

# TEORÍA VS REALIDAD

UTILIZACIÓN DE METODOLOGÍAS ÁGILES  
EN UNA EMPRESA DE DESARROLLO DE  
SOFTWARE



**Modalidad: Trabajo de Aplicación y técnicas de administración en  
Ambiente Real**

**Alumna: Ariana Maria Haskour**

**arihaskour@gmail.com**

**Tutor: Esteban Mulki**

**2023**



## Índice

|  |           |
|--|-----------|
| <b>Resumen .....</b>                                   | <b>2</b>  |
| <b>Introducción .....</b>                              | <b>4</b>  |
| <b>Problema.....</b>                                   | <b>5</b>  |
| <b>Preguntas de Investigación .....</b>                | <b>5</b>  |
| <b>Objetivo General .....</b>                          | <b>6</b>  |
| <b>Objetivos Específicos.....</b>                      | <b>6</b>  |
| <b>Marco Teórico .....</b>                             | <b>6</b>  |
| Modelo secuencial de procesos .....                    | 7         |
| Explorando el mundo de las “Metodologías Ágiles” ..... | 8         |
| Manifiesto Ágil .....                                  | 9         |
| Principios del manifiesto agile.....                   | 11        |
| Metodología ágil Kanban.....                           | 26        |
| <b>Desarrollo.....</b>                                 | <b>32</b> |
| <b>Comparaciones y contrastes .....</b>                | <b>36</b> |
| <b>Criticidad de los hallazgos .....</b>               | <b>51</b> |
| <b>Conclusiones.....</b>                               | <b>56</b> |
| <b>Recomendaciones.....</b>                            | <b>57</b> |
| <b>Bibliografía .....</b>                              | <b>60</b> |
| <b>Apéndice .....</b>                                  | <b>61</b> |



## Resumen

Las metodologías ágiles se han convertido en una tendencia cada vez más popular en el desarrollo de software. Estas metodologías se enfocan en la entrega de valor al cliente de manera rápida y constante, a través de la colaboración y la adaptación a los cambios. En este contexto, las empresas de desarrollo de software han adoptado estas metodologías para mejorar su eficacia y eficiencia en la entrega de productos de calidad.

A pesar de la formación teórica recibida por los equipos que utilizan dicha metodología, se observan discrepancias en la implementación práctica dentro del contexto empresarial, lo cual puede generar consecuencias significativas en el resultado final del proyecto. Para lograr una utilización exitosa de metodologías ágiles en proyectos de desarrollo de software, es importante identificar las causas de esta brecha y proponer soluciones efectivas para superarla.

El objetivo de esta investigación es evaluar y proponer soluciones para reducir la brecha entre lo teórico y lo práctico en la utilización de metodologías ágiles con el fin de mejorar la eficiencia del equipo y la calidad del producto final. Para ello se propone un estudio de caso en la empresa de desarrollo de software “Agility”, con un enfoque cualitativo utilizando como método inductivo el diseño de “teoría fundamentada”. Como diseño adicional se plantea investigación-acción ya que se realizan propuestas que ayuden a la empresa a resolver las dificultades que surjan del análisis.

La recolección de datos se sustenta en entrevistas semi-estructuradas a “project managers” de la empresa Agility, observación de los procesos realizados participando de eventos principales que ofrece la metodología y revisión documental sobre los resultados de los proyectos en la empresa.



Del análisis de los resultados, se observa la importancia de reconocer y abordar esta brecha en las empresas que adoptan metodologías ágiles. Esto implica proporcionar una formación continua y adaptada a las necesidades del contexto empresarial, fomentando una cultura de colaboración y aprendizaje. Finalmente se destaca que si bien, cada proyecto tiene sus propias características y puede requerir adaptaciones en la metodología ágil utilizada, es fundamental no perder de vista los fundamentos y principios básicos. Mantener un enfoque coherente y sólido en las prácticas ágiles permitirá aprovechar al máximo los beneficios que estas ofrecen.

**Palabras claves:** Metodologías ágiles – Brecha entre lo teórico y lo práctico – Scrum – Kanban.



## Introducción

El sector TIC, especialmente del software, es uno de los grandes impulsores de la economía mundial. La importancia del software en todos los aspectos de la vida cotidiana lo hace merecedor de invertir esfuerzos en investigación sobre cómo mejorar su proceso de desarrollo.

Desafortunadamente gestionar este tipo de proyectos no es fácil: el software posee propiedades que lo hacen particularmente complejo. Los métodos clásicos o prescriptivos de gestión no se adaptan al entorno del software que se caracteriza por los continuos cambios e incertidumbre. Por lo tanto, profundizar en las Metodologías Ágiles como alternativa a los métodos actuales de gestión, es una oportunidad para innovar en los procesos de gestión, aumentar la productividad y obtener mejores resultados en los proyectos software.

Las metodologías ágiles se definen como un conjunto de tareas y procedimientos dirigidos a la gestión de proyectos; permiten adaptar la forma de trabajo a las condiciones del proyecto. Surgieron a principios de la década de 2000 como una respuesta a los enfoques tradicionales de gestión de proyectos, que se consideraban demasiado rígidos y burocráticos para el desarrollo de software. En 2001, un grupo de expertos en desarrollo de software crearon el Manifiesto Ágil, un conjunto de valores y principios que se centran en la entrega continua de software de alta calidad, la colaboración entre los miembros del equipo y la adaptabilidad a los cambios en los requisitos del proyecto. Desde entonces, las metodologías ágiles se han convertido en una alternativa popular a los enfoques tradicionales de gestión de proyectos.

Molina E. (2015) destaca que la principal particularidad de las metodologías ágiles es la flexibilidad, los proyectos en desarrollo son subdivididos en proyectos más pequeños, incluye una comunicación constante con el usuario, son altamente colaborativos y es mucho más



adaptable a los cambios.

La adopción de dichas metodologías en una empresa puede generar una brecha entre la teoría y la práctica. En teoría, el agilismo promueve la entrega continua de software de alta calidad, la colaboración entre los miembros del equipo y la capacidad de adaptarse a los cambios en los requisitos del proyecto. Sin embargo, en la práctica, pueden surgir desafíos que dificultan su implementación efectiva.

Esto hace aún más relevante la importancia de una correcta aplicación. Al reconocer y abordar esta brecha, las empresas pueden mitigar los riesgos asociados con una implementación inadecuada y optimizar los beneficios que estas metodologías ofrecen.

### **Problema**

Las metodologías ágiles desempeñan un papel crucial en el ámbito de las tecnologías de la información (IT), ya que permiten a los equipos de desarrollo de software y proyectos adaptarse a los cambios del entorno y a las necesidades del cliente. La utilización de metodologías ágiles en proyectos de desarrollo de software ha demostrado ser altamente efectiva para mejorar la calidad del trabajo y aumentar la productividad.

Sin embargo, la adopción de dichas metodologías en la empresa bajo estudio, a pesar de la formación teórica recibida en los equipos, muestra discrepancias en la realidad empresarial, lo que puede tener un impacto significativo en el resultado final del proyecto.

### **Preguntas de Investigación**

- ¿Cuáles son las principales diferencias entre la teoría y la práctica en la utilización de metodologías ágiles en la empresa Agility?
- ¿Cuáles son las principales causas de la brecha teórica-práctica en la utilización de metodologías ágiles en la empresa bajo estudio?



- ¿Cómo afecta dicha brecha en la utilización de metodologías ágiles a la calidad del software y la eficiencia del equipo de desarrollo?
- ¿Qué soluciones se pueden proponer para reducir esta brecha y mejorar la eficiencia en la utilización de las metodologías ágiles?

### **Objetivo General**

Evaluar y proponer soluciones para reducir la brecha entre lo teórico y lo práctico en la utilización de metodologías ágiles en la empresa Agility, con el fin de mejorar la eficiencia del equipo y la calidad del producto final.

### **Objetivos Específicos**

-Analizar la brecha entre lo teórico-conceptual y lo práctico en la utilización de metodologías ágiles en la empresa Agility.

-Identificar las principales causas de la brecha entre lo teórico y lo práctico en la utilización de metodologías ágiles en la empresa bajo estudio.

-Evaluar el impacto de la brecha mencionada en la calidad del software y la eficiencia del equipo de desarrollo en Agility.

### **Marco Teórico**

El desarrollo de software no es una disciplina sencilla. En las últimas décadas los e varias herramientas intentaron sin éxito posicionarse como solución eficaz para resolver algunos de sus problemas. No fue sino hasta la adopción amplia y consciente de las metodologías de desarrollo, que se encontraron soluciones adecuadas a muchos problemas. La mayoría de estas



metodologías fueron introducidas desde la ingeniería civil, lo que resultó en un exhaustivo control sobre los procesos y las tareas.

### Modelo secuencial de procesos

Según Alaimo (2013) **El Modelo Secuencial de Procesos**, también conocido como Waterfall Model o Modelo en Cascada, se convirtió en el modelo metodológico más utilizado dentro de la industria. Data de principios de los años setenta y tiene sus orígenes en los ámbitos de la manufactura y la construcción, ambientes físicos altamente rígidos donde los cambios se vuelven prohibitivos desde el punto de vista de los costos, sino prácticamente imposibles. Como no existía proceso alguno en la industria del software, esta condición no impidió su adopción.

El proceso Waterfall sugiere una evolución secuencial. Por ejemplo: primero se realiza la fase de especificación de requerimientos. Una vez que se encuentra completa se procede a un "sign-off" (firma/aprobación) que congela dichos requerimientos, y es recién aquí cuando se inicia la fase de diseño del software, fase donde se produce un plano o "blueprint" del mismo para que los programadores lo implementen. Hacia el final de la fase de implementación, diferentes componentes desarrollados son integrados con el fin de pulir las interfaces entre ellos. El siguiente paso es la fase de verificación en la que los testers someten el sistema a diferentes tipos de pruebas funcionales mientras los programadores corrigen el código donde sea necesario. Una vez que el sistema responde satisfactoriamente a la totalidad de las pruebas, se inicia una etapa de instalación y mantenimiento posterior.

Los problemas detectados en los modelos tradicionales o de tipo Waterfall se fundamentan, por un lado, en el entorno altamente cambiante propio de la industria, y por el otro, en el proceso mismo de desarrollo de software donde el resultado depende de la actividad cognitiva de las personas más que de las prácticas y controles empleados.





A medida que han pasado los años, y con el advenimiento de las economías globalizadas y los entornos web, el contexto de negocio de los sistemas ha pasado de ser relativamente estable a convertirse en un contexto altamente volátil, donde los requerimientos expresados hoy, en muy pocas oportunidades son válidos unos meses más tarde. Bajo esta nueva realidad, las metodologías Waterfall resultaron muy “pesadas” y prohibitivas para responder satisfactoriamente a los cambios de negocio

Bajo este contexto surgieron nuevas metodologías, como, por ejemplo:

- Metodologías en Espiral
- Metodologías Iterativas
- Metodologías Ágiles

Tanto las Metodologías en Espiral como las Metodologías Iterativas se encuentran fuera del alcance de este trabajo, por lo que pasaremos directamente a entender las Metodología Ágiles.

### **Explorando el mundo de las “Metodologías Ágiles”**

Las **metodologías ágiles** se definen como un conjunto de tareas y procedimientos dirigidos a la gestión de proyectos; permiten adaptar la forma de trabajo a las condiciones del proyecto. Es importante destacar que “Agile” es mucho más que una metodología, es una filosofía que supone una forma distinta de trabajar y de organizarse.

Molina E. (2015) destaca que la principal particularidad de las metodologías ágiles es la flexibilidad, los proyectos en desarrollo son subdivididos en proyectos más pequeños, incluye una comunicación constante con el usuario, son altamente colaborativos y es mucho más adaptable a los cambios.



Una característica importante en el uso de las metodologías ágiles es que no se necesita definir, al inicio de los proyectos, la totalidad del alcance del mismo; lo que se convierte en una ventaja debido a la flexibilidad en la ejecución del proyecto.

Asimismo, el resultado final de la aplicación de las metodologías ágiles es un producto o proyecto que satisfaga las necesidades de los clientes y que se ha elaborado con costos bajos, mínimos desechos y en menor tiempo (Kurup y Kala, 2015).

### **Manifiesto Ágil**

En el año 2001, se reunieron en Utah (EEUU) un grupo de diecisiete profesionales reconocidos del desarrollo de software, y referentes de las metodologías livianas existentes al momento, con el objetivo de determinar los valores y principios que les permitirían a los equipos desarrollar software de forma más acertada con las necesidades del cliente y responder mejor a los cambios que pudieran surgir a lo largo de un proyecto de desarrollo. Se pretendía ofrecer una alternativa a los procesos de desarrollo de software tradicionales, caracterizados por la rigidez y dominados por la documentación. También se declaró la piedra angular del movimiento ágil, conocida como **Manifiesto Ágil**, el cual se compone de 4 valores y 12 principios:

Los cuatro valores del manifiesto agile son:

- **Individuos e interacciones por encima de procesos y herramientas:** La agilidad propone crear el equipo y que éste construya su propio entorno y procesos en base a sus necesidades.
- **Software que funciona por encima de excesiva documentación:** Tradicionalmente, se genera una gran cantidad de documentación. Es más importante gastar tiempo y energías en un software que funcione y que poder entregar al cliente. La regla a seguir



es "no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante".

- **Colaboración con el cliente por encima de las negociaciones:** La relación entre el cliente y el equipo de desarrollo debe ser fluida, es importante generar un ambiente de confianza. Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo.
- **Respuesta al cambio por encima del seguimiento de un plan establecido:** Es necesario que el equipo esté preparado para los cambios que puedan surgir durante el proceso de desarrollo de software. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Figura 1: Manifiesto por el Desarrollo Ágil de Software



Fuente: Recuperado de <http://agilemanifesto.org/iso/es/manifesto.html>



### Principios del manifiesto agile

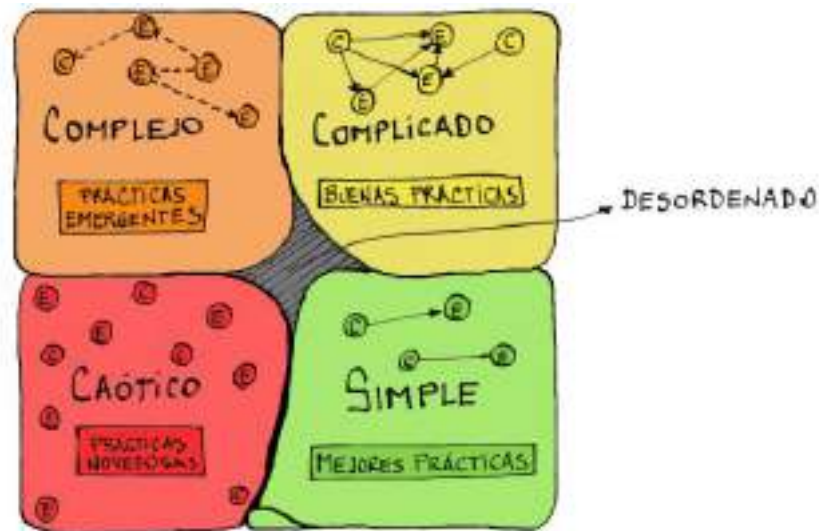
Los 12 principios incluidos en el manifiesto, surgen de los valores descritos:

1. La satisfacción del “cliente” mediante tempranas y continuas entregas de software
2. Aceptar de forma positiva los cambios y otorgar una ventaja competitiva
3. Dividir el desarrollo en entregas frecuentes
4. Posibilidad de retroalimentación entre el equipo y el cliente durante el proyecto
5. Individuos motivados. Proporcionar el entorno y apoyo necesario y confiar en ellos.
6. El diálogo cara a cara como método para comunicar información dentro de un equipo de desarrollo
7. El software que funciona es la medida principal de progreso
8. Los procesos ágiles promueven un desarrollo sostenible. Promotores, desarrolladores y usuarios deben ser capaces de mantener una paz constante
9. La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
10. La simplicidad es esencial
11. Las mejores arquitecturas, requisitos y diseños surgen de los equipos auto organizados
12. Regularmente, el equipo reflexiona sobre cómo llegar a ser más efectivo.

Alaimo (2013) propone que una de las formas posiblemente más claras para explicar el contexto en el cual Scrum es más eficiente tiene que ver con el enfoque de la complejidad del Marco Cynefin o **Modelo Cynefin** acuñado en 2000 por Dave Snowden, el cual compara las características de cinco dominios de complejidad diferentes: simple, complicado, complejo, caótico y desordenado, en el centro.

Modelo Cynefin

Figura 2 : Dimensiones del Modelo Cynefin



Fuente: Alaimo (2013). Proyectos ágiles con Scrum.

A continuación, se presentan las definiciones realizadas por Snowden (2000) de cada dominio:

**Dominio simple:** Dave Snowden lo define como un dominio en donde es muy fácil identificar las causas y sus efectos. Por lo general, la respuesta correcta es clara, conocida por todos e indiscutible. En este dominio existen las mejores prácticas, soluciones conocidas para problemas conocidos.

**Dominio complicado:** En este dominio encontramos problemas complejos, buenas prácticas y perfiles expertos. Hay múltiples soluciones correctas para una misma problemática, pero se requiere del involucramiento de expertos para poder identificarlas.



**Dominio complejo:** Nos encontramos con un Dominio Complejo cuando nos enfrentamos a problemas complejos, cuyos resultados se vuelven más impredecibles. No existen ni mejores ni buenas prácticas catalogadas para las situaciones frente a las cuales nos podemos encontrar.

**Dominio caótico:** Para Dave Snowden, nos movemos en el espacio desordenado cuando no sabemos en qué dominio estamos. En el Modelo Cynefin a esta zona la clasifica como una zona peligrosa, ya que no podemos medir las situaciones ni determinar la forma de actuar

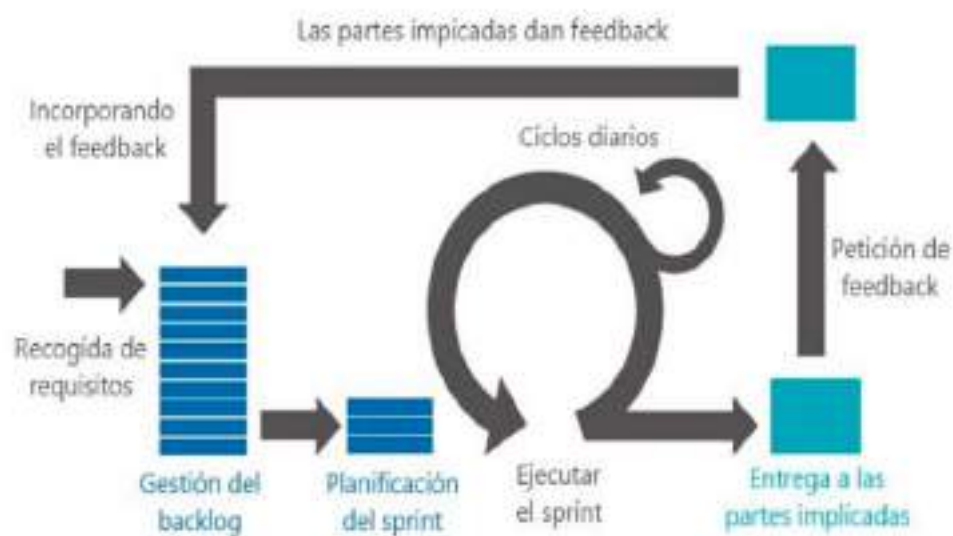
**Dominio desordenado:** Nos movemos en el espacio desordenado cuando no sabemos en qué dominio estamos! Se la clasifica como una zona peligrosa, ya que no podemos medir las situaciones ni determinar la forma de actuar. El gran peligro del dominio desordenado es actuar de manera diferente a la que se necesita para resolver ciertos problemas.

### **Y entonces, ¿Qué es scrum?**

Alaimo (2013) define **Scrum** como un marco de trabajo que nos permite encontrar prácticas emergentes en dominios complejos, como la gestión de proyectos de innovación. No es un proceso completo, y mucho menos, una metodología. En lugar de proporcionar una descripción completa y detallada de cómo deben realizarse las tareas de un proyecto, genera un contexto relacional e iterativo, de inspección y adaptación constante para que los involucrados vayan creando su propio proceso. Esto ocurre debido a que no existen ni mejores ni buenas prácticas en un contexto complejo. Es el equipo de involucrados quien encontrará la mejor manera de resolver sus problemáticas. Este tipo de soluciones serán emergentes.

Schwaber y Sutherland (2020) proponen que Scrum se basa en el **empirismo** y el **pensamiento Lean**. El empirismo afirma que el conocimiento proviene de la experiencia y de la toma de decisiones con base en lo observado. El pensamiento Lean reduce el desperdicio y se enfoca en lo esencial. Scrum emplea un enfoque iterativo e Incremental para optimizar la previsibilidad y controlar el riesgo. También involucra a grupos de personas que colectivamente tienen todas las habilidades y experiencia para hacer el trabajo y compartir o adquirir dichas habilidades según sea necesario.

**Figura 3: Proceso Scrum**



Fuente: Recuperado de la web

Esta metodología, combina eventos formales para inspección y adaptación dentro de un evento contenedor, el **Sprint**. Estos eventos funcionan porque implementan los pilares empíricos de Scrum de transparencia, inspección y adaptación.

### Transparencia



El proceso y el trabajo emergentes deben ser visibles tanto para quienes realizan el trabajo como para quienes lo reciben. La transparencia permite la inspección. La inspección sin transparencia es engañosa y derrochadora.

### **Inspección**

Los artefactos de Scrum y el progreso hacia los objetivos acordados deben inspeccionarse con frecuencia para detectar variaciones o problemas potencialmente indeseables. La inspección permite la adaptación. La inspección sin adaptación se considera inútil. Los eventos Scrum están diseñados para provocar cambios.

### **Adaptación**

Si algún aspecto de un proceso se desvía fuera de los límites aceptables o si el producto resultante es inaceptable, el proceso que se aplica o los materiales que se producen deben ajustarse. El ajuste debe realizarse lo antes posible para minimizar una mayor desviación. La adaptación se vuelve más difícil cuando las personas involucradas no están empoderadas ni se autogestionan. Se espera que un Scrum Team se adapte en el momento en que aprenda algo nuevo a través de la inspección.

A continuación, se presentan los diferentes roles definidos por Alaimo (2013) que integran el "Scrum Team"

### **Equipo de desarrollo**

El equipo de desarrollo está formado por todos los individuos necesarios para la construcción del producto en cuestión. Es el único responsable por la construcción y calidad del producto.





El equipo de desarrollo es auto-organizado. Esto significa que no existe un líder externo que asigne las tareas ni que determine la forma en la que serán resueltos los problemas. Es el mismo equipo quien determina la forma en que realizará el trabajo y cómo resolverá cada problemática que se presente. La contención de esta auto-organización está dada por el objetivo a cumplir: transformar las funcionalidades comprometidas en software funcionando y con calidad productiva, o en otras palabras, producir un incremento funcional potencialmente entregable.

Es recomendable que un equipo de desarrollo se componga de hasta nueve personas. Cada una de ellas debe poseer todas las habilidades necesarias para realizar el trabajo requerido. Esta característica se conoce como multi-funcionalidad y significa que dentro del equipo de desarrollo no existen especialistas exclusivos, sino más bien individuos generalistas con capacidades especiales.

El equipo de desarrollo tiene tres responsabilidades tan fundamentales como indelegables. La primera es proveer las estimaciones de cuánto esfuerzo será requerido para cada una de las características del producto. La segunda responsabilidad es comprometerse al comienzo de cada Sprint a construir un conjunto determinado de características en el tiempo que dura el mismo. Y finalmente, también es responsable por la entrega del producto terminado al finalizar cada Sprint.

### **Product Owner**

Se focaliza en maximizar la rentabilidad del producto. La principal herramienta con la que cuenta para poder realizar esta tarea es la priorización. De esta manera puede reordenar la cola de trabajo del equipo de desarrollo para que éste construya con mayor anticipación las características o funcionalidades más requeridas por el mercado o la competitividad comercial



Otra responsabilidad importante del Product Owner es la gestión de las expectativas de los stakeholders mediante la comprensión completa de la problemática de negocio y su descomposición hasta llegar al nivel de requerimientos funcionales.

### **Scrum Master**

El ScrumMaster es el Coach del equipo y es quien lo ayuda a alcanzar su máximo nivel de productividad posible. Tomando algunas referencias de Leonardo Wolk (2003) podemos decir que el ScrumMaster, en tanto que coach, es un líder, facilitador, provocador, detective y soplador de brasas.

Se espera, además, que el ScrumMaster acompañe al equipo de trabajo en su día a día y garantice que todos, incluyendo al Product Owner, comprendan y utilicen Scrum de forma correcta. Su responsabilidad es asegurar que se cumpla con el proceso de Scrum sin interferir directamente en el desarrollo del producto final. Es importante establecer que el equipo de Scrum elige la forma de trabajo que más prefiera, siempre que se cumplan las pautas básicas de Scrum, por ello mientras lo hagan no existe una forma “errónea” de trabajar.

Debe detectar problemas y conflictos interpersonales dentro del equipo de trabajo. Para respetar la filosofía auto-organizativa del equipo, lo ideal es que el equipo mismo sea quien resuelva estas cuestiones.

El rol del Scrum Master también incluye asegurar que el desarrollo del producto tenga la mayor probabilidad de ser completado de forma exitosa. Para lograr este cometido, trabaja de cerca con el Product Owner asegurando una correcta priorización de los requerimientos, por un lado, y con el equipo de desarrollo para convertir los requerimientos en un producto funcionando, por el otro.



### ¿A qué nos referimos con stakeholders?

El concepto de **stakeholders** fue acuñado por el filósofo y profesor de administración empresarial estadounidense Robert Edward (1984), define el concepto de stakeholders o grupos de interés como “cualquier individuo u organización que, de alguna manera, es impactado por las acciones de determinada empresa”.

Asimismo, un grupo de investigadores de la Universidad Wharton, de Pennsylvania, en su teoría Gestión empresarial basada en los Stakeholders, orientada a la acción, sostiene que los stakeholders son “aquellos grupos o individuos que pueden influir sobre la consecución de los objetivos de una organización o verse afectados por ella”.

En definitiva, a partir de ambas definiciones, podemos establecer que los stakeholders o grupos de interés son aquellos actores que tienen algún tipo de relación con una empresa; de manera que cualquiera de las decisiones estratégicas de la compañía puede afectarles de forma directa o indirecta.

Los **elementos de Scrum** son componentes fundamentales de este marco de trabajo ágil y se utilizan para gestionar y controlar el desarrollo de proyectos. Los elementos principales de Scrum definidos por Alaimo (2013) son los siguientes:

#### Elementos de un scrum

##### a) **Product Backlog**

El Backlog del Producto es básicamente un listado de ítems (PBI) o características del producto a construir, mantenido y priorizado por el Product Owner. Es importante que exista una clara priorización, ya que la misma determinará el orden en el que el equipo de desarrollo transformará las características (ítems) en un producto funcional acabado.



Esta prioridad es responsabilidad exclusiva del Product Owner y, aunque el equipo de desarrollo pueda hacer sugerencias o recomendaciones, es el Product Owner quién tiene la última palabra sobre la prioridad final de los ítems.

- **Priorización por valor de negocio de cada PBI**

Una forma de priorizar los ítems del Product Backlog es según su valor de negocio. Podemos entender el valor de negocio como la relevancia que un ítem tiene para el cumplimiento del objetivo de negocio que estamos buscando.

- **Priorización por retorno de la inversión (ROI) de cada PBI**

Un enfoque diferente de medir la prioridad de un determinado ítem del Backlog es calcular el beneficio económico que se obtendrá en función de la inversión que se deba realizar

- **Prioridades según la importancia y el riesgo de cada PBI**

Ya sea que los ítems del Backlog se prioricen por valor de negocio o por ROI, en cualquier caso llamémosle “priorizar por importancia”, éstos pueden verse complementariamente afectados por el nivel de riesgo asociado a cada uno de ellos. De esta manera, deberíamos aprovechar la construcción iterativa y evolutiva de Scrum para mitigar riesgos en forma implícita: construyendo primero aquellas características con mayor riesgo asociado y dejando las que poseen menor riesgo para etapas posteriores.

Cuando hablamos de eficiencia, hablamos de obtener el mayor beneficio con el menor esfuerzo posible. Este concepto llevado al Product Backlog significa invertir el esfuerzo de exploración y especificación de la manera más inteligente posible para evitar retrabajos y desperdicios. Por esto, fomentamos un Product Backlog donde sus ítems más prioritarios están expresados con un nivel de detalle mucho mayor que los ítems de menor prioridad, los cuales

están descritos a un nivel más alto, ya que son los más susceptibles de ser alterados o reemplazados.

Alaimo (2013) Hace referencia al “alcance variable” del product backlog en función a que asumimos que no es posible conocer de manera anticipada y con un nivel muy fino de detalle todas las características del producto que pretendemos construir, sino que es un viaje de descubrimiento que emprendemos junto con el cliente, este Backlog es un elemento vivo que muta a través del tiempo, al ritmo que vamos aprendiendo sobre el mismo con las entregas iterativas y el feedback frecuente.

Si bien, tradicionalmente, el alcance se ha intentado fijar desde el comienzo de un proyecto, y así manejar el costo y el tiempo como los elementos variables, desde la agilidad, esta ecuación se invierte: el tiempo y el costo son los componentes fijos del proyecto, mientras que el alcance es el componente variable.

Figura 4: Triángulo de Hierro



Fuente: Alaimo (2013) Proyectos ágiles con Scrum.



## b) **Sprint (Iteración)**

Las iteraciones en Scrum se conocen como Sprints. Scrum, como todos los enfoques ágiles, es un proceso de desarrollo incremental e iterativo. Esto significa que el producto se construye en incrementos funcionales entregados en periodos cortos para obtener feedback frecuente.

En general, Scrum recomienda una duración de Sprint de entre 1 y 4 semanas, siendo 2 o 3 semanas lo más habitual que encontraremos en la industria. Una de las decisiones que debemos tomar al comenzar un proyecto o al adoptar Scrum es justamente la duración de los Sprints.

Muchas veces podemos encontrar situaciones en donde el equipo de desarrollo se atrase o se adelante. En estos casos, la regla del timeboxing no nos permitirá modificar (adelantar o postergar) la fecha de entrega o finalización del Sprint. La variable de ajuste en estos casos será el alcance del Sprint, esto es, en el caso de adelantarnos deberemos incrementar el alcance del Sprint agregando nuevos PBIs y reducirlo en el caso de retrasarnos.

## c) **Sprint Backlog**

El Sprint Backlog es el conjunto de PBIs que fueron seleccionados para trabajar en ellos durante un cierto Sprint, conjuntamente con las tareas que el equipo de desarrollo ha identificado que debe realizar para poder crear un incremento funcional potencialmente entregable al finalizar el Sprint.

**Incremento funcional** porque es una característica funcional nueva (o modificada) de un producto que está siendo construido de manera evolutiva. El producto crece con cada Sprint.



**Potencialmente entregable** porque cada una de estas características se encuentra lo suficientemente validada y verificada como para poder ser desplegada en producción (o entregada a usuarios finales) si así el negocio lo permite o el cliente lo desea.

#### **d) Sprint Planning Meeting (Planificación de Sprint)**

Al comienzo de cada Sprint se realiza una reunión de planificación del Sprint donde serán generados los acuerdos y compromisos entre el equipo de desarrollo y el Product Owner sobre el alcance del Sprint. Esta reunión de planificación habitualmente se divide en dos partes con finalidades diferentes: una primera parte estratégica y enfocada en el “qué”, y una segunda parte táctica cuyo hilo conductor principal es el “cómo”.

Podríamos decir que se trata de un taller donde el Product Owner expone todos y cada uno de los PBIs que podrían formar parte del Sprint, mientras que el equipo de desarrollo realiza todas las preguntas que crea necesarias para conocer sus detalles y así corroborar o ajustar sus estimaciones.

El objetivo buscado durante esta parte de la reunión es identificar “qué” es lo que el equipo de desarrollo va a realizar durante el Sprint, es decir, todos aquellos PBIs que el equipo se comprometerá a transformar en un producto funcionando y utilizable.

Durante este espacio de tiempo el equipo de desarrollo determinará la forma en la que llevará adelante el trabajo. Esto implica la definición inicial de un diseño de alto nivel, el cual será refinado durante el Sprint mismo y la identificación de las actividades que el equipo en su conjunto tendrá que llevar a cabo. Es recomendable que las actividades duren idealmente menos de un día. Esto permitirá detectar bloqueos o retrasos durante las reuniones diarias



**e) Scrum Diario/Daily meeting**

Los siguientes objetivos: 1) incrementar la comunicación 2) explicitar los compromisos y 3) dar visibilidad a los impedimentos, son logrados mediante las reuniones diarias de Scrum. Estas reuniones tienen, como su nombre lo indica, una frecuencia diaria y no deberían llevar más de 15 minutos. Estos 15 minutos son un timebox, es decir, que no se pueden superar.

Esta reunión es facilitada por el ScrumMaster. Todos y cada uno de los miembros toman turnos para responder las siguientes tres preguntas, y de esa manera comunicarse entre ellos:

1. ¿Qué hice desde la última reunión diaria hasta ahora?
2. ¿En qué voy a estar trabajando desde ahora hasta la próxima reunión diaria?
3. ¿Qué problemas o impedimentos tengo?

Es importante destacar que en ningún momento se trata de una reunión de reporte de avance o status al ScrumMaster ni a otras personas. Por el contrario, es un espacio de estricta comunicación entre los miembros del equipo de desarrollo.

**f) Revisión de Sprint**

Al finalizar cada Sprint se realiza una reunión de revisión del Sprint, donde se evalúa el incremento funcional potencialmente entregable construido por el equipo de desarrollo (el “qué”). En esta reunión el Equipo Scrum y los Stakeholders revisan el resultado del Sprint.

Los Stakeholders evalúan el producto construido y proveen feedback. Este feedback puede ser acerca de cambios en la funcionalidad construida o bien nuevas funcionalidades que surjan al ver el producto en acción. El Product Owner cuenta para esto con la priorización de los ítems del Backlog como herramienta para la toma de este tipo de decisiones.





#### g) Retrospectiva

En un método empírico como Scrum, la retrospectiva del equipo es el corazón de la mejora continua y las prácticas emergentes. Mediante el mecanismo de retrospectiva, el equipo reflexiona sobre la forma en la que realizó su trabajo y los acontecimientos que sucedieron en el Sprint que acaba de concluir para mejorar sus prácticas. Todo esto sucede durante la reunión de retrospectiva.

Esta reunión tiene lugar inmediatamente después de la reunión de revisión. Mientras que la reunión de revisión se destina a revisar el producto (el “qué”), la retrospectiva se centra en el proceso (el “cómo”).

#### h) Refinamiento del Product Backlog

El refinamiento del Backlog es una actividad constante a lo largo de todo el Sprint, aunque algunos equipos prefieren concentrarse en una reunión que se realiza durante el Sprint y en función de las necesidades. Su objetivo es profundizar en el entendimiento de los PBIs que se encuentran más allá del Sprint actual y así dividirlos en PBIs más pequeños, si lo requieren, y estimarlos. Idealmente se revisan y detallan aquellos que potencialmente se encuentren involucrados en los próximos dos o tres Sprints.

La participación de todo el Equipo Scrum es esencial para el éxito de esta reunión. Sin ellos, esta actividad no tendría sentido.

#### Scrum como “Desarrollo Evolutivo”

En un desarrollo iterativo e incremental el proyecto se planifica en diversos bloques temporales llamados **iteraciones**. Las iteraciones se pueden entender como pequeños proyectos: en todas las iteraciones se repite un proceso de trabajo similar para proporcionar un

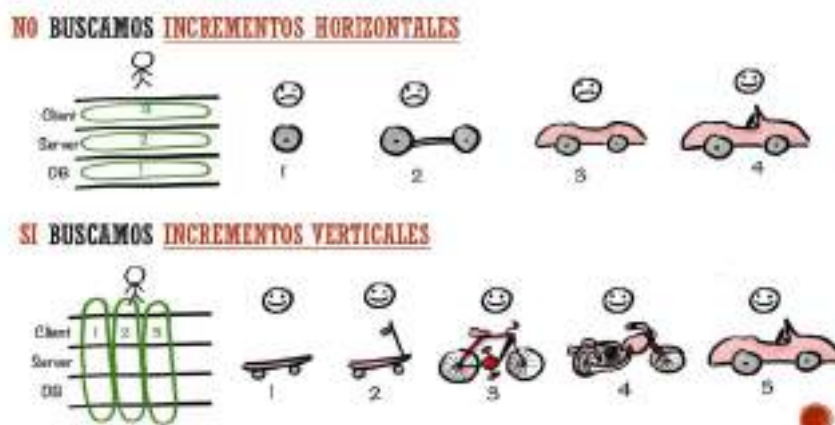
resultado completo sobre producto final, de manera que el cliente pueda obtener los beneficios del proyecto de forma incremental.

Para ello, cada requisito se debe completar en una única iteración: el equipo debe realizar todas las tareas necesarias para completarlo (incluyendo pruebas y documentación) y que esté preparado para ser entregado al cliente con el mínimo esfuerzo necesario. De esta manera no se deja para el final del proyecto ninguna actividad arriesgada relacionada con la entrega de requisitos.

En cada iteración el equipo evoluciona el producto (hace una entrega incremental) a partir de los resultados completados en las iteraciones anteriores, añadiendo nuevos objetivos/requisitos o mejorando los que ya fueron completados. Un aspecto fundamental para guiar el desarrollo iterativo e incremental es la priorización de los objetivos/requisitos en función del valor que aportan al cliente.

En la figura 5, se presenta una analogía visual sobre el desarrollo de un auto para facilitar la comprensión del **desarrollo evolutivo** que se lleva a cabo en la metodología scrum.

Figura 5: Analogía de desarrollo ágil



Fuente: Material Bibliográfico Catedra Análisis y Diseño de Sistemas. UNT



Partiendo de esta base, vamos a introducir dos conceptos complementarios entre sí:

Minimum Viable Product, Minimum Marketable Feature.

#### - **Minimum Viable Product**

El Minimum Viable Product (MVP) es la versión mínima de un producto, tal que nos permita recolectar la mayor cantidad de información de nuestro mercado y clientes con el menor esfuerzo posible. Consiste en hacer foco en las características mínimas y necesarias para que el producto pueda lanzarse al mercado. Esto nos permitirá:

- Evitar crear productos que nadie necesita
- Maximizar el aprendizaje por dólar invertido

#### - **Minimum Marketable Features**

Todas las metodologías ágiles coinciden en que un producto debe construirse de forma evolutiva en pequeñas entregas. No es suficiente dividir el producto en tres o cuatro entregas sucesivas, sino que debemos hacerlo de forma criteriosa para que cada entrega pueda aportar valor suficiente a los usuarios finales. Esos grupos de características se denominan MMF: Minimum Marketable Features, y pueden definirse como “el conjunto más pequeño posible de funcionalidad que, por sí misma, tiene valor en el mercado”

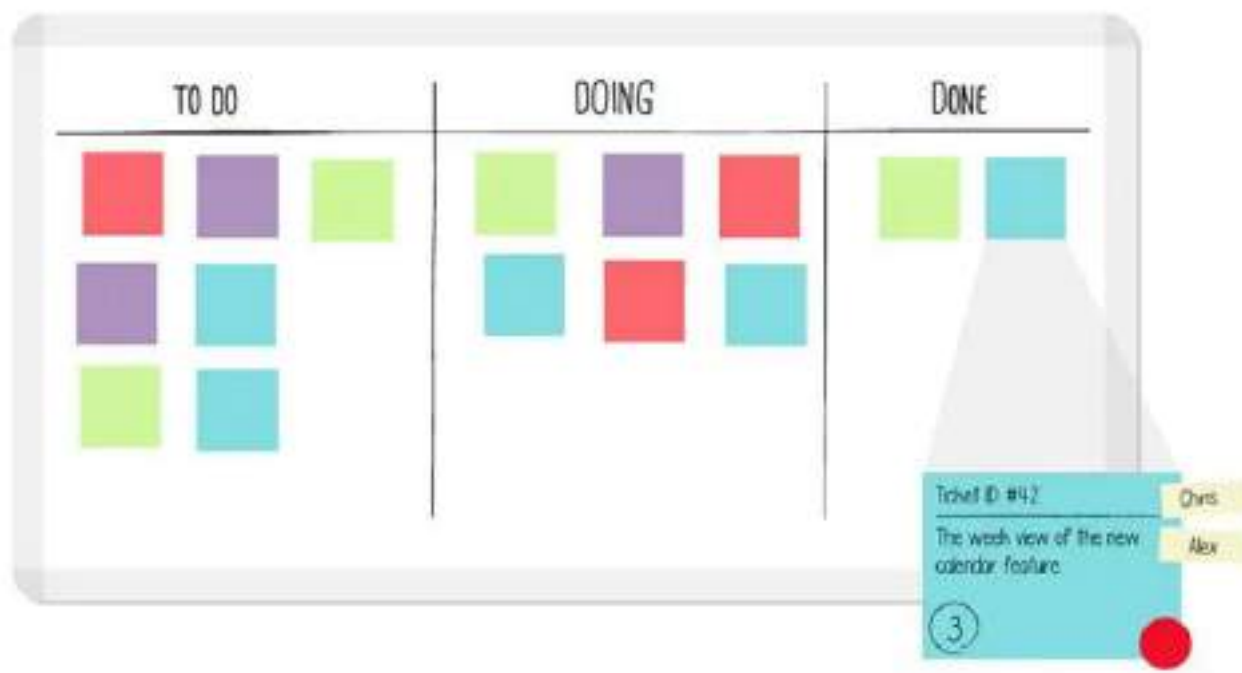
### **Metodología ágil Kanban**

La metodología ágil Kanban es una técnica visual de gestión de proyectos que se utiliza en el desarrollo de software para mejorar la eficiencia y la productividad del equipo. Kanban se originó en la industria manufacturera japonesa y se ha adaptado para su uso en el desarrollo de software.



Esta metodología se basa en la idea de que el trabajo debe fluir de manera constante y sin interrupciones. Para lograr esto, se utiliza un tablero Kanban que muestra el estado de las tareas del proyecto. El tablero se divide en columnas que representan las etapas del proceso, y cada tarea se representa con una tarjeta, como se puede observar en la figura 6.

Figura 6: Ejemplo de tablero Kanban



Fuente: Recuperado de <https://abantian.es/blog/tablero-kanban-sistema-visual-organizar-gestionar-las-tareas-del-equipo/>

Esta técnica está centrada en la mejora continua y en la eliminación de los cuellos de botella en el proceso de desarrollo de software, se utilizan límites de trabajo en progreso que limitan el número de tareas que se pueden trabajar en cada etapa del proceso. Esto ayuda a evitar la sobrecarga de trabajo y a mantener un flujo constante de trabajo.



En el uso de Kanban es fundamental la colaboración y la comunicación entre los miembros del equipo. Se fomenta la transparencia y la visibilidad del proceso de desarrollo de software para que todos los miembros del equipo puedan ver el estado de las tareas y colaborar en la resolución de problemas.

**Los principios fundamentales de Kanban**, también conocidos como los "Principios Kanban", son una serie de prácticas y conceptos clave que guían la implementación de la metodología Kanban. Estos principios, desarrollados por Anderson D.J. (2010) y la comunidad Kanban, se centran en mejorar el flujo de trabajo y la entrega continua de valor. A continuación, se presentan los siete principios Kanban:

#### **Los principios Kanban**

1. **Empezar con lo que hay:** Se parte del punto de partida actual, respetando el sistema y los procesos existentes. No se requiere un cambio completo o una reorganización radical para comenzar a implementar Kanban. Se trabaja con el sistema existente y se van realizando mejoras de forma gradual.

2. **Aceptar los cambios incrementales y evolutivos:** Los cambios se realizan de manera gradual y se basan en una comprensión profunda del sistema y del contexto. Se buscan mejoras paso a paso en lugar de realizar cambios drásticos y disruptivos.

3. **Respetar el proceso actual, los roles y las responsabilidades:** Se respeta y valora el trabajo y los roles existentes dentro del equipo y la organización. Kanban se enfoca en mejorar el flujo y la colaboración, sin cambiar de manera significativa los roles y responsabilidades existentes. A diferencia de otras metodologías ágiles, Kanban no tiene roles integrados.



4. **Fomentar el liderazgo en todos los niveles:** El liderazgo se fomenta en todos los niveles de la organización. Cada miembro del equipo puede ejercer liderazgo y tomar decisiones que beneficien al flujo de trabajo y al logro de los objetivos.

5. **Establecer políticas explícitas:** Se definen y comunican claramente las políticas y las reglas del proceso. Las políticas establecen las expectativas y los acuerdos sobre cómo se lleva a cabo el trabajo, permitiendo una comprensión común y una toma de decisiones más efectiva.

6. **Implementar sistemas de retroalimentación:** Se establecen mecanismos de retroalimentación para aprender y mejorar continuamente. El feedback es esencial para identificar áreas de mejora, resolver problemas y adaptar el sistema a medida que se obtiene más información.

7. **Mejorar colaborativamente:** La mejora continua es un esfuerzo colectivo. Se alienta a todos los miembros del equipo y a las partes interesadas a colaborar en la identificación y aplicación de mejoras en el proceso y el sistema en general.

También, el autor hace referencia a 5 prácticas básicas de la metodología que ayudan a guiar la mentalidad del equipo hacia una mejora continua al momento de abordar un flujo de trabajo y a lograr un crecimiento progresivo: los principios esenciales del marco kanban.

#### **Las 5 prácticas de la metodología Kanban**

1. **Visualizar el trabajo:** Consiste en representar visualmente el flujo de trabajo utilizando un tablero Kanban. El tablero se divide en columnas que representan las etapas del proceso, y se utilizan tarjetas para representar las tareas o elementos de



trabajo. Esto permite una comprensión clara y transparente de cómo fluye el trabajo a lo largo del proceso.

2. **Limitar el trabajo en curso (WIP):** Se establece un límite máximo para la cantidad de tareas que pueden estar en proceso en cada etapa del flujo de trabajo. Esto evita la sobrecarga y ayuda a mantener un flujo constante y equilibrado, evitando la acumulación excesiva de trabajo en alguna etapa.

3. **Gestionar el flujo:** Se busca lograr un flujo continuo y sin interrupciones del trabajo a lo largo del proceso. Se analizan los cuellos de botella y los puntos de bloqueo para identificar y abordar los obstáculos que afectan el flujo eficiente. El objetivo es optimizar el tiempo de entrega y reducir los tiempos de espera.

4. **Hacer que las políticas del proceso sean explícitas:** Se definen claramente las reglas, políticas y procedimientos para cada etapa del proceso. Esto proporciona transparencia y claridad sobre cómo se debe realizar el trabajo, así como las expectativas y los acuerdos del equipo.

5. **Mejora continua:** Se fomenta la búsqueda constante de mejoras en el proceso y en el sistema en general. Se alienta a los equipos a experimentar, aprender de los resultados y adaptar el proceso para obtener mejores resultados. La mejora continua es un principio fundamental de Kanban. Otros sistemas podrían funcionar bien junto con Kanban. Ya sea Scrum o alguna otra metodología, debes estar siempre dispuesto a colaborar, experimentar y desarrollar tus procesos si es necesario.



### Marco Metodológico

Para el desarrollo del trabajo se utilizará un **enfoque cualitativo** utilizando como método inductivo el diseño de **“teoría fundamentada”**, a partir de los procesos recolección y análisis de datos no numéricos se contrastan teorías existentes con emergentes y prácticas en la realidad empresarial. Como diseño adicional se plantea **investigación-acción** ya que se realizan propuestas que ayuden a la empresa a resolver las dificultades que surjan del análisis.

La recolección de datos se sustenta principalmente en entrevistas semi-estructuradas a “project managers” de la empresa Agility con el propósito de comprender los procesos de desarrollo utilizados, la adaptación de la metodología en proyectos con distintas características, los inconvenientes por los que atraviesan y su percepción de dichos problemas. También se recolectarán datos a partir de la observación de los procesos realizados en el contexto de las metodologías “Scrum” y “Kanban” participando de eventos principales que ofrece la metodología: “Sprint planning” y “daily meeting”. Se realizará revisión documental y bibliográfica sobre las metodologías bajo estudio.

Como técnicas de análisis de datos, a partir de las entrevistas se llevará a cabo un proceso de “codificación” en el cual se establecen categorías y subcategorías de los tópicos principales con el objetivo de encontrar, a partir de un análisis a profundidad, las brechas teoricas-practicas. También se comparan proyectos con distintas características que apliquen las metodologías mencionadas.







La mayor parte de los comentarios se centraron en los roles presentes en los equipos de trabajo. Cabe destacar que dos de los equipos migraron recientemente de metodologías “Kanban” a metodología “Scrum”, el tercer equipo trabaja con “Scrum” y el cuarto utiliza Kanban”.

Emergió un tema relevante en ambas entrevistas relacionado a su propio puesto de project manager. En la empresa Agility, se asigna a los manager el puesto de Scrum Master. Uno de los entrevistados, está en condiciones de cumplir con las características de dicho puesto de una manera más involucrativa, siendo que el otro entrevistado a causa de sus responsabilidades operativas y administrativas no puede realizarlo de igual manera.

Como complemento al puesto mencionado, la empresa añade un líder técnico y en caso de dos proyectos, un Business Analysis. El primero con el objetivo de acompañar, orientar y brindar soluciones al equipo de desarrollo en cuestiones técnicas y específicas del desarrollo de software. Puede haber también un líder técnico propuesto por el cliente ya que pone su perspectiva técnica para supervisar el trabajo del equipo. Por otro lado el BA, está en contacto con el cliente continuamente y analiza sus necesidades.

Otro aspecto que emergió es que el rol de “product owner” siempre es propuesto por el cliente y son los dueños del product backlog. En algunos casos el PO está muy involucrado en el proceso de desarrollo y en otros casos solo supervisa. En un caso particular, la empresa trabaja con un PO el cual no comparte la idea de integrar al equipo de desarrollo en las reuniones propuestas por la metodología “scrum”. Dichas reuniones se llevan a cabo con roles de alto nivel en el proyecto y repercute en el desconocimiento e incertidumbre de las tareas por parte del equipo al momento de planificar el sprint.



Ambos entrevistados consideran fundamental y determinante para la dinámica del proyecto el arreglo que se tenga con el cliente y lo que él mismo sepa sobre la metodología ya que en algunos casos, la modifican según sus criterios perdiendo así los beneficios que brinda “scrum”. Los tipos de contrato que maneja la empresa determinan la manera en que se trabaja en función a sus características, requisitos y obligaciones, limitando así la ejecución de las metodologías ágiles tal como las conocemos.

En base a lo relacionado con el equipo de desarrollo, se habló sobre su estructura e integrantes, su autogestión, los procesos por los que transitan durante un sprint, la comunicación entre los mismos y la estimación de esfuerzo de los tickets del product backlog, las cuales no en todos los proyectos las realizan los responsables según lo determinado por la teoría, es decir, el equipo de desarrollo.

También, emergieron cuestiones relacionadas al incremento de valor entregable en la finalización de un sprint, es uno de los equipos solo se realizan reuniones con el cliente para mostrar el trabajo realizado, la totalidad del producto se entrega al finalizar del contrato. Para dicho proyecto se utiliza un contrato llamado “fixed price” en el cual se establece un tiempo límite y un presupuesto determinado.

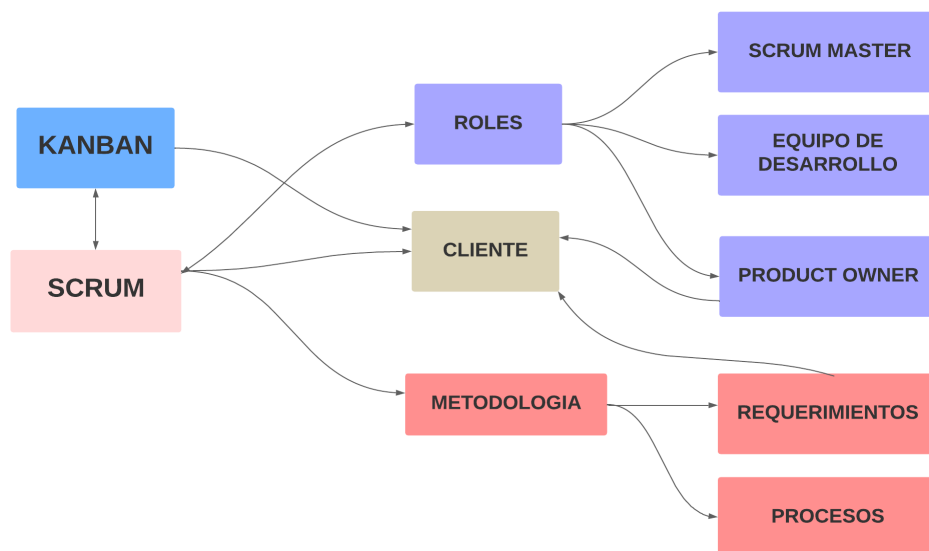
En base a la metodología ágil “Kanban”, utilizada por uno de los proyectos, no se encuentran diferencias significativas al momento de realizar la entrevista. Uno de los entrevistados menciona que se lo considera más sencillo, lo aplicaban sabiendo los conceptos básicos de la metodología y con eso funciona. Básicamente se basa en atender prioridades, estar bien informado, resolver bien el backlog, siempre tomar lo de mayor prioridad, tratando de resolver los bloqueos que surgen en el medio teniendo en cuenta la “urgencia” del trabajo siguiente, destaca la comunicación y la visualización.



Por otro lado, el entrevistado 2 afirma que el cliente prioriza los tickets para el equipo de desarrollo, les establecen una determinada cantidad posible de trabajo en proceso. Se comentan sobre los problemas que se enfrentaron anteriormente en la utilización de la metodología, como la falta de priorización por parte del cliente que quedaba en manos del equipo. También la falta de tickets en el tablero para iniciar un nuevo trabajo. A partir de estos inconvenientes se establecieron políticas, mejora en la comunicación, procesos y lineamientos que mejoraron la manera de trabajar y la utilización de kanban.

Luego de finalizar el análisis de la primera etapa de recolección de datos cualitativos y de su posterior análisis por medio de **técnicas de codificación**, se categorizaron los hallazgos encontrados provenientes de la brecha teórica-práctica en la utilización de metodologías ágiles en diferentes segmentos y sus subcategorías. A continuación se presenta un mapa cognitivo que resume las categorías identificadas tanto en el marco teórico como en la empresa bajo estudio y las relaciones entre las mismas:

Figura 8: Mapa cognitivo



Fuente: Elaboración propia mediante lucid app



En función a las categorías y subcategorías presentadas, se realizó un análisis exhaustivo de comparación entre la teoría sobre metodologías ágiles presentes en el marco teórico y la realidad práctica empresarial de los proyectos analizados utilizando comentarios textuales de los miembros de la empresa entrevistados. La finalidad de este proceso es encontrar los contrastes relevantes que tengan efectos negativos en los resultados del proyecto.

No se realizaron distinciones significantes entre los proyectos, ya que el objetivo es explorar y utilizar su diversidad para obtener una comprensión más profunda del problema objeto de estudio.

Se realizaron tablas para cada subcategoría, a continuación, se presentan las mismas:

### Comparaciones y contrastes

Tabla 1: Categoría: ROLES – Subcategoría: Product Owner

| CATEGORÍA: ROLES  |  |
|---|--|
| SUBCATEGORÍA: Product Owner   |  |
| TEORÍA  | REALIDAD EN AGILITY  |
| <b>Los roles en el equipo Scrum:</b><br><br><b>Product Owner:</b> Es el responsable desde el punto de vista del negocio | Existen los puestos de “líderes técnicos”, uno del lado de agility y el otro del lado del cliente. |
|   | La empresa adhiere a un rol de “Business Analysis”. Hace el análisis de lo que necesita el         |



|  |   |
|--|---|
| <p><b>Scrum Master:</b> Es el responsable de que el equipo sea productivo, ayudándole en todo momento a conseguir el objetivo acordado y asegurar que los principios se están respetando.</p> <p>El <b>equipo.</b> Es el responsable de la construcción del producto El es auto-organizado. Esto significa que no existe un líder externo que asigne las tareas ni que determine la forma en la que serán resueltos los problemas.</p> | <p>cliente y en base a eso se determinan los requerimientos.</p> <p>El product owner está en contacto sobre todo con nuestro Business Analysis.</p> |
| <p>El Product Owner es parte del Equipo Scrum y trabaja colaborativamente con el resto de los individuos dentro del equipo para asegurarse que el producto construido tenga la mayor cantidad posible de valor al final de cada iteración.</p>   | <p>“En algunos equipos el líder por parte del cliente está muy involucrado en el desarrollo y en otros equipos, solo supervisa.”</p>                |

Fuente: Elaboración propia



Tabla 2: Categoría: ROLES – Subcategoría: Scrum Master

| CATEGORÍA: ROLES   |  |
|--|--|
| SUBCATEGORÍA: Scrum Master   |  |
| TEORÍA   | REALIDAD EN AGILITY  |
| <p>En Scrum, el rol de Scrum Master es independiente del rol de Project Manager.</p>   | <p>“Se busca que un project manager tenga también el rol de scrum master”</p>  |
| <p>El primero es responsable de facilitar el proceso Scrum, asegurando que se sigan las prácticas y los principios de Scrum. Su enfoque principal es el equipo de desarrollo y el cumplimiento de las reglas de Scrum.</p> | <p>“No hay un rol específico de scrum master, hay. Lo común y lo esperado es que se haga cargo el project manager”</p> |
| <p>El Project Manager, por otro lado, se centra más en la gestión del proyecto en su conjunto, incluida la gestión del alcance, los recursos y los plazos</p>  |  |
| <p>El hecho de tener "demasiada gente" no debería ser un obstáculo para asumir el rol de Scrum Master. En Scrum, el tamaño del equipo de desarrollo no determina la idoneidad o viabilidad</p>                             | <p>“Soy project manager e intento ser scrum master. Porque tengo demasiada gente como para serlo”</p>                  |



|  |  |
|--|--|
| <p>del Scrum Master. Incluso en equipos grandes, el rol del Scrum Master sigue siendo esencial para facilitar el proceso Scrum y mantener el enfoque en la mejora continua.</p>  |  |
| <p>Si bien existen variaciones en la forma en que las organizaciones implementan Scrum, es importante tener en cuenta que los principios y valores fundamentales deben mantenerse intactos para lograr los beneficios esperados. Adaptar Scrum de manera desordenada o sin una comprensión adecuada de sus fundamentos puede afectar negativamente la eficacia y los resultados del proceso.</p> | <p>“Podes tener manager distintos, que tienen distintos conceptos de lo que es scrum o de la metodología que estén aplicando”</p>                                    |
|  | <p>“Cada uno lo adapta como puede, como te da el tiempo, como te permite lo que estabas trabajando, cambia un monton”</p>  |
| <p>En Scrum, no se menciona el rol específico de Business Analyst.</p> <p>Se espera, que el Scrum Master acompañe al equipo de trabajo en su día a día y garantice que todos comprendan y utilicen Scrum de forma correcta.</p>  | <p>“El Business Analyst es un gran apoyo con los temas de metodología, entre los dos vamos llevando que se aplique lo más que se pueda dentro de lo que tenemos”</p> |
| <p>La priorización de los elementos del backlog es responsabilidad exclusiva del Product Owner. El Product Owner es el único responsable</p>   | <p>“La priorización de los tickets la solemos hacer nosotros, del lado de la empresa”</p>  |





|  |  |
|--|--|
| de definir y priorizar los elementos del backlog según el valor y las necesidades del cliente o usuario final. |  |
|--|--|

Fuente: Elaboración propia

Tabla 3: Categoría: ROLES – Subcategoría: Equipo de desarrollo.

| CATEGORÍA: ROLES  |   |
|---|---|
| SUBCATEGORÍA: Equipo de desarrollo  |   |
| TEORÍA  | REALIDAD EN AGILITY   |
| Es recomendable que un equipo de desarrollo se componga de hasta nueve personas.  | Existen equipos de 17 personas.   |
|   | “Lo que determina la dinámica del equipo es también la cantidad de gente inevitablemente, cuanto menos gente tenes te permite mayor agilidad” |
| El equipo debe ser capaz de autogestionarse. Esto lo diferencia de un equipo tradicional donde un jefe de proyectos asigna a cada persona tareas concretas y fijas. | “Alguien tiene que asumir el rol de líder, si los grupos se dejan estar se arma un desastre”.   |



|  |  |
|--|--|
| <p>Al comienzo de cada Sprint se realiza una reunión de planificación del Sprint donde serán generados los acuerdos y compromisos entre el equipo de desarrollo y el Product Owner sobre el alcance del Sprint.</p> <p>El equipo de desarrollo tiene como una de sus responsabilidades tanto fundamentales como indelegables., proveer las estimaciones de cuánto esfuerzo será requerido para cada una de las características del producto.</p> | <p>Para las estimaciones de esfuerzo en las sprint planning, puede pasar que los equipos no son tan grandes como para que haya un consenso entre por lo menos dos personas</p> <p>“No hacemos “poker planning” por el hecho de tiempo, para puntuar las historias se utiliza más que nada el criterio de expertos, los líderes puntúan las estimaciones y después lo arreglan con su equipo”</p> |
| <p>En Scrum, se fomenta la formación de equipos autónomos y multidisciplinarios.</p> <p>Las dependencias externas pueden generar desafíos y afectar la autonomía y la eficiencia del equipo, contradiciendo algunos principios de Scrum</p>  | <p>“El equipo de diseño es del cliente, nos pasan los diseños que tenemos que desarrollar”</p>   |



Tabla 4: Categoría: CLIENTE

| CATEGORÍA: CLIENTE  |  |
|---|--|
| TEORÍA  | REALIDAD EN AGILITY  |
| La forma en que se arma la dinámica del proyecto y se sigue la metodología Scrum es una responsabilidad compartida entre el Product Owner, el equipo de desarrollo y el Scrum Master. | “El cliente tiene sus criterios y decide la forma en que se va a armar la dinámica del proyecto” |
| Si el cliente tiene un conocimiento limitado sobre Scrum, es responsabilidad del equipo Scrum, especialmente del Scrum Master, educar y guiar al cliente en el proceso ágil.          | “Es determinante lo que el cliente sepa sobre la metodología”                                    |
| Para que un método pueda considerarse ágil, debe ser adaptativo, es decir, debe considerar simple la posibilidad de introducir modificaciones y cambios en cualquier etapa.           | “En mi caso no tenemos la “agilidad” de tal manera, por el tipo de contrato que manejamos”       |
| Los métodos ágiles sugieren formas de construir un producto que acepten los cambios, se abre la puerta a adaptar nuevas técnicas prácticas útiles para el equipo y el producto.       | “En mi proyecto no puedo ser flexible ni cambiar cosas, como lo dice la metodología”             |



|  |  |
|--|--|
| <p>La participación del equipo de desarrollo en las reuniones de planificación de sprint es fundamental para establecer acuerdos y compromisos realistas. Su conocimiento técnico y experiencia les permite evaluar el alcance, proporcionar estimaciones y contribuir a la definición de tareas.</p>                          | <p>En un proyecto donde al cliente no le gusta tanto que el equipo participe de las meeting, se producen muchas restricciones.</p>   |
| <p>El refinamiento del Backlog es una actividad constante a lo largo de todo el Sprint. Su objetivo es profundizar en el entendimiento de los PBIs que se encuentran más allá del Sprint actual y así dividirlos en PBIs más pequeños. La participación de todo el Equipo Scrum es esencial para el éxito de esta reunión.</p> | <p>Cuando el cliente no quiere que el equipo participe en la meeting de refinamiento sólo participan el líder técnico y el scrum master, quienes determinan los puntos en esfuerzos estimados.</p> |

Fuente: Elaboración propia



Tabla 5: Categoría: METODOLOGÍA – Subcategoría: Requerimientos

| CATEGORÍA: METODOLOGÍA  |  |
|---|--|
| SUBCATEGORÍA: REQUERIMIENTOS  |  |
| TEORÍA  | REALIDAD EN AGILITY  |
| <p>El equipo estimará cada uno de los requisitos en función de su complejidad. Teniendo en cuenta la prioridad marcada por el Product Owner y la estimación realizada por el equipo, se acordará la cantidad de trabajo que se va a abordar en el siguiente sprint.</p> | <p>“El cliente llega con una lista de cosas que quieren que salgan en la app y con una sugerencia de “timeline” proponemos nuestro timeline según lo que nosotros consideramos que es lo óptimo”</p> |
| <p>Se parte de un presupuesto y unas fechas de entregas fijas, a partir de ahí, se trabaja para implementar la funcionalidad más valiosa para el cliente en cada momento (Ver Figura 4 - Triángulo de hierro)</p>   | <p>“Si el cliente nos cambia la complejidad en el medio del desarrollo, es otro precio, es otro tiempo”</p>  |

Fuente: Elaboración propia



Tabla 6: Categoría: METODOLOGÍA – Subcategoría: Procesos

| CATEGORÍA: METODOLOGÍA   |  |
|--|--|
| SUBCATEGORÍA: PROCESOS   |  |
| TEORÍA   | REALIDAD EN AGILITY  |
| En Scrum, uno de los principios fundamentales es entregar valor de forma incremental y continua al cliente o usuario final. La metodología promueve la entrega de incrementos de producto funcionales al final de cada sprint.     | “No estamos haciendo un incremento de valor sino que, entregamos al final del contrato todo lo que se prometió”  |
| El equipo de desarrollo es responsable de planificar, estimar, desarrollar y entregar incrementos de valor. No hacerse cargo de las entregas es una desviación del principio central de Scrum de entregar valor de forma continua. | “En uno de los proyectos no nos hacemos cargo de las entregas, sino que simplemente gestionamos gente, el cliente va pidiendo cosas y el equipo las va haciendo” |
| Scrum permite construir un producto de forma incremental. No se construirán trozos de producto por separado que luego tendremos que encajar, se construye contemplando la totalidad desde un principio.                            | “La metodología te dice que pasa por todas los procesos en un mismo sprint y eso no es siempre real”   |

Fuente: Elaboración propia



Como siguiente método aplicado en la recolección de datos, se realizó la **revisión de informes** sobre el proceso scrum en dos proyectos distintos a cargo de la empresa bajo estudio, con la ayuda de uno de los “project manager” entrevistados, se logró la comprensión de dichos informes, utilizando principalmente gráficas llamadas “velocity chart” armado por la empresa como principal herramienta que refleja los resultados del proyecto, destacando las causas y consecuencias de los mismos. A continuación se presentan la información y los gráficos analizados.

### **Proyecto 1**

El proyecto arranca con la asignación de un líder técnico que no estaba involucrado en el equipo ni se encontraba familiarizado con la metodología scrum, tampoco participaba en las planing ni en las reuniones de refinamiento. Los que sí se involucraron en dichos procesos eran los product owners.

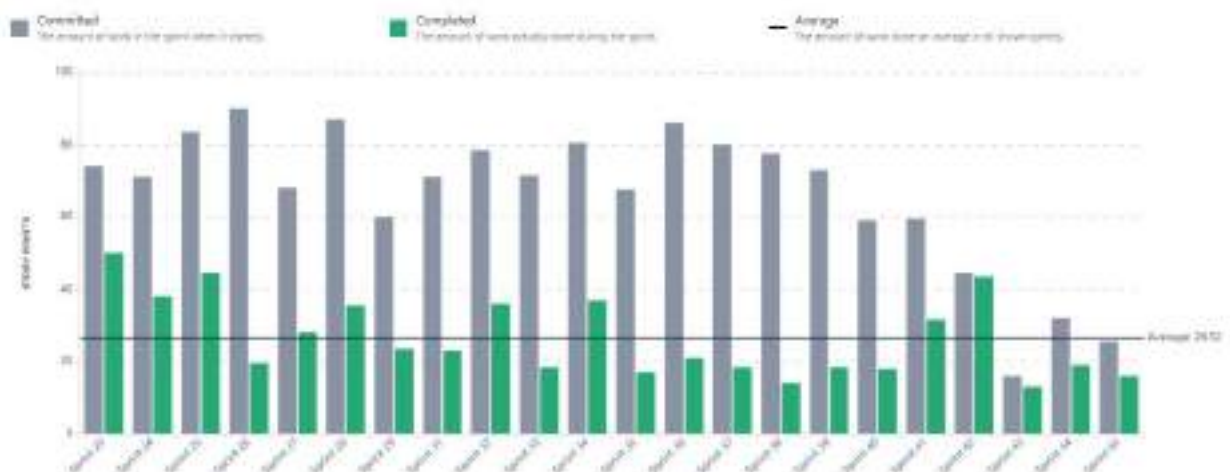
Se generaba la situación de que, mientras el equipo, dentro de su flujo de trabajo con marco scrum, realizaban todos los eventos de la metodología para la planificación del trabajo, en medio del sprint, aparecía el manager y pidiendo pausar lo que se venía haciendo para priorizar cuestiones urgentes de tecnología, también se pedía que dichas actividades estimadas en dos sprint se terminarían en solo uno. Los product owner por otro lado, no daban lugar a cuestiones tecnológicas ya que se enfocaban en mejoras de productos o temas relacionados. Se produce ahí un conflicto de intereses entre las prioridades, y por lo general siempre se hacía lo pedido por el manager ya que eran cuestiones urgentes y determinantes.

Esto generaba una pausa en lo planificado y si quedaba tiempo para realizar lo comprometido originalmente recién ahí se hacía, por lo contrario, se lo dejaba pendiente para los sprints.



A continuación, se presenta un “velocity chart” de los últimos 12 meses bajo análisis elaborado por la empresa en el cual las barras grises representan los comprometido por sprint y las barras verdes representan lo completado. Se puede interpretar como una foto de todo lo que entró al sprint y todo lo que se cerró en el mismo.

Gráfico de Barras 1 : Velocity chart de Proyecto 1



Fuente: Informe de la empresa bajo estudio “Agility”

Se puede ver en la gráfica cómo se va formando una “bola de nieve” con las cuestiones comprometidas que nunca se llegaron a completar (barras grises).

En el sprint 37 se genera un cambio drástico en la metodología utilizada para este proyecto, se realiza una división de 3 niveles para afrontar los potenciales problemas técnicos que surjan en el proceso sin perjudicar lo comprometido para el sprint. Las división mencionadas se caracterizan por la diferencia en sus actividades:





**Nivel 1:** Atienden los incidentes en el momento en que llegan a la empresa, si es posible los resuelven y si no es posible lo delegan a alguien de nivel más bajo para que sea priorizado dentro del backlog y lo trabajen como correspondan.

**Nivel 2:** Integrado por las personas que tienen un mayor conocimiento “técnico” dentro de cada equipo, atienden cuestiones más complejas y específicas de tecnología. Se manejan también con scrum, las prioridades/urgencias las determina el manager de este nivel. Tienen los roles necesarios para ser autosuficientes y atender las urgencias que se les asignen.

**Nivel 3:** Es el equipo de desarrollo para cada proyecto. Atienden cuestiones de producto.

El proyecto pasó a manos de otro manager. A partir de este corte, se empiezan a tener las prioridades bien definidas dentro del sprint y se comienza a achicar la brecha entre lo comprometido y lo completado. También disminuye y se estabiliza el compromiso tomado.

Como consecuencia, se fue generando también un cuello de botella en la parte de control de calidad (QA), y finalmente, en el sprint 42, como se puede ver en la gráfica, se cerró todo lo que se había pactado.

Otra consecuencia de la reestructuración, fue que el equipo del nivel 3 se empezó a enfocar en el backlog del producto y no cuestiones tecnológicas ni prioridades que no son visibles al cliente. Actualmente dichas prioridades tecnológicas que no tienen impacto en el producto final pero que son urgentes para el funcionamiento de todo el proyecto son atendidas por el nivel 2

## **Proyecto 2**

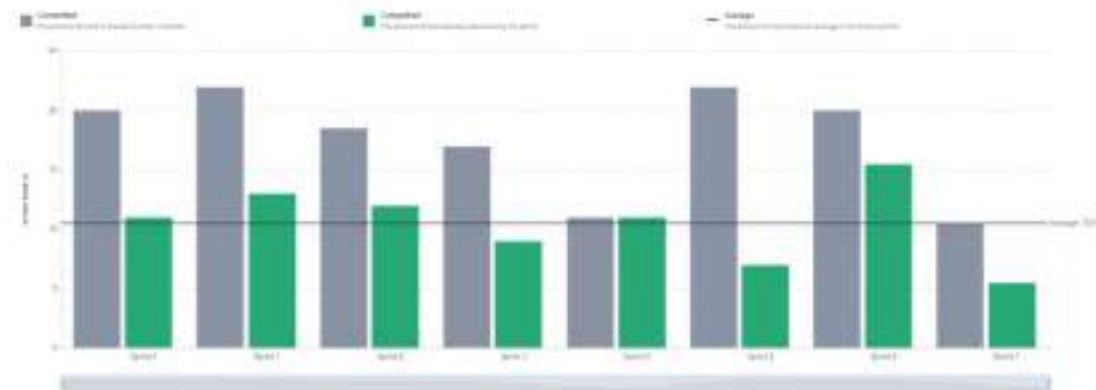
En este equipo, se llevó a cabo una transición de la metodología Kanban a Scrum. También, tiene una persona nueva en el rol de Product Owner y el Tech Manager es el mismo que anteriormente estaba en el proyecto 1 hasta el momento de quiebre que se menciona.



En este proyecto, el equipo no participa en todos los eventos de scrum. Como consecuencia el equipo no se entera de las actividades a realizar hasta las “planning meeting”, en las cuales sí participan.

Para el siguiente proyecto, se presenta un “velocity chart” desde su inicio hasta la fecha actual de análisis, los valores corresponden desde el sprint 0 hasta el sprint 7, Al igual que la figura anterior, las barras grises representan los comprometido por sprint y las barras verdes lo completado.

**Gráfico de Barras 2: Velocity chart de Proyecto 2**



**Fuente: Informe de la empresa bajo estudio “Agility”**

Para este caso, como se puede ver en la gráfica, la brecha entre lo comprometido y lo cerrado en el sprint está presente. El project manager del equipo considera que un factor generador de la brecha se debe a que “los primeros sprints son siempre de aprendizaje”. También asegura que una diferencia de 5 puntos es aceptable ya que puede haber dependencia en otros equipos que genera demoras en la validación del ticket o en las pruebas de calidad. El entrevistado no considera que sea real cumplir con todo lo comprometido en el sprint,



argumenta que en esa situación es muy probable que se esté trabajando en paralelo en cuestiones fuera del sprint.

En el Sprint 7 tuvieron un bloqueo en el ambiente de pruebas por un cambio generado por otro equipo y tomó bastante tiempo encontrar la causa de los errores presentes. Esto llevó a que se genere un “carryover” (pendientes) de varias tareas. Por otro lado, también se sumó un desarrollador nuevo que aumenta la capacidad del equipo para los próximos sprints.

Para continuar con la recolección de datos sobre la empresa bajo estudio, la inmersión en el campo y comprensión del mundo de las metodologías ágiles, se participó como “observador” de los siguientes eventos propios de la metodología scrum, cabe destacar, que en dichos eventos sólo participan trabajadores de la empresa “Agility”:

- **Daily meeting:** En dicha reunión participaron 17 personas, entre ellos el project manager correspondiente al proyecto y el equipo de desarrollo. La reunión fue liderada por uno de los miembros del equipo via “google meet”. El proyecto se encuentra en el Sprint 23. Se pudo observar como cada integrante, en orden, realizan un breve comentario sobre los temas en los que estuvo trabajando y también comentan en caso de tener un bloqueo, que fue lo que lo genera. Al finalizar, se realizó un “sorteo” para determinar la persona que liderará la próxima daily meeting.

- **Pre-planning:** En esta reunión de una hora de duración y con la participación de 16 miembros del equipo, se analizó rápidamente lo trabajado en el sprint 23, así como se planeó que tickets entrarían en el siguiente sprint. Se visualizó el product backlog del proyecto donde se especifica para cada ticket el estado de avance, la persona a la cual está asignado, el sprint al que pertenece, entre otros. El equipo, fue analizando uno por uno y decidiendo conjuntamente, con la principal influencia de la persona a la cual fue asignado, si son capaces de terminar el



trabajo a tiempo o se pasará al siguiente sprint (carryover), también se fue modificando el estado del ticket en función a lo que comentan los responsables de dicho avance. En segunda instancia, se relevaron todos los tickets pertenecientes al próximo sprint y se establece el “story point”, es decir, se le asignó un puntaje en función a la dificultad que conlleva la realización del trabajo, surgieron dudas sobre la descripción de algunos ticket para la asignación de puntos y concluyeron que al no tener dichas especificaciones, se le asigna un mayor puntaje, también se menciona la posibilidad de una pronta actualización de las características por parte del product owner. En caso que no sean actualizados, los tickets se enviarán al próximo sprint, lo cual afecta el proyecto debido a la dependencias de múltiples tickets sobre estos incompletos. Cabe aclarar, que esta reunión se realiza de manera previa al sprint planning oficial en la cual participa el cliente.

### **Criticidad de los hallazgos**

Tras el análisis exhaustivo y la identificación de las principales discrepancias entre la teoría y la práctica en cuatro proyectos de la empresa Agility, se llevaron a cabo entrevistas adicionales con el propósito de evaluar la importancia de estas diferencias en los resultados del proyecto y la eficacia del equipo de desarrollo. Para establecer esta evaluación, se implementó un criterio de criticidad basado en una escala compuesta por los siguientes niveles:

- No es crítico
- Moderadamente crítico
- Crítico
- Muy crítico
- Extremadamente crítico



La asignación de cada nivel fue realizada de manera subjetiva por los dos “project managers” entrevistados, tomando en cuenta su experiencia y las particularidades de los proyectos que encabezan. Los resultados se muestran a continuación:

**Tabla 7: Criticidad de las brechas teóricas-prácticas - Entrevistado 1**

| ENTREVISTADO 1                            | Tipo de contrato con el cliente                              | Falta de participación y acompañamiento del Scrum Master          | Falta de colaboración e involucramiento por parte del Product Owner | Gran cantidad de personas en el equipo de desarrollo  |
|---|--|---|---|---|
|   |  | Moderadamente crítico   | Muy crítico   | Extremadamente crítico                                |
| Falta de madurez del equipo de desarrollo | Ausencia del equipo en las reuniones de Scrum con el cliente | Mala estimación de esfuerzos para los tickets del product backlog | Mala gestión de dependencias externas                               | Cambios introducidos por el cliente durante el sprint |
| Moderadamente crítico                     | Crítico  | Moderadamente crítico   | Muy crítico   | Crítico   |

Fuente: Elaboración propia

A continuación de la asignación de nivel de criticidad para cada factor se procedió a analizar las razones de la elección.

Considerando el tipo de contrato con el cliente, el entrevistado lo considera “moderadamente crítico” ya que, en su caso, el contrato sólo determina la cantidad de horas de trabajo que se asignan a cada proyecto. No es en gran medida relevante para la eficiencia del equipo y del resultado final.

Asigna a la falta de participación y acompañamiento del Scrum Master el nivel “Muy crítico” ya que, en función a su experiencia, debido a su temporal ausencia los equipos no realizaban las reuniones y eventos que propone la metodología scrum dificultando así la comunicación y el seguimiento.



La falta de colaboración o involucramiento por parte del product owner lo considera extremadamente crítico. Argumenta que, sin su involucramiento, el equipo no tiene una guía para seguir lo solicitado por parte del cliente.

En base a la cantidad de personas en el equipo de desarrollo, menciona que debe ser un número relativamente bajo, sus equipos no tienen una gran cantidad de integrantes, lo que considera óptimo.

En función a la falta de madurez del equipo de desarrollo le asigna el nivel “moderadamente crítico” ya que los equipos van aprendiendo, especializándose. Lo que sí se destaca es la alta rotación de integrantes en uno de sus proyectos lo que dificulta lograr dicha madurez.

La ausencia del equipo en las reuniones de scrum con el cliente son críticas en la eficiencia del equipo ya que, en sus proyectos participan únicamente los roles del scrum master y líder técnico, los cuales pueden no tener en cuenta cuestiones necesarias de aclarar para facilitar el trabajo del equipo.

Una mala estimación del esfuerzo para los tickets del product backlog es moderadamente crítica, generalmente depende del equipo. Puede pasar que los responsables de realizar el trabajo no se animen a negar o minimizar el tiempo propuesto por alguien de mayor nivel. También se da el caso en que no existe charla o acuerdo entre las partes responsables de la estimación.

En base a una mala gestión de las dependencias externas, el entrevistado lo considera muy crítico, ya que es clave tenerlas en cuenta al momento de estimar el compromiso del trabajo a entregar. Pueden surgir demoras y bloqueos a razón de dichas dependencias, también pueden generarse cuellos de botellas.



Para finalizar, los cambios introducidos por el cliente durante el sprint se consideran críticos. Es normal que se presenten en el proceso del sprint. Su impacto recae en la agregación de nuevos tickets en el product backlog y/o en el cambio de las fechas de entregas acordadas.

**Tabla 8: Criticidad de las brechas teóricas-prácticas - Entrevistado 2**

| ENTREVISTADO 2                            | Tipo de contrato con el cliente                              | Falta de participación y acompañamiento del Scrum Master          | Falta de colaboración e involucramiento por parte del Product Owner | Gran cantidad de personas en el equipo de desarrollo  |
|---|--|---|---|---|
|   | Extremadamente crítico                                       | Crítico   | Muy crítico   | Muy crítico   |
| Falta de madurez del equipo de desarrollo | Ausencia del equipo en las reuniones de Scrum con el cliente | Mala estimación de esfuerzos para los tickets del product backlog | Mala gestión de dependencias externas                               | Cambios introducidos por el cliente durante el sprint |
| Crítico                                   | Moderadamente crítico  | Muy crítico   | Extremadamente crítico  | Moderadamente crítico                                 |

Fuente: Elaboración propia

En el caso del proyecto de este entrevistado 2, se considera extremadamente crítico el tipo de contrato con el cliente ya que trabajan con una modalidad de contrato llamada **“fixed price”**. En esta modalidad, el cliente solicita un producto, en este caso una funcionalidad con las características requeridas. Posteriormente la empresa analiza el lapso de tiempo para su desarrollo y entrega final, analizan la complejidad, las cantidades de personas necesarias en el equipo, paralización de tareas, entre otros ítems. A partir de ese análisis se presenta un presupuesto y un compromiso de tiempo de entrega. Estos últimos dos factores se mantienen fijos en función a lo pactado. Otra característica de dicho contrato, es que la entrega de valor se realiza al finalizar el lapso de tiempo comprometido, a diferencia de la teoría, no se realizan entregas continuas de valor en cada iteración. Este hallazgo es fundamental para la comprensión de los siguientes factores.



La falta de participación y acompañamiento del Scrum Master es crítica. En este proyecto en particular se cuenta con el rol de un Business Analysis el cual trabaja conjuntamente con el project manager en cuestiones de relación con el cliente y demás responsabilidades inherentes al puesto.

La falta de colaboración e involucramiento por parte del Product Owner se considera muy crítico, ya que es quien determina las especificaciones y criterios de aceptación de cada ticket a trabajar, la falta de los mismos desemboca en bloqueos y demoras del trabajo restante.

Una gran cantidad de personas en el equipo de desarrollo se considera también muy crítico. El equipo de desarrollo en cuestión cuenta con 17 personas, lo que dificulta la agilidad, comunicación e intercambio de ideas entre los mismos.

En base a la mala estimación de esfuerzos para los tickets del product backlog, en este tipo de contrato, se considera muy crítico, ya que la estimación se realiza al momento de pactar con el cliente y abarca todo el proyecto de desarrollo. Una mala estimación puede afectar el presupuesto calculado y llevar a la empresa a incurrir en costos adicionales.

Teniendo en cuenta una mala gestión de dependencias externas, el project manager lo considera muy crítico ya que es una situación frecuente en su flujo de trabajo. Estas dependencias que emergen mientras se avanza con los tickets producen retrasos y bloqueos.

Relacionado a los cambios introducidos por el cliente durante el sprint, no se consideran tan relevantes ya que, debido al tipo de contrato, el cliente puede introducir cambios que modifican el tiempo y el presupuesto pactado.





## Conclusiones

En el transcurso de esta investigación, se ha abordado inicialmente el marco teórico relacionado con las metodologías ágiles y sus características. Esto se debe a que en los últimos años ha habido un aumento significativo en su adopción por parte de las empresas en la industria del desarrollo de software impulsado por su flexibilidad, adaptabilidad y otros beneficios. Si bien cada proyecto adapta estas metodologías según sus particularidades, se destaca la importancia de que una incorrecta utilización de estas metodologías, limita su efectividad y los beneficios que pueden brindar.

Para llevar a cabo esta investigación, se utilizaron técnicas de recolección de datos cualitativas que permitieron, en primera instancia, identificar y analizar las discrepancias teóricas-prácticas presentes en la empresa Agility.

En segunda instancia, se buscó comprender las causas y el impacto de estas discrepancias. Los hallazgos obtenidos en la primera etapa se tradujeron en categorías y subcategorías con el objetivo de agrupar la información en cuadros comparativos, en relación a las características de las discrepancias y su correspondencia con el marco teórico propuesto. Además, se analizaron informes y participación en eventos de la empresa para complementar estos hallazgos.

Posteriormente, se determinaron las causas más significativas y se les asignó un nivel de criticidad en función al impacto en los resultados del proyecto y la eficiencia del equipo. El análisis de los resultados reveló la importancia de encontrar alternativas de acción que minimicen las consecuencias de adaptar y abordar la metodología de manera desviada de su objetivo. En el análisis se identificó, debido a la rigidez de un proyecto en particular, que se aproxima al enfoque clásico de cascada y utiliza los recursos propuestos por scrum.



Asimismo, se estableció un nivel de criticidad para dichos factores, en relación al logro efectivo de los resultados del proyecto y la eficiencia del equipo, resaltando la relevancia de cumplir con los estándares básicos propuestos por las metodologías ágiles.

Finalmente, se destaca que, si bien existen diversas características y proyectos que condicionan la utilización de las metodologías ágiles y proponen adaptaciones, es fundamental no perder de vista los fundamentos y principios básicos, apartarse de las prácticas ágiles establecidas puede desencadenar una serie de problemas y dificultades en el desarrollo del proyecto. Al no seguir las propuestas metodológicas de manera coherente, se corre el riesgo de perder la agilidad y los beneficios asociados.

### **Recomendaciones**

En función de los resultados obtenidos se recomienda a la empresa Agility las siguientes alternativas de acción:

- Generar conciencia sobre los beneficios y la importancia de una cultura ágil. Esto implica comunicar de manera efectiva los valores y principios ágiles, así como su impacto en la eficiencia, colaboración y adaptabilidad.
- Es importante que todos los managers involucrados en la aplicación de Scrum tengan una comprensión común de los principios y prácticas ágiles. Realizar capacitaciones y talleres para asegurar que todos tengan conocimientos sólidos sobre Scrum y cómo aplicarlo de manera efectiva.
- Considerar la posibilidad de ajustar la estructura organizativa para facilitar la adopción de Scrum de manera más efectiva. Esto podría implicar la creación de equipos multidisciplinarios autoorganizados con sus propios Scrum Masters dedicados, permitiendo así que el Project Manager se enfoque en la gestión general del proyecto



### Recomendaciones "Project Manager" 1

- Proporcionar al cliente recursos y materiales educativos sobre metodologías ágiles. Esto puede incluir guías, documentación y ejemplos prácticos que ayuden a comprender mejor los conceptos, beneficios y prácticas ágiles. Además, se pueden ofrecer capacitaciones específicas según las necesidades del cliente, con el fin de fortalecer su comprensión y participación en el proceso ágil.
- Fomentar el desarrollo de habilidades de liderazgo en todos los miembros del equipo. Animar a los mismos a asumir responsabilidades de liderazgo en diferentes momentos y situaciones, promoviendo así un liderazgo distribuido y compartido.
- Al comenzar un proyecto, es importante establecer expectativas claras con respecto al nivel de involucramiento del líder por parte del cliente. Definir qué tipo de participación se espera, ya sea como miembro activo del equipo de desarrollo o como supervisor, y comunicar estas expectativas a todos los involucrados.
- Asegurarse de que los miembros relevantes del equipo de desarrollo participen activamente en las reuniones de Scrum con el cliente. Esto ayudará a evitar posibles malentendidos y asegurar una alineación adecuada entre el cliente y el equipo.
- Reevaluar los procesos actuales y determinar si la transición de Scrum a Kanban es la opción más adecuada para el proyecto. Analizar las características del trabajo, los flujos de trabajo existentes y los desafíos específicos que se están enfrentando.

### Recomendaciones "Project Manager" 2

- Evaluar la modalidad de contrato "fixed price" y considerar la posibilidad de adoptar un enfoque más flexible, como contratos basados en iteraciones. Esto permitirá una mayor adaptabilidad a los cambios y una entrega continua de valor al cliente.



- Organizar la creación de sub equipos más pequeños, asegurando una distribución equilibrada de habilidades y experiencia en cada uno. Establecer criterios claros para la división, por ejemplo, naturaleza de los proyectos, las áreas de expertise, las habilidades requeridas, la carga de trabajo y la necesidad de mayor agilidad y autonomía en la toma de decisiones.
- Explorar la posibilidad de introducir incrementos de valor en el proyecto, incluso dentro del contexto del contrato "fixed price". Identificar funcionalidades o características adicionales que puedan ser incorporadas en entregas parciales a lo largo del desarrollo, de manera que el cliente pueda recibir valor de manera temprana y continua.
- Establecer prácticas y técnicas más sólidas para la estimación de esfuerzos, especialmente en el contexto de contratos "fixed price". Realizar un análisis detallado de los requerimientos y considerar la experiencia previa para realizar estimaciones más precisas y realistas.
- Implementar prácticas de gestión ágil del alcance, como la técnica de "priorización MoSCoW" (Must, Should, Could, Won't), en la cual se establecen niveles de prioridad para los requisitos y funcionalidades del proyecto.
- Implementar un proceso de gestión de dependencias más riguroso y proactivo. Establecer mecanismos para identificar y gestionar las dependencias externas de manera oportuna y eficiente. Esto ayudará a minimizar los retrasos y bloqueos causados por dependencias no controladas.



## Bibliografía

- Alaimo, D. M. (2013) Proyectos ágiles con Scrum: flexibilidad, aprendizaje, innovación y colaboración en textos complejos. Ciudad Autónoma de Buenos Aires. Kleer.
- García, J. (2015). Metodologías ágiles: Una alternativa para el desarrollo de software. Revista de Investigación Académica.
- García, J. (2017). Agile: Desarrollo de software con Scrum. Alfaomega.
- Hernández-Sampieri, R., & Mendoza, C. (2018). Metodología de la investigación. Las rutas cuantitativa, cualitativa y mixta. México. McGraw Hill.
- Jordán D. (2020). Estudio sobre las metodologías ágiles y metodologías tradicionales para gestión de proyectos de software. Trabajo Fin de Master en Dirección de Proyectos Informáticos. Escuela Politécnica Superior de la Universidad de Alcalá. 61 p.
- López, J. (2018). Metodologías ágiles y Scrum: Gestión de proyectos ágiles. Marcombo.
- Quesada Reyes N. T. (2020) Estudio sobre Metodologías Ágiles en los Proyectos Software. Propuesta de Plan de Implantación para PYMES. Trabajo Fin de Máster en Organización Industrial y Gestión de Empresas. Universidad de Sevilla
- Rubio, F. (2016). Metodologías ágiles: Scrum, Kanban, Lean. Ra-Ma.
- Rubio, F. (2017). Scrum: Una metodología ágil para el desarrollo de software. Revista de Tecnología e Innovación.
- Kurup, D., & Kala, S. (2015). Agile Project Management – Benefits and Challenges. Research Paper for ISM6316.001 Project Management. University of South Florida.



- Wolk, L. (2003). Coaching – El arte de soplar brasas (pp. 22-23).

Ciudad: Editorial.

- Freeman, R. E. (2010). Strategic Management. Cambridge University

Press.

- Anderson, D. J. (2010). Kanban: Successful Evolutionary Change for

Your Technology Business. Seattle, WA: Blue Hole Press.

## Apéndice

*Entrevista a Project Manager/Scrum master de la empresa Agility en el área  
“mantenimiento a aplicaciones”*

- **¿Qué metodologías utilizan en la empresa para el desarrollo de sus productos?**

- **¿Cuando empezaste a trabajar en la empresa, como fue el tipo de capacitación que recibiste en base a las metodologías que se aplican?**

- **Con respecto a tu puesto de trabajo ¿sos project manager y dentro de cada proyecto te desarrollas como scrum master?**

- **¿Cómo manejar los conflictos que se generan en el equipo de desarrollo?**

- **¿Esos líderes por parte del cliente, integran los equipos?**

- **¿Cómo te aseguras que exista una buena comunicación entre el equipo?**

- **¿Cuáles consideras que son los factores que determinan la dinámica del equipo?**



- ¿El product owner puede ser llevado por el cliente y/o del lado de la empresa?
- ¿Cómo se organizan para cumplir con los plazos y los requerimientos del sprint? ¿Qué pasa si no se cumple con el compromiso tomado, ya sea atraso o adelanto?
- ¿Cómo manejan los cambios en los requerimientos durante el sprint?
- ¿Cómo se manejan los equipos de desarrollo “autogestionados”?
- ¿Cómo se estiman los esfuerzos para comprometerse con los requisitos para un sprint?
- ¿De qué manera miden los resultados de cada sprint?
- ¿Tienen alguna forma para medir la calidad del producto que se entrega al cliente?
- ¿Qué reuniones se llevan a cabo a lo largo del proyecto?
- ¿Consideras que están utilizando al máximo el potencial de las metodologías Agile en su equipo de trabajo? ¿Hay algún aspecto en el que sentís que deben mejorar?

*Entrevista a Project Manager/Scrum master de la empresa Agility*

- En función a tu puesto ¿Sos project manager y dentro de tus proyectos sos scrum master?
- Dentro de tus responsabilidades ¿cuáles consideras que son las más importantes?
- ¿Me podrías contar sobre los proyectos en los que participas?



- **¿Me podrías comentar el proceso de organización en scrum y kanban desde que el cliente se presenta en la empresa hasta que trabajan en el proyecto?**
- **En base a tu experiencia ¿cómo se capacita a una persona que empieza a trabajar con metodologías ágiles?**
- **¿Cuáles crees que son los principales beneficios que te brinda la metodología?**
- **¿Crees que existen desventajas en la utilización de las mismas?**
- **¿Cuáles son los factores que marcan la diferencia entre grupo y grupo?**
- **¿Cuáles son los mayores desafíos que se te presentaron utilizando metodologías ágiles?**
- **¿En el caso de tus proyectos, el product owner está conectado con el equipo de desarrollo?**
- **¿Tienen indicadores o métricas que te den un indicio de la calidad del producto o satisfacción del cliente?**
- **¿En tu mismo equipo tenes gente de QA?**
- **¿Cómo gestionas que la aplicación de la metodología se esté haciendo correctamente?**
- **En función a tu experiencia ¿existen los equipos autogestionados?**