PROPUESTA DE GUIA PARA LA GESTIÓN DE RIESGOS

BAJO EL MARCO DE TRABAJO SCRUM

Trabajo de Grado para optar al título de Especialistas en Gerencia de Proyectos

Yohana María Orozco Cardona

Gustavo Alonso Naranjo Benavides

Asesor

PhD. Marcelo Torres Arango

Corporación Universitaria Minuto de Dios

Bello

2014

TABLA DE CONTENIDO

INTRODUCCIÓN

El desarrollo de este trabajo consistió en construir una metodología de gestión de riesgos que se pudiera implementar en proyectos de software bajo la metodología Ágil SCRUM, pero que a la vez, permitiera que dichos proyectos siguieran manteniendo las cualidades y naturaleza ofrecida por las metodologías Ágiles.

SCRUM es un marco ampliamente utilizado para acelerar el desarrollo de software, con nuevos paradigmas de gestión, enfoques orientados a la agilidad y empoderamiento de equipos de trabajo orientados al logro y objetivos.

El procedimiento o guía que se propuso, será entonces, un proceso que se adicione a los procesos de gestión de proyectos de software, que se lleven en cualquier empresa, pero que tengan como base, el marco de trabajo SCRUM.

Los objetivos específicos que se plantearon fueron: 1ro - Buscar información sobre los avances que se han propuesto para la gestión de riesgos en el marco de trabajo Scrum, 2do - plantear las características requeridas para la gestión de riesgos en marcos de trabajo Ágiles y 3ro - establecer un procedimiento para realizar gestión de riesgos bajo el marco de trabajo Scrum.

El alcance del trabajo cubrió los aspectos básicos de la gestión de riesgos: Identificación, Evaluación, Mitigación y Control.

Se abordó la investigación pertinente, con diversos autores, que han expuesto sus teorías acerca de la importancia de la gestión de riesgos, sobre autores que explican por qué se deben gestionar riesgos en metodologías ágiles como SCRUM, y sobre autores que sostienen, que en particular

SCRUM, no requiere el manejo de riesgos como tal, debido a que dicha gestión está implícita en ella.

Se introdujo una explicación sobre metodologías Ágiles, y propiamente, sobre el marco de trabajo SCRUM, específicamente, sobre cómo se debe trabajar con este tipo de metodologías, sus principales componentes y características y el nicho hacia el que están dirigidas.

Se abordó el estado actual de la gestión de riesgos, en metodologías Ágiles y que tipos de metodologías se usan actualmente.

También se explicó por qué en particular se usó SCRUM, como metodología Ágil para la propuesta de guía metodológica y se introdujo al detalle, cada una de las fases que se deben considerar en la gestión de riesgos, desde el enfoque propuesto con el SCRUM Alliance.

Se explicó de forma detallada, como se relacionan las diferentes etapas implícitas en SCRUM contra las fases que se deben usar para la gestión de riesgos, como base para la propuesta metodológica. El uso apropiado de taxonomías como medio para identificar riesgos en base a preguntas, la evaluación numérica en base a probabilidades para determinar la posibilidad de los eventos y las herramientas de clasificación requeridas para abordar los riesgos.

En el desarrollo del trabajo se utilizó como tipo de estudio, el descriptivo soportada con investigaciones en papers, tesis, libros y documentos web calificados sobre gestión de riesgos.

Como método de investigación se usó uno mixto (cualitativo y cuantitativo), utilizando fuentes secundarias como herramientas para la investigación y desarrollo del proyecto.

Los resultados obtenidos, permitieron afirmar, la conveniencia de implementar este tipo de metodologías en proyectos basados en SCRUM y sus beneficios, la viabilidad para realizar

gestión de riesgos en metodologías Ágiles sin perder la esencia o naturaleza del proceso, entregando como producto final una guía para gestionar riesgos en proyectos de software ejecutados con Scrum, que permite solucionar el problema planteado en este trabajo y cumplir el objetivo general establecido.

1. PROBLEMA DE INVESTIGACIÓN

1.1 PLANTEAMIENTO:

Hoy en día se tiene un alto grado de certeza sobre la importancia de realizar una adecuada gestión de riesgos en cualquier proyecto, particularmente, el sector de la construcción de software ha buscado hacer uso de estas buenas prácticas de gestión de riesgos en el desarrollo de software.

A nivel mundial, se están generando nuevas metodologías y marcos de trabajo, que buscan dar enfoques diferentes a los esquemas tradicionales de desarrollo de software. No siempre estas metodologías cubren de forma adecuada la gestión de riesgos.

Dentro de estas metodologías ha cobrado fuerza en especial el marco de trabajo Scrum.

> Sin embargo, existen expertos que afirman que el desarrollo ágil de software, debido a su naturaleza iterativa, implícitamente hace la gestión de riesgos una parte del ciclo de vida del proyecto. Los miembros de la comunidad Ágil discutieron si se requiere la gestión del riesgo explícito, la capacidad de Scrum para gestionar todo tipo de riesgo y que debe hacer la gestión de riesgos. (Hazrati, 2008)

Se evidencia entonces una tendencia en los proyectos de software en los que se aplican metodologías ágiles (como Scrum), donde la gestión de riesgos por parte del equipo de trabajo no sigue pautas claras y definidas que tengan como base estándares ya establecidos.

Se hace necesario plantear un modelo que permita gestionar riesgos de forma adecuada sin cambiar la filosofía propuesta por estas metodologías ágiles.

1.2 PREGUNTA DE INVESTIGACIÓN:

Dentro de las metodologías ágiles, particularmente en el marco de trabajo Scrum, ¿qué elementos son necesarios para proponer una guía de gestión de riesgos, para que un equipo de trabajo pueda identificar, analizar, responder, seguir y controlar los riesgos en un proyecto de software?

1.3 PREGUNTAS COMPLEMENTARIAS:

1.3.1 ¿Qué avances se han propuesto para la gestión de riesgos por parte de la comunidad Ágil?

1.3.2 ¿Qué aspectos se deben considerar para la gestión de riesgos en Scrum?

1.3.3 ¿Qué mecanismos se pueden proponer para gestionar los riesgos en Scrum?

1.4 OBJETIVOS

1.4.1 Objetivo General.

Proponer una guía para la identificación, el análisis, la respuesta, el seguimiento y el control de riesgos en un proyecto de software bajo el marco de trabajo Scrum.

1.4.2 Objetivos Específicos.

1.4.2.1 Buscar información sobre los avances que se han propuesto para la gestión de riesgos en el marco de trabajo Scrum.

1.4.2.2 Plantear las características requeridas para la gestión de riesgos en marcos de trabajo Ágiles.

1.4.2.3 Establecer un procedimiento para realizar gestión de riesgos bajo el marco de trabajo Scrum.

1.4 ALCANCE

Este proyecto de investigación tiene como alcance proponer una guía que permita identificar, analizar, responder, seguir y controlar los riesgos en proyectos de software que se ejecutan bajo el marco de trabajo Scrum.

No se construyeron modelos o métodos complejos, sino una guía o propuesta sencilla que sea útil y de fácil manejo por las personas que la deseen incorporar durante el desarrollo de sus proyectos, siempre y cuando utilicen metodologías ágiles, como el marco de trabajo Scrum.

El entregable no garantiza que la gestión de los riesgos estará completamente controlada y que los riesgos desaparecen en los proyectos, sino que sirve como herramienta de apoyo.

No se construyó un software ni otra herramienta tecnológica con este proyecto, solo unas pautas, procedimiento o guía que se puede utilizar de manera sencilla en la gestión de riesgos de proyectos de software ágiles.

1.5 JUSTIFICACIÓN DE LA INVESTIGACIÓN

Este proyecto de investigación está aportando beneficios a nivel académico en el área de la gestión de riesgos, para proyectos de software que se desarrollan utilizando metodologías ágiles en particular el marco de trabajo Scrum, debido a que la gestión de los riesgos en estas metodologías está inmerso en el día a día de la ejecución del proyecto, pero no se tiene herramientas o espacios propios para la gestión de los riesgos identificados. Al final de este proyecto se realizó un aporte con la guía que se propuso en una temática donde existen pocos modelos o métodos que se están utilizando en la práctica laboral, y que permitirá a otras investigaciones o trabajamos futuros de otros estudiantes utilizar la guía propuesta y los hallazgos realizados en este trabajo en sus propuestas.

El tema seleccionado es una metodología de trabajo para aplicar en el desarrollo de software, la cual en los últimos años ha estado incursionando y aplicándose en empresas tanto de servicios de tecnología y TI, como empresas de otros sectores que tienen gran cantidad de personas y proveedores dentro de sus áreas de desarrollo y tecnología, los cuales buscan mejorar los tiempos de entrega a los usuarios finales que solicitan los productos de software a la medida, y que se encontraban muy insatisfechos por los retrasos en los cronogramas, y la falta de flexibilidad para incluir cambios que el mismo negocio o cambios de ley les exigen durante el desarrollo del proyecto o de los requerimientos de software solicitados.

Scrum es una metodología ágil con muchas bondades, pero que también necesita gestionar los riesgos que pueden surgir durante el desarrollo de los proyectos. Por esta razón en esta

investigación deseamos enfocarnos dentro de Scrum en los modelos para gestionar riesgos, mejorando cumplimiento y calidad en los productos finales entregados.

En la práctica de Scrum se realiza continuamente una validación de riesgos de forma indirecta e informal que puedan surgir en el día a día, lo cual no garantiza que se estén controlando y mitigando los riesgos, o que se lleven a cabo las acciones preventivas o correctivas necesarias para minimizar al máximo el riesgo sin afectar el proyecto.

La propuesta de entregar una guía para identificar, analizar, responder, seguir y controlar riesgos en el marco de trabajo Scrum, aportará control a la gestión de riesgos dentro de la metodología, beneficiando el proceso de desarrollo de software.

La gestión de los riesgos es necesaria en el desarrollo de cualquier proyecto incluyendo los de software, los cuales son muy cambiantes y pueden tener actividades no planeadas y/o externas que afecten los tiempos o la calidad del desarrollo del proyecto.

Por esta razón se considera que es importante investigar sobre la gestión de riesgos dentro de un modelo de trabajo ágil, que también deben controlarse y que está siendo utilizado en nuestro medio por sus beneficios.

2. MARCO DE REFERENCIA

2.1 MARCO TEÓRICO

Para la comprensión del problema de investigación planteado en este proyecto, a continuación se explican los conceptos relacionados en la pregunta problema a resolver, para contextualizar y explicar los conceptos técnicos propios del tema, facilitando la comprensión de esta investigación y el resultado presentado.

A continuación se listan conceptos teóricos sobre las variables relacionadas en el problema planteado.

2.1.1 METODOLOGÍAS ÁGILES.

Según el autor Borrego quien habla de manera general para definir que son las metodologías ágiles, dice:

> Son metodologías normalmente orientadas a proyectos de software, que aportan una elevada simplificación de los procedimientos de ingeniería de software sin renunciar a las prácticas esenciales para asegurar la calidad del producto, pero enfocándose muy fuertemente en la gente y en los resultados palpables.
>
> Por otro lado, todas las metodologías ágiles se basan en el "Manifiesto por el desarrollo Ágil de Software" creado por 17 expertos en la industria del software,

cuyo documento contempla 12 principios y 4 valores que son los ejes rectores de los procesos de dichas metodologías y de las actitudes que se esperan en un equipo de trabajo que esté bajo el enfoque ágil.

Para entender de una mejor manera la naturaleza de las metodologías ágiles, se muestra la siguiente tabla comparativa (Tabla 1) entre las metodologías tradicionales y las ágiles.

No se encuentran elementos de tabla de ilustraciones.	**Tradicional**
Basadas en prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparadas para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo).	Impuestas externamente
Pocos artefactos y roles	Más artefactos y roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
El cliente se considera parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo solo en reuniones
Grupos pequeños (menos de 10 integrantes)	Grupos grandes y a veces distribuidos

y trabajando en el mismo sitio	

***TABLA 1* - COMPARACIÓN ENTRE LAS METODOLOGÍAS ÁGILES Y LAS TRADICIONALES**

Metodologías ágiles más utilizadas en el ámbito de la industria del software son:

- Combinación de Scrum, XP y/o una metodología personalizada
- XP
- Metodologías personalizadas o híbridas.
- Dynamic Systems Development Method. (Borrego, Portela, Tolano, Cruz, Amavizca y Vázquez, 2013)

A continuación se explica que es el manifiesto ágil mencionado anteriormente y en el cual se fundamentan las metodologías agiles como Scrum, además de los conceptos generales sobre el tema:

Según la autora Gracia Peña una definición de Manifiesto Ágil es:

Este manifiesto surgió en 2001 en una reunión celebrada en SnowBird, Utah. Su principal impulsor fue Kent Beck. Aunque muchas de las metodologías que se tratan en este proyecto son anteriores al manifiesto ágil, se puede decir que a partir de la reunión surgió un movimiento que no ha parado de captar adeptos tanto individualmente como empresas. Además, se creó la "alianza ágil" cuya finalidad es promover el desarrollo ágil en los proyectos.

Principios:

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
- Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.
- Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Características de Scrum:

- Se trabaja en iteraciones de 1 a 4 semanas, donde se debe acabar con un producto entregable.
- El equipo es auto organizado, los coordinadores y clientes deben trabajar en todo momento con el equipo de desarrollo, facilitando las tareas y resolviendo dudas.
- Se deben tener unos requisitos perfectamente priorizados reflejando el valor del negocio.
- Se debe mantener un ritmo de trabajo constante, permite que no haya descuidos y retrasos en el sprint.

Roles:

- Product owner: Dueño del producto
- Scrum master: Es el coordinador del equipo, controla al equipo para la consecución de los objetivos.
- Equipo: Desarrolladores del producto.

Reuniones:

- Planificaciones del sprint (1 hora por semana/iteración). Debe finalizar con un objetivo claro de lo que se va abordar y con el backlog adecuado, el equipo elegirá los ítems que consideran que pueden realizar y los dividirán entre todos.
- Reunión diaria (10-15 min). Cada componente del equipo comenta que está realizando, las tareas que ha terminado y si se tiene alguna dificultad.
- Revisión del sprint (1 hora por semana/iteración). Una vez finalizado el sprint se debe realizar una reunión para comprobar el producto entregado junto con el cliente. Es el momento de saber que se está construyendo.
- Retrospectiva (1 hora). En esta reunión se deben ver que se puede mejorar dentro del equipo, aquí se analiza la forma en la que se está trabajando.

Kanban:

Es una palabra Japonesa que literalmente significa “Tarjetas visuales”. Estos métodos ya tienen una larga trayectoria en las cadenas de producción, transmitido a desarrollo de software es bastante reciente de 2004.

Es un método visual muy recomendado para gestionar proyectos donde los requisitos cambian constantemente. También es útil en planificaciones y estimaciones de un equipo se alarguen o dejen de ser productivas así como cuando un equipo no se puede comprometer a trabajar en iteraciones fijas.

Se debe disponer del panel de Kanban, donde figuren las etapas de la iteración desde el principio hasta el final.

La primera columna estará el backlog o listado de tareas a realizar, una buena práctica es dividir las tareas en cargas de trabajo similares.

Es importante limitar al número de tareas permitidas por cada columna, de esta forma se eliminan posibles cuellos de botella y los problemas que puedan ocasionar un ritmo constante de trabajo.

Medir el tiempo empleado en una iteración completa, es importante conocer cuándo se ha tardado en realizar la tarea y tomarse su tiempo en analizar cómo reducir ese tiempo. Medir el tiempo también proporciona mayor facilidad para estimar y predecir futuras tareas.

El principal hito de Scrum es gestionar proyectos basados en pequeños grupos de trabajo, de cuatro a nueve personas, desarrollándolo de una forma iterativa y constante. Para un buen funcionamiento se necesita de una comunicación fluida y diaria, de una mejora continua y la apuesta por la calidad del código de todos los miembros del equipo. Son normas muy sencillas, muchas de ellas de sentido común, pero no hay que olvidar que precisamente la apuesta por lo sencillo y lo ágil es lo que proporciona mayor valor a Scrum.

A continuación, se aporta una descripción formal de Scrum según Ken Schwaber and Jeff Sutherland, en The Scrum Guide (www.scrum.org).

Scrum se fundamenta en la teoría empírica de control de procesos, o empirismo. El empirismo asegura que el conocimiento procede de la experiencia y de tomar decisiones basándose en lo que se conoce. Scrum emplea una aproximación iterativa e incremental para optimizar la predictibilidad y controlar el riesgo. Tres pilares soportan toda implementación del control empírico de procesos: transparencia, inspección y adaptación. (Gracia, 2013)

La apreciación sobre las metodologías ágiles y particularmente sobre SCRUM según Collar es:

> Tradicionalmente el software ha sido un instrumento para que las organizaciones mejoraran su productividad interna a partir de acciones de automatización. Sin embargo, este enfoque está cambiando en la medida que el software es utilizado como plataforma para mejorar, o incluso producir, los ingresos y por lo tanto estar sujeto a constantes presiones competitivas para producir soluciones innovadoras en tiempos cortos. Este es un contexto muy volátil donde la metodología de desarrollo debe permitir cambios rápidos.
>
> Desde que Ken Beck desarrolló los fundamentos de las metodologías ágiles hasta su masiva adopción al presente han proliferado descripciones en la bibliografía sobre la plataforma conceptual, espacios de utilización, marco de proceso, fortalezas y debilidades, métricas, evaluación económica y ejemplos de aplicaciones prácticas en distintos ámbitos.
>
> El enfoque ágil es capturado en su esencia en el Manifiesto Ágil cuyos principales pilares de valor son priorizar a los individuos y sus interacciones por encima del proceso y las herramientas, utilizar el software mismo como documentación, cooperar con el cliente por encima de negociar con él y, quizás uno de los aspectos más relevantes a este trabajo, responder al cambio en vez de seguir un plan establecido.
>
> Como parte de su evolución se ha delimitado cuando es más apropiado utilizar metodologías ágiles y cuando no; así como métodos para utilizarlas en diferentes

sistemas organizacionales, principios de gestión e incluso evaluarlas rigurosamente con modelos de referencia de calidad como SEI-CMMI.

Erdogmus y Favaro aplicaron principios de análisis económico para demostrar que Extreme Programming (XP) crea más valor económico que otros métodos de desarrollo de software tradicionales.

Maller discutió la justificación teórica para hacer el mapa entre los requerimientos de una organización acreditada como SEI-CMMI de Nivel 5 y XP; al mismo tiempo Banerjee documenta los aspectos a considerar necesarios para operar como metodologías ágiles con equipos remotos tal como los encontrados en operaciones off-shore. Otros autores sustentan también la relevancia de la aplicación de estás metodologías al abordaje de las operaciones off-shore. (Colla, 2012)

A continuación se complementa esta sección con otros conceptos sobre las metodologías ágiles más destacadas, diferentes a Scrum, que permita aclarar cómo funcionan, ver sus diferencias y similitudes.

XP (eXtreme Programming)

Típicamente un proyecto con XP lleva 10 a 15 ciclos o iteraciones.

Fases

Fase de exploración

Es la fase en la que se define el alcance general del proyecto. En esta fase, el cliente define lo que necesita mediante la redacción de sencillas “historias de

usuario". Los programadores estimas los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que están basadas en datos de muy alto nivel), y podrían variar cuando se analicen en más detalle en cada iteración.

Esta fase dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado.

Fase de planificación

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas que se detallará en la sección "Reglas y Practicas".

Fase de iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo.

Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

Fase de puesta en producción

Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta tanto no se tenga la funcionalidad completa.

En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste.

Reglas de XP

Planificación

La metodología XP plantea la planificación como un dialogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores o gerentes. El proyecto comienza recopilando "Historias de usuarios", las que sustituyen a los tradicionales "casos de uso". Una vez obtenidas las "historias de usuarios", los programadores evalúan rápidamente el tiempo de desarrollo de cada una.

Si alguna de ellas tiene "riesgos" que no establecer con certeza la complejidad del desarrollo, se realizan pequeños programas de prueba ("spikes"), para reducir estos riesgos. Una vez realizadas estas estimaciones, se organiza una reunión de planificación, con los diversos actores del proyecto (cliente, desarrolladores, gerentes), a los efectos de establecer un plan o cronograma de entregas ("Release Plan") en los que todos estén de acuerdo. Una vez acordado este cronograma,

comienza una fase de iteraciones, en dónde en cada una de ellas se desarrolla, prueba e instala unas pocas “historias de usuarios”.

Los planes en XP se diferencian de las metodologías tradicionales en tres aspectos:

• Simplicidad del plan. No se espera que un plan requiera de un “gurú” con complicados sistemas de gerenciamiento de proyectos.

• Los planes son realizados por las mismas personas que realizarán el trabajo.

• Los planes no son predicciones del futuro, sino simplemente la mejor estimación de cómo saldrán las cosas. Los planes son útiles, pero necesitan ser cambiados cuando las circunstancias lo requieren. De otra manera, se termina en situaciones en las que el plan y la realidad no coinciden, y en estos casos, el plan es totalmente inútil.

Los conceptos básicos de esta planificación son los siguientes:

Historias de usuario

Las “Historias de usuarios” sustituyen a los documentos de especificación funcional, y a los “casos de uso”. Estas “historias” son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido. Las historias de usuario deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo.

Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios.

Las historias de usuarios deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia.

Plan de entregas (Release Plan)

El cronograma de entregas establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los actores del proyecto (cliente, desarrolladores, gerentes, etc.).

XP denomina a esta reunión "Juego de planeamiento" ("Planning game"), pero puede denominarse de la manera que sea más apropiada al tipo de empresa y cliente (por ejemplo, Reunión de planeamiento, "Planning meeting" o "Planning workshop").

Típicamente el cliente ordenará y agrupará según sus prioridades las historias de usuario. El cronograma de entregas se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores.

Después de algunas iteraciones es recomendable realizar nuevamente una reunión con los actores del proyecto, para evaluar nuevamente el plan de entregas y ajustarlo si es necesario.

Plan de iteraciones

Las historias de usuarios seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden preestablecido.

Al comienzo de cada ciclo, se realiza una reunión de planificación de la iteración. Cada historia de usuario se traduce en tareas específicas de programación.

Asimismo, para cada historia de usuario se establecen las pruebas de aceptación. Estas pruebas se realizan al final del ciclo en el que se desarrollan, pero también al final de cada uno de los ciclos siguientes, para verificar que subsiguientes iteraciones no han afectado a las anteriores.

Las pruebas de aceptación que hayan fallado en el ciclo anterior son analizadas para evaluar su corrección, así como para prever que no vuelvan a ocurrir.

Reuniones diarias de seguimiento

El objetivo de tener reuniones diarias es mantener la comunicación entre el equipo, y compartir problemas y soluciones. En la mayoría de estas reuniones, gran parte de los participantes simplemente escuchan, sin tener mucho que aportar. Para no quitar tiempo innecesario del equipo, se sugiere realizar estas reuniones en círculo y de pie.

Diseño

La metodología XP hace especial énfasis en los diseños simples y claros. Los conceptos más importantes de diseño en esta metodología son los siguientes:

Simplicidad

Un diseño simple se implementa más rápidamente que uno complejo. Por ello XP propone implementar el diseño más simple posible que funcione. Se sugiere nunca

adelantar la implementación de funcionalidades que no correspondan a la iteración en la que se esté trabajando.

Características fundamentales del código: Testeable, legible, comprensible y explicable.

Metáforas

Una “metáfora” es algo que todos entienden, sin necesidad de mayores explicaciones.

La metodología XP sugiere utilizar este concepto como una manera sencilla de explicar el propósito del proyecto, y guiar la estructura y arquitectura del mismo. Por ejemplo, puede ser una guía para la nomenclatura de los métodos y las clases utilizadas en el diseño del código. Tener nombres claros, que no requieran de mayores explicaciones, redunda en un ahorro de tiempo.

Es muy importante que el cliente y el grupo de desarrolladores estén de acuerdo y compartan esta “metáfora”, para que puedan dialogar en un “mismo idioma”. Una buena metáfora debe ser fácil de comprender para el cliente y a su vez debe tener suficiente contenido como para que sirva de guía a la arquitectura del proyecto.

Solución “spike”

Una solución “spike”, es una solución muy simple para plantear posible soluciones, de manera, que solamente se aborda el problema en concreto y se aísla de otro tipo de preocupaciones.

Refactorización

La recodificación consiste en escribir nuevamente parte del código de un programa, sin cambiar su funcionalidad, a los efectos de hacerlo más simple, conciso y/o entendible.

Muchas veces, al terminar de escribir un código de programa, pensamos que, si lo comenzáramos de nuevo, lo hubiéramos hecho en forma diferente, más clara y eficientemente.. Las metodologías de XP sugieren recodificar cada vez que sea necesario. Si bien, puede parecer una pérdida de tiempo innecesaria en el plazo inmediato, los resultados de ésta práctica tienen sus frutos en las siguientes iteraciones, cuando sea necesario ampliar o cambiar la funcionalidad. La filosofía que se persigue es, como ya se mencionó, tratar de mantener el código más simple posible que implemente la funcionalidad deseada.

Implementación

Cliente disponible

Uno de los requisitos de XP es tener al cliente disponible. No solo para ayudar al equipo de desarrollo, sino para ser parte del mismo. Todas las fases de XP requieren comunicación con el cliente.

Las historias de usuario son escritas por el cliente con la ayuda de los desarrolladores, además de establecer la prioridad de las mismas. Su presencia asegura que los desarrollos cubren toda la funcionalidad descrita.

Durante la reunión de planificación el cliente negocia las historias que se incluirán en la próxima entrega, así como su duración.

El cliente es imprescindible a la hora de realizar pruebas funcionales, en caso de error él será el encargado de decidir si el código puede pasar a producción o no.

Estándares de codificación

Todos los programadores deben escribir y documentar el código en la misma manera.

El código debe seguir los estándares de codificación. Las normas de codificación ayudan a mantener el código legible y fácil de mantener y refactorizar.

Programación en parejas

XP promueve que todo el código sea escrito en parejas trabajando en el mismo ordenador. La programación en parejas incrementa la calidad del código sin impactar en la fecha de entrega.

En contra de lo que parece, dos personas que trabajan en un mismo equipo añadirán la misma funcionalidad que dos personas trabajando por separado, excepto que el código será de mucha mayor calidad.

Integración secuencial

Todos los desarrolladores necesitan trabajar siempre con la "última versión". Realizar cambios o mejoras sobre versiones antiguas causan graves problemas, y retrasan al proyecto. Es por eso que XP promueve publicar lo antes posible las nuevas versiones, aunque no sean las últimas, siempre que estén libres de errores. Idealmente, todos los días deben existir nuevas versiones publicadas.

Para evitar errores, solo una pareja de desarrolladores puede integrar su código a la vez.

Propiedad colectiva del código

La propiedad colectiva anima a todos los miembros del equipo a aportar nuevas ideas.

Cualquier desarrollador puede cambiar líneas de código para añadir funcionalidad, solucionar errores, mejorar el diseño o refactorizar el código.

Cuando una persona realiza un desarrollo tiene que subir al repositorio el código más sus pruebas unitarias funcionando al 100%, de manera, que otra persona que se lo descargue puede confiar en que tiene un código que funciona y desarrollar a partir del mismo.

Ritmo constante

La metodología XP indica que debe llevarse un ritmo sostenido de trabajo. Anteriormente, ésta práctica se denominaba “Semana de 40 horas”. Sin embargo, lo importante no es si se trabajan, 35, 40 o 42 horas por semana. El concepto que se desea establecer con esta práctica es el de planificar el trabajo de manera de mantener un ritmo constante y razonable, sin sobrecargar al equipo.

Pruebas Unitarias

Las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Por otra parte, como se mencionó anteriormente, las pruebas deben ser definidas antes de

realizar el código (“Test-driven programming”). Que todo código liberado pase correctamente las pruebas unitarias es lo que habilita que funcione la propiedad colectiva del código. En este sentido, el sistema y el conjunto de pruebas debe ser guardado junto con el código, para que pueda ser utilizado por otros desarrolladores, en caso de tener que corregir, cambiar o recodificar parte del mismo.

Kanban

Su objetivo es gestionar de manera general como se van completando tareas, pero en los últimos años se ha utilizado en la gestión de proyectos de desarrollo software.

Las principales reglas de Kanban son las siguientes:

1. Visualizar el trabajo y las fases del ciclo de producción o flujo de trabajo.
2. Determinar el límite del “trabajo en curso” (WIP - Work In Progress).
3. Medir el tiempo en completar una tarea (Lead time).

Visualizar el trabajo y las fases del ciclo de producción o flujo de trabajo

Kanban se base en el desarrollo incremental, dividiendo el trabajo en partes. Una de las principales aportaciones es que utiliza técnicas visuales para ver la situación de cada tarea, y que se representa en pizarras llenas de post-it.

El trabajo se divide en partes, normalmente cada una de esas partes se escribe en un post-it y se pega en una pizarra. Los post-it suelen tener información variada, si

bien, aparte de la descripción, debieran tener la estimación de la duración de la tarea.

La pizarra tiene tantas columnas como estados por los que puede pasar la tarea (ejemplo, en espera de ser desarrollada, en análisis, en diseño, etc.).

Determinar el límite del trabajo en curso (Work In Progress)

Quizás una de las principales ideas del Kanban es que el trabajo en curso (Work In Progress) debería estar limitado, es decir, que el número de tareas que se pueden realizar en cada fase debe ser algo conocido. Independientemente de si un proyecto es grande o Guía Comparativa de Metodologías Ágiles

44 pequeño, simple o complejo, hay una cantidad de trabajo óptima que se puede realizar sin sacrificar eficiencia, por ejemplo, puede ser que realizar diez tareas a la vez nos lleve una semana, pero hacer dos cosas a la vez nos lleve sólo unas horas, lo que nos permite hacer quince tareas en la semana.

En Kanban se debe definir cuantas tareas como máximo puede realizarse en cada fase del ciclo de trabajo (ejemplo, como máximo 4 tareas en desarrollo, como máximo 1 en pruebas, etc.), a ese número de tareas se le llama límite del "work in progress". A esto se añade otra idea tan razonable como que para empezar con una nueva tarea alguna otra tarea previa debe haber finalizado.

Medir el tiempo en completar una tarea (Lead time)

El tiempo que se tarda en terminar cada tarea se debe medir, a ese tiempo se le llama "lead time". El "lead time" cuenta desde que se hace una petición hasta que se hace la entrega.

Aunque la métrica más conocida del Kanban es el “lead time”, normalmente se suele utilizar también otra métrica importante: el “cycle time”. El “cycle time” mide desde que el trabajo sobre una tarea comienza hasta que termina. Si con el “lead time” se mide lo que ven los clientes, lo que esperan, y con el “cycle time” se mide más el rendimiento del proceso.

Puede haber más métricas, pero las anteriores son las realmente importantes y necesarias para el control y mejora continua.

Roles

La metodología Kanban no prescribe roles. Tener un papel asignado y las tareas asociadas a dicho papel crean una identidad en el individuo. Por lo tanto, pedir que adopten un nuevo papel o un nuevo puesto de trabajo puede ser entendido como un ataque a su identidad. Habría una resistencia al cambio. Kanban trata de evitar esa resistencia emocional, entiende que la ausencia de papeles es una ventaja para el equipo.

Scrumban

Scrumban es una metodología derivada de los métodos de desarrollo Scrum y Kanban.

De Scrum

• Roles: Cliente, equipo (con los diferentes perfiles que se necesiten).

• Reuniones: reunión diaria.

• Herramientas: pizarra

De Kanban

• Flujo visual

• Hacer lo que sea necesario, cuando sea necesario y solo la cantidad necesaria.

• Limitar la cantidad de trabajo (WIP)

• Optimización del proceso.

Es un modelo de desarrollo especialmente adecuado para proyectos de mantenimiento o proyectos en los que las historias de usuarios (requisitos del software) varíen con frecuencia o en los cuales surjan errores de programación inesperados durante todo el ciclo de desarrollo del producto. Para estos casos, los sprints (periodos de duración constante en los cuales se lleva a cabo un trabajo en sí) de la metodología Scrum no son factibles, dado que los errores/impedimentos que surgirán a lo largo de las tareas son difíciles de determinar y por lo tanto, no es posible estimar el tiempo que conlleva cada historia. Por ello, resulta más beneficioso adoptar flujo de trabajo continuo propio del modelo Kanban. Aunque hay diferencias entre ambos métodos, por ejemplo, las reglas de Kanban son muchas menos que las de Scrum, Kanban no define iteraciones (Sprints), Kanban limita explícitamente las tareas que se pueden realizar por fase (con el límite de work in progress), mientras que Scrum lo hace de manera indirecta por medio del sprint planning, etc. (Pérez, 2012)

2.1.2 Porque Scrum.

A pesar de recibir XP la mayoría de la atención bibliográfica las organizaciones están crecientemente enfocando su atención en la metodología ágil denominada SCRUM, la cual aplica las mismas premisas conceptuales pero para resolver un problema ligeramente distinto como es el de desarrollo evolutivo de aplicaciones, el problema abordado ocupa, incidentalmente, una parte significativa de los recursos de la industria del desarrollo de software y este hecho explica mucho del interés sobre este enfoque metodológico.

La metodología SCRUM está representada esquemáticamente por la figura 1. (Santimacnet, 2010)

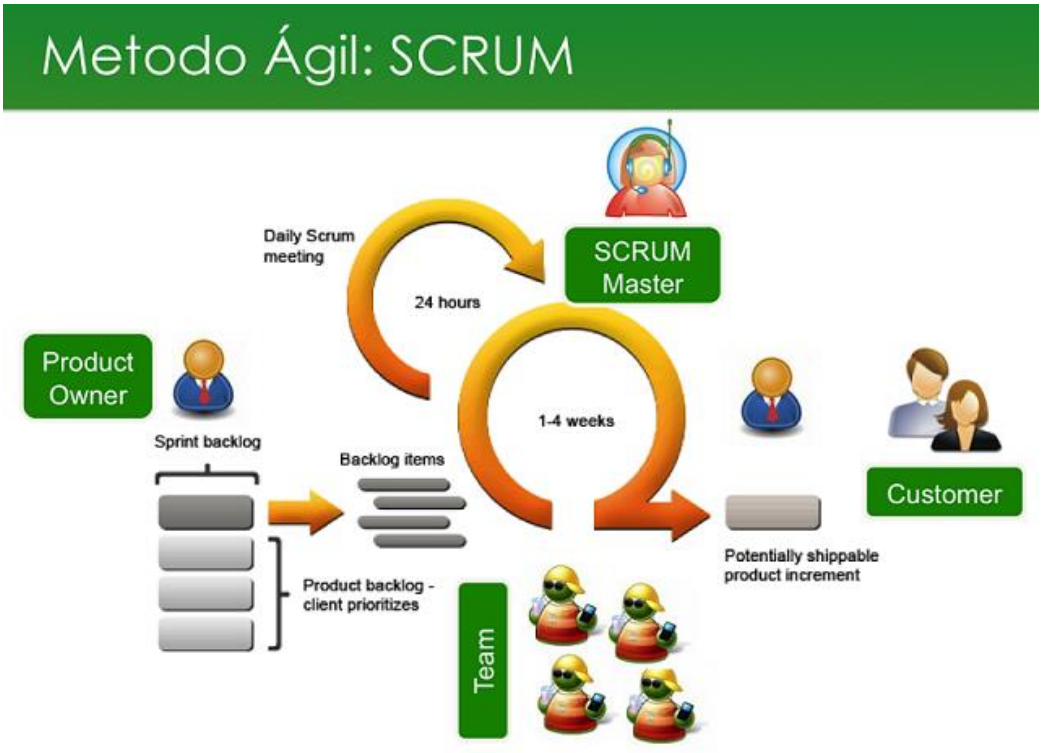


FIGURA 1 **- MÉTODO ÁGIL: SCRUM**

La totalidad de los requerimientos a desarrollar, denominados historias de usuario son divididos en grupos en función de su prioridad relativa para luego ser implementados en ciclos de esfuerzos relativamente cortos (del orden de un mes de duración) llamados sprints; las tareas son organizadas en el equipo de tal manera que las asignaciones y prioridades se revisan diariamente en una reunión breve llamada Scrum que le da su nombre a la metodología. En este enfoque se siguen los principales criterios del Manifiesto obteniendo liberaciones parciales incrementales del producto bajo desarrollo.

La evidencia es consistente que al abrazar la hoja de ruta y comprometer las inversiones necesarias para desplegar formalmente esta metodología también se abordan al mismo tiempo aspectos clave del despliegue de prácticas maduras de proceso.

En tal sentido SCRUM ha sido exitosamente comparada contra los requisitos a satisfacer para alcanzar una de evaluación bajo niveles 2 y 3 del modelo de referencia SEI-CMMI demostrando que la ejecución rigurosa satisface la mayoría de los objetivos necesarios para obtener estos niveles; las pocas áreas de proceso no cubiertas directamente por no ser requeridos por SCRUM son en la práctica un requisito para el correcto desempeño de una organización dedicada a la construcción de software, como por ejemplo las prácticas de gestión de la configuración, que deben ser adoptadas de todas formas.

A los efectos de este trabajo esto brinda el marco conceptual para asumir, formalidades de evaluaciones rigurosas al margen, que una organización que despliega SCRUM abraza la mayoría de las prácticas genéricas y especificas requeridas por los niveles 2 y 3 de SEI-CMMI, y por lo tanto puede aspirar a sus

beneficios. Este factor es particularmente atractivo dado el relativamente bajo esfuerzo organizacional e inversión para desplegar e institucionalizar SCRUM respecto a otras alternativas metodológicas.

Si bien la bibliografía muestra reportes consistentes de éxito en la aplicación de metodologías ágiles en general, y SCRUM en particular, a la solución de proyectos de diferentes tamaños y complejidades no hay al presente un abordaje sistemático sobre la cuestión del valor aportado por la metodología, siendo su aplicación basada primordialmente en la intuición y resultados empíricos obtenidos reportados por las mismas organizaciones. (Colla, 2012)

Otra diferencia de Scrum con las metodologías tradicionales es que no trata el proceso de desarrollo de software como un proceso lineal, en el que se sigue la secuencia de análisis, diseño, codificación y testing. En Scrum, el proyecto puede iniciarse con cualquier actividad, y cambiar de una a otra en cualquier momento [5]. (Puello, Rodríguez y Cabarcas, 2012)

2.1.3 Marco de Trabajo Scrum.

"SCRUM es una mejora del ciclo de desarrollo orientado a objetos iterativo incremental utilizado comúnmente" (Sutherland, 2014, p.58).

Otra definición que da el mismo autor sobre Scrum dice que es un proceso de desarrollo ágil de software diseñado para añadir energía, concentración, claridad y

transparencia a los equipos de proyectos de desarrollo de software. Se usa para aumentar la velocidad de desarrollo, alinear los objetivos individuales y de organización, crear una cultura impulsada por rendimiento, lograr estabilidad y constante comunicación en todos los niveles y mejorar el desarrollo individual y la calidad de vida del equipo. (Sutherland, Viktorov, Blount y Puntikov, 2007, p.1)

Otro punto de vista del autor Yu Xiaodan para complementar la definición sobre Scrum dice que es una metodología de desarrollo de software alternativo que se originó a partir de la práctica para fomentar la colaboración entre los desarrolladores y los usuarios, para aprovechar los ciclos de desarrollo rápido, y para responder a los cambios en un entorno dinámico. (Xiaodan y Stacie, 2014, p.911-921)

Adicionalmente Sutherland dice: debido a que el proceso de desarrollo de sistemas es complicado y complejo, se requiere la máxima flexibilidad y control apropiados. La evolución favorece aquellos que operan con la máxima exposición a los cambios del ambiente y han maximizado la flexibilidad. La evolución anula la selección de aquellos que se han aislado del cambio ambiental y han minimizado el caos y la complejidad de su entorno.

Es necesario un enfoque que permita a los equipos de desarrollo operar dentro de un entorno complejo utilizando procesos imprecisos. El desarrollo de un sistema complejo ocurre bajo circunstancias cambiantes rápidamente. La producción de sistemas ordenados bajo circunstancias caóticas requiere una flexibilidad máxima.

La metodología bien puede ser el factor más importante en la determinación de la probabilidad de éxito.

Las metodologías que animan y apoyan la flexibilidad tienen un alto grado de tolerancia para los cambios en otras variables. Con estas metodologías, el proceso de desarrollo es considerado como impredecible en el inicio, y se establecen mecanismos de control para gestionar la imprevisibilidad.

La metodología Cascada y Espiral establecen el contexto y definición de la entrega al inicio de un proyecto. Scrum y las metodologías iterativas planean inicialmente el contexto y la amplia definición entregable, y luego evolucionan la entrega durante el proyecto basado en el medio ambiente. Scrum reconoce que los procesos de desarrollo subyacentes no están completamente definidos y utiliza mecanismos de control para mejorar la flexibilidad.

La diferencia principal entre los métodos definidos (cascada, espiral e iterativo) y la aproximación empírica (Scrum), es que el enfoque de Scrum asume que los procesos de análisis, diseño y desarrollo en la fase de un Sprint son impredecibles. Se utiliza un mecanismo de control para administrar la imprevisibilidad y el control del riesgo. La flexibilidad, la capacidad de respuesta y la confiabilidad son los resultados de Scrum. (Sutherland y Schwaber, 2014, p.64-65)

Las metodologías de desarrollo tradicionales están diseñadas únicamente para responder a la imprevisibilidad de los entornos externos y el desarrollo en el inicio de un ciclo de mejora. Estos nuevos enfoques como la metodología de la espiral

> de Boehm y sus variantes son todavía limitados en su capacidad para responder a las cambiantes necesidades una vez que ha iniciado el proyecto.
>
> La metodología Scrum, por otro lado, está diseñada para ser muy flexible. Proporciona mecanismos de control para la planificación de un producto y luego para la gestión de las variables que el proyecto tenga cuando avance. Esto permite a las organizaciones cambiar el proyecto y los resultados en cualquier punto en el tiempo, para la entrega de la versión más apropiada.
>
> La metodología Scrum da libertad a los desarrolladores para idear soluciones ingeniosas en el proyecto, producto del aprendizaje y los cambios del entorno. (Sutherland y Schwaber, 2014, p.72-73)

Otro aporte que complementa estas definiciones lo da el autor Toapanta quien afirma que

> La metodología SCRUM aplicada al desarrollo de software, está basada en el modelo de las metodologías ágiles, incrementales, basadas en iteraciones y revisiones continuas. El objetivo principal es elevar al máximo la productividad del equipo de desarrollo. Reduce al máximo las actividades no orientadas a producir software funcional y produce resultados en periodos cortos de tiempo.
>
> Como método enfatiza valores y prácticas de gestión, sin pronunciarse sobre requerimientos, prácticas de desarrollo, implementaciones y demás cuestiones técnicas. Más bien delega completamente al equipo la responsabilidad de decidir la mejor manera de trabajar para ser lo más productivos posibles. (Toapanta y Kleber, 2014)

Sin embargo el autor Plinio tiene otra visión diferente y opina que

> La metodología Scrum asume que el proceso de desarrollo de software es impredecible, y lo trata como a una "caja negra" controlada, en vez de manejarlo como un proceso completamente definido. El resultado final en esta metodología se consigue de forma iterativa e incremental. Al comienzo de cada iteración se determina qué partes se van a construir, tomando como criterios la prioridad para el negocio, y la cantidad de trabajo que se podrá abordar durante la iteración.
>
> Otra diferencia de Scrum con las metodologías tradicionales es que no trata el proceso de desarrollo de software como un proceso lineal, en el que se sigue la secuencia de análisis, diseño, codificación y testing. En Scrum, el proyecto puede iniciarse con cualquier actividad, y cambiar de una a otra en cualquier momento.
>
> La administración de un proyecto utilizando Scrum implica una organización mediante iteraciones, también definidas como sprints, las cuales pueden llegar a tener una duración de entre una y cuatro semanas, y se van sucediendo una detrás de otra.
>
> Finalmente, Scrum se define como un modo de desarrollo de carácter adaptable y orientado a las personas antes que a los procesos, lo cual lo identifica como un método diferente de los tradicionales de desarrollo de software.(Puello, Rodriguez y Cabarcas, 2001, p.11)

Entre otras definiciones o apreciaciones que se pueden encontrar sobre Scrum tenemos el siguiente aporte del señor Gutiérrez que está de acuerdo y coincide desde su punto de vista con el autor Sutherland citado previamente.

Gutiérrez dice:

> Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen la forma de trabajo en equipos altamente productivos.
>
> En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de un mes y hasta de dos semanas, si así se necesita). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite. (Gutiérrez, 2013, p.54)

Un aporte complementario a las definiciones previas lo da el autor Rivadeneira quien dice que

> Scrum está indicado para proyectos en entornos complejos, donde se necesitan rápidos resultados y los requerimientos son altamente cambiantes o poco definidos. Es adaptativo, ágil, auto-organizado y con pocos tiempos muertos. Fue concebido para utilizar en combinación con otras metodologías. Su proceso se caracteriza por sprints o iteraciones de un mes. El resultado del sprint es un incremento ejecutable que se muestra al cliente. Otra característica son las

> reuniones diarias que no llevan más de 15 minutos y su objetivo es coordinar e integrar el producto a entregar. El proceso se compone de 4 fases (pre-juego – planeamiento, pre-juego – montaje, juego y post-juego), 6 roles (scrum master, propietario, equipo, cliente, gestor y usuario), dispone de prácticas y herramientas para la gestión de sus fases tales como: product backlog, sprint backlog, estimación de esfuerzos, gráfico burn-down, gráfico burn-up, planning poker.(Rivadeneira, Vilanova, Miranda y Cruz, 2013, p.383-387)

La definición sobre Scrum que tiene Puello Marrugo es:

> ¿Qué es Scrum? Scrum es una metodología para gestionar proyectos de software, mejor definida como metodología para la gestión del trabajo. Según Juan Palacio [3]: "Scrum es una metodología de desarrollo muy simple, que requiere trabajo duro porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto". Existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso, las metodologías tradicionales se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos (documentación, programas, etc.) que se deben producir, y las herramientas que se usarán [4].
>
> Su nombre proviene del rugby, deporte en el que un Scrum es una jugada que permite reiniciar el juego luego de una falta accidental. La elección del nombre busca rescatar el principio de trabajo en equipo que se observa en un Scrum de

rugby: varios jugadores se toman de los hombros y se esfuerzan para lograr –por sí solos y rápidamente– un objetivo común, que consiste en adueñarse de la pelota y llevarla hacia delante [5].

La metodología Scrum asume que el proceso de desarrollo de software es impredecible, y lo trata como a una "caja negra" controlada, en vez de manejarlo como un proceso completamente definido. Ésta es una de las principales diferencias entre Scrum y otras metodologías, como los modelos de espiral o de cascada, en los cuales el proceso de desarrollo se define por completo desde el inicio [5]. El resultado final en esta metodología se consigue de forma iterativa e incremental. Al comienzo de cada iteración se determina qué partes se van a construir, tomando como criterios la prioridad para el negocio, y la cantidad de trabajo que se podrá abordar durante la iteración [4].

La administración de un proyecto utilizando Scrum implica una organización mediante iteraciones, también definidas como sprints, las cuales pueden llegar a tener una duración de entre una y cuatro semanas, y se van sucediendo una detrás de otra.

Finalmente, Scrum se define como un modo de desarrollo de carácter adaptable y orientado a las personas antes que a los procesos, lo cual lo identifica como un método diferente de los tradicionales de desarrollo de software.(Marrugo, Rodriguez y Cabarcas, 2012)

2.1.3.1 CEREMONIAS EN SCRUM.

> Scrum es una metodología ágil de desarrollo de software que está centrada en entregar la funcionalidad de más valor para el cliente en el tiempo más corto posible. Al mismo tiempo ofrece la transparencia y control necesarios para el éxito del proyecto. La palabra Scrum no es un acrónimo, es el nombre por el que se conoce al esfuerzo esencial de un equipo de Rugby para lograr el objetivo de mover el balón hacia adelante en busca de la meta.
>
> Scrum es un proceso cíclico de sprints donde el resultado de cada uno debe ser una pieza de software funcionando por encima de diagramas, textos de diseño, etc. Para lograr lo anterior de forma exitosa, la planeación es clave. Sin embargo se planea solamente el esfuerzo que se puede realizar durante la duración de un sprint.(Estrella, 2007)

Bahit (2011) afirma: “Scrum propone realizar cuatro "ceremonias", en cada iteración (Sprint).

Éstas son:

- Planificación (Spring Planning)
- Reunión diaria (Daily Standup Meeting)
- Revisión (Sprint Review)
- Retrospectiva (Sprint Retrospective)

Ceremonia de Planificación en Scrum (Spring Planning**)”**

La reunión de planeación tiene dos objetivos:

- Definir el objetivo del sprint.
- Hacer una estimación del tiempo de los requerimientos de más alta prioridad y seleccionar sobre esa base cuantos se pueden lograr terminar en el sprint.

El resultado de la planeación del sprint es lo siguiente:

- La meta del sprint
- El backlog del sprint.(Estrella, 2007)

La planificación es lo primero que debe hacerse al comienzo de cada Sprint. Durante esta ceremonia, participan el Dueño de Producto, el Scrum Master y el Scrum Team.

El objetivo de esta ceremonia, es que el Dueño de Producto pueda presentar al equipo, las historias de usuario prioritarias, comprendidas en el Backlog de producto; que el equipo comprenda el alcance de las mismas mediante preguntas; y que ambos negocien cuáles pueden ser desarrolladas en el Sprint que se está planificando.

Una vez definido el alcance del sprint, el equipo dividirá cada historia de usuario, en tareas, las cuales serán necesarias para desarrollar la funcionalidad descrita en la historia.

Estas tareas, tendrán un esfuerzo de desarrollo estimado (generalmente mediante técnicas como Planning Poker), tras lo cual, serán pasadas al backlog de Sprint y de allí se visualizarán en el tablero una vez que cada miembro se haya asignado aquellas que considere puede realizar. La planificación puede demandar solo unas horas, o toda una jornada laboral completa.

Reuniones diarias en Scrum (Daily Standup Meeting**)**

Las reuniones diarias para Scrum, son "conversaciones" de no más de 5-15 minutos, que el Scrum Master tendrá al comienzo de cada día, con cada miembro del equipo.

En esta conversación, el Scrum Master deberá ponerse al día de lo que cada miembro ha desarrollado (en la jornada previa), lo que hará en la fecha actual, pero por sobre todo, conocer cuáles impedimentos estén surgiendo, a fin de resolverlos y que el Scrum Team pueda continuar sus labores, sin preocupaciones.(Bahit, 2011)

Durante estos 15 minutos cada miembro del equipo responde a tres preguntas:

- ¿Qué hiciste/lograste ayer?

- ¿Qué vas a hacer/lograr hoy?
- ¿Qué impedimentos tienes para lograrlo?

Las primeras 2 preguntas son en base a tareas del backlog del sprint. Los equipos Scrum son auto-administrados y cada miembro toma libremente tareas de la lista del backlog del sprint en lugar de ser asignadas por el Scrum master; por lo anterior, la respuesta a las primeras 2 preguntas giran alrededor de tareas que se lograron terminar en las últimas 24 horas y cuales se pueden lograr durante las siguientes 24.(Estrella, 2007)

Revisiones en Scrum (Sprint Review**)**

Durante la ceremonia de revisión en Scrum, el equipo presentará al Dueño de Producto las funcionalidades desarrolladas. Las explicará y hará una demostración de ellas, a fin de que, tanto Dueño de Producto como la audiencia, puedan experimentarlas.

En la ceremonia de revisión es donde el Dueño de Producto podrá sugerir mejoras a las funcionalidades desarrolladas, aprobarlas por completo o porque no, también rechazarlas, aunque esto último, jamás lo he visto en la práctica.

La ceremonia de revisión se lleva a cabo el último día del Sprint, y no tiene una duración fija. En la práctica, se utiliza el tiempo que sea necesario.

Retrospectiva en Scrum (Sprint Retrospective**)**: en la búsqueda de la perfección

> No es en vano la frase "en la búsqueda de la perfección". Como última ceremonia del Sprint, Scrum propone efectuar al equipo, una retrospectiva en forma conjunta con el Scrum Master y opcionalmente, el Dueño de Producto.
>
> El objetivo de esta retrospectiva, como su nombre lo indica, es "mirar hacia atrás", realizar un análisis de lo que se ha hecho y sus resultados correspondientes, y decidir qué medidas concretas emplear, a fin de mejorar esos resultados.
>
> La retrospectiva en Scrum suele ser vista como una "terapia de aprendizaje", donde la finalidad es "aprender de los aciertos, de los errores y mejorar todo aquello que sea factible". (Bahit, 2011)

Estrella (2007) afirma: “Cada uno de los miembros del equipo hace una retrospectiva del sprint contestando las siguientes preguntas:

- Que SI funciona para seguir haciéndolo...
- Que NO funciona para dejar de hacerlo y...
- Que podemos empezar a hacer para que funcione MEJOR ”

Otra ceremonia adicional que se debe tener en cuenta por su importancia para la identificación de riesgos es el Spike, que está definida por el ScrumStudy así:

> Un concepto que puede ser útil en la identificación de riesgos es el de riesgo basado en spike. Un (spike) pico es un experimento que consiste en la investigación o la creación de prototipos para entender mejor los riesgos potenciales. En un pico, se realiza un ejercicio intenso en dos o tres días (preferiblemente al comienzo de un proyecto antes de los procesos de creación

> priorizada del Product Backlog) para ayudar al equipo a determinar las incertidumbres que podrían afectar al proyecto. Los Spike son útiles cuando el Scrum Team (Equipo Scrum) se está acostumbrando a trabajar con nuevas tecnologías o herramientas, o cuando las historias de usuario son largas. También ayudan a estimar el tiempo y el esfuerzo con mayor precisión. (SCRUM Study, 2013, p.121)

Otros conceptos necesarios para aclarar el tema tratado y las variables planteadas en el problema de investigación son los siguientes.

2.1.4 Gestión de Riesgos en Scrum.

En relación con la gestión de riesgos dentro del marco de trabajo Scrum otros autores dicen lo siguiente:

Gosh (2012) afirma: “SCRUM ataca y cierra estratégicamente los riesgos en cada Sprint” (p.30).

Gosh (2012) afirma: “SCRUM gestiona proactiva e iterativamente los riesgos antes de que estos surjan” (p.33).

Vähä-Sipilä (2013) afirma: “aquellos riesgos que no son controlados, necesitan ser aceptados en un Sprint Review, como riesgos residuales. En caso contrario, el Sprint, no estaría completo” (p.17).

Vähä-Sipilä (2013) afirma: “Un riesgo residual solo puede ser controlado o aceptado. No pueden ser rechazados o ignorados. Esto equivaldría a aceptar la materialización del riesgo como un hecho.” (p.25).

Según el autor Ravi y otros autores, las prácticas específicas para gestionar riesgos en Scrum son las siguientes:

a. SG 1: Prepararse para la gestión de riesgos
   i. SP1.1 Determinar categorías y fuentes de los riesgos
   ii. SP1.2 Definir parámetros de riesgo
   iii. SP1.3 Establecer una estrategia para la gestión de riesgos
b. SG 2: Identificar y analizar los riesgos
   i. SP 2.1 Identificar riesgos
   ii. SP 2.2 Evaluar, categorizar y priorizar riesgos
c. SG 3: Mitigar riesgos
   i. SP 3.1 Desarrollar planes de mitigación de riesgos
   ii. SP 3.2 Aplicar los planes de mitigación de riesgos.(Ravi, Reddaiah , p.469, Movva, Kilaparthi, 2012, p.469)

Ravi y otros autores (2012) afirman: “El proceso de identificar riesgos ocurre en múltiples niveles.

- Visión del producto
- Hoja de ruta del producto
- Planeación de entregas y liberaciones de producto

- Planeación de los Sprint
- Reuniones diarias" (p.472).

Sin embargo el autor Freire (2012) opina que "el riesgo es un proceso implícito, ya que, los problemas y potenciales incidentes, son discutidos durante las reuniones de planeación." (p.8).

Un aporte que indica lo contrario hasta ahora dicho es del autor Preis (2012) quien dice "SCRUM, satisface todos los objetivos específicos de la gestión de proyecto, menos la gestión de datos y la gestión de riesgos" (p.17).

Otra opinión que respalda la visión anterior la da el señor Akif (2012), quien dice: "No existe plan o estrategia en SCRUM para gestionar el riesgo. Este es un factor importante en cualquier proyecto, por consiguiente, de obligatoria introducción en SCRUM" (p.3).

Concluyendo, un concepto general sobre la gestión de riesgos es dado por el autor Dalipi (2013) quien dice: "Cuatro fases para el proceso de gestión de riesgos

- Identificar riesgos potenciales
- Medir o valorar el riesgo
- Implementar planes para direccionar el riesgo
- Volver a medir o valorar el riesgo durante el proyecto." (p.24)

2.1.4.1 TÉCNICAS DE IDENTIFICACIÓN DE RIESGOS.

Dentro de la gestión de riesgos, y en particular en la primer fase de Identificación de riesgos dentro del marco de trabajo Scrum, según el SCRUMstudy

Las siguientes técnicas se utilizan comúnmente para identificar riesgo:

Técnicas de Identificación de Riesgos

1. **Repasar las lecciones aprendidas de Retrospect Sprint (Retrospectiva) o de procesos de Retrospectiva de Proyectos**.

El aprender de proyectos similares y de Sprints anteriores del mismo proyecto, al igual que la exploración de las incertidumbres que afectaron esos proyectos y Sprints puede ser una forma útil de identificar riesgos.

2. **Risk Checklists**

Risk checklists puede incluir los puntos claves a tener en cuenta cuando se identifican los riesgos, riesgos comunes encontrados en el proyecto Scrum, o incluso las categorías de riesgos que deben ser atendidas por el equipo. Las listas de verificación son herramientas valiosas para ayudar a asegurar un nivel de identificación de riesgo detallado.

3. **Risk Prompt Lists**

Risk prompt lists se utilizan para estimular pensamientos con respecto a la fuente de donde los riesgos se pueden originar. Risk prompt lists para industrias y proyectos diversos están disponibles al público.

4. **Brainstorming**

Sesiones donde los stakeholders relevantes y los miembros del Equipo Central de Scrum pertinentes comparten abiertamente ideas a través de discusiones y sesiones de intercambio de conocimientos, algo que normalmente es llevado a cabo por un facilitador.

5. **Risk Breakdown Structure (RBS)**

Una de las herramientas claves que se utilizan en la identificación de riesgos es un risk breakdown structure (Estructura de Descomposición del Riesgo). En esta estructura, los riesgos se agrupan en función de sus categorías o elementos comunes.

Por ejemplo, el conjunto de riesgos puede ser categorizado como algo financiero, técnico o relacionado con la seguridad. Esto permite que el equipo planifique mejor y se encargue de cada riesgo. (SCRUMStudy, 2013, p.120-121)

El SEI dice lo siguiente como método de identificación de riesgos:

Los riesgos en un proyecto de desarrollo de software pueden ser conocidos, desconocidos, o imposibles de conocer. Los riesgos conocidos son los que uno o más personas del proyecto son conscientes, si no explícitamente como riesgos, al menos en lo que respecta a ellos. Los riesgos desconocidos son los que salieron a la superficie así se les dé al personal del proyecto el derecho, oportunidad, señales, y la información. Los riesgos imposibles de conocer son aquellos que, incluso en principio, nadie podía prever. Por lo tanto estos riesgos, son potencialmente críticos para el éxito del proyecto, y están más allá del alcance de cualquier método de identificación de riesgos. (Carr, Marvin, Konda y Suresh, 1993)

2.1.4.2 Evaluación del Riesgo en Scrum.

Según el SCRUMstudy

> Esta técnica de evaluación del riesgo implica la clasificación de la magnitud de los riesgos lo que ayuda al Scrum Team (equipo de trabajo) a identificar los riesgos en el orden de su impacto potencial en el proyecto. Por ejemplo, en la figura 2, el Riesgo 1 tiene el mayor impacto y preferiblemente debería abordarse primero. (SCRUMStudy, 2013, p.123)

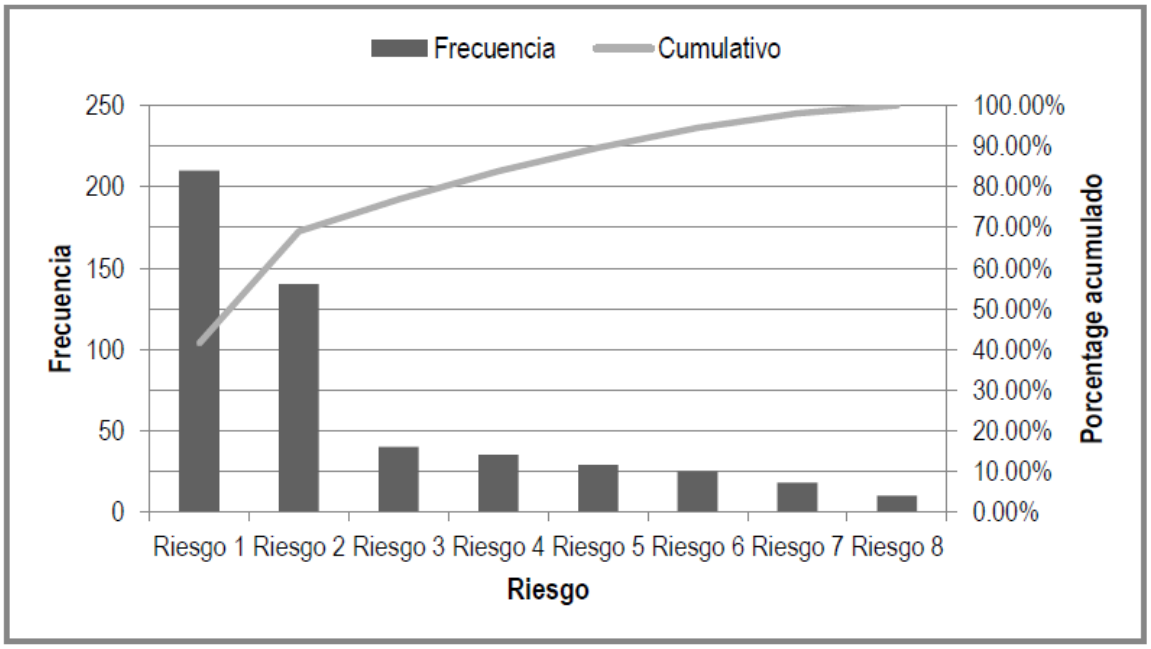


***FIGURA 2* - EJEMPLO DEL DIAGRAMA DE PARETO.**

2.1.4.3 Respuesta al Riesgo en Scrum.

Desde el punto de vista del SBOK del SCRUMstudy

La respuesta a cada riesgo dependerá de la probabilidad y el impacto del riesgo. Sin embargo, la naturaleza iterativa de Scrum, con sus ciclos de tiempo de respuesta y retroalimentación rápida permite que las fallas se detecten de forma temprana; por lo tanto, hablando en términos prácticos, tiene una función de mitigación natural construida dentro del sistema. El riesgo puede ser mitigado mediante la implementación de una serie de respuestas. En la mayoría de las situaciones, las respuestas son proactivas/ o reactivas. En el caso de un riesgo, un plan B puede ser formulado, que se puede utilizar como una alternativa en caso de que el riesgo se materialice—en este caso, plan B es una respuesta reactiva. A veces, los riesgos se aceptan y son un ejemplo de una respuesta al riesgo que no es ni preventivo ni reactivo. Los riesgos se aceptan debido a varias razones, como en una situación en la que la probabilidad o el impacto del riesgo son muy bajos para una respuesta. La aceptación también puede ser el caso en una situación en la que la aprehensión de riesgos secundarios puede disuadir al Product Owner de tomar cualquier acción. El esfuerzo realizado por el Product Owner para reducir la probabilidad o el impacto, o ambos, del riesgo es un ejemplo de una respuesta preventiva a la mitigación de riesgos.

Una vez identificados los riesgos se incluyen como parte del producto priorizado, varios riesgos se mitigan durante el proceso de crear entregables cuando las tareas

> relacionadas con las historias de usuario son definidas en el proceso de priorizar el Product Backlog.
>
> En Scrum, la responsabilidad del riesgo es claramente del Product Owner para la gestión de riesgos en relación con los aspectos del negocio, y también del Scrum Team por implementar respuestas de los riesgos durante el transcurso de un Sprint. El Scrum Master mantiene una estrecha vigilancia sobre los riesgos potenciales que podrían afectar al proyecto y mantiene al Product Owner y al Scrum Team informados. (SCRUMStudy, 2013, p.126)

2.1.4.4 SEGUIMIENTO Y CONTROL DEL RIESGO.

Un concepto adicional sobre esta fase de seguimiento y control de riesgos según Castillo Brenes es:

> Durante la ejecución del proyecto, se debe monitorear constantemente el ambiente en caso de que alguno de los detonantes definidos en los procesos anteriores se dispare. Es ahí donde los registros de riesgos ayudan a confirmar riesgos previstos y dar seguimiento a las acciones definidas en el plan de respuesta. De igual manera, debe utilizarse como una herramienta para identificar, cuantificar y responder periódicamente a las situaciones de riesgos detectadas a lo largo del proyecto. Es posible que en el desarrollo del proyecto surjan nuevos riesgos que no estaban presentes al principio del mismo, y de la misma manera la probabilidad

> de otros eventos se hace nula por lo que es necesaria una continua labor de gestión de riesgos periódicamente. Por esta razón podemos decir que la gestión de riesgos no es una serie de actividades en serie sino más bien un ciclo de actividades cuyo fin llega cuando el proyecto se cierra. (Castillo, 2009)

2.1.4.5 COMUNICACIÓN DEL RIESGO EN SCRUM.

Sobre la Comunicación del Riesgo, el SBOK dice:

> Debido a que los stakeholders tienen un interés en el proyecto, es importante comunicarse con ellos con respecto a los riesgos. La información proporcionada a los stakeholders relacionada con el riesgo debe incluir el impacto potencial y los planes para hacerle frente a cada riesgo. Esta comunicación siempre está en curso y debe ocurrir en paralelo con los pasos secuenciales discutidos hasta ahora- Identificación de riesgos, la evaluación o análisis, y la mitigación o respuesta al riesgo. El Scrum Team (equipo de trabajo) también puede discutir riesgos específicos relacionados con sus actividades con el Scrum Master durante Daily Standup Meetings. El Product Owner es responsable de la priorización de los riesgos y de comunicarle la lista de prioridades al Scrum Team. (SCRUMStudy, 2013, p.127)

2.2 MARCO CONTEXTUAL

La orientación de este trabajo es proponer una guía que sirva para la identificación, el análisis, la respuesta, el seguimiento y el control de riesgos, que le permita a las empresas dedicadas a la construcción de software, aplicar dicho modelo en cualquier proyecto SCRUM, para gestionar los riesgos de forma más objetiva.

Por lo tanto, el contexto donde se desarrollará el objeto de estudio de este trabajo, será cualquier empresa cuyo nicho sea el desarrollo de software, (a la medida o como producto), y dentro de éstas, aquellas empresas, que hagan uso de metodologías ágiles para el desarrollo de software, específicamente, el marco de trabajo SCRUM.

Respecto a las metodologías ágiles en el medio se habla sobre ellas y la calidad de los proyectos que se realizan bajo dichos marcos de trabajo como Scrum. En Barcelona se realizó un encuentro ágil en mayo del 2009 con el título "*Calidad y Agilidad*" y según el señor Xavier Albaladejo, los temas tratados en la experiencia dicen lo siguiente:

2.2.1 CALIDAD Y AGILIDAD.

Conceptos de calidad

En el ciclo de mejora continua de Deming tiene las siguientes actividades:

[Plan] Planificar cómo conseguir unos objetivos.

[Do] Ejecutar esta planificación.

[Check] Verificar los resultados conseguidos.

[Act] Definir acciones correctoras a realizar en el siguiente ciclo para mejorar los resultados.

Las dimensiones de la calidad

Simplificando el modelo de calidad ISO 9126, podríamos considerar que la calidad tiene las siguientes dimensiones:

1- Cumplimiento de expectativas del Cliente (Product Owner).

Yendo un poco más lejos y utilizando el concepto de Lean de mejorar el proceso de producción de valor (analizar desde la concepción de la idea hasta que alguien la utiliza o paga por ella, cuando se recupera la inversión realizada), más que cumplir con las expectativas del cliente, hay que cumplir con las expectativas del consumidor o usuario final de producto, que es quien va a comprarlo o verse beneficiado en su utilización. Por ello, para no crear un producto que nadie va a comprar o una aplicación que nadie va a utilizar, el cliente debería conocer quién es y qué necesita el consumidor o usuario final, hacer participar a personas representativas, investigar el modelo de negocio (cómo va a ganar dinero) y contar con las ideas y aportaciones del equipo.

2- Calidad externa, funcionamiento correcto. El producto debe comportarse de la manera esperada, tanto a nivel funcional como no funcional (rendimiento, usabilidad, etc.).

3- Calidad interna. El producto debe ser mantenible, debe poder evolucionar a un ritmo sostenido.

En el encuentro se añadió, desde una perspectiva ágil, una cuarta dimensión:

4- Satisfacción del equipo, que esté orgulloso de su trabajo, para mantener su motivación, su compromiso con el proyecto y con la empresa.

Agilidad, calidad y mejora continua

Scrum es un proceso que en bloques temporales cortos y fijos (entre 2 y 4 semanas) entrega un incremento de producto final. En cada iteración de Scrum se ejecuta el ciclo de mejora continua de Deming en los siguientes ámbitos:

Las expectativas del cliente, a las cuales se va acercando iteración a iteración mediante las siguientes actividades:

La planificación al inicio de la iteración, en que el equipo se reúne con el cliente, quien ha expresado sus expectativas mediante la lista priorizada de objetivos del proyecto (product backlog). El equipo coge tantos objetivos como se vea capaz de cumplir en la iteración, los refina con preguntas al cliente y hace una primera descomposición en las tareas necesarias para completar cada uno de ellos, creando la definición de completado y la lista de tareas de la iteración (sprint backlog). [Es importante que el cliente haya preparado las iteraciones para que tengan una meta

clara, que no sean un conjunto de objetivos poco cohesionados que produzcan pocas sinergias].

La iteración se desarrolla orientada a objetivos, minimizando el "número de objetivos en curso" (WIP, Work In Progress), para tener alguno completado cuando acabe la iteración y no muchos iniciados o a punto de finalizar.

La demostración de los resultados completados que el equipo realiza al cliente al final de la iteración. El feedback recibido del cliente permite ir acercándose a sus expectativas mediante ajustes a realizar en siguientes iteraciones, bien porque el equipo no consiguió entender correctamente sus necesidades, bien porque es el cliente quien ahora entiende mejor el producto, bien porque la velocidad de desarrollo no es la esperada o bien por cambios externos al proyecto que obligan a cambiar objetivos. Estos ajustes se incorporan al product backlog y el cliente lo re prioriza.

Notar que dentro de la propia iteración hay otro ciclo de Deming: la reunión diaria de sincronización del equipo, que le permite inspeccionar lo que está sucediendo y realizar las adaptaciones necesarias para poder conseguir los objetivos de la iteración, añadiendo o quitando tareas del sprint backlog durante la iteración.

Se puede observar que uno de los principales beneficios de Scrum es la flexibilidad frente a cambios en el contexto, el hecho de poder modificar y re priorizar los objetivos del proyecto iteración a iteración, así como las de tareas para poder conseguir un objetivo (se pueden llegar a cambiar todas las tareas si el planteamiento inicial ya no es aplicable).

Los procesos de trabajo del equipo, mediante:

La re planificación de las tareas a ejecutar.

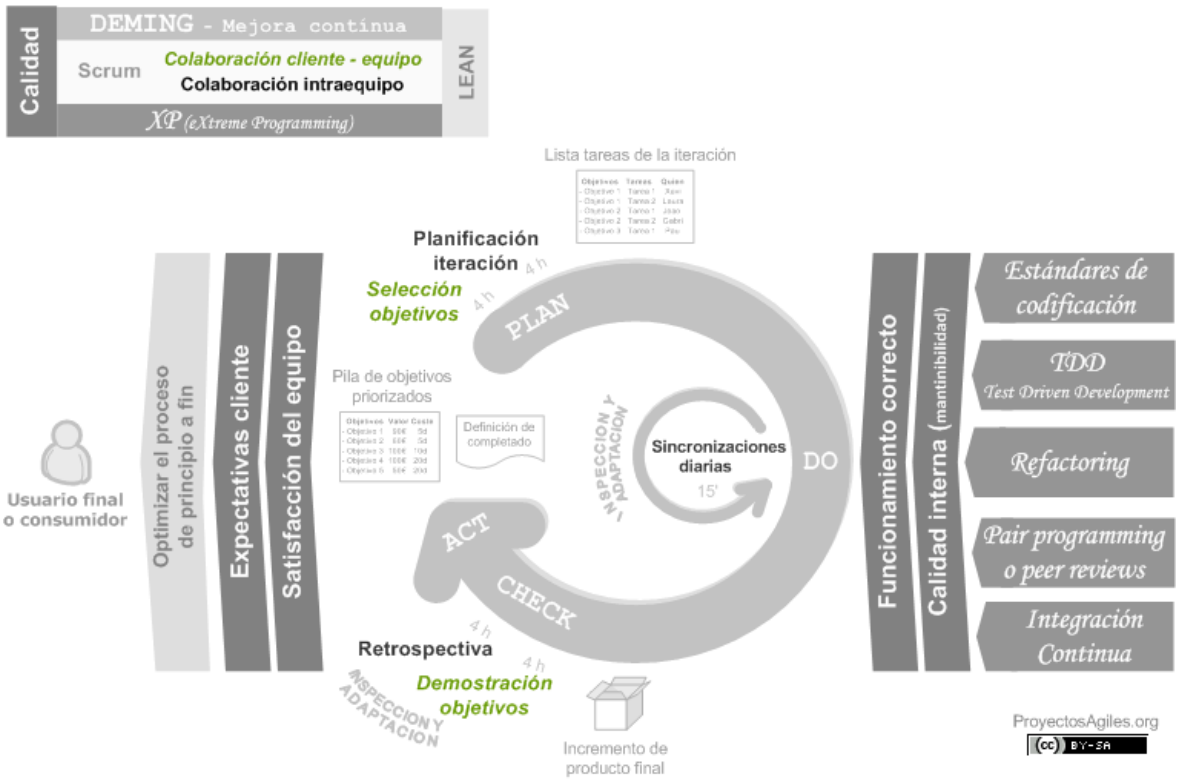
La retrospectiva de cómo fue su ejecución, qué cree que hay que mantener, mejorar o probar hacer de manera diferente, con lo que se generar objetivos y tareas de ajuste de los procesos a ejecutar en las siguientes iteraciones.

La satisfacción del equipo se consigue mediante:

El hecho de poder entregar resultados finales de forma regular, de forma que el cliente los pueda utilizar o entregar a los usuarios/consumidores finales.

El mejor entendimiento de las expectativas del cliente, que permitirá dar mejores resultados en las siguientes iteraciones.

La mejora continua del proceso de trabajo entre el equipo y con terceros, cosa que le permite ser más productivo, reducir re-trabajo y tareas de poco valor.



Agilidad y calidad interna

Es necesario construir el producto con calidad interna para poder tener una velocidad de desarrollo sostenible y crecer de manera iterativa e incremental. Ya desde la segunda iteración se debe construir sobre la parte del producto final ya existente sin poner parches que al final se transformen en “deuda técnica” (Technical Debt). Si la deuda técnica fuese creciendo, al final el coste de ir incrementando la funcionalidad se haría enorme. Para ello, eXtreme Programming utiliza las siguientes prácticas de ingeniería:

Estándares de codificación, incluyendo código bien comentado, para facilitar la propiedad colectiva.

Pair programming y/o peer review, para conseguir un código más simple, fácil de mantener y revisado por otra persona, con lo que la probabilidad de errores se reduce (arquitectónicos, de diseño o de funcionamiento).

Arquitecturas flexibles tanto en interfaz de usuario como en patrones de diseño como en información, de manera que en el futuro se puedan introducir cambios minimizando el impacto sobre el resto del sistema / los objetivos ya completados. Esto no debe implicar preparar la arquitectura o el código para requisitos futuribles (que puede que no lleguen nunca), sino simplemente cubrir cada objetivo conforme se va desarrollando intentando no hacer demasiado en contra de los futuribles.

TDD (Test Driven Development) para desde el inicio conseguir un código se puede probar así como su simplicidad. Para ello se escribe primero las pruebas (funcionales, de integración o unitarias) y después se escribe el mínimo código necesario para pasarlas.

Refactorización, para simplificar y mejorar el código una vez la prueba ha pasado, con la garantía de que sus pruebas asociadas actuarán como regresión protegiendo de la introducción de errores.

Ejecutar pruebas automatizadas en un sistema de integración continua para poder realizar cambios controlados. De esta manera, si algún componente que ya existía deja de funcionar, las pruebas asociadas inmediatamente detectan el fallo tan pronto se integra. Dado que estos errores colaterales no tardan en ser detectados,

su corrección es más rápida ya que el desarrollador todavía tiene en mente el código que los produjo.

Es importante contar con métricas para ir evaluando de manera objetiva si cómo se están produciendo las mejoras.

¿La calidad es negociable?

El cumplimiento de las expectativas del usuario es lo más importante: sin esto el proyecto no puede ser un éxito. Por ello, en metodologías tradicionales, el cliente puede llegar a negociar bajar la calidad funcional e interna con el objeto de cumplir en fechas (ver el concepto del triángulo de hierro). Al cliente no le sirve tener algo perfecto, sino tener lo que necesita a tiempo.

Para no tener que llegar a bajar la calidad, las metodologías ágiles priorizan los objetivos del proyecto en función del valor que aportan al cliente. De esta manera, para llegar en fechas se prescinde de algún objetivo poco relevante, en lugar de bajar la calidad. Esto también permite que el equipo se sienta orgulloso de la calidad de su trabajo.

Reducción de riesgos de calidad y re trabajo

Se puede utilizar una iteración (para poner un timeboxing a la no producción de resultados), llamada “iteración cero”, donde hacer la lista priorizada de objetivos del proyecto y pruebas de concepto muy ligeras de:

La arquitectura del proyecto, con una prueba extremo a extremo de la introducción/recuperación de información a través de distintas capas y tecnologías.

La usabilidad y el diseño e interacción del usuario con la aplicación, mediante wireframes o prototipos de baja fidelidad. Este modelo, un diagrama de procesos y otro de entidad/relación (todos a alto nivel) permiten complementar el modelo de historias de usuario (texto escrito e interpretable) y reducir la incertidumbre sobre las expectativas del cliente ("para el usuario, las pantallas son la aplicación").

Estas pruebas de concepto se irán refinando y adaptando iteración a iteración, según se desarrollen los objetivos por orden de prioridad, por ejemplo construyendo maquetas navegables a partir de los wireframes, como acompañamiento al backlog en la planificación de cada iteración.

CMMI vs agilidad

CMMI define unos niveles de madurez de una empresa que permiten tener un orden mejora/implantación de procesos. Por ejemplo, si no tienes nada (Nivel 1 de madurez), una de las primeras cosas que deberías hacer es un control de versiones (Nivel 2).

Algunas de las principales diferencias entre CMMI y Scrum son que CMMI se orienta a procesos y su institucionalización en la empresa, con la idea de que los procesos serán los que permitirán conseguir los objetivos del cliente y mejorar la predictibilidad de los proyectos, mientras que Scrum se orienta directamente a

conseguir los objetivos del cliente en un proyecto mediante el trabajo en equipo, el cual puede mejorar sus procesos según sea el contexto del proyecto.

Dado que la base de Scrum es la gestión de proyectos y el trabajo en equipo, CMMI y PMBOK se pueden utilizar para complementar Scrum cuando sea necesario en las áreas que Scrum no aborda por que están fuera de su alcance como, por ejemplo, la gestión de riesgos (que en Scrum son más reducidos por hacer iteraciones cortas con feedback tanto del cliente como del equipo y por limitar la complejidad que cabe en una iteración), la institucionalización en la empresa de los procesos ágiles, la gestión de proveedores, etc.

Respecto a la certificación de procesos ágiles, es más fácil certificar ISO que CMMI, ya que ISO certifica lo que dices que haces. El proceso de Scrum se puede certificar casi directamente en nivel 3 de CMMI y existen empresas certificadas en nivel 5 que al adoptar Scrum han reducido mucho esfuerzo innecesario. El problema de certificar CMMI es que los certificadores exijan que se implemente y se registre la ejecución de un proceso de una determinada manera, normalmente siguiendo metodologías tradicionales (en cascada o waterfall). Sin embargo, el propio SEI (que como el PMI está empezando a tener en cuenta la agilidad) lo desmiente: lo importante son los objetivos del proceso, no la manera cómo se implementa (que puede ser ágil).

El objetivo de Q/A en metodologías ágiles

Evitar que se produzcan errores. Utilizando el pensamiento Lean de evitar el desperdicio como el re-trabajo en búsqueda y corrección de errores, es necesario

impedir que aparezcan los errores que se deban a malas interpretaciones. Por ello, las personas que se encargan de las pruebas funcionales deben hacen TDD a nivel de requisitos:

Para ayudar a que tanto el cliente como el equipo compartan la misma visión, los testers participan en la toma de objetivos/requisitos (creación/modificación [fuera de la iteración] y detalle del backlog [en la planificación de la iteración]), asociando a cada objetivo sus condiciones de satisfacción, normalmente en forma de pruebas de aceptación (por ejemplo en el reverso de las tarjetas de historias de usuario). De esta manera el equipo y el cliente desambiguan el objetivo, le dan el detalle que consideran necesario e impiden la aparición de errores por falta de entendimiento de los objetivos.

Notar que la búsqueda de errores después de la programación sigue siendo necesaria (un programador programa e introduce errores al mismo tiempo). Mediante las prácticas de eXtreme Programming enumeradas anteriormente se puede reducir la tasa de errores por línea de código.

Los testers deben estar dentro del propio equipo, para conseguir que el equipo tenga la responsabilidad de entregar objetivos completados (probados) y no sea necesario hacer un trabajo adicional sobre ellos. De esta manera cuando el equipo haga una demostración, el cliente podrá tomar decisiones contando con que puede disponer de los resultados que se le han mostrado.

Las pruebas de aceptación por parte del cliente o con usuarios beta sigue siendo necesaria, dado que las pruebas realizadas por el propio equipo (unitarias,

funcionales, etc.) no lo pueden cubrir todo. La perspectiva del producto del cliente o del usuario final o consumidor puede ser diferente de la del equipo de desarrollo y llevar a cabo acciones no previstas: pueden esperar un comportamiento del sistema que daban por sobreentendido pero no fue explicitado en su momento, o bien no cayeron en su necesidad cuando se planteó el objetivo y al probar el producto se dan cuenta de algo adicional que es necesario.

En ciertos proyectos pueden ser necesarias iteraciones específicas cada vez que se haga una entrega de producto (por ejemplo de la mitad de duración de una iteración de desarrollo) para incluir las últimas modificaciones/errores detectados por el feedback del cliente en la última demostración.

Si el proyecto es muy grande y participan varios equipos, puede ser necesario un equipo de integración que recoja el trabajo de todos los equipos y se encargue de que todo funcione correctamente. Para mejorar la comunicación, las personas de este equipo pueden llegar a trabajar la mitad de su tiempo en cada equipo de desarrollo y el resto en el propio equipo de integración, de manera que se asegure que en cada iteración todo el producto funciona.

Colaborar en la mejora de los procesos de trabajo, buscar maneras de reducir esfuerzos innecesarios. (Abadalejo, 2009)

2.2.2 METODOLOGÍA ÁGIL QUE APLICA A CUALQUIER PROYECTO.

Otra conferencia realizada en Colombia sobre metodologías ágiles y la aplicación que tienen en los proyectos, fue realizada en Bogotá en Agosto del 2014 y tiene por título "Metodología Ágil que aplica a cualquier proyecto", en ella se hace la siguiente introducción que muestra los beneficios de Scrum:

> El ciclo de la gestión de proyectos clásica, desde el "cierre de oferta – toma detallada de requisitos – diseño – desarrollo – entrega final", no presenta soluciones metodológicas consistentes a los desafíos a los que se enfrentan los equipos de proyecto, ya que constantemente surgen nuevas necesidades, funcionalidades no contempladas inicialmente, cambios de requisitos, refinamientos del diseño, etc., que son ignorados, y normalmente considerados como "fuera del alcance inicial". Sin embargo, el cliente necesita una metodología que sea capaz de manejar toda estas "excepciones". Como respuesta a este problema nacen las metodologías ágiles, que como SCRUM garantizan un mayor nivel de flexibilidad y adaptabilidad frente a los cambios que se producen a lo largo del ciclo de vida del proyecto.
>
> SCRUM se focaliza en maximizar el valor entregado al cliente desde el principio. Éste tiene visibilidad inmediata sobre la evolución del producto, lo que minimiza los riesgos debidos a desviaciones entre lo esperado y lo recibido. Asimismo, los miembros del equipo SCRUM se encuentran más motivados e involucrados con el objetivo final del proyecto, lo que redunda en una mayor productividad y una mayor satisfacción. (Bühl, 2014)

2.2.3 AGILIDAD ES CALIDAD Y COMPETITIVIDAD.

Otro aporte es el artículo que se tomó como base para el mini keynote de apertura del Agile Open Spain 2009, que se realizó en octubre del 2009 con el título: "*Agilidad es calidad y competitividad*", en el cual dicen lo siguiente:

> **Agilidad es calidad**
>
> La agilidad es mayor satisfacción para TODOS los que participan en un proyecto:
>
> Nuestros clientes, que reciben un mejor servicio.
>
> Los trabajadores (nosotros), que podemos realizarnos profesionalmente en nuestro trabajo y que disfrutamos más.
>
> Esto es lo que vende muchas las metodologías, pero con diferencias fundamentales en cómo entendemos y conseguimos esa calidad:
>
> Calidad es satisfacer las expectativas de nuestros clientes, y para ello no hay nada mejor que priorizar los objetivos del proyecto en función de los que aportan más valor al cliente, enseñarle de manera regular el producto final (una aplicación funcionando) y siendo flexibles a cambios (control empí-rico en contraposición al control predictivo de metodologías más tradicionales, que suponen que es posible conocer desde el principio todos los detalles del sistema a construir y que, por tanto, estos no van a cambiar durante el desarrollo).

Calidad es que lo que nuestros productos puedan crecer de manera sostenible, que estos cambios tengan un impacto controlado, de manera que estos cambios sean baratos. Para ello:

Utilizamos prácticas de ingeniería (patrones de diseño, peer reviews, pair programming, coding standards) y cuidamos la calidad interna de nuestros productos empezando primero por las pruebas (TDD, refactoring).

Mientras construimos nuestros productos, comprobamos de manera continua que se comportan de manera adecuada. Para que esas comprobaciones sean baratas (dado que hay que repetirla infinidad de veces) automatizamos integraciones y pruebas.

Las metodologías ágiles no se olvidan del capital humano que lo hace posible. Nosotros creemos que la calidad de un proyecto depende de la calidad de TODAS las personas que trabajan en él y de cómo colaboran, mucho más que de tener procesos bien definidos y documentados.

Agilidad es competitividad

Por otro lado, nunca podremos competir con India o China en precio. La agilidad nos puede ayudar a tener un país más competitivo, es decir, a ser más innovadores, a proporcionar mayor calidad y a ser más productivos. Necesitamos adoptar modelos de gestión que nos permitan adaptarnos de manera continua a un entorno

de incertidumbre, en continua evolución, y ser muy ágiles en el desarrollo de iniciativas. La clave vuelve a ser:

Modelos dirigidos por valor.

Flexibilidad a cambios.

Potenciación del equipo, multidisciplinar y auto gestionado, para que aporte creatividad en el producto y que sus sinergias y la mejora continua le hagan más eficiente a través de la colaboración de sus miembros.

¿Qué se necesita para que una empresa sea ágil?

Para que una empresa empiece a ser ágil es conveniente realizar los siguientes requisitos:

Un cambio de cultura profundo en individuos y organizaciones, y especialmente en las direcciones y los gestores de las empresas. En lugar del ordeno y mando, para poder crear este tipo de equipos súper productivos y motivados se necesita cambiar a un modelo de gestión basado en la facilitación de la colaboración entre las personas que participan en los proyectos, así como el coaching del equipo, para que adquiera conocimiento de forma colaborativa, que desaprenda, se equivoque y que vuelva a aprender.

En esta misma lí-nea, hay que transformar las relaciones de cliente/proveedor en relaciones de socios de proyecto en que todos tenemos que colaborar y todos tenemos que ganar.

A nivel de ingeniería, facilidad para realizar cambios.

Enfocarnos en proporcionar valor en todas las tareas que hacemos, desde que aparece la idea del producto hasta que el usuario final o consumidor lo reciben.

Todo esto acabamos de ver es, esencialmente, el Manifiesto Ágil, Scrum, eXtreme Programming y Lean Software Development.

¿Quién hace Agile?

Hoy dí-a, las metodologí-as ágiles se están extendiendo en paí-ses punteros en tecnología (EEUU y su área de influencia, países del norte de Europa y potencias emergentes como India o China).

Entre las empresas que han realizado este cambio de paradigma para ser más competitivas y atraer talento podemos encontrar a Google, Amazon, Nokia, British Telecom, Microsoft, SAP, IBM, Bank of America, General Dinamics, Blizzard, Ubisoft, etc. y españolas como Telefónica I+D, Double You, Plain Concepts, Proyectalis, Biko2, Gailén, Autentia, etc.

De cualquier manera, en España las metodologías ágiles todavía son poco conocidas. En poco tiempo, nuestros profesionales y empresas pueden no ser competitivos y ni siquiera colaborar con empresas extranjeras porque no nos habremos adaptado a estas nuevas formas de trabajar.

En resumen

> La agilidad es una gran oportunidad para tener empresas más competitivas y a la vez disfrutar más de nuestros trabajos, oportunidad que no deberí-amos perder para no quedarnos atrás como país. (Abadalejo, 2009)

2.2.4 QA EN SCRUM.

El aseguramiento de la calidad en los proyectos ejecutados bajo metodologías ágiles como Scrum, es un tema que se debe también validar, y un aporte dado por Priyanka Hasija quien participó en la conferencia Agile Tour-Hyderabad en 2011, dice:

> **Mi experiencia como un QA en Scrum**
>
> Scrum es un enfoque ágil para el desarrollo de software, que se centra en la entrega de funciones de negocio valiosos en iteraciones de desarrollo cortos de 2 a 4 semanas. Equipos de Scrum tienen dos características que definen - son multi-funcional (es decir, que incluyen todos los conocimientos necesarios para realizar el trabajo) y que son auto-organización (es decir, se espera que el equipo de averiguar la mejor manera de hacer el trabajo). Después de trabajar durante casi dos años como una garantía de calidad (QA) analista en un equipo Scrum, he aprendido que el papel de QA en Scrum es mucho más que sólo escribir los casos de prueba y la presentación de informes a los errores del equipo.

Contrariamente a las actividades síncronas de un proyecto cascada tradicional, Scrum espera las actividades de desarrollo que se deben realizar en el orden en que se necesitan - es decir, de forma asíncrona. Esta ruptura de la tradición plantea una pregunta común por muchos de los clientes, desarrolladores y grupos de interés de negocios nuevos a Agile - "¿Cómo pueden los profesionales de pruebas se involucran de manera efectiva durante un sprint antes de que algo se ha construido?" Este artículo se centra en explicar cómo realiza la función QA pruebas de agilidad y el lugar de importancia que poseen en un equipo Scrum.

Más que sólo la construcción de casos de prueba

En proyectos Cascada tradicionales el papel QA sólo está involucrado en el final del proyecto - una vez que toda la codificación es completa. En estos proyectos, el papel QA por lo general se le daría un documento de requisitos y el código completo y se espera que para escribir y ejecutar los casos de prueba que verifican que la aplicación hace exactamente lo que dice el documento de requisitos. Sin embargo, el papel QA en Scrum no es sólo acerca de la ejecución de casos de prueba y reportar errores.

En un equipo Scrum los analistas QA participan y cumplen una variedad de responsabilidades conjuntamente con otros miembros del equipo. Ellos están involucrados desde el comienzo de un proyecto en el que trabajan en estrecha colaboración con los analistas de negocio y desarrolladores. En Scrum, el papel QA no es un equipo independiente que pone a prueba la aplicación que se está construyendo. En cambio, el equipo de Scrum es un equipo multifuncional donde

desarrolladores, analistas de negocios y QA todos trabajan juntos. Aparte de la construcción de casos de prueba, evaluaciones de calidad también puede ayudar al propietario del producto (PO) escribir los casos de prueba de aceptación por interpretar el papel de dueño del producto. También pueden interactuar con el propietario del producto, haciendo preguntas y suposiciones difíciles para ayudar a clarificar los requisitos del negocio.

Participar en la estimación de Historias

Analistas de garantía de calidad suelen ser muy buenos en la creación de escenarios de casos de prueba basados en las necesidades del usuario. También sobresalen en la identificación y captura de escenarios de casos de prueba complejos y negativos. De hecho, por lo general son mucho mejores que los desarrolladores, que tienden a centrarse sobre todo en el "camino feliz" de la historia de usuario. Incluyendo probadores durante la liberación y la iteración de planificación, cuando se estiman las historias de usuario, puede ayudar al equipo a pensar más allá de la trayectoria feliz. Esto ayuda al equipo a producir una estimación más realista, ya que ambos "feliz" y caminos "infelices" han sido considerados. La estimación es una tarea difícil y es una buena práctica tener todo el equipo participa en subir con la estimación.

Colaborar con los clientes y desarrolladores

Una de las principales responsabilidades de la función QA es proporcionar retroalimentación de su experiencia de la prueba para el propietario del producto, así como la recogida de información de ellos. QA trabajar en estrecha

colaboración con el propietario del producto para ayudarles a desarrollar criterios de aceptación detallados para sus historias de usuario. Sobre la base de lo que el equipo aprende durante cada sprint, QA también puede ayudar al propietario del producto modificar o mejorar las historias de usuario existentes para reflejar mejor las necesidades verdaderas.

Conclusión

Mientras QA todavía escribir pruebas e informar errores, también apoyan muchas otras funciones y responsabilidades en el equipo. Ellos son una parte importante del equipo y están involucrados en el derecho proyecto desde el principio.

Trabajando como analista QA en un equipo Scrum durante los últimos dos años ha sido una gran experiencia y ha proporcionado muchas oportunidades de aprendizaje. He llenado muchos papeles y responsabilidades diferentes, incluyendo el analista QA, propietario de producto, ayudando a los desarrolladores escribir casos de prueba de unidad, en calidad de conciencia de la calidad del equipo, y hacer el seguimiento de los problemas y errores de software. En resumen, esta experiencia ha añadido muchas habilidades maravillosas a mi caja de herramientas y me ha ayudado a aprender a jugar muchos roles diferentes - todo al mismo tiempo. Lo más importante que me ha enseñado a hacer preguntas en lugar de sólo tienes que seguir la documentación y hacer todo lo posible para ayudar al equipo a tener éxito.

Mientras QA todavía escribir pruebas e informar errores, también apoyan muchas otras funciones y responsabilidades en el equipo. Ellos son una parte importante

del equipo y están involucrados en el derecho proyecto desde el principio. (Priyanka, 2012)

“Gestión de la Calidad en el Mundo de Scrum y Desarrollo de Sistemas de TI ágil

Se pidió a Russell Pannone, "¿Tiene la garantía de calidad tiene un lugar en el desarrollo ágil de software?" Su respuesta instintiva fue que sí, pero lo que la forma y la función de gestión de la calidad se llevan depende de muchos factores.

Recientemente me preguntó: "¿Tiene la garantía de calidad tiene un lugar en el desarrollo ágil de software?"

Mi respuesta instintiva fue que sí, pero lo que la forma y la función de gestión de la calidad se lleva depende de muchos factores. Algunos de estos factores son:

¿Está usted en el negocio de fabricación y venta de instrumentos de monitoreo cardíaco o pelotas de baloncesto?

¿Su negocio altamente regulado por Estado, local y el gobierno federal?

¿Emplea su negocio 6 o 60.000 personas?

¿Están sus empleados colocarse ni se propagan por todo el mundo?

¿Hay lugar en las políticas corporativas y procedimientos estrictos para garantizar el cumplimiento con la Ley Sarbanes-Oxley?

Vamos a empezar por acordar la calidad es de todos la responsabilidad (del equipo) y debería ser una segunda naturaleza para los analistas de negocios,

arquitectos, desarrolladores, etc., y no sólo de la exclusiva responsabilidad de un analista de la calidad. Gestión de la Calidad (QM) debe considerarse un software de gestión y filosofía de desarrollo totalmente integrado y enfoque que se practica en toda la organización, de arriba a abajo y de forma consistente y aplicada "kaizened" [1] día tras día.

Dr. W. Edwards Deming, quien es considerado por muchos como el padre de la gestión simplificada de gestión de calidad de calidad moderna en un proceso de resolución de problemas en cuatro pasos iterativos; Planificar, Hacer, Verificar, Actuar (PDCA) como se muestra en la Figura 3.0.

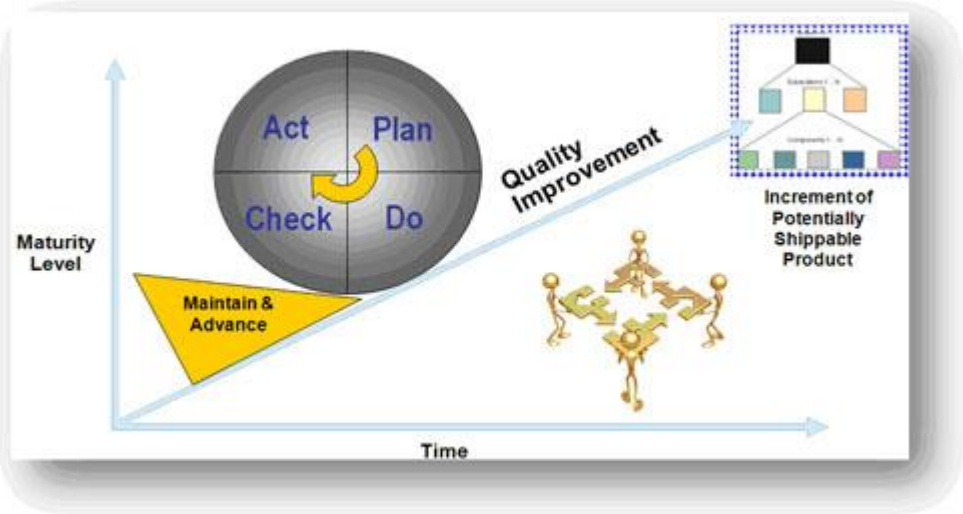


FIGURA 3 - THE DEMING QUALITY CYCLE

Mirando primero y luego en la gestión de calidad en el mundo de Scrum, un popular ciclo de vida de desarrollo de producto ágil, vemos que se alinea muy bien con el ciclo PDCA:

Plan - Emisiones y planificación de Sprint.

Do - Equipo y colaboración del cliente mediante la elaboración de los requerimientos, haciendo un poco de diseño, haciendo un poco de codificación, crear composiciones y haciendo un poco de integración, y haciendo algunas pruebas, en una sola pasada de tiempo en caja a través de un ciclo de vida de desarrollo de productos.

Check - pruebas concurrentes, integración continua, diaria stand-ups, exhibición y al final de una retrospectiva del sprint.

Act - Adaptar la forma en que los equipos de trabajo sobre la base de lo que se aprendió de la retrospectiva.

Ahora bien, en el sentido más tradicional de gestión de calidad es un conjunto de tres piezas esenciales e interrelacionados como se muestra en la Figura 4.0.

Planeación

Aseguramiento de la Calidad

Control de Calidad

FIGURA 4 - QUALITY MANAGEMENT

En conclusión, los estudios han demostrado que alrededor del 50 por ciento del tiempo se dedica a reproceso evitable y no en el trabajo con valor agregado, que es, básicamente, trabajo que ha hecho bien la primera vez. Una vez que una pieza de software esta lista y funcionando, el costo de arreglar un error puede ser 100 veces más alto, que el que habría sido durante la etapa de desarrollo. (Russell, 2009)

Ahora se exponen conceptos e ideas sobre la gestión de riesgos.

2.2.5 GESTIÓN DE RIESGOS.

La mayoría de las etapas en un proyecto, involucran riesgo. Cuando se incrementa la complejidad tecnológica en los proyectos, aumenta el nivel de riesgo de los mismos, y por lo tanto, es indispensable contar con un procedimiento formal para evaluar dichos riesgos y los efectos de éstos en los proyectos.

Etapas en la gestión de riesgos:

- Identificación de Riesgos: En esta etapa, se determinan los riesgos que pueden afectar el proyecto, se documentan sus características y se descubren los riesgos antes de que se conviertan en problemas.
- Análisis del Riesgo: Se divide en dos subetapas, Análisis Cualitativo del Riesgo y Análisis Cuantitativo del Riesgo. El análisis cualitativo, permite priorizar los riesgos para realizar otros análisis o acciones posteriores evaluando y combinando su probabilidad de ocurrencia y su impacto. El análisis cuantitativo permite analizar numéricamente el efecto de los riesgos identificados.
- Respuesta al Riesgo: En esta etapa, se planifican y desarrollan opciones y acciones para mejorar las oportunidades y reducir las amenazas a los objetivos del proyecto.
- Seguimiento y Control: En esta etapa se hace seguimiento a los riesgos identificados, se supervisan, los riesgos residuales, posiblemente se identifican nuevos riesgos, se ejecutan los planes de respuesta a los riesgos y se evalúa su efectividad a lo largo del ciclo de vida del proyecto. Bajo el marco de trabajo Scrum en esta fase, adicionalmente, se realiza la comunicación del riesgo a los stakeholders relevantes del proyecto.

Según el SBOK de SCRUMstudy:

> Los miembros del Scrum Team deberían tratar de identificar todos los riesgos que podrían afectar el proyecto.
>
> Sólo mirando el proyecto desde diferentes perspectivas y utilizando una variedad de técnicas, se puede hacer este trabajo a fondo. La identificación de riesgos se realiza a lo largo del proyecto y los riesgos se convierten en entradas para varios procesos de Scrum incluyendo la creación priorizada del Product Backlog (pila del producto), y la demostración y validación del Sprint. (SCRUMStudy, 2013, 120-121)

2.2.5.1 TAXONOMÍA DEL SEI.

La taxonomía del SEI proporciona un marco para identificar riesgos técnicos y del proceso utilizado para el desarrollo de software, el cual consiste en 194 preguntas clasificadas en tres grandes categorías: "Ingeniería del Producto", "Entorno del Desarrollo" y "Restricciones del Producto".

El objetivo de dicho cuestionario, es colaborar en la identificación de riesgos según categorías típicas, facilitando a los equipos de proyectos, explorar riesgos potenciales que podrían no haber sido considerados de otro modo.

La siguiente imagen resume la taxonomía propuesta por los integrantes del SEI: Carr, Konda y Suresh (1993), para la realización de un cuestionario particular.

A. Product Engineering
1. Requirements
   a. Stability
   b. Completeness
   c. Clarity
   d. Validity
   e. Feasibility
   f. Precedent
   g. Scale
2. Design
   a. Functionality
   b. Difficulty
   c. Interfaces
   d. Performance
   e. Testability
   f. Hardware Constraints
   g. Non-Developmental Software
3. Code and Unit Test
   a. Feasibility
   b. Testing
   c. Coding/Implement-ation
4. Integration and Test
   a. Environment
   b. Product
   c. System
5. Engineering Specialties
   a. Maintainability
   b. Reliability
   c. Safety
   d. Security
   e. Human Factors
   f. Specifications

B. Development Environment
1. Development Process
   a. Formality
   b. Suitability
   c. Process Control
   d. Familiarity
   e. Product Control
2. Development System
   a. Capacity
   b. Suitability
   c. Usability
   d. Familiarity
   e. Reliability
   f. System Support
   g. Deliverability
3. Management Process
   a. Planning
   b. Project Organization
   c. Management Experience
   d. Program Interfaces
4. Management Methods
   a. Monitoring
   b. Personnel Management
   c. Quality Assurance
   d. Configuration Management
5. Work Environment
   a. Quality Attitude
   b. Cooperation
   c. Communication
   d. Morale

C. Program Constraints
1. Resources
   a. Schedule
   b. Staff
   c. Budget
   d. Facilities
2. Contract
   a. Type of Contract
   b. Restrictions
   c. Dependencies
3. Program Interfaces
   a. Customer
   b. Associate Contractors
   c. Subcontractors
   d. Prime Contractor
   e. Corporate Management
   f. Vendors
   g. Politics

FIGURA 5 **- TAXONOMÍA DE RIESGOS EN DESARROLLO DE SOFTWARE**

2.2.5.2 ANÁLISIS DE LA GESTIÓN DEL RIESGO.

Tanto el SEI, como el PMI, como el SCRUM ALLIANCE, coinciden en que el proceso para la evaluación de un riesgo (es decir, el método o la forma como se mide cuantitativa y cualitativa en su conjunto el riesgo) se debe realizar teniendo en cuenta:

- Un valor de probabilidad
- Un valor de proximidad
- Un valor de impacto

Probabilidad

Es un valor numérico, obtenido a partir de una técnica probabilística, que indica que tan probable es que un riesgo se materialice. Entre más alto es el valor de dicha probabilidad, más posibilidades hay, de que el riesgo se haga realidad.

Proximidad

Es un valor numérico, que indica con qué rapidez se presentaría un riesgo, si no se emprende ninguna acción, es decir, si no se aplica el plan de respuesta al riesgo.

Impacto

Es un valor numérico, que permite medir el efecto que tendría el riesgo, si se llega a materializar. A menudo, se define este valor como una relación indirecta a:

- Costos (aumento)
- Cronograma (atraso)

- Alcance (incumplimiento)
- Calidad (disminución)

Existen varias técnicas para encontrar la probabilidad de un riesgo, entre ellas las siguientes:

Reunión de riesgos

Es una reunión donde de forma subjetiva y en base a la experiencia de los integrantes del Scrum Team, se identifican los riesgos que a su consideración podrían afectar el proyecto. Bajo esta técnica, no se tienen en cuenta formas estándares de medición, por tanto la evaluación hecha para los impactos y la proximidad de ocurrencia (en aquellos casos donde se llegue a tal detalle, la mayoría de estas reuniones, se conforman con obtener una medida de prioridad para clasificar los riesgos), tiene un alto grado de incertidumbre, por lo que esto, en sí mismo, se convierte en un riesgo para el proyecto.
Sin embargo, es usada a menudo como fuente para visualizar a un alto nivel, el grado de claridad y validez que tiene el alcance del proyecto. En algunos casos, una alto número de riesgos identificados bajo esta técnica, da una indicación de que el proyecto y su alcance no están lo suficientemente claros o bien definidos.

Árbol de probabilidad

Es una forma gráfica de representar los posibles caminos (eventos) que podría tomar un riesgo identificado, bajo determinadas circunstancias. La forma que adopta es de una especie de árbol. El riesgo identificado es el tronco, y los caminos o resultados que pueda tomar dicho riesgo, son sus ramas. Cada rama representa una probabilidad de ocurrencia para ese camino,

y sus hojas, representan el impacto esperado en caso de hacerse efectiva dicha probabilidad. La evaluación del riesgo, entonces se basa en el cálculo de cada probabilidad y su correspondiente impacto.

Análisis Pareto

Este método permite mediante una gráfica, representar los riesgos que son más representativos y que por ende, los que deben ser atacados más rápidamente.

Es útil, cuando se quiere predecir la posibilidad de ocurrencia, con base en información previa de riesgos, sobre otros proyectos o, sprints del proyecto actual.

Debido a que la gráfica de Pareto requiere calcular los porcentajes individuales y acumulados de cada incidencia, es necesario que éste se alimente de los datos previamente recolectados de pasadas incidencias (ya sea en otros proyectos o sprint anteriores al actual), para ver la tendencia que pueda mostrar un riesgo en especial.

De esta forma, es posible concentrarse solo en los pocos problemas representativos, y dejar de lado los menos importantes, invocando así el principio de Pareto.

Matriz de probabilidad e impacto

Es una representación en forma de tabla donde es posible establecer el grado de calificación para un riesgo (bajo, medio alto), en base a la probabilidad de ocurrencia y el impacto que tendría en la ejecución del proyecto.

Los valores dentro de la tabla son la combinación de una probabilidad contra un impacto, y se determinan multiplicando ambos valores.

Estos valores, se usan entonces, para evaluar la severidad de un riesgo, a la vez que permite

tener una vista clara de las oportunidades y las amenazas percibidas en el proyecto. La idea principal consiste en organizar los riesgos, para priorizar el tiempo y el esfuerzo requeridos para su gestión.

Para el desarrollo de este procedimiento, se utilizó la técnica de Matriz de probabilidad e impacto, ya que comparativamente hablando, es la que ofrece mejores mecanismos de medición y de ordenamiento de los riesgos, al permitir clasificarlas en categorías simples.

Para la construcción de la matriz, se usa una escala lineal y una escala no lineal.
La escala lineal E.L se usa para representar los niveles de probabilidad y una escala no lineal para describir los valores de impacto.

A continuación se muestran las escalas usadas para la conformación de la matriz.
Se puede usar una escala relativa que represente los niveles de probabilidad desde “muy improbable” hasta “casi certeza”. Como alternativa, se pueden usar probabilidades numéricas en base a una escala general. Dichas escalas pueden ser de tipo lineal o de tipo no lineal.

TABLA 2 **- ESCALA RELATIVA DE PROBABILIDAD**

Muy improbable	**Poco probable**	**Probable**	**Altamente probable**	**Casi cierto**
0,1	0,3	0,5	0,7	0,9

TABLA 3 **- ESCALA RELATIVA DE IMPACTO**

“muy bajo”;	**“bajo”;**	**“moderado”;**	**“alto”;**	**“muy alto”**
0,05	0,1	0,2	0,4	0,8

La siguiente tabla permite relacionar los objetivos con las probabilidades de impacto descritas antes.

TABLA 4 **- ESCALA DE PROBABILIDADES PARA OBJETIVOS**

Categorías/ Objetivos	Escala				
	muy bajo	bajo	moderado	alto	muy alto
Costo	0,05	0,10	0,20	0,40	0,80
Tiempo	0,05	0,10	0,20	0,40	0,80
Alcance	0,05	0,10	0,20	0,40	0,80
Calidad	0,05	0,10	0,20	0,40	0,80

La escala de impacto, refleja la importancia del impacto, ya sea negativo por las amenazas que implica o positivo por las oportunidades que genera, sobre cada objetivo del proyecto si se produce un riesgo. Las escalas de impacto, son específicas del objetivo que puede verse impactado, el tipo y tamaño del proyecto, las estrategias y estado financiero de la empresa así como su sensibilidad a impactos específicos.

La severidad de cualquier riesgo se define en términos del impacto en los objetivos del proyecto y la probabilidad de ocurrencia del riesgo. La exposición al riesgo se calcula multiplicando la probabilidad de riesgo por el impacto.

Se recomienda crear una matriz que tenga en cuenta las posibles combinaciones de las puntuaciones y las clasifique en categorías de riesgo bajo, medio o alto.

La siguiente tabla representa la matriz de probabilidad e impacto, para establecer la priorización de los riesgos.

TABLA 5 **- MATRIZ DE PROBABILIDAD E IMPACTO**

	0,05	0,1	0,2	0,4	0,8
0,9	0,05	0,09	0,18	0,36	0,72
0,7	0,04	0,07	0,14	0,28	0,56
0,5	0,03	0,05	0,10	0,20	0,40
0,3	0,02	0,03	0,06	0,12	0,24
0,1	0,01	0,01	0,02	0,04	0,08
	Oportunidad: De derecha a izquierda				
	Amenaza: De izquierda a derecha				

La siguiente lista describe las tres escalas de exposición al riesgo que se representan en la anterior tabla.

- Riego Bajo: Verde con los valores debajo de 0,05.
- Riesgo Medio: Amarillo con los valores entre 0,05 y 0,15.
- Riesgo Alto: Rojo con los valore encima de 0,15.

El área roja representa los riesgos más altos, el área de verde representa los riesgos menos importantes y el área amarilla representa los riesgos moderados.

Aquellos riesgos que se encuentran en el área roja, son riesgos que tienen un alto impacto en los objetivos del proyecto y requieren prioridad sobre los otros.

2.2.5.3 RESPUESTA AL RIESGO.

Las respuestas a los riesgos, deben ser congruentes, ser realistas dentro del contexto del proyecto, y estar acordadas entre todos los implicados en la gestión del proyecto.

Algunos componentes que son entradas importantes para la respuesta al riesgo, son los umbrales de riesgo, para las categorías, bajo, moderado y alto; estas ayudan a entender los riesgos para los cuales se necesita atención o respuesta, la asignación de personal, preparación de cronogramas y presupuestos.

Para cada riesgo, se debe seleccionar la estrategia o combinación de ellas, con mayor probabilidad de ser efectiva.

Existen tres estrategias para ocuparse de los riesgos:

- Evitar: Implica cambiar el plan de gestión del proyecto para eliminar la amenaza que representa el riesgo y aislar los objetivos del proyecto del impacto del riesgo. Algunas de las medidas que se pueden aplicar en esta estrategia son:
  - Ampliación del cronograma
  - Ampliación del alcance
  - Aclaración de requerimientos mediante reuniones
  - Implementar canales de comunicación efectivos
  - Recopilación de la mayor cantidad de información alrededor del negocio
  - Adquisición de experiencia
- Transferir: Implica trasladar el impacto negativo de una amenaza a un tercero. Simplemente se da la responsabilidad de su gestión a otro recurso, pero con esto no se garantiza su eliminación. Algunas de las medidas que se pueden aplicar en esta estrategia son:
  - Clausulas
  - Contratos
  - Acuerdos
- Mitigar: Implica reducir la probabilidad y/o impacto de un evento o riesgo a un umbral aceptable. Algunas de las medidas que se pueden aplicar en esta estrategia son:
  - Adopción de procesos menos complejos
  - Ampliación del marco de pruebas
  - Selección de proveedores más experimentados
- Aceptar: Se adopta cuando no es posible eliminar o mitigar el riesgo, cuando el equipo

del proyecto ha decidido no cambiar el plan de ejecución, sino, dejar que se continúe a pesar del riesgo presente. A menudo, se hace, cuando se ha detectado que el riesgo, no representa mayores complicaciones para los objetivos del proyecto, o cuando es imposible o inviable su tratamiento. Algunas de las medidas que se pueden aplicar en esta estrategia son:

- Planes de contingencia
- Reevaluación de objetivos

2.2.5.4 SEGUIMIENTO Y CONTROL DEL RIESGO.

El proceso básico en esta etapa, es el de evaluar, el resultado de aplicar las estrategias planeadas en la etapa de respuesta al riesgo. Observar los cambios resultantes de aplicar las estrategias y reevaluar la lista de riesgos para actualizarla en caso de ser necesario.

Esto implica un proceso cíclico, en el que en esta última etapa, se retroalimenta la fase de identificación de riesgos, para dar inicio nuevamente al ciclo de gestión de riesgos.

Es importante aclarar, que este ciclo solo es aplicable al proyecto, mientras éste, presente riesgos, o los riesgos sean al menos de grado importante como para ser tenidos en cuenta por el equipo evaluador.

2.2.5.4.1 COMUNICACIÓN DEL RIESGO.

Una herramienta utilizada para comunicar riesgos dentro de la fase de seguimiento y control del riesgo es un tablero Kanban.

El Kanban es un tablero que se divide en 3 columnas, la primera para colocar todas las actividades pendientes por hacer, esa columna se llama “TO DO”; en la segunda columna se ubican las actividades que se están realizando y se llama “DOING”; y la tercera columna es para ubicar las actividades que ya finalizaron y están listas, la columna se llama “DONE”. Las actividades se escriben en pequeños papeles, y la idea es que continuamente la persona o personas encargadas de las actividades estén moviéndolas por el Kanban, según el estado en que se encuentre la actividad, con la finalidad de estar mostrando a todos los interesados el estado actual de las tareas, y mostrar atrasos, pendientes y actividades ya hechas.
Esta herramienta se puede utilizar como apoyo para la comunicación de los riesgos, debido a que el Product Owner y/o el Scrum Master del proyecto pueden escribir cada riesgo identificado y ubicarlo en el Kanban, también las actividades necesarias para su análisis y evaluación, la mitigación de los riesgos, tareas para aplicar como respuesta a los riesgos, y otras actividades propias de la gestión de riegos que consideren necesarias. Cada vez que se esté gestionando o haciendo una de esas actividades ubicadas en el Kanban, se debe ubicar el papelito correspondiente en la columna de Doing, y una vez finalizada se debe mover la tarea para la columna del Done. Así los interesados del proyecto pueden estar enterados continuamente del avance, y estado de la gestión de riesgos; cumpliendo así con la fase de comunicar los riesgos, con una herramienta visual simple pero muy útil dentro de Scrum.

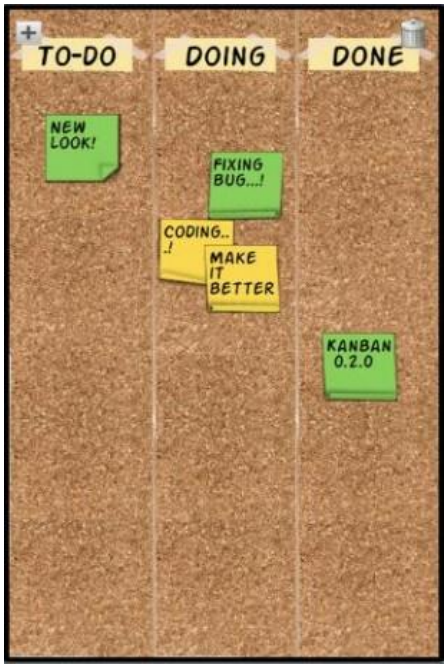


FIGURA 6 **- TABLERO KANBAN**

“Calendario de Temas” (Issues Calendars)

Consiste en un tablero dividido en columnas, las cuales corresponden a cada uno de los días dentro del sprint.

El Issues Calendars sirve para colocar en papelitos los riesgos que se identifican diariamente, los cuales se ubican en la columna correspondiente al día de la semana en que se detectaron, permitiendo así tener una visualización de la cantidad de riesgos identificados y del momento en que se dieron, comunicando a todos los interesados del proyecto los avances en la identificación de riesgos.

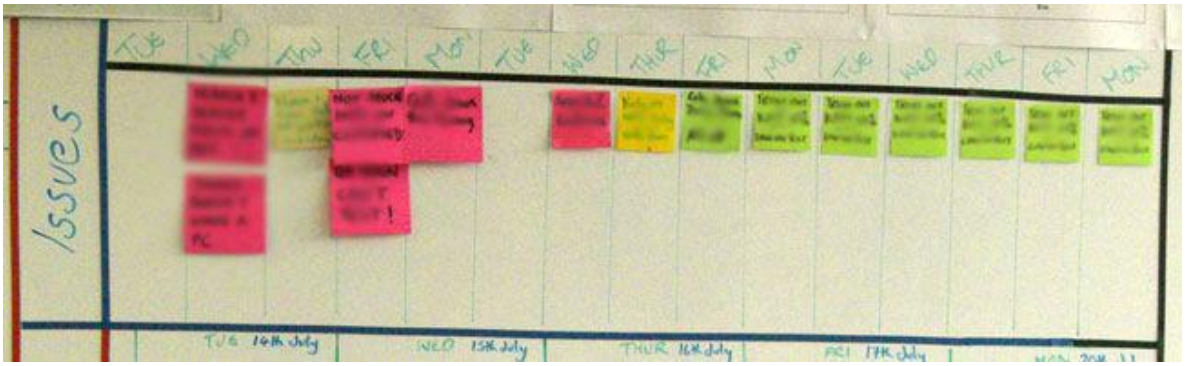


FIGURA 7 **- ISSUES CALENDARS (CALENDARIO DE TEMAS)**

3. DISEÑO METODOLÓGICO

A continuación se describen los procedimientos y herramientas que fueron utilizados para desarrollar el tema de investigación propuesto en este proyecto y obtener de esta manera los resultados que permitan resolver el problema planteado, cumpliendo los objetivos propuestos.

Para recoger el contexto, y solucionar el problema planteado se utilizó como tipo de estudio el descriptivo, como método de investigación se usó uno mixto (cualitativo y cuantitativo), utilizando fuentes secundarias como herramientas para la investigación y desarrollo del proyecto.

3.1 TIPO DE ESTUDIO

El tipo de estudio seleccionado para este proyecto fue el descriptivo. El cual según Roberto Hernández Sampieri es utilizado cuando

> El propósito del investigador es describir situaciones y eventos. Esto es, decir cómo es y se manifiesta determinado fenómeno. Los estudios descriptivos buscan especificar las propiedades importantes de personas, grupos, -comunidades o cualquier otro fenómeno que sea sometido a análisis (Dankhe, 1986). Miden y evalúan diversos aspectos, dimensiones o componentes del fenómeno o fenómenos a investigar. Desde el punto de vista científico, describir es medir. Esto es, en un estudio descriptivo se selecciona una serie de cuestiones y se mide cada

> una de ellas independientemente, para así -y valga la redundancia- describir lo que se investiga.
>
> Los estudios descriptivos miden conceptos.
>
> Es necesario hacer notar que los estudios descriptivos miden de manera más bien independiente los conceptos o variables con los que tienen que ver. Aunque, desde luego, pueden integrar las mediciones de cada una de dichas variables para decir cómo es y se manifiesta el fenómeno de interés, su objetivo no es indicar cómo se relacionan las variables medidas.
>
> Así como los estudios exploratorios se interesan fundamentalmente en descubrir, los descriptivos se centran en medir con la mayor precisión posible. Como mencionan Selitiz (1965), en esta clase de estudios el investigador debe ser capaz de definir qué se va a medir y cómo se va a lograr precisión en esa medición. Asimismo, debe ser capaz de especificar quién o quiénes tienen que incluirse en la medición.
>
> La investigación descriptiva, en comparación con la naturaleza poco estructurada de los estudios exploratorios, requiere considerable conocimiento del área que se investiga para formular las preguntas específicas que busca responder (Dankhe, 1986). La descripción puede ser más o menos profunda, pero en cualquier caso se basa en la medición de uno o más atributos del fenómeno descrito. (Hernández, 1992, p.71-72)

En esta investigación aplicando dicho tipo de estudio, se buscó información en papers, tesis, libros y documentos webs calificados sobre la gestión de los riesgos, el método del SEI (SEI-CRM) para gestionar riesgos, las propuestas y prácticas actuales dentro de Scrum para

gestionar riesgos en proyectos, así como la propuesta del SBOK de SCRUMstudy, las fases o etapas para gestionar riesgos propuestas en metodologías tradicionales, y en que procesos de Scrum pueden aplicarse, entre otros conceptos relacionados con la identificación, análisis, respuesta, seguimiento y control del riesgo. Con el propósito de validar la información existente, analizarla y proponer una guía para la identificación, el análisis, la respuesta, el seguimiento y el control de riesgos en un proyecto de software bajo el marco de trabajo Scrum.

3.2 MÉTODO DE ESTUDIO

Para el desarrollo de esta investigación el método de estudio aplicado a este proyecto es un método mixto (cualitativo y cuantitativo), los cuales consisten en:

> **La metodología cualitativa**, como indica su propia denominación, tiene como objetivo la descripción de las cualidades de un fenómeno. Busca un concepto que pueda abarcar una parte de la realidad. No se trata de probar o de medir en qué grado una cierta cualidad se encuentra en un cierto acontecimiento dado, sino de descubrir tantas cualidades como sea posible.
>
> En investigaciones cualitativas se debe hablar de entendimiento en profundidad en lugar de exactitud: se trata de obtener un entendimiento lo más profundo posible.

Dentro de las características principales de esta de metodología podemos mencionar:

- La investigación cualitativa es inductiva.
- Tiene una perspectiva holística, esto es que considera el fenómeno como un todo.
- Se trata de estudios en pequeña escala que solo se representan a sí mismos
- Hace énfasis en la validez de las investigaciones a través de la proximidad a la realidad empírica que brinda esta metodología.
- No suele probar teorías o hipótesis. Es, principalmente, un método de generar teorías e hipótesis.
- No tiene reglas de procedimiento. El método de recogida de datos no se especifica previamente. Las variables no quedan definidas operativamente, ni suelen ser susceptibles de medición.
- La base está en la intuición. La investigación es de naturaleza flexible, evolucionaría y recursiva.
- En general no permite un análisis estadístico
- Se pueden incorporar hallazgos que no se habían previsto
- Los investigadores cualitativos participan en la investigación a través de la interacción con los sujetos que estudian, es el instrumento de medida.
- Analizan y comprenden a los sujetos y fenómenos desde la perspectiva de los dos últimos; debe eliminar o apartar sus prejuicios y creencias.

La Metodología Cuantitativa es aquella que permite examinar los datos de manera numérica, especialmente en el campo de la Estadística.

Para que exista Metodología Cuantitativa se requiere que entre los elementos del problema de investigación exista una relación cuya Naturaleza sea lineal. Es decir, que haya claridad entre los elementos del problema de investigación que conforman el problema, que sea posible definirlo, limitarlos y saber exactamente donde se inicia el problema, en cual dirección va y que tipo de incidencia existe entre sus elementos.

Los elementos constituidos por un problema, de investigación Lineal, se denominan: variables, relación entre variables y unidad de observación.

Edelmira G. La Rosa (1995) Dice que para que exista Metodología Cuantitativa debe haber claridad entre los elementos de investigación desde donde se inicia hasta donde termina, el abordaje de los datos es estático, se le asigna significado numérico.

El abordaje de los datos Cuantitativos son estadísticos, hace demostraciones con los aspectos separados de su todo, a los que se asigna significado numérico y hace inferencias.

- La objetividad es la única forma de alcanzar el conocimiento, por lo que utiliza la medición exhaustiva y controlada, intentando buscar la certeza del mismo.

- El objeto de estudio es el elemento singular Empírico. Sostiene que al existir relación de independencia entre el sujeto y el objeto, ya que el investigador tiene una perspectiva desde afuera.
- La teoría es el elemento fundamental de la investigación Social, le aporta su origen, su marco y su fin.
- Comprensión explicativa y predicativa de la realidad, bajo una concepción objetiva, unitaria, estática y reduccionista.
- Concepción lineal de la investigación a través de una estrategia deductiva.
- Es de método Hipotético – Deductivo. (Mendoza, 2006)

Siguiendo la metodología cualitativa se investigó que es Scrum, como funciona el proceso, cuáles son las reuniones que se llevan a cabo según la etapa del ciclo que se está ejecutando, además de la descripción de cada reunión, con la finalidad de exponer y aclarar cómo funciona Scrum en el desarrollo de un proyecto. Para proponer luego en cada reunión la inclusión de actividades o etapas de la gestión de riesgos y los roles Scrum responsables de ellas, que harán parte de la guía propuesta.

Adicionalmente se analizó la información escrita encontrada sobre las propuestas o guías generadas para identificar riesgos en metodologías tradicionales de desarrollo de software, se analizaron las características propias del marco de trabajo Scrum y de otras metodologías ágiles, y se definió como se podía adaptar la taxonomía del SEI utilizada en metodologías tradicionales a la agilidad de Scrum, con una taxonomía propia más sencilla pero con los elementos esenciales para que la identificación y gestión de riesgos se pueda aplicar de manera

apropiada sin la riguridad de otras metodologías, para construir la guía que se propuso en este trabajo como objetivo del proyecto.

Aplicando la metodología cuantitativa se complementó la manera como se propuso la guía para lograr definir y establecer la fase de análisis o evaluación de la gestión de riesgos, relacionada con el análisis de los riesgos previamente identificados. En esta fase se hace una revisión de los datos en base a cálculos de probabilidad que permiten tomar decisiones posteriores dentro de la gestión de los riesgos de cómo tratar cada riesgo identificado.

Uniendo la información teórica investigada, realizando cálculos probabilísticos, realizando observación de un grupo de trabajo Scrum, y construyendo formatos, fue como se logró construir la guía propuesta.

3.3 POBLACIÓN Y MUESTRA

Castillo (2009) afirma: "La población se define como el conjunto de individuos de los cuales se considera importante obtener información para la investigación, y en los cuales se puede presentar una característica determinada a estudiar." (p.44)

Bernal (2010) También la define como:

> El conjunto de todos los elementos a los cuales se refiere la investigación. Se puede definir también como el conjunto de todas las unidades de muestreo.
>
> Según Jany (1994), la población es la totalidad de elementos o individuos que tienen ciertas características similares y sobre las cuales se desea hacer inferencia. (Bernal, 2010, p.160)

En este proyecto la población se compone del equipo Scrum (Scrum Team), el Scrum Master, el Product Owner (Dueño del Producto) y el Gerente del proyecto, por ser personas que de una u otra manera tiene relación directa o indirecta con la gestión de los riesgos en los proyectos de software que se ejecutan utilizando el marco de trabajo Scrum.

Sobre la muestra, Bernal afirma: "Es la parte de la población que se selecciona, de la cual realmente se obtiene la información para el desarrollo del estudio y sobre la cual se efectuarán la medición y la observación de las variables objeto de estudio." (p.161)

Para desarrollar esta investigación la muestra que se seleccionó fue un equipo de trabajo Scrum con 2 años de experiencia en desarrollar proyectos de software con esta metodología, tras haber trabajado 4 años antes con una metodología tradicional el desarrollo de los proyectos de software. El equipo de trabajo está formado por 7 personas, 1 Scrum Master, 1 Product Owner y 1 Gerente de proyectos, en total son 10 personas que han desarrollado varios proyectos.

Ellos fueron seleccionados para ser observados y complementar la información encontrada y analizada en el proyecto.

3.4 Variables o Categorías de Análisis

Las variables o categorías que se analizaron en este proyecto fueron la gestión de riesgos, que incluye la identificación de riesgos, el análisis de riesgos, la respuesta al riesgo, el seguimiento y el control de riesgos; los proyectos de software, y el marco de trabajo Scrum.

3.5 Técnicas e Instrumentos de Recolección y Análisis de la Información

Fuentes de Información

> Las fuentes de información para el presente trabajo serán primarias y secundarias. La fuente de información primaria se encuentra en los portadores de las mismas, por lo tanto se empleó la entrevista y la observación. Para la fuente de información secundaria se realizó una búsqueda bibliográfica, dado que esta información ha sido retransmitida o impresa en un documento o medio digital (Eyssautier, 2002). (Castillo, 2009, p.43)

En este proyecto la fuente principal de información utilizada fueron las fuentes secundarias, consultando bibliografía, papers, tesis, artículos y en general documentos de base de datos de universidades y documentos disponibles en sitios calificados sobre el tema del proyecto.

Instrumentos de Recolección

Toda medición o instrumento de recolección de los datos debe reunir dos requisitos esenciales: confiabilidad y validez. La confiabilidad de un instrumento de medición se refiere al grado en que su aplicación repetida al mismo sujeto u objeto, produce iguales resultados. La validez, en términos generales, se refiere al grado en que un instrumento realmente mide la variable que pretende medir.

La validez es una cuestión más compleja que debe alcanzarse en todo instrumento de medición que se aplica. Kerlinger (1979, p. 138) plantea la siguiente pregunta respecto a la validez: ¿Está usted midiendo lo que usted cree que está midiendo? Si es así, su medida es válida; si no, no lo es.

La validez es un concepto del cual pueden tenerse diferentes tipos de evidencia (Wiersma, 1986; Gronlund, 1985): 1) evidencia relacionada con el contenido, 2) evidencia relacionada con el criterio y 3) evidencia relacionada con el constructo.

La validez de contenido se refiere al grado en que un instrumento refleja un dominio específico de contenido de lo que se mide. Es el grado en que la medición representa al concepto medido (Bohrnstedt, 1976). Por ejemplo, una prueba de

operaciones aritméticas no tendrá validez de contenido si incluye sólo problemas de resta y excluye problemas de suma, multiplicación o división (Carmines y Zeller, 1979).

La validez de criterio establece la validez de un instrumento de medición comparándola con algún criterio externo. Este criterio es un estándar con el que se juzga la validez del instrumento (Wiersma, 1986). Entre los resultados del instrumento de medición se relacionen más al criterio, la validez del criterio será mayor.

La validez de constructo es probablemente la más importante sobre todo desde una perspectiva científica y se refiere al grado en que una medición se relaciona consistentemente con otras mediciones de acuerdo con hipótesis derivadas teóricamente y que conciernen a los conceptos (o constructos) que están siendo medidos.

Un constructo es una variable medida y que tiene lugar dentro de una teoría o esquema teórico.

VALIDEZ TOTAL = VALIDEZ DE CONTENIDO + VALIDEZ DE CRITERIO + VALIDEZ DE CONSTRUCTO (Hernández, 1991, p-286-288)

Para la recolección de los datos y definición de que instrumento utilizar, se siguieron los siguientes referentes:

“El documento en línea educar-argentina.com.ar, (CASSINI, 2008) define los instrumentos como: “…cualquier recurso de que pueda valerse el investigador para acercarse a los fenómenos y extraer de ellos información.”

De acuerdo con (SABINO, 2005): “un instrumento de recolección de datos es, en principio, cualquier recurso de que se vale el investigador para acercarse a los fenómenos y extraer de ellos información.” (Bastardo, 2010)

Por esta razón en este proyecto se utilizó la documentación como principal instrumento, ya que fue avalado en otros trabajos de grado de maestrías y especializaciones parecida a este proyecto, en términos de trabajar la gestión de riesgos como parte de su investigación; este instrumento ya fue aprobado en dichos trabajos.

A continuación se listan instrumentos utilizados:

Documentación

Con esta técnica se recolectó la información teórica necesaria de artículos, informes, libros, documentos web y tesis sobre los temas expuestos en el marco de referencia y que ayudan a solucionar la pregunta de investigación planteada. Este fue el principal instrumento utilizado en este trabajo.

Observación Directa

Se realizó observación sobre la muestra definida en la sección 3.3

Internet, Bibliotecas y Otras fuentes

Internet y las bibliotecas fueron utilizadas para establecer el marco de referencia y realizar las investigaciones necesarias para el trabajo. Como otras fuentes se utilizaron tesis impresas y otro material técnico sobre el tema problema tratado.

Análisis de la información

Para analizar la información recolectada durante la investigación se realizó una revisión preliminar y general del contenido pertinente que se pudiera utilizar en el trabajo, posteriormente se clasificó por temas o variables, es decir, información sobre la gestión de riesgos, información sobre metodologías agiles, sobre Scrum, sobre gestión de riesgos en Scrum, e información sobre guías y taxonomías para utilizar en la propuesta a construir. Luego se analizó la información y se utilizó en el desarrollo del trabajo.

4. RESULTADOS

En esta sección se enunciaran los resultados obtenidos con esta investigación, a partir de los conceptos teóricos encontrados y asociados en el capítulo 2 como marco de referencia, que posteriormente fueron analizados y utilizados para construir la propuesta entregada como resultado del trabajo, como un aporte propio que solucionara la pregunta de investigación formulada en la sección 1.2 y cumpliera el objetivo general de la sección 1.4.1.

Como se expone en el marco teórico de la sección 2.1, se inició con la explicación sobre metodologías ágiles para el desarrollo de software (sección 2.1.1), explicando que son, cuales son las más reconocidas, sus características y ventajas respecto a metodologías tradicionales utilizadas en la ingeniería de software.

Posteriormente en la sección 2.1.2 se explica porque Scrum es la metodología ágil seleccionada como variable dentro del problema de investigación.

En la sección 2.1.3 se detalla teóricamente con varias referencias que es el marco de trabajo Scrum, sus características, beneficios, procesos y conceptos, mostrando como funciona y llevando a detalles del proceso que permitan comprender como se desarrolla un proyecto siguiendo dicho marco, los roles que se definen y participan en el proyecto, las responsabilidades que tiene cada uno, las reuniones o ceremonias definidas que se realizan, el orden y objetivo de cada ceremonia y como se realizan las entregas a los usuarios al finalizar un sprint. En la sección 2.1.3.1 se expone con más detalle cómo funcionan las ceremonias, las cuales fueron analizadas para este proyecto, junto con los roles participantes y como se realiza o se puede proponer incluir la gestión de riesgos dentro del proceso Scrum para un proyecto de software.

Una ceremonia adicional que se incluyó como propuesta de mejora para la gestión de riesgos en Scrum, es utilizar una reunión para realizar un "Spike" del proyecto, es decir, una revisión para analizar detalles y posiblemente identificar riesgos, la cual fue expuesta en la sección 2.1.3.1 del marco teórico.

Para analizar la información dentro de la gestión del riesgo y su inclusión en el marco de trabajo Scrum, se construyó como aporte propio (ver tabla 6) una asociación entre las ceremonias, las fases de gestión de riesgo y los roles Scrum responsables, el cual se detalla a continuación.

4.1 Asociación de las ceremonias Scrum, con las fases de gestión de riesgos y los roles responsables

Una vez descritas las ceremonias o reuniones que se realizan durante un sprint en la sección 2.1.3.1 (que puede durar de 2 a 4 semanas, o según lo defina el equipo de trabajo en Scrum), a continuación se relacionan estas reuniones con las diferentes fases de la gestión de riesgos, y los roles de Scrum que serían responsables de gestionarlas, como propuesta propia de este trabajo:

TABLA 6 **- ASOCIACIÓN DE LAS CEREMONIAS SCRUM, CON LAS FASES DE GESTIÓN DE RIESGOS Y LOS ROLES RESPONSABLES**

Ceremonia Scrum	Fase de Gestión de Riesgos	Rol en Scrum Responsable
Spike	Identificación de riesgos	Dueño del Producto (*Product*

		Owner)
Spring Planning (Planificación)	Identificación de riesgos	Equipo de Trabajo (*Scrum Team*)
	Análisis de riesgos	Gerente del Proyecto Dueño del Producto (*Product Owner*)
	Respuesta al riesgo	Gerente del Proyecto Dueño del Producto (*Product Owner*)
Daily Standup Meeting (Reunión Diaria)	Seguimiento y control al riesgo	Gerente del Proyecto Dueño del Producto (*Product Owner*) *Scrum Master*
Sprint Review (Revisión)	Seguimiento y control al riesgo	Dueño del Producto (*Product Owner*) Gerente del Proyecto *Scrum Master*

Retrospective Sprint (Retrospectiva)	Seguimiento y control al riesgo	Dueño del Producto (*Product Owner*) Gerente del Proyecto *Scrum Master*

En la propuesta planteada en la tabla 6, se puede observar la distribución de las fases de gestión de riesgos en las ceremonias de Scrum y los roles asignados como responsables, pero existen algunas fases de la gestión de riesgos que hacen parte de las ceremonias y no se especificaron en la tabla debido a que son continuas durante el desarrollo del Sprint.

En las secciones siguientes se explican las fases de gestión de riesgos y la relación que tienen con la tabla 6.

Adicionalmente se realizó un análisis del proceso de gestión de riesgos, con la información expuesta en la sección 2.2.5 del marco contextual, para llegar a proponer en la guía como se debe realizar la identificación de riesgos dentro del marco de trabajo Scrum teniendo en cuenta que dicha fase debe ser más sencilla y ágil que las propuestas para otras metodologías.

4.2 IDENTIFICACIÓN DE RIESGOS

El proceso de identificación de riesgos tiene como propósito identificar los riesgos que pueden afectar el proyecto potencialmente. La identificación de riesgos es un proceso iterativo, porque se pueden descubrir nuevos riesgos a medida que el proyecto avanza a lo largo del ciclo de vida.

Como propuesta para la guía del trabajo se plantea lo siguiente:

La fase de Identificación de Riesgos se inicia en la ceremonia Spike dentro de Scrum, donde el Product Owner es el responsable, pero en la identificación de posibles riesgos que puedan generarse en el proyecto, también participan el Scrum Team, Scrum Master y cualquier otro Stakeholder del proyecto, explicados en la sección 2.1.3.1 del marco teórico.

Durante la reunión de Spring Planning también se identifican riesgos dentro de las historias de usuario planeadas, en esta ceremonia el responsable es el Scrum Team, aunque cualquier otro participante puede identificar riesgos.

Durante las Daily standup meeting de manera implícita se identificación posibles nuevos riesgos, cuando el Scrum Team comenta las dificultades presentadas el día anterior, el Scrum Master, el Product Owner o el Gerente del proyecto pueden posteriormente aplicar el procedimiento propuesto para determinar si alguna de esas dificultades se puede catalogar como riesgo para el proyecto, y se debe incluir en el proceso para su gestión.

En la reunión de retrospectiva (Sprint Retrospective) al realizar una revisión de lo realizado en el sprint que culminó, es posible que se encuentren situaciones o factores que afectaron negativamente, y que se puedan revisar por parte del Producto Owner y Gerente de proyecto para determinar si se pueden catalogar como riesgos, y así incluirlos en la gestión de riesgos del proyecto.

Es una situación implícita en el sprint que se debe tener en cuenta día a día, cualquier situación que puedan calificarse como riesgo debe ser incluido en la gestión de riesgos y notificada al rol responsable de esta fase.

Posteriormente se analizó que técnicas existen para identificar riesgos y que propuestas se han planteado dentro del marco de trabajo Scrum, expuestas en la sección 2.1.4.1 del marco teórico, para poder llegar a seleccionar una de ellas y utilizarlas dentro de la guía propuesta en este proyecto.

4.2.1 TÉCNICA DE IDENTIFICACIÓN DE RIESGOS SELECCIONADA.

En este trabajo para la propuesta que se realizó, se seleccionó como técnica para identificar riesgos dentro de la gestión de riesgos, la quinta técnica: **Risk Breakdown Structure (RBS)** expuesta en la sección 2.1.4.1 del marco teórico.

Debido a que el SCRUMstudy no define una taxonomía para la técnica RBS de identificación de riesgos, se utilizó como base para la propuesta creada la taxonomía del SEI.

Una vez seleccionada la técnica a utilizar se expone en la siguiente sección la propuesta propia que permita identificar riesgos en proyectos de software que se desarrollan utilizando el marco de trabajo Scrum como metodología ágil.

4.2.2 PROPUESTA PARA IDENTIFICAR RIESGOS EN SCRUM.

El proceso de Identificación de Riesgos se divide en dos fases: Identificación de la lista inicial de los riesgos del proyecto, y proceso de identificación y evaluación continua de los riesgos.

Para la Identificación de la lista inicial de riesgos del proyecto, se construyó un cuestionario de identificación de riesgos propio, el cual, se basó en la taxonomía de riesgos del SEI, una vez se analizó la información de la sección 2.2.5.1 del marco contextual.

En este proyecto debido a que el marco de trabajo Scrum es ágil, y no contempla la rigurosidad de las metodologías tradicionales para desarrollar proyectos de software, se seleccionaron 24 preguntas del cuestionario propuesto por el SEI, para identificar riesgos (ver anexo A), como propuesta del trabajo y aporte para la guía de gestión de riesgos en proyectos bajo Scrum como marco de trabajo.

Las preguntas fueron formuladas como afirmaciones de tal manera que al evaluarlas, éstas puedan ser clasificadas dentro de la lista de valores de certeza que se define a continuación:

Muy improbable, Poco probable, Probable, Altamente probable, Casi cierto

A modo de ejemplo se toma la siguiente pregunta:

"*Los criterios de aceptación de las historias no especifican lo que el cliente desea*"

Esta pregunta está construida de tal manera, que al momento de ser evaluada permite identificar uno de los 5 grados de certeza asociados, teniendo en cuenta para el proyecto específico, las condiciones presentes.

Para el ejemplo, si los criterios de aceptación no están bien definidos, es altamente probable que se materialice un riesgo durante la ejecución de un sprint, o en caso contrario, si los criterios de aceptación están bien definidos, es poco probable que se materialice el riesgo, ya que están dadas partes de las condiciones adecuadas para cumplir con el sprint.

El Anexo A "Plantilla Identificación de Riesgos para Scrum" fue diligenciado a modo de ejemplo para categorizar y calificar las preguntas seleccionadas para identificar riesgos en marcos de trabajo Scrum, en ningún momento estos valores son fijos en dicha plantilla, para cada proyecto debe ser diligenciada cada pregunta según aplique el contexto y situación en que se desarrolla el proyecto de software evaluado.

4.3 ANÁLISIS DE RIESGOS

El análisis de riesgos o evaluación de riesgos, permite entender cómo se vería afectado el proyecto, si se llegara a materializar de alguna forma cualquiera de los riesgos identificados. De acuerdo a lo explicado en la fase anterior, una adecuada identificación de riesgos es clave para verificar si el proyecto tiene amenazas latentes, sin embargo, ¿cuál podría ser la forma más adecuada, para saber si el riesgo identificado se volverá o no un problema?

El análisis o evaluación de riesgos propuestos se fundamenta en la referencia encontrada en la sección 2.2.5.2 del marco contextual y en la sección 2.1.4.2 del marco teórico para Scrum, la información sobre esta fase se analizó y se construyó una propuesta como parte de la guía generada en el trabajo.

El proceso de análisis o evaluación del riesgo consiste en encontrar la probabilidad de ocurrencia de cada una de las preguntas del cuestionario del Anexo A, y la categoría que resultaría afectada.

A continuación se muestra la lista maestra de riesgos final propuesta en el proyecto.
Ver Anexo A: Plantilla Identificación de Riesgos para Scrum.

El siguiente paso es, conformar la lista maestra de riesgos, con base en el cuestionario de línea base, y la tabla de probabilidad de ocurrencia y las categorías de impacto expuestas en la sección 2.2.5.2. Esta lista maestra de riesgos, será usada como plantilla para encontrar los riesgos del proyecto analizado.

Retomando la Tabla 6 (Asociación de las ceremonias Scrum, con las fases de gestión de riesgos y los roles responsables), se observa como la fase de análisis de riesgos en Scrum está asociada en la ceremonia del Sprint Planning (Planificación), donde el Product Owner (dueño del producto) y el Gerente del proyecto son los responsables de realizar el análisis de los riesgos identificados hasta el momento en el proyecto. Este análisis de riesgos puede iniciarse en la ceremonia de planificación, pero si requiere más tiempo del establecido para la reunión,

dichos roles serán responsables de terminar de realizar el análisis de riesgos, para garantizar con el cumplimiento de esta fase en la gestión de riesgos del proyecto y tener insumos para la siguiente fase. Es posible que si en otra ceremonia se identificaron nuevos riesgos dichos roles deben realizar el análisis de riesgos necesario en cualquier momento del sprint.

4.4 RESPUESTA AL RIESGO

La respuesta al riesgo, consiste en el diseño de un plan para determinar acciones para mejorar las oportunidades y reducir las amenazas a las actividades del proyecto. El principal objetivo de esta fase, es controlar los riesgos más importantes identificados durante el análisis de riesgos.

El análisis de la información referencia en la sección 2.2.5.3 sobre esta fase de gestión de riesgos se utilizó para proponer como parte de la guía construida en este trabajo la respuesta al riesgo en Scrum.

Existen tres estrategias que normalmente se ocupan de los riesgos que tienen impactos negativos sobre los objetivos del proyecto: Evitar, Transferir, Mitigar o Aceptar, las cuales son detallas en la sección 2.2.5.3 del marco contextual, y sirvieron para definir como propuesta del trabajo cómo se debe dar respuesta a los riesgos en la guía generada en el trabajo.

Según el SBOK la respuesta al riesgo se genera con estrategias preventivas, proactivas, reactivas, o aceptación del riesgo; expuestas y analizadas con las referencias de la sección 2.1.4.3 del marco teórico.

En este proyecto se tomó como estrategia la propuesta para riesgos que tienen impactos negativos de la sección 2.2.5.3, donde el riesgo se debe evitar, transferir, mitigar o aceptar.

A continuación se dará respuesta a la lista de riesgos propuesta en la fase de identificación de riesgos, como una forma de aplicar las reglas de análisis descritas anteriormente, donde a cada ítem se le indica la respuesta al riesgo y una o varias medidas para aplicar, siguiendo el ejemplo con el cual se diligenció la guía en la fase de identificación de riesgos.

Ver Anexo B: Respuesta al Riesgo, como propuesta propia dentro del proyecto.

Retomando la Tabla 6 (Asociación de las ceremonias Scrum, con las fases de gestión de riesgos y los roles responsables), se observa como la fase de respuesta al riesgo en Scrum está asociada en la ceremonia del Sprint Planning (Planificación), donde el Product Owner (dueño del producto) y el Gerente del proyecto son los responsables de realizar la respuesta de riesgos a cada uno de los riesgos identificados y analizados hasta el momento en el proyecto. Esta respuesta de riesgos puede iniciarse en la ceremonia de planificación, pero si requiere más tiempo del establecido para la reunión, dichos roles serán responsables de terminar de realizar la fase de respuesta de riesgos, para garantizar con el cumplimiento de esta fase en la gestión de riesgos del proyecto y tener insumos para la siguiente fase. Es posible que si en otra ceremonia se identificaron nuevos riesgos dichos roles deben realizar el análisis de riesgos y entrar a evaluar la respuesta al riesgo necesaria en cualquier momento del sprint.

4.5 Seguimiento y Control del Riesgo

El seguimiento de riesgos se encarga de supervisar, el estado de los riesgos y el progreso de sus planes de acción.

El seguimiento de riesgos incluye la revisión de las probabilidades, impactos y exposiciones de la lista de riesgos identificados en el proyecto.

La fase se control, tiene como objetivo corregir las desviaciones y controlar los riesgos de la lista de riesgos actuales del proyecto. Por lo tanto se debe de estar atento a nuevos riesgos que puedan aparecer en su entorno a medida que el proyecto avanza.

En esta fase se hizo revisión teórica y contextual para analizarla la información y proponer como trabajarla en la guía propuesta en el trabajo, revisar secciones 2.2.5.4 del marco contextual y la sección 2.1.4.4 del marco teórico.

Dentro del marco de trabajo Scrum, esta fase adicionalmente debe incluir la Comunicación de los Riesgos.

En esta etapa se comunica el resultado de las fases anteriores de la gestión del riesgo realizadas, como la identificación del riesgo, el análisis de los riesgos y la respuesta de los riesgos a los stakeholders apropiados del proyecto, además de indicar su percepción sobre los eventos o situaciones inciertas.

4.5.1 COMUNICACIÓN DEL RIESGO.

Se analizó la información ubicada en la sección 2.1.4.5 del marco teórico sobre esta fase para Scrum, con la finalidad de incluirla en la guía propuesta en este trabajo.

Como propuesta del trabajo para esta fase se retoma la Tabla 6 (Asociación de las ceremonias Scrum, con las fases de gestión de riesgos y los roles responsables), se observa como la fase de seguimiento y control al riesgo en Scrum está asociada a 3 ceremonias del sprint, Daily Standup Meeting (Reunión Diaria), Sprint Review (Revisión) y Retrospective Sprint (Retrospectiva), donde el Product Owner (dueño del producto), el Gerente del proyecto y el Scrum Master son los responsables de realizar el seguimiento y control de riesgos a cada uno de los riesgos identificados, analizados y mitigados hasta el momento en el proyecto. Este seguimiento y control de riesgos puede iniciarse en la reunión diaria, en la reunión de revisión o en la reunión de retrospectiva, pero si requiere más tiempo del establecido para la reunión que es sólo de 15 minutos en la Daily, dichos roles serán responsables de terminar de realizar la fase de seguimiento y control de riesgos, para garantizar con el cumplimiento de esta fase en la gestión de riesgos del proyecto. Es posible que si en otra ceremonia se identificaron nuevos riesgos dichos roles deben realizar el análisis de riesgos, entrar a evaluar la respuesta al riesgo necesaria en cualquier momento del sprint, y finalmente realizarle el seguimiento y control al riesgo, siendo una labor cíclica durante el sprint y el desarrollo del proyecto.

Como adicionalmente, se hace una Comunicación del Riesgo a todos los interesados del proyecto, se utiliza como herramienta de apoyo para la comunicación un Tablero Kanban, el

cual es descrito en la sección 2.2.5.4.1 del marco contextual en el trabajo. Esta herramienta se propone dentro del planteamiento de la guía en este trabajo.

Otra herramienta que sirve de ayuda para la Comunicación de los riesgos es el "Calendario de Temas" (Issues Calendars). El cual es descrito en la sección 2.2.5.4.1 del marco contextual del trabajo, como otra propuesta de este trabajo para complementar las herramientas utilizadas en esta fase de comunicar riesgos.

Para finalizar los resultados de este proyecto se remite al Anexo C de este trabajo (Detalle paso a paso de uso de la guía para gestionar los riesgos en los proyectos de desarrollo de software bajo marco de trabajo Scrum), en el cual se dan las indicaciones de como utilizar la guía construida.

5. CONCLUSIONES Y RECOMENDACIONES

El propósito inicial, cuando se planteó la idea de crear una guía que permitiera gestionar riesgos en proyectos que hicieran uso de metodologías SCRUM, era el de facilitar a los integrantes del equipo de desarrollo, la implementación de pautas que se ajustaran a los procesos ágiles de la metodología SCRUM, sin sacrificar en absoluto, las cualidades que ofrece SCRUM de facto. La guía fue construida de tal forma que los equipos siguieran teniendo la agilidad requerida para ejecutar las actividades del proyecto, pero que a la vez pudieran llevar un adecuado registro de los riesgos.

Con esta guía los equipos de trabajo, podrán tener herramientas sencillas y completamente adheridas a las pautas de SCRUM, con las que podrán identificar de forma temprana riesgos que de otro modo, afectarían la correcta finalización de los Sprint del proyecto.

Los resultados de la investigación, nos llevan a pensar, que si es adecuado optar por implementar una gestión de riesgos en proyectos basados en SCRUM, y que, suponer que dicha gestión está implícita, es de hecho, generar un riesgo en si mismo.

Siempre será posible aplicar algún procedimiento para apoyar la gestión de riesgos, pero se requiere de paciencia, constancia, un cambio de actitud, proactividad hacia el cambio, cualidades éstas, que estarán orientadas al logro de los objetivos y de los resultados.

Sin embargo, la efectividad de la misma, depende de cada equipo, la forma y el momento en que se use. Dependerá del adecuado entendimiento de las etapas propuestas, y sobre todo, de la correcta asimilación de cada fase del ciclo de gestión de riesgos, con las ceremonias requeridas por SCRUM. Como se mostró en el capítulo de resultados, es primordial identificar correctamente, cuales son las ceremonias en las que se deben ejecutar las actividades necesarias para la identificación, evaluación, mitigación y respuesta al riesgo. Es de esta manera, como se podrá hacer una correcta aplicación de la guía.

Otro aspecto importante y que se debe resaltar, es que, debe existir una disposición positiva, referente a lo relacionado con la gestión del riesgo. El procedimiento resultaría inadecuado, si no existe un claro convencimiento por parte del equipo, de que se obtendrán beneficios al aplicarlo.

Depende en un 100% del líder del proyecto, que el equipo entero, tenga una disposición a la gestión de riesgos, máximo, cuando SCRUM, de cierto modo, induce a que los logros estén orientados a la consecución del resultado esperado, por consiguiente, ignorando en la mayoría de los casos, recomendaciones y buenas prácticas para el manejo del riesgo. Esto implica, en muchos casos, que durante los sprints del proyecto, es decir, las fases del proyecto, surjan problemas y que estos sean manejados de forma empírica, con juicios subjetivos, y sin evaluaciones de impacto.

Una característica importante de la guía, es su esquema de modularidad y desacoplamiento entre elementos. Es decir, que existe independencia, entre el procedimiento para identificar riesgos, y el procedimiento para evaluar riesgos, y el procedimiento para mitigar los riesgos. Los pasos descritos por la guía aun que son de seguimiento obligatorio, se pueden ejecutar independiente de procedimiento usado para identificar, o evaluar. Quiere decir esto, que los equipos, o el Product Owner, o el Scrum Master, tienen la libertad de implementar el procedimiento que más les convenga, de acuerdo a las propiedades del proyecto. Es posible, cambiar la taxonomía de preguntas que permite identificar riesgos o cambiar el método de medición de probabilidades o la taxonomía para responder al riesgo.

Otro punto a tener presente, es que la guía será susceptible de ajustes y/o modificaciones, debido principalmente, a modificaciones efectuadas a la metodología SCRUM, o al Manifiesto SCRUM, por lo que ayuda mucho el esquema de modularidad explicado antes, para realizar los cambios requeridos sobre la guía, de forma práctica y adecuada a procedimientos propios de la empresa.

A largo plazo, la guía propuesta, puede ser implementada a través de herramientas de software que permitan automatizar la capturar la información relacionada con la gestión de riesgos en los proyectos, y alimentar así, históricos sobre riesgos identificados en proyectos y como se abordó la gestión de los mismos, para tener referencias a futuras gestiones.

Este trabajo puede servir como base para trabajos futuros que deseen construir modelos o metodologías más robustas dentro de la gestión de riesgos en proyectos de software, o también para construir procedimientos para gestionar riesgos en otras metodologías ágiles diferentes a Scrum.

La guía propuesta para la identificación, el análisis, la respuesta, el seguimiento y el control de riesgos en un proyecto de software bajo el marco de trabajo Scrum, como solución al problema planteado, y que cumple con el objetivo general propuesto, se expuso en las secciones del capítulo 4 de resultados.

La guía a seguir inicia con la fase de identificación de riesgos propuesto en la sección 4.2 de resultados y que se puede consultar en el Anexo A, posteriormente se debe realizar el análisis de los riesgos identificados como se indica en la sección 4.3, una vez evaluados estos riesgos, se debe aplicar la fase de respuesta al riesgo propuesta en la sección 4.4 y que se puede observar en la plantilla construida en la Anexo B, para luego pasar al seguimiento y control de estos riesgos, planteado en la sección 4.5, la cual incluye la comunicación del riesgo a los interesados del proyecto, donde se pueden utilizar 2 herramientas propuestas y que se explican en la sección 4.5.1 del capítulo de resultados. Ver Anexo C.

6. GLOSARIO

Aceptar el riesgo: Una estrategia de respuesta a los riesgos según la cual el equipo del proyecto decide reconocer el riesgo y no tomar ninguna medida a menos que el riesgo ocurra.

Amenaza: Riesgo que tendría un efecto negativo sobre uno o más objetivos del proyecto.

Brainstorming: Lluvia de ideas.

Daily Standup Meeting: Cada día de un sprint, se realiza la reunión sobre el estado de un proyecto. La reunión tiene una duración fija de 15 minutos. Cada miembro del equipo responde 3 preguntas: ¿Qué hiciste/lograste ayer?, ¿Qué vas a hacer/lograr hoy?, ¿Qué impedimentos tienes para lograrlo?

Evitar el riesgo: Una estrategia de respuesta a los riesgos según la cual el equipo del proyecto actúa para eliminar la amenaza o proteger al proyecto de su impacto.

Gestión de riesgos: Es un enfoque estructurado para manejar la incertidumbre relativa a una amenaza, a través de una secuencia de actividades humanas que incluyen evaluación de riesgo, estrategias de desarrollo para manejarlo y mitigación del riesgo utilizando recursos gerenciales.

Las estrategias incluyen transferir el riesgo a otra parte, evadir el riesgo, reducir los efectos negativos del riesgo y aceptar algunas o todas las consecuencias de un riesgo particular. El objetivo de la gestión de riesgos es reducir diferentes riesgos relativos a un ámbito preseleccionado a un nivel aceptado por la sociedad.

Guía: Es un documento que contiene en forma ordenada y sistemática las instrucciones e información sobre los procedimientos de cada una de las actividades que se realizan para ejecutar más adecuadamente el trabajo, señalando y estableciendo los canales de comunicación entre sus distintas dependencias en forma coherente. Es el documento que contiene la descripción de actividades que deben seguirse en la realización de las funciones de una unidad administrativa, o de dos ó más de ellas.

Kanban: Es una palabra Japonesa que literalmente significa "Tarjetas visuales". Estos métodos ya tienen una larga trayectoria en las cadenas de producción, transmitido a desarrollo de software es bastante reciente de 2004. Es un método visual muy recomendado para gestionar proyectos donde los requisitos cambian constantemente.

Metodología ágil: El desarrollo ágil de software refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan mediante la colaboración de grupos auto organizado y multidisciplinario. Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en lapsos cortos.

El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar de una a cuatro semanas.

Mitigar el Riesgo: Una estrategia de respuesta a los riesgos según la cual el equipo del proyecto actúa para reducir la probabilidad de ocurrencia o impacto de un riesgo.

Product Owner: Representa al cliente, y es el encargado de negociar con el equipo, con el Scrum Master por medio como facilitador, la prioridad del trabajo a realizar. Esto desde una perspectiva del retorno de inversión para el negocio.

Retrospect Sprint: Este término también se aplica a la ingeniería de software, en cuyo marco una retrospectiva refiere a una reunión cuando se ha finalizado un proyecto o se está cerca de finalizarlo, y que trata sobre los aciertos y errores del mismo, su proyección y posibilidades a futuro, las enseñanzas que pueden extraerse de este emprendimiento.

Riesgo: es un evento o condición incierta que, de producirse, tiene un efecto positivo o negativo en uno o más de los objetivos del proyecto, tales como el alcance, el cronograma, el costo y la calidad.

Risk Breakdown Structure (RBS): Estructura de Desglose del Riesgo (RBS). Es una representación jerárquica de los eventos inciertos, los cuales son identificados y ordenados por categoría de riesgo y subcategoría, reconociendo las distintas áreas y causas de probables riesgos.

Risk Checklists: Lista de control de riesgos. Las listas de verificación son herramientas valiosas para ayudar a asegurar un nivel de identificación de riesgo detallado.

Risk Prompt Lists: Se utilizan para estimular pensamientos con respecto a la fuente de donde los riesgos se pueden originar.

SEI: Software Engineering Institute (SEI) es un instituto federal estadounidense de investigación y desarrollo, fundado por Congreso de los Estados Unidos en 1984 para desarrollar modelos de evaluación y mejora en el desarrollo de software, que dieran respuesta a los problemas que generaba al ejército estadounidense la programación e integración de los sub-sistemas de software en la construcción de complejos sistemas militares. Financiado por el Departamento de Defensa de los Estados Unidos y administrado por la Universidad Carnegie Mellon. Es un referente en Ingeniería de Software por realizar el desarrollo del modelo SW-CMM (1991) que ha sido el punto de arranque de todos los que han ido formando parte del modelo que ha desarrollado sobre el concepto de capacidad y madurez, hasta el actual CMMI.

Scrum: Scrum es un modelo de desarrollo ágil caracterizado por:

- Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
- Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados.
- Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o de cascada.

SCRUM es un modelo de referencia que define un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto.

Spike: Un (spike) pico es un experimento que consiste en la investigación o la creación de prototipos para entender mejor los riesgos potenciales en Scrum.

Sprint: Iteración. En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de un mes natural y hasta de dos semanas). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto que sea potencialmente entregable, de manera que cuando el cliente (Product Owner) lo solicite sólo sea necesario un esfuerzo mínimo para que el producto esté disponible para ser utilizado.

Scrum Master: Rol dentro de Scrum. El Scrum Master es un "líder servicial", que ayuda al resto del equipo Scrum a seguir su proceso. Debe tener una buena comprensión de Scrum y la habilidad de capacitar a otros en sus sutilezas.

SCRUMstudy: Es el organismo de acreditación mundial para Scrum y Agile certificaciones. Ha sido autor de la Guía SBOK ™ como una guía completa para entregar proyectos exitosos utilizando Scrum.

Spring Planning: Es una ceremonia o reunión propia de Scrum, se realiza al inicio de cada sprint y pretender seleccionar y definir qué trabajo se hará durante el sprint, detallando las actividades a realizar en cada historia de usuario comprometida.

Sprint Review: Ceremonia o reunión de Scrum, que se realiza al finalizar el sprint y tiene como objetivo revisar el trabajo realizado, informar si fue terminado por completo o no el total comprometido en el sprint, se muestra al Product Owner los resultados y se hacen pruebas sobre lo entregado.

Scrum Team: Es el equipo de trabajo. El equipo tiene la responsabilidad de entregar el producto. Un pequeño equipo de 3 a 9 personas con las habilidades transversales necesarias para realizar el trabajo (análisis, diseño, desarrollo, pruebas, etc.).

Transferir el Riesgo: Una estrategia de respuesta a los riesgos según la cual el equipo del proyecto traslada el impacto de una amenaza a un tercero, junto con la responsabilidad de la respuesta.

Taxonomía: Clasificación u ordenación en grupos de cosas que tienen unas características comunes.

XP: Método ágil para la gestión de desarrollo software.

ANEXOS

ANEXO A: PLANTILLA IDENTIFICACIÓN DE RIESGOS PARA SCRUM

A. INGENIERÍA DEL PRODUCTO			CERTEZA		VALORACIÓN		EXPOSICIÓN	PRIORIDAD
CATEGORIA		PREGUNTAS						
Historias de usuario	Completitud	Las historias de usuario no están bien especificadas	Probable	0,5	Muy Alto	0,8	0,40	Alto
	Claridad	No se entienden las historias o criterios de aceptación completamente	Probable	0,5	Muy Alto	0,8	0,40	Alto
	Valides	Los criterios de aceptación de las historias no especifican lo que el cliente desea	Probable	0,5	Muy Alto	0,8	0,40	Alto
	Viabilidad	La historias son difíciles de implementar	Probable	0,5	Alto	0,4	0,20	Alto
Codificación y Pruebas	Pruebas	No hay tiempo suficiente para realizar pruebas	Altamente probable	0,7	Moderado	0,2	0,14	Medio
	Codificación e implementación	Dificultades para codificar debido a factores técnicos	Probable	0,5	Alto	0,4	0,20	Alto
		Se usan herramientas de desarrollo NO adecuadas	Poco probable	0,3	Moderado	0,2	0,06	Medio
		NO hay replicación de ambientes de desarrollo con ambientes destino	Poco probable	0,3	Moderado	0,2	0,06	Medio
Integración y pruebas	Entorno	NO existen las plataformas adecuadas para integraciones y pruebas	Poco probable	0,3	Moderado	0,2	0,06	Medio
	Producto	NO hay disponibilidad de los ambientes destino cuando se requieran	Poco probable	0,3	Moderado	0,2	0,06	Medio
		Requerimientos difíciles de verificar	Probable	0,5	Alto	0,4	0,20	Alto
		No hay tiempo requerido para pruebas e integración	Altamente probable	0,7	Alto	0,4	0,28	Alto
B. ENTORNO DE DESARROLLO								
CATEGORIA		PREGUNTAS						
Proceso de Desarrollo	Familiaridad	La gente está incómoda con el proceso de desarrollo	Poco probable	0,3	Alto	0,4	0,12	Medio
	Control de Producto	No existe un proceso formal de control de cambios	Probable	0,5	Moderado	0,2	0,10	Medio
Sistema de Desarrollo	Familiaridad	Hay poca experiencia en los miembros del proyecto con el sistema de desarrollo	Poco probable	0,3	Alto	0,4	0,12	Medio
	Soporte del sistema	No hay expertos o proveedores de apoyo para el sistema	Altamente probable	0,7	Bajo	0,1	0,07	Medio
Ambiente de Trabajo	Actitud de Calidad	El personal NO está orientado hacia los procedimientos de calidad	Altamente probable	0,7	Moderado	0,2	0,14	Medio
	Cooperación	La gente NO trabaja con eficacia hacia los objetivos comunes	Poco probable	0,3	Moderado	0,2	0,06	Medio
	Comunicación	NO hay una buena comunicación entre los miembros del equipo	Poco probable	0,3	Moderado	0,2	0,06	Medio
C. LIMITACIONES DEL PROGRAMA								
CATEGORIA		PREGUNTAS						
Recursos	Personal	Existen áreas en las que las habilidades técnicas requeridas están faltando	Poco probable	0,3	Alto	0,4	0,12	Medio
		No cuenta con el personal adecuado para el proyecto	Muy improbable	0,1	Alto	0,4	0,04	Bajo
	Presupuesto	El presupuesto NO está basado sobre una estimación realista	Poco probable	0,3	Moderado	0,2	0,06	Medio
		Existe alguan actividad para la cual no es suficiente el presupuesto asignado	Poco probable	0,3	Moderado	0,2	0,06	Medio
Contrato	Dependencias	Hay dependencias externas o servicios que puedan afectar el proyecto o el presupuesto	Probable	0,5	Alto	0,4	0,20	Alto

Taxonomia_Tesis_SEI.xlsx

ANEXO B: RESPUESTA AL RIESGO

A. INGENIERÍA DEL PRODUCTO			**RESPUESTA AL RIESGO**	**MEDIDAS PARA APLICAR**
CATEGORIA		PREGUNTAS		
Historias de usuario	Completitud	Las historias de usuario no están bien especificadas	EVITAR	• Recopilar la mayor cantidad de información relacionada con las historias de usuario • Realizar una adecuada planeación de reuniones que involucren los Stakeholders apropiados • Realizar sondeo de percepción sobre la historia de usuario con el fin de lograr consenso en su definición.
	Claridad	No se entienden las historias o criterios de aceptación completamente	EVITAR	• Recopilar la mayor cantidad de información relacionada con los criterios de aceptación • Realizar una adecuada planeación de reuniones que involucren el Producto Owner • Realizar sondeo de percepción sobre los criterios de

				aceptación con el fin de lograr consenso en su definición.
	Valides	Los criterios de aceptación de las historias no especifican lo que el cliente desea	EVITAR	• Recopilar la mayor cantidad de información relacionada con los criterios de aceptación • Realizar una adecuada planeación de reuniones que involucren el Producto Owner • Realizar sondeo de percepción sobre los criterios de aceptación con el fin de lograr consenso en su definición.
	Viabilidad	La historias son difíciles de implementar	MITIGAR	• Adopción de procesos menos complejos • División de funcionalidades en historias más pequeñas para el tiempo definido del sprint
Codificación y Pruebas	Pruebas	No hay tiempo suficiente para realizar pruebas	MITIGAR	• Priorizar las pruebas planeadas para el sprint • Ejecutar las pruebas más importantes para el sprint
	Codificación e implementación	Dificultades para codificar debido a factores técnicos	MITIGAR	• Realizar plan de contingencia

		Se usan herramientas de desarrollo NO adecuadas	MITIGAR	• Identificar herramientas requeridas para el proyecto • Verificar disponibilidad de herramientas adecuadas • Adopción de las herramientas adecuadas
		NO hay replicación de ambientes de desarrollo con ambientes destino	TRANSFERIR	• Revisión de las consideraciones y acuerdos sobre infraestructura requerida por el proyecto • Realizar plan de contingencia
Integración y pruebas	Entorno	NO existen las plataformas adecuadas para integraciones y pruebas	TRANSFERIR	• Revisión de las consideraciones y acuerdos sobre infraestructura requerida por el proyecto • Realizar plan de contingencia
	Producto	NO hay disponibilidad de los ambientes destino cuando se requieran	TRANSFERIR	• Revisión de las consideraciones y acuerdos sobre infraestructura requerida por el proyecto • Realizar plan de contingencia
		Requerimientos difíciles de verificar	EVITAR	• Recopilar la mayor cantidad de información relacionada con las historias de usuario • Realizar una adecuada planeación de reuniones que involucren los Stakeholders apropiados

				• Realizar sondeo de percepción sobre la historia de usuario con el fin de lograr consenso en su definición.
		No hay tiempo requerido para pruebas e integración	MITIGAR	• Priorizar las pruebas planeadas para el sprint • Priorizar las integraciones planeadas para el sprint • Ejecutar las pruebas más importantes para el sprint • Realizar las integraciones más importantes para el sprint
B. ENTORNO DE DESARROLLO				
CATEGORIA		PREGUNTAS		
Proceso de Desarrollo	Familiaridad	La gente está incómoda con el proceso de desarrollo	MITIGAR	• Realizar reuniones de clima organizacional • Ejecutar planes de motivación • Realizar ajustes al interior del equipo
	Control de Producto	No existe un proceso formal de control de cambios	ACEPTAR	• Gestionar los controles de cambio según se requiera

Sistema de Desarrollo	Familiaridad	Hay poca experiencia en los miembros del proyecto con el sistema de desarrollo	MITIGAR	• Realizar un sondeo sobre el nivel de experiencia de los participantes del equipo • Planear las capacitaciones requeridas según requerimientos tecnológicos del proyecto
	Soporte del sistema	No hay expertos o proveedores de apoyo para el sistema	ACEPTAR	• Solicitar los apoyos de expertos según se requiera
Ambiente de Trabajo	Actitud de Calidad	El personal NO está orientado hacia los procedimientos de calidad	EVITAR	• Realizar una adecuada inmersión al equipo de trabajo en los procedimientos de calidad esperados • Ajustar el procedimiento de calidad a las características del proyecto • Involucrar personal experto en gestión de calidad para apoyar procesos de adopción
	Cooperación	La gente NO trabaja con eficacia hacia los objetivos comunes	EVITAR	• Realizar una adecuada inmersión al equipo de trabajo en los objetivos esperados para el proyecto • Realizar acompañamiento en la evaluación de los logros alcanzados por el equipo para reorientar de ser

				necesario
	Comunicación	NO hay una buena comunicación entre los miembros del equipo	EVITAR	Implementar canales de comunicación efectivos
C. LIMITACIONES DEL PROGRAMA				
CATEGORIA		PREGUNTAS		
Recursos	Personal	Existen áreas en las que las habilidades técnicas requeridas están faltando	MITIGAR	• Realizar un sondeo sobre el nivel de experiencia de los participantes del equipo • Planear las capacitaciones requeridas según requerimientos tecnológicos del proyecto
		No cuenta con el personal adecuado para el proyecto	EVITAR	• Realizar un sondeo sobre el nivel de experiencia de los participantes del equipo • Planear las capacitaciones requeridas según requerimientos tecnológicos del proyecto
	Presupuesto	El presupuesto NO está basado	ACEPTAR	• Ajustar la curva de estimación planeada

		sobre una estimación realista		• Ajustar alcance al finalizar cada sprint • Identificar el presupuesto real
		Existe alguna actividad para la cual no es suficiente el presupuesto asignado	ACEPTAR	• Ajustar la curva de estimación planeada • Ajustar alcance al finalizar cada sprint • Identificar el presupuesto real
Contrato	Dependencias	Hay dependencias externas o servicios que puedan afectar el proyecto o el presupuesto	EVITAR	• Realizar un listado de las dependencias externas al proyecto y recopilar la mayor cantidad de información • Informar al Product Owner sobre las posibles incidencias • Generar un plan de acción

Respuesta_Al_Riesgo.xlsx

ANEXO C: DETALLE PASO A PASO DE USO DE LA GUÍA PARA GESTIONAR LOS RIESGOS EN LOS PROYECTOS DE DESARROLLO DE SOFTWARE BAJO MARCO DE TRABAJO SCRUM

1. Identificación de Riesgos

Crear la matriz de riesgos para el proyecto con base en la lista maestra de riesgos. Ver anexo A.

En la lista maestra de riesgos, se deben identificar por cada pregunta, el grado de certeza esperado del evento, para determinar la probabilidad esperada de ocurrencia de dicho evento.

Cada riesgo identificado, se debe colocar en el tablero Kanban

Los valores posibles dentro de la lista de certeza son:

- Muy Improbable (0,1). Es una certeza que difícilmente o nunca ocurriría, pero que sin embargo, podría llegar a pasar en los peores casos.
- Poco Probable (0,3). Es una certeza que puede ocurrir pero, con condiciones muy específicas, y que se puede predecir.
- Probable (0,5). Es una certeza que puede ocurrir o no. No hay determinantes específicos para la ocurrencia.
- Altamente probable (0,7). Es una certeza que ocurrirá, con condiciones muy especificas
- Casi Cierto (0,9). Es una certeza que ocurrirá, en cualquier caso.

2. Análisis de Riesgos

Obtener para cada riesgo en la matriz de riesgos la valoración, seleccionando un nivel de valoración. Por cada nivel de valoración, se aplicará una probabilidad. Esta nueva probabilidad, será usada para obtener el valor de exposición.

Los valores posibles dentro de la valoración son:

- Muy Bajo (0,05). Indica que el impacto en el proyecto no es de consideración.
- Bajo (0,1). Indica que si bien, el riesgo es de poca magnitud, se debe tener en consideración.
- Moderado (0,2). Indica que el riesgo, se debe considerar, y revisar con cautela.
- Alto (0,4). Indica que el riesgo, representa un impacto considerable para el proyecto.
- Muy Alto (0,8). Indica que el riesgo es demasiado peligroso para los objetivos del proyecto.

El valor de la exposición es calculado de forma automática por la matriz de riesgos, con la fórmula establecida para tal:

Exposición = Certeza * Valoración

La exposición, es el valor que se calcula para establecer el grado de prioridad que tendrá en riesgo. Esto ayudará a establecer el orden de ejecución de los riesgos.

El valor de prioridad, se calcula de forma automática por la matriz de riesgos, con la fórmula establecida para tal:

Prioridad = Alta para valores > 0,15

Prioridad = Media para valores entre 0,05 y 0,15 excluyentes.

Prioridad = Baja para valores entre < 0,05

Esta priorización, ayudará al equipo de trabajo a concentrar los esfuerzos en mitigar los riesgos más importantes o clasificados como “Altos”, de manera que impacten menos los objetivos propuestos para cada sprint.

3. Respuesta al Riesgo

Mitigar el riesgo, según las estrategias propuestas en la matriz del Anexo B de respuesta al riesgo. En esta matriz están cubiertas todas las estrategias para mitigar los riesgos, que la empresa haya determinado.

4. Seguimiento y Control del Riesgo

Verificar el estado del riesgo según las acciones tomadas, en los tableros Kanban y en el calendario de Incidentes.

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

BIBLIOGRAFÍA REFERENCIADA

Akif, R., & Majeed, H. Issues and Challenges in Scrum Implementation. Traducción propia. Recuperado de http://www.ijser.org/researchpaper%5CIssues-and-Challenges-in-Scrum-Implementation.pdf

Bahit, Eugenia. Ceremonias en Scrum. Septiembre 2011. Recuperado de: http://www.desarrolloweb.com/articulos/ceremonias-scrum.html

Bashir, M. S., & Qureshi, M. R. J. HYBRID SOFTWARE DEVELOPMENT APPROACH FOR SMALL TO MEDIUM SCALE PROJECTS: RUP, XP & SCRUM. Cell, 966, 536474921. Traducción propia.

Bernal, César A. Metodología de la investigación. Tercera Edición. Pearson. Colombia 2010. Pág. 160-161.

Bhatti, M. (2012). EMGT 835 FIELD PROJECT: Implementing Scrum in Distributed Teams (Doctoral dissertation, The University of Kansas). Traducción propia. Recuperado de http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4076936&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4076936

Borrego, G., Portela, T., Tolano, K., Cruz, J., Amavizca, L., & Vázquez, J. (2013). Experiencias del uso de una metodología semi-ágil como apoyo en la enseñanza de la ingeniería de software. In Congreso Nacional de Tecnologías de la Información y Comunicación: CONATIC 2013 (p. 36, 37). Recuperado de: http://www.conatic2014.com.mx/conatic2013/CONATIC2013.pdf#page=36

Carr, J. Marvin, Konda, Suresh L. Taxonomy-Based Risk Identification. Technical Report. CMU/SEI-93-TR-6. ESC-TR-93-183. Junio 1993. Software Engineering Institute. Carnegie Mellon University. Pittsburgh, Pennsylvania 15213. Páginas: 7, Anexo A. Recuperado de: http://www.sei.cmu.edu/reports/93tr006.pdf

Castillo Brenes, Eddy Mysael. Proyecto de elaboración de la metodología de gestión de riesgos en proyectos de desarrollo de software para la empresa consultora CV3. Junio 2009. Recuperado de: http://www.uci.ac.cr/Biblioteca/Tesis/PFGMAP655.pdf

Colla, P. E. Marco para evaluar el valor en metodología SCRUM. 2012. Recuperado de: http://www.41jaiio.org.ar/sites/default/files/086_ASSE_2012.pdf

Conferencia: BOGOTÁ: METODOLOGÍA ÁGIL QUE APLICA A CUALQUIER PROYECTO. Agosto 25, 2014. Bogotá Auditorios Clínica de la Mujer, Carrera 19 C # 90 - 30, Piso 6. Invitado especial: Ralf Bühl M. - Altus Training-Coaching-Projects SAS. Recuperado de: http://pmicolombia.org/events/conferencia-una-metodologia-agil-que-aplica-cualquier-proyecto/

Dalipi, F., Rufati, E., & Idrizi, F. Applying Agile Software Development Methodology in a Dynamic Business Environment. Página 24. Traducción propia. Recuperado de http://files.figshare.com/1276484/D08032025.pdf

de Barros Jerónimo, T., & de Medeiros, D. D. Scrum As Community of Practice to Small and Medium-Sized High Technology Enterprises Realize the Strategic Plan. Traducción propia.

Estrella, Mario. Scrum en muchas palabras. Diciembre 2007. Recuperado de: http://scrumenespanol.blogspot.com/2007/12/scrum-en-muchas-palabras.html

Freire, S. Agile Challenges: Achieving self-organizing status and adopting CMMI principles. Traducción propia.

Gracia Peña, R. (2013). Gestión de proyectos con metodologías ágiles. Recuperado de: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/2/mtrigasTFC0612presentacion.pdf

Gran Bretanya. Office of Government Commerce. Éxito en la gestión de proyectos con PRINCE2. Publisher Stationery Office, TSO, 2009. Recuperado de: http://books.google.com.co/books?id=2dwfqoHrfZoC&pg=PA93&lpg=PA93&dq=probabilidad,+proximidad+e+impacto&source=bl&ots=NMIU3dN3eI&sig=NhHvsOG5TnCNqGcR4YIrQgO7J7M&hl=es-419&sa=X&ei=QVNqVNK4IImaNtCkg_AJ&ved=0CB0Q6AEwAA#v=onepage&q=probabilidad%2C%20proximidad%20e%20impacto&f=false

Gutiérrez, F. A. (2013). INTEGRACIÓN DE ADM Y MÉTODOS DE DESARROLLO DE SOFTWARE. Tecnología, Investigación y Académia, 1(1), 49-56. Revista Digital Tecnología, Investigación y Academia TIA [Vol.1] [No.1]. Recuperado de: http://ingenieria.udistrital.edu.co/digital/index.php/tia/article/view/165/243

Haydary, Y. (2013). Major differences between Scrum & RUP. Traducción propia. Disponoble en http://softwareprocess.jeroenpeeters.nl/images/a/af/RUP_vs_Scrum.pdf

Hernández Sampieri, Roberto. Metodología de la investigación. McGraw-Hill. 1991. Recuperado de: http://www.upsin.edu.mx/mec/digital/metod_invest.pdf

Jeldi, N. P., & Chavali, V. K. M. Software Development Using Agile Methodology Using Scrum Framework. Traducción propia. Recuperado de http://www.ijsrp.org/research-paper-0413/ijsrp-p16119.pdf

Mendoza Palacios, Rudy. Investigación cualitativa y cuantitativa - Diferencias y limitaciones. 2006. Recuperado de: http://www.monografias.com/trabajos38/investigacion-cualitativa/investigacion-cualitativa.shtml

Menezes Jr, J., Gusmão, C., & Moura, H. (2013). Defining Indicators for Risk Assessment in Software Development Projects. CLEI Electronic Journal, 16(1), 11-11. Traducción propia. Recuperado de http://www.scielo.edu.uy/scielo.php?pid=S0717-50002013000100011&script=sci_arttext

Pérez, M. J. (s.f.). 2012. Guía Comparativa de Metodologías Ágiles. Universidad de Valladolid. Segovia. Obtenido de https://uvadoc.uva.es/bitstream/10324/1495/1/TFG-B.117.pdf

Plinio Puello Marrugo, Julio Rodríguez Ribón, Amaury Cabarcas Álvarez. Scrum: conceptos y aplicaciones open source. INGENIATOR. Volumen 3, Número 3, 2011, página 11. Recuperado de: http://letravirtual.usbctg.edu.co/index.php/ingeniator/article/viewFile/185/202

Preis, A. (2012). Integration Evaluation of Scrum and CMMI. Traducción propia. Recuperado de http://www.arminpreis.at/files/2012/pub/PREIS-Integrate-CMMI-Scrum-Projectmanagement.pdf

Priyanka Hasija. Julio 2012. My Experience as a QA in Scrum. Traducción propia. Recuperado de: http://www.infoq.com/articles/experience-qa-scrum

Puello Marrugo, P., Rodríguez Ribón, J., & Cabarcas Álvarez, A. (2012). Scrum: conceptos y aplicaciones Open Source. INGENIATOR, 2(3). Recuperado de: http://letravirtual.usbctg.edu.co/index.php/ingeniator/article/view/185/202 Ghosh, S., Forrest, D., DiNetta, T., Wolfe, B., & Lambert, D. C. (2012). Enhance PMBOK® by Comparing it with P2M, ICB, PRINCE2, APM and Scrum Project Management Standards. PM World Today, 14(1). Páginas 30 y 33. Traducción propia. Recuperado de http://www.theopengroup-library.com/Player/eKnowledge/comparison-of-pm-frameworks.pdf

Ravi, S. P., Reddaiah, B., Movva, L. S., & Kilaparthi, R. A Critical review and empirical study on success of risk management activity with respect to scrum. Páginas 469 y 472. Traducción propia. Recuperado de http://www.estij.org/papers/vol2no32012/17vol2no3.pdf

Rivadeneira, S., Vilanova, G., Miranda, M., & Cruz, D. (2013, June). El modelado de requerimientos en las metodologías ágiles. In XV Workshop de Investigadores en Ciencias de la Computación. Páginas 383-387. Recuperado de: http://sedici.unlp.edu.ar/bitstream/handle/10915/27196/Documento_completo.pdf?sequence=1

Russell Pannone. Quality Management in the World of Scrum and Agile IT System Development. Marzo de 2009. Traducción propia. Recuperado de: http://www.agileconnection.com/article/quality-management-world-scrum-and-agile-it-system-development?page=0%2C1

Santimacnet. Curso gratis Scrum día a día. Noviembre 2010. Recuperado de: https://santimacnet.wordpress.com/category/metodologias-agiles/page/2/

SCRUMstudy. Una guía para el CONOCIMIENTO DE SCRUM. (GUÍA SBOK™). Edición 2013. Arizona USA. SCRUMstudy™.Página: 120-126.

Shankarmani, R., Mantha, S. S., & Babu, V. Inclusion of e-Assist to increase Agile Adoption. Traducción propia. Recuperado de http://www.ijsrp.org/research-paper-1112/ijsrp-p1108.pdf

Stålhane, T., Myklebust, T., & Hanssen, G. K. (2012). The application of Safe Scrum to IEC 61508 certifiable software. ESREL. Traducción propia. Recuperado de https://www.nbl.sintef.no/upload/final-PSAM-11-ESREL-2012.pdf

Stålhane–IDI, T. Safety standards and Scrum–A synopsis of three standards. Traducción propia. Recuperado de http://www.sintef.org/upload/IKT/9013/Safety%20standards%20and%20Scrum_May2013.pdf

Sutherland J., Schwaber K. The Scrum Papers: Nuts, Bolts, and Origins of an Agile Process [Internet]. [Consultado 2014 abril 5]. Traducción propia. Recuperado de: http://assets.scrumfoundation.com/downloads/2/scrumpapers.pdf?1285932052

Sutherland J., Viktorov A., Blount J., Puntikov N. (2007). Distributed Scrum: Agile Project Management with Outsourced Development Teams [Internet]. [Consultado 2014 Abril 5]. Traducción propia. Recuperado de: http://jeffsutherland.com/SutherlandDistributedScrumHICCS2007.pdf

Toapanta C., Kleber M. Artículo Científico - Método Ágil Scrum, aplicado a la implantación de un sistema informático para el proceso de recolección masiva de información con Tecnología Móvil [Internet]. [Consultado 2014 marzo 25]. Traducción propia. Recuperado de: http://repositorio.espe.edu.ec/handle/21000/5899

Vähä-Sipilä, A. (2013). Product security risk management in agile product management. OWASP AppSec Research 2010 - Stockholm, Sweden. Páginas 17 y 25. Traducción propia. Recuperado de https://owasp.org/images/c/c6/OWASP_AppSec_Research_2010_Agile_Prod_Sec_Mgmt_by_Vaha-Sipila.pdf

Verner, J. M., Brereton, O. P., Kitchenham, B. A., Turner, M., & Niazi, M. (2012). Risk Mitigation Advice for Global Software Development from Systematic Literature Reviews. Traducción propia. Recuperado de http://www.keele.ac.uk/media/keeleuniversity/facnatsci/scm/e-risk/Mitigation%20TR-2012-02%20Feb%202012.pdf

Vikas Hazrati. Managing Risk with Scrum. Julio 2008. Traducción propia. Recuperado de: http://www.infoq.com/news/2008/07/managing-risk-with-scrum/

Washington, G. (2003). Using risk to balance agile and plan-driven methods. Traducción propia. Recuperado de http://www.eici.ucm.cl/Academicos/ygomez/descargas/ing_sw1/Using%20Risk%20to%20Balance%20Agile%20and%20Plan-Driv.pdf

Xavier Albaladejo. Agilidad es calidad y competitividad. Octubre del 2009. Evento Agile Open Spain 2009. Recuperado de: http://www.proyectosagiles.org/agilidad-calidad-competitividad

Xavier Albaladejo. Calidad y agilidad - Resultados del cuarto encuentro ágil en Barcelona. Mayo 2009. Recuperado de: http://www.proyectosagiles.org/calidad-agilidad-cuarto-encuentro-agil-barcelona

Yu, Xiaodan, Petter, Stacie. Understanding agile software development practices using shared mental models theory. Information and Software Technology. Elsevier. Volumen 56, número 8, Agosto 2014, páginas 911 - 921. Traducción propia. Recuperado de: http://dx.doi.org.ezproxy.uniminuto.edu:8000/10.1016/j.infsof.2014.02.010

BIBLIOGRAFÍA CONSULTADA

Acosta Rodríguez, Vanessa. Importancia de la gerencia de riesgos en el desarrollo de un proyecto. Abril 2009. Recuperado de: http://ri.bib.udo.edu.ve/bitstream/123456789/1023/1/TESIS.Gerencia%20de%20riesgos.pdf

Babu Yegi, Sanni. Risk and Issue Management in the Scrum Process. Abril 2014. Traducción propia. Recuperado de: https://www.scrumalliance.org/community/articles/2014/april/risk-and-issue-management-in-scrum-process

Bastardo E., Francisco A (Mayo 2010). Diseño de un Modelo de Gestión para la Administración y Control de los proyectos en desarrollo de la empresa IMPSA Caribe, C.A. Tesis de Maestría en Ingeniería Industrial. Universidad Nacional Experimental Politécnica "Antonio José de Sucre", Puerto Ordaz. Recuperado de: http://www.monografias.com/trabajos-pdf4/modelo-gestion-administracion-y-control-proyectos-impsa-caribe-ca/modelo-gestion-administracion-y-control-proyectos-impsa-caribe-ca.pdf

Bird, Jim. Why isn't Risk Management included in Scrum?. Mayo 2010. Recuperado de: http://swreflections.blogspot.com/2010/05/why-isnt-risk-management-included-in.html

Cohn, Mike. Managing Risk on Agile Projects with the Risk Burndown Chart. Abril 2010. Recuperado de: http://www.mountaingoatsoftware.com/blog/managing-risk-on-agile-projects-with-the-risk-burndown-chart

Del Risco Serje, Vanessa Rocío (2013). Análisis cualitativo de factores de riesgos financieros en proyectos de construcción de tipo residencial en la ciudad de Cartagena bajo la metodología del PMI®. Caso de estudio: Edificio Portovento. Tesis de pregrado para Ingeniería Civil.

Universidad de Cartagena, Cartagena de Indias. Recuperado de: http://190.25.234.130:8080/jspui/bitstream/11227/302/1/TESIS%20FINAL.pdf

Glen B. Alleman. Como elaborar la matriz de riesgos del proyecto. Abril 2013. Recuperado de: http://ivanrivera-pmp.blogspot.com/2013/04/como-elaborar-la-matriz-de-riesgos-del.html

Hazrati, Vikas. Managing Risk with Scrum. Julio 2008. Recuperado de: http://www.infoq.com/news/2008/07/managing-risk-with-scrum/

J. Esteves, Instituto de Empresa, J.A. Pastor, Universidad Internacional de Catalunya, N. Rodríguez, Universidad Politécnica de Catalunya, & R. Roy, Universidad Politécnica de Catalunya. Implementación y Mejora del Método de Gestión Riesgos del SEI en un proyecto universitario de desarrollo de software. IEEE LATIN AMERICA TRANSACTIONS, vol. 3, No. 1, Marzo de 2005. Recuperado de: http://www.ewh.ieee.org/reg/9/etrans/ieee/issues/vol03/vol3issue1March2005/3TLA1_13Esteves.pdf

Martínez Barranco, María Cristina. RDS 2 – Plan Estratégico 2-4. Abril 2011. Recuperado de: http://www.todostartups.com/actualidad/rds-2-plan-estrategico-2-4

Monje Álvarez, Carlos Arturo (2011). Metodología de la Investigación cuantitativa y cualitativa. 2011. Tesis de Maestría en Comunicación. Universidad Surcolombia, Neiva. Recuperado de: http://carmonje.wikispaces.com/file/view/Monje+Carlos+Arturo+-+Gu%C3%ADa+did%C3%A1ctica+Metodolog%C3%ADa+de+la+investigaci%C3%B3n.pdf

Morris, Valerie. Managing Risk in Scrum, part 2. Noviembre 2011. Recuperado de: http://www.solutionsiq.com/managing-risk-in-scrum-part-2/

Payne, Richard. Does Scrum Eliminate Project Risk?. Marzo 2013. Recuperado de: http://www.scrumexpert.com/knowledge/does-scrum-eliminate-project-risk/

Portero Portero, Hernan Santiago. Risk in project management: Scrum vs Prince 2. Recuperado de: http://www.monografias.com/trabajos95/scrum-vs-prince2/scrum-vs-prince2.shtml

Presentación web, Recuperado de: http://www.lsi.us.es/docencia/get.php?id=473

Prince2 TM. Éxito en la Gestión de Proyectos con PRINCE2 TM. 2009. Recuperado de: http://books.google.com.co/books?id=2dwfqoHrfZoC&pg=PA93&lpg=PA93&dq=probabilidad,+proximidad+e+impacto&source=bl&ots=NMIU3dN3eI&sig=NhHvsOG5TnCNqGcR4YIrQgO7J7M&hl=es-419&sa=X&ei=QVNqVNK4IImaNtCkg_AJ&ved=0CB0Q6AEwAA#v=onepage&q=probabilidad%2C%20proximidad%20e%20impacto&f=false

Rayhan, Syed. 2008. A Practical Guide to implementing Agile QA process on Scrum Projects. Recuperado de: https://www.scrumalliance.org/system/resource_files/0000/0459/AgileQA.pdf

SCRUMstudy. Identifying Risk in Scrum. Noviembre 2013. Recuperado de: http://www.scrumstudy.com/blog/identifying-risk-in-scrum/

SCRUMstudy. Risk mitigation in Scrum. Febrero 2014. Recuperado de: http://www.scrumstudy.com/blog/risk-mitigation-in-scrum/

Sitio web. ¿Qué es el diagrama de pareto?. Recuperado de: http://www.quees.info/diagrama-de-pareto.html

Thekku Veethil, Satheesh. Risk Management in Agile. Mayo 2013. Recuperado de: https://www.scrumalliance.org/community/articles/2013/2013-may/risk-management-in-agile

Video. 2005. Recuperado de: http://www.ustream.tv/recorded/1395614

Video. 2005. Recuperado de: http://www.ustream.tv/recorded/1395897