PESQUIMAT 26(2): 67–80 (2023) ISSN:1560-912X/ ISSN-E:1609-8439
https://doi.org/10.15381/pesquimat.v26i2.26312 Facultad de Ciencias Matemáticas – UNMSM

Desafíos de la ingeniería de requerimientos en la metodología programación extrema (XP): una revisión sistemática

Oscar Benito Pacheco[1], *Juan Luna Valdez*[2] *y Edinson Montoro Alegre.*[3]

Resumen: La práctica de la Ingeniería de Requerimientos (IR) en el desarrollo ágil es sustancialmente diferente al desarrollo tradicional. La programación extrema – (XP) (Beck, 2004) es uno de los enfoques de estos métodos ágiles. Las metodologías tradicionales con su propuesta en cascada, en fases pre establecidos, vienen siendo reemplazado por las metodologías agiles, su desarrollo de tipo interactivo e incremental, con requerimientos no estáticos. El objetivo de estudio, describe una investigación inductiva y empírica sobre desafíos y prácticas con la metodología XP e IR basada en un análisis de datos recopilados en la literatura académico, científico. Para alcanzar los resultados se implementó una estrategia que permitió hallazgos de documentos, siguiendo las pautas de una Revisión Sistemática de la Literatura (RSL). De esta manera, contribuimos con una descripción de desafíos en la IR en relación con el desarrollo de proyectos con la metodología ágil XP.

Palabras clave: Ingeniería de requerimientos; metodologías ágiles; Practicas en ingeniería de requerimientos agiles; programación extrema.

Challenges of requirements engineering in the extreme programming (XP) methodology: a systematic review

Abstract: The practice of Requirements Engineering (RI) in agile development is substantially different from traditional development. Extreme programming – (XP)(Beck, 2004) is one of the approaches of these agile methods. Traditional methodologies with their waterfall proposal, in pre-established phases, are being replaced by agile methodologies, their interactive and incremental development, with non-static requirements. The objective of the study describes an inductive and empirical investigation on challenges and practices with the XP and IR methodology based on an analysis of data collected in academic and scientific literature. To achieve the results, a strategy was implemented that allowed document findings, following the guidelines of a Systematic Literature Review (RSL). In this way, we contribute with a description of challenges in IR in relation to the development of projects with the agile XP methodology.

Keywords: Requirement's engineering; agile methodologies; Practices in agile requirements engineering, extreme programming.

Recibido: 15/05/2023. *Aceptado:* 17/12/2023. *Publicado online:* 30/12/2023.

[1]UNMSM, Facultad de Ciencias Matemáticas. e-mail: obenitop@unmsm.edu.pe
[2]UNMSM, Facultad de Ciencias Matemáticas. e-mail: jlunav@unmsm.edu.pe
[3]UNMSM, Facultad de Ciencias Matemáticas, e-mail:emontoroa@unmsm.edu.pe

1. Introducción

En los momentos actuales, los cambios en los negocios o transacciones comerciales son muy rápidos debido a que nos encontramos en un mundo competitivo y la frecuente aparición de nuevos productos en el mercado. Estas modificaciones en los requerimientos del negocio desencadenan variantes en los requerimientos de los productos de Tecnología de Información (TI). Gestionar los requerimientos del negocio y de TI que cambian rápidamente, y desarrollar el producto software persistente y adaptable resultan un desafío y por ende estos resultan costosos. En estas circunstancias los modelos tradicionales de desarrollo de software, como la metodología en cascada, no son eficientes en tales circunstancias. Por tanto, se hace necesario avanzar hacia técnicas innovadoras como los ágiles de desarrollo de software. Las metodologías ágiles en la industria de la Ingeniería de software han venido constituyéndose en un factor importante, para el desarrollo de aplicaciones en las organizaciones de TI. Cabe destacar el retorno de la inversión con mayor valor y mejores posibilidades de éxito de los proyectos. La programación extrema – (XP) (Beck, 2004) es uno de los enfoques de mayor difusión entre los métodos ágiles. En particular la actividad de Ingeniería de Requerimientos (IR) no es labor explicita en XP, no obstante, se trata de entender los desafíos que presentan para proponer superarlos como producto de prácticas implementadas y estas han sido publicadas en diferentes artículos que es materia de la investigación del presente trabajo.

Motivación

Partiendo de la importancia que tiene la correcta realización de los procesos asociados a la IR para el desarrollo exitoso del producto de software, mostramos las tablas del informe elaborado en CHAOS REPORT 2015 del Standish Group (The Standish Group., 2016), el mismo se muestra en la tabla 1.

Tabla 1: Comportamiento de los proyectos desde 2011 al 2015

	2011	**2012**	**2013**	**2014**	**2015**
Exitosos	29 %	27 %	31 %	28 %	29 %
Cuestionados	49 %	57 %	50 %	55 %	52 %
Fracaso	22 %	17 %	19 %	17 %	19 %

Se considera:

Éxitosos: proyectos finalizados dentro del plazo y presupuesto y cumpliendo todos los requerimientos.

Cuestionados: proyectos finalizados, pero fuera de plazo, fuera de presupuesto y sin cumplir todos los requerimientos.

Fracaso: proyectos cancelados durante el desarrollo.

Tabla 2. Factores relacionados con el destino final de los proyectos de software según el CHAOS Report

Factores de éxito	**Causas de problemas**	**Causas de fracasos**
Implicación de los usuarios	Falta de información por parte de los usuarios	Requisitos incompletos
Apoyo de los directivos	Especificaciones y requisitos incompletos	Falta de implicación de los usuarios
Enunciado claro de los requisitos	Especificaciones y requisitos cambiantes	Falta de recursos
Planificación adecuada	Falta de apoyo de los directivos	Expectativas no realistas
Planificación adecuada de recursos TI	Incompetencia tecnológica	Falta de apoyo de los directivos
Expectativas realistas	Falta de recursos	Especificaciones y requisitos cambiantes
Hitos de proyectos pequeños	Expectativas no realistas	Falta de planificación
Personal competente	Objetivos poco claros	Ya no lo necesito
Sentimiento de propiedad	Plazos temporales no realista	Falto de registro de TIC
Visión y objetivos claros	Nueva tecnología	Desconocimiento de la tecnología
Trabajo duro y personal concentrado		

Así mismo, según informe el Chaos Report 2015, muy pocos proyectos considerados muy grandes y/o grande en la tabla 3, se les califica bien, atendiendo a las 3 restricciones de la gestión de proyectos, costo, tiempo y alcance. Situación diferente presentan los proyectos pequeños, que tienen más de un 62 % de posibilidades de éxito; un proyecto muy grande o grande no tiene prácticamente ninguna posibilidad de terminar a tiempo, dentro del presupuesto y dentro de su alcance, que es la definición del Standish Group de un proyecto exitoso. Los proyectos muy grandes tienen un mayor margen de terminar fuera del plazo, fuera de presupuesto, en contraposición a los proyectos pequeños y moderados.

Tabla 3. Destino de los proyectos según su tamaño, basado en datos extraídos del Chaos Report 2015

	Exitosos	**Cuestionados**	**Fracasos**
Muy grandes	2 %	7 %	17 %
Grandes	6 %	17 %	24 %
Medianos	9 %	26 %	31 %
Moderados	21 %	32 %	17 %
Pequeños	62 %	16 %	11 %
	100 %	**100 %**	**100 %**

Las metodologías tradicionales con su propuesta en cascada, en fases pre establecidos vienen siendo reemplazados por las metodologías ágiles, haciendo su desarrollo de tipo interactivo e incremental, por ejemplo, con requerimiento no estáticos. La Ingeniería de Requerimientos (IR) para el desarrollo de software ágil es diferente de la Ingeniería de Requerimientos tradicional. Porque la obtención de los requerimientos, su validación y su gestión son llevadas a cabo por

especialistas de IR, en una fase separada en el tiempo de las fases del análisis, del diseño y desarrollo, y documentados en artefactos de especificación de requerimientos.

El objetivo principal del presente trabajo es estudiar e identificar los desafíos que se presentan en la ingeniería de requerimientos en la metodología ágil Programación Extrema (XP).

Este objetivo logrará:

- Obtener información sobre el problema y su contexto, esto es, descubrir los principales retos que enfrentan los profesionales, con respecto a la ingeniería de requerimientos.
- Recopilar una colección de enfoques (prácticas, métodos, técnicas) seguidos por los profesionales.
- Comparar las contribuciones extraídas de las bases de datos electrónicas con los datos empíricos obtenidos de la industria del *software*. Esta comparación se establecerá considerando los desafíos encontrados, los enfoques propuestos y los impactos observados.

Preguntas de Investigación (PI)

Las preguntas de investigación general que impulsa el presente trabajo son:

PI 1: ¿Qué desafíos existen con respecto a ingeniería de requerimientos en proyectos con el uso de la metodología ágil XP?

PI 2: ¿Qué prácticas se siguen para superar estos desafíos?

2. Trabajos relacionados

2.1. Interacción entre el equipo de desarrollo y el cliente

Tener un cliente accesible inmediatamente, es un punto clave en todos los enfoques ágiles (Paetsch et al., 2003) también señala que el cliente está involucrado durante todo el tiempo de desarrollo, pero no se garantiza que todos los usuarios o clientes con los antecedentes necesarios estén presentes.

2.2. Ingeniería de requerimientos iterativos

En la metodología XP, las funcionalidades se liberan en ciclos pequeños y frecuentes. Esto permite que el equipo de desarrollo obtenga más comentarios en tiempo real del cliente (Corral et al., 2013). Al comienzo del proyecto, el equipo involucrado adquiere una comprensión de alto nivel de las características del software. Lo que motiva no poner demasiado esfuerzo en la actividad de ingeniería de requerimientos al principio, por lo que presentan la alta volatilidad de los requerimientos, el conocimiento incompleto de la tecnología utilizada o los clientes que no pueden describir claramente los requerimientos antes de verlos. Después de la breve identificación de los requerimientos al principio, la ingeniería de requerimientos ágil continúa en cada ciclo de desarrollo. Al comienzo de cada ciclo, el cliente y el equipo de desarrollo se reúnen y discuten sobre las características que deben implementarse (Cao & Ramesh, 2008b).

2.3. Priorización de requerimientos

En el desarrollo, las características de mayor prioridad se implementan temprano para que los clientes puedan obtener el mayor valor comercial. La priorización debe repetirse con frecuencia durante todo el proceso de desarrollo porque aumenta la comprensión del proyecto y se agregan nuevos requerimientos durante el desarrollo (Paetsch et al., 2003). (Cao & Ramesh, 2008b)

recomiendan que los requerimientos se prioricen al comienzo de cada ciclo de desarrollo. El cliente y el equipo de desarrollo asignan prioridades para que cada característica se asegure que los requerimientos más importantes se implementen primero.

3. Metodología

A A continuación, se expone la metodología de investigación y la Revisión Sistemática de la Literatura (RSL) en torno al tema de ingeniería de requerimientos en el ámbito de desarrollo de software ágil implementado con la metodología XP, cumpliendo los protocolos o pautas establecidos (Kitchenham & Charters, 2007) para realizar revisiones sistemáticas de la literatura en ingeniería de software (Okesola et al., 2019). Se describe en la tabla 4.

En primer lugar, en la ejecución del presente trabajo, se definió el problema a resolver, lo cual permitió establecer el objetivo de la investigación.

Luego, las preguntas de investigación fueron formuladas, revisadas y desarrolladas para proporcionar una respuesta a cada una de ellas y se derivaron conclusiones sobre el tema para la apertura de la discusión en esta materia.

Tabla 4. Escenario metodológico de la investigación.

Escenario	Descripción
Método de enfoque	Inductivo
Naturaleza de los datos	Cualitativos
Sobre el alcance	Revisión Sistemática de la literatura
Método de Procedimiento	Temática de Análisis y Síntesis
Variables independientes (X)	Ingeniería de requerimientos y metodología ágil XP
Variables dependientes (Y)	Desafíos y limitaciones

Revisión sistemática de la literatura (RSL)

La referencia principal utilizada para llevar a cabo la RSL es una guía para realizar revisiones sistemáticas de la literatura en ingeniería de software, desarrollada (Kitchenham & Charters, 2007). Las razones para usar estas pautas son las siguientes: están respaldadas por muchas citas, son ampliamente utilizadas. Estas pautas establecen llevar a cabo la RSL en tres etapas principales: 1) planificación; 2) realización de investigaciones; 3) elaboración de informes y difusión. Cada una de estas etapas está compuesta por varias actividades.

4. Resultados de la revisión sistemática de la literatura (RSL)

Como se mencionó en el párrafo anterior, la estrategia para localizar fuentes relevantes ha consistido en revisar todas las referencias y citas de (Inayat et al., 2015). La revisión sistemática de la literatura, que se ha tomado como punto de partida, repite sucesivamente el proceso para encontrar todo el material relevante disponible.

Esta estrategia ha proporcionado 271 fuentes potenciales para analizar, lo que constituye una cantidad aceptable de contribuciones relacionadas con el tema.

En la Tabla 5, se puede encontrar una descomposición detallada de las fuentes de interés encontradas y los filtros sucesivos aplicados para eliminar las irrelevantes. Así mismo son 19 documentos los que tratan con mayor detalle el tema de investigación.

Tabla 5. Criterio de filtro de las fuentes

Filtros de fuentes	Citaciones	Referencias	Resultados
Títulos	81	108	189
Resumen	15	29	44
Lectura rápida	8	17	25
Lectura exhaustiva	6	7	13
Subtotal	110	161	271
Preseleccionado	7	12	19

Resultados de RSL

Revisando las preguntas de investigación de manejo de esta RSL, ha sido posible encontrar los resultados que se exponen a continuación.

PI 1. ¿Qué desafíos con respecto a la ingeniería de requerimientos se encuentran en proyectos ágiles en XP?

Para alcanzar una visión global de los desafíos que surgen con respecto a la Ingeniería de Requerimientos en ambientes de desarrollo ágil, podemos describir los siguientes considerandos antes y después del desarrollo.

A. Delimitación de idoneidad

La propuesta de desarrollo ágil no funciona para todo tipo de proyecto; esta es una opinión compartida entre la mayoría de las fuentes incluidas en la RSL (Boehm & Turner, 2003)(Lehman & Sharma, 2011). Existen ciertas condiciones que restringen a priori la aplicabilidad de los principios ágiles; estos se pueden clasificar como exponemos a continuación.

- La propuesta de desarrollo ágil no es adecuada para proyectos de gran envergadura

En muchos estudios precisan que la estimación del presupuesto y el cronograma es un reto para las organizaciones que siguen métodos ágiles. Aunque la práctica de métodos ágiles permite la valoración inicial de un proyecto, no es posible hacer estimaciones iniciales debido a requerimientos volátiles, planificación dinámica y diseño (Ramesh et al., 2010). El alcance de un proyecto se basa en historias de usuarios conocidas y algunas de estas pueden descartarse en próximas iteraciones. Por lo tanto, las estimaciones a veces se ven significativamente afectadas (Ramesh et al., 2010).

- Conocimiento del dominio y complejidad del proyecto

Aunque no todas las fuentes mencionan explícitamente este reto, se puede deducir la importancia de la experiencia en el contexto ágil. Cuando un equipo tiene experiencia en un área específica, su dominio permite la posibilidad de crear soluciones incrementales (Farid & Mitropoulos, 2012) siempre que el equipo posea una comprensión profunda del problema.

- Tamaño del equipo

Esto es muy importante; las metodologías ágiles requieren una gran coordinación y comunicación entre el equipo. Trabajando con equipos muy grandes, incluso los equipos distribuidos pueden significar una dificultad inevitable para la correcta aplicación de los principios ágiles (Lehman & Sharma, 2011).

B. Desafíos del proyecto

Esta categoría se compone de problemas que aparecen durante la ejecución del proyecto, como en la sección anterior. Estas dificultades son obvias para el equipo o el cliente y pueden significar esfuerzos muy costosos para resolverlas sobre la marcha (Lehman & Sharma, 2011) (Farid & Mitropoulos, 2012) (Bjarnason et al., 2011) (Daneva et al., 2013).

- Falta de calidad, gestión profesional y compromisos contractuales

Esto se refiere al hecho de que muy a menudo en proyectos ágiles, la calidad desde las perspectivas de las normas Organización Internacional de Estandarización (ISO), Modelo de Madurez de Capacidad e Integración (CMMI) no se considera relevante; el tiempo, el presupuesto y la funcionalidad llaman la atención del equipo (Farid & Mitropoulos, 2012) y el software resultante no presenta la calidad requerida (Sarkan et al., 2011) (Abrahamsson et al., 2004).

Como consecuencia a los compromisos y limitaciones contractuales a ello se suma la volatilidad de los requerimientos son importantes al no permitir cambios en los requerimientos después de la firma del contrato. Los cambios pueden causar un aumento en el costo y, a veces, el fracaso de los proyectos, por lo tanto, se deben tomar medidas legales para evitar tal situación y manejar adecuadamente la naturaleza flexible de la ingeniería de requerimientos ágiles. Sin embargo, estos problemas pueden resolverse mediante un pago fijo por lanzamiento, que protege las inversiones y evita la volatilidad de los requerimientos. También es probable que los contratos de pago fijo reduzcan los riesgos específicos del proyecto a nivel de proveedor en proyectos externalizados basados en el desarrollo ágil (Daneva et al., 2013).

C. Desafíos del equipo

Aquí se agrupa los desafíos que afectan directamente al equipo, ya sea en su percepción de las tareas que están llevando a cabo, su comunicación (malentendidos, discusiones, dificultades para establecer comunicación) o en la dependencia o su experiencia para realizar un trabajo efectivo(Farid & Mitropoulos, 2012)(Ramesh et al., 2010).

Confianza en la experiencia

Esto refleja que las metodologías ágiles dependen demasiado de la experiencia de las personas que componen el equipo, no del método en sí. Esto puede conducir a una dependencia de las personas lo que podría significar grandes problemas si tales personas no estén en el equipo (Farid & Mitropoulos, 2012)(Ramesh et al., 2010).

Problemas de comunicación del equipo

El rol del gestor del proyecto es un vehículo muy útil para permitir la comunicación entre el equipo; todos los miembros del equipo pueden compartir una visión común del software consultando el diseño del sistema, por ejemplo, en XP se trata como una metáfora y contribuye a tener una visión. Quitar este vehículo puede llevar a perder la perspectiva común del sistema (Farid & Mitropoulos, 2012).

D. Desafíos del cliente

Estos desafíos comparten la característica común de que todos involucran directamente al cliente; ya sea durante el proyecto o cuando se entrega el software (Abdullah et al., 2011).

El cliente compartiendo el área de desarrollo es asumido y defendida en los ambientes ágiles. Sin embargo, esta suposición a menudo no es realista, ya que los estudios empíricos confirmaron

que la disponibilidad del cliente y el acceso son un desafío general (Pichler et al., 2006)(Ramesh et al., 2010). Si bien ningún estudio discute que los cambios en los requerimientos pueden ser determinados directamente por el cliente para acelerar el proceso, la disponibilidad del cliente se considera muy escasa debido a muchos factores desde una perspectiva comercial, como el tiempo, costo y carga de trabajo del representante del cliente (Racheva et al., 2010).

- Problemas de comunicación con el cliente

La ingeniería de requerimientos es básica para permitir una comunicación fluida con el cliente a fin de garantizar la obtención de sus necesidades respecto al software que se está desarrollando. La falta de ingeniería de requerimientos hace que esta comunicación sea mucho más complicada y aparecen interpretaciones no siempre coincidentes entre el equipo y el cliente(Abdullah et al., 2011)].

La incapacidad y la falta de conformidad del cliente son los dos problemas principales, aparte de la disponibilidad (Daneva et al., 2013). La incapacidad del cliente se refiere a su incompetencia en términos de toma de decisiones y conocimiento completo del dominio, y la conformidad del cliente se trata del consenso de más de un grupo de clientes involucrados en un proyecto (Pichler et al., 2006)(Ramesh et al., 2010). El desacuerdo entre los grupos de clientes afecta el rendimiento, especialmente en ciclos cortos de desarrollo (Ramesh et al., 2010).

- Falta de documentación

Sin la especificación del software, el cliente no tiene ningún documento de referencia que le permita comprender completamente su sistema; en cambio, existe una dependencia total de las personas que crearon el sistema para conocerlo (Ramesh et al., 2010). Si esto sucede, el cliente podría encontrarse en una situación en la que no conoce su propio sistema y no tiene ninguna herramienta (excepto del código fuente) para obtener este conocimiento. Por tanto, la documentación mínima es un desafío vital, en los métodos ágiles, para los equipos de desarrollo (Cao & Ramesh, 2008a) (Ramesh et al., 2010). Cada vez que hay un lapso de comunicación debido a cambios repentinos en los requerimientos, la falta de disponibilidad de representantes apropiados del cliente, la complejidad del proyecto y la falta de documentación, surgen múltiples problemas. Además, si se supone que los requerimientos deben comunicarse a los clientes en ubicaciones geográficas distribuidas y no en el sitio, se vuelve engorroso abordar tal situación con poca o ninguna documentación (Barta & Bhatnagar, 2019).

PI. 2. ¿Qué prácticas se siguen para superar estos desafíos?

- Enfoques propuestos

Contrariamente a la intuición inicial de que la mejor manera de combinar la Ingeniería de Requerimientos y desarrollo de software ágil sería modificar los procesos ágiles para apoyar las tareas de obtención de necesidades por parte del cliente, los resultados de la RSL han demostrado que agregar la actividad explícitamente a la agilidad es una posibilidad, pero otra posibilidad es hacer que la actividad de ingeniería de requerimientos sea más ágil. Una opción adicional consiste en olvidar todo y desarrollar una metodología completamente nueva que combine: el dinamismo de lo ágil con la disciplina de la ingeniería de requerimientos de software y los enfoques tradicionales.

- Enfoque ágil e ingeniería de requerimientos

Este enfoque abarca prácticas que se agregan a un proyecto ágil para permitir la actividad de ingeniería de requerimientos. Consiste en variar las actividades que generalmente se llevan a cabo en el proyecto ágil con prácticas tradicionales que se pueden insertar y usar tratando de

afectar lo menos posible a la agilidad del proceso general. Las fuentes incluidas en la RSL que siguen el enfoque ágil con ingeniería de requerimientos se describen en las siguientes referencias (Carlson & Matuzic, 2010)(Daneva et al., 2013)(Ernst et al., 2014b)(Lehman & Sharma, 2011).

Las prácticas asociadas a este enfoque son las siguientes:
Realizar análisis de impacto en cada iteración
Iteración cero
Cliente involucrado en tareas de identificación de necesidades
Roles dedicados
Uso del conocimiento
Ingeniería de requerimientos basada en la experiencia
Mejorar la calidad

- Categorización de prácticas

Anteriormente, se presentaron las prácticas agrupándolas atendiendo al enfoque general al que pertenecen. Pero puede surgir una categorización diferente siguiendo el espíritu de la clasificación de los desafíos. En este caso, se atiende a la naturaleza de las prácticas propuestas.

- Prácticas ágiles propias

En este caso, estas prácticas se conciben como procesos o principios a partir de enfoques ágiles. Esto consiste en transformar procedimientos ágiles para hacerlos compatibles con el trabajo de IR (Farid & Mitropoulos, 2012)(Lehman & Sharma, 2011)(Haugset & Stålhane, 2012)(Mahmud & Veneziano, 2011)(Ramesh et al., 2010)(Beck, 2004)(Daneva et al., 2013).Estas práticas son las siguientes:
Iteración cero
Priorización de los requerimientos
Proceso iterativo e incremental

- Prácticas de comunicación

En la práctica, la mayoría de los equipos ágiles tienen sustitutos o clientes sustitutos para desempeñar el papel de un cliente real, por ejemplo, propietarios de productos (Daneva et al., 2013). Otras organizaciones implementan la práctica del «desarrollador en el sitio» desplazándose un representante del desarrollador al sitio del cliente (Racheva et al., 2010).

Estas Estas prácticas se centran en mejorar la comunicación; persiguen habilitar el trabajo de obtención de requerimientos a través de la mejora de la comunicación entre el equipo y el cliente (Jun et al., 2010) (Beck, 2004) (Paetsch et al., 2003)(Cohn, 2009). Estas práticas son:
Reuniones de revisión
Cliente involucrado en tareas identificación de necesidades
Uso de herramientas de comunicación
El facilitador como habilitador de comunicación.

- Prácticas de calidad

En un enfoque similar a las prácticas de comunicación, este tipo de prácticas está orientado a mejorar la calidad del software creado, y en ese contexto se necesita unas reuniones de revisión (Farid & Mitropoulos, 2012)(Pichler et al., 2006)(Ramesh et al., 2010). Estas prácticas son:
Mejorar la calidad
Los atributos de calidad

- Definición de prácticas

Estas prácticas analizan las consecuencias de los cambios introducidos en cada iteración para identificar riesgos y abordarlos (Zaki & Moawad, 2010)(Ramesh et al., 2010).

- **Iteración cero**

Esta práctica consiste en incluir una iteración «cero» (anterior a las comunes) para llevar a cabo un análisis general del proyecto, crear diseños preliminares y establecer el contexto general del proyecto (Farid & Mitropoulos, 2012)(Lehman & Sharma, 2011)(Jun et al., 2010)(Zaki & Moawad, 2010)(Pichler et al., 2006)(Ramesh et al., 2010).

- **Reuniones de evaluación**

Esta práctica consiste en incluir la evaluación en las reuniones diarias. De esta forma, se convierte en una práctica conjunta de todo el equipo. Este enfoque combina el trabajo de consenso con el principio ágil de "trabajar juntos"(Jun et al., 2010)(Daneva et al., 2013)(Cho, 2009).

- **Uso de conocimiento**

Esto se refiere a la confianza en la experiencia de los miembros del equipo y también en la captura de conocimiento para su posterior reutilización o compra de soluciones predefinidas que satisfagan las necesidades del proyecto (Hruschka, 2018).

- **Mejorar la calidad**

Esta práctica consiste en remarcar la importancia de la calidad al momento de medir el éxito de un proyecto; también se refiere a medidas específicas tomadas para cuidar aspectos de calidad del software (Farid & Mitropoulos, 2012)(Pichler et al., 2006)(Ramesh et al., 2010).

- **Los atributos de calidad**

Esta práctica se basa en el uso de atributos de calidad como algo fundamental para obtener un análisis y diseño del software. La identificación de los atributos de calidad tiene lugar en reuniones donde el cliente puede estar presente (Ramesh et al., 2010)(Pichler et al., 2006).

- **Proceso iterativo e incremental**

Esto indica que las tareas de obtención e identificación de necesidades se abordan varias veces y, a veces, de forma incremental; esto es, enfocándose progresivamente en dominios más grandes. Cabe señalar que estas dos características pueden aparecer juntas o separadas, esto es, de formas iterativas, incrementales o ambas al mismo tiempo (Cao & Ramesh, 2008a)(Carlson & Matuzic, 2010)(Daneva et al., 2013)(Ernst et al., 2014a)(Daneva et al., 2013)(Racheva et al., 2010)(Sarkan et al., 2011).

- **Uso de herramientas de comunicación**

Hace referencia al uso de herramientas como un medio para mejorar la comunicación del equipo (usando un wiki, chat, entornos colaborativos). Esta práctica puede significar costos adicionales para adquirir e implementar estas herramientas (Farid & Mitropoulos, 2012).

- **Dividir equipos**

Esta práctica está indicada para superar las limitaciones relacionadas con los grandes equipos y los problemas de comunicación que esto conlleva. Por lo general, la división del equipo va junto con la descomposición del problema o con el trabajo concurrente (Zaki & Moawad, 2010)[29].

- **Enfoque híbrido**

Esta práctica intenta superar las dificultades que encuentran las metodologías ágiles en algunos contextos (por ejemplo, en dominios desconocidos). El enfoque consiste en seguir una metodología híbrida tradicional-ágil para cumplir con el dominio y al mismo tiempo aplicar principios ágiles cuando sea posible (Farid & Mitropoulos, 2012).

- **Evaluación y caracterización del proyecto**

Esta práctica consiste en evaluar y analizar el proyecto atendiendo a cinco factores (personal, dinamismo, cultura, tamaño y criticidad)(Boehm & Turner, 2003).

5. Conclusiones

Esta sección incluye el resumen del presente trabajo de investigación al revisar las preguntas de investigación.

Respecto a la PI 1: ¿Qué desafíos con respecto a ingeniería de requerimientos de software se encuentran en proyectos ágiles?

Se ha podido identificar con la expresión de idoneidad, por lo que varios documentos coinciden que su aplicación de los métodos ágiles se limita a cierto tipo de proyectos por ejemplo no aplica a proyectos de gran envergadura, conocimiento del dominio y complejidad del proyecto, así como al tamaño del equipo, ello por su alto nivel de coordinación y comunicación entre todos los miembros. Así mismo respecto al desafío del proyecto en sí, donde el objetivo es entregar software funcionando sin seguir necesariamente las pautas del proceso de calidad que propone la ISO o CMMI. Con respecto al desafío identificado respecto al equipo se debe precisar que las metodologías ágiles precisan de un equipo conformado por miembros con bastante experiencia, el cual deriva en algunos casos en la dependencia de estos, aquí también se precisa la comunicación fluida del equipo, todos ellos deben compartir una misma visión del proyecto. También el cliente está involucrado como miembro del equipo, debe de compartir los ambientes de desarrollo a su vez debe ser el facilitador en el desarrollo del proyecto.

Respecto a la PI 2: ¿Qué prácticas se siguen para superar estos desafíos?

Sobre las prácticas o enfoque que se proponen podemos resumir de la siguiente manera.

El enfoque ágil e IR, las prácticas asociadas a este enfoque son las siguientes: Realizar análisis de impacto en cada iteración, iteración cero, cliente involucrado en tareas de identificación de necesidades, roles dedicados, uso del conocimiento, IR basada en la experiencia y mejora de la calidad del producto.

Prácticas ágiles propias. Iteración cero, priorización de los requisitos, proceso iterativo e incremental.

Prácticas de comunicación. Reuniones de revisión, cliente involucrado en tareas identificación de necesidades, uso de herramientas de comunicación, el facilitador como habilitador de comunicación.

Prácticas de calidad. Mejorar la calidad del producto, cumplir con los atributos de calidad.

6. Trabajo futuro

Luego de los resultados de la RSL, queda claro que se requiere mucho más trabajo empírico sobre el tema.

Parece aconsejable realizar experimentos controlados para evaluar los efectos causados por las técnicas propuestas y junto con las percepciones del equipo. El objetivo principal debe ser crear prácticas adaptativas que puedan usarse dependiendo del proyecto bajo ciertas condiciones.

Algunas de las métricas interesantes que podrían usarse para la evaluación de las prácticas podrían ser tiempo (medir diferentes procesos, o iteraciones, o todo el proceso de desarrollo), esfuerzo requerido para corregir defectos, sentimientos de los miembros del equipo (estrés, presión, felicidad, armonía, etc.), desviaciones en el presupuesto, satisfacción del cliente, etc.

Considerando, estas métricas deben tomarse tanto después como antes de introducir las prácticas propuestas para medir su impacto, y en aquellos casos en que esto no sea posible, las medidas deberían tomarse mediante la comparación entre proyectos con características similares.

Referencias bibliográficas

[1] Abdullah, N. N. B., Honiden, S., Sharp, H., Nuseibeh, B., & Notkin, D. (2011). Communication patterns of agile requirements engineering. Proceedings of the 1st Agile Requirements Engineering Workshop, AREW'11 - In Conjunction with ECOOP'11, October 2015.

[2] Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jäälinoja, J., Korkala, M., Koskela, J., Kyllönen, P., & Salo, O. (2004). Mobile-D: An agile approach for mobile application development. Proceedings of the Conference on Object-Oriented Programming Systems, Languages, and Applications, OOPSLA, 174–175.

[3] Barta, M., & Bhatnagar, A. (2019). A Research Study on Critical Challenges in Agile Requirements Engineering. International Research Journal of Engineering and Technology (IRJET), 6(6), 1214–1219.

[4] Beck, K. (2004). What is XP? In Addison-Wesley (Ed.), Extreme Programming Explained: Embrace Change (Segunda ed).

[5] Bjarnason, E., Wnuk, K., & Regnell, B. (2011). A case study on benefits and side-effects of agile practices in large-scale requirements engineering. Proceedings of the 1st Agile Requirements Engineering Workshop, AREW'11 - In Conjunction with ECOOP'11.

[6] Boehm, B., & Turner, R. (2003). Observations on balancing discipline and agility. Proceedings of the Agile Development Conference, ADC 2003, 32–39.

[7] Cao, L., & Ramesh, B. (2008a). Agile requirements engineering practices: An empirical study. IEEE Software, 25(1), 60–67.

[8] Carlson, D., & Matuzic, P. (2010). Practical agile requirements engineering. 13th Annual Systems Engineering Conference.

[9] Cho, J. (2009). a Hybrid Software Development Method for Large-Scale Projects: Rational Unified Process With Scrum. Issues In Information Systems, 340–348.

[10] Cohn, M. (2009). User Stories Applied: For Agile Software Development (I. Pearson Education (ed.)).

[11] Corral, L., Silliti, A., & Succi, G. (2013). Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface. Agile Software Development Processes for Mobile Systems: Accomplishment, Evidence and Evolution, 8093 LNCS(July 2016).

[12] Daneva, M., Van Der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., Ajmeri, N., Ramteerthkar, U., & Wieringa, R. (2013). Agile requirements prioritization in large-scale outsourced system projects: An empirical study. Journal of Systems and Software, 86(5), 1333–1353.

[13] Ernst, N. A., Borgida, A., Jureta, I. J., & Mylopoulos, J. (2014a). Agile requirements engineering via paraconsistent reasoning. Information Systems, 43(July), 100–116.

[14] Farid, W. M., & Mitropoulos, F. J. (2012). NORMATIC: A visual tool for modeling nonfunctional requirements in agile processes. Conference Proceedings - IEEE SOUTHEASTCON, 978.

[15] Haugset, B., & Stålhane, T. (2012). Automated acceptance testing as an agile requirements engineering practice. Proceedings of the Annual Hawaii International Conference on System Sciences, May 2014, 5289–5298.

[16] Hruschka, P. (2018). Handbook of RE @ Agile According to the IREB Standard. October, 1–95.

[17] Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior, 51, 915–929.

[18] Jun, L., Qiuzhen, W., & Lin, G. (2010). Application of agile requirement engineering in modest-sized information systems development. Proceedings - 2010 2nd WRI World Congress on Software Engineering, WCSE 2010, 2(1), 207–210.

[19] Kitchenham, B. A., & Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering (Software Engineering Group, Department of Computer Science, Keele .... Technical Report EBSE 2007- 001. Keele University and Durham University Joint Report, January.

[20] Lehman, T. J., & Sharma, A. (2011). Software development as a service: Agile experiences. Proceedings - 2011 Annual SRII Global Conference, SRII 2011, 749–758.

[21] Mahmud, I., & Veneziano, V. (2011). Mind-mapping: An effective technique to facilitate requirements engineering in agile software development. 14th International Conference on Computer and Information Technology, ICCIT 2011, Iccit, 157–162.

[22] Okesola, J., Adebiyi, M., Okokpujie, K., Odepitan, D., Goddy-Worlu, R., Iheanetu, O., Omogbadegun, Z., & Adebiyi, A. (2019). A systematic review of requirement engineering practices in agile model. International Journal of Mechanical Engineering and Technology, 10(2), 671–687.

[23] Paetsch, F., Eberlein, A., & Maurer, F. (2003). Requirements engineering and agile software development. Proceedings of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE, 2003-Janua(July), 308–313

[24] Pichler, M., Rumetshofer, H., & Wahler, W. (2006). Agile requirements engineering for a social insurance for occupational risks organization: A case study. Proceedings of the IEEE International Conference on Requirements Engineering, July 2014, 251–256.

[25] Racheva, Z., Daneva, M., & Herrmann, A. (2010). A conceptual model of client-driven agile requirements prioritization: Results of a case study. ESEM 2010 - Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement.

[26] Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile requirements engineering practices and challenges: an empirical study. Information Systems Journal, 20(5), 449–480.

[27] Sarkan, H. M., Ahmad, T. P. S., & Bakar, A. A. (2011). Using JIRA and redmine in requirement development for Agile methodology. 2011 5th Malaysian Conference in Software Engineering, MySEC 2011, 408–413.

[28] The Standish Group. (2016). The Standish Group. 2015.

[29] Zaki, K. M., & Moawad, R. (2010). A hybrid disciplined agile software process model. INFOS2010 - 2010 7th International Conference on Informatics and Systems, April 2010.