y programación en pares. Se percibe que los equipos de calidad en vez de ayudar al equipo están siendo un obstáculo para que el equipo sea más veloz.

Algunas empresas optaron por desaparecer el área de calidad dándole herramientas técnicas al equipo para que hiciera esas tareas de calidad, sin embargo el verdadero reto es como cambiar la mentalidad de miembros del equipo que creen que la responsabilidad de la calidad es solo de los que realizan pruebas o inspecciones, sin sacrificar roles tan necesarios como el de pruebas.

Esta necesidad quizá surge por prácticas complementarias a *Scrum* que sugieren la automatización de pruebas como una forma de aumentar la eficiencia del proceso, consultores afirman que al inicio se tardará más tiempo pero luego se ve el retorno a la inversión.

Por otro lado se mencionó en las conversaciones sobre la necesidad de fortalecer en los equipos habilidades blandas como la asertividad, la comunicación, la auto organización, la auto gestión, visión global:

*"Como equipo hay que buscar mecanismos para que esa persona vaya a tu mismo ritmo de trabajo porque o si no todo el equipo va a la velocidad del más lento y a veces recurrimos al Scrum Master para que remueva ese impedimento, ellos como equipo no lo hacen porque dicen que no son sapos, es un tema cultural."*P8, Gerente de proyecto.

Parece fundamental mejorar las capacidades de auto gestión y auto organización de los equipos, un problema común es la asignación de tareas pues se viene de un esquema en el cual se espera la asignación de una tarea por parte de un jefe y aun no se tiene la disciplina de auto asignárselas.

"…Se asignaban entonces las tareas, pues el esquema de dejarlo libre eso era de todos y no era de nadie" P8, Desarrollador.

Entrenamiento en prácticas ágiles

¿Cuál es el conocimiento que deben tener los miembros de equipos ágiles? Es una de las preguntas que se hacen las empresas, para responder esto debemos pensar en un entrenamiento que incluya prácticas en TDD, manejo de iteraciones, refactoring, integración continua, etc.

Este entrenamiento se soporta básicamente en leer y participar de las comunidades agiles, aprendiendo de ellas. Durante la adopción se sugiere iniciar con una retrospectiva donde el equipo manifieste los dolores y problemas. Las primeras retrospectivas pueden llegar a ser largas, pues se busca identificar los problemas y proponer cómo resolverlos. De esta manera se realiza una adopción de tipo orgánico, paso a paso y las necesidades de

entrenamiento tanto de tipo técnico como en habilidades blandas serán guiadas por este plan personalizado para la adopción.
Una estrategia que han realizado algunas empresas para convencer al cliente de los resultados de las prácticas ágiles es entrenarlo en dichas prácticas, buscando que hable el mismo lenguaje que el equipo de trabajo en el proyecto.

Seguimiento y métricas en *Scrum*

Se percibe una mala interpretación del control, y no se ve como una herramienta de mejora, más bien se toma como excusa que los equipos son auto gestionados para no hacer uso de métricas y estadísticas que solo buscan ser un radiador de la información al mismo equipo. Se observa que algunos miembros de equipo no tienen disciplina con la actualización de métricas en *Scrum* como las gráficas de *burndown*.

Otro conflicto evidente durante la adopción de *Scrum* es el hecho de que aún se busca tener un control muy estricto de las horas efectivas de trabajo de los empleados para poder hacer el cobro respectivo a los clientes y para eso se ha propuesto en una de las empresas el uso de un tablero de indicadores de productividad neta, por proyecto, individual. Que iría un poco en contravía con la filosofía de *Scrum* sobre sr un radiador de información para el equipo y no para hacer seguimiento y control por parte de un jefe.

Relaciones de confianza

Se evidencia que aún existe desconfianza en los equipos de trabajo y aunque solo existe una entrada en los datos al respecto, este concepto afecta notablemente las interacciones entre individuos y consecuentemente la eficacia de *Scrum*, por esta razón se tuvo en cuenta como contribuidor de esta categoría.
"¿Cómo evitar que el equipo maquille los resultados del velocity sobreestimando o subestimando las tareas?" -P8, Gerente de proyecto.

Retos relacionados con la Colaboración con el cliente

Negociar con el cliente e influenciarlo

Muchas de las preocupaciones de las empresas que inician con la adopción de prácticas ágiles como *Scrum* están relacionadas con la negociación con el cliente y como lograr convencerlo e influenciarlo sobre las ventajas de *Scrum* y engancharlo como pieza clave para el éxito en la adopción.

La necesidad de que la disponibilidad del cliente este como una condición contractual es evidencia de la falta de colaboración con el cliente, como se puede ver en la siguiente frase: *"Si el usuario está disponible, que sea una condición con el cliente (Buena clausula)"*–P6, Gerente de proyecto.

El manejo de la influencia sobre el cliente es otro reto que se presenta durante la adopción de *Scrum*, para lograr tener más influencia, se aconseja tener una buena relación interpersonal entre proveedor y cliente, convencer a personal estratégico en la compañía y que las nuevas metodologías si resuelvan un dolor actual.

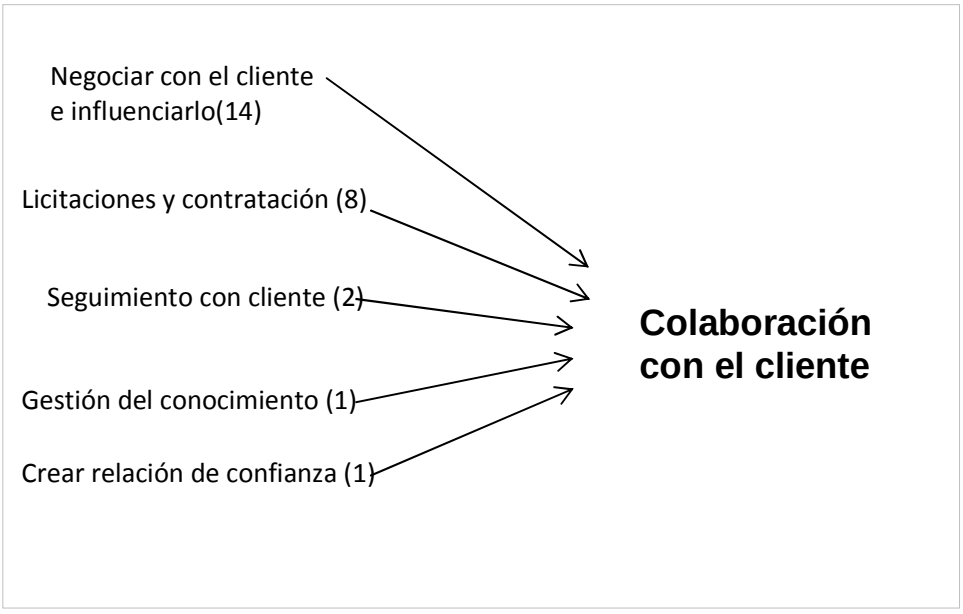

Figura 4.5: Retos asociados a Valores del Manifiesto Ágil – Colaboración con el cliente

Actualmente se percibe por parte de los equipos que los cambios que solicita el cliente son costosos y que una manera de cubrirse la espalda es que los requisitos estén definidos y aprobados en su totalidad, no se tiene conciencia sobre la gestión del cambio como una manera de dar valor agregado al producto para que el cliente esté satisfecho y el negocio sea rentable y más bien se ve como una herramienta para defenderse del cliente, sobre todo con clientes que cambian de parecer frecuentemente.

La negociación basada en el alcance es parte fundamental en *Scrum*, existen varias dudas relacionadas con este tema, por ejemplo, ¿cómo definir que se entrega en cada *release*?, ¿cómo reconocer lo que le proporciona valor?, ¿qué tipo de entregable tendría valor para el cliente?

Se sugiere convencer al cliente con el software funcionando y venderle en vez de reuniones de seguimiento unos espacios para demostración de los incrementos de producto e ir incluyendo actividades de *Scrum* de manera evolutiva. Se recomienda priorizar las funcionalidades de más valor usando *Pareto* y tener en mente que el cliente sabe qué es lo más valioso para su negocio.

Licitaciones y contratación

Existe un tema de fondo relacionado con el cliente y es el tema de contratación, este es uno de los mayores retos al cual se enfrentan las empresas, siendo considerado incluso más difícil de manejar que la misma implantación de *Scrum*, algunos clientes perciben que el proveedor está del lado cómodo y que es el cliente quien asume los riesgos de incumplimiento con las entregas.

Existen dudas sobre cómo manejar los contratos con los clientes para el tema ágil, sobre todo para los proyectos de tiempo y costo fijos que afectan poder negociar con el alcance a ser entregado.

Los proyectos a tiempo y costo fijo ha sido un referente en la manera de contratar con los proveedores de software y este tipo de contrato requiere casi siempre una estimación total

del proyecto, lo cual entra en conflicto con el manejo de la incertidumbre que propone *Scrum* a través de estimaciones que van siendo más precisas a medida que se adquiere más conocimiento.

El manejo de las licitaciones, sobre todo con el sector gobierno, hacen parte de los retos alrededor de las prácticas ágiles:
"¿Cómo hacemos con la presión de la competencia cuando nos dicen que no en una licitación?"–P4, Gerente de proyecto.
"…Existe un contrato marco que se regula con ANS. Somos CMMI nivel 5 y la metodología ágil solo se ha usado para productos a la medida en el área de ingeniería, no ha tenido impacto en la organización. Esta es nuestra primera experiencia en el tema…"–P1, Gerente de proyecto.

Otros conceptos como el seguimiento con el cliente, crear relaciones de confianza y colaboración en la gestión del conocimiento son tratados a continuación:
La transferencia de conocimiento es un valor tanto para el cliente como para el proveedor, las curvas de aprendizaje son necesarias para mejora del equipo en beneficio de ambas partes y las tareas de apoyo para esta transferencia como el análisis de cómo trabaja el equipo o contratar a un experto no debería ser asumido solo por el proveedor sin embargo, la mayoría de las veces es este quien termina asumiendo los esfuerzos de la transferencia de conocimiento de negocio. Para evitar esto es necesario fortalecer la colaboración y confianza del cliente a través de convencerlo, hablar el mismo idioma, darle visibilidad, mostrarle beneficios y casos de éxito sobre las practicas adoptadas.

Retos relacionados con el Software funcionando

La categoría de Software Funcionando emerge de conceptos como release y modelos en *Scrum*, el primero de estos conceptos se refiere a los retos relacionados con las entregas: Estrategias para las liberaciones, valor de negocio y problemas de integración al liberar. El segundo concepto de modelos tiene en cuenta las dudas y retos sobre la manera de modelar

en *Scrum*. ¿Qué tipos de modelos se usan en ágil? ¿Cómo se mezclan con las metodologías actuales, cómo se define la arquitectura?

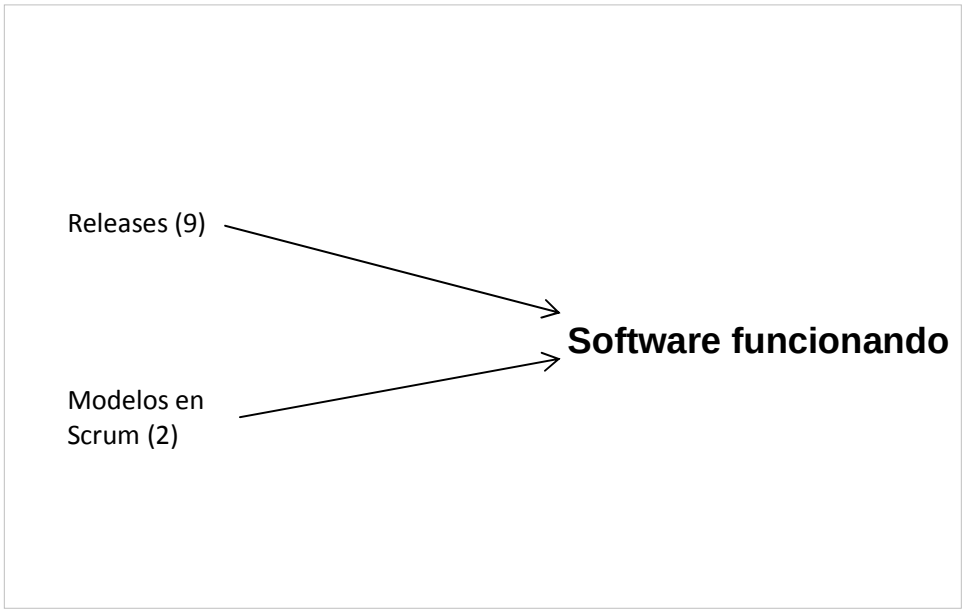

Figura 4.6: Retos asociados a Valores del Manifiesto Ágil – Software funcionando.

Retomando el tema de las liberaciones se evidencian dudas sobre cómo planear las liberaciones y las estrategias para identificar las liberaciones con mayor valor para el cliente.

Al momento de la implantación el reto es cómo lidiar con problemas de integración, paso a ambientes de producción y cambios sobre temas aun no implantados en producción.
El concepto de modelos en *Scrum* se relaciona con dudas sobre como modelar en *Scrum*. Por ejemplo, hasta que nivel de detalle es necesario modelar los diagramas de UML.

Retos relacionados con Respuesta al cambio

Esta última categoría asociada a los valores ágiles emerge de conceptos como Soporte y mantenimiento con prácticas ágiles y cuales etapas del desarrollo de software se recomiendan para implantar *Scrum*.

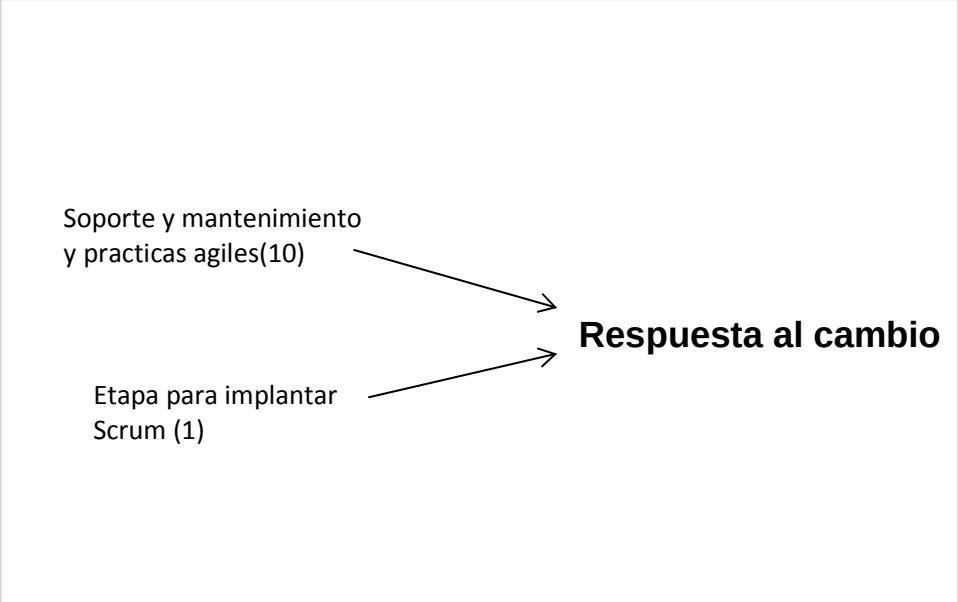

Figura 4.7: Retos asociados a Valores del Manifiesto Ágil – Respuesta al cambio.

Ambos conceptos están basados en una respuesta al cambio de manera oportuna, por un lado las empresas están en la búsqueda de mejores maneras de responder al cambio intrínseco en las solicitudes de soporte y mantenimiento y por otro lado la respuesta al cambio es necesaria cuando se desean implantar prácticas nuevas en los procesos actuales de las empresas.

Decidir entre *Scrum* y *Kanban* o una combinación de estas prácticas es uno de los retos de las empresas en busca de la mejora en la eficiencia y productividad del servicio de soporte y mantenimiento.

Para el tema de soporte se sugiere no manejar sprints sino flujos continuos tipo Kanban, que el equipo esté dedicado y no le sean asignadas tareas de otros proyectos o manejar un 70% de disponibilidad, si no se puede evitar el multitasking. Algunas actividades del flujo

Scrum se pueden realizar, por ejemplo las reuniones de planeación al inicio de la semana de las solicitudes que lo permitan, se pueden hacer las reuniones diarias con sus 3 preguntas como un canal de comunicación para informar al equipo y lo ideal es hacerlo frente al tablero, la reunión de revisión donde se muestra la implementación de los cambios y al final la reunión retrospectiva para revisar el proceso, como está trabajando el equipo.

4.2. Análisis de resultados posterior a la observación

La investigación fue conducida usando una encuesta anónima, dirigida a gerentes de proyecto, desarrolladores, personal de calidad y de gestión humana que tuvieran por lo menos 1 mes de estar trabajando en empresas desarrolladoras de software que hubieran ejecutado algún programa de mejora de procesos con CMMI y actualmente estuvieran adoptando prácticas ágiles como *Scrum*.

Los datos se recolectaron entre los meses de diciembre de 2012 y junio del año 2013, se enviaron 84invitaciones por correo para participar en la encuesta, se invitó a diferentes roles dentro de las 8 empresas participantes del estudio, y se recibió respuesta de 41participantes, de los cuales 4 fueron invalidas por ser de empresas fuera de la muestra, 2 de las personas no respondieron la encuesta de manera completa y las otras 43personas no respondieron la encuesta, (a pesar de que se ofreció el incentivo de participar en la rifa de un bono para la compra de un libro), para una tasa de respuesta del 41,6% sobre el total de la población seleccionada para la encuesta.

4.2.1. Respuestas a las preguntas de investigación de la encuesta

En esta sección reportaremos los hallazgos sobre datos demográficos y percepciones sobre la adopción de prácticas ágiles en las empresas.

Información general del encuestado y el proyecto

RQ1. ¿A cuál empresa pertenece Ud.?

En la figura 4.8 se presenta la distribución de las encuestas respondidas respecto a la población total de empresas seleccionadas para el estudio; se puede observar que las encuestas respondidas forman una muestra equilibrada de la población.

Se evidencia que el 87% de las empresas participantes en el estudio tuvieron al menos un representante que respondió la encuesta.

Según los resultados presentados en la figura 4.8, se encuentra que la mayoría de los participantes que respondieron la encuesta pertenecen a las empresas de E5, E8, E3 y E1. Dentro de estas 4 empresas, las 2 primeras son nivel 5 de CMMI, 1 de ellas es nivel 4 y otra es nivel 2.

Las figuras 4.9, 4.10 y 4.11 muestran el perfil de las empresas que participaron en el estudio de acuerdo al número de empleados, distribución por tamaño y la distribución según la adopción de prácticas de CMMI.

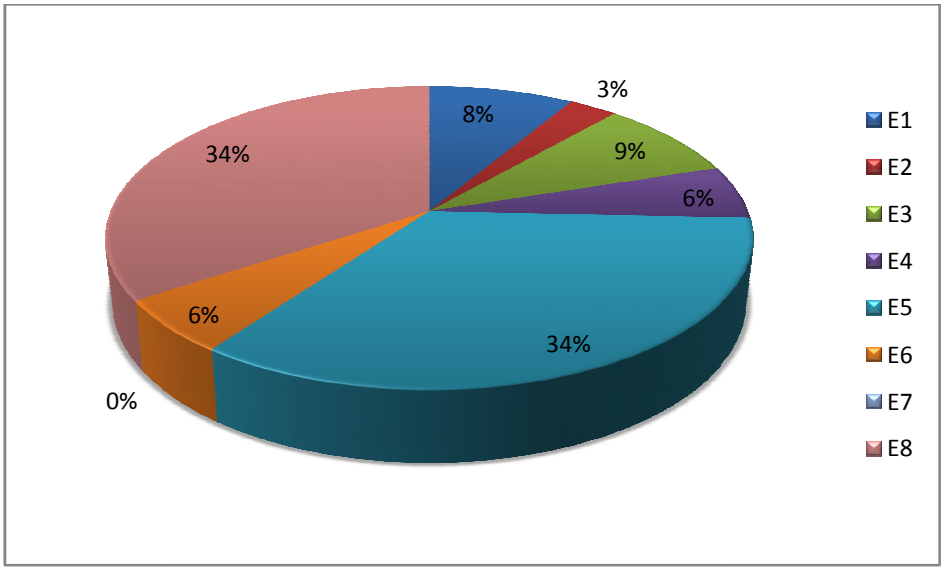

Figura 4.8: Distribución de participantes según empresas.

La distribución de las empresas según valoración del SEI presenta la información de las empresas que han seguido procedimientos formales de evaluación de procesos con CMMI, sin embargo el otro 50% de las empresas del estudio aunque no se han evaluado formalmente también han adoptado practicas orientadas a planes basadas en el modelo CMMI, TSP/PSP o han sido certificadas con ISO basando su proceso en prácticas tradicionales.

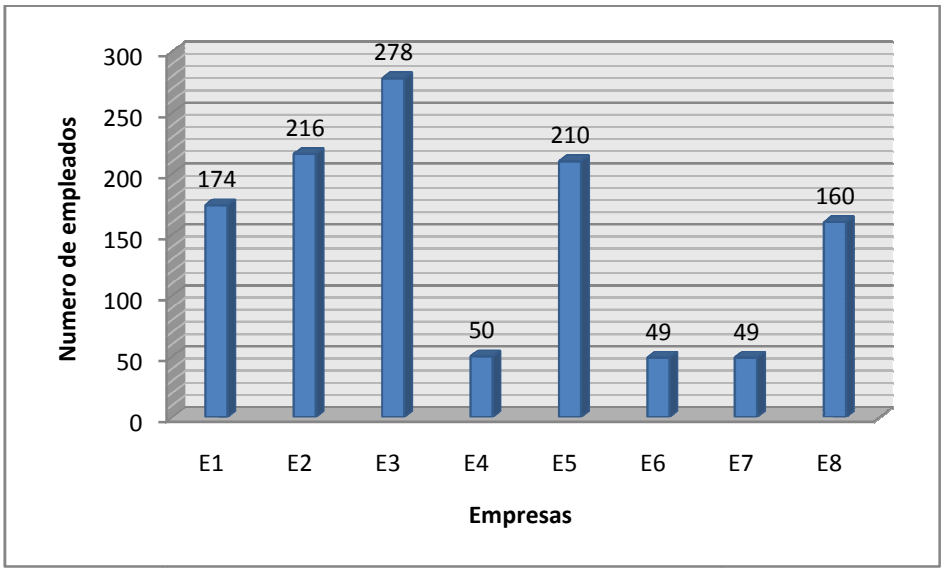

Figura 4.9: Distribución de empresas por número de empleados.

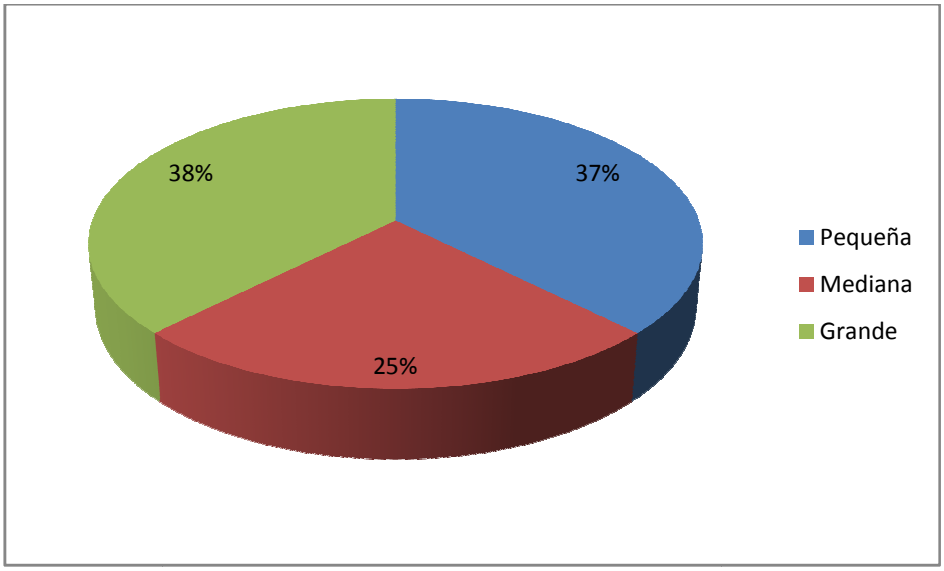

Figura 4.10: Distribución de empresas según tamaño.

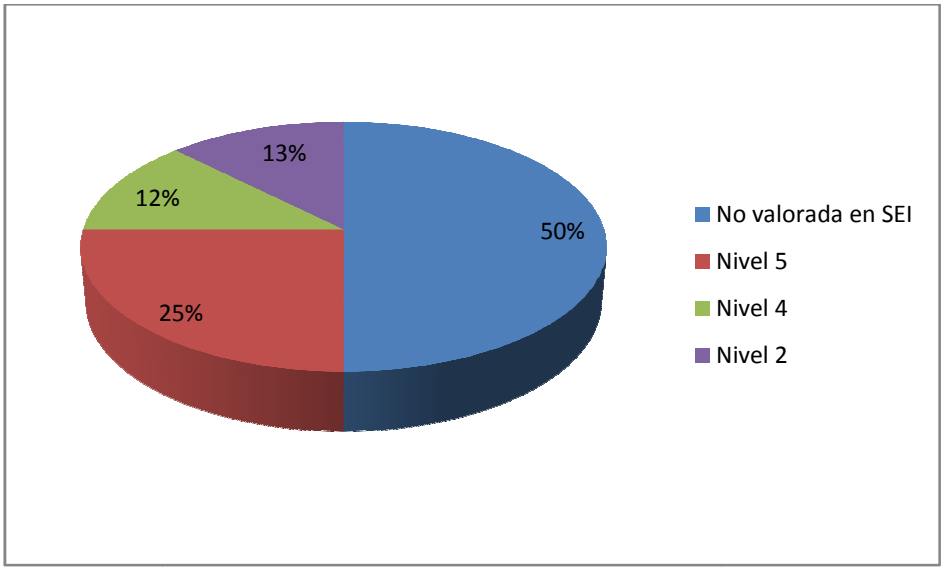

Figura 4.11: Distribución de empresas según valoración CMMI.

RQ2. ¿Cuántos años de experiencia tiene Ud. en el desarrollo de software?

Los participantes que respondieron tienen un promedio de 8.79 años de experiencias en el desarrollo de software (La desviación estándar fue de 5.24; lo mínimo fue de 1 año y el máximo fue 25 años de experiencia).

RQ3. ¿Cuál es el cargo o rol que Ud. desempeña en su empresa? Por favor escoja todas las que apliquen.

De todos los participantes que respondieron la encuesta 40% eran desarrolladores, 20% eran gerentes de proyectos, 17% eran arquitectos, 8.8% eran analistas de calidad o testers y 22.2% jugaban otros roles en el desarrollo, entre ellos:

Coordinador área de requisitos

Gerente de procesos y tecnología

Scrum Master/ Analista funcional

Consultor, Analista

Consultora en Experiencia de Usuario

Ingeniero de Metodología y Productividad

Analista de Integración

Líder de procesos

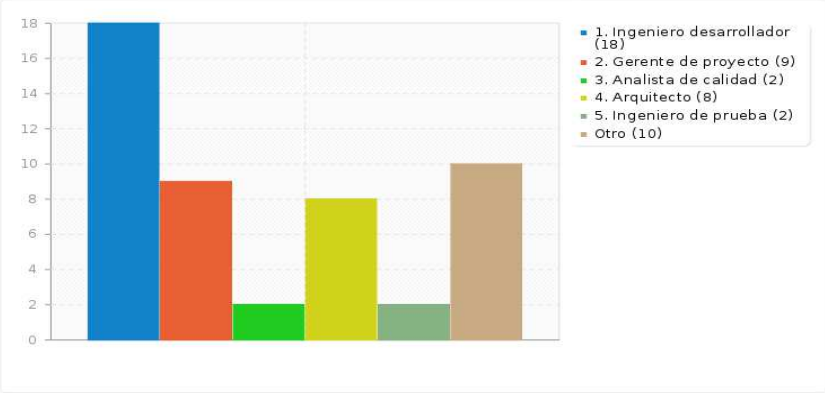

Figura 4.12: Distribución de participantes según roles en el desarrollo de software.

RQ4. ¿Actualmente forma parte de un equipo de trabajo *Scrum*? ¿Desde hace cuánto tiempo?

Alrededor del 49% de los participantes formaban parte de un equipo de *Scrum* al momento de responder la encuesta, el resto no pertenecían a un equipo de trabajo *Scrum* (46%) o no respondieron la pregunta (5%).

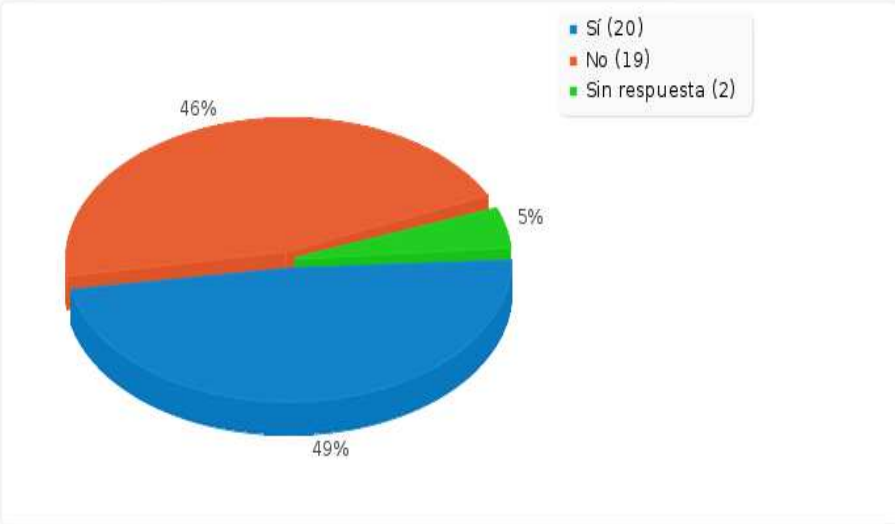

Figura 4.13: Porcentaje de participantes en equipos *Scrum*.

Cuando se les preguntó a las personas que formaban parte de un equipo *Scrum* desde hace cuánto tiempo lo hacían, el promedio del tiempo fue de 7.9 meses, el mínimo tiempo que han estado trabajado fue de 1.5 meses y el máximo tiempo que alguien ha estado trabajando con *Scrum* es de 36 meses.

RQ5. ¿Ha trabajado en equipos *Scrum* anteriormente?

Según los resultados presentados en la figura 4.14, se encuentra que 22% de las personas usaron *Scrum* en sus anteriores equipos de trabajo. Al comparar con el 49% de los participantes que actualmente usan *Scrum* al momento de responder la encuesta, se

evidencia que las organizaciones han incrementado en un 27% el número de personas que trabajan en equipos que usan este marco de trabajo.

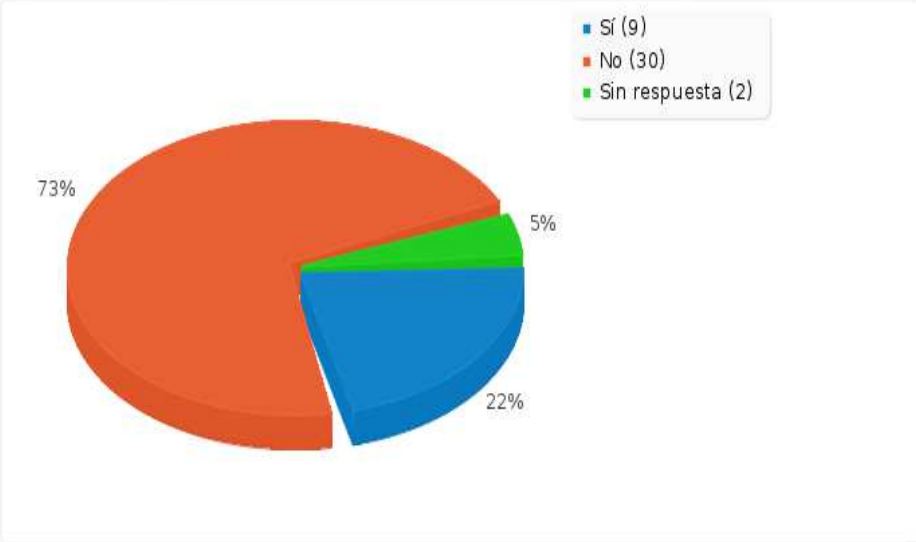

Figura 4.14: Porcentaje de encuestados que han estado en equipos *Scrum*.

RQ6. ¿Indique en meses cuánto tiempo lleva su equipo usando *Scrum* en el proyecto actual?

Los participantes que respondieron tienen un promedio de 2.8 meses usando *Scrum* en el proyecto actual (La desviación estándar fue de 3.82); lo mínimo fue de 0 meses y el máximo fue 16 meses.

RQ7. ¿Está trabajando actualmente en la primera versión del producto o en mantenimiento?

El porcentaje de personas que estaban trabajando en la primera versión de un producto fue del 44%, un 37% de las personas estaban realizando mantenimiento del producto y un 6% ambos. Resalta el hecho de que la mayoría de los proyectos están desarrollando la primera versión del producto. Esto se podría explicar porque se trata de equipos de desarrollo

adoptando un nuevo método o practica y por lo tanto se seleccionó como proyecto piloto un nuevo desarrollo.

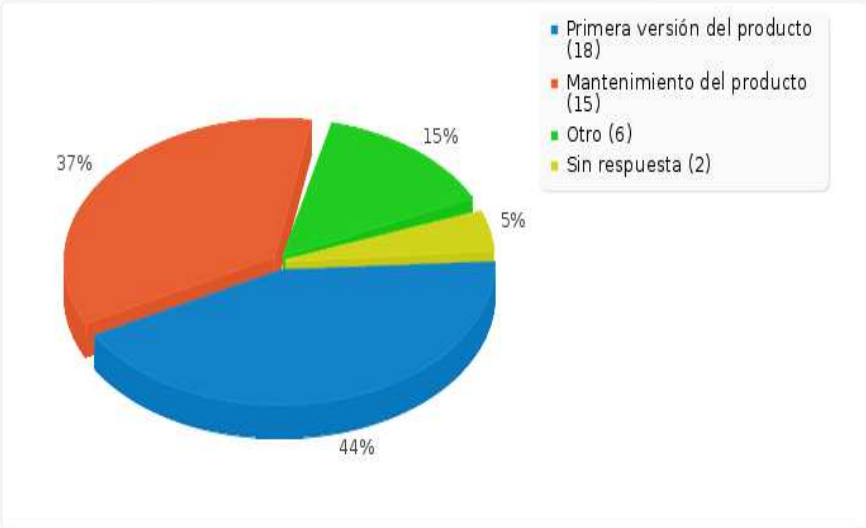

Figura 4.15: Distribución del desarrollo según fases del ciclo de vida de sistemas.

Adopción actual de *Scrum* y prácticas complementarias

RQ8. De las siguientes prácticas de *Scrum* indique el nivel de adopción que hasta el momento percibe en su organización.

Practicas ágiles	**Descripción**
Entrega incremental	Cada producto resultante es usable y cada versión construida sobre la versión anterior adiciona funcionalidad visible al usuario.
Timebox	Manejo de duración fija durante tareas del desarrollo. Es un periodo de tiempo acordado previamente durante el cual se trabaja firmemente hasta la completitud de una meta. Se suspende el trabajo

	cuando el tiempo límite es alcanzado y se evalúa lo que se ha cumplido.
Iteraciones	Es un timebox durante el cual el desarrollo toma lugar. Se relaciona con el manejo de sprints.
Daily Stand-up Meetings	Reuniones diarias con todo el equipo: cada miembro describe brevemente cualquier contribución completada y cualquier obstáculo que encuentre. Usualmente se usan las tres preguntas y se realiza en frente del taskboard.
Burndown Chart	Gráfico de avance relacionado con la cantidad de trabajo restante y el tiempo transcurrido desde el inicio del proyecto o sprint.
TaskBoard	Tablero de tareas con los estados básicos: Por hacer, En progreso y Hecho. Es un "radiador de información" al equipo.
Definition of Done	Definición de Hecho. El equipo está de acuerdo con los criterios que deben ser cumplidos antes que un incremento de producto es considerado hecho.
Definition of Ready	Definición de Listo. El equipo hace visible y explicito que las historias de usuario cumplen los criterios para ser aceptadas en una iteración.
Reunión de planeación	Ceremonia de *Scrum* que se realiza al inicio de un sprint para definir que se hará en el sprint y como se realizará.
Estimación por puntos	Estimación expresada en unidades diferentes a horas-hombre. Usualmente las unidades son los puntos de historia.
Estimación relativa	Estimar tareas o historias de usuario, no de manera separada o en horas, sino comparándolas o

	agrupándolas en ítems de dificultad equivalente.
PlanningPoker	Método de estimación basado en Wideband Delphi en donde se estima basado en la analogía, el juicio experto y la desagregación.
Backlog	Pila de producto que contiene las características priorizadas del producto a ser desarrollado.
Backlog grooming	Se relaciona con la limpieza del backlog, removiendo las historias de usuario no relevantes, creando nuevas historias por las nuevas necesidades, revalorando las prioridades, asignando estimaciones a las historias, dividiendo historias en otras más pequeñas.
Lista de impedimentos	Listado de los obstáculos que surgen durante el desarrollo y que el *Scrum Master* debe ayudar a remover en conjunto con el equipo.
Reunión Retrospectiva	Ceremonia de *Scrum* que permite la mejora del proceso, se realiza después de cada sprint y genera planes de acción a ejecutar en el próximo sprint.

Tabla 1: Prácticas ágiles asociadas a Scrum [45].

Para realizar este análisis se seleccionaron 16 prácticas ágiles, con base en [45], las cuales se muestran en la tabla 4.1 Las preguntas acerca del nivel de adopción se formularon de manera cualitativa y la respuesta podía tomar uno de los siguientes valores:

1=Alto
2=Medio
3=Bajo
4=No se usa
5=Se planea usar
6=No requerida

7=No sé
Sin respuesta

Para propósitos prácticos, las respuestas 4, 5 y 6 se agruparon en una sola categoría denominada “No se usa”.
La figura 4.16, muestra algunas prácticas asociadas a *Scrum* y su nivel de adopción dentro de las empresas.

De esta figura se resalta que las practicas con mayor nivel de adopción, combinando los niveles de adopción alto y medio, en las empresas son:
Reunión de Planeación con un 83,33% de adopción en las empresas, seguido por la práctica de Entrega Incremental con un 72,22% de adopción y luego las técnicas: Iteraciones 66,6% y TaskBoard 66,6% tienen igualmente un nivel de adopción alto comparadas con otras técnicas.

Otro aspecto a resaltar son las técnicas *Scrum* que tienen un bajo nivel de adopción en las empresas, entre ellas:
TimeBoxing 33,3%, las asociadas a métricas como el BurnDown Chart con un 30,56% y el uso del listado de impedimentos con un 27,78%.

Dentro de las prácticas no usadas están la Definición de Listo con un 25% y la estimación por puntos con un 16,67%.

Las técnicas de Backlog grooming 22,2%, PlanningPoker y Estimación relativa ambas con un 19,44% no eran conocidas por las personas encuestadas.

Análisis adicional:
Si se comparan estos resultados con las categorías emergentes y conceptos de Grounded Theory podemos darnos cuenta que coinciden en la práctica de reunión de planeación. En

la primera parte del estudio surgieron varias preguntas asociadas a esta práctica, por ejemplo, ¿Cuáles roles son necesarios en las reuniones de planeación por parte del cliente: personal de tecnología, personal del área financiera?, ¿Cuáles roles se recomienda deberían asistir a la reunión de planeación por parte del proveedor: técnicos, área de comercial? ¿Qué tipo de técnicas usar para lograr una adecuada priorización del backlog?, ¿cómo identificar lo que tiene valor para el cliente?

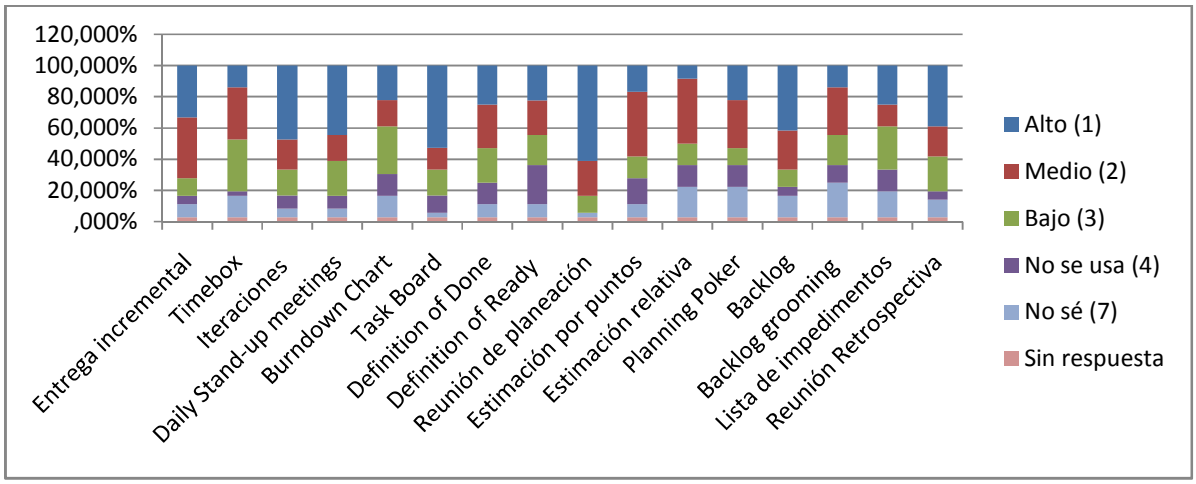

Figura 4.16: Nivel de adopción de prácticas asociadas a *Scrum*.

RQ9. De las siguientes prácticas ágiles complementarias a *Scrum* indique el nivel de adopción de la práctica en su organización.

Para analizar el nivel de adopción de las prácticas ágiles complementarias en las empresas, los participantes escogieron una de las siguientes opciones:

1=Alto

2=Medio

3=Bajo

4=No se usa

5=Se planea usar

6=No requerida

7=No sé

Sin respuesta

La figura 4.17, muestra los resultados encontrados para esta pregunta. Donde se resalta que las practicas complementarias con un nivel alto de adopción son las siguientes:

Control de versiones. Con una frecuencia de uso en el 77,78% de las empresas.
Interacción directa con el cliente con una frecuencia de uso en el 61,11% de las empresas.
Integración continúa de código con una frecuencia de uso en el 47,22% de las empresas.

Según los resultados del estudio, se encuentra también que las prácticas complementarias con menor frecuencia de uso son las siguientes:
XP (Programación Extrema). Esta práctica no está siendo utilizada por un 36,11% de las empresas.
TDD (Desarrollo guiado por pruebas).Esta práctica no está siendo utilizada, por 25% de las empresas.
Programación en pares. Esta práctica no está siendo utilizada, por un 22,22% de las empresas. Asimismo esta práctica obtuvo el mayor porcentaje para la categoría No requerida, un 5,56%.

Análisis adicional: Cabe resaltar que a pesar de que algunas investigaciones muestran que la combinación de *Scrum* y XP, incluyendo prácticas como TDD y Pair Programming, resulta en equipos ágiles más efectivos [46]; se puede observar que en las empresas involucradas en el estudio, son estas las practicas ágiles de ingeniería menos usadas.
Este resultado coincide con los conceptos y categorías identificados a través de GT en la primera parte del estudio, donde se identificaron como retos dentro de la categoría de prácticas complementarias los conceptos de TDD, Pair Programming y XP.
El bajo nivel de uso percibido en la práctica de programación en pares se asocia al poco conocimiento sobre la mejora de la productividad o calidad relacionada con su uso además del grado de dificultad para aprenderla. Además no se percibe la ventaja de realizar esta práctica pues algunas veces al realizar el trabajo en pares uno de los desarrolladores no

participa de forma activa y más bien adopta una posición relajada y que no aporta conocimiento. En cuanto a TDD, quizá la implantación es difícil por la alta curva de aprendizaje que es necesaria para usar adecuadamente TDD, adicionalmente las prácticas de inspección, elemento clave en las practicas ágiles, de alguna manera son realizadas a través de prácticas de verificación y validación muy maduras en los enfoques tradicionales.

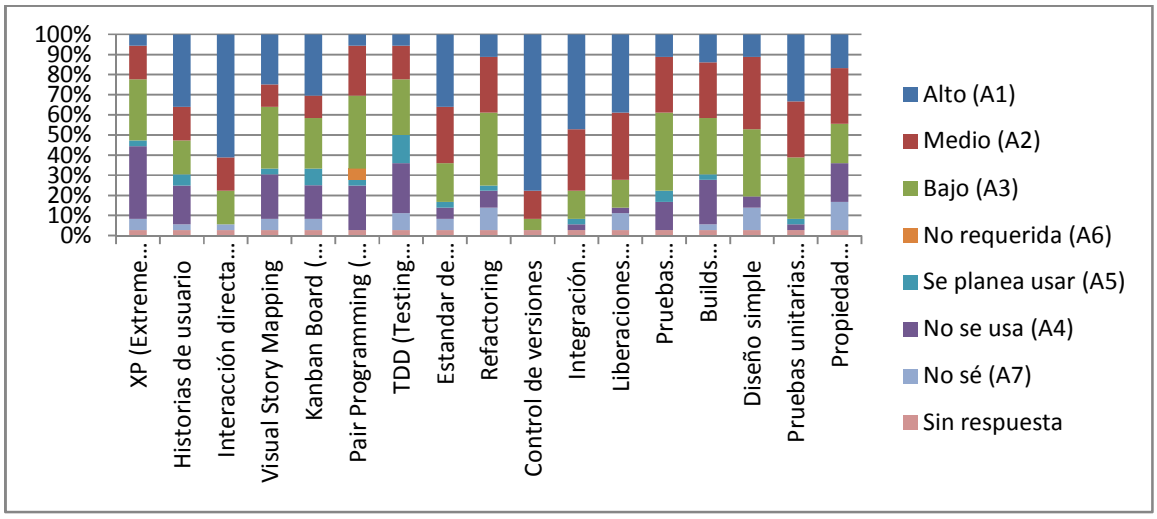

Figura 4.17: Nivel de adopción de prácticas ágiles complementarias.

RQ10. ¿Cuáles son las principales dificultades que Usted ha observado en la adopción de *Scrum* dentro de su organización, desde el punto de vista organizacional, técnico, humano y de integración con prácticas existentes como CMMI, PSP/TSP?(Identifique mínimo 3 dificultades)

Esta pregunta abierta recopila las percepciones de los encuestados sobre los retos durante la adopción de *Scrum* en las empresas. Se encontraron las siguientes dificultades, las cuales fueron clasificadas en las categorías que abajo se detallan:

INTEGRACIÓN CON OTRAS PRÁCTICAS ACTUALES DE LAS ORGANIZACIONES

- Se percibe que la definición del proceso *Scrum* y la adaptación de las prácticas actuales es una tarea que lleva tiempo.
- En algunas empresas aunque indirectamente se implementa *Scrum*, dependiendo del escenario, se hace de forma muy esporádica y no está formalmente documentada.
- La dificultad radica cuando se deben seguir los procesos definidos, trabajar con los indicadores definidos y usar los artefactos definidos organizacionalmente y al tratar de implantar un framework como *Scrum*, se percibe resistencia en una organización de este tipo.
- La integración con CMMI nivel 5 es difícil pues CMMI a este nivel exige documentación innecesaria, que va en contradicción con los fundamentos agilistas. Adicionalmente la integración con CMMI exige dejar registros y documentación más extensiva de las prácticas de ingeniería y gestión y elaborar planes a largo plazo lo cual es difícil con *Scrum*. Aún no está muy bien definido el proceso a seguir con *Scrum*, es decir, para el proceso normal se tiene CMMI pero para *Scrum* no es muy claro que estándar se debe seguir.
- La integración entre procesos PSP y TSP con el framework *Scrum* es un reto pues existen reprocesos al registrar horas y reuniones.
- Luego de una certificación, implementando metodologías PSP / TSP, la organización se vuelve un tanto reacia a la implementación de nuevas metodologías esperando que las recién implantadas maduren en conjunto con los procesos.
- Se encontró el reto de aplicar *Scrum* a proyectos de soporte y mantenimiento de software, pues se tiene una dinámica frecuente en los cambios de prioridades. Cuesta mucho adoptar estos nuevos enfoques, aunque los resultados a la fecha son positivos.
- Aún no está muy bien definido el proceso a seguir con *Scrum*, es decir, para el proceso normal se tiene CMMI pero para *Scrum* no es muy claro que estándar se debe seguir.

CONFORMACIÓN DE EQUIPOS AUTO ORGANIZADOS

- Se percibe retos asociados a la conformación de un equipo especializado de personas con capacidad de autogestión y auto organización, pues es difícil conseguir personas con las siguientes habilidades:
    - Disciplina.
    - Auto-organización y auto gestión
    - Iniciativa
    - Confianza
    - Capacidad de cambio
    - Interés en adquirir nuevas experiencias
    - Madurez
    - Colaboración

- Otro aspecto importante es que no es fácil constituir equipos de trabajo de más de tres personas con los tipos de proyectos que se deben desarrollar.

CAMBIO CULTURAL EN LA ORGANIZACIÓN

- El lento cambio en la cultura organizacional ha dificultado la adopción total de prácticas ágiles en el tiempo estipulado. Aun se percibe resistencia al cambio. Por ejemplo no hay disciplina con las reuniones diarias. Ha sido difícil la interacción directa del usuario con el equipo de trabajo y es necesario elevar el nivel de comunicación entre las personas. Aún existe la costumbre de usar el ciclo de vida en cascada.

FALTA DE ENTRENAMIENTO Y COACHING

- Se percibe la falta de coaching con experiencia desde el inicio de la adopción. También se percibe la falta de capacitación previa en *Scrum* para el equipo, lo cual no permite aumentar el bajo conocimiento actual sobre el framework y la forma de integrarlo a las necesidades de la empresa.

- La falta de conocimiento por no tener adecuado entrenamiento conlleva a mantener la concepción errada que se tiene sobre los sprints y sobre el seguimiento a realizar en ellos.

COLABORACIÓN CON EL CLIENTE Y CONTRATACIÓN

- Los encuestados perciben que es difícil lograr que el cliente entienda y se adapte a estas nuevas prácticas ágiles, se resalta la falta de conocimiento del framework por parte del cliente y poca claridad del cliente de cuando usar *Scrum* y que esta selección depende del contexto del proyecto.
- Otro reto son los esquemas de contratación tradicionales que algunos clientes desean mantener (costo, tiempo y alcance fijos).
- Se percibe un nivel alto de burocracia en las organizaciones que hace difícil la adopción de las prácticas ágiles.

MANEJO DE ROLES EN *SCRUM*

- No existe una clara diferencia entre Gerente de proyecto y su nuevo rol como *Scrum Master*. Se percibe a algunos gerentes de proyectos sin compromiso y renuentes a la implementación de *Scrum*.
- Es difícil tener un product owner comprometido y con las habilidades necesarias para hacer sus tareas.
- No se tienen claras cuáles son las expectativas de lo que debe ser un *Scrum Master*.
- Algunas empresas optaron por eliminar roles como el equipo de calidad pues se tiene la percepción de que la calidad es responsabilidad de todo el equipo y que no es necesario este rol.

DIFICULTAD CON PRACTICAS ÁGILES COMPLEMENTARIAS

- Se percibe dificultad con la implantación de la práctica de programación en pares, la estimación por puntos de historia y la estimación de tareas, así como con las pruebas automatizadas. No se encuentran ejemplos concretos sobre estimación por puntos.

- Se percibe dificultad con el uso del tablero Kanban.
- Es difícil seguir el uso de los procesos técnicos a pesar de estar bajo *Scrum*.

GESTIÓN DEL CONOCIMIENTO

- Es difícil el manejo de la trazabilidad con los requisitos con *Scrum.*
- Ha sido un reto encontrar un balance con la documentación.
- No es claro cómo manejar las evidencias de decisiones.

TIEMPO DE RESPUESTA AL MERCADO

- La adopción de nuevos procesos en muchas ocasiones hacen que los tiempos de respuesta a los clientes se desaceleren, lo que dificulta dar respuestas rápidas que son en la mayoría de los casos las necesidades que se plantean en los inicios de los proyectos.
- La continuidad del proceso.
- Cumplimiento con los cronogramas mientras se adoptaba la metodología.

RQ11. ¿Cuáles son los principales logros o beneficios que Usted ha observado hasta el momento por la adopción de *Scrum* dentro de su organización, desde el punto de vista organizacional, técnico, humano y de integración con prácticas existentes como CMMI, PSP/TSP? (Identifique mínimo 3 logros)

Esta pregunta abierta recopila las percepciones de los encuestados sobre los logros o beneficios durante la adopción de *Scrum* en las empresas, categorizados desde el punto de vista organizacional, técnico y humano:

- **Organizacional**
    - Dentro de los beneficios organizacionales por la adopción *Scrum*, resaltan los asociados a la entrega continua de valor para el cliente, con

estas prácticas ágiles el cliente recibe incrementos de producto en menos tiempo y no se debe esperar al final del proyecto para ver los resultados.

- Se aprecia una mayor interacción entre el cliente y el equipo del proyecto, hay una mejora en la comunicación con el cliente pues este debe involucrarse más y también se percibe un aumento en la transparencia entre el equipo y el cliente.
- Los equipos están enfocados a la mejora propia de forma natural y no inducida lo cual permite la inclusión continua de nuevas prácticas y métodos en el proceso tradicional de desarrollo de software.
- A nivel de organización ha permitido una gestión de proyectos más simple y más humana donde el trabajo en equipo se valora más y el crecimiento y conocimiento de las personas se incrementa de manera exponencial.
- La apropiación y apertura del conocimiento de los talentos que están trabajando con esta metodología.
- Se disminuye la resistencia al control de cambios en el producto.
- Se ha reducido en gran medida el sobreesfuerzo.
- Se ha aumentado la productividad del equipo.
- Se han reducido los costos.
- Es más fácil la eliminación de recursos o personas críticas para el éxito del proyecto.
- Se manejan esquemas de contratación más equitativos para las partes.
- Se ha eliminado la micro-administración de los equipos.
- Con la adopción de *Scrum* se está a la vanguardia en nuevas metodologías.

- **Técnico**
  - Hay una mejor definición de las necesidades del usuario.
  - Se evidencia una planeación más ordenada y participativa, y una mejor visibilidad de las actividades y sus responsables.

- Se tiene más enfoque del equipo en las tareas asignadas, con una mayor visibilidad de compromisos y bloqueos.
- Se ha aumentado la agilidad para solucionar los problemas y se ha aprendido a solucionar escenarios complicados en corto tiempo.
- Los logros más significativos están enfocados hacia el control y seguimiento de las actividades y a los objetivos del proyecto.
- Hay un control real sobre el proyecto, cada persona actualiza sus tareas realizadas lo que ha permitido indicadores son más precisos.
- El seguimiento constante de las actividades hace que se puedan tomar acciones correctivas de forma rápida para encausar los problemas.
- Las entregas frecuentes y revisiones del producto por iteraciones hace que se puedan minimizar los riesgos de una entrega completa.
- La retroalimentación temprana por parte del usuario la hace más efectiva, permitiendo que se encuentren más rápido las dificultades y se puedan hacer ajustes a medida que se va avanzando.
- La entrega de valor rápido y frecuente al cliente a través de software funcional permite tener a los usuarios utilizando versiones parciales del producto.
- Mejora de la calidad del producto. Evitando el reproceso.

- **Humano**
  - Dentro de los beneficios más recurrentes que mencionan los encuestados está el aumento en la satisfacción del cliente con las entregas tempranas y continuas y un incremento de la confianza entre cliente y equipo de desarrollo.
  - Otro beneficio identificado es una mejora en la dinámica de trabajo en equipo, se percibe mejora en la coordinación, colaboración, cohesión y sinergia entre miembros del equipo.

- Los equipos se muestran más motivados y sincronizados y se percibe mayor entusiasmo y alegría de las personas del equipo.
- Se percibe disciplina y mejor comunicación en el equipo de trabajo.
- Empoderamiento de las personas para resolver los impedimentos y la orientación al logro han cumplido un papel bastante importante en los proyectos que han tenido éxito con esta metodología.
- Oportunidad del equipo para aprender nuevas habilidades y destrezas y proponer nuevas ideas.
- Aplicación de "Conocimiento Neto" con el toque de experiencia de cada integrante que conforma el equipo.

- **Integración con prácticas existentes como CMMI, PSP/TSP.**

  - Fácil adaptación a los modelos TSP/PSP que se aplican dentro de la compañía

RQ12. ¿Cuáles son las lecciones aprendidas que Ud. ha identificado hasta el momento respecto a la apropiación de *Scrum* en su organización? (Identifique mínimo 3 lecciones aprendidas).

De la anterior pregunta se obtuvieron los siguientes resultados alrededor del factor humano:

- Los participantes consideran que se debe tomar lo mejor de cada práctica para crear un proceso adecuado a las necesidades de la organización y no solo implantar el framework *Scrum* como la bala de plata que solucionará todo.
- La adopción de *Scrum* exige la capacitación de las personas en todos los niveles.
- Se recomienda hacer la adopción de manera incremental con equipos empoderados y comprometidos buscando que los miembros del equipo no sean todos junior y que el equipo trabaje en el mismo sitio.

- Involucrar solo parte de la organización alrededor de un proyecto *Scrum* (Crear islas de proyectos).
- Es muy importante aclarar las expectativas que se tiene de un equipo auto-organizado y del rol de *Scrum Master*.
- Es necesario mantener una comunicación clara, permanente y efectiva como factor clave para el éxito del proyecto.
- Otra lección aprendida es asesorarse de una empresa con experiencia en *Scrum* para que realice coaching a los equipos de trabajo. Igualmente el *Scrum Master* necesita estudiar técnicas de coaching si quiere hacer bien su labor.
- Es importante sensibilizar al Product Owner externo sobre *Scrum* y sus ventajas, mostrando resultados.
- Involucrar y capacitar al cliente desde el principio y todo el proyecto para que tome conciencia del papel que juega dentro del equipo.
- El dueño del producto debe ser siempre alguien del cliente (involucramiento del usuario).
- La disponibilidad y comunicación continua entre el equipo y el *Product Owner* es un factor de éxito primordial, un *Product Owner* debe estar disponible para el proyecto mínimo un 45% del tiempo y el ideal un 80% o más.
- Para que un proveedor de desarrollo de software implemente adecuadamente metodologías ágiles debe contar con un marco contractual adecuado con los clientes.
- Con relación a las prácticas de ingeniería, en lo posible se debe complementar *Scrum* con prácticas técnicas de desarrollo de software, como las propuestas por XP.
- Siempre se deben realizar las revisiones pares para evitar errores en el código.
- Se debe realizar un control de versiones constante.

Como parte de las lecciones aprendidas se identifican algunas relacionadas con los elementos de *Scrum*:

- Es importante trabajar en conjunto desde la planeación.

- Definir metas alcanzables en cada sprint y que agreguen valor real de negocio permite que los proyectos avancen y los indicadores de satisfacción del cliente sean buenos.
- Cada iteración debe estar aprobada por el usuario para evitar reprocesos costosos.
- Si se desconoce una tarea es mejor no estimarla o no ser optimista en la estimación.
- Se recomienda que el "Sprint 0" entregue software funcionando.
- La definición de hecho es un instrumento valioso a la hora de entregar el producto.
- La reunión retrospectiva al final de cada entrega crea lazos de confianza.
- Seguir y cumplir con lo pactado en el sprint y no extender las fechas de entrega.
- No personalizar *Scrum* a los requerimientos de la organización.
- Documentar las actividades permite una trazabilidad adecuada de errores.
- Se recomienda para los proyectos de soporte y mantenimiento de aplicaciones la combinación de *Scrum* y *Kanban*.

Percepciones generales

RQ13. Indique como ve reflejado a su equipo durante la adopción de *Scrum*.

De la siguiente figura se observa que un 29,4% de participantes está muy de acuerdo con que al equipo le ha gustado trabajar con *Scrum*. Asimismo se resalta que un 26,47% de los participantes está en desacuerdo con que el *Scrum Master* aun asigna las tareas al equipo. Una conclusión que se podría sacar es que los participantes sienten que sus equipos están más comprometidos ahora con los resultados del proyecto que antes de *Scrum*, quizá por el peso que esta práctica le da a mostrar la importancia de cada funcionalidad, quien se beneficia y como, basados en una visión de negocio que guía al equipo.

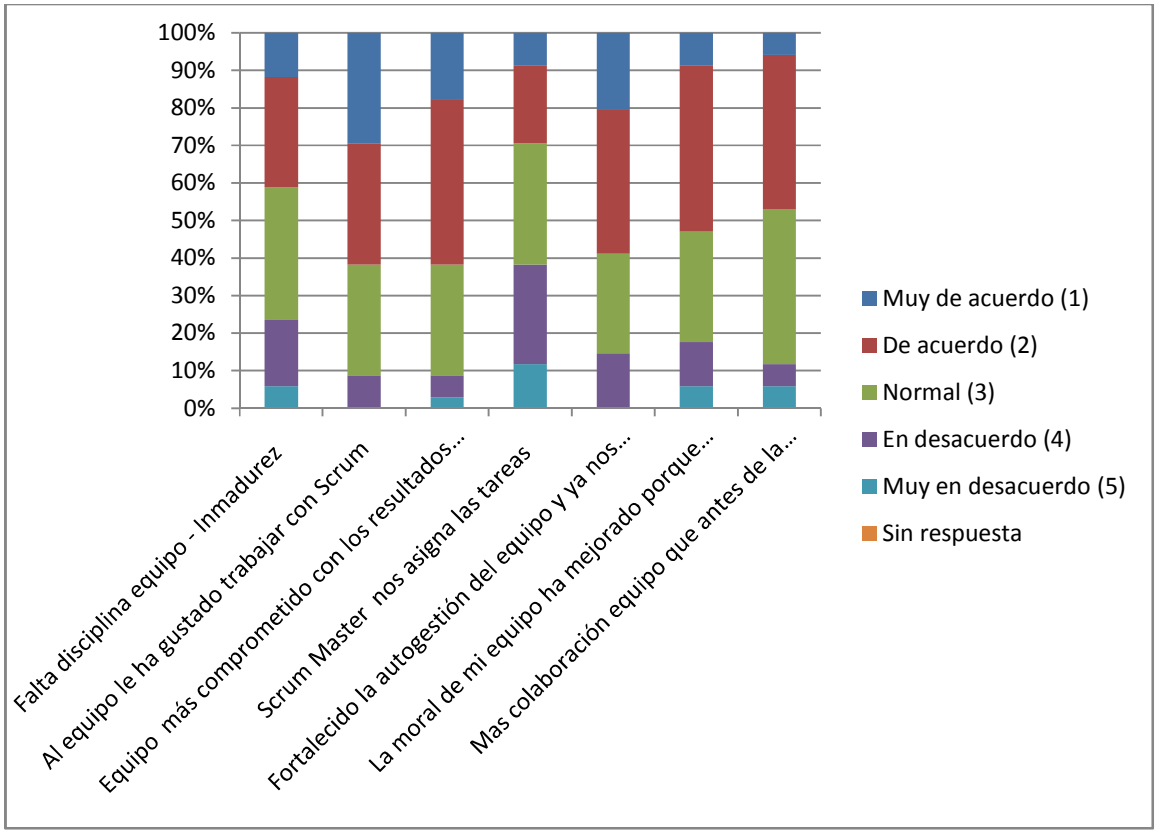

Figura 4.18: Reflejo del equipo durante adopción de *Scrum*.

RQ14. Indique cómo se ve Ud. reflejado dentro del equipo de trabajo *Scrum* al cual pertenece. Puede seleccionar varias opciones.

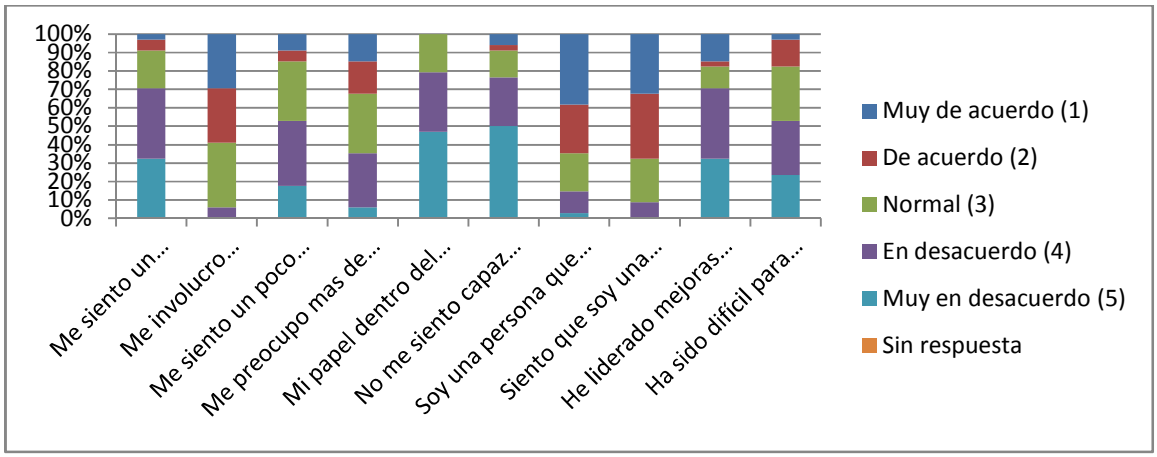

Figura 4.19: Reflejo del participante dentro del equipo *Scrum.*

Para analizar cómo se reflejan los participantes de la encuesta dentro de los equipos *Scrum*, ellos calificaron el nivel de acuerdo o desacuerdo con las siguientes afirmaciones:

1=Me siento un especialista en mi área y me parece que mi trabajo en equipos *Scrum* disminuyen mi productividad personal.
2= Me involucro activamente en las tareas del backlog en cada iteración
3=Me siento un poco frustrado porque considero que mis compañeros no están comprometidos con el proyecto.
4=Me preocupo más de los individuos y sus interacciones y me olvido de los procesos.
5=Mi papel dentro del equipo ha sido pasivo.
6=No me siento capaz de auto-asignarme tareas
7=Soy una persona que fácilmente me adopto a los cambios que se hagan en la organización.
8=Siento que soy una persona que se auto-gestiona y doy ejemplo a mis compañeros.
9=He liderado mejoras de procesos con otros modelos (CMMI, PSP/TSP) y aun no estoy convencido (da) que exista valor agregado al introducir *SCRUM*.
10=Ha sido difícil para mí cambiar de ser gerente de proyecto a *Scrum Master*

En la figura 4.19 se puede observar que los participantes sienten que su participación es muy activa, que es fácil adaptarse a los cambios en la organización y que son capaces de auto-gestionar su trabajo. Esto insinúa una gran apertura al cambio desde prácticas tradicionales a prácticas ágiles o a la mezcla de ambas buscando un balance.

RQ15. ¿Cuál es su nivel de satisfacción con respecto a la manera en que se está llevando a cabo la adopción de *SCRUM* en su organización?

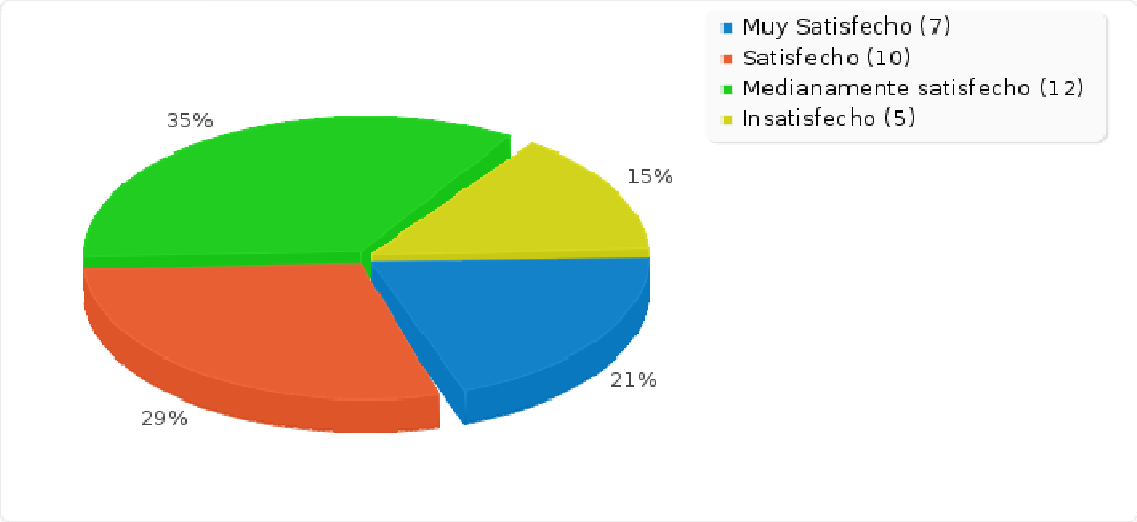

Figura 4.20: Nivel de satisfacción con la manera de adoptar *Scrum*.

Para analizar el nivel de satisfacción con respecto a la manera en que se está llevando a cabo la adopción de *Scrum* en la organización, los participantes de la encuesta escogieron una de las siguientes opciones:

1. Muy Satisfecho
2. Satisfecho
3. Medianamente satisfecho
4. Insatisfecho

La figura 4.20 muestra los resultados encontrados en esta pregunta. Se resalta que la mayoría de los participantes 35%, se encuentran medianamente satisfechos con la manera de adoptar *Scrum*. El 29% se encuentra satisfecho, mientras que el 21% de los participantes se encuentran muy satisfechos. Por esta razón se evidencia que el proceso de adopción de *Scrum* ha tenido gran acogida entre los participantes de la encuesta y en las empresas donde trabajan. Pero cabe resaltar que aún hay temas por mejorar en la adopción

de prácticas ágiles dentro de las empresas. Por ejemplo mejorar la selección de proyectos que en realidad ameritan seguir un proceso con prácticas ágiles.

Capítulo 5. Conclusiones y Trabajos Futuros

Este capítulo final resalta las contribuciones de la investigación al campo de estudio pero también considera las limitaciones del estudio. El capítulo también proporciona conclusiones generales, en donde las preguntas y objetivos, como se expusieron en el capítulo 1, son revisitados, al final se hacen sugerencias para trabajos futuros.

5.1. Contribuciones

La principal contribución del estudio es el desarrollo de una teoría sustentada en datos empíricos de sesiones de acompañamiento y una encuesta. Esta teoría es específica al área de mejora en procesos software y mapea datos de la adopción de *Scrum* en empresas que anteriormente han implantado CMMI y las percepciones sobre los retos y lecciones aprendidas durante dicha adopción. Tres categorías finales fueron generadas directamente basadas sobre datos empíricos: 1) Categoría GT: Valores establecidos en el manifiesto ágil, 2) Categoría GT: Los elementos de *Scrum*, 3) Categoría GT: Las prácticas complementarias a *Scrum*. Al explicar las relaciones entre estas categorías, una teoría sobre el fenómeno bajo estudio fue desarrollada. La teoría explica que los retos generados durante la apropiación de prácticas ágiles (*Scrum*) se organizan alrededor de 3 categorías principales: los valores establecidos en el manifiesto ágil, los elementos de *Scrum* y las prácticas complementarias a *Scrum*.

Se pueden resaltar otras contribuciones, por ejemplo:
La realización de un estudio invivo aplicando un método formal de tipo cualitativo.
El análisis particular de escenarios cuando las empresas con trayectoria en mejora de procesos empiezan a adoptar *Scrum*

El complemento con encuestas individuales.

5.2. Limitaciones

Las limitaciones de este estudio pueden ser caracterizadas como sigue:
Primero, se debe mencionar que los hallazgos y análisis de este estudio están basados en un número limitado de personas encuestadas que respondieron. Un número de 16 entrevistas fueron conducidas y la encuesta fue completada por 45 encuestados. Es decir, los hallazgos del estudio están basados en una pequeña muestra y no se generalizan a una población mayor. En respuesta a esto, se resalta que el propósito de este estudio no fue conducir un estudio cualitativo completo con validez estadística. En vez de esto, la metodología Grounded Theory fue usada para desarrollar una teoría a pequeña escala para explicar los fenómenos relacionados con la adopción de *Scrum* en empresas con CMMI. Las limitaciones de la metodología GT han sido consideradas en el capítulo de Marco Teórico.

También es importante advertir que la autora del estudio tiene poco tiempo usando el método Grounded Theory y dado que la recolección de datos y los resultados de la investigación son altamente dependientes de las habilidades de la investigadora y del rigor en el análisis de datos, la calidad de la investigación puede estar influenciada por sus habilidades y experiencia. Cabe anotar que este estudio muestra otro ejemplo de que el método Grounded Theory puede ser una herramienta muy útil para estudiar las percepciones de los individuos en la mejora de procesos más allá de las áreas en las que tradicionalmente se usa como lo es las ciencias sociales o la salud.

5.3. Conclusiones

En el capítulo anterior se presentaron los resultados del estudio acerca de los retos y lecciones aprendidas durante la adopción de *Scrum* en empresas de desarrollo de software que anteriormente han adoptado prácticas orientadas a planes.

Uno de los hallazgos claves que encontramos al realizar este estudio es que los retos y dudas criticas generadas durante el proceso de apropiación de prácticas ágiles (*Scrum)* en empresas de desarrollo de software que han adoptado CMMI, se organizan alrededor de 3 categorías principales: los valores establecidos en el manifiesto ágil, los elementos de *Scrum* y las prácticas complementarias a *Scrum*.

Este estudio también proporciona algunas recomendaciones que pueden ayudar a las organizaciones a evitar obstáculos en la adopción de *Scrum* dentro de sus prácticas de desarrollo de software integrándolas con prácticas orientadas a planes.

Se destaca que el tipo de recolección de datos usada en el estudio proporcionó información que no es posible obtener sólo con las encuestas, por ejemplo las actitudes y experiencias de los participantes. Sin embargo se consume más tiempo para conducirlo y analizar los datos.

A partir del análisis de datos obtenidos usando el Grounded Theory y complementando con los datos arrojados por la encuesta se lograron dar respuesta a las siguientes preguntas de investigación e identificar conclusiones importantes, las cuales se describen a continuación:

Retos y dudas generadas durante la adopción

Los retos y dudas criticas generadas durante el proceso de apropiación de prácticas ágiles (*Scrum)* en empresas de desarrollo de software que han adoptado CMMI, se organizan

alrededor de 3 categorías principales: los valores establecidos en el manifiesto ágil, los elementos de *Scrum* y las prácticas complementarias a *Scrum*.

A su vez, estas categorías emergieron de los siguientes conceptos que merecen ser analizadas en detalle:

- Prácticas ágiles complementarias a *Scrum*: Se refiere a las prácticas de ingeniería que complementan a *Scrum*, entre ellas: refactoring, pruebas automatizadas, pruebas unitarias y de integración, integración continua, reuso de componentes, XP, trabajo en pares, TDD y DSDM. Coinciden las dos partes del estudio en que es un reto para las empresas la adopción de las siguientes prácticas complementarias:
    - XP (Programación Extrema). Esta práctica no está siendo utilizada por un 36,11% de las empresas.
    - TDD (Desarrollo guiado por pruebas).Esta práctica no está siendo utilizada, por 25% de las empresas.
    - Programación en pares. Esta práctica no está siendo utilizada, por un 22,22% de las empresas. Asimismo esta práctica obtuvo el mayor porcentaje para la categoría No requerida, un 5,56%.
- Historias de usuario: No es claro para las empresas hasta qué nivel de detalle se manejan las historias de usuario, cómo se documentan las conversaciones adicionales, cuáles son los elementos de una historia de usuario, cómo se manejan los requisitos no funcionales, cómo se realiza una transición entre casos de uso y las historias de usuario, si una técnica reemplaza a la otra o se pueden complementar. Se destaca que el porcentaje del nivel alto de adopción de la práctica complementaria de Historia de usuario es de un 36,11% lo cual explica todas las dudas generadas por las empresas durante el periodo de observación.
- Rol de equipo: Aunque este concepto emergió en el periodo de observación alrededor del multitasking, el cual se refiere a cómo manejar el hecho de que un empleado esté en varios proyectos a la vez; al realizar la encuesta se evidencia que el único reto en común entre las dos partes del estudio es la conformación de los equipos; cuando una empresa decide emplear *Scrum*, es necesaria una reorganización de sus equipos de desarrollo existentes en equipos *Scrum*. Los datos

mostraron que las organizaciones presentan retos en la definición del Rol de *Scrum Master* debido a que no existe una clara diferencia entre Gerente de proyecto y el rol como *Scrum Master*, cabe resaltar que ha surgido un nuevo rol como resultado de la mezcla entre estos dos cargos que trata de balancear sus responsabilidades. En algunas empresas se percibe a algunos gerentes de proyectos sin compromiso y renuentes a la implementación de *Scrum* y no se tienen claras cuáles son las expectativas de lo que debe ser un *Scrum Master*.

También han surgido retos en la conformación del equipo relacionados con el rol del tester o analista de calidad; algunas empresas optaron por eliminar roles como el equipo de calidad pues se tiene la percepción de que la calidad es responsabilidad de todo el equipo y que no es necesario este rol.

- Artefactos *Scrum*: Se generaron varias dudas alrededor de la gestión del product backlog. Se tiene muchas dudas sobre este listado de funcionalidades técnicas y de negocio que necesitan ser desarrolladas, ¿cómo se construye y quién y cómo lo debe priorizar?, son algunas de los retos más comúnmente encontrados. Algunas empresas de desarrollo toman la responsabilidad de crear este artefacto, sin embargo es el Product Owner quien por definición de *Scrum* debe tener esa tarea. Para los proyectos en mantenimiento es difícil tener un product backlog priorizado y actualizado por las altas tasas de cambio y priorización de los requerimientos en soporte y mantenimiento; también cabe resaltar el hecho de que en empresas del gobierno el tema de los acuerdos de niveles de servicio guía la priorización de los incidentes y requisitos.
- Negociar con el cliente e influenciarlo: Se percibe dificultad el lograr que el cliente entienda y se adapte a estas nuevas prácticas ágiles, se resalta la falta de conocimiento del framework por parte del cliente y poca claridad del cliente de cuando usar *Scrum* y que esta selección depende del contexto del proyecto. Otro reto son los esquemas de contratación tradicionales que algunos clientes desean mantener (costo, tiempo y alcance fijos). También se percibe un nivel alto de burocracia en las organizaciones que hace difícil la adopción de las prácticas ágiles

- Cambio de cultura y actitud: Unas de las principales dificultades para la adopción de prácticas como *Scrum* es el lento cambio en la cultura organizacional. Aun se percibe resistencia al cambio. Por ejemplo no hay disciplina con las reuniones diarias. Ha sido difícil la interacción directa del usuario con el equipo de trabajo y es necesario elevar el nivel de comunicación entre las personas. Aún existe la costumbre de usar el ciclo de vida en cascada. Sin embargo, es importante resaltar que cuando se encuestó de manera individual a las personas, se percibió que no es un reto para ellas adoptarse a los cambios que se hagan en la organización (64%).
- Releases: Se refiere a la dificultad de la gestión de entregas frecuentes al cliente y la definición del alcance de cada sprint
- Manejo de Soporte y Mantenimiento: Este concepto se relaciona con la necesidad de definir guías de trabajo que orienten a los equipos a reconocer las más efectivas practicas ágiles para la gestión del soporte y mantenimiento, teniendo en cuenta los cambios de prioridad que se presenta como uno de los principales retos.
- Integración de *Scrum* con prácticas robustas: Cabe resaltar este concepto pues además de ser parte de los datos analizados con Grounded Theory también surgió como uno de los más importantes retos expuestos por los participantes de la encuesta. Al analizar el tema de la integración de *Scrum* con procesos definidos orientados a planes como aquellos basados en modelos como CMMI o PSP/TSP, es un reto principal decidir cuales artefactos son suficientes y necesarios buscando balancear los requisitos mínimos para seguir cumpliendo con esos procesos definidos y seguir siendo ágil. Es bien sabido que prácticas como CMMI además de elaborar planes a largo plazo, exigen dejar evidencia directa de todas las prácticas de ingeniería y gestión, por tanto reconocer como generar esas evidencias se percibe como una tarea difícil al intentar integrar *Scrum* y CMMI. Es importante enfocar los esfuerzos de las compañías en un proceso de adaptación de los procesos que permita decidir primero cuales prácticas son las adecuadas para cada situación o contexto de proyecto pero que también dé lineamientos acerca de cómo ser efectivos en la creación y mantenimiento del modelos o documentación del proyecto; quizá

definiendo guías sobre cuales modelos ágiles evolucionan hasta ser parte de la documentación oficial del sistema.

Practicas ágiles adoptadas y su nivel de adopción

Es importante resaltar las prácticas ágiles adoptadas por los diferentes equipos de trabajo en las empresas y su nivel de adopción.

Las practicas con mayor nivel de adopción, combinando los niveles de adopción alto y medio, en las empresas que anteriormente han adoptado CMMI son:
Reunión de Planeación con un 83,33% de adopción en las empresas, seguido por la práctica de Entrega Incremental con un 72,22% de adopción y luego las técnicas: Iteraciones 66,6% y Task Board 66,6% tienen igualmente un nivel de adopción alto comparadas con otras técnicas.
Si se comparan estos resultados con las categorías emergentes y conceptos emergentes en Grounded Theory podemos darnos cuenta que coinciden en la práctica de reunión de planeación, quizá las empresas tiene fortaleza en la planeación por el hecho de que las empresas han adoptado anteriormente prácticas orientadas a planes.

Similitud de los retos del estudio con los retos en la literatura

También se concluye que los retos identificados y caracterizados en el estudio durante la apropiación de las prácticas ágiles son similares a los retos encontrados en la literatura.

Entre los retos más relevantes encontrados en la literatura y que coinciden con los encontrados en el estudio podemos listar los siguientes:

- No se respetó el timebox de los sprints iniciales y se extendieron para acomodar más trabajo o para corregir problemas surgidos durante el sprint [12].
- Poca preparación del rol de *Scrum Master* y Product Owner [12].

- Se considera que hay muchas reuniones las cuáles toman mucho tiempo y algunas veces es difícil concentrarse [6].
- Mantener la calidad de los coaches agiles [6].
- Ineficiencia de las reuniones guiadas por un *Scrum Master* poco enfocado [6].
- Al implementar *Scrum* en un ambiente formalizado el reto fue el desarrollo de un buen product backlog [6].
- El cambio cultural desde un trabajo de especialistas aislados a un estilo de trabajo colaborativo multi funcional [6].
- La falta de claridad con la gestión del product backlog [6].
- El involucramiento del QA, dado que hace parte de varios equipos ágiles y tradicionales, es un reto tener el tiempo continuo que requiere un proyecto ágil para dar a los ingenieros retroalimentación inmediata [6].
- El conocimiento del dominio y la necesidad de especializado conocimiento del dominio hace complejo asegurar un suficiente multi entrenamiento [6].
- Las personas no están seguras de cómo se orientará su carrera profesional en un equipo multi funcional [6].
- Actitud y mente abierta [20].
- Composición del equipo y entrenamiento [20].

En la literatura se pueden encontrar también los siguientes beneficios después de que *Scrum* fue introducido; En [12] se menciona que la satisfacción del cliente aumentó y permitió a los equipos de trabajo trabajar a un ritmo sostenible debido a la reducción del sobreesfuerzo. La retroalimentación del cliente también fue positiva en cuanto al nivel en el cual se involucraba durante el proceso de desarrollo.

Beneficios generados por la adopción de prácticas ágiles

Un aspecto positivo es el hecho de que los profesionales de las empresas reconocen los siguientes beneficios generados por la adopción de prácticas ágiles:

- **A nivel organizacional**

- Dentro de los beneficios organizacionales por la adopción *Scrum*, resaltan los asociados a la entrega continua de valor para el cliente, con estas prácticas ágiles el cliente recibe incrementos de producto en menos tiempo.
- Se aprecia una mayor interacción entre el cliente y el equipo del proyecto, hay una mejora en la comunicación con el cliente pues este debe involucrarse más y también se percibe un aumento en la transparencia entre el equipo y el cliente.
- Los equipos están enfocados a la mejora propia de forma natural y no inducida lo cual permite la inclusión continua de nuevas prácticas y métodos en el proceso tradicional de desarrollo de software.
- A nivel de organización ha permitido una gestión de proyectos más simple y más humana donde el trabajo en equipo se valora más y el crecimiento y conocimiento de las personas se incrementa de manera exponencial.
- La apropiación y apertura del conocimiento de los talentos que están trabajando con esta metodología.
- Se disminuye la resistencia al control de cambios en el producto.
- Se ha reducido en gran medida el sobreesfuerzo.
- Se ha aumentado la productividad del equipo.
- Se han reducido los costos.
- Se manejan esquemas de contratación más equitativos para las partes.
- Se ha eliminado la micro-administración de los equipos.

- **A nivel técnico**
  - Hay una mejor definición de las necesidades del usuario.
  - Se evidencia una planeación más ordenada y participativa, y una mejor visibilidad de los compromisos, sus responsables y los bloqueos.
  - El seguimiento constante de las actividades hace que se puedan tomar acciones correctivas de forma rápida para encausar los problemas.

- Las entregas frecuentes y revisiones del producto por iteraciones hace que se puedan minimizar los riesgos de una entrega completa.
- La retroalimentación temprana por parte del usuario la hace más efectiva, permitiendo que se encuentren más rápido las dificultades y se puedan hacer ajustes a medida que se va avanzando.
- Mejora de la calidad del producto. Evitando el reproceso.
- Se ha aumentado la agilidad para solucionar los problemas y se ha aprendido a solucionar escenarios complicados en corto tiempo.

- **A nivel humano**
  - Incremento de la confianza entre cliente y equipo de desarrollo que mejora la dinámica de trabajo, coordinación, colaboración, cohesión y sinergia entre miembros del equipo.
  - Los equipos se muestran más motivados y sincronizados y se percibe mayor entusiasmo y alegría de las personas del equipo.
  - Se percibe disciplina y mejor comunicación en el equipo de trabajo.
  - Empoderamiento de las personas para resolver los impedimentos
  - Orientación al logro que influye en el éxito con esta metodología.
  - Oportunidad del equipo para aprender nuevas habilidades y destrezas y proponer nuevas ideas.
  - Aplicación de "Conocimiento Neto" con el toque de experiencia de cada integrante que conforma el equipo.

Lecciones aprendidas durante proceso de apropiación de prácticas ágiles

Es importante resaltar las siguientes lecciones aprendidas durante el proceso de apropiación de prácticas ágiles en empresas que han adoptado prácticas orientadas a planes:

- Se debe tomar lo mejor de cada práctica para crear un proceso adecuado a las necesidades de la organización y no solo implantar el framework *Scrum* como la bala de plata que solucionará todo.

- La adopción de *Scrum* exige la capacitación de las personas en todos los niveles de la organización.
- Se recomienda hacer la adopción de manera incremental con equipos empoderados y comprometidos buscando que los miembros del equipo no sean todos junior y con equipos co-localizados.
- Es muy importante aclarar las expectativas que se tiene de un equipo auto-organizado y del rol de *Scrum Master*.
- Se recomienda asesorarse de una empresa con experiencia en *Scrum* para que realice coaching a los equipos de trabajo. Igualmente el *Scrum Master* necesita estudiar técnicas de coaching si quiere hacer bien su labor.
- Es importante sensibilizar al *Product Owner* externo sobre *Scrum* y sus ventajas, mostrando resultados.
- Involucrar y capacitar al cliente desde el principio y todo el proyecto para que tome conciencia del papel que juega dentro del equipo.
- El *Product Owner* debe ser siempre alguien del cliente (involucramiento del usuario).
- La disponibilidad y comunicación clara y continua entre el equipo y el *Product Owner* es un factor de éxito primordial.
- Un *Product Owner* debe estar disponible para el proyecto mínimo un 45% del tiempo y el ideal un 80% o más.
- Para que un proveedor de desarrollo de software implemente adecuadamente metodologías ágiles debe contar con un marco contractual adecuado con los clientes.
- Con relación a las prácticas de ingeniería, en lo posible se debe complementar *Scrum* con prácticas técnicas de desarrollo de software, como las propuestas por XP.
- Siempre se deben realizar las revisiones pares para evitar errores en el código.
- Es importante trabajar en conjunto desde la planeación.
- Cada iteración debe estar aprobada por el usuario para evitar reprocesos costosos.
- Si se desconoce una tarea es mejor no estimarla o no ser optimista en la estimación.

- Se recomienda que el "Sprint 0" entregue software funcionando.
- La definición de hecho es un instrumento valioso a la hora de entregar el producto.
- La reunión retrospectiva al final de cada entrega crea lazos de confianza.
- Seguir y cumplir con lo pactado en el sprint y no extender las fechas de entrega.
- No personalizar *Scrum* a los requerimientos de la organización.
- Documentar las actividades permite una trazabilidad adecuada de errores.

Se recomienda para los proyectos de soporte y mantenimiento de aplicaciones la combinación de *Scrum* y *Kanban.*

5.4. Trabajos futuros

Este estudio de maestría no proporciona el alcance y los recursos para levantar esta teoría a un nivel más generalizado. Ese podría ser un tema para trabajos futuros dado que muchas teorías específicas pueden construir la base para una teoría más general. Otros trabajos que dan continuidad a este estudio son los siguientes:

Realizar un estudio focalizado al conjunto de empresas del programa inicial de Ruta N sobre los impactos de la adopción a largo plazo en cuanto al tema de calidad, mantenibilidad de producto y satisfacción del cliente y del equipo, con el propósito de compararlos con los impactos de prácticas orientadas a planes.

Realizar este estudio en otras empresas del país para identificar posibles factores de éxito relacionados con el tema de la cultura organizacional y regional.

Bibliografía

[1] Goede, Roelien. De Villiers, Carina. *The Applicability of Grounded Theory as Research Methodology in studies on the use of Methodologies in IS Practices*, 2003.

[2] Schwaber, Ken. *Agile Project Management with Scrum.* Redmond: Microsoft Press, 2004.ISBN 0-7356-1993-X.

[3]Barney, Glaser G. Strauss, Anselm L. *The discovery of grounded theory: Strategies for qualitative research.* New Jersey: Transaction Books, 2009. ISBN: 978-0-202-30260-7.

[4]Basili, Victor Robert. *The past, present, and future of experimental software engineering.* J. Braz. Comp. Soc. [online].2006, vol.12, n.3, pp. 7-12.ISSN 0104-6500.

[5] Bustard, David. Wilkie, George. Greer, Des.*The Maturation of Agile Software Development Principles and Practice: Observations on Successive Industrial Studies in 2010 and 2012.*pp.139-146, 2013 20th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS), 2013. Ver en: http://origin-www.computer.org.ezproxy.eafit.edu.co/csdl/proceedings/ecbs/2013/4991/00/4991a139.pdf

[6] Marchenko, Artem. Abrahamsson, Pekka. *Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges.* pp. 15-26, Agile 2008 Conference, Toronto, 2008.ISBN 978-0-7695-3321-6.

[7] de Souza Mariz, Leila M. R...França, A. César C. da Silva, Fabio Q. B. *An Empirical Study on the Relationship between the Use of Agile Practices and the Success of Software Projects that Use Scrum.*2010 Brazilian Symposium on Software Engineering, 2010.

[8] Overhage, Sven. Schlauderer, Sebastian. *Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects.*2012 45th Hawaii International Conference on System Sciences, IEEE, 2012.

[9] CMMI. *Published Appraisal Results*. https://sas.cmmiinstitute.com/pars/pars.aspx/ (Visitado en 2013)

[10] Pino, Francisco J. García, Félix. Piattini, Mario. *Revisión sistemática de mejora de procesos software en micro, pequeñas y medianas empresas.*pp. 6-23. REICIS, Revista Española de Innovación, Calidad e Ingeniería del Software, Vol. 2, número 001, Madrid, 2006.

[11] Janes, Andrea. Succi, GianCarlo. *The Dark Side of Agile Software Development*, ACM, Tucson, AZ, 2012.

[12] Mann. Chris, Maurer. Frank. *A Case Study on the Impact of Scrum on Overtime and Customer Satisfaction.*EnProceedings of the Agile Development Conference (ADC '05). IEEE Computer Society, Washington, DC, USA, 2005.

[13] *Teoría fundada: arte o ciencia.*Universidad de Antioquia. Facultad de Ciencias Sociales y Humanas. Centro de Estudios de Opinión.

[14] Escobar-Sarmiento, Victor. Linares-Vásquez, Mario. *A Model for Measuring Agility in Small and Medium Software Development Enterprises.*pp. 1-10. Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En. Medellín. 2012. ISBN: 978-1-4673-0794-9

[15] Begel, Andrew.Nagappan, Nachiappan. *Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study.* ESEM '07:255–264, IEEE Computer Society, Washington, 2007.

[16] Sutherland, Jeff. Jakobsen, Carsten Ruseng.Johnson, Kent. *Scrum and CMMI Level 5: The Magic Potion for Code Warriors.* Hawaii International Conference on System Sciences, Proceedings of the 41st Annual. IEEE, Waikoloa, HI.2008. ISSN: 1530-1605

[17] Hoda, Rashina. Noble, James. Marshall, Stuart. *Self-Organizing Roles on Agile Software Development Teams*. IEEE Transactions on Software Enginnering. Vol. 39, No 3, 2013.

[18] KentBeck et al., "*Manifesto for Agile Software Development.*"2001. Disponible en: <http://agilemanifesto.org/>. (Visitado en: 4/2013).

[19] R. Anaya, L. Gómez. *Lecciones Aprendidas en el Acompañamiento Masivo para Mejora de Procesos en Empresas de Software: Un Caso Colombiano.* Memorias XV Congreso Iberoamericano de Ingeniería de Software CIBSE. Buenos Aires. Abril Del 2012.

[20] Senapathi, Mali. Srinivasan, Ananth. *Sustained Agile Usage: A Systematic Literature Review*. pp. 119-124. EASE '13 Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering. ACM New York, USA. 2013. ISBN: 978-1-4503-1848-8.

[21] Zhang, Lina. Shao, Dan. *Research on Combining Scrum with CMMI in Small and Medium Organizations.* 2012 International Conference on Computer Science and Electronics Engineering. IEEE. Hangzhou. 2012. ISBN: 978-1-4673-0689-8.

[22] Requisitos de postulación Programa para la transferencia de *SCRUM* – Ruta N. 2012. En www.rutanmedellin.org/info/Paginas/invitacion_programa_piloto_*Scrum*.aspx. (Visitado en 2013).

[23] Attention- Juggling in the High-Tech Office. http://www.nytimes.com/2006/06/04/business/yourmoney/04advi.html?_r=0 (Visitado en 2013)

[24] Travassos, G. H., Kalinowski, M. iMPS 2011: Desempeño de las empresas que adoptaron el modelo MPS de 2008 a 2011. Campinas, SP: SOFTEX, 2012.36p. 2011. En http://www.softex.br/wp-content/uploads/2013/08/iMPS-2010.pdf (Visitado en 2013)

[25] "IEEE Standard Glossary of Software Engineering Terminology", IEEE STD 610.12-1990, 1990.

[26] Definición de Kanban Board. http://guide.agilealliance.org/guide/kanban.html. (Visitado en 2013)

[27]Hanakawa, Noriko. *A process refactoring for software development with process complexity and activity priority lists*.Proceedings of the Sixth International Conferenceon Software Process and Product Measurement. Nara. 2011. ISBN: 978-1-4577-1930-1

[28] Aaen, Ivan. Arent, Jesper. Mathiassen, Lars. Ngwenyama, Ojelanki. *A conceptual MAP of Software Process Improvement.* Scandinavian Journal of Information Systems, 2001.

[29] Agile Modeling. http://www.agilemodeling.com/ (Visitado en 2013)

[30] Goldenson, Dennis R. y Gibson, Diane L. *Demonstrating the Impact and Benefits of CMMI®: An Update and Preliminary Results*.SEI, Pittsburgh, PA. 2003. CMU/SEI-2003-SR-009

[31] Ruiz-González, Francisco. *MANTIS: Definición de un Entorno para laGestión del Mantenimiento de Software.* Tesis Doctoral. Universidad de Castilla-La Mancha, España. Junio de 2003.

[32] Montoni, Mariano A. Rocha, Ana R.*Applying Grounded Theory to Understand Software Process Improvement Implementation*, 2010 Seventh International Conference on the Quality of Information and Communications Technology. IEEE. Porto, 2010. ISBN: 978-0-7695-4241-6

[33] Sjoberg, Dag I.K. Dyba, Tore. Jorgensen, M.*The Future of Empirical Methods in Software Engineering Research*. Future of Software Engineering, 2007.FOSE '07, Minneapolis, MN. 2007.

[34] *CMMI para Desarrollo, Versión 1.3*. CMU/SEI-2010-TR-033. Software Engineering Institute, Carnegie Mellon University. 2010.

[35] Thomas Chau, Frank Maurer, Grigori Melnik. *"Knowledge Sharing: Agile Methods vs. Tayloristic Methods"*. In Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03). IEEE. 2003.

[36] Travassos. Guillerme. H. Basili. Victor. *Experimental Software Engineering: an Introduction.*1st Experimental Software Engineering Latin American Workshop.http://lens.cos.ufrj.br:8080/eselaw/papers/minitutorial.pdf. (Visitado en 2013)

[37] McFeeley, B. *IDEAL^{SM} : A Users Guide for Software Process Improvement. Handbook.* CMU/SEI-96-HB-001. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PE, USA.1996. http://www.sei.cmu.edu/reports/96hb001.pdf

[38] Tor Erlend Fægri. *Trends within Software ProcessImprovement (SPI)*, PhD Trial Lecture.http://www.idi.ntnu.no/research/doctor_theses/fagri_trial_lecture.pdf (Visitado en 2013)

[39]People Capability Maturity Model® (P-CMM®), Version 2.0. CMU/SEI-2001-MM-01.Software Engineering Institute, Carnegie Mellon University. 2001.

[40] Lehmann, Hans. *Grounded Theory and Information Systems:Are We Missing the Point?*.Proceedings of the 43rd Hawaii International Conference on System Sciences – 2010.IEEE.Honolulu, HI. 2010. ISBN: 978-1-4244-5510-2

[41] Kent Beck, *Extreme Programming Explained*, First Edition. September 29, 1999. ISBN: 0201616416

[42]Kniberg, Henrik. Skarin, Mattias. *Kanban and Scrum - making the most of both.*C4Media. 2010. Descargado de http://www.infoq.com/minibooks/kanban-*Scrum*-minibook.

[43] Hammond, Susan. Umphress, David. *Test Driven Development: The State of the Practice.*Pp. 158-163. ACM-SE '12 Proceedings of the 50th Annual Southeast Regional Conference New York, USA. 2012.

[44] Capability Maturity Model[SM]for Software, Version 1.1. CMU/SEI-93-TR-024.Software Engineering Institute, Carnegie Mellon University. 1993.

[45]Mapa de tribus de prácticas ágiles. http://guide.agilealliance.org/subway.html (Visitado en 2013)

[46] Kniberg, Henrik. Farhang, Reza.*Bootstrapping Scrum and XP under crisis: A story from the trenches.*pp. 436-444, Agile 2008 Conference, Toronto, 2008. ISBN 978-0-7695-3321-6

[47] Boehm, B.Turner,R. *Observations on balancing discipline and agility*.pp.32-39.Proceedings of the Agile Development Conference, Salt Lake City, USA. 2003.

[48]Paulk, M. *"Extreme Programming from a CMM Perspective"*. IEEE Software. 2001.

[49] Schwaber, Ken. Sutherland, Jeff. *La Guía de Scrum.* Scrum.org. Julio de 2013.

Anexos

ANEXO A: ENCUESTA

Los datos fueron recolectados a través de una encuesta que está estructurada en las siguientes 3 secciones:

Sección I. Información general del encuestado y el proyecto

Pregunta 1. **¿A cuál empresa pertenece Ud.?**
Se listaron las empresas que participaron del programa de Ruta N.

Pregunta 2. **¿Cuántos años de experiencia tiene Ud. en el desarrollo de software?**

Pregunta 3. **¿Cuál es el cargo o rol que Ud. desempeña en su empresa? Por favor escoja todas las que apliquen.**

1. Ingeniero desarrollador
2. Gerente de proyecto
3. Gerente de proyecto
4. Arquitecto
5. Ingeniero de prueba
6. Otro ¿Cuál?

Pregunta 4. **¿Actualmente forma parte de un equipo de trabajo *Scrum*? ¿Desde hace cuánto tiempo?**

a. Sí
b. No

Comentarios**:**

Pregunta 5. **¿Ha trabajado en equipos *Scrum* anteriormente?**

a. Sí
b. No

Pregunta 6. **Indique en meses cuánto tiempo lleva su equipo usando *Scrum* en el proyecto actual.**

Pregunta 7. **¿Está trabajando actualmente en la primera versión del producto o en mantenimiento?**

a. Primera versión del producto
b. Mantenimiento del producto
c. Otro

Sección II. Adopción actual de *SCRUM* y prácticas complementarias

Pregunta 8**. De las siguientes prácticas de *Scrum* (ver Tabla 6.1), indique el nivel de adopción que hasta el momento percibe en su organización. De acuerdo a la siguiente convención:**

(1) Alto
(2) Medio
(3) Bajo
(4) No se usa
(5) Se planea usar
(6) No requerida
(7) No sé

	Valor						
Prácticas de *Scrum*	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Entrega incremental (Cada producto resultante es usable)							
Timebox (Manejo de duración fija durante tareas del desarrollo)							
Iteraciones - Manejo de sprints							
Daily Stand-up meetings (Reuniones diarias con todo el equipo)							
Burndown Chart (Grafico de avance)							
Task Board (Tablero de tareas)							
Definition of Done (Definición de Hecho)							
Definition of Ready (Definición de Listo)							
Reunión de planeación							
Estimación por puntos							
Estimación relativa							
Planning Poker							
Backlog							
Backlog Grooming (Limpieza del backlog)							
Lista de impedimentos							
Reunión Retrospectiva							

Tabla 2: Prácticas de *Scrum*.

Pregunta 9**. De las siguientes prácticas ágiles complementarias a *Scrum* (Ver Tabla 6.2), indique el nivel de adopción de la práctica en su organización. De acuerdo a la siguiente convención.**

(1) Alto

(2) Medio

(3) Bajo

(4) No se usa

(5) Se planea usar

(6) No requerida

(7) No sé

	Valor						
Practicas complementarias a *Scrum*	**1**	**2**	**3**	**4**	**5**	**6**	**7**
XP (Extreme Programming)							
Historias de usuario							
Interacción directa con el cliente							
Visual Story Mapping							
Kanban Board (Tablero Kanban)							
Pair Programming (Programación en pares)							
TDD (Testing Driven Development)							

Estándar de codificación del equipo							
Refactoring							
Control de versiones							
Integración continua de código							
Liberaciones frecuentes							
Pruebas automatizadas							
Builds automatizados							
Diseño simple							
Pruebas unitarias (Código que prueba código)							

Tabla 3: Prácticas ágiles complementarias a *Scrum*.

Pregunta 10.**¿Cuáles son las principales dificultades que Usted ha observado en la adopción de *Scrum* dentro de su organización, desde el punto de vista organizacional, técnico, humano y de integración con prácticas existentes como CMMI, PSP/TSP? (Identifique mínimo 3 dificultades).**

Pregunta 11.**¿Cuáles son los principales logros o beneficios que Usted ha observado hasta el momento por la adopción de *Scrum* dentro de su organización, desde el punto de vista organizacional, técnico, humano y de integración con prácticas existentes como CMMI, PSP/TSP? (Identifique mínimo 3 logros).**

Pregunta 12. **¿Cuáles son las lecciones aprendidas que Ud. ha identificado hasta el momento respecto a la apropiación de *Scrum* en su organización? (Identifique mínimo 3 lecciones aprendidas)**

Sección III. Percepciones generales

Pregunta 13.**Indique como ve reflejado a su equipo durante la adopción de *Scrum* (Ver tabla 6.3). De acuerdo a la siguiente convención.**

- **Muy de acuerdo**
- **De acuerdo**
- **Normal**
- **En desacuerdo**
- **Muy en desacuerdo**

	Valor				
Reflejo del equipo durante adopción	**1**	**2**	**3**	**4**	**5**
A mi equipo le falta disciplina por la inmadurez de algunos de sus miembros					
A mi equipo le ha gustado trabajar con *Scrum*					
Siento que mi equipo está más comprometido ahora con los resultados del proyecto que antes de *Scrum*					

Aun es el *Scrum Master* quien nos asigna las tareas para realizar durante la iteración					
Siento que se ha fortalecido la autogestión del equipo y ya nos auto asignamos las tareas de la iteración					
La moral de mi equipo ha mejorado porque adoptamos *Scrum*					
Los desarrolladores y testers de mi equipo colaboran más ahora que antes de la adopción de *Scrum*					

Tabla 4. Reflejo del equipo durante la adopción de *Scrum*.

Pregunta 14. **Indique cómo se ve Ud. reflejado dentro del equipo de trabajo *Scrum* al cual pertenece. Puede seleccionar varias opciones (Ver Tabla 6.4) y se califica de acuerdo a la siguiente convención.**

(1) Muy de acuerdo

(2) De acuerdo

(3) Normal

(4) En desacuerdo

(5) Muy en desacuerdo

	Valor				
Reflejo del equipo durante adopción	**1**	**2**	**3**	**4**	**5**
Me siento un especialista en mi área y me parece que mi trabajo en equipos *Scrum* disminuyen mi productividad personal					
Me involucro activamente en las tareas del backlog en cada iteración					
Me siento un poco frustrado porque considero que mis compañeros no están comprometidos con el proyecto					
Me preocupo más de los individuos y sus interacciones y me olvido de los procesos					
Mi papel dentro del equipo ha sido pasivo					
No me siento capaz de auto asignarme tareas					
Soy una persona que fácilmente me adopto a los cambios que se hagan en la organización					
Siento que soy una persona que se auto gestiona y doy ejemplo a mis compañeros					
He liderado mejoras de procesos con otros modelos (CMMI, PSP/TSP)					

y aun no estoy convencido(da) que exista valor agregado al introducir *SCRUM*					
Hay sido difícil para mí cambiar de ser gerente de proyecto a *Scrum* master					

Tabla 5. Reflejo del encuestado durante la adopción de *Scrum*.

Pregunta 15. **¿Cuál es su nivel de satisfacción con respecto a la manera en que se está llevando a cabo la adopción de *SCRUM* en su organización?**

ANEXO B: TRANSCRIPCIONES DE OBSERVACIONES

A continuación se muestran de manera detallada algunas de las transcripciones que soportan el análisis de los datos recolectados; de acuerdo al orden en el capítulo 4.

Transcripciones de observaciones asociadas a la categoría Prácticas Complementarias a *Scrum*.

"*En unos de los cursos se trató el tema de refactoring para no tener cosas repetidas. El desarrollador lo hace al final de los sprint, estos temas consumen mucho tiempo si se contemplan dentro de todos los sprints*." – P6, Gerente de proyecto.

"*No se usa integración continua y no se tienen pruebas automatizadas.*" – P8, Ingeniero de desarrollo.
"*¿Qué herramientas de automatización podrías recomendar de las que conoces? ¿Será que Selenium es buena*?" – P3, Analista.

"*Sabemos la importancia de que el analista de certificación o calidad trabaje con el desarrollador pero ¿cómo se logra que el tiempo que comparten si sea valioso? Y que no sea solo el analista certificador simplemente viendo lo que el desarrollador codifica."* – P3, Analista.

"*¿Qué tan detalladas pueden estar las historias de usuario? ¿Las historias de usuario son igual a requisitos*?" – P1, Ingeniero de desarrollo.

"*Se tienen dudas sobre la documentación en requisitos, casos de uso, historias de usuario. Entendemos que es importante que se use la técnica que permita tener la documentación necesaria.*" –P3, Analista.

"*¿Hasta qué nivel de detalle se deja una historia de usuario? ¿Las conversaciones? Si deseamos hacer un uso intensivo de prototipos donde guardamos el detalle. ¿Qué herramientas usar para requerimientos, se pueden usar los casos de uso*?" – P5, Arquitecto.

"*¿Cómo conviven Scrum y PSP/TSP y CMMI? Dado que la técnica de PSP usan plantillas de juntas y estas no pueden ser cambiadas por ejemplo por un video. (Metodología). Adicionalmente la trazabilidad de los artefactos en proceso se debe mantener*." – P3, Analista de procesos.

"*Actualmente no tenemos una deuda metodológica grande de acuerdo a CMMI, pensamos realizar un sprint para actualizar lo que nos falta del proceso.*"-P8, Gerente de proyectos.
"*En TSP/PSP planeo por semana y luego asignamos a cada persona lo que se debe realizar… El rol de Scrum Master tiene que ver con la revisión de las tareas que pasaran al today?*" –P2, Gerente de proyecto.

"[Analizamos las] diferencias entre PSP, TSP y metodologías ágiles y son parecidos, la diferencia es la documentación generada que en PSP es más detallada. PSP y TSP se habla del postmortem, ¿todo eso es facturado, la retrospectiva es facturada o es una lección aprendida del grupo que no afecta el cliente?"–P3, Analista.

"*¿Usando Kanban, puede ser el product backlog [usado], para colocar todas las Oc que están pendientes por atender? En la actualidad tenemos una planeación de la asignación de las OC para el proyecto fénix, esto puede ser usado con [como] product backlog*." –P1. Gerente de proyecto.

"*No tenemos límites de OC´s en cada una de las etapas del flujo de trabajo (todo depende de la capacidad disponible en el equipo de trabajo). Tenemos dudas en ¿cómo especificar el límite en cada etapa del flujo de atención definido, de acuerdo a lo que se dice con Kanban? En los [tableros] Kanban el WIP depende de la capacidad en horas disponibles*

por semana, ¿cómo especificar limite en ese WIP? ¿Cómo identificar capacidad en cada etapa para pasar a la siguiente?" –P1. Gerente de proyecto.

Transcripciones de observaciones asociadas a la categoría Elementos *Scrum*.

Rol de Equipo

"¿Las personas del equipo como se conforman? Tenemos el rol de ingeniero desarrollador, esa persona hace todo el ciclo de desarrollo de software, anteriormente teníamos un área de calidad fija que era responsable de la calidad, aquí el responsable de la calidad no es un área sino un equipo, es un cambio que inicio hace 15 o 20 días."–P8. Gerente de proyecto.

Artefactos Scrum

"*Pasar del listado inicial hecho en Excel al resultado del Visual Story Mapping, cuáles irían en el primer nivel. Se entiende que no hay correspondencia uno a uno entre las actividades y las funcionalidades desglosadas"* –P5, Comercial.

"La forma de priorizar estará dado por los acuerdos contractuales y ANS establecidos en el proyecto." –P1, Gerente.

"En los libros se refiere a que toda la estimación que se hace es después de tener las historias de usuarios, es decir ya se tiene el detalle, entonces lo que estamos diciendo es que al desglosar estas tareas se deben incluir también no solo el diseño sino una reunión con el usuario, codificación y el postmortem? es cierto?"-P3, Analista.

"Sobre la cartelera que se hace con las columnas de por hacer, en progreso, hecho, al realizar la estimación del proyecto por ser tan corto se decidió que se estimara más bien como la etapa de diseño en general y no la tarea, '¿hasta qué punto y cómo se deben desglosar las tareas del proyecto para que no sean tan a muy gran nivel pero tampoco tan detalladas?" -P3, Analista.

Rol de Product Owner

"En el daily meeting las reglas se expusieron al cliente, que era un PMP, pero había problema de que interviniera en la reunión, era el gerente del proyecto por parte del cliente. Quizá porque era de Bogotá y no entendía."–P6, Gerente.

"En el tema técnico, nos hemos encontrado con [venga] saquémosle provecho a esta persona que es más costosa. Al inicio el PO quiere estar en la reuniones pero con el tiempo sabe en cual reunión si genera valor."–P8, Gerente.

Rol de Scrum Master

"[El rol de] *Scrum Master: Estaría a cargo del director del proyecto, con el objetivo de hacer seguimiento y mentoring constante, de forma tal que se apliquen adecuadamente la combinación de conceptos."* –P1, Gerente.

Herramientas

"El problema es que usamos ya unas herramientas BOBJ… Es complicado usar herramientas del mercado. Lo ideal es que todo sea ágil. Como se maneja la integración. El mayor dolor es el tema de BI." – P6, Gerente de proyecto.

Sprint

"A partir de allí se han realizado 2 sprint. El primer sprint de 5 semanas y se extendió a 6 semanas. El segundo sprint de 5 semanas y nuevamente se extendió a 6 semanas. Los retrasos fueron causados por problemas de infraestructura de Sura, paso entre ambientes que afectaron la ejecución de lo estimado." –P8, Gerente de proyectos.

Reunión diaria

"Las reuniones diarias permiten colaboración pues es necesario saber cómo lo va a tocar [orden de cambio] y cuando lo va a tocar. Como los sistemas son más complejos se pueden hacer cambios de asignación." –P1, Desarrollador.

"Se realizan las reuniones de daily meeting pero nos dispersamos." –P8, Desarrollador.

Transcripciones de observaciones asociadas a la categoría Valores del manifiesto ágil

Retos asociados a Individuos y su interacción

Cambio de cultura y actitud

"Hay un reto identificado, el manejo de la cultura organizacional. Se necesita un cambio de mentalidad." P6, Gerente de proyecto

*"Las reuniones diarias, a veces estas reuniones son complicadas porque no llegaban todos a tiempo, [hay] impuntualidad. Quizá por el rango de edades [de los equipos de trabajo] - 20 a 26 años, cierto esquema que aún no es disciplinado."*P6, Gerente de proyecto

*"Se daba el caso que preguntaba el PO… porque no hacemos la daily[meeting] a las 7 y 15 y no a la 7 am porque necesitamos un tiempito para prepararnos… sin una justificación, el mismo se va retirando de esa reunión."*P8, Gerente de proyecto

"La gente todavía tiene esa figura de jefe que da la orden y señala para poder actuar y no solo pasa en personas jóvenes, a veces no se siente con la autoridad para mejorar el equipo o tomar decisiones , ahí si recurro al jefe o al gerente de proyecto para remover al más lento, para no parecer yo el sapo." P8, gerente de proyecto.

Gestión del conocimiento

"No tratamos de meternos con lo de los demás porque podemos obstaculizar la curva de aprendizaje." P8, Desarrollador.

"Cómo hago como desarrollador para negociar con mi gerente aquellos procesos de consulta e investigación que son cíclicos no lineales y a veces podemos tomarnos más de lo necesario para estudiar un problema." P4, Desarrollador.

"El analista de certificación colaboro en prueba unitaria pero fue más lento por ponerse de acuerdo con el tema."–P3, Gerente de proyecto.

Fortalecer habilidades en los equipos

*"La gente cree ser auto-gestionada, manejar equipos adecuadamente, pero hay que desarrollar la comunicación, no solo el correo, usar el cara a cara, manejo de equipos, sacar el provecho a los equipos. Hay personas pasivas, hay que conocer el equipo, el tema de CMMI eso lo dicen ellos. El tema del capataz… hasta PMI ya tiene un tema para control de equipos agiles"*P8, Gerente de proyecto.

"Debemos apalancarnos para esto en la transferencia de conocimiento, aquí la auto-organización es libre pero acá [Sergio] es quien asigna, se toma unos 20 0 30 min al día y creo que así nos ha dado resultado porque no tenemos cultura de auto-asignación." P1, Gerente de proyecto.

Retos asociados a Colaboración con el Cliente

Negociar con el cliente e influenciarlo

"¿Qué podemos hacer desde nuestro lugar para que el cliente haga lo que necesitamos?"– P8, Desarrollador.

"¿El cliente exigía trabajar con metodologías agiles, es importante que el cliente este en este cuento pero como influenciarlo, como se vende la idea?"–P1, Gerente de proyecto.

Licitaciones y contratación

"Se explica que se tienen holguras pero como el proyecto es a tiempo y costo cerrado no es full Scrum, luchan porque las horas productivas sean 6 horas al día. Si son menos horas tenemos un grave problema."–P8, Gerente de proyecto.

"¿Cómo manejar licitaciones del gobierno, a costo fijo y tiempo fijo? Donde el cliente no tiene tiempo para reunirse de manera presencial Caso gobierno."–P4, Gerente de proyecto.

"La principal preocupación es que el cliente cambia mucho de parecer y no cambia costo ni tiempo."–P5, Gerente de proyecto.

"¿Cómo vender los skelleton al cliente? La metodología, ¿cómo se hará la negociación con el cliente? Sobre todo el tema de las entregas."–P5, Gerente de proyecto.

"… si el proyecto sigue siendo de valor cerrado como no ir en contravía con SCRUM?"–P8, Gerente de procesos.
"Cómo se cotiza un proyecto en el cual se va a cotizar Scrum?"–P7, Consultor.

Retos asociados a Software Funcionando

"Cómo saber para cada post it su alcance y no vamos a fallar metiéndolo en un release."–P4, Gerente de proyecto.
"¿Cómo vender los skelleton al cliente? La metodología, cómo se hará la negociación con el cliente? Sobre todo el tema de las entregas."–P5, Comercial.

Retos asociados a Respuesta al Cambio

"Para proyectos de soporte y mantenimiento (bandejas de atención de incidentes y requerimientos), el cambio de priorización, es algo que se presenta todos los días, por lo cual es complicado planear sprint, por lo cual decidimos usar Kanban. ¿Es una buena elección?" –P1, Gerente de proyecto.
"¿Cómo planear con un esquema tan dinámico como en soporte?"–P1, Gerente de proyecto.

"*El backlog es un listado por hacer, ¿será lo que está asignado o no en la herramienta del cliente?*".–P1, Gerente de proyecto.