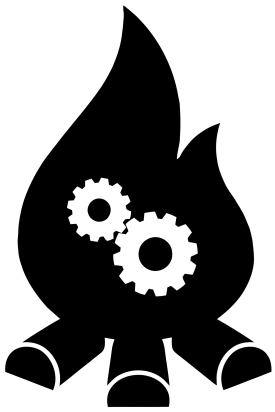
Software Project Management Plan



Fire Engineering

Karen Celis, Brayan García, Dorian Moreno, Camilo Muñoz, y Leyder Vera

Pontificia Universidad Javeriana

27 de Agosto de 2019

1. Historial de cambios

Fecha de cambio	Autor	Descripción de cambio
16/08/2019	Dorian Moreno	Creación del documento, títulos
18/08/2019	Karen Celis	Primera versión de la sección 6.3, 6.4, 6.5 y 6.6
18/08/2019	Camilo Muñoz	Primera versión de la sección 9.1
18/08/2019	Dorian Moreno	Primera versión de 7.2 y 7.3
18/08/2019	Brayan García	Primera versión sección 7.1
19/08/2019	Camilo Muñoz	Se agrega diagrama BPMN de sección 9.1, corrección sección 9.1
19/08/2019	Karen Celis	Primera versión de la sección 6.1 y 6.2
19/08/2019	Brayan García	Corrección sección 7.1
19/08/2019	Camilo Muñoz	Primera versión sección 9.2
20/08/2019	Leyder Vera	Primera versión sección 11.3 y 11.4
20/08/2019	Leyder Vera	Agregado diagrama BPMN de sección 11.4
21/08/2019	Camilo Muñoz	Primera versión sección 11.5
21/08/2019	Dorian Moreno	Segunda versión de sección 7.3 y primera versión de sección 8.2
21/08/2019	Camilo Muñoz	Agregado tabla de riesgos sección 11.5
22/08/2019	Karen Celis	Primera versión sección 10 y 11.1
22/08/2019	Dorian Moreno	Primera versión de 8.3 y segunda versión de 8.2
22/08/2019	Karen Celis	Corrección de la sección 6.2.3 y 6.3
22/08/2019	Camilo Muñoz	Corrección sección 9.2
23/08/2019	Camilo Muñoz	agregada sección 9.3
24/08/2019	Camilo Muñoz	Corrección sección 9.1
24/08/2019	Leyder Vera	Corrección diagramas sección 11.4
25/08/2019	Dorian Moreno	Corrección de WBS en sección 8.3
26/08/2019	Dorian Moreno	Versión final de FCL en sección 8.3
27/08/2019	Dorian Moreno	Corrección de WBS en sección 8.3
27/08/2019	Leyder Vera	Corrección diagrama BPMN sección 11.4.3

Tabla 1: Historial de cambios

2. Resumen

En el mundo más de la mitad de los animales de compañía no tienen hogar. Muchos porque tuvieron un hogar pero este los desechó como a un objeto ó se extraviaron, otros porque nunca lo han tenido.

PetFriend es un proyecto que busca disminuir el número de animales de compañía en la calle, usando el concepto de compañía tipo agente, en el que facilitamos la búsqueda, relación y comunicación entre la persona que tiene o vio el animal, con la persona interesada en adoptar o encontrar su mascota perdida.

En la parte inicial de este documento encontraremos la intención y propósito del proyecto, definiendo los objetivos, alcance, restricciones, y suposiciones, que son necesarias para el desarrollo. Después, se introducen los elementos claves que definen el trabajo de nuestro equipo, tales como el modelo de ciclo de vida, las herramientas a utilizar, la organización jerárquica de nuestro equipo y presupuesto.

Además, en la parte final podremos encontrar las estimaciones, análisis de riesgos, monitoreo del proyecto, control y evaluación de actividades, ambiente de trabajo y control de calidad del producto final.

3. Tabla de contenidos

Índice

1. Historial de cambios **2**

2. Resumen **3**

3. Tabla de contenidos **4**

4. Lista de figuras **6**

5. Lista de tablas **7**

6. Vista general del proyecto **8**

6.1. Visión del producto 8

6.2. Propósito, alcance y objetivos 8

6.2.1. Propósito 8

6.2.2. Alcance 8

6.2.3. Objetivos 9

6.3. Supuestos y restricciones 9

6.3.1. Supuestos 9

6.3.2. Restricciones 10

6.4. Entregables 11

6.5. Evolución del plan 11

6.6. Glosario 11

7. Contexto del proyecto **12**

7.1. Modelo de ciclo de vida 12

7.1.1. SCRUM 12

7.1.2. Extreme programming (XP) 13

7.1.3. Kanban 13

7.1.4. Conclusión de los modelos de ciclo de vida 13

7.2. Lenguajes y herramientas 15

7.3. Plan de aceptación del producto 18

7.4. Organización del proyecto y comunicación 19

7.4.1. Entidades externas o Stakeholders: 19

7.4.2. Organigrama y descripción de roles 20

8. Administración del proyecto **21**
8.1. Métodos y herramientas de estimación 21
8.1.1. Metodología: 21
8.1.2. Herramientas: 21
8.2. Inicio del proyecto 22
8.3. Planes de trabajo del proyecto 23

9. Monitoreo y control del proyecto **24**
9.1. Administración de requisitos 24
9.1.1. Planteamiento 24
9.1.2. Análisis y adecuación 24
9.1.3. Implementación 25
9.1.4. Entrega y retroalimentación 25
9.2. Monitoreo y control del progreso. 25
9.2.1. Unidades para medir el progreso. 25
9.2.2. Actividades para reportar el progreso 26
9.2.3. Acciones correctivas 26
9.3. Cierre del proyecto 27

10. Entrega del producto **27**

11. Procesos de soporte **27**
11.1. Ambiente de trabajo 27
11.2. Análisis y administración de riesgos 28
11.2.1. Matriz DOFA 29
11.3. Administración de configuración y documentación 30
11.4. Control de calidad. 32
11.4.1. Control de calidad de documentación. 32
11.4.2. Control de calidad de desarrollo. 33
11.4.3. Control de calidad del entregable final. 34

4. Lista de figuras

Índice de figuras

1. Organigrama 20
2. BPMN proceso de cambio a ítem de configuración. 32
3. BPMN proceso de control de calidad de documentación. 33
4. BPMN proceso de control de calidad de desarrollo. 34
5. BPMN proceso de control de calidad de entregable final. 35

5. Lista de tablas

Índice de cuadros

1. Historial de cambios 2
2. Lista de entregables 11
3. Comparación de metodologías 14
4. Criterios de aceptación de documentos 18
5. Criterios de aceptación del producto 19
6. Roles dentro del proyecto 20
7. Descripción de roles 21
8. Tareas periódicas para el correcto funcionamiento del proyecto. 23
9. Cabezotes del tablero de Kanban y SCRUM 28
10. Escala de riesgos 29
11. Matriz DOFA 29
12. Riesgos 30
13. Documentos de configuración. 31

6. Vista general del proyecto

6.1. Visión del producto

PetFriend busca ser una plataforma web de apoyo, preferida por aquellas personas amantes de los animales y su bienestar que quisieran adoptar una mascota, además de brindar un espacio para que las personas puedan publicar su mascota perdida e interactuar con otros usuarios.

La plataforma en desarrollo permitirá a sus usuarios buscar un animal de compañía para adoptar, publicar un animal para dar en adopción, animal perdido ó encontrado . Así mismo, el usuario puede ver las veterinarias, refugios y lugares de venta de productos para sus mascotas más cercanas y todos los servicios que prestan estos establecimientos.

6.2. Propósito, alcance y objetivos

6.2.1. Propósito

Hoy en día encontrar un sitio web que aglomere los animales en adopción es difícil de encontrar, es agotador para el usuario buscar en diferentes sitios y tal vez no concretar ninguna adopción o dado el caso no llegar a una búsqueda eficaz de su mascota si se le ha perdido. PetFriend tiene como propósito ser un canal sencillo y eficiente en donde se pueda publicar y buscar animales de compañía, para así poder darle una oportunidad a un animal de pertenecer a una familia que lo amará por toda su vida.

6.2.2. Alcance

Tener en el hogar un animal doméstico se ha vuelto común, es por esto que hoy en día las familias antes de tener hijos prefieren tener mascotas, pero muchas veces por tiempo, distancias, falta de información y desorden en las publicaciones, estos hogares no encuentran una mascota ideal. PetFriend propone ser una plataforma web que solucione los problemas mencionados anteriormente. A continuación se enuncian algunas características que se espera que tenga la plataforma:

- Crear un perfil de usuario para acceder a los servicios de la página.
- Publicar un animal en adopción para que el animal encuentre un hogar.
- Ver la lista de animales en adopción para encontrar un amigo.
- Buscar centros veterinarios y tiendas para animales para suplir las necesidades del animal.
- Ver la información y servicios que ofrecen las veterinarias.
- Reportar un animal encontrado para ayudar a su dueño a encontrarlo.
- Publicar un animal perdido para ayudar a recuperarlo.
- Contactar a otros usuarios vía chat para averiguar un poco más sobre un animal.

- Proporcionar retroalimentación para mejorar la experiencia en la página.
- Donar para ayudar al sostenimiento del proyecto.
- No se venderán animales.
- No se venderán productos para los animales.

6.2.3. Objetivos

General

- Desarrollar una plataforma web que mejore la búsqueda de animales de compañía, facilitando el proceso de adopción y recuperación de animales perdidos.

Específicos

- Incentivar la adopción de animales.
- Disminuir la población de animales perdidos.
- Concientizar a las personas de la cantidad de animales sin hogar.
- Facilitar la comunicación entre el adoptante y el hogar de paso.
- Aprender a utilizar herramientas para el desarrollo de software.
- Adquirir experiencia en cuanto al desarrollo de un proyecto de software utilizando los conocimientos adquiridos durante la carrera.
- Comprender lo valioso que es la buena planeación de un proyecto de software para entregar un producto de calidad.
- Aprender a utilizar estrategias y metodologías para el desarrollo de un proyecto de software.
- Entender lo importante que es el trabajo en equipo con roles definidos.

6.3. Supuestos y restricciones

6.3.1. Supuestos

- El grupo contará con disponibilidad y entrega para el desarrollo del trabajo que permita el cumplimiento del cronograma dentro de los marcos temporales establecidos.
- Las herramientas de desarrollo de software elegidas serán gratuitas y tendrán documentación para poder ser usadas en el transcurso del proyecto.
- En su totalidad los miembros del grupo cuentan con equipos que tienen los requisitos recomendados para el correcto funcionamiento del software de desarrollo.

- Se cuenta con un profesor de la asignatura el cual guía el proceso de aprendizaje y se encuentra en constante revisión del desarrollo y evolución del proyecto.
- Los usuarios de la plataforma estarán dispuestos a brindar información personal.
- Las personas no utilizarán esta plataforma para vender animales
- Las personas únicamente publicarán animales de compañía.
- Todos los animales en adopción están vivos.

6.3.2. Restricciones

- Es importante resaltar que un presupuesto real no existe debido a que este es un proyecto universitario con fines académicos desarrollado en el curso de Ingeniería de Software, dado lo anterior, solo se pueden utilizar herramientas dadas por la universidad o gratuitas .
- Los integrantes del equipo de trabajo son estudiantes de Ingeniería de Sistemas, es decir que puede que no estén capacitados para realizar todas las tareas necesarias para el desarrollo de un proyecto de software que aportan diferentes disciplinas, por ejemplo: animaciones, diseño gráfico, marketing, publicidad, etc. Esto limita algunas características del producto o su calidad final, ejemplo: el diseño de la ayuda audiovisual, interfaces, logos e iconos que tendrá la plataforma.
- El proyecto debe darse por finalizado antes de que finalice el semestre académico en marcha.
- Es necesaria la conexión a internet para que la plataforma funcione.
- La plataforma debe ser desarrollada bajo el paradigma orientado a objetos.
- La plataforma debe tener persistencia de los datos.
- El grupo de trabajo estará conformado únicamente por cinco (5) personas, y se espera que no cambie a lo largo del proyecto.
- La plataforma debe tener siempre conexión a internet.
- Las publicaciones deben ser de Bogotá.
- No se publicarán animales gestantes.

6.4. Entregables

Nombre	Descripción	Remitente	Fecha
SPMP	Documento del plan de administración del proyecto.	Cliente	27/08/2019
SRS, corrección SPMP y caso de uso más importante	Documento con la descripción del software y la corrección del SPMP. Aplicación con el caso de uso más importante funcionando correctamente.	Cliente	17/10/2019
Entrega final	Aplicación completa con la documentación, SDD y corrección del SRS	Cliente	17/11/2019

Tabla 2: Lista de entregables

6.5. Evolución del plan

A lo largo del desarrollo de este documento se utilizaron dos herramientas para controlar los cambios y modificaciones del documento. El primero de ellos es la utilización del historial de cambios de LaTex con la herramienta online Overleaf, el cual tiene gestión de cambios y versiones, además cada integrante debe colocar en la sección de historial de cambios la modificación que realizó en el documento con su respectiva fecha y descripción. La segunda herramienta es "Trello" la cual aparte de permitirnos estimar, nos brinda instrumentos para gestionar tareas, las cuales se listan y se van dando por terminadas.

6.6. Glosario

- **Adoptar un animal :** Acoger a un animal como mascota [1].
- **Animal:**Son aquellos seres vivos que poseen movimiento, cumplen el ciclo vital de nacer, crecer, reproducirse y morir, sienten, y se alimentan de sustancias orgánicas, presentes en el mundo exterior, que les proporcionan energía [2].
- **Mascota :** Es un término que procede del francés *mascotte* y que se utiliza para nombrar al animal de compañía. Estos animales, por lo tanto, acompañan a los seres humanos en su vida cotidiana, por lo que no son destinados al trabajo ni tampoco son sacrificados para que se conviertan en alimento[3] .
- **Animal de Compañía :**Especies animales que han pasado por el proceso de domesticación, y se asocian con el ser humano para bienestar común, sin utilizarse en aprovechamiento económico o alimenticio[4] .
- **Animal doméstico y domesticado :** animal que adquiere caracteres fisiológicos, morfológicos o de comportamientos, que luego se convierten en hereditarios; siendo resultado de la interacción deliberada y prolongada de dicho animal con el hombre[4] .

- **Plataforma web :** Es un conjunto de herramientas y tecnologías para construir sitios y aplicaciones web[5] .
- **LaTex :** Es un sistema de composición de textos que está orientado especialmente a la creación de documentos científicos que contengan formulas matemáticas, cuadros y tablas. Además, también se pueden crear otros tipos de documentos, que pueden ser desde cartas sencillas hasta libros completo[6].
- **Framework :** Es el entorno pensado para hacer más sencilla la programación de cualquier aplicación o herramienta actual [7].

7. Contexto del proyecto

7.1. Modelo de ciclo de vida

En esta sección se presentan todas las metodologías consideradas, el porqué fueron consideradas, y la metodología a utilizar para nuestro proyecto. Cabe resaltar que solo se consideraron las metodologías ágiles, que al ser flexibles con los cambios, permiten una mejora continua y mejoran la experiencia del cliente.

7.1.1. SCRUM

Es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Es un proceso ideal para proyectos en entornos complejos, que requieren resultados rápidos; requisitos cambiantes o sin definir, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Para cumplir con lo anterior, en SCRUM se realizan entregas parciales y regulares del producto final, esto con el fin de beneficiar al cliente.[8]

Se compone de:

- **Roles:**
  - **Product Owner:** Interesados en el producto final.
  - **ScrumMaster:** Responsable del proceso de SCRUM.
  - **ScrumTeam:** Transforma las tareas del Spring Backlog en un incremento de la funcionalidad de software.
- **Sprint:**
  - Etapa en la que se desarrolla un incremento de la funcionalidad en máximo 30 días
- **Artefactos:**
  - **Product Backlog:** Listado con los requisitos del sistema, incorpora constantemente necesidades del sistema.

- **Sprint Backlog:** Lista de tareas extraídas del Product Backlog para ser desarrolladas.

- **Comunicación:** Se realizan distintas reuniones, daily scrum (reuniones al inicio del día contando el trabajo), reuniones para planificar los Sprint (Sprint Planning); la correspondiente retrospectiva(Sprint Retrospective) y el resumen del Sprint(Sprint Review).

7.1.2. Extreme programming (XP)

Metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo. Se define como adecuada para proyectos con requisitos especialmente imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.[9]

- **Características**
  - Metodología basada en prueba y error
  - Fundamentada en Valores y prácticas
  - Expresada en forma de 12 practicas, la novedad es juntarlas.

7.1.3. Kanban

Metodología de desarrollo de software centrada en la entrega 'just-in-time' de la funcionalidad y la gestión de 'Work In Progress'. Es un sistema de trabajo de capacidad-producción de forma individual, sólo cuando existe la capacidad de procesar una tarea esta se ejecutará, al igual que Scrum se emplean tarjetas para la creación de flujo visual de trabajo. Cada tarjeta es una historia o tarea que estará en el proceso de producción hasta su finalización, esto indica que el flujo ha sido liberado y se puede realizar una nueva tarea. Esto se hace para que el trabajo individual se realice de manera continua y rápida, ya que se evita la sobrecarga de tareas hacia los miembros del equipo.[10]

- **Reglas:**
  - **Ver progreso de un proceso:** Poder ver todo el proceso de desarrollo de cada tarea, usando gráficos o tableros que representen el avance porcentual de la tarea.
  - **Limitar el trabajo en curso:** Se debe limitar el número de tareas por proceso o ciclo.
  - **Optimización del trabajo:** Limitar el número de tareas por proceso, para dar un máximo número de tareas por integrante, evitando su sobrecarga.

7.1.4. Conclusión de los modelos de ciclo de vida

Para poder escoger el modelo de ciclo de vida que mejor se acopla para el proyecto, se establecieron los siguientes criterios:

- **Definir funcionalidades del sistema:** El método debe permitir visualizar los requisitos del sistema y su funcionalidad gráficamente, siendo flexible en el uso de herramientas para la facilidad de uso.
- **Reuniones:** Las reuniones son obligatorias con estricto horario, sin restricción de la postura.
- **Control de proceso:** Control claro y especifico de la división del proceso en fila, ejecutándose ó terminado.
- **Separación de tareas:** Herramientas suficientes para separar e identificar tareas atómicas.
- **Control de las entregas:** Debe permitir herramientas para la definición de fechas, tanto de entregas como de ejecución de tareas, y avance.
- **Tiempo:** Organización del cronograma para cada entrega, para terminar el proyecto a tiempo, evitando la sobrecarga por retardos en el proceso de ejecución, ya que el plazo de entrega es corto.
- **Retroalimentación:** Debe permitir momentos de retroalimentación flexibles en horario, además de reportes de bugs, para la mejora continua en el desarrollo del proyecto.

Partiendo de los anteriores criterios, se procede a comparar los modelos escogidos, dando un valor para cada modelo en cada criterio. Se da un valor entero de 1-5 donde 1 es ”no cumple con el criterio” y 5 es ”Cumple completamente con el criterio”.

Criterio	**XP**	**SCRUM**	**Kanban**
Funcionalidades del sistema	5	5	5
Reuniones	2	4	4
Control de proceso	3	3	3
Separación de tareas	5	5	5
Control de entregas	4	4	4
Tiempo	3	3	5
Retroalimentación	4	5	3
Total	26	29	29

Tabla 3: Comparación de metodologías

Conclusión: Partiendo de la tabla 3, se puede observar que SCRUM y Kanban tienen el mismo puntaje, pero dadas las condiciones del proyecto se ha escogido Kanban. esto porque entre sus ventajas están la flexibilidad, manejo de tiempo para terminar ”just-in-time”, la gestión de ’Work In Progress’, que lo hacen un modelo muy acorde para lo que planteamos en los criterios de comparación. Es una metodología que, a diferencia de SCRUM con los sprints, el trabajo es continuo hasta finalizarse. Mientras en los sprints hay unas tareas estipuladas para realizar, y si algún integrante del equipo termina, esta persona queda en un estado inactivo o espera (esperando que finalice el sprint), mientras que este tiempo podría ser aprovechado, como sucedería en Kanban. Además, se decidió añadir más reuniones de retroalimentación y progreso. Esto con el fin de tener un mayor

control en el proyecto, porque nuestro equipo es Junior y no tiene la suficiente experiencia para el desarrollo con poca retroalimentación.

7.2. Lenguajes y herramientas

Las herramientas a usar se dividirán en las siguientes categorías:

- **Planeación del proyecto:** Herramientas que ayuden con la creación de documentos, diagramas y manejo de fuentes.
- **Implementación del proyecto:** Herramientas que permitan implementar proyectos de desarrollo web y manejar las versiones del mismo.

Debido a las cualidades y restricciones del proyecto, las herramientas deberían cumplir lo mejor posible las siguientes especificaciones:

- Ser software libre o tener una licencia gratuita. Al ser un trabajo no remunerado, el grupo no puede invertir dinero en licencias.
- Tener una licencia que nos permita trabajar en nuestro proyecto. Independientemente del costo de la licencia, esta no debe tener conflictos con la licencia de nuestro proyecto: GNU GPL v3.
- Ser competitiva en el mercado. De manera que no se quede atrás frente a la competencia y aprender la tecnología sea práctico.
- Que su etapa de aprendizaje sea corta. Como se usarán tantas herramientas nuevas, no se puede dedicar demasiado tiempo a cada una. Esto implica que sean herramientas que el grupo ya conozca parcialmente o sean fácil de utilizar.
- Poder integrarla con el resto de herramientas en su categoría. Evitando invertir tiempo innecesario en unir el proyecto, en lugar de invertirlo en el propio desarrollo.
- Estar bien documentada y encontrar información al respecto sea sencillo. Lo que facilita el uso de las herramientas, reduce su tiempo de aprendizaje y ayuda a resolver problemas que se presenten sobre la marcha.

Basándonos en estas necesidades el grupo ha escogido las siguientes herramientas para cada categoría:

- **General:**
  - **Trello:** Una herramienta en la que gestionar las tareas, roles y el progreso general del proyecto. Esta herramienta es especialmente importante al participar en todos los aspectos del proyecto. Es una buena herramienta para el proyecto porque:
    - El grupo ya se empezó a familiarizar con ella, gracias a que posee una gran cantidad de guías y ejemplos, para ajustar los tableros al ciclo de vida que se decida usar[11].

    - Al tener un solo proyecto pequeño, su versión gratuita es suficiente para satisfacer las necesidades del grupo [12].
    - Al ser una herramienta de gestión de tiempo y recursos, no impone ninguna limitación sobre la licencia utilizada, pues no se involucra directamente con el proyecto, y como su función es apoyar la metodología utilizada, no aplica la necesidad de ser competitiva.

- **Planeación del proyecto:**

  - **Overleaf v2:** Editor en línea de LaTeX en el que un grupo de personas puede editar paralelamente un mismo documento. Es una buena herramienta para el proyecto porque:
    - El plan gratuito de Overleaf es suficiente para la realización del proyecto [13], la única limitación es el número de colaboradores que tiene esa versión. Esa limitación se puede eliminarse mediante el sistema de invitaciones a usuarios, por lo que tampoco supone un problema.
    - Al no estar involucrado directamente con el desarrollo del proyecto, no lo limita de ninguna forma.
    - LaTeX es un estándar de la industria por su facilidad de escribir fórmulas matemáticas y separar el texto del formato.
    - Al trabajar en documentos sencillos de LaTeX, no es muy complicado aprender a utilizarlo.
    - Overleaf tiene una gran cantidad de ejemplos y documentación para sí mismo, y para LaTeX [14].
  - **Zotero:** Herramienta para manejar librerías de referencias, permite al grupo compartir referencias, analizar duplicados y exportarlas a casi cualquier formato. Es una buena herramienta para el proyecto porque:
    - Aunque vende espacio de almacenamiento, las referencias no son muy pesadas y con el plan estándar es suficiente para el proyecto [15].
    - Al no estar involucrado directamente con el desarrollo del proyecto no lo limita de ninguna forma.
    - Tiene un plugin para navegador que permite obtener las referencias de una página o documento más rápido que muchos de sus competidores.
    - Para el uso que se le va a dar, basta con conocer los aspectos más básicos de la herramienta.
    - Puede importar a casi cualquier estilo de referencia, incluido BibTeX(sistema de referencia de LaTeX) [16].
    - Está bien documentado y se pueden encontrar fácilmente los elementos básicos de su funcionamiento [17].
    - Se conecta fácilmente con Overleaf v2.

- **Implementación del proyecto:**

  - **Angular:** Framework orientado al desarrollo de Front-end. Es una buena herramienta para el proyecto porque:
    - Es Open Source y no se cobra por su uso.

- Utiliza *The MIT License* [18] y por lo tanto puede ser utilizado para este proyecto.
- Es un framework bastante nuevo (2016), pero es bastante usado actualmente y es apreciado por gran parte de los desarrolladores [19].
- El grupo está familiarizado con HTML y Javascript, pero no con TypeScript. Es necesario aprender a usar el framework y TypeScript.
- Angular obedece a protocolos y estándares de comunicación entre front-end y back-end típicos en HTML y Javascript. Al ser herramientas populares (Angular y Spring), existen múltiples guías de la comunidad para su integración.
- Tiene una buena documentación y ejemplos [20].

- **Spring:** Framework para Java orientado al desarrollo de back-end. Es una buena herramienta para el proyecto porque:
  - Es Open Source y usarlo no tiene ningún costo. [21].
  - Spring utiliza Apache License 2.0 [22], que es compatible con GPL v3.0.
  - Es querido por la comunidad y está creciendo en la industria [19].
  - El grupo está familiarizado con Java, de manera que solamente se tiene que aprender a utilizar el framework.
  - Obedece a protocolos y estándares de comunicación entre front-end y back-end, por lo que no debería ser difícil de integrar. Al ser herramientas populares (Angular, Spring y PostgreSQL), existen múltiples guías de la comunidad para su integración.
  - Tiene buena documentación y ejemplos en su sitio web [23].
- **PostgreSQL 11.5:**Base de datos relacional basada en SQL. En ella se almacenarán todos los datos de los clientes, información proporcionada por ellos y sus transacciones. Es una buena herramienta para el proyecto porque:
  - Es una base de datos relacional Open Source [24], utilizarla no tiene ningún costo para el proyecto.
  - Utiliza la licencia PostgreSQL License [25], una licencia permisiva que permite utilizar la herramienta para proyectos de cualquier licencia.
  - Aunque no es la base de datos más utilizada, sí es una de las más utilizadas profesionalmente [19].
  - El grupo es competente en el uso de bases de datos relacionales, y el aprendizaje de la herramienta se espera que sea sencillo.
  - PostgreSQL obedece a protocolos y estándares de comunicación entre back-end y base de datos relacional, tales como JPA o JDBC. Al ser herramientas populares (Spring y PostgreSQL), existen múltiples guías de la comunidad para su integración.
  - Posee una buena documentación [26], con información suficiente para su uso en nuestro proyecto.
- **GitHub:** Sistema de control de versiones basado en **Git**. Permite guardar el proyecto en un repositorio en la nube, dentro del cual podemos mantener un control de que cambios se han hecho y quién los realizó[27]. Es una buena herramienta para el proyecto porque:

- Git es Open Source [28] y, gracias a la universidad, el grupo cuenta con una licencia de GitHub Pro [29].
- GitHub usa CC0 [30] y por lo tanto podemos utilizarlo para nuestro proyecto.
- Git es estándar de la industria [19] y GitHub la herramienta que usa Git más popular del mercado, además de estar en constante crecimiento [31].
- Posee una gran cantidad de ejemplos y documentación muy bien ilustrada [32].
- Es compatible con los lenguajes utilizados en el proyecto.
- Es una herramienta familiar para el grupo.

7.3. Plan de aceptación del producto

Todas las entregas pueden ser aceptadas o rechazadas dependiendo de los criterios específicos. Al hablar de acuerdos en esta sección, haremos referencia a acuerdos entre el cliente y el grupo de trabajo para ese criterio en específico.

Para todas las entregas de documentos se deben cumplir los requisitos cuando apliquen, expuestos en la tabla 4.

Criterio	Se acepta si:	Se rechaza si:
Alcance	Los temas tratados dentro del documento son exactamente los definidos para el documento a presentar.	Los temas tratados dentro del documento discrepan de los definidos en el documento a presentar.
Planes claros	Los planes de acción presentados, son suficientemente específicos en sus actividades para que el cliente sepa en que etapa del desarrollo se encuentra el proyecto.	Los planes de acción presentados, son demasiado desordenados en sus actividades como para que el cliente sepa en que etapa del desarrollo se encuentra el proyecto.
Fechas establecidas	Las actividades tienen fechas definidas, tanto para su inicio como su fin.	Las actividades no tienen fechas claras para su inicio o para su final.
Resultados objetivos	Los resultados esperados son específicos y medibles, siendo claros los criterios de validación.	Los resultados esperados no son específicos o medibles, haciendo imposible determinar criterios claros para su validación.
Ortografía y redacción	Los errores de ortografía y redacción no son suficientes para malinterpretar el texto.	Los errores de ortografía y redacción hacen posible la malinterpretación del texto.

Tabla 4: Criterios de aceptación de documentos

Para todas las entregas de software, se deben cumplir los requisitos que apliquen, expuestos en la tabla 5.

Criterio	Se acepta si:	Se rechaza si:
Proceso básico	Las funcionalidades del producto entregado son iguales a las funcionalidades acordadas.	Existen diferencias entre las funcionalidades del producto entregado y las previamente acordadas.
Velocidad de respuesta	El tiempo entre realizar una acción y obtener su resultado es inferior o igual al límite acordado.	El tiempo entre realizar una acción y obtener su resultado es mayor al límite acordado.
Número de clicks	El número de clicks necesarios para realizar una acción es menor o igual al límite acordado.	El número de clicks necesarios para realizar una acción es mayor al límite acordado.
Utilización de recursos	La aplicación consume menor o igual cantidad de recursos (Memoria, tiempo de procesamiento y banda ancha) que lo acordado.	La aplicación consume mayor cantidad de recursos (Memoria, tiempo de procesamiento y banda ancha) que lo acordado.
Pantallas acordadas	Las pantallas implementadas tienen los mismos componentes que las pantallas acordadas.	Las pantallas implementadas tienen componentes diferentes a las pantallas acordadas.
Fidelidad a estética	Los colores usados son iguales a los acordados.	Los colores usados son distintos a los acordados.

Tabla 5: Criterios de aceptación del producto

7.4. Organización del proyecto y comunicación

En esta sección se evidenciará la organización del equipo, sus responsabilidades, y se darán a conocer las entidades externas al proyecto.

7.4.1. Entidades externas o Stakeholders:

Podemos ver las entidades externas en la tabla 6.

Nombre	Descripción	Responsabilidad	Datos de contacto
Carlos Andrés Parra Acevedo	Profesor encargado de la asignatura Ingeniería de software en la Pontificia Universidad Javeriana Bogotá	Guiar al equipo en la planeación, dando a conocer los fundamentos teóricos de la ingeniería de software. Además de dar criticas constructivas, resolver dudas e inquietudes, y dar retroalimentación para el mejoramiento continuo del equipo y el proyecto.	Correo: ca.parra@javeriana.edu.co
Personas	Todas las personas que están interesadas en adoptar un animal, interesados en dar un animal en adopción o productos para sus mascotas. E interesados en el bienestar de su mascota.	Dar un buen uso a la plataforma, publicando información veraz y completa.	Datos ingresados por el usuario: Correo electrónico, número telefónico
Refugios	Entidad que publica los animales que están en adopción.	Actualizar la pagina con los animales que se encuentran en adopción, han sido adoptados. Además de dar retroalimentación en la aplicación.	Datos ingresados por el usuario: Correo electrónico, número telefónico o dirección
Tiendas de animales	Entidad que publica sus tiendas (sucursales)	Dar un buen uso a la plataforma, publicando información veraz y completa. Actualizar la información de la plataforma constantemente.	Datos ingresados por el usuario: Correo electrónico, número telefónico o dirección
Veterinarias	Entidad que publica sus servicios y datos de contacto.	Dar un buen uso a la plataforma, publicando información veraz y completa. Actualizar la información de la plataforma constantemente.	Datos ingresados por el usuario: Correo electrónico o número telefónico
Angular	Framework para programación web [33]	Responsable de procesar la página del lado del cliente, y la relación vista-modelo	https://angular.io/
PostgreSQL	Proveedor de servidor y base de datos [34]	Responsable del alojamiento de los datos, la relación servidor-cliente, y la integridad de los datos	https://www.postgresql.org/
Java Spring	Framework de desarrollo back-end [35]	Responsable de procesar los datos ingresados por el usuario (vista-modelo) y la comunicación con el servidor (modelo-servidor)	https://spring.io/

Tabla 6: Roles dentro del proyecto

7.4.2. Organigrama y descripción de roles

- **Organigrama**

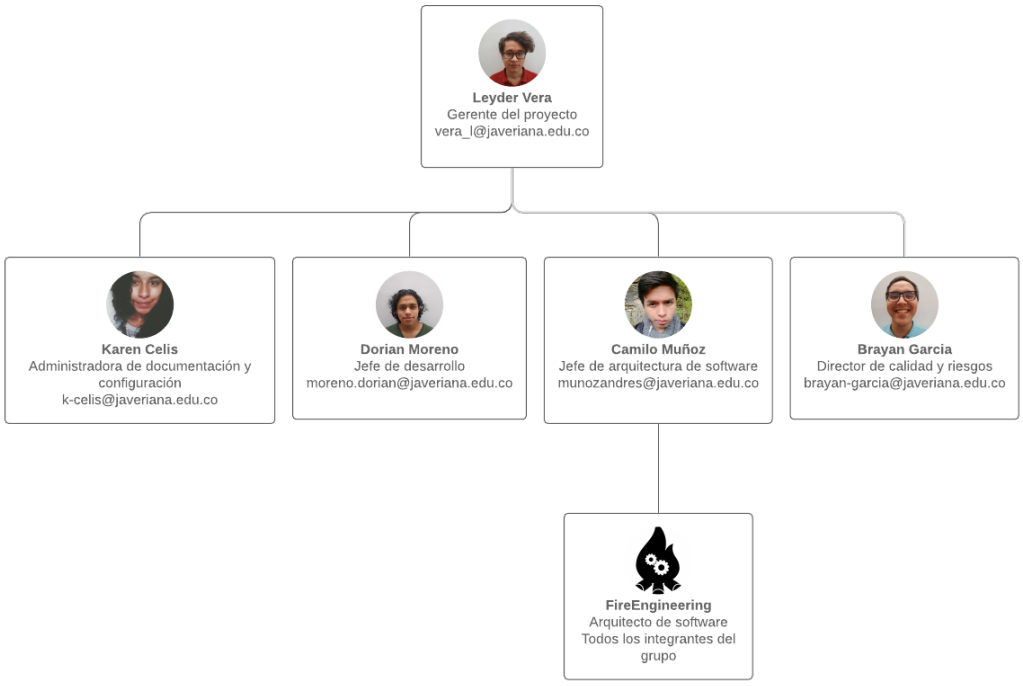


Figura 1: Organigrama

- **Descripción de roles**

Rol	Descripción	Responsabilidad
Gerente del proyecto	Persona encargada de guiar al equipo de trabajo hacia el cumplimiento de los objetivos. [36]	Organizar la distribución del trabajo; el cronograma de cada mes para el cumplimiento de la entrega a tiempo, supervisar el trabajo del equipo, y convocar las reuniones.
Jefe de desarrollo	Persona que capacita a los otros desarrolladores, y ayuda a resolver los problemas que surjan en el desarrollo. [37]	Revisa que el código haya sido escrito usando buenas prácticas de programación (comprueba la calidad del código). Capacita a los demás desarrolladores en las herramientas a utilizar.
Director de calidad y riesgos	Supervisa los estándares de calidad para clasificar el proyecto como satisfactorio. Gestiona los riesgos que se puedan producir a lo largo de la ejecución, se hace cargo de las auditorías internas, y seguridad de la información. [38]	Identifica los riesgos a lo largo del proyecto. Establece las políticas de calidad del software. Al igual que el jefe de desarrollo revisa la calidad del código (practicas de programación, optimización, espacio, complejidad, etc).
Arquitecto de software	Persona encargada de traducir los requerimientos del cliente para dar una adecuada solución de software. De esa traducción depende el plan de desarrollo del proyecto. [37]	Seguimiento a la estructura planteada para mantener la calidad del software. Crea el diseño técnico del software.
Administrador de documentación y configuración	Establece las normas de documentación, revisa el código continuamente (documentación del código). Revisa la ortografía, redacción y coherencia de los documentos.	Comunica las normas de documentación. Realiza las correcciones a los documentos. Unifica el trabajo realizado por los distintos integrantes. Organiza el contenido de cada entrega.

Tabla 7: Descripción de roles

8. Administración del proyecto

8.1. Métodos y herramientas de estimación

8.1.1. Metodología:

Para el proyecto, en la estimación hay dos grandes factores que se tuvieron en cuenta. El esfuerzo humano para desarrollar cada actividad y costos del proyecto. Para cada uno de ellos explicaremos cómo se hizo y los resultados estarán en el anexo *Presupuesto Y Estimaciones*. Se siguió la metodología ágil Kanban para dicha estimación, ya que decidimos crear historias para dividir de una mejor manera las actividades que se debían realizar, para completar el proyecto. Esto para que cada integrante desarrolle la misma cantidad de trabajo que los demás o para que no se sobrecargue.

La métrica que usamos para medir la estimación del proyecto fue los puntos de historia [39], efectuando el proceso con las tareas de cada historia previamente identificada y dicha estimación se realizó en una reunión especial con este objetivo. Posteriormente, se definió que cada punto de historia equivaldría a dos horas de esfuerzo humano. Además, decidimos hacer la estimación con los puntos de casos de uso [40] para comparar ambos resultados y probar cuál sería el más acertado.

8.1.2. Herramientas:

Para estimar el esfuerzo humano se usó la aplicación móvil Planning Poker [41], dando a cada actividad que se debía realizar en el proyecto una cantidad de puntos, estimados por cada uno de los participantes que votó. Estos puntos se daban pensando en la cantidad de trabajo o la dificultad de la actividad en cuestión. Esto dio un resultado que nos indica que la duración del proyecto tomará al rededor de 2 meses con el método de puntos de historia, y para el método de UCP calculamos la fórmula y nos dio como resultado de 5.3 meses. Para más información de estas estimaciones ver anexo *Presupuesto Y Estimaciones*.

Para estimar el costo en dinero del proyecto se hizo un presupuesto de las herramientas, incluyendo software y hardware que posee cada uno de los integrantes, y que se va a utilizar para el desarrollo del mismo. En gastos se concluyó que se va a invertir 0 (cero) $ COP, puesto que, el software que se va a usar es gratuito y no se va a comprar ningún tipo de hardware.

Sin embargo, las horas de esfuerzo humano también tienen un costo. Se investigó cuánto era el salario promedio de un ingeniero de sistemas y se hizo una conversión a horas para sacar la estimación tomando en cuenta los resultados de los puntos de historia de todo el proyecto. Para más detalles de las estimaciones ver anexo *Presupuesto Y Estimaciones*.

8.2. Inicio del proyecto

Para cada una de las herramientas debe darse el siguiente proceso inicial:

1. Cada persona debe conocer el objetivo de cada herramienta y su función general dentro del proyecto. Esto es un esfuerzo grupal de los arquitectos.
2. Todos los miembros del grupo deben instalar las aplicaciones que lo requieran y crear las cuentas pertinentes. Debe ser realizado por todos los arquitectos con asistencia del administrador de configuración.
3. Empezar con tareas ligeras para los integrantes que no estén familiarizados con la herramienta, de manera que puedan familiarizarse poco a poco. El gerente debe asignar las tareas y miembros con más experiencia en las herramientas para que ayuden en los pasos iniciales, con el fin de agilizar el proceso y promover las buenas prácticas.

Hay herramientas con las que el grupo completo ya es familiar y ya utiliza en el proyecto. Los pasos necesarios a dar con las otras herramientas son los siguientes:

1. **Zotero:**
   - No lo saben utilizar todas las personas, pero hay miembros del grupo que pueden guiarlas.
   - Se puede aprender a lo largo de la realización de los documentos.
2. **Angular:**
   - Ningún miembro ha utilizado Angular o TypeScript, pero el grupo tiene experiencia con HTML y JavaScript.
   - Se puede aprender a medida que avanza el desarrollo del proyecto.
3. **Spring:**
   - Ningún miembro ha utilizado el framework antes, pero todo el grupo tiene experiencia con Java.
   - Se puede aprender desarrollando los componentes del caso de uso más difícil.
4. **PostgreSQL:**
   - Ningún miembro ha usado PostgreSQL antes, pero el grupo tiene experiencia con SQL y bases de datos relacionales.
   - Se puede aprender desarrollando los componentes del caso de uso más difícil.

Para mantener el proyecto en funcionamiento se deben realizar, periódicamente, las tareas en la tabla 8.

Tarea	Responsable(s)	Frecuencia
Control de calidad	Director de calidad asistido por demás miembros	Una vez a la semana y una vez antes de entregarlo
Monitoreo de avances (General)	Gerente del proyecto	Dos veces a la semana
Monitoreo de avances en el plan de desarrollo	Jefe de desarrollo	Una vez a la semana
Revisión de documentación	Administrador de documentación asistido por demás miembros	Una vez a la semana
Mantenimiento de los repositorios	Administrador de configuración asistido por demás miembros	Una vez cada dos semanas
Asignación de tareas	Gerente del proyecto	Una vez cada dos semanas. Puede realizarse fuera de horario si así lo considera.

Tabla 8: Tareas periódicas para el correcto funcionamiento del proyecto.

8.3. Planes de trabajo del proyecto

El desglose de los entregables se puede evidenciar en el anexo *WBS del proyecto*.

Las fechas de entrega al cliente son las siguientes:

1. **Inicio del proyecto:** El **30 de Julio** el grupo comienza el desarrollo del proyecto con el SPMP.
2. **Primera revisión:** Se debe mostrar el SPMP con las reglas definidas entre el cliente y la empresa. Sucede el **22 de Agosto** y permite una última corrección antes de la entrega del documento.
3. **Primera entrega:** Se entrega la versión final del SPMP. Sucede el **27 de Agosto**. Después de esta entrega se empiezan a levantar los requisitos para realizar el SRS.
4. **Fin del diseño inicial:** Desde la primera entrega hasta el **10 de Septiembre** se realiza la versión inicial del diseño y los requerimientos. Con la propuesta inicial del diseño, el grupo empieza la implementación del caso de uso más importante en el proyecto.
5. **Segunda revisión:** Se muestra una primera versión del SRS y el caso de uso más difícil al cliente el **15 de Octubre**. Después de la revisión, se realizarán cambios en el documento y la implementación para obtener la versión final.
6. **Segunda entrega:** Se entrega la versión final del SRS y el caso de uso más importante del proyecto el día **17 de Octubre**. Una vez completada la entrega, se continua con el desarrollo de los demás casos de uso y se empieza el SDD.
7. **Tercera revisión:** Se presenta una versión tentativa del SDD y los casos de uso ya implementados. Esta revisión tiene lugar el día **5 de Noviembre** y, después de ella, el grupo realizará los cambios pertinentes para crear la versión final del entregable.
8. **Entrega final:** El día **12 de Noviembre** se entregan todos los documentos corregidos hasta la fecha: SPMP, SRS y SDD. También se entregará un prototipo de la aplicación funcional.

El diagrama de Gantt con estas fechas se encuentra en el anexo *Gantt del proyecto*

El flujo de caja esperado en el tiempo de vida del proyecto se puede evidenciar en el anexo *Flujo de caja libre*. Este flujo de caja se hizo para un tiempo promedio entre las dos estimaciones obtenidas.

9. Monitoreo y control del proyecto

9.1. Administración de requisitos

Los requisitos son esenciales para concretar el objetivo y las funcionalidades propias del producto a realizar. Estos deben ser claros, medibles, sostenibles, posibles, etc.

Con el fin de evitar obstáculos a la hora de realizar el proyecto, es necesario tener claro cómo manejar aquellos nuevos requisitos. A continuación, se procederá a explicar las fases a realizar, desde el momento en que el cliente decida agregar un nuevo requisito hasta el momento de entrega del avance.

Cabe recalcar que este procedimiento se realizará únicamente para los nuevos requisitos y/o aquellos que deban ser cambiados. Se considera que aquellos requisitos definidos al iniciar el proyecto ya fueron estudiados con anterioridad para no generar contratiempos.

Se puede ver en el anexo *Administracion de requisitos* el diagrama BPMN del proceso explicado a continuación.

9.1.1. Planteamiento

Esta fase inicia desde el momento en que el cliente desea cambiar algún requisito existente o agregar uno nuevo. Primeramente, el cliente se acerca al director del proyecto por medio de una reunión (La cual puede ser presencial o virtual), en esta reunión se espera que el cliente deje claro cuál cambio/adición desea hacer, el por que es necesario esto y como puede llegar a afectar a los demás requisitos.

Una vez el director tenga clara la acción a realizar, ha de convocar una reunión donde explicará detalladamente la petición del cliente. Todo el grupo discutirá si es viable o no para el proyecto. De ser posible, se procederá a la siguiente fase, de lo contrario, se archivará la petición y se le comentará la decisión al cliente por medio del director. (De realizarse un cambio en la petición rechazada, se tomará como una nueva petición por lo que seguirá el mismo protocolo).

9.1.2. Análisis y adecuación

Después de ser aprobada, se realizará un borrador del funcionamiento del requisito con la ayuda de todo el equipo. Al hacer esto se evaluarán las implicaciones que conlleva realizarlo, viendo su dificultad, los recursos disponibles y el tiempo que podría tomar.

El director recopilará lo que se logre definir en la reunión para así comunicarle al cliente lo analizado. De existir alguna inconsistencia, se discutirá con el cliente con el fin de llegar a un acuerdo sobre la petición, o en el remoto caso, reformular el requisito entre director y cliente, esto para volver a postularlo con el equipo del proyecto. Por el contrario, si el cliente no tiene algún inconveniente se procederá a continuar con la siguiente fase.

9.1.3. Implementación

El director convocará una reunión con el equipo encargado del proyecto, donde expondrá el plan de acción a seguir y se discutirá a profundidad cómo realizar lo solicitado. Al final de la reunión se tendrán las nuevas tareas a realizar o cuáles son las que se deben cambiar. Seguido de esto se procederá a ejecutarlas.

9.1.4. Entrega y retroalimentación

En las reuniones que se tengan programadas se mostrará el avance o culminación del requisito. Se debe mostrar ante el director del proyecto lo que se realizó, de tener que hacerse algún cambio, la persona encargada lo realizará. Por el contrario, si se tiene el requisito completo, se documentará el mismo y, por parte del director, notificará al cliente del resultado obtenido.

9.2. Monitoreo y control del progreso.

Al tener definidas las tareas debemos tener una manera para verificar el progreso de cada una de estas, por lo que es necesario buscar métricas con las cuales se pueda dar un tiempo estimado para realizar las tareas del proyecto. Con estas medidas, no solo se tendrá una percepción de cómo va el proyecto actual, sino que también sirven para referencia en el caso de realizar un proyecto similar.

En esta sección del documento se explicarán las métricas a utilizar para controlar y medir el progreso del proyecto.

9.2.1. Unidades para medir el progreso.

Como se mencionó anteriormente, la metodología a utilizar es Kanban. Esto porque permite la utilización de dos unidades de medida para evaluar el progreso del proyecto, las cuales serán explicadas a continuación:

Tareas completadas: Será nuestro principal método de medida para el progreso del proyecto, esto, ya que permite evidenciar el avance de la entrega basándose en la cantidad de tareas que se ha realizado. Por medio de esta se podrá tener una documentación de cuanto se ha avanzado y, de igual manera, cuanto falta para completar el proyecto. Se calculará con la siguiente ecuación:

$$AvanceDeLasTareas = \frac{TareasTerminadas}{TareasTotales} * 100 \tag{1}$$

Este resultado se expresará en términos porcentuales, siendo 100 % el total de las tareas realizadas.

Valor puntos: Debido a que cada una de las tareas realizadas tiene una cantidad de puntos, los cuales representan horas, podemos ver la cantidad de horas que se deberán utilizar para cada semana de desarrollo. Se procederá a comparar el tiempo que se estimo contra el tiempo real que a la persona encargada le tomó.

Con esto se podrá conseguir un aproximado de cuánto equivaldría cada uno de los puntos, se puede conseguir cada uno de estos puntos por regla de 3.

$$CantidadHorasPorPunto = \frac{Puntos}{HorasUtilizadas} \tag{2}$$

Gracias a esta medida, se puede tener un aproximado de la fecha de finalización del proyecto. Se puede evidenciar gracias a la gráfica BurnDown Chart de la metodología Scrum (Esta se calcula promediando el número de puntos completados en los sprints anteriores).

Lead time: Medida obtenida de la metodología Kanban, tiene el propósito de registrar el tiempo comprendido entre el momento de creación de una tarea y el momento de su entrega. [42] Por medio de esta medida se podrá observar el tiempo que le toma al equipo ejecutar una tarea, además de prevenir si acaso se debe agilizar el proceso de desarrollo de las mismas.

9.2.2. Actividades para reportar el progreso

Para los proyectos de gran magnitud es importante tener retroalimentación constantemente, por lo que todos los integrantes del equipo deben tener conocimiento de como van las funcionalidades y el proyecto en sí. Los acuerdos para el reporte de progreso son:

Reuniones semanales: Se realizarán reuniones semanales, en donde cada uno de los integrantes hablará sobre el avance/culminación de las tareas que le hayan sido asignadas, de igual manera podrá hablar de los inconvenientes presentados a la hora de implementar sus tareas. Se tendrá en cuenta el porcentaje y la velocidad de avance de las tareas asignadas, de manera que se pueda ver si la persona encargada necesita ayuda o no está haciendo su trabajo.

Para dar por completada una tarea, esta debe ser aprobada por el jefe de desarrollo y el administrador de documentación, de manera que ya este probada y no tengan errores a la hora de integrarse con el proyecto.

Estas reuniones se realizarán de manera presencial y virtual. Teniendo la reunión presencial como horario los viernes de 6 a 8 pm. En cuanto a las reuniones virtuales, se realizará al menos una a la semana donde se reporten los avances o dudas que se tengan con el proyecto.

Revisiones a los tableros de Trello: Con el fin de estar todos enterados del estado del proyecto, se manejará la herramienta Trello mencionada anteriormente, con la cual se pondrán clasificar las historias como: tareas aún no implementadas, en proceso y finalizadas.

Al terminar o iniciar una nueva tarea se deberá ordenar las tarjetas en sus columnas correspondientes, de esta manera se podrá evidenciar el estado del proyecto. Cada movimiento de tarjetas que se llegue a realizar debe ser comunicado al grupo; de igual manera, de estar seguro que una tarea se ha completado, inmediatamente se debe mover su tarjeta a su respectiva columna (finalizado), esto para evitar confusiones en el proceso de esta.

9.2.3. Acciones correctivas

La puntualidad a la hora de crear un proyecto resulta importante, no solo por el hecho de tener responsabilidad, sino que esto conlleva costos adicionales para la empresa. Si la realización de tareas se retrasa, generaría problemas con el tiempo estipulado de entrega del proyecto.

Con el fin de evitar lo anterior, se estipuló varias acciones para penalizar a aquellas personas que incurran en estos actos. Las estrategias que se aplicarán son:

- Si un miembro no es responsable con las actividades asignadas, se procederá a revisar el caso en una reunión del grupo completo. Si no se presenta una excusa válida, el paso siguiente es imponerle un castigo ejemplar considerado por el grupo.

- Si el acto cometido es un fallo en la planeación, se procede a replantear el cronograma, lo que conlleva a ajustar el trabajo para cada miembro del equipo.

Se espera que en cualquiera de las situaciones los miembros del grupo acepten las acciones anteriormente mencionadas.

9.3. Cierre del proyecto

Para cada entrega del proyecto, se convocará una reunión donde se discutirán las acciones a realizar como consecuencia de las observaciones obtenidas por parte del cliente. De igual manera, se realizará una inspección final de calidad a todo el proyecto realizado, siguiendo los parámetros establecidos en la sección de control de calidad.

Para la elaboración del reporte gerencial y análisis post-mortem, se tendrá en cuenta toda la información recopilada durante cada una de las reuniones, el desempeño de cada integrante, el trabajo realizado, la participación, los recursos e insumos utilizados por cada uno. Además, mediante la opinión de cada uno de los integrantes del equipo se dará a conocer una retrospectiva general de cómo fue planeado y ejecutado el trabajo, esto con el fin de identificar fortalezas, amenazas, áreas de oportunidad y acciones de mejora, con esto se espera establecer nuevos lineamientos para tener en cuenta en el futuro del proyecto o en nuevos proyectos para la empresa.

10. Entrega del producto

La entrega final de PetFriend como prototipo de plataforma web, estará segmentada en cuatro pilares esenciales. La documentación formal del proyecto, el software funcional, la sustentación final del mismo y la retroalimentación del proyecto. Cada uno de estos serán entregados conforme a las fechas establecidas.

En cuanto a la documentación, se encuentra el documento (SPMP); la Especificación de Requisitos de Software (SRS); y la Descripción del Diseño del Software (SDD). Además de estos, es importante resaltar y suministrar aquellos documentos que soportan las funcionalidades del prototipo a presentar, así como los que complementan y fundamentan cada entregable dado.

Para el segundo pilar se agrupan aquellos componentes que dan forma y solidifican al software funcional, en este caso para PetFriend. Consecuentemente siguen todas aquellas actividades que conformarán la sustentación del proyecto entre el equipo de trabajo y el cliente; para así finalizar con la retroalimentación en la cual se atenderán aspectos de mejora de PetFriend.

11. Procesos de soporte

11.1. Ambiente de trabajo

Hoy en día es común que los proyectos de desarrollo de software impliquen trabajo en grupo, pues se necesita de una gran gama de conocimientos e ideas para desarrollar un proyecto de calidad, al mismo tiempo, esto permite simplificar tareas y agilizar procesos.

El establecimiento de reglas y mecanismos de trabajo aseguran el cumplimiento y control de tareas conjuntamente con una buena convivencia en el grupo.

- **Seguimiento de tareas mediante la plataforma Trello:** La herramienta Trello da la facilidad de hacer un seguimiento mediante listas de tarjetas, en ellas se plantea la metodología ágil Kanban la cual permite mover diferentes tarjetas que contienen historias y tareas a las diferentes listas.

Historias	Para hacer	En progreso	Para verificar	Completo

Tabla 9: Cabezotes del tablero de Kanban y SCRUM

- **Reuniones físicas al menos una vez por semana:** Cada semana se realizará una reunión de máximo tres horas donde se dialogarán temas acordes a las tareas y actividades propuestas, también se discutirán temas como: los estados, tiempos, dificultades, posibles soluciones de las tareas. En adición, si se llegase a acabar la reunión antes del tiempo estipulado, se utilizaría el tiempo restante para escribir código o hacer revisiones a tareas cuyo proceso ya fue terminado.El plazo máximo de llegada es de 15 minutos, después de esto se contará como una falta en donde el infractor deberá disculparse con los demás integrantes del grupo, cada excusa para inasistencia a las reuniones virtuales deberá ser revisada por el grupo y aprobada por todos, preferiblemente las excusas deben ser citas médicas, enfermedad, razones de fuerza mayor, eventos respaldados por entidades formales como la universidad.
- **Reuniones virtuales:** Cada semana se realizará una reunión de máximo tres horas por semana donde se hablarán temas acordes a las tareas y actividades propuestas, a diferencia de las reuniones físicas están se realizan por medio de internet mediante la plataforma Discord. Esta herramienta permite comunicar un grupo grande de personas con la facilidad de separarlos por canales, es decir, permite separar el grupo en diferentes salas. El plazo máximo de llegada es de 15 minutos, después de esto se contará como una falta en donde el infractor deberá disculparse con los demás integrantes del grupo, cada excusa para inasistencia a las reuniones virtuales deberá ser revisada por el grupo y aprobada por todos, preferiblemente las excusas deben ser citas médicas, enfermedad, razones de fuerza mayor, eventos respaldados por entidades formales como la universidad.
- **Historial de cambios:** Para hacer seguimiento y regulación de los documentos, estos traen la sección de historial de cambios donde se mostrará la fecha, el autor y la descripción del cambio realizado. Es deber de todos los integrantes actualizar esta tabla pues demuestra el trabajo realizado en el documento además de ser parte de este.

11.2. Análisis y administración de riesgos

El equipo de trabajo dispone de un tiempo limitado y definido para la entrega del proyecto, por lo que un retraso en este podría acarrear costos para la empresa al no cumplir con un contrato. Con el fin de mitigar los inconvenientes que puedan surgir al avanzar en el proyecto, el equipo de trabajo definió un protocolo de administración para el control de los riesgos que puedan llegar a surgir. Por medio de esta administración se espera prevenir los riesgos identificados en la fase inicial del proyecto.

Para ejecutar el plan de administración de riesgos se tiene en cuenta dos factores importantes a la hora de analizar un riesgo: La probabilidad que el riesgo ocurra y el impacto del riesgo sobre el proyecto.

Cabe aclarar que la probabilidad de un riesgo se midió en un porcentaje de 0 % a 100 %. Por otro lado, el impacto de un riesgo se midió en una escala de 1 a 5 como se puede notar en la tabla 10 en la página 29.

Impacto	Severidad	Descripción
1	Débil	Se puede ignorar
2	Limitado	Afecta muy poco a la ejecución del proyecto. Se puede gestionar rápidamente
3	Notable	Afecta el proyecto, pero tiene una mitigación sencilla
4	Fuerte	Afecta el proyecto, pero el plan de mitigación puede reducir el impacto
5	Muy fuerte	Afecta el proyecto,y no hay plan de mitigación concreto con el cual se pueda reducir el impacto de manera efectiva

Tabla 10: Escala de riesgos

11.2.1. Matriz DOFA

La matriz DOFA es la técnica de planificación empresarial aplicada dentro del ámbito personal. Donde se analizan varios aspectos importantes de la organización y su entorno. [43] Por medio de esta matriz se tuvo un reconocimiento de los riesgos en la fase inicial del proyecto, para posteriormente planear su prevención y mitigación.

Cabe resaltar que el riesgo no siempre supone consecuencias negativas en el proyecto, por lo que analizar tanto riesgos positivos como negativos resulta clave para este.

Con base en el proyecto se realizó la matriz DOFA, que se puede observar en la tabla 11 que se encuentra en la página 29.

Debilidades	Oportunidades
- Tiempo corto para resolución del proyecto - Poca experiencia en el uso de algunos lenguajes - El equipo tiene otras actividades que pueden llegar a ocupar el tiempo del proyecto	- Aumento de personas que quisieran adoptar una mascota - Refugios y personas quieren publicitar los animales - Personas buscan una alternativa para ver los animales
Fortalezas	**Amenazas**
- Buena comunicación tanto con el equipo de trabajo como con el cliente - El proyecto se realiza en un entorno de aprendizaje: es posible equivocarse - Equipo de trabajo especializado en diversas áreas	- Otros sistemas con funcionalidades similares

Tabla 11: Matriz DOFA

En la tabla 12 se pueden evidenciar los riesgos identificados para el proyecto.

Riesgo	Impacto	Ocurrencia	Prevención	Planes de mitigación
Atraso en las entregas	5	5 %	Tener un control estricto en el cronograma de entregas.	
Mala planificación del tiempo	4	35 %	Elaborar correctamente la planificación de tiempo	Modificar la carga de trabajo para cada uno de los integrantes del equipo
Mala distribución de las tareas	4	35 %	En cada reunión notificar la carga de trabajo asignada	Redistribuir las tareas, intentando dar un trabajo equilibrado para cada uno de los miembros del equipo
Desarrollo incorrecto de funcionalidades	4	20 %	Seguir el diseño planteado y estar informando de llegar a presentarse algún problema	Distribuir el trabajo con otro miembro del equipo
Cambio en las fechas de entregas	4	10 %	Estar en constante comunicación con el cliente	Aumentar el trabajo por cada uno de los integrantes.
Cambio de los requisitos	4	5 %	Identificar plenamente los requisitos al iniciar el proyecto	Verificar costo de agregar/cambiar el requisito (ver administración de requisitos)
Fallo en los computadores de trabajo	4	5 %	Subir todo trabajo a la nube.	Utilizar los computadores de la universidad
Omisión de un requisito	4	5 %	Realizar lista de verificación de cada uno de los requisitos	Distribuir el requisito entre los miembros del equipo
Problemas de comunicación	3	25 %	Definir todo en las reuniones que se tengan	
Personal inexperto	3	15 %	Estudiar las herramientas que se van a utilizar para el proyecto	
Incompatibilidad del código	3	15 %	Definir entradas y salidas de los códigos a realizar	En las reuniones se explicará el código que realizo, y ayudará a hacer compatible el código
Manejo de una versión antigua	3	10 %	De realizarse un cambio importante notificar en que versión queda el proyecto	Ver la compatibilidad entre versiones, para ver como debe cambiar/introducir su funcionalidad
Herramienta de trabajo descontinuada	3	5 %	Llevar un control sobre las versiones de las herramientas utilizadas	
Deserción de un integrante	3	5 %	Ser colaboradores entre los integrantes del grupo	Distribución del trabajo del integrante entre todo el equipo
Discusiones entre los miembros del grupo	2	10 %	Establecer una relación de respeto entre cada uno de los miembros del equipo	Charlas grupales que ayuden a resolver conflictos
Calamidad doméstica de un miembro del equipo	2	5 %		
Pérdida de documentación	1	10 %	Usar adecuadamente el repositorio	
Repetición de código	1	5 %	Mirar detalladamente las actividades asignadas para uno de los integrantes	

Tabla 12: Riesgos

11.3. Administración de configuración y documentación

A continuación se presentarán los ítems de configuración identificados por el grupo, que son de vital importancia para tener una trazabilidad. También servirán para tener una idea del porcentaje del producto completado.

Los ítems son los siguientes:

- **Plan de administración del proyecto o SPMP** (por sus siglas en inglés), es el plan que define las actividades, funciones y gestión del proyecto que son necesarios para cumplir con los requisitos de un producto[44].
- **Descripción del diseño del software o SDD** (por sus siglas en inglés), especifica los componentes del software a desarrollar[45].
- **Listado de requisitos para el producto.** También llamado Software Requirements Specification (SRS), es la especificación del comportamiento del sistema y los detalles que debe tener para cumplir las expectativas del cliente.
- **Diagrama de casos de uso**, es un diagrama de comportamiento que describe de forma visual algunos requisitos del sistema o casos de uso[46].
- **Diagrama de clases**, es un diagrama que modela las clases de un sistema e información como atributos, funciones y relaciones[47].
- **Diagrama entidad-relación**, se utiliza para modelar entidades relevantes de una base de datos y sus interrelaciones[48].
- **Código fuente**, es el documento de texto con las indicaciones para ejecutar el sistema o programa[49].

- **Manual del usuario**, es un documento técnico que sirve para dar asistencia a las personas que usan el sistema[50].

La etapa de desarrollo de cada uno de estos ítems se muestra en la tabla 13.

ÍTEM	**Etapa desarrollo**
Plan de administración del proyecto o SPMP	Primera etapa
Listado de requisitos para el producto o SRS	Segunda etapa
Diagrama de casos de uso	Segunda etapa
Diagrama de clases	Segunda etapa
Diagrama entidad-relación	Segunda etapa
Descripción del diseño del software o SDD	Tercera etapa
Código fuente	Segunda etapa hasta final de tercera etapa
Manual del usuario	Cierre del proyecto

Tabla 13: Documentos de configuración.

Los cambios que se deseen efectuar a cualquiera de los artefactos o ítems anteriormente nombrados, deberán pasar por un proceso para verificar si es estrictamente necesario.

El proceso es el siguiente:

- Primero se debe notificar al encargado o director del proyecto, para que posteriormente se pase a la siguiente fase.
- Luego, el encargado o director del proyecto revisa la necesidad del cambio que se desea efectuar y los impactos que tiene en el proyecto.
- Si el encargado lo rechaza, termina el ciclo del cambio.
- Si el encargado da el visto bueno, se procede a plantear dicho cambio al equipo completo en una reunión con ese fin.
- En la reunión se define por medio de una votación si se efectúa el cambio.

Este proceso está representado en la figura 2 en la pagina 32.

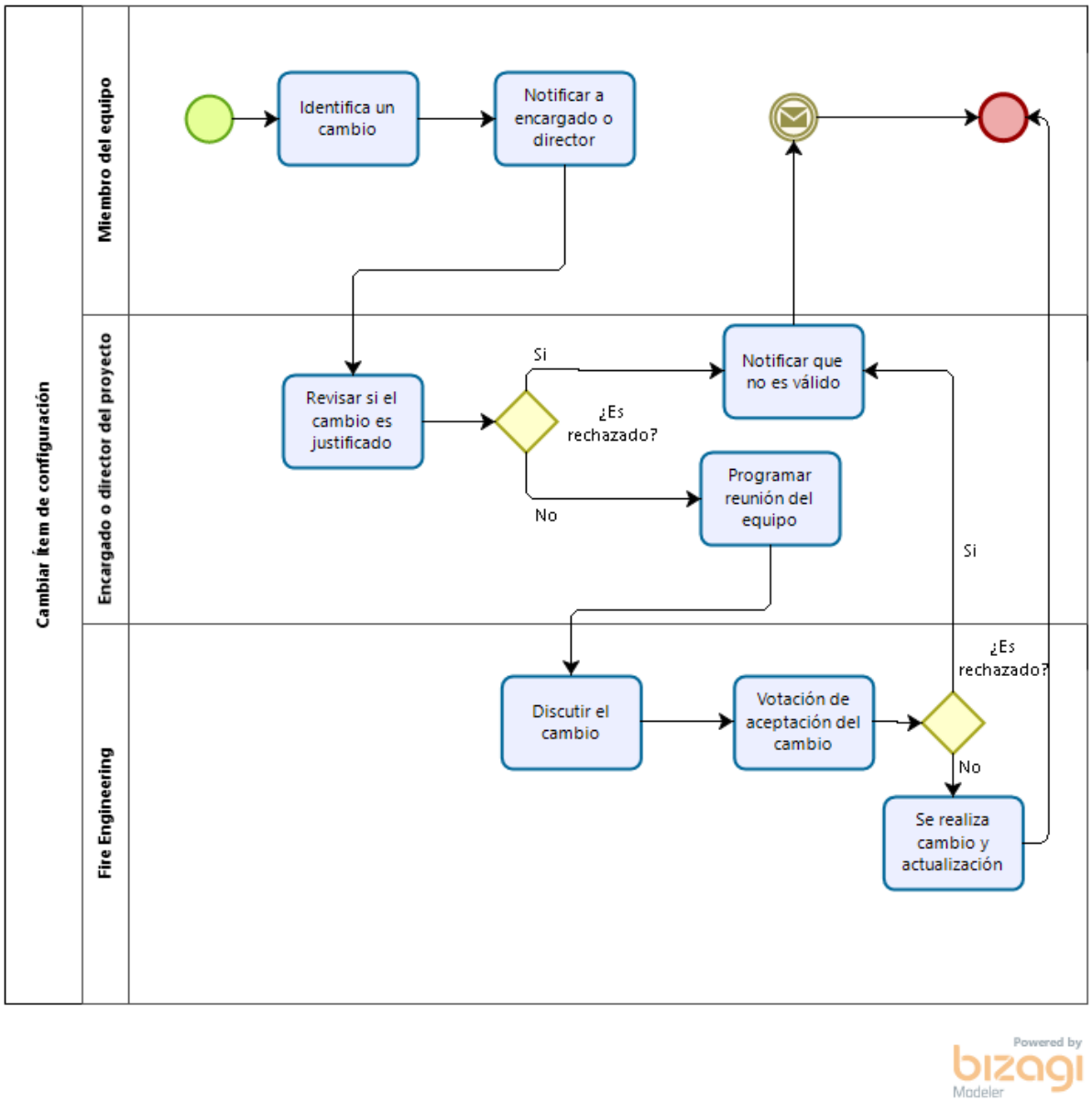


Figura 2: BPMN proceso de cambio a ítem de configuración.

11.4. Control de calidad.

Para el control de calidad se definieron tres procesos para las diferentes etapas. Primero, es el control de calidad de documentación, otro es el control de calidad de desarrollo y por último, el control de calidad del entregable final. A continuación se explicará cada uno de estos procesos.

11.4.1. Control de calidad de documentación.

En este proceso, cuando un integrante del grupo termina una tarea específica relacionada con algún ítem (sección o subsección) de un documento entregable, tales como el SPMP, SRS o SDD. Debe notificar al director de calidad y, al administrador de documentación y configuración para

que se inicie el proceso de verificación. Una vez estos son notificados, la tarea es revisada para ver si cumple con los requisitos que esta debe tener.

Si se encuentra algún error o falta cumplir algún requisito, se asigna a la tarea de nuevo a algún miembro para corregirla, una vez corregida, el proceso inicia de nuevo.

Si la tarea cumple con los requisitos en su totalidad, entonces es agregado al documento final y el proceso termina. Este proceso es ilustrado en la figura 3.

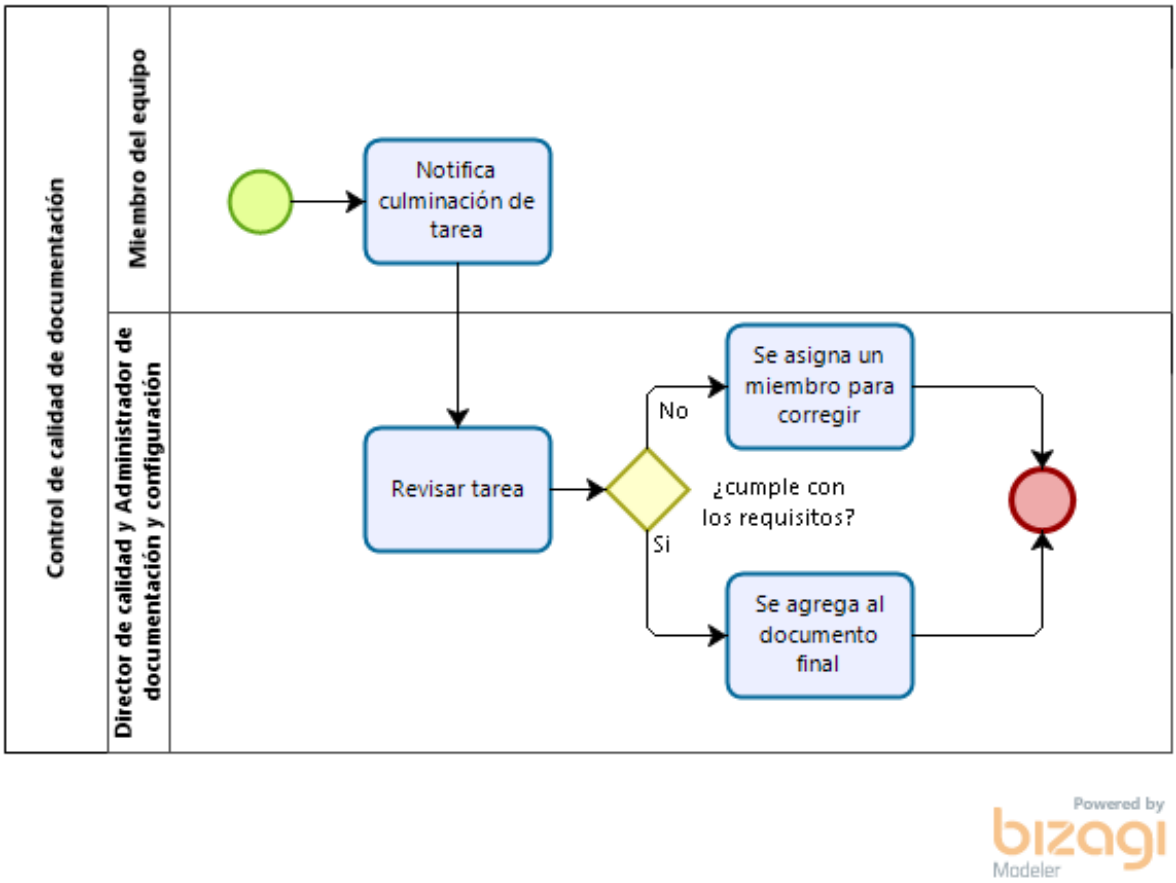


Figura 3: BPMN proceso de control de calidad de documentación.

11.4.2. Control de calidad de desarrollo.

En este proceso el inicio es igual que el anterior, el integrante que termina una funcionalidad del sistema informa al director de calidad y, al administrador de documentación y configuración para que empiece el proceso. Cuando ellos reciben dicha notificación se procede a probar la implementación de la funcionalidad en una rama del repositorio para verificar que cumpla con los requisitos funcionales y de calidad.

Si en la prueba se encuentra un error o inconsistencia, inmediatamente se asigna a alguien a la funcionalidad para que proceda a corregir los errores e iniciar de nuevo el proceso una vez termine. Pero, si no se halla ninguna clase de error, se integra esta funcionalidad a la rama Master del repositorio. Este proceso se ilustra en la figura 4.

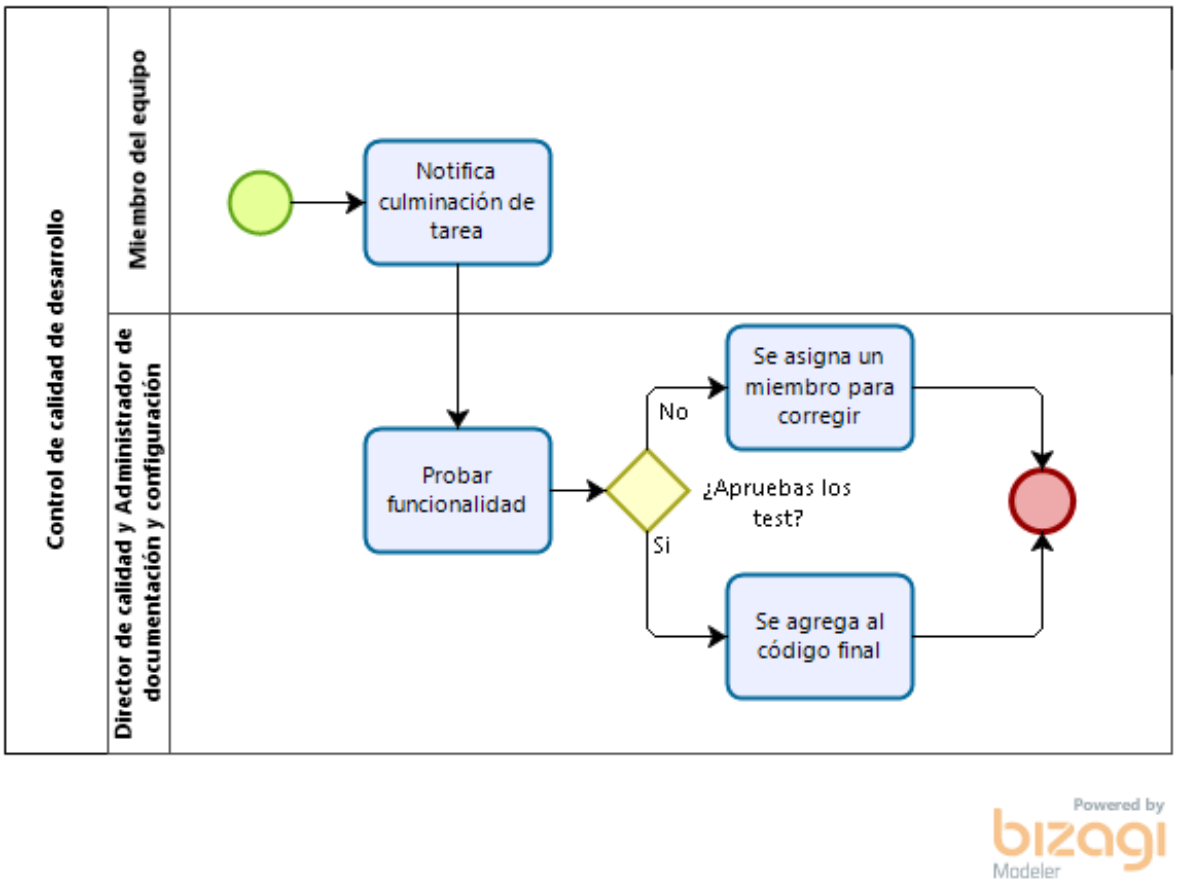


Figura 4: BPMN proceso de control de calidad de desarrollo.

11.4.3. Control de calidad del entregable final.

En este último proceso, el inicio es la culminación de un entregable(SPMP, SRS, SDD, prototipo funcional o manual del usuario). Para este proceso se empieza convocando una reunión donde asisten todos los integrantes del grupo para revisar el producto final. En la reunión se revisa componente a componente el producto, para verificar si está completo y cumple con los requisitos de calidad.

Si se encuentra alguna inconsistencia, se verifica con los responsables del componente en cuestión y se establece una solución. De lo contrario, el entregable se marca como listo para envío. Este proceso está ilustrado en la figura 5.

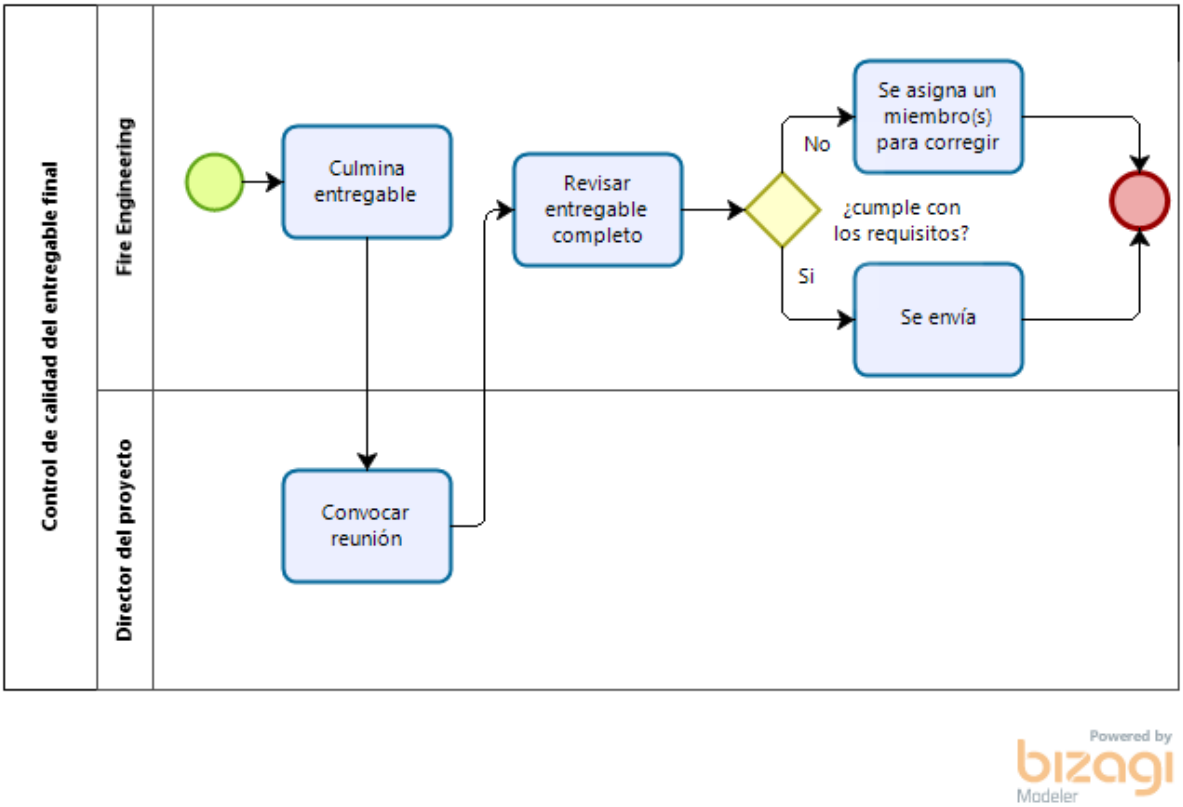


Figura 5: BPMN proceso de control de calidad de entregable final.

Referencias

[1] H. P. Managers, *Métricas en Kanban. Algo más de metodologías ágiles*, es. dirección: https://uv-mdap.com/blog/metricas-en-el-kanban-algo-mas-de-metodologias-agiles/ (visitado 26-08-2019).

[2] D. Conceptos.com, *Concepto de animales*, es. dirección: https://deconceptos.com/ciencias-naturales/animales (visitado 26-08-2019).

[3] Definicion.de, *Definición de mascota*, es. dirección: https://definicion.de/mascota/ (visitado 26-08-2019).

[4] P. y bienestar animal, *POLÍTICA PÚBLICA DE PROTECCIÓN Y BIENESTAR ANIMAL*, es. dirección: http://www.ambientebogota.gov.co/c/document_library/get_file?uuid=68eccc33-d792-49a1-89d8-43306e0fb429&groupId=10157 (visitado 26-08-2019).

[5] M. del web, *Herramientas adecuadas para el diseño y desarrollo de un sitio web*, es. dirección: http://www.maestrosdelweb.com/herramientas-adecuadas-para-el-diseno-y-desarrollo-de-un-sitio-web/ (visitado 26-08-2019).

[6] Anonimo, *LaTeX*, es. dirección: https://www.ecured.cu/LaTeX (visitado 26-08-2019).

[7] NeoAttack, *Concepto de Framework*, es. dirección: https://neoattack.com/neowiki/framework/ (visitado 26-08-2019).

[8] *Qué es SCRUM*, es-ES, ago. de 2008. dirección: https://proyectosagiles.org/que-es-scrum/ (visitado 26-08-2019).

[9] *XP - Extreme Programing Ingenieria de Software*. dirección: http://ingenieriadesoftware.mex.tl/52753_xp---extreme-programing.html (visitado 26-08-2019).

[10] *Metodología Kanban: ventajas y características*, es. dirección: https://www.getbillage.com/es/blog/metodologia-kanban-ventajas-y-caracteristicas (visitado 26-08-2019).

[11] *Introducción a Trello*. dirección: https://trello.com/es/guide (visitado 18-08-2019).

[12] *Precios de Trello*. dirección: https://trello.com/pricing (visitado 18-08-2019).

[13] *Plans and Pricing*. dirección: https://www.overleaf.com/user/subscription/plans (visitado 18-08-2019).

[14] *Documentación*. dirección: https://es.overleaf.com/learn/latex/Main_Page (visitado 18-08-2019).

[15] *Zotero | Storage*. dirección: https://www.zotero.org/storage (visitado 18-08-2019).

[16] *Kb:Importing Formatted Bibliographies [Zotero Documentation]*. dirección: https://www.zotero.org/support/kb/importing_formatted_bibliographies (visitado 18-08-2019).

[17] *Start [Zotero Documentation]*. dirección: https://www.zotero.org/support/ (visitado 18-08-2019).

[18] *Angular*. dirección: https://angular.io/license (visitado 18-08-2019).

[19] *Stack Overflow Developer Survey 2018*. dirección: https://insights.stackoverflow.com/survey/2018/?utm_source=so-owned&utm_medium=social&utm_campaign=dev-survey-2018&utm_content=social-share (visitado 18-08-2019).

[20] *Angular - Introduction to the Angular Docs*. dirección: https://angular.io/docs (visitado 18-08-2019).

[21] *Spring-Projects/Spring-Framework: Spring Framework*. dirección: https://github.com/spring-projects/spring-framework (visitado 18-08-2019).

[22] *Spring Framework. Contribute to Spring-Projects/Spring-Framework Development by Creating an Account on GitHub*, Spring, 18 de ago. de 2019. dirección: https://github.com/spring-projects/spring-framework (visitado 18-08-2019).

[23] *Spring Documentation*. dirección: https://spring.io/docs (visitado 19-08-2019).

[24] *Mirror of the Official PostgreSQL GIT Repository. Note That This Is Just a *mirror*: We Don't Work with Pull Requests on Github. To Contribute, Please See https://wiki.postgresql.org/wiki/Subm.*. PostgreSQL, 22 de ago. de 2019. dirección: https://github.com/postgres/postgres (visitado 22-08-2019).

[25] *PostgreSQL: License*. dirección: https://www.postgresql.org/about/licence/ (visitado 22-08-2019).

[26] *PostgreSQL: Documentation: 11: PostgreSQL 11.5 Documentation*. dirección: https://www.postgresql.org/docs/11/index.html (visitado 22-08-2019).

[27] *Git Handbook · GitHub Guides*. dirección: https://guides.github.com/introduction/git-handbook/ (visitado 18-08-2019).

[28] *Git Source Code Mirror: This Is a Publish-Only Repository and All Pull Requests Are Ignored. Please Follow Documentation/SubmittingPatches Procedure for Any of Your Improvements. - Git/Git*, Git, 18 de ago. de 2019. dirección: https://github.com/git/git (visitado 18-08-2019).

[29] *Pricing · Plans for every developer*. dirección: https://github.com/pricing (visitado 18-08-2019).

[30] *GitHub Terms of Service - GitHub Help*. dirección: https://help.github.com/en/articles/github-terms-of-service (visitado 18-08-2019).

[31] *Platform*. dirección: https://octoverse.github.com/platform.html (visitado 18-08-2019).

[32] *GitHub Guides*. dirección: https://guides.github.com/ (visitado 18-08-2019).

[33] *Angular Introduction: What It Is, and Why You Should Use It*, en, mar. de 2018. dirección: https://www.sitepoint.com/angular-introduction/ (visitado 26-08-2019).

[34] *PostgreSQL: About*. dirección: https://www.postgresql.org/about/ (visitado 26-08-2019).

[35] *spring.io*. dirección: https://spring.io/ (visitado 26-08-2019).

[36] E. G. S. o. Business, *10 competencias que todo gerente debe tener*, es. dirección: http://www.esan.edu.pe/apuntes-empresariales/2015/05/10-competencias-que-todo-gerente-debe-tener/ (visitado 26-08-2019).

[37] *php architect's Guide to Enterprise PHP Development - PDF Free Download*, en. dirección: https://epdf.pub/php-architects-guide-to-enterprise-php-development.html (visitado 26-08-2019).

[38] *¿Cuáles son las funciones de un analista de riesgo?* Dirección: https://www.isotools.org/2015/08/17/cuales-son-las-funciones-de-un-analista-de-riesgo/ (visitado 26-08-2019).

[39] M. Cohn, *What Are Story Points?*, ago. de 2016. dirección: https://www.mountaingoatsoftware.com/blog/what-are-story-points.

[40] ——, *Estimating With Use Case Points*, oct. de 2005. dirección: https://www.mountaingoatsoftware.com/articles/estimating-with-use-case-points.

[41] *Scrum Poker Cards (Agile) - Apps on Google Play*. dirección: https://play.google.com/store/apps/details?id=artarmin.android.scrum.poker&hl=en.

[42] H. P. Managers, *Métricas en Kanban. Algo más de metodologías ágiles*, es. dirección: https://uv-mdap.com/blog/metricas-en-el-kamban-algo-mas-de-metodologias-agiles/ (visitado 26-08-2019).

[43] A. Cajal, *¿Qué es la Matriz DOFA Personal y Cómo se Hace?*, es. dirección: https://www.lifeder.com/matriz-dofa-personal/ (visitado 26-08-2019).

[44] *Software Engineering | Software Project Management Plan (SPMP)*, en-US, mayo de 2019.

[45] «IEEE Standard for Information Technology–Systems Design–Software Design Descriptions», *IEEE STD 1016-2009*, págs. 1-35, jul. de 2009. DOI: 10.1109/IEEESTD.2009.5167255.

[46] P. Tiwari y L. Ananthamurthy, *Creating UML Use Case Diagrams*, mar. de 2003. dirección: https://www.developer.com/design/article.php/2109801.

[47] *UML 2 Class Diagrams: An Agile Introduction*. dirección: http://www.agilemodeling.com/artifacts/classDiagram.htm.

[48] *Modelo Entidad Relación*. dirección: http://aprende.colombiaaprende.edu.co/sites/default/files/naspublic/curriculos%5C_ex/n2g10%5C_pweb1/nivel2/web1/unidad2/leccion3.html.

[49] *Source Code Definition by The Linux Information Project*. dirección: http://www.linfo.org/source%5C_code.html.

[50] *Online Technical Writing: User Guides*, ago. de 2009. dirección: https://web.archive.org/web/20090819022032/http://www.io.com/%5Ctextasciitilde%7B%7Dhcexres/textbook/user%5C_guides.html.