‘Active’ SPMP

Versión 1.8.3

Proyecto de Ingeniería de software

Entrega 1

Active
Juan Miguel Gómez
Carlos Eduardo Camacho
Natalia Sofía Otero
Luis Felipe Hurtado
Andrés García

Bogotá D.C
29/08/2017
Versión 1.0

Historial de Cambios

	Fecha	Descripción	Secciones Afectadas	Responsables
1	25/07/17	Creación del documento	General	Natalia Sofía Otero O.
2	25/07/17	Redacción de sección	1	Carlos Camacho
3	05/08/17	Redacción de sección	5.2.3 6.4	Andrés García Juan Gómez
4	07/08/17	Redacción secciones	5.4	Carlos Camacho
5	10/08/17	Correcciones sección	5.4	Carlos Camacho
6	11/08/17	Redacción de sección	5.5	Carlos Camacho
7	12/08/17	Redacción de sección	6.4.2	Juan Gómez
8	13/08/17	Redacción de sección	6.3-6.4.1	Andrés Gómez
9	14/08/17	Corrección de sección	6.4.2	Juan Gómez
10	14/08/17	Redacción de sección	5.3	Juan Gómez
10	15/08/17	Redacción de sección	7.3.4	Carlos Camacho
11	16/08/17	Corrección de sección	7.3.4	Carlos Camacho
12	16/08/17	Redacción de sección	10.1	Carlos Camacho
13	16/08/17	Anexo	Logo	Luis Urdaneta
14	17/08/17	Corrección de sección	10.1	Luis Urdaneta
15	17/08/17	Redacción de sección	9	Luis Urdaneta
16	17/08/17	Corrección de sección	6.3-6.4.1	Andrés García
17	17/08/17	Redacción de sección	5.2.1-5.2.2	Luis Urdaneta
18	18/08/17	Redacción de sección	7.2.1	Natalia Otero
19	18/08/17	Corrección de sección	7.2.1	Natalia Otero
20	18/08/17	Redacción de sección	7.3.1	Natalia Otero
21	18/08/17	Redacción de sesión	6.1	Juan Gómez
22	18/08/17	Redacción de sección	10.4	Andrés Gómez
23	18/08/17	Redacción de sección	7.1.1	Andrés Gómez
24	18/08/17	Redacción de sección	6.1.2	Juan Gómez

25	18/08/17	Redacción de sección	10.2	Carlos Camacho
26	19/08/17	Anexo	Riesgos	Carlos Camacho
27	20/08/17	Modificación de anexo	Riesgo	Juan Gómez
28	20/08/17	Corrección de sección	10.4	Andrés García
29	20/08/17	Redacción de sección	7.3.4	Juan Gómez
30	20/08/17	Corrección de sección	6.4.1	Adres García
31	20/08/17	Corrección de sección	6.1	Juan Gómez
32	20/08/17	Redacción de sección	8.1	Juan Gómez
33	20/08/17	Redacción de sección	6.2	Felipe Urdaneta
34	21/08/17	Anexo	Diagrama de Gantt	Natalia Otero
35	22/08/17	Redacción de sección	8.2	Juan Gómez
36	22/08/17	Redacción de sección	7.1.1-7.3	Andrés García
37	23/08/17	Redacción de sección	10.5	Lui Urdaneta
38	24/08/17	Modificación de sección	6.4.2	Natalia Otero
39	24/08/17	Corrección de sección	10.4	Andrés García
40	24/08/17	Redacción de sección	7.3.2	Natalia Otero
41	24/08/17	Redacción de sección	8.3	Juan Gómez
42	25/08/17	Corrección de sección	7.3.3	Natalia Otero
43	25/08/17	Anexo	Tabla de estimación	Andrés García
44	25/08/17	Anexo	Diagramas Casos de uso	Andrés García
45	26/08/17	Corrección de sección	10.4	Juan Gómez
46	27/08/17	Corrección de sección	10.5	Luis Urdaneta

Tabla 1 Historial de Cambios

1. Prefacio

En el presente documento se presenta el plan de gestión y el plan de desarrollo del proyecto *"CampusFinder"*; este documento se encuentra dirigido a todo aquel que quiera conocer en detalle la planificación y desarrollo de la aplicación.

A través del Software Project Management Plan (SPMP, por sus siglas en inglés) se dará a conocer nuestro proyecto, esto se logrará por medio del desarrollo de diferentes apartados como la visión general del proyecto en el cual se plante el propósito que deseamos cumplir, el alcance que esperamos obtener, los objetivos generales y específicos entre otros.

De acuerdo con la estructura de este documento se encontrarán en los siguientes apartados, las especificaciones de contextualización del proyecto y los diferentes elementos que se emplearon para la elaboración del trabajo, tales como, el modelo de ciclo de vida, análisis y toma de decisión de las diferentes herramientas y lenguajes implementados, gestión de recursos humanos, asignación de tareas, comunicación entre el equipo de trabajo y demás variables para el desarrollo del proyecto.

Otro aspecto importante es la administración del proyecto donde se establecen los planes de trabajo del proyecto en el cual se especifican las actividades, cronogramas, asignación de recursos entre otros; este apartado se enfoca en dar explicación del manejo que se dio al proyecto. Por último, encontramos el apartado de monitoreo y control del proyecto donde especificamos la forma de medición del progreso del proyecto, control de calidad y demás parámetros para lograr cumplir los objetivos y obtener un proyecto de calidad.

Para finalizar en este documento se le presenta al lector las metas y objetivos del grupo Active y cuál es la estrategia por seguir para cumplir con lo anteriormente mencionado.

2. Tabla de contenidos

3. Lista de Figuras

4. Lista de Tablas

5. Vista General del Proyecto

5.1. Visión del producto

CampusFinder es un aplicativo web propuesto por el grupo *Active* que tiene como objetivo y funcionalidad principal facilitar la búsqueda y ubicación de los edificios del campus de la Universidad Javeriana. Tomando como referencia *GoogleMaps* y cómo guía de diseño *Waze*.

Adicionalmente, el aplicativo contará con funcionalidades cómo: notificaciones de eventos culturales, información relevante de los edificios y sitios de interés para la comunidad a la que está dirigido el proyecto y, para los estudiantes de la Javeriana, tendrá una calculadora de notas, tanto de cada una de las asignaturas como para el promedio ponderado general.

La información de los edificios que se brindará será en cuanto a lugares de posible interés cómo baños, auditorios, cafeterías, facultades, oficinas, salas de estudio, horarios, ubicación de las entradas y las salidas disponibles.

De acuerdo con los datos suministrados por las encuestas se obtienen los siguientes puntos o información de interés para los usuarios:

- # Pisos
- Pisos de los Baños (hombres/ mujeres)
- Salas de estudio/ lugares de estudio
- Facultades/carreras (cual y cual piso) //puntos de pago
- Auditorios
- # Ascensores
- Entradas/Salidas
- Puntos de alimentación (cafeterías, máquinas)
- Horarios
- Nombre/#
- Sitios de descanso/espera
- Oficinas (DTI, consulta E, tesorería, precio/min, cosas perdidas, vigilancia) (Horarios de atención)
- Bancos/cajeros
- Laboratorios
- Microondas
- Casilleros
- Puntos de impresión

5.2. Propósito, Alcance y Objetivos

5.2.1. Propósito:

Este proyecto se desarrolla pensando en las necesidades de todos los individuos que en algún momento interactúen con el campus del medio universitario, en especial los estudiantes y los funcionarios que diariamente transitan por la universidad y se verían beneficiados por la información que Campus Finder les puede brindar.

Campus Finder ofrecerá información actualizada y concisa acerca del estado de diferentes elementos y espacios de la universidad. Con el propósito de que el usuario pueda organizar mejor su tiempo gracias a sus diferentes herramientas como el horario de clases, la calculadora de notas, el sistema de

alarmas personalizadas y su red de usuarios cercanos; con los cuales podrá compartir y coordinar reuniones o rutas personalizadas.

Con el desarrollo de este proyecto, se espera que los integrantes apliquen los conocimientos teóricos que se ven durante las clases con el objetivo de adquirir experiencia acerca del proceso de ingeniería de software aplicados en el desarrollo de proyectos. Después de realizar el proyecto, los integrantes del grupo tendrán una mejor idea acerca de los diferentes eventos que pueden llegar a ocurrir durante el desarrollo de un proyecto y de cómo afrontar estas situaciones, siempre buscando mejorar para poder entregar un producto que satisfaga las necesidades del cliente.

5.2.2. Alcance

El proyecto se planea presentar como una herramienta de soporte con la cual el usuario pueda organizar, planear y orientarse dentro del campus del medio universitario sin necesidad de conocer la planta física de la universidad con anterioridad, ya que, por medio de una interfaz intuitiva y dinámica, el usuario podrá ubicarse fácilmente, interactuar con otros usuarios y manejar su información personal.

El proyecto implementara las siguientes funcionalidades:

- La aplicación permitirá crear una cuenta de usuario.
- El sistema registrará en la base de datos la información de los usuarios.
- El usuario puede cambiar su información en cualquier momento.
- El sistema mostrara un mapa interactivo de la planta física de la universidad.
- El usuario podrá buscar la ubicación de un espacio, punto de interés o de un edificio.
- El sistema mostrara puntos de interés como pueden ser: bici parqueaderos, cafeterías, kioscos de alimentos, baños, salas y espacios de estudio.
- La aplicación mostrara información de interés de cada edificio.
- El usuario podrá ver las diferentes rutas disponibles para el trayecto entre diferentes espacios.
- El usuario podrá filtrar rutas agregando características adicionales a la búsqueda.
- El usuario podrá marcar sus edificios y puntos de interés favoritos.
- Las rutas creadas por el sistema podrán ser guardadas por los usuarios.
- El usuario podrá compartir su ubicación y sus rutas preferidas con otros usuarios.
- El sistema mostrara al usuario un calendario con eventos planeados.
- El usuario contará con una calculadora de notas con la cual podrá llevar un registro de su progreso académico en diferentes materias.

- El usuario podrá manejar las notificaciones y agregar sus propias alarmas.
- El usuario puede guardar su horario dentro de la aplicación.
- El usuario podrá calificar el estado de algunos de los elementos del campus.
- El sistema mostrara la retroalimentación dada por los usuarios de los diferentes puntos de interés.
- La base de datos contendrá la información actualizada de todos los elementos del campus de la universidad.
- La aplicación tendrá una interfaz intuitiva y agradable visualmente.

La plataforma no se encargará de manejar y asignar los horarios de los espacios de la universidad, solo se le presentara al usuario la información que la universidad y el usuario voluntariamente han entregado (se asegura que la información sensible no será compartida con terceros). El sistema no contara con opciones de personalización y no se asegura su completa funcionalidad en todos los dispositivos.

5.2.3. Objetivos:

5.2.3.1. Objetivo General:

Desarrollar una aplicación móvil que facilite la ubicación y movilidad de la comunidad educativa en el campus universitario.

5.2.3.2. Objetivos Específicos

- Ofrecer una herramienta que facilite la ubicación de los edificios del campus a toda la comunidad educativa.
- Ofrecer una solución indicada para optimizar el tiempo de cada persona de la comunidad educativa a la hora de trasladarse de un lugar a otro dentro del campus.
- Proponer rutas a la comunidad educativa con discapacidad física una ruta adecuada de acuerdo con su condición.
- Fomentar el conocimiento de todo el campus y los diferentes espacios que tiene.
- Desarrollar la aplicación web tomando en cuenta todos los procesos que corresponden a la ingeniería de software.

5.3. Supuestos y Restricciones

5.3.1. Supuestos:

Un supuesto se define como un factor que se considera cierto, real o certero sin pruebas o demostraciones que adicionalmente describe el impacto de estos factores en caso tal de ser probados como falso. (Software Engineering Institute, s.f.) Los supuestos agregan un riesgo al proyecto dado a que pueden ser falsos (Weese, 2009)

- Supuestos:
  - ✓ El sistema se ejecutará desde un navegador. El navegador debe ser: Google Chrome versión 60 o mayor, Safari versión 10 o mayor, Firefox versión 55 o mayor o Opera versión 47 o mayor
  - ✓ El sistema proveerá siempre información correcta al usuario.
  - ✓ Los integrantes del grupo permanecerán en el grupo y cumplirán debidamente con su trabajo siguiendo las normas previamente establecidas.
  - ✓ El desarrollo del proyecto no necesitara de un presupuesto amplio.
  - ✓ El manejo de riesgos se efectuará de manera correcta.
  - ✓ La comunicación entre miembros del equipo y entre el equipo y el cliente será constante, clara y concisa.
  - ✓ Cada integrante del grupo tendrá los materiales y herramientas necesarias para el desarrollo del proyecto.

5.3.2. Restricciones:

Una restricción es un factor limitante externo que afecta la ejecución de un proyecto o proceso. (Project Management Institute Staff, 2013) Es una condición de frontera o limitante de lo que puedes hacer. Se hace una distinción de dos tipos de restricciones: de negocio y técnicos. Las restricciones de negocio normalmente se enfocan en recursos, como el tiempo o el presupuesto. Las restricciones técnicas son decisiones arquitectónicas que limitan el diseño de la solución. Suelen ser inflexibles e invariantes. Ejemplos de este tipo de restricción incluyen lenguaje de programación, hardware, software, etc. (Weese, 2009)

- Restricciones de Técnicas:
  - ✓ El producto deberá poder ser usado desde cualquier dispositivo móvil Android v4.1 o IOS 10, que posea un navegador con conexión a internet.
  - ✓ El producto deberá usar una arquitectura cliente/servidor
  - ✓ El producto deberá hacer uso de un sistema de persistencia
  - ✓ El producto deberá tener un manejo fuerte de interfaz gráfica de usuario(GUI)
  - ✓ El producto deberá hacer uso del paradigma orientado a objetos
  - ✓ El proyecto deberá hacer uso de algún control de versiones

- Restricciones de Negocio:
  - ✓ El proyecto se realizará durante el transcurso del semestre 1701 de la pontificia universidad javeriana
  - ✓ El presupuesto y otros recursos del proyecto dependerá de los recursos que tengan los integrantes.

- ✓ Las entregas del proyecto son inamovibles y consta de tres entregas a lo largo del semestre.
- ✓ No se permite el uso de generadores de código.
- ✓ Para la realización del proyecto solo se cuenta con los integrantes del grupo Active y no podrá recibir ayuda directa de externos.

5.4. Entregables

Como entregables se especifican que son todos aquellos archivos, documentos y anexos que se le entregan al cliente y a los integrantes del grupo con fin de reportar resultados del proyecto.

El primer entregable contendrá todos aquellos documentos con el SPMP. Como aspectos relevantes de este entregable encontramos: la vista general del proyecto, donde se especifican los propósitos, los objetivos y el alcance del proyecto; también encontramos las explicaciones de los diferentes ciclos de vida y herramientas propuestas para el desarrollo y la respectiva sustentación de la toma de decisión para estos. Por último, como otro aspecto relevante encontraremos la planificación respectiva de los integrantes del proyecto por medio de un organigrama.

El segundo entregable contendrá la documentación relacionada con los requerimientos de software. Como secciones a destacar encontramos: todas las funciones y restricciones del producto, los requerimientos, el proceso de desarrollo, clasificación y verificación de los requerimientos y por último se realizará la entrega de un prototipo para la revisión por el cliente.

La tercera está conformada por los documentos relacionados con el diseño de software. En este entregable destacamos la arquitectura del producto, el diseño y el prototipo final con la documentación del código, el manual de usuario y de instalación aparte de un reporte gerencial en el cual se especifican las etapas de desarrollo del proyecto. Los criterios de evaluación se encuentran establecidos en el documento. En cuanto a la aceptación del proyecto, será el cliente quien de la aprobación final.

5.5. Resumen de calendarización y presupuesto

5.5.1. Calendarización

Para el desarrollo de “CampusFinder” fueron especificadas tres entregas por el scrum owner, las cuales se especifican en la sección *5.4 Entregables.* Para cada una de las entregas especificadas se planearon sprints como se especifica en la sección *6.1 Modelos de ciclos de vida*. En cada sprint se especifican las actividades a desarrollar para mejorar el control del desarrollo del proyecto. Aparte de los sprint se plantean encuentros con el scrum owner, lo cual nos permite encaminar el proyecto con lo que realmente quiere el cliente.

A continuación, se presentan las diferentes actividades e hitos para el desarrollo de este proyecto

Tipo	Actividad	Fecha de inicio
Actividad	SPMP	17/Julio/2017
Actividad	Especificaciones de Sprint	1/Agosto/2017
Hito	Entrega de Sprint	
Hito	Entrega SPMP	29/Agosto/2017
Hito	Inicio Sprint 1	30/Julio/2017
Hito	Inicio Sprint 2	16/Agosto/2017
Actividad	Sustentación SPMP	29/Agosto/2017
Hito	Post-Mortem	24/Noviembre/2017

Tabla 2 Calendario

5.5.2. Presupuesto

De acuerdo con lo especificado en la sección 7.*3.4 Asignación de presupuesto* y justificación, se estima una inversión de $3'276.000COP para el desarrollo. Como el desarrollo del proyecto se realiza de manera iterativa por lo cual en la especificación de cada sprint se especifica la asignación del presupuesto

5.6. Evolución del plan

Para el manejo y control de cambios, el grupo Active decidió trabajar bajo la metodología Scrum y según la organización propuesta por ITIL. (Project Management Institute Staff, 2013)

Por facilidad y organización, se destinó un "Administrador de documento" que es el encargado de la gerencia y modificación del documento, por lo tanto, cada cambio de contenido que se quiera realizar durante el proceso,

Según la ITIL hay dos posibles clases de cambios que se pueden originar durante la elaboración del documento, siendo estos cambios normarles o cambios de emergencia. El grupo Active estableció que los cambios normales son por cambios de contenido de una o más secciones; para estos cambios se debe pasar primero por los *Scrum Masters,* donde se revisa la redacción y la coherencia de la propuesta, buscando establecer un cambio concreto eliminando la mayor cantidad de ambigüedades posibles. Posteriormente, se ejecuta la validación de este, por parte de todos los miembros del equipo Active, puesto que el cambio puede llegar a implicarla modificación de otras secciones del documento. Finalmente, y con la validación del grupo, el encargado del documento procederá a realizar la modificación.

Los cambios de emergencia encierran cambios de ortografía, orden de ideas o cambios surgidos una vez terminado el documento. Dichos cambios tendrán cavidad en las reuniones de *Daily Scrum* y, el Administrador de Documento, será

responsable de hacer las modificaciones al documento durante la reunión. Una vez finalizado el documento, cada uno de los integrantes tendrá acceso de lectura, para que, en la reunión de cierre se hagan las últimas modificaciones y correcciones garantizando la mejor calidad del producto.

5.7. Glosario

- **ITIL:** Information Technology Infrastructure Library, es la Biblioteca de Infraestructura de las Tecnologías de la Información más aceptadas para Servicios TI en el mundo.
- **Product Owner:** Según los roles estipulados por Scrum el Product owner tiene la función de: definición de características del producto; definición de la fecha de lanzamiento del producto y su contenido; Rentabilidad del producto.
- **Servidor:** Sistema que provee datos y/o servicios a otro sistema sobre la red
- **Daily Scrum:**
- **Sistema de control de versiones:** Es una herramienta para el registro de cambios de uno o más proyectos guardando versiones del producto a de acuerdo con las fases de este.
- **Servidor web:** Es un programa encargado de gestionar aplicativos del lado del servidor, a través de conexiones bidireccionales o unidireccionales con una sincronización síncrona o asíncrona con el cliente, generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente.
- **SQL:** Structured Query Language o por su definición en español lenguaje de consultas estructurado.
- **HTTP:** Hypertext Transfer Protocol o por su definición en español Protocolo de transferencia de hipertexto.
- **PHP:** Hypertext Preprocessor o por su definición en español preprocesador de hipertexto.
- **JavaScript:** Lenguaje de programación interpretado, dialecto del estandarizado; es un lenguaje orientado a objetos basado en prototipos imperativos, débilmente tipados y dinámico.
- **CSS:** Cascading Style Sheet o por su definición en español Hoja de estilo en cascada.
- **Framework:** Estructura o infraestructura dedicada a servir como soporte o guía para la construcción de algo, con el fin de expandirse para crear algo útil.

6. Contexto del proyecto

Todo proyecto debe de tener un modelo de ciclo de vida que defina como va a ser desarrollado. Esto con el propósito de darle un orden a las actividades que se van a ejecutar y así lograr la terminación adecuada del proyecto. El ciclo de vida que el grupo Active escogió para el proyecto es una adaptación de Scrum con XP.

6.1. Modelo de ciclo de vida

6.1.1.1.Scrum

Es un marco de trabajo ligero para manejo de proyectos, principalmente usado para el desarrollo de software. Describe un acercamiento iterativo e incremental para el trabajo del proyecto. (Scrum Alliance, s.f.) Los componentes principales de Scrum son:

I. Roles

a. *Scrum Team*

Es una colección de individuos que trabajan juntos para entregar el producto. Para lograr ser efectivo, el equipo debe tener un objetivo común, seguir normas establecidas y mostrar respeto entre sus integrantes. El *scrum team* se trata como un conjunto, es decir, un error o un logro se atribuye al equipo, no a individuos.

Las características notables de un *scrum team* son:

- Los integrantes comparten y siguen las mismas reglas
- Todo el equipo es responsable de su entrega
- El equipo tiene poder de decisión
- Trabaja de la manera más autónoma posible
- Se auto-organiza
- Las habilidades dentro del equipo están balanceadas
- Es pequeño e indivisible
- Integrantes trabajan tiempo completo con el equipo

Por otro lado, las reglas de un *scrum team* deben contener unos puntos clave que son el tiempo y ubicación de las reuniones, la definición de terminado, como se debe programar y que herramientas se va a usar. (Scrum Alliance, s.f.)

b. Scrum Master

Es el líder que le sirve al equipo, no al contrario. Se ve como un protector del equipo, por ende, debe asegurar que todos aquellos involucrados con el proyecto y en especial el equipo de desarrollo, puedan concentrarse en su trabajo sin distracciones. Esto incluye proteger al equipo del *product owner,* o de problemas organizacionales y distracciones internas, por ejemplo, arreglar computadores o generar un ambiente más callado de trabajo.

Además de proteger al equipo, el *scrum master* debe proteger el proceso de Scrum como tal. Debe ser experto en los procesos que componen a Scrum y como se deben aplicar. Deber asegurar que el equipo se mantenga dentro de Scrum. (International Scrum Institute, s.f.)

c. *Scrum Product Owner*

Es de donde emerge el valor de negocio. Es el responsable de definir que se va a hacer y de priorizar el trabajo. Debe saber que se debe entregar y porque es importante para los clientes, para el mercado y para la organización. Debe ser un guía experto capaz de llevar el equipo a completar el proyecto. Lo más importante y distinto de este rol es que el *product owner* deber involucrarse en el proyecto y permanecer activo durante su ejecución. (SCRUMstudy, 2016)

II. Proceso

Scrum es un proceso iterativo e incremental. Para este, se requiere de ciertos artefactos y procesos propios de Scrum que son:

a. Product Backlog

Es una lista de todas las cosas que necesitan hacerse dentro del proyecto. Reemplaza los artefactos de especificación de requerimientos tradicionales. Los ítems pueden ser tanto técnicos como centrados a usuarios. El dueño del Scrum Product Backlog es el Scrum Product Owner

Un Scrum Product Backlog se diferencia de una lista simple de actividades dado a las siguientes características:

- Toda entrada en el Scrum Product Backlog le adiciona valor al cliente.
- Las entradas están priorizadas y ordenadas de manera acorde.
- El nivel de detalle depende de la posición de la entrada dentro del Scrum Product Backlog
- Todas las entradas son estimadas.
- El Scrum Product Backlog es un documento vivo, es decir cambiante durante el desarrollo del proyecto.
- El Scrum Product Backlog no especifica acciones ni actividades de bajo nivel.

Dado a que es un documento vivo, el Scrum Product Backlog cambia a medida que el proyecto se desarrolla. Si se encuentran nuevos ítem en el desarrollo del proyecto, se deben agregar al Scrum Product Backlog.

b. Scrum User Stories

Normalmente, los ítems dentro de un Scrum Product Backlog son descritos como historias de usuario. Esto es una historia corta acerca de alguien usando el producto. Tiene nombre, una

narrativa pequeña, criterio de aceptación y condiciones para que la historia se complete. Ejemplos de plantillas para historia de usuario son (International Scrum Institute, s.f.):
Como un [actor], Yo [quiero|debo] [acción] para [logro]
Como un [actor], Yo [quiero|debo] [logro]

Actor: El "dueño" de la historia

Acción: Lo que el actor hace. Si es obligatoria debe ir precedida de "debo". De lo contrario se debe usar "quiero"

Logro: Lo que el actor quiere lograr al ejecutar la acción.

c. Sprint

Es el corazón de Scrum. Es un lapso de un mes o menos, en el un incremento de producto terminado, usable y potencialmente liberable se crea. Son de tiempo constante durante el proyecto y deben empezar donde el otro termina. (Scrum.Org, s.f.)

d. Sprint Planning:

El trabajo que se va a ejecutar en un Sprint es planificado en el Sprint Planning. (Scrum.Org, s.f.) Debe comenzar con un WHAT-Meeting. El propósito de esta sesión es de definir un Sprint Backlog realista con los ítems que pueden ser implementados completamente hasta el final del Sprint. Después se debe hacer un HOW-Meeting que se define las tareas concretas necesarias para la implementación completa de las entradas del Scrum Product Backlog. (The Interantional Scrum Institute)

e. Sprint Backlog

Es un set de ítems del Product Backlog seleccionados para un Sprint, adicionando un plan para entrega del incremento del producto y completar el objetivo del Sprint. (Scrum.Org, s.f.)

f. Scrum Burndown Chart

Es una herramienta de medición visual que muestra el trabajo hecho por día contra la tasa de trabajo proyectada para la entrega actual. Su propósito es el de habilitar que el proyecto esté en el camino correcto para hacer la entrega de la solución esperada dentro del tiempo deseado.

g. Definition of Done (DOD)

En el momento que un item del Scrum Product Backlog se describe como "terminado", todos deben saber que significa "terminado". Por ende, se debe definir que significa "terminado" para el Scrum Team. La definición de terminado debe ir expandiéndose a medida que el equipo madure para tener un estándar. (Scrum.Org, s.f.)

h. Daily Scrum Meeting / Daily Standup Meeting:

Es una reunión diaria que se caracteriza por ser corta e idealmente al principio de cada día de trabajo. Cada miembro que trabaje para completar el sprint debe participar. En la reunión, todos deben responder las siguientes preguntas:

- ¿Qué has logrado desde la reunión anterior?
- ¿Qué vas a lograr para la siguiente reunión?
- ¿Qué te impide que logres tus tareas?

Todos los miembros del equipo deben atender a la reunión y se recomienda que estén de pie. Adicionalmente se recomienda que la reunión no dure más de 15 minutos. La falta de tiempo hace que no se pueda ignorar cualquier inconveniente que los integrantes reporten y el Scrum Master debe anotarlos.

i. Sprint Review Meeting

Es una reunión que se lleva a cabo al final de cada sprint. En esta, se hace seguimiento de que ítems se completaron del Product Backlog durante el sprint. Se puede hacer mediante demostraciones de funcionalidad. Aquello que no esté completo no se debe demostrar. Esta reunión debe ser informal y rápida.

j. Sprint Retrospective Meeting

Esta reunión se debe hacer después de la *Sprint Review Meeting* y se reúnen el Scrum Master y el Scrum Team. Esta reunión debe tener un tiempo definido. El propósito es responder tres preguntas acerca del sprint pasado:

- ¿En que nos fue bien durante el sprint?
- ¿En que nos fue mal durante el sprint?
- ¿Cómo podemos mejorar para el siguiente sprint?

Scrum agrega valor al proyecto, en el aspecto gerencial, por medio de todos los artefactos y procesos mencionados anteriormente. En especial da claridad a la cantidad de trabajo hecho y por hacerse y además propone roles que ayudan a la simplicidad estructural del grupo. Es por lo anterior que el grupo Active lo escogió.

6.1.1.2. Extreme Programming (XP)

Extreme programming es un marco de trabajo ágil para desarrollo de software que busca producir software de mayor calidad, y mayor calidad de vida para el equipo de desarrollo. XP es el más específico de los marcos de trabajo agiles en cuanto a prácticas apropiadas para desarrollo de software (Agile Alliance, s.f.)

Este marco de trabajo se caracteriza por cinco valores con los cuales se definen un set de reglas para cada parte del desarrollo de software. Según la definición del autor Don Wells, estos son:

- Valores
  - ✓ **Simplicidad:** Se hará lo que es necesario y lo que se pide, pero no más. Esto maximiza el valor creado por la inversión hecha. Se tomarán simples y pequeños pasos para lograr nuestro objetivo y mitigar las fallas a medida que aparezcan. Crearemos algo que nos haga orgullosos y lo mantendremos a largo plazo por costos razonables.
  - ✓ **Comunicación:** Todos son parte de un mismo equipo y nos comunicamos de cara a cara a diario. Trabajaremos juntos en todo desde requerimientos hasta código. Crearemos la mejor solución juntos.
  - ✓ **Retroalimentación:** Tomaremos cada compromiso de iteración seriamente por medio de la entrega de software funcional. Demostraremos el software de manera temprana y a menudo y escucharemos atentamente y haremos los cambios necesarios. Hablaremos sobre el proyecto y adaptaremos nuestros procesos a él, no al revés.
  - ✓ **Respeto:** Todos dan y sienten el respeto que merecen como un integrante valorado del equipo. Todos contribuyen valor, aunque sea solo con entusiasmo. Los desarrolladores respetan la experiencia del cliente y viceversa. Gerencia respeta nuestro derecho de aceptar responsabilidad y de tener autoridad sobre nuestro trabajo.
  - ✓ **Coraje:** Diremos la verdad acerca nuestro progreso y estimaciones. No documentamos excusas para el fracaso porque planeamos tener éxito. Nosotros no le tememos a nada porque nadie nunca trabaja solo. Nos adaptaremos a los cambios cuando sucedan.
- Reglas

  Las siguientes reglas son el producto natural de la maximización de los valores mencionados previamente. Es por lo que Don Wells menciona explícitamente que XP no es un set de reglas si no una manera de trabajar en armonía con valores personales y corporativos y sugiere agregar reglas propias que reflejen estos valores.

  - ✓ Reglas de Planificación
    - ➢ Historias de usuario son descritas.
    - ➢ La planificación de entregas crea el horario de entregas.
    - ➢ Has entregas pequeñas y frecuentes.
    - ➢ El proyecto se divide en iteraciones.

- ✓ Reglas de Gerencia
  - ➢ Dale al equipo un espacio abierto de trabajo dedicado.
  - ➢ Ten un ritmo sostenible.
  - ➢ Una reunión de pie al comienzo de cada día.
  - ➢ La velocidad de proyecto es medida. Esto es simplemente cuanto trabajo se está haciendo en tu proyecto.
  - ➢ Mueve personas
  - ➢ Arregla el XP cuando se rompa. Esto significa que XP se debe adaptar en caso de que no funcione. Se sugiere empezar con las reglas básicas y modificarlas para ir acorde al equipo.
- ✓ Reglas de Diseño
  - ➢ Simplicidad.
  - ➢ Escoge una metáfora del sistema.
  - ➢ Usa tarjetas CRC (Clases, Responsabilidades y Colaboración) para decisiones de diseño.
  - ➢ Crea soluciones de pico ("*spike solutions*") para reducir el riesgo. Esto es un programa simple que explore soluciones potenciales a un solo problema.
  - ➢ Ninguna funcionalidad se agrega tempranamente.
  - ➢ Refactoriza cuando y donde sea posible.
- ✓ Reglas de Programación
  - ➢ El cliente siempre está disponible.
  - ➢ El código debe ser escrito bajo estándares acordados.
  - ➢ Programa las pruebas unitarias primero.
  - ➢ Toda la producción de código se hace en parejas.
  - ➢ Solo un par integra código a la vez.
  - ➢ Integra código frecuentemente.
  - ➢ Prepara una computadora dedicada solo para integración.
  - ➢ Haz uso de propiedad colectiva.
- ✓ Reglas de pruebas
  - ➢ Todo el código debe tener pruebas unitarias.
  - ➢ Todo el código debe pasar todas las pruebas unitarias antes de entregarse.
  - ➢ Cuando un *bug* es encontrado, pruebas son creadas.
  - ➢ Pruebas de aceptación se corren frecuentemente y el resultado es publicado

6.1.1.2.1.Plan Utilizado

Para llevar a cabo el proyecto, el grupo Active decidió usar la metodología ágil de Scrum en conjunto con XP. La razón principal para esta decisión es la flexibilidad y simplicidad relativa en comparación a

los demás modelos de ciclo de vida. Adicionalmente, no se aplicará en su totalidad ambos marcos de trabajo, si no, se adaptará para mejor coincidir con el tamaño y alcance del proyecto y con los recursos disponibles. Se tomarán los aspectos gerenciales de Scrum para complementar con las técnicas de desarrollo que ofrece XP.

El modelo de ciclo de vida usado tomará ciertos elementos de Scrum y otros de XP para poder llevar a cabo el proyecto. Dentro de los artefactos de Scrum, se tomó en cuenta el Product Backlog. Este se usará para hacer seguimiento del trabajo efectuado y el trabajo por hacer. Por otro lado, se hará uso de los sprints de scrum. Estos nos permiten organizar el trabajo y dividir las tareas entre diferentes sprints para lograr llevarlas a cabo. Se hará uso de la definición y funcionalidad del scrum master para gerenciar a los integrantes del grupo. Adicionalmente, se hará uso de la idea de entregas frecuentes y funcionales, diseño simple, pruebas frecuentes y la programación por pares descrita en XP para poder llevar a cabo el desarrollo del producto.

El grupo se dividió en varios grupos funcionales que trabajan como scrum teams con los mismos scrum masters. Cada grupo funcional representa un grupo de actividades que se deben llevar a cabo durante el proyecto. A su vez cada grupo funcional es dividido por roles. Los roles son responsables de actividades específicas dentro de las actividades de cada grupo. Sin embargo, los grupos funcionales se ven como un solo *lane* en el *Anexo: Diagrama BPMN de Modelo de Ciclo de vida* y se tratan como un solo scrum team.

El comienzo del proyecto se da con las especificaciones dadas por la Scrum Product Owner, que en este caso será la Ingeniera Anabel Montero Posada. Es ella quien le suministró al grupo Active los requisitos para la aprobación del producto y las especificaciones para el proyecto. Después de esto, se definió el set de roles y responsabilidades necesarias para llevar a cabo el proyecto. Estos roles se definieron dentro de grupos funcionales que describen un conjunto de roles que llevan a cabo procesos a fines para llevar a cabo el proyecto. Cada rol y grupo funcional son descritos en la sección *6.4.2 Organigrama y descripción de roles.* Adicionalmente, se definió los procesos necesarios para la ejecución del proyecto. Esto incluye: análisis de riesgos, selección de modelo de ciclo de vida, selección de herramientas, definición de propósito, alcance y objetivos, definición de reglas de convivencia, etc.

El proyecto se ejecutará mediante varios sprints. Antes de empezar los sprints, se hará una elección de actividades. Cada sprint tendrá una lista con las tareas necesarias y posibles a ejecutar dentro del sprint. Durante el transcurso de los sprints, se harán Daily Scrum Meetings,

donde cada integrante deberá reportar el trabajo efectuado, el trabajo que va a hacer y en caso tal de tener problemas o contratiempos, avisarlos para darles solución. Adicionalmente, se harán múltiples revisiones por el Product Owner del trabajo efectuado para así tener una retroalimentación directa para mejorar la calidad del trabajo.

6.1.2. Análisis de alternativas y Justificación

6.1.2.1. Modelo en Cascada

Este modelo toma los procesos fundamentales de las actividades de especificación, desarrollo, validación, y evolución y las representa como fases separadas tales como especificación de requerimientos, diseño de software, implementación, pruebas, entre otras. (SOMMERVILLE)

Las etapas que constituyen a este modelo son:

1. **Análisis y Diseño de Requerimientos:** Los servicios, restricciones, y objetivos del sistema son establecidos por consulta con los usuarios del sistema. Son definidos en detalle y sirven como especificaciones del sistema.
2. **Diseño de Sistema y de Software:** Establece los requerimientos de hardware y/o software por medio del establecimiento de una arquitectura general del sistema. El diseño de software involucra identificar y describir las abstracciones fundamentales del software y sus relaciones.
3. **Implementación y pruebas de unidades:** El diseño es implementado como un set de programas o unidades de programa. La prueba de unidades involucra la verificación de cada unidad.
4. **Integración y pruebas de sistema:** Las unidades de programa individuales son integradas y puestas a prueba como un sistema completo para asegurar que los requerimientos de software se hayan cumplido. Después de pasar las pruebas, el sistema de software se entrega al cliente.
5. **Operación y mantenimiento:** Se trata de corregir errores que no fueron descubiertos en etapas anteriores, mejorando el sistema a medida que nuevos requerimientos son descubiertos.

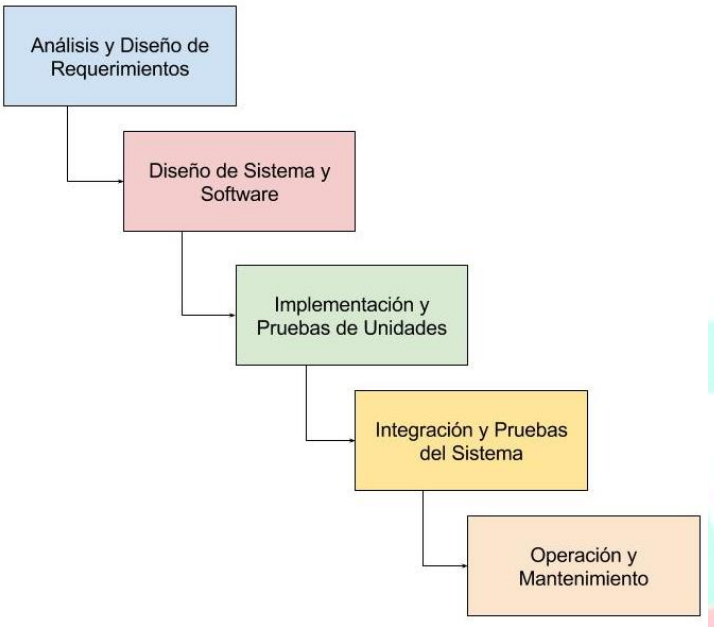

Ilustración 1: Modelo de ciclo de vida cascada

En principio, el resultado de cada fase es uno o más documentos que son aprobados. Como se puede ver en la *Ilustración 1: Modelo de Ciclo de Vida en Cascada*, las fases son en serie y una fase no debe comenzar hasta que la anterior termine. En practica, las fases se sobreponen y alimentan información entre ellas. (SOMMERVILLE)

Este modelo solo se debe usar cuando los requerimientos son comprendidos y no están propensos al cambio radical durante el desarrollo del sistema. (SOMMERVILLE, 2011) Adicionalmente, requiere de una documentación rigurosa y posible repetición de trabajo en cada iteración por lo cual puede incurrir en un mayor volumen de trabajo. Adicionalmente, el cambio en el trascurso del proyecto puede generar trabajo excesivo por lo cual se optó por descartar este modelo.

6.1.2.2. Modelo en Espiral

Boehm, en su paper *"Spiral Development: Experience, Principles, and Refinements"*, lo define inicialmente como una familia de procesos de desarrollo de software caracterizados por la iteración de un set de procesos elementales de desarrollo y de manejo de riesgo apara activamente reducirlo. También lo describe como un generador de modelos de proceso, orientado a riesgos. Como tal, este tiene dos características principales: un acercamiento cíclico para el crecimiento de un sistema mientras se disminuye el riesgo, y un conjunto de hitos de ancla para asegurar que compromiso de los *stakeholders* y llegar a soluciones mutuamente satisfactorias.

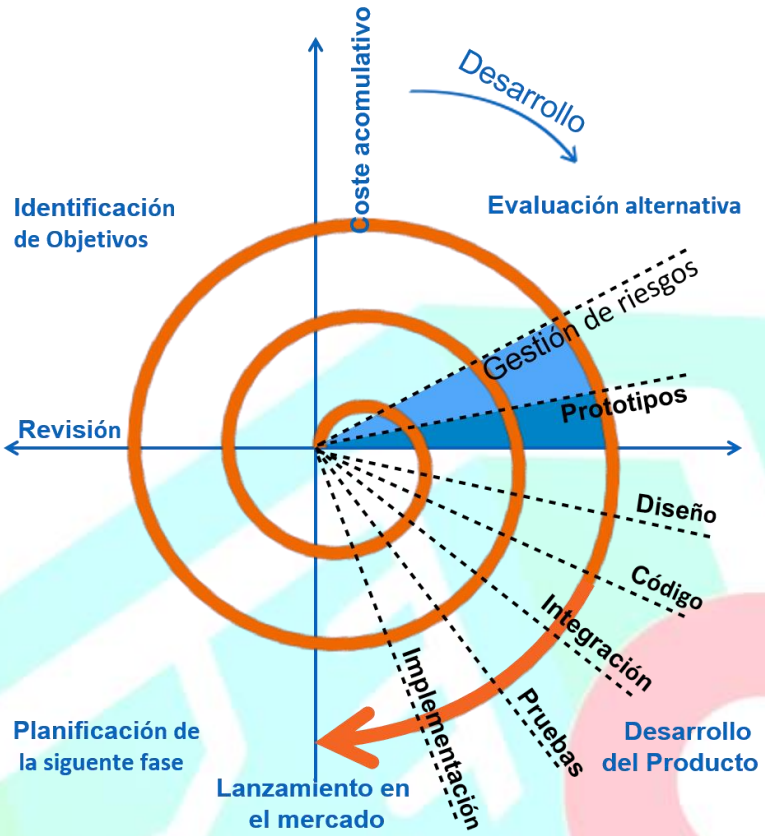

Ilustración 2: Modelo de ciclo de vida en espiral

Como se puede ver en la *Ilustración 2: Modelo de Ciclo de Vida en Espiral,* este está dividido en cuatro cuadrantes. Cada cuadrante es una fase del desarrollo. Con cada iteración se ejecuta cada fase para generar un crecimiento en el sistema y una disminución de riesgos. Primero se ejecuta la fase de identificación de objetivos. Es aquí donde se hace un levantamiento de requerimientos. Posteriormente se hace la fase de evaluación de alternativas que a su vez es un análisis de riesgos donde se sugieren alternativas para la mitigación de riesgo y se implementan. Después se desarrolla el producto. Por último, se evalúa con el cliente el resultado de la iteración y se planifica la siguiente iteración para repetir el proceso otra vez.

Por un lado, este acercamiento otorga un fuerte control de documentación, la posibilidad de agregar funcionalidad en momentos avanzados del desarrollo y una producción temprana de software. Sin embargo, esto con lleva a un alto costo para su uso y requiere de conocimiento y dominio altamente especifico en análisis de riesgos. Lo anterior, genera una dependencia en el análisis de riesgos por lo cual el éxito del proyecto depende de la misma. Adicionalmente no es muy eficiente en proyectos pequeños. (ISQTB)

A causa del tamaño del grupo, cantidad de recursos, falta de experiencia suficiente en el análisis de riesgo y de la rigurosidad de este modelo, se concluye que el modelo no es acorde al proyecto.

6.1.2.3. Modelo en V

El modelo en V es una variación del modelo en cascada que muestra cómo se relacionan las actividades de prueba con el análisis y el diseño. (Ciclo de vida en "V" - INGENIERIA DE SOFTWARE, s.f.) Este modelo promueve el uso de pruebas constante durante el desarrollo. Sin embargo, mantiene la linealidad del modelo en cascada.

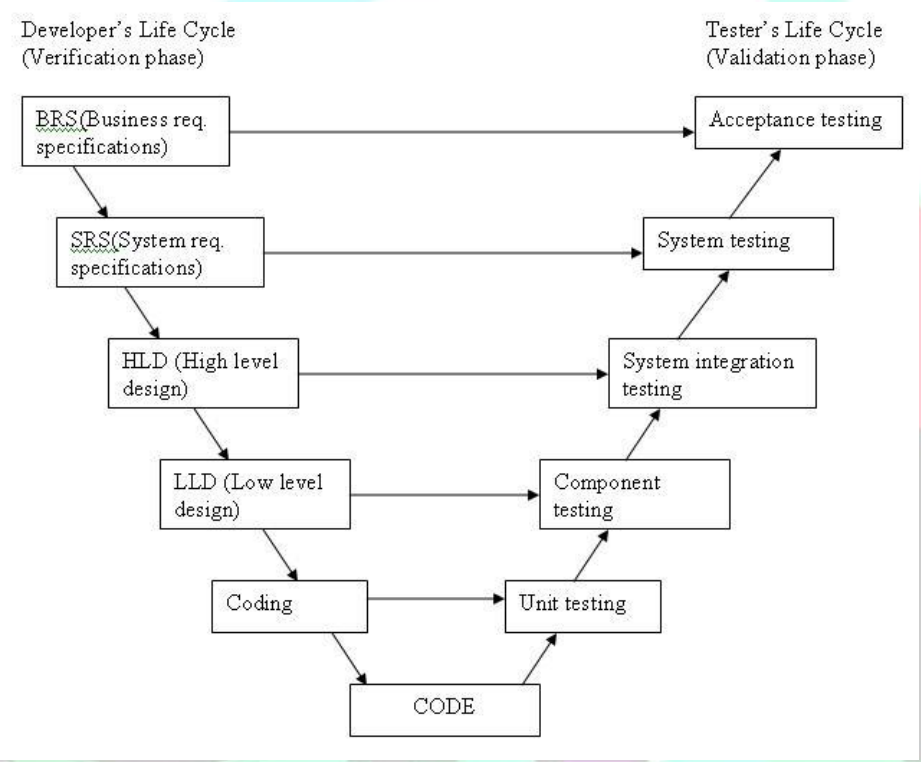

Ilustración 3:Modelo de ciclo de vida en V

Este modelo ofrece múltiples ventajas:

- Simple y fácil de usar
- Actividades de pruebas se ejecutan mucha antes de programar, lo cual ahorra tiempo.
- Rastreo de defectos proactivo
- Evita propagación de defectos
- Funciona en proyectos pequeños donde los requerimientos son fáciles de entender

Por otro lado, presenta las siguientes desventajas:

- Rígido y poco flexible
- No genera prototipos
- Si se efectúan cambios en medio del proceso, entonces los documentos de prueba y de requerimientos deben actualizarse

Este modelo ofrece ventajas en el área de pruebas y se enfoca en ello. Sin embargo, la falta de prototipos significa un alto riesgo para la entrega del producto ya que este podría no satisfacer las necesidades del cliente. Esta desventaja hace que este modelo no sea elegido para su uso en el proyecto.

6.1.2.4. Modelo Diente de Sierra(Sawtooth)

Todos los ciclos de vida previamente mencionados en esta sección trabajan bajo un mismo supuesto que es que los requerimientos no cambiaran drásticamente durante el desarrollo del proyecto. Lo que busca el modelo de diente de sierra es mostrar la percepción del sistema por parte del usuario y por parte del desarrollador por medio de diferentes niveles de abstracción en el tiempo (Software Life Cycle Models, s.f.). Este modelo es en realidad una versión modificado del modelo en V que incluye intersecciones.

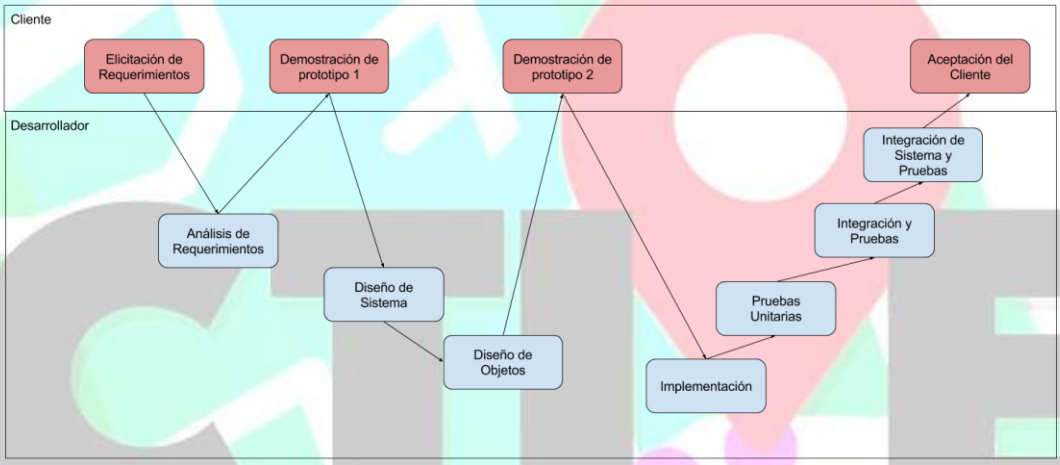

Ilustración 4:Modelo de Diente de Sierra

Al comienzo de este ciclo de vida, los desarrolladores y el cliente se encuentran en el mismo nivel de abstracción. A medida que el proyecto progresa, esto cambia. El usuario se mantiene a nivel de requerimientos mientras que el desarrollador se enfoca en factibilidad. El proceso de desarrollo de software debe garantizar que ambos puntos de vista coincidan al final del proyecto. El modelo lo logro introduciendo actividades adicionales que involucran al cliente desde su nivel de abstracción. Adicionalmente, en este modelo de ciclo de vida, se agrega dos demostraciones de prototipo lo cual ayuda a que el cliente este consiente del progreso y pueda tener una idea del funcionamiento del producto para así dar una retroalimentación.

Dado a que este modelo comparte características con el modelo en cascada, incurre en los mismos problemas del modelo en cascada: La naturaleza lineal del modelo y rigidez puede generar trabajo extra en caso tal que cambios grandes en los requerimientos se den durante el

desarrollo. Adicionalmente, el hecho que las demostraciones se hagan en una etapa temprana puede generar una imagen muy diferente de producto final en el cliente. Es por estas razones que el grupo Active opto por no usar este modelo.

6.1.3. Modelo Diente de Tiburón(Sharktooh)

Este modelo es una versión refinada del modelo de diente de sierra. Agrega un nivel adicional de abstracción al modelo, el del gerente. Esto agrega actividades adicionales por parte del gerente. Se hace una distinción entre dientes "grande" y dientes "pequeños". Los dientes grandes son las demostraciones a clientes y los dientes pequeños son las revisiones gerenciales y demostraciones a gerencia. Entonces las demostraciones podrían ser de vistas diferentes del sistema: Al cliente se le mostraría una demostración orientada a interfaces gráficas y funcionalidad, mientras que al gerente se le demuestra una demostración que muestre la factibilidad de las funcionalidades. Entonces los dientes pequeños son interacciones con el grupo gerencial y los dientes grandes con los clientes.

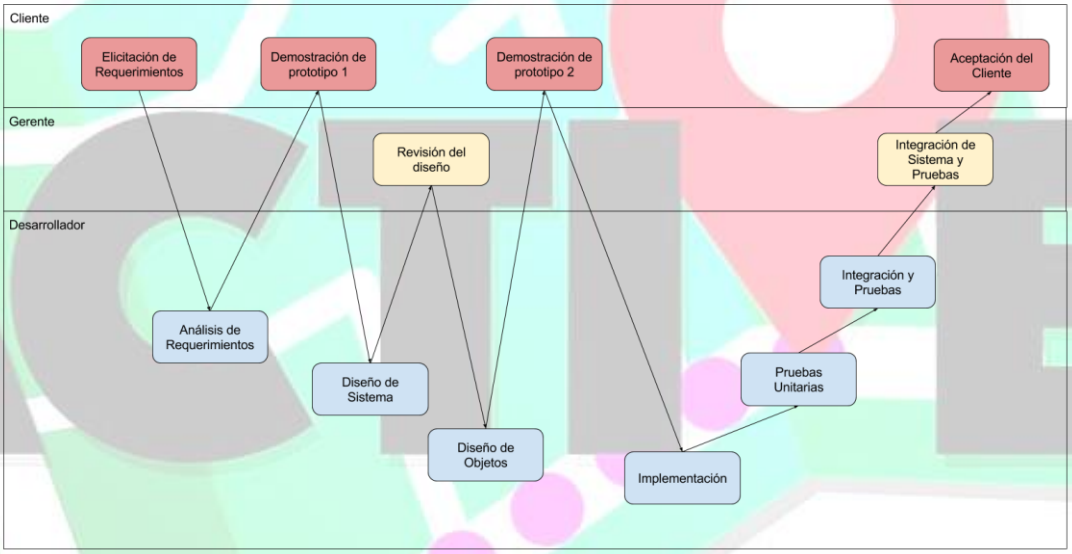

Ilustración 5:Modelo de ciclo de vida Sharktooth

A pesar de ser una expansión al modelo de diente de sierra, no soluciona los problemas que este presenta. Es por esto por lo que se descarta este ciclo de vida.

6.1.4. Modelo RUP

El Proceso Unificado de Rational (RUP, por sus siglas en inglés) es un ejemplo de un modelo de procesos moderno que se deriva del trabajo en el Lenguaje de Modelado Unificado (UML, por sus siglas en ingles), y de los asociados del *Unified Software Development Process* (SOMMERVILLE, 2011). RUP reconoce que los modelos de procesos convencionales presentan una vista única del proceso. Por otro lado, RUP describe tres perspectivas (SOMMERVILLE, 2011):

1. Una perspectiva dinámica, que muestra las fases del modelo en el tiempo
2. Una perspectiva estática, que muestra el proceso de actividades que se establecen
3. Una perspectiva práctica, que sugiere buenas prácticas a ser usadas durante el proceso

RUP es un modelo divido en fases que identifica cuatro fases en el proceso de desarrollo de software. A diferencia del modelo en cascada, que las fases son equivalentes a actividades del proceso, en este modelo las fases están más relacionadas con la empresa que con aspectos técnicos. La *Ilustración 6: Fases de RUP* muestra las fases que son:

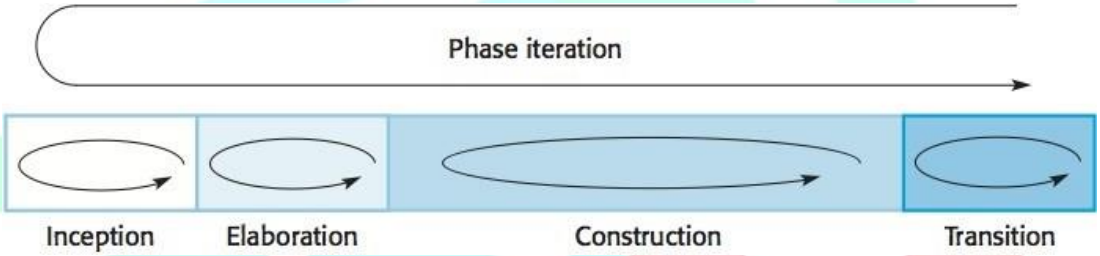

Ilustración 6:Fases de RUP

1. Concepción (*Inception*): El objetivo de esta fase es establecer los casos de negocio para el sistema. Se debe identificar las entidades externas que interactuaran con el sistema y definir las interacciones entre estos. Esta información se usa para definir el valor que el sistema le da al negocio. Si la contribución no es relevante, el proyecto se puede cancelar después de esta fase. (SOMMERVILLE, 2011)

2. Elaboración (*Elaboration*): El objetivo de esta fase es entender el dominio del problema, establecer un marco de trabajo para la arquitectura del sistema, desarrollar el plan de proyecto e identificar los riesgos clave del proyecto. Al final de esta fase, se debe generar un modelo de requerimientos para el sistema. Este puede ser un set de casos de uso, una descripción de la arquitectura y un plan de desarrollo del software. (SOMMERVILLE, 2011)

3. Construcción (*Construction*): Esta fase involucra el diseño, programación y pruebas. Partes del sistema son desarrolladas en paralelo e integradas durante esta fase. Esta fase debe generar un sistema de software funcional y su respectiva documentación, lista para entregar a los clientes. (SOMMERVILLE, 2011)

4. Transición (*Transition*): Esta fase se trata de mover el sistema de la comunidad de desarrollo a la comunidad de usuarios y hacerla funcionar en el ambiente real. (SOMMERVILLE, 2011) Al final de esta

fase se debe contar con un sistema de software documentado, funcional en su ambiente de operación. (SOMMERVILLE, 2011)

Como se mencionó previamente, la perspectiva practica de RUP describe buenas prácticas de ingeniería de software que son recomendadas para uso de desarrollo de sistemas. Las seis practicas recomendadas según Sommerville son:

1. *Desarrolla software iterativamente.* Planea incrementos del sistema basado en prioridades del cliente y desarrolla las características del sistema de alta prioridad en las etapas iniciales del proceso de desarrollo.
2. *Gestiona requerimientos.* Explícitamente documenta los requerimientos del cliente y haz seguimiento de los cambios en estos. Analiza el impacto de los cambios en el sistema antes de aceptarlos.
3. *Has uso de arquitecturas basadas en componentes*. Estructura la arquitectura del sistema en componentes.
4. *Modela visualmente el software.* Usa modelos gráficos de UML para presentas vistas estáticas y dinámicas del software
5. *Verifica la calidad del software*. Asegura que el software cumpla con los estándares de calidad de la organización.
6. *Controla los cambios en el software*. Gestiona los cambios en el software por medio de un sistema de manejo de cambios y configuración de manejo de procedimientos y herramientas.
5. El RUP no es adecuado para todos los tipos de desarrollo (SOMMERVILLE, 2011). Sin embargo, ofrece múltiples ventajas: la naturaleza iterativa le permite que el proceso crezca con cada iteración, divide en etapas el proceso y hace explicito lo que desarrolla en cada etapa. Este modelo se ve adecuado en proyectos medianos y grandes donde se requiere de un seguimiento robusto de procesos. Por otra parte, RUP puede ser un poco abrumador para proyectos pequeños y la vista gerencial de este modelo puede no ser adecuada para proyectos independientes, con grupos pequeños. Por estas razones se optó por no usar RUP para el proyecto

6.2. Lenguajes y Herramientas

Con el objetivo de desarrollar el producto, es necesario utilizar algunas herramientas y lenguajes de programación, documentación y diagramación, además de poder manejar el versionamiento del producto que realizaremos.

6.2.1. Lenguajes

6.2.1.1. HTML 5

Hyper Text Markup Language, o HTML es un lenguaje de marcado diseñado para estructurar y etiquetar los componentes de un archivo de texto; para que este sea después procesado por un navegador web y muestre la información que fue marcada con *tags* de manera estática. HTML no es un lenguaje de programación, pero ofrece elementos que pueden ser manipulados por otros lenguajes con los cuales se modifica el contenido de la página web de acuerdo con las acciones del usuario. (Introduction to HTML, n.d.) Para el proyecto se utilizará la última versión de HTML 5 ya que ofrece más elementos con los cuales se puede interactuar.

6.2.1.2. CSS

Cascading Style Sheet, o CSS es un marco de trabajo y lenguaje de estilo que interactúa directamente con los elementos escritos en un archivo HTML (W3.CSS Home, n.d.), manejando la apariencia de esos elementos, facilitando la organización y lectura de los mismos para los diseñadores; organizando los elementos en módulos que pueden ser manipulados por lenguajes de programación web.

6.2.1.3. JavaScript

Es un lenguaje de programación interpretado, utilizado principalmente para definir los eventos y los resultados de la interacción de un usuario con una página web. JavaScript interactúa con los elementos escritos en HTML y CSS para hacer una página web dinámica e interactiva. JavaScript puede interactuar con otras aplicaciones como base de datos, lo cual se piensa usar en este producto.

6.2.1.4. PHP

Es un lenguaje de programación orientado al desarrollo de scripts para páginas web, que, a diferencia de JavaScript, se ejecuta directamente sobre el servidor en el cual se encuentra implementada la página web y las aplicaciones que lo acompañan. PHP tiene la ventaja de que se puede escribir junto al HTML y es un lenguaje relativamente fácil de aprender, por lo que existen muchas aplicaciones de terceros que son fáciles de implementar y tienen un buen soporte técnico.

6.2.1.5. NoSQL

Modelo de base de datos no relacional que no utiliza el lenguaje SQL como lenguaje de consultas y no posee como estructura base tablas donde se encuentra la información. Se suele utilizar una clave de partición para recuperar valores, conjuntos de columnas o documentos JSON o XML semiestructurados, así como otros documentos que contengan atributos de elementos relacionados. (Amazon Web Services, n.d.). Para el tipo de producto que se va a desarrollar se decidió utilizar este esquema de base de datos ya que facilita el diseño de la arquitectura y la implementación en el producto.

6.2.2. Herramientas

6.2.2.1. Herramientas de desarrollo

a. **Procesadores de texto.**

Para el desarrollo del proyecto es necesario utilizar procesadores de texto ya que es en estos donde se escribe el código del producto y en donde se puede manipular la información de las bases de datos. A diferencia de los editores de texto, los procesadores de texto vienen con herramientas adicionales que facilitan y aceleran la escritura de código, ya que agregan funcionalidades como visualización en vivo del código, verificación de sintaxis, plantillas de segmentos de código común, etc. Se utilizarán diferentes procesadores de texto ya que cada miembro del equipo tiene preferencias y tienen experiencia con estos programas. Los procesadores se usarán para el desarrollo del producto son:

- **Atom:** procesador de texto multiplataforma de código abierto completamente personalizable.

- **Sublime Text:** procesador de texto multiplataforma de uso gratuito, caracterizado por su desempeño, interfaz minimalista, funciones de selección múltiple y soporte de complementos desarrollados por la comunidad. El programa requiere de una licencia para poder utilizarlo a largo plazo, pero esta compra está más orientada a ser una donación a los desarrolladores.

- **Brackets:** procesador de texto orientado a facilitar el desarrollo web con herramientas de visualización en vivo de páginas web y soporte de preprocesadores de código web.

b. **MongoDB**

MongoDB es una base de datos orientada a documentos que utiliza NoSQL con archivos. json y es compatible con JavaScript.

c. **SASS**

SASS es un metalenguaje que es utilizado para optimizar y agilizar el desarrollo de páginas web, ya que asemeja la creación y la manipulación del código CSS de una página, al de un lenguaje de programación más tradicional. El código en SASS pasa por un preprocesador que lo convierte a CSS clásico, lo que no implica que exista la necesidad de instalar librerías o scripts para que se interprete este código en las páginas web.

d. **Softaculus**

Aplicación de servidor que facilita la instalación y administración de varios tipos de aplicaciones, desde bases de datos hasta la creación de un servidor de email o de almacenamiento en la nube con facilidad. Softaculus es gratis y contiene varios scripts que serán utilices para agilizar el desarrollo del producto.

e. Adobe Ilustrator

Aplicación para la creación de gráficos vectoriales. Se usará para la creación de diferentes gráficos tales como el logo de la empresa o los iconos que se utilizaran en el producto.

6.2.2.2. Herramientas de versionamiento

a. Git

Git es un proyecto de código abierto para el control de versiones, desarrollado con la idea de que sea robusto y eficiente, en especial cuando se utiliza con un proyecto grande en el que una gran cantidad de desarrolladores están trabajando al mismo tiempo; gracias a que su diseño permite que cada individuo trabaje localmente y realice solicitudes para que el nuevo código se revise y se anexe al programa principal usando *Branches* o ramas que evitan conflictos y abren posibilidades a la hora de desarrollar.

b. GitHub

Github es una plataforma online de repositorios, principalmente para proyectos que utilicen Git como sistema de versionamiento. Github es la plataforma más popular en la actualidad gracias a su facilidad de uso, a la gran cantidad de usuarios que respaldan y ayudan a mejorar la plataforma. Se escogió Github como la plataforma donde se almacenará el repositorio del proyecto ya que miembros del equipo de desarrollo están familiarizados con esta aplicación, y saben aprovechar sus beneficios como sus opciones para mantener una wiki y una página para cada proyecto.

6.2.2.3. Navegadores web

a. Firefox

Firefox es un navegador web de código abierto multiplataforma, programado pensando en la seguridad del usuario y en la velocidad de navegación. Firefox cuenta con una amplia gama de opciones para desarrolladores web y posee una gran cantidad de complementos que ayudan al usuario.

b. Safari

Safari es el navegador web por defecto para los productos de Apple. Muchos usuarios utilizan computadores de escritorio Mac,

IPhone y IPad; dispositivos que son utilizados por una parte significativa de los usuarios objetivo de la aplicación, razón por la cual se utilizara este navegador para asegurar la compatibilidad del producto y permitir que más usuarios puedan usar la aplicación sin problemas

c. Google Chrome
Navegador web multiplataforma más popular del mundo, desarrollado por Google y cuenta con herramientas que facilitan el desarrollo web. Se caracteriza por ser rápido, intuitivo y cuenta con facilidades para los usuarios que utilicen los demás servicios de Google.

d. Navegador nativo de Android
Se utilizará este navegador para verificar la compatibilidad del producto en diferentes dispositivos Android, ya que muchos de los usuarios potenciales tienen dispositivos Android.

6.2.2.4. Manejo de documentos

a. Google Drive
Servicio multiplataforma gratuito de almacenamiento en la nube creado por Google. Se utilizará este servicio de almacenamiento ya que es fácil de usar, mantiene un registro de los cambios que se han hecho a los archivos que se manipulan y facilita la vista y manipulación de archivos gracias a que permite visualizar diferentes archivos, presentaciones e incluso hojas de cálculo por medio de la suite ofimática online de Google.

b. Microsoft Office
Es una suite ofimática compuesta por varios programas que serán necesarios para el desarrollo del proyecto. Se utilizarán Word 2016 como editor de texto en el cual se realizarán la mayoría de los documentos escritos del proyecto; Excel 2016 para la creación y uso de hojas de cálculo en los que se mantendrán registros ordenados de la diferente información, estado de los trabajos y para la creación de gráficos y PowerPoint 2016 aplicación diseñada para la creación de presentaciones.
Es necesario tener una licencia para poder utilizar los programas de la suite, y gracias a la universidad, todos los integrantes cuentan con una licencia para utilizar estos programas.

6.2.2.5. Herramientas de modelamiento

a. **Bizagi Modeler**

Programa para la creación de diagramas BPMN (Business Process Model and Notation). Puede usarse de forma gratuita. Todos los miembros tienen experiencia utilizando el programa.

b. **Enterprise Architect**

Software de diseño y Análisis usado para para modelar y documentar de acuerdo estándar UML. Se usará principalmente para la creación de los diagramas de casos de uso. Este programa necesita una licencia, pero se utilizará el trial durante el desarrollo del proyecto.

6.2.2.6. Herramientas de soporte

a. **CodeAcademy**

CodeAcademy es una plataforma gratuita de enseñanza en la cual se pueden aprender diferentes lenguajes de programación de forma interactiva y con facilidad. Esta plataforma se utilizará para que integrantes del grupo amplíen sus conocimientos y puedan interactuar más fácilmente con el equipo de desarrollo.

b. **W3Schools**

W3Schools es una página web que contiene referencias para diferentes lenguajes de desarrollo web como HTML, CSS, JavaScript, PHP, SQL, W3.CSS y Bootstrap; y a su vez ofrece cursos y tutoriales para el aprendizaje de esto lenguajes. Los integrantes del grupo utilizaran esta página para documentarse y complementar aquellas áreas en las que necesiten ampliar sus conocimientos. Se puede acceder a su página web por medio de la siguiente dirección "*https://www.w3schools.com/*".

6.2.3. Análisis de alternativas y justificación

6.2.3.1. Justificación Lenguajes

a. **Python**

Poderoso lenguaje de programación interpretado de código abierto, orientado a la escritura de scripts y con un soporte de una gran comunidad. Python cuenta con una gran cantidad de librerías y aplicaciones que son compatibles con varios de los programas escogidos; pero se decidió dejarlo como lenguaje alternativo ya que pocos de los integrantes del grupo están familiarizados con el lenguaje y sus aplicaciones.

b. **Ruby on rails**

Framework de código abierto para el desarrollo web escrito en Ruby. Ruby on rails se usa en especial para las transacciones y solicitudes a bases de datos, pero se tiene como opción de respaldo en caso de que PHP y JavaScript no sean suficientes para el desarrollo y el correcto funcionamiento de la aplicación.

6.2.3.2. Justificación Herramientas

1) Herramientas de desarrollo

a) Leaf paf

Editor de texto de código abierto, implementado como editor por defecto en varias distribuciones de Linux; razón por la cual se tendrá en cuenta como alternativa cuando se esté trabajando en este sistema operativo.

b) cPANEL

Panel de control de páginas web que facilita la instalación y el manejo de sitios y aplicaciones que se deseen implementar en un servidor.

c) MySQL

MySQL es sistema de manejo de bases de datos relacionales de código abierto. MySQL fue escrito originalmente en C y C++, pero interactúa fácilmente con otros lenguajes de programación, lo que lo hace una opción atractiva para su uso con lenguajes de programación web como PHP para generar bases de datos.

2) Herramientas de versionamiento

a) Bitbucket

BitBucket es una plataforma online de repositorios y es la principal competencia de GitHub. Esta plataforma está más orientada al manejo de versiones de proyectos grandes que utilicen git o mercurial. Se decidió esta plataforma como alternativa ya que posee los elementos básicos para la continuación del proyecto en caso de que algo suceda con nuestro repositorio principal.

3) Manejo de documentos

a) OpenOffice

Suite ofimática de código abierto compatible con los formatos que utilizan los programas de la suite de Microsoft office. Se considera para ser una aplicación de respaldo ya que tiende a ser más limitada que la suite de office, pero en caso de que sea necesario se puede continuar el trabajo en este conjunto de programas sin que existan problemas a la hora de volver a usar la opción principal.

b) Dropbox

Servicio gratuito con opción paga de almacenamiento en la nube. Se tendrá en cuenta como alternativa, pero tiene limitaciones como la necesidad de crear una cuenta, el limitado espacio de almacenamiento que se le da a las cuentas gratis y no tiene equivalente a la interacción con archivos que provee Google Drive.

4) Herramientas de modelamiento

a) Camunda

Aplicación para el modelado y la creación de diagramas BPMN más parecida a Bizagi Modeler.

b) StarUML

Programa para el modelado de diagramas y elementos de acuerdo con el estándar UML. Se tiene como alternativa ya que cuenta con una interfaz parecida a la de Enterprise Architect por lo que sería fácil adaptarse en caso de que se necesite. Este programa necesita una licencia, pero se utilizará el trial durante el desarrollo del proyecto.

6.3. Plan de aceptación del producto

El plan que se verá a continuación tiene como objetivo asentar las normas por parte del Active y el cliente final.

6.3.1. Criterios de aceptación del producto

Para que el cliente final este satisfecho con el producto final, se tendrá como propósito que cada norma que se establezca deberá ser objetiva, medible y alcanzable. Todo esto se tomará de acuerdo con las especificaciones del Product Owner.

6.3.1.1. Criterios de aceptación del producto por el Product Owner

a. **Software de calidad:** El cliente acudió a Active para un producto que se le proporcionara con calidad y que cumpla con los criterios de la ISO 9126 el cual define estándares de calidad para software. De acuerdo con lo anterior se tendrá en cuenta las siguientes pautas:

- **Funcional:** La aplicación web tiene que tener exactitud en cuanto a operaciones y funcionalidades que se realicen. Este producto debe ser seguro y eficaz para la comunidad educativa.

- **Confiable:** La aplicación web debe ser flexible (rutas inadecuadas, discapacidad cliente, tiempos) y hacer una recuperación y retroalimentación para solucionar el problema de una manera rápida y eficiente.
- **Eficiente:** Esta aplicación debe de ser efectiva en tiempos de respuesta para generar soluciones en cuanto a rutas adecuadas, discapacidad del cliente final, tiempos de trayecto. Así se generará en el usuario final seguridad y conformidad.
- **Fácil de usar:** El producto debe ser visualmente agradable para el cliente final. Además de que sea de fácil comprensión y que a la hora de generar una ruta o una ubicación no sea confusa para el usuario.
- **Mantenible:** Este producto debe ser adaptable a el cambio que pueda tener el campus universitario. Y a su vez que permita un crecimiento para el análisis de defectos que se puedan presentar.

b. **Documentación:** La documentación del producto debe ser completa y clara, además debe de ser entendible. Cada sección debe cumplir los objetivos que fueron establecidos anteriormente.

c. **Características de criterios de aceptación:**
   - **No ambigüedad:** Deben de poder ser interpretados de la misma forma por cualquier persona que los lea.
   - **Atomicidad:** Sólo deben de ser evaluados según los estados descritos en sección *8.2.2 Control de progreso*.
   - **Verificables:** Deben de ser escritos como pruebas concretas para que el cliente los evalúe de manera rápida.
   - **Completos:** El conjunto de los criterios de aceptación rige el estado y el progreso de los documentos, para darse por finalizado un apartado tiene que cumplir con las condiciones descritas anteriormente

6.3.1.2. Criterio de aceptación del producto por el grupo Active

Para Los criterios de trabajo que manejara Active para la aceptación tanto del proyecto como del producto se podrán verificar en las secciones *10.1 Ambiente de trabajo* en el apartado "Reglas de trabajo". Como requisitos mínimos para la entrega de una sección esta debe poseer:

- **Coherencia:** El contenido presentado en el documento debe ser acorde a el propósito que se quiere llegar. si la sección no contiene una coherencia adecuada se rechazará desacuerdo a lo establecido en la sección *8.2.2 Control de progreso.*

- **Objetividad:** el contenido de cada sección debe contener un objetivo, esto con el propósito de que el usuario final entienda el propósito del apartado o sección.
- **Completitud:** Cada sección debe estar completamente terminada con los soportes necesarios.
- **Estructurado:** El documento debe presentar un orden adecuado para una fácil lectura por parte del usuario final. Este documento debe estar separado por apartados o secciones para que haya un orden y así el usuario final podrá encontrar de forma rápida el contenido.
- **Relevancia:** La información presentada debe ser relevante y veraz para el apartado correspondiente, esto con el fin de suministrar al usuario información relevante.

6.3.1.3. Herramientas y criterios de evaluación

Utilizando la metodología SCRUM como base del proyecto. Se generarán diferentes roles con los que se pretende crear filtros para un mejor control de proceso. Esto hace que Active sea más eficiente y eficaz.

En la metodología SCRUM se tienen como relevancia la existencia de las entregas, por lo cual los trabajos realizados por cada miembro del equipo Active serán evaluados bajo las métricas expuestas anterior mente y con las especificaciones de la sección *10.5 Control de calidad*. Estos trabajos serán evaluados con las métricas mencionadas anteriormente.

Para las entregas se tendrán líderes en las diferentes áreas que maneja active. Como lo son el líder de diseño y programación que verificara que la información entregada por su compañero este correcta y apruebe los estándares solicitados anteriormente.

Las técnicas que utilizaremos como mencione anteriormente fueron implementadas durante el desarrollo de las entregas (sprint) con el fin de corregir errores, tomando como modelo la metodología de cadena critica. Esta es una metodología de gestión de proyectos basada en la teoría de las restricciones (TOC), pensada para maximizar el avance del proyecto, teniendo en consideración que los proyectos están sometidos a incertidumbre y una serie de limitaciones.

La metodología de cadena critica se basa en aplicar estos tres principios:

- **Identificación de las restricciones:** normalmente esta restricción se muestra como el conjunto de tareas, que bien por limitaciones temporales o de recursos, definen la duración mínima del

proyecto. Esto se llama la cadena critica, o camino critico cuando no se tiene en cuenta la restricción introducida por los recursos.

- **Asignar prioridad a las tareas dentro la cadena critica:** los esfuerzos de director de proyecto deben centrarse en la ejecución de aquellas tareas que formen parte de la cadena critica, ya estas determinaran la finalización del proyecto.
- **Subordinar:** Asignar un encargado para las tareas de la cadena crítica

6.4. Organización del proyecto y comunicación

6.4.1. Interfaces externas

Para una realización optima, eficiente y eficaz de este aplicativo web será necesario tener diferentes servicios. El más importante para el desarrollo de este proyecto será el servidor ya que en ese va a contener el aplicativo web. Adicionalmente se tendrán analizadores y diseñadores para la realización del proyecto, estos cargos ya están bajo el equipo Active. Esto con el fin de generar un buen diseño y seguir requisitos de control de calidad para que todo sea realizado satisfactoriamente de acuerdo con las especificaciones dadas por el scrum owner.

A continuación, se mostrará la descripción de las interfaces externas que se utilizaran en la realización del proyecto.

Nombre	Descripción	Responsabilidad	Medio de comunicación
Anabel Montero	Cliente	Responsable de comunicar todos los requisitos necesarios para la realización del aplicativo web	**Correo:** anmontero @javeriana.edu.co **Teléfono:** 3208320 Ext 5993
1and1	Proveedor de servidor	Empresa escogida. Para el servicio de servidor. En este se tendrá la aplicación	**Página web:** https://www.1and1.es/servidores **Centro de ayuda y contacto:** https://ayuda.1and1.es/servidores-c66735
Arvixe	Proveedor de dominio de internet	Escogido para el servicio de dominio de la aplicación web	**Página web:** https://www.arvixe.com/ **support:** https://support.arvixe.com/

Tabla 3 Interfaces externas

6.4.2. Organigrama y descripción de roles

La metodología Scrum distingue 3 roles diferentes. Cada rol tiene responsabilidades definidas y podrán terminar el proyecto si y solo si cumplen con estas responsabilidades, interactúan entre si y trabajan juntos. Estos tres roles son Scrum Master, Scrum Team y Scrum Product Owner, que fueron previamente definidos en la sección 6.*1 Modelo de Ciclo de Vida.*

El rol de *Scrum master* lo cumplen dos integrantes del grupo: Natalia Otero y Juan Miguel Gómez. Su responsabilidad es la de asegurar que el proyecto se lleve a cabo de manera adecuada y siguiendo el *framework* de Scrum.

En el caso del grupo Active, por el número reducido de integrantes, armar grupos únicos e individuales para cada actividad del proyecto no es viable. Es por esto, por lo que el grupo se divide en tres equipos funcionales, en los cuales un integrante puede hacer parte de varios al mismo tiempo. Cada equipo funcional se trabaja como un Scrum team y todos comparten Scrum master. Adicionalmente, cada equipo cuenta con roles específicos que permite una mayor claridad y mejor organización en cuanto a responsabilidades.

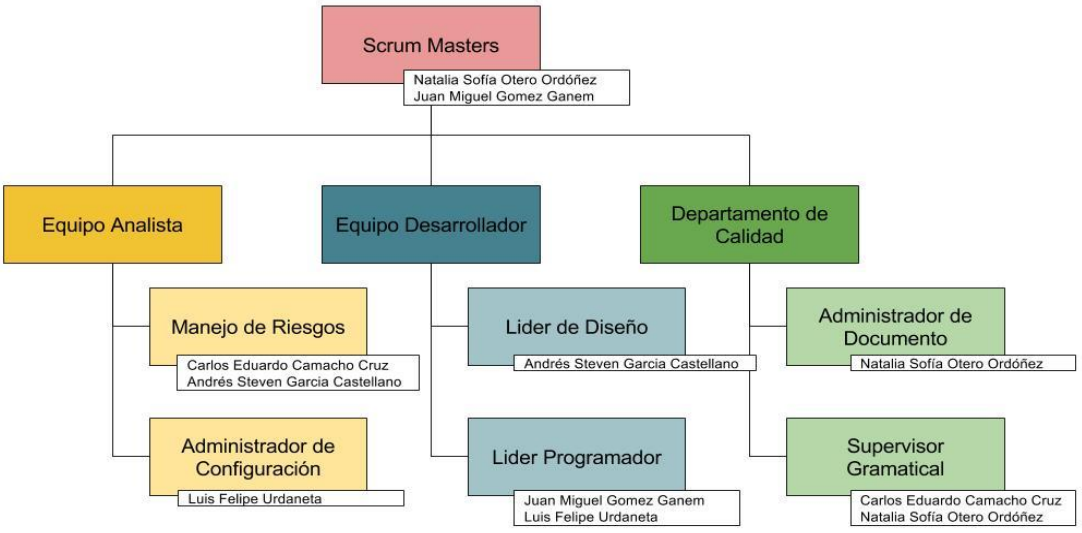

Ilustración 7:Organigrama

Los roles específicos asignados a los equipos funcionales se pueden evidenciar en la *Ilustración 4: Organigrama.* Estos son:

- Scrum Masters: Deben liderar el proyecto y asegurar que todos los integrantes sean eficientes y solucionar de la mejor manera posible cualquier inconveniente que impida que los demás integrantes cumplan con sus responsabilidades. Como fue previamente mencionado, este rol será desarrollado por Natalia Otero y Juan Miguel Gómez.

- **Manejo de Riesgos:** Este rol es responsable de hacer un análisis de riesgos. Esto incluye identificar los riesgos posibles, determinar el impacto que podrían tener con respecto al proyecto, hacer monitoreo de los cambios en los riesgos a medida que se desarrolla el proyecto y hacer planes de respuesta para los riesgos. Lo anterior se verá en detalle en la sección *10.2 Análisis y administración de riesgos* y para este rol se asignó dos integrantes: Carlos Camacho y Andrés García

- **Administración de Configuración:** El administrador de configuración debe encargarse de la infraestructura del manejo de configuración general y del ambiente del equipo de desarrollo. El administrador de configuración debe asegurarse que los desarrolladores tengan los espacios de trabajo adecuados para el desarrollo del producto. Adicionalmente debe escribir el plan de manejo de configuración. Este cargo lo ocupa Luis Felipe Urdaneta

- **Líder de Diseño:** Actuará como arquitecto de software. Principalmente, estará encargado de crear la arquitectura correcta para resolver el problema. Adicionalmente, debe asegurarse de comunicar la arquitectura a todos los miembros del grupo, y garantizar entendimiento de todos los involucrados. (Software Engineering Institute). Para este rol se asignó a Andrés Steven García Castellano

- **Líder Programador:** Es el/los encargados del desarrollo del producto. Esto incluye todo lo referente a programación, herramientas requeridas para el desarrollo del producto, interfaces externas necesarias para el producto, lenguajes de programación que se usaran para la creación del producto, la lógica del producto y las interfaces de usuario. Este rol se llevará a cabo por los integrantes Juan Miguel Gómez Ganem y Luis Felipe Urdaneta.

- **Administrador de Documento:** Es el responsable del control, seguridad, accesibilidad y de lo oportuno de los documentos organizacionales que puedan ser usados por uno o más empleados, como lo son políticas, procedimientos, guías, formularios, plantillas y materiales de entrenamiento.

Dentro de sus actividades esenciales podemos distinguir: desarrollar un plan de gerencia de documentos y actualizarlo si es necesario, manejar documentación organizacional a través de su ciclo de vida, preservar los documentos organizacionales y su sistema de manejo, identificar e investigar

la necesidad de diferentes documentos, asegurar que los documentos pasen por un proceso de revisión y aprobación, asegurar la confiabilidad, accesibilidad y disponibilidad de los documentos, asignar privilegios y planificar y llevar a cabo reuniones y presentaciones relacionadas con el manejo de documentos (Bizmanualz). Para llevar a cabo este rol se designó a la integrante Natalia Otero.

Supervisor Gramatical: Los supervisores gramaticales se encargarán de que todos los documentos escritos estén adecuadamente redactados. Esto implica que tengan sintaxis correcta, buena ortografía, y léxico apropiado para el documento. Este cargo lo ocupan Natalia Otero y Carlos Camacho.

7. Administración del proyecto

7.1. Métodos y herramientas de estimación

En esta sección se darán a conocer las estimaciones que realizo el equipo Active para el progreso del proyecto. Todas estas estimaciones fueron realizadas en presencia de todos los miembros del equipo. Esto con el fin de que los integrantes del equipo tengan en cuenta los elementos necesarios para la realización y finalización de este proyecto.

7.1.1. Estimaciones en la planeación del proyecto

a. Tiempo

Para la estimación de tiempo tomamos como base las fechas que representa cada entrega (Sprint), estas fechas podrán revisarse en el diagrama de Gantt. Con esto cada entrega contiene un número de actividades las cuales se deben cumplir. Con esto cada miembro del grupo cumplirá con el tiempo asignado (generalmente 8 días). En ocasiones habrá tareas que se deberán realizar en equipo. Para estas se destinará un día para la reunión y realización de la actividad.

b. Costos

Los costos que se tuvieron en cuenta para la realización del proyecto fue el del servidor de 1and1. El hosting no nos genera un precio ya que el equipo Active ya cuenta con uno. Con esto generamos un ahorro y podemos destinar este dinero a cualquier gasto que se le pueda presentar a Active.

c. Esfuerzo

Para este proyecto cada actividad tendrá un índice de esfuerzo. Algunas actividades tendrán mucho más valor ya que la carga de trabajo es mucho más alta. Al final todos los miembros del equipo deben tener un porcentaje similar de trabajo para que el esfuerzo sea equitativo por cada persona del grupo.

d. Recurso

En cuanto a los recursos se tomó el hosting que tiene el grupo ya que es necesario para el desarrollo del proyecto. Además, se tendrán los conocimientos en diferentes áreas de cada miembro del equipo active.

7.1.2. Estrategias de estimación

Se tomarán varias estrategias de estimación para este proyecto. Esto con el fin de identificar cada uno de los costes que tendrá el proyecto. Las siguientes estrategias que se presentan a continuación se tendrán en cuenta para la realización del proyecto.

- **El Juicio de Expertos:** consiste en preguntar y guiarnos por los conocimientos y experiencias de personas que han realizado un trabajo igual o semejante al cual le estamos determinando el coste. Muchas veces se "abusa" de esta técnica, debido a la falta de datos cuantitativos de proyectos anteriores, y la ausencia de una buena gestión del Conocimiento.

- **Estimación por analogía**: referenciada también como Top-Down, porque se cuestiona el coste desde lo más general a lo más específico. Es utilizada cuando se cuenta con experiencia en proyectos anteriores, análogos o similares, que pueden servir de referencia. Es una técnica menos costosa y más rápida, pero tiene como desventaja que es menos exacta y que se necesita de experiencia y documentación.

- **Análisis de la reserva**: Este análisis nos permite, basado en la incertidumbre, estimar una cantidad adicional al coste que hemos identificado, generando lo que se conoce como "reserva de contingencia". Se debe utilizar cuando a la actividad en la cual recae el coste le ha sido identificado algún riesgo. Para calcular esta reserva utilizaremos lo que se denomina Análisis del Valor Monetario Esperado (VME), para lo cual, es necesario que el riesgo haya sido valorado de manera cuantitativa, es decir, que su impacto haya sido estimado en términos de dinero y/o tiempo.

En la siguiente tabla se presenta las aproximaciones en horas de las actividades del SPM:

Actividad	Tiempo (horas)	Costo	Esfuerzo	Recursos
Historial de cambios	6	Ninguno	2	Office
Prefacio	4	Ninguno	2	Office
Tabla de contenidos	2	Ninguno	1	Office

Lista de figuras	2	Ninguno	1	Office
Lista de tablas	2	Ninguno	1	Office
Vista general del proyecto	20	Ninguno	5	Office
Contexto del proyecto	26	Ninguno	5	Office
Administración del proyecto	24	Ninguno	5	Office
Monitoreo y control del proyecto	16	Ninguno	4	Office
Entrega de producto	8	Ninguno	3	Office
Procesos de soporte	20	Ninguno	5	Office
Anexos	4	Ninguno	2	Office, bizagi modeler
Referencias	2	Ninguno	1	Office
Otros documentos	20	Ninguno	5	Office, Enterprise Architec

Costos: En este avance del proyecto no se evidencian costos.
Esfuerzo: El esfuerzo que tiene cada sección se medirá de uno (1) a cinco (5), tomando en cuenta de que 1 generará poco esfuerzo y 5 mucho esfuerzo.

7.2. Inicio del proyecto

7.2.1. Entrenamiento del personal

Para el desarrollo del proyecto se definió que cada integrante del grupo "Active", debe contar mínimamente con los conocimientos básicos de las asignaturas necesarias para cursar "Ingeniería de Software", como por ejemplo, elementos básicos de programación, diseño de software, en este se encuentran los conocimientos para la elaboración de los diagramas de casos de uso, BPMN y sobre el levantamiento de requerimientos; como parte de los conocimientos básicos considerados por el grupo es la redacción y exposición de ideas en un trabajo escrito, ya que, como se puede evidenciar, es una parte bastante importante en el desarrollo del proyecto.

Teniendo conocimiento de esto, todos los integrantes del equipo serán responsables de repasar e investigar sobre lo anteriormente mencionado, con el fin de que solamente se organicen jornadas de transferencia de conocimiento para dar a conocer y aprender nuevas herramientas de control y desarrollo que se requieran para la elaboración del aplicativo CampusFinder. Estas jornadas serán guiadas y lideradas por el miembro del equipo que tenga mayor conocimiento de estas herramientas, fomentando tiempos de práctica

en los que todo el equipo aprende y refuerza conocimientos y habilidades que serán de gran importancia para que el producto salga de la mejor calidad.

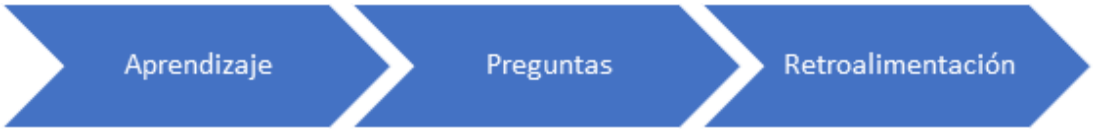

Ilustración 8:Etapas entrenamiento del personal

Para iniciar el plan de transferencia del conocimiento en cuanto a las herramientas de desarrollo (lenguajes de programación) los integrantes interesados en comprender o capacitarse en dichas herramientas dirigirán al equipo de desarrollo quienes, con la validación de gerencia, organizarán un programa de entrenamiento el cual será puesto en acción según lo organizado con todo el equipo. Este entrenamiento ocurrirá en tres etapas de acuerdo con la Ilustración 5: Etapas entrenamiento del personal: Aprendizaje, preguntas y, retroalimentación. La primera consistirá en una sesión tipo clase en donde los encargados expondrán los tópicos más importantes y los que son necesarios para el proyecto; la segunda etapa consiste en una sesión de preguntas por parte de los integrantes buscando aclarar dudas para ser de ayuda al momento de utilizar en forma estas herramientas. Finalmente, la etapa de retroalimentación consiste en que en el Daily Scrum más cercano se hagan comentarios, recomendaciones y correcciones de las actividades propuestas después de dos sesiones, con el fin de corregir los errores, verificar conocimientos y mejorar.

7.2.2. Infraestructura

En esta sección se describirán las herramientas que en conjunto con los paquetes de software descritos en la sección 6.2 serán necesarios para el desarrollo del producto.

- **Hardware:**

Dentro de los activos de hardware encontramos que se necesita un servidor web sobre el cual se implemente el back-end del producto, computadores; ya que son indispensables para que se desarrolle el producto y celulares como medio de comunicación. los integrantes del equipo cuentan con los siguientes computadores como plataformas de desarrollo principales para completar cada una de las tareas a realizar:

Marca	Modelo	Sistemas operativos	Procesador	RAM	Propietario
ASUS	A55VD	Windows 8.1 pro 64 bits	Intel Core i5 - 3230M	8GB	Luis Urdaneta
ASUS	Asus UX305F	Windows 10 64 bits.	Intel Core M5Y10	8GB	Natalia Otero

ASUS	ZenBook UX3031	Windows 10 64 bits.	Intel Core i5 - 6200u	8GB	Juan Gómez
ASUS	X455L	Windows 10 64 bits	Intel Core i5 - 5200U	8GB	Andrés García
ASUS	N56JR	Windows 10 64 bits.	Intel Core i7 – 4800HQ	8 GB	Carlos Camacho

Ilustración 9:Especificaciones equipos de Active

Cada integrante es responsable de sus herramientas de trabajo y de los periféricos (Mouse, audífonos, cables, etc.) que utilicen. Cada integrante puede trabajar en diferentes dispositivos siempre y cuando mantengan una copia de su trabajo en los computadores que se especificaron en la *Ilustración 9 Especificaciones equipos de Active* y en la carpeta de Google Drive que se le asigne.

- Servidores y hosting web: La aplicación requiere de un servidor para proveer todos los servicios que el producto planea ofrecer. El equipo de desarrollo se encargará del manejo del servidor y de interactuar con la empresa proveedora (Arvixe), quienes proveerán servicio técnico en caso de que se requiera. Ya se cuenta con el servicio de hosting y los miembros del equipo de desarrollo ya están capacitados para administrar e implementar elementos de la aplicación.

- Espacios físicos: Los espacios definidos para las reuniones y trabajos en grupo son los salones donde se tiene clase presencial, el espacio de estudio del 4 piso de la facultad de ingeniería, las salas de estudio de la biblioteca (estas salas deben pedirse con anterioridad por correo electrónico) y los hogares de los integrantes del grupo. Para el desarrollo del trabajo individual cada integrante es libre de escoger donde trabajar, siempre y cuando no comprometa las tareas asignadas. Juan Miguel Gómez será el encargado solicitar las salas y los diferentes espacios de trabajo cuando sea necesario.

- Software: La mayoría del software descrito en la *sección 6.2 lenguajes y herramientas* no tiene dificultades para su adquisición, aunque algunos de ellos tienen un límite de tiempo en los que se pueden utilizar sin necesidad de adquirir una licencia. Todos los integrantes instalaran las herramientas de modelado, una suite ofimática, 2 de los navegadores web y si lo desean la interfaz gráfica de GitHub. Los desarrolladores instalaran todos los elementos necesarios para programar y

manejar el producto e indicaran a los demás miembros que programas necesitaran instalar si lo requieren.

- ✓ **Google Drive:** El scrum master será el encargado de administrar los archivos que se suban a las carpetas e indicará a los demás integrantes como se manejará el almacenamiento en la nube. Juan Miguel Gómez y Natalia Sofia Otero diseñaran y mantendrán consistente el manejo de archivos y la documentación que se suba a la plataforma.

- ✓ **Repositorio en GitHub:** De acuerdo a lo establecido anteriormente, se utilizará la plataforma de GitHub para el manejo del versionamiento del producto. El encargado del repositorio y mantenimiento será Luis Felipe Urdaneta, quien revisará y será el responsable de crear una rama para cada integrante, revisar cada commit y manejar el master del repositorio.

- ✓ **WhatsApp Messenger**: Como herramienta de comunicación principal todos los integrantes utilizaran WhatsApp Messenger, que no tendrá un encargado especifico ya que todos los miembros seguirán las siguientes reglas para asegurar que la aplicación tiene un uso específico:

    1. Todos los integrantes tienen claro el objetivo por el que se creó del grupo de WhatsApp.
    2. Todos los mensajes que se envíen tendrán que ser relevantes para el grupo, evitando tocar temas personales, mensajes mal intencionados que atenten contra las creencias de otros miembros del grupo o mensajes que inciten a discusiones sin razón.
    3. Los mensajes deben ser concisos y preferiblemente deben estar relacionados al tema del que se esté hablando.
    4. El uso de imágenes está permitido mientras sean adecuados.
    5. Evitar el uso de emoticones, emojis y kaomojis a menos de que sean pertinentes.
    6. Si dos miembros utilizan el chat para interactuar directamente, se les indicara a que se

comuniquen por privado y no utilicen el chat principal.

7. No utilizar muchos mensajes cortos en serie.

Si alguno de los miembros o varios miembros del grupo violan repetidamente las reglas descritas se procederá a realizar el proceso de penalización de acuerdo con lo establecido anteriormente.

7.3. Planes de trabajo del proyecto

7.3.1. Descomposición de actividades

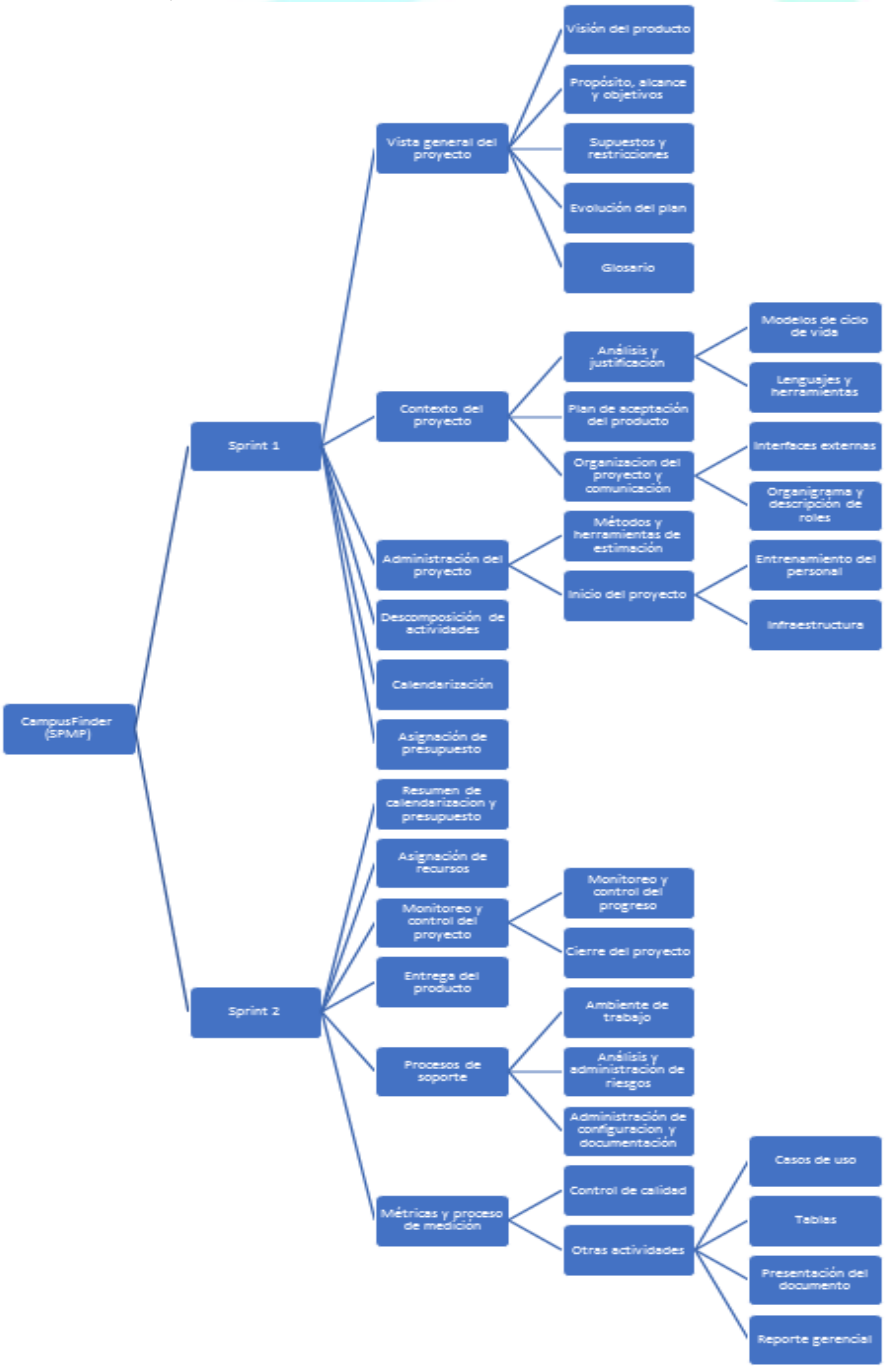

Ilustración 10: Descomposición de actividades

La Ilustración 6 muestra la división de las actividades a elaborar en cada sprint para el desarrollo del SPMP.

Según la metodología SCRUM, el tiempo de proyecto debe ser dividido por Sprints donde en cada uno se desarrollan actividades específicas y se realiza la respectiva retroalimentación. Esta entrega se tiene organizada para terminarla en 2 sprints. Cada uno está dividido en actividades en la *Ilustración 6: Descomposición de actividades* SPMP. Adicionalmente, se planificaron 5 sprints más para la elaboración de la segunda entrega SRS y de la tercera entrega SDD; siendo un total de 8 sprints a lo largo de todo el desarrollo del proyecto.
En la sección *7.3.2 Calendarización*, se podrá encontrar el tiempo y los responsables asignados a cada una de las actividades mencionadas. Para esto, se elaboró un diagrama Gantt el cual evidencia el orden de desarrollo de las actividades y los responsables.

7.3.2. Calendarización

La calendarización fue desarrollada mediante un diagrama de Gantt el cual se encuentra en el *Anexo: Diagrama de Gantt,* las convenciones correspondientes para este diagrama son las siguientes:

Convenciones para estado de la actividad

- En desarrollo: Actividad que hasta el momento de la revisión no ha sido terminada o que la fecha planteada para su finalización es posterior a la fecha actual.
- Terminada: Actividad que finalizó su ejecución, el producto ya cumplió con los estándares de calidad expuestos en la sección *10.5 Control de calidad* y la sección *6.3 Plan de aceptación del producto* y por lo tanto es aceptado y agregado al documento final.
- Pendiente: Actividad que hasta el momento no ha iniciado su ejecución.

Convenciones de eventos

- Reunión del equipo: Momento que el grupo Active se reúne para tocar temas acerca del documento y del producto.
- Informe de gestión: Documento donde los Scrum Masters explican la metodología de trabajo que se manejó durante el intervalo del tiempo al que se refiere y los resultados que se obtuvo por parte del equipo.
- Entrega de producto: Entrega final del documento al Product Owner. Por lo tanto, ya ha sido elaborado, corregido y revisado para tal fin.
- Hito: Inicio y final de cada uno de los Sprint que se realizan durante las tres entregas.

7.3.3. Asignación de recursos

Con el objetivo de desarrollar un producto de calidad es necesario especificar como se planean asignar los recursos para que todos los integrantes del grupo Active puedan desarrollar un producto que satisfaga las necesidades del

cliente de la manera más satisfactoria posible. A parte de los elementos que componen a la infraestructura del proyecto, es importante recalcar que el recurso más importante y quizás el más difícil de manejar es el tiempo de los integrantes.

Estimar el tiempo requerido para la realización de cada tarea no es fácil y tal como se muestra en la *sección 7.1.1 Métodos y herramientas de estimación*, el desarrollo del proyecto se planeará usando un Diagrama Gantt (ver anexos), con el cual, se facilita la asignación y la visualización del tiempo de cada uno de los integrantes, ya que cada uno de los miembros tiene asignado una serie de tareas y responsabilidades que tienen que cumplir con los estándares de calidad reflejados en la sección *10.5 Control de calidad.*

Para la creación del proyecto se declararon en la *sección 6.2 Lenguajes y herramientas* todos los recursos de Hardware y software que son necesarios para desarrollar el producto, y en la sección 7.2.2 Infraestructura se explica cómo se manejarán y que integrantes son responsables de estos recursos.

Aunque se tiene claridad acerca de que recursos se tienen, cuales se necesitan, como y quienes los manejan; hay que recalcar que al momento de la entrega de este documento solo estamos presentando una propuesta acerca de cómo se planean asignar los recursos, ya que durante el desarrollo del producto no es posible controlar todos los eventos que pueden ocurrir, y por este motivo se propone una asignación de recursos relativamente flexible con la cual aseguramos que el producto se presente sin importar que los integrantes del grupo Active se encuentren con una situación adversa.

7.3.4. Asignación de presupuesto y justificación

De acuerdo con lo planteado en el modelo de ciclo de vida (*Sección 6.1: Modelo de ciclo de vida*) y con las especificaciones de un modelo iterativo, el presupuesto se encuentra discriminado por cada uno de los sprint. Por consiguiente, para el desarrollo de SPMP, se realiza la descomposición de actividades respectivas (*Sección 8.3.1 Descomposición de actividades*), la cual está conformada por dos sprints de los cuales encontramos la discriminación presupuestal a continuación. El presupuesto se encuentra compuesto por los costos de horas laboradas por cada integrante y por los costos en inversiones tecnológicas requeridas para el desarrollo del producto, es decir equipos y herramientas (*Sección 7.2.2 Infraestructura*). De acuerdo con la *Ilustración 6* se muestra un salario promedio, de acuerdo con los años de experiencia que tienen los ingenieros de sistemas en Colombia en el año 2016. Teniendo como base este salario, se calculó el valor por hora trabajada, llegando a ser un promedio de $13.000 COP. (Tusalario, 2017)

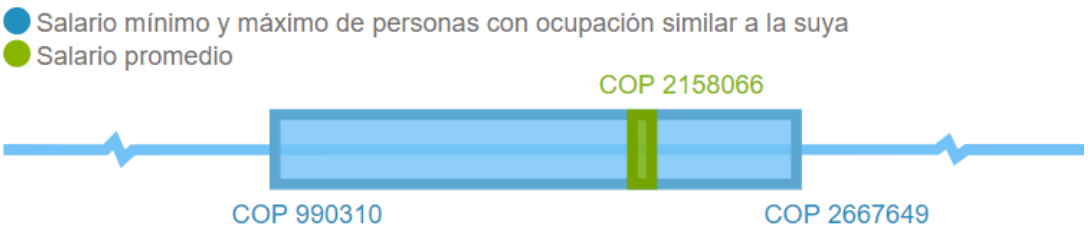

Ilustración 11:Salario bruto mensual de Ingeniero de sistemas colombiano

Sprint 1

A continuación, se muestra la discriminación presupuestal por horas de trabajo de los integrantes de CampusFinder para el primer Sprint en la siguiente tabla:

Empleado	Horas laboradas	Total
Andres Garcia	25	312500
Carlos Camacho	31	387500
Juan Gomez	32	400000
Luis Urdaneta	31	387500
Natalia Otero	31	387500
Total	150	1875000

Tabla 4 Presupuesto para sprint 1

Tabla 5 Salario presupuestado del primer Sprint

Sprint 2

A continuación, se muestra la discriminación presupuestal por horas de trabajo de los integrantes de CampusFinder para el primer Sprint en la siguiente tabla:

Empleado	Horas Laboradas	Total
Andres Garcia	30	375000
Carlos Camacho	30	375000
Juan Gomez	30	375000
Luis Urdaneta	30	375000
Natalia Otero	30	375000
Total	150	1875000

Tabla 6 Presupuesto para sprint 2

De acuerdo con los valores presupuestales presentados anteriormente, se desarrolló el siguiente plan de estimación de costos para cada Sprint, tomando como base la calendarización establecida. Para el trabajo estimado para cada Sprint se calcula un promedio de 3 horas diarias por lo cual solo se requiere la

multiplicación de horas por día y el total de días por sprint. Con esto obtenemos el total de horas requeridas por cada uno de los integrantes de CampusFinder, por lo cual es necesario multiplicar este valor por el número de integrantes dándonos como resultado el total de horas requeridas por Sprint

El tiempo estimado para cada Sprint es una duración de 3 semanas, por ende, es un total de 21 días, los cuales al multiplicarlos por el número de horas diarias requeridas nos da un total de 63 horas por persona. Al ser un total de 4 integrantes de CampusFinder, nos da un total de 252 horas por Sprint, teniendo en cuenta que el salario promedio por hora es de $13.000COP nos da un valor estimado de $3’276.000COP por cada Sprint realizado.

8. Monitoreo y control del proyecto

8.1. Administración de requerimientos

La administración de requerimientos es un proceso que tiene por objetivo comprender y controlar los requerimientos. (Administracion de Requerimientos, s.f.) Las etapas que comprenden dicho proceso son:

1. Requerimientos duraderos y volátiles.
2. Planeación de la administración de requerimientos.
3. Administración del Cambio de los requerimientos.

Dado a que la segunda entrega del proyecto, según lo especificado con la Ingeniera Anabel Montero, es un documento de especificación de requerimientos de software (SRS, por sus siglas en ingles). En este se hará la administración de requerimientos del proyecto. Los casos de usos de las funcionalidades del aplicativo web se encuentran en la sección *Anexo: Casos de uso.*

8.2. Monitoreo y control de progreso

8.2.1. Medidas

Para realizar una medición de trabajo efectuado, se usarán los porcentajes que la Ingeniera Anabel Montero definió para la calificación del proyecto. Así, según el valor que la profesora le otorgó a cada actividad, se definirá la prioridad que tiene para el proyecto. El grupo Activo consideró que esta manera era la más adecuada dado que la profesora actúa como cliente y es ella quien asignó los porcentajes. Estos porcentajes representan la prioridad que ella como cliente le da a cada elemento.

Con estas medidas se puede hacer un control de trabajo más concreto; se puede ver claramente el porcentaje de completitud del proyecto. Adicionalmente, se puede hacer una priorización de etapas según su porcentaje definido. También, cada actividad se le puede asignar un valor numérico según los ítems de la rúbrica que este representa. Los ítems se pueden ver detalladamente en el *Anexo: Rubrica del Proyecto.*

8.2.2. Control de progreso

Para hacer un control de progresó, se planteó un grupo de actividades dedicadas a esto, que son:

- Reuniones Semanales: Son reuniones donde se hace un monitoreo del trabajo efectuado durante la semana para así definir el proceso a seguir. En caso de dudas o dificultades, este espacio se debe usar para solucionarlas. Adicionalmente, se premiará a aquellos que cumplan con su trabajo asignado para motivar el trabajo del equipo.

- Reuniones de Trabajo: Estas reuniones serán efectuadas cada sprint. El propósito de estos espacios es de integrar el trabajo efectuado por los integrantes del grupo y hacer correcciones en el trabajo de cada uno en conjunto. Todo problema que se haya identificado antes o durante estas reuniones debe ser resuelto de manera inmediata a menos que sea imposible. También se hará revisión por pares del trabajo efectuado.

- Historial de Trabajo: Este artefacto se usa para comunicar el estado de cada actividad, su respectivo integrante responsable y para poder medir el trabajo completo de cada integrante. En este, se enumeran las actividades de cada entrega y sus responsables. Cada actividad tendrá uno de los siguientes estados durante el desarrollo de la entrega: En proceso, por revisar, por corregir y aceptado. Cuando una actividad se denomina “En proceso” significa que ya fue asignada un responsable y aquel deberá realizar la actividad. Al momento que los responsables de una actividad entreguen una versión que consideren completa, este cambia a estado de “Por revisar” donde el producto de su trabajo debe ser revisado por un Scrum Master para decidir si está listo para ser integrado o si requiere correcciones. En caso de que un elemento entregado requiera de correcciones, este entra en estado “Por corregir” donde debe estar acompañado de las correcciones necesarias para aprobar el elemento. Si el integrante responsable hace caso omiso a las correcciones, la actividad asignada se dará a otro integrante y el integrante responsable original será penalizado. Por último, cuando se decide que un ítem está listo para ser entregado, pasa a estado de “Aceptado”.

8.2.3. Acciones correctivas

Las acciones correctivas se ejecutarán acorde con las normas de trabajo especificadas en la sección *10.1 Ambiente de Trabajo*. Estas acciones deben ser impuestas por un Scrum Master, y en el caso de haber sido un Scrum Master quien incurrió en una falta, otro Scrum Master debe hacerse cargo de imponer la acción correctiva. Toda falla debe estar documentada en el Reporte Gerencial para hacer su debido seguimiento.

En caso de retrasos identificados en los sprints, los scrum masters deberán apoyar a los integrantes que presenten dificultades. En caso de que no se pueda lograr todos los objetivos o que la estimación no se cumpla de manera correcta por estancamiento de trabajo, se debe recurrir a realizar las actividades con mayor prioridad y, si se ve necesario, redistribuir las responsabilidades de los miembros para aumentar la eficiencia. Adicionalmente, al finalizar cada sprint, se hará una *Sprint Review Meeting* y una *Sprint Retrospective Meeting*. En la primera se deberá analizar el trabajo realizado para en caso tal de que aparezca una falla en el proceso, por medio de la *Sprint Retrospective Meeting,* corregir estos y mejorar la calidad del siguiente sprint.

8.3. Cierre del proyecto

En la siguiente sección, se hablará de los procesos necesarios para culminar los sprints mencionados anteriormente y determinar si los entregables cumplen con los requisitos de entrega al cliente. El proceso a seguir para la terminación de la entrega sigue el marco de trabajo de Scrum, donde al final de cada sprint se hace un análisis del trabajo hecho y el cierre se hará acorde a esto. Las actividades son:

- Reunión de revisión de sprint:
    1. Revisión de completitud de ítems de backlog: En esta etapa de la reunión, se debe verificar que los ítems entregados para integración cumplan con la definición de terminado y no requieran de trabajo adicional. Dependiendo de lo que se concluya, el entregable quedara integrado o se realizaran los cambios necesarios para que este se integre. En caso de que falte algún ítem, se asignara un responsable y se ejecutara de inmediato.

    2. Revisión de calidad: Aquí se revisa el entregable cumpla con los estándares de calidad, según lo definido en la *Sección 10.5: Estándares de Calidad*. En caso tal de encontrar errores se deben corregir de inmediato.

- Reunión de retrospectiva de sprint:
    1. Análisis de problemas encontrados durante sprint: En esta etapa de la reunión, cada integrante dirá los aspectos positivos y negativos que evidenció en el proceso de desarrollo. Esta información se usará para mejorar la calidad de los siguientes sprints. Adicionalmente, los problemas se deben enumerar y encontrar maneras proactivas de ser resueltos.

    2. Actualización documentos de seguimiento: Aquí se actualizan los documentos de gerencia, seguimiento de trabajo, y de historial de cambios.

3. Valoración de trabajo efectuado: En esta etapa, se le dara una valoración según su trabajo y comportamiento durante el desarrollo del sprint para premiar aquellos que hayan tenido un comportamiento excepcional y efectuar las medidas correctivas a aquellos que tengan un comportamiento negativo.

- Reunión de retroalimentación con el Product Owner (post-mortem):
    1. Recepción de evaluación por parte de Product Owner: En este paso, el Product Owner otorgara una medida de aceptación de los entregables en manera de una nota. Aquí el cliente comunicara si está de acuerdo con lo especificado en el proyecto. Adicionalmente el Product Owner hará una retroalimentación para mejora del proyecto.

    2. Realizar correcciones por parte de Product Owner: En esta etapa se construir actividades a partir de las correcciones dadas por el Product Owner y ejecutarse antes de seguir con el desarrollo del proyecto. Las correcciones se organizarán según su prioridad y se ejecutarán en ese orden.

9. Entrega del producto

La entrega del producto es responsabilidad de todos los integrantes de Active, ya que todo el equipo hizo parte del desarrollo del proyecto; siendo los Scrum Masters los encargados de manejar y verificar que se lleve a cabo lo aceptado en la *sección 8.3 Cierre del proyecto,* para poder continuar a la entrega del producto.
Se dará por entregado el producto y finalizado el proyecto cuando al cliente se le presenten los siguientes elementos:

1. SPMP - Software Project Management Plan: Documento en el que se presentaran los objetivos del proyecto, el cronograma, los costos, las herramientas, la metodología de desarrollo que se va a utilizar y la organización de los miembros del equipo.

2. SRS - Software Requirements Specification: Documento en el que se especificaran los requerimientos documentados en el SPMP y funciones que debe poseer el producto.

3. SDD - Software Design Description: En este documento se presenta la arquitectura del producto, el manual del usuario y un reporte gerencial en el que se encuentran las etapas del desarrollo del proyecto.

4. **Prototipo del aplicativo:** Al cliente se le entregará una implementación del aplicativo que cubrirá al menos el 75% de la funcionalidad y usabilidad que tendrá el producto final. Este prototipo será el resultado de varias iteraciones como resultado de la retroalimentación del cliente durante las diferentes reuniones y entregas que se realizarán a lo largo del semestre.

5. **Presentación del producto**: Se le presentara formalmente al cliente una recopilación del proceso de desarrollo junto al prototipo mencionado anteriormente. Se realizará una presentación presencial en donde se llevará a cabo la sustentación del desarrollo del proyecto, en donde se mostrará el producto, los procesos realizados y el estado en el que se hace entrega del mismo.

Todos los componentes expuestos han sido revisados por todos los integrantes del grupo, y solo serán presentados cuando los miembros del departamento de calidad verifiquen que cumplan con las condiciones que se encuentran en la *sección 10.5 Control de calidad,* y cumplan con lo estipulado en la *sección 6.3 Plan de aceptación del producto.* Para más información sobre la documentación correspondientes a los componentes SPMP, SRS y SDD remítase a la *sección 5.4 Entregables.*

Este producto no necesita de ninguna capacitación para los desarrolladores a futuro más allá de la que se encuentra descrita en los documentos entregados y no requiere de la implementación o transición a otra plataforma, pues la entrega de este producto incluye la plataforma sobre la que se trabajó el mismo.

10. Procesos de soporte

10.1. Ambiente de trabajo

Con el fin de mantener un ambiente de trabajo productivo y agradable, el grupo de trabajo Active estableció un sistema de control mediante puntos positivos, en gran medida, y puntos negativos, con el fin de poder promover el trabajo. Al finalizar cada sprint, la persona con más puntos positivos será reconocida y merecedora de un incentivo.

Se decide establecer las siguientes reglas tanto para el grupo, como para el trabajo. También especificaran a continuación las diversas faltas y las respectivas acciones correctivas.

1. Reglas de grupo
    a. Los integrantes de Active se comunicarán por medio de Trello, Slack, Google Hangouts y WhatsApp, sin embargo, la administración de documentos será gestionada a través de Google Drive.
    b. Todos los integrantes deben esmerarse para obtener trabajos de calidad y correctos.

c. Todos los integrantes deben destacarse por ser respetuosos, responsables y proactivos.
d. Los integrantes deben estar en óptimas condiciones para las reuniones y presentaciones.
e. Todas las propuestas serán escuchadas para tener mayor diversidad de puntos de vista.
f. Todas las acciones que perjudiquen la convivencia grupal serán dialogadas en equipo con el fin de encontrar una solución.
g. El documento final será manejado únicamente por el administrador de documentos.

2. Reglas de trabajo

Por medio de estas reglas se busca agilizar las metodologías de trabajo, con el fin de que cada miembro trabaje en la sección asignada, sin intervenir ni afectar el trabajo de sus compañeros. También se realiza con el fin de mejorar la calidad del trabajo realizando la revisión de las diferentes secciones de manera independiente.

a. Cada integrante del grupo deberá cumplir con las fechas de entrega asignadas en cada sprint para el trabajo que se le asigne al comienzo de este.
b. Cada sección asignada será montada para su revisión en Google Drive, en la carpeta correspondiente de su sección.
c. Los integrantes que se encuentren bajo la responsabilidad de una sección serán los encargados de especificar, definir y referenciar las diferentes palabras del glosario.
d. Los integrantes que se encuentren bajo la responsabilidad de una sección serán los responsables de las referencias y citar toda la información externa que se utilice en la sección.
e. Todas las referencias del proyecto serán hechas bajo las normas APA (sexta edición).
f. Todas las imágenes agregadas deberán poseer el título en su parte inferior de la siguiente manera: “Figura XXXX:<Nombre de la figura>”.
   Ejemplo: Figura XXXX: Ciclo de vida
g. En el caso de que se requiera hacer referencia a una sección, se hará de la siguiente manera <#sección><Nombre de la sección>. Esta descripción también se usará para la referencia de los anexos.
h. En caso de que un miembro acabe el trabajo asignado, este debe notificar y reportar su avance en el apartado de documentos formales poniendo en la sección de estado “Por revisar”, para que así, los gerentes procedan con su revisión y retroalimentar los puntos clave del trabajo entregado.
i. Una vez el documento haya sido revisado existen dos estados: el estado “Por corregir”, en el cual se encuentran comentarios que lleven a la corrección del trabajo presentado y el estado

"Aceptado", el cual muestra la aprobación de los miembros encargados de la calidad del documento.

j. Para la revisión de las secciones, los encargados de calidad del documento deben realizar un voto unánime para la aprobación del mismo. En caso contrario se solicitará al encargado de la sección que ejecute las correcciones pertinentes.
k. Los miembros que muestren su interés, responsabilidad y proactividad serán otorgados una estrella, la cual cumple el papel de motivar al integrante. Las estrellas se tendrán en cuenta al final de cada sprint para premiar al miembro que cuente con mayor número de estrellas.

3. Reglas para las reuniones
    a. Las reuniones se realizarán en las instalaciones de la universidad o en puntos acordados previamente por los miembros del grupo "Active".
    b. Todos los integrantes deben asistir a las reuniones los días jueves de 6:00 pm a 8:00pm o a las reuniones extraordinarias anunciadas con anterioridad. En caso de que alguno de los integrantes no pueda asistir, es su deber informar el motivo de su ausencia.
    c. Los integrantes que no asistan a las reuniones deberán justificar su ausencia. Dicha justificación será anexada a la bitácora de reuniones
    d. El uso de objetos distractores (celulares, audífonos, etc.) está prohibido en las reuniones de discusión.
    e. El tiempo de llegada máximo para las reuniones será de 15 minutos después a la hora asignada para la reunión.
4. Acciones correctivas
    a. Faltas leves: Son definidas como las acciones que afectan parcialmente la sana convivencia o el desarrollo de las actividades. Entre estas encontramos:
        i. No asistir a las reuniones sin tener una excusa valida.
        ii. Llamado de atención reiterativos para solicitar su trabajo.
        iii. Falta de calidad de su trabajo en las revisiones. Se asigna como un límite un máximo de 3 reuniones.
        iv. Incumplir con las reglas de trabajo o reuniones asignadas.
    b. Faltas graves: Son todas las acciones que puedan detener el desarrollo del proyecto y perjudiquen el ambiente de trabajo
        i. Agredir a los integrantes del equipo.
        ii. Ser impuntual para las actividad y reuniones del proyecto.
        iii. Incurrir en 3 faltas leves.

En el caso de que un integrante incurra en una falta grave, se procederá a realizar un proceso disciplinario para este integrante. Los procesos disciplinarios serán expuestos ante los otros miembros del grupo para definir las acciones a tomar. La ocurrencia de faltas graves tendrá como consecuencia un memorando para el

miembro implicado, llevando a su expulsión si se repiten después de ignorar las advertencias, según el proceso que se ha definido.

10.2. Análisis y administración de riesgos

Esta sección tiene como objetivo enseñar el proceso que se tendrá con los riesgos del proyecto, es decir, se mostrará tanto el análisis de riesgos, como la gestión respectiva que se tendrá con cada uno de estos, en esta sección también definiremos las diferentes metodologías a seguir en el caso de presentarse. De acuerdo con la *Ilustración 7*: *Tipos de Riesgos* se puede observar los componentes que conforman un riesgo, La incertidumbre, y su capacidad de perdida(impacto), también encontramos los diferentes tipos de riesgo que se pueden presentar. (Bustamante Barajas Juan Manuel, 2015).

Es importante mencionar y tener conocimiento que no todos los riesgos generan un efecto negativo en el desarrollo del proyecto, es decir, un proyecto posee riesgos negativos y riesgos positivos también conocidos como oportunidades. De acuerdo con lo anterior buscamos maximizar tanto los efectos, como la probabilidad de ocurrencia de las oportunidades. Por otro lado, a los riesgos se busca minimizar su impacto y su probabilidad de ocurrencia en el proyecto

Ilustración 12:Tipos de riesgos

Para el manejo de riesgos nos basaremos en la guía que nos ofrece el Project Management Body of Knowledge (PMBOK, por sus siglas en ingles), la cual se basa en un sistema repetitivo para la detección, gestión y monitorización de los riesgos. La guía suministrada por el PMBOK busca facilitar el desarrollo de los proyectos con el fin de aumentar sus probabilidades de éxito.

Desde la creación del PMBOK en 1969 este ha buscado establecer una guía para el desarrollo de proyectos, dándonos un apartado completo para la identificación, gestión y monitorización de riesgos. Como podemos observar en la *Ilustración 5: Tipos de Riesgos* se encuentran las diferentes fases para el plan de gestión de riesgos, el cual busca por medio de un proceso cíclico la identificación, gestión y

monitorización de los riesgos desde la creación del proyecto a su finalización de este.

Actividad	Descripción	Herramientas	Fuentes de Informacion
Planificacion de Gestion de Riesgos	Elaborar Plan de Gestion	PMBOK PMI Compendium	Sponsor y Usuarios en general PM y equipo de proyecto
Identificacion de Riesgos	Identificar riesgos del proyecto	Checklist de Riesgos RBS estandar	Sponsor y Usuarios en general PM y equipo de proyecto PM's y equipos de proyectos similares Consultores y proveedores tecnologicos Archivos históricos de proyectos
Analisis Cualitativo de Riesgos	Evaluar probabilidad e impacto Establecer ranking de importancia	Definicion de Probabilidad Definicion de Impacto Matriz de Probab-Impacto Welcom Risk	Sponsor y Usuarios en general PM y equipo de proyecto PM's y equipos de proyectos similares Consultores y proveedores tecnologicos
Análisis Cuantitativo de Riesgos	Evaluar prob e impacto global Calculo de Reserva de Contingencia	@Risk 4.1 for MS Project	Sponsor y Usuarios en general PM y equipo de proyecto PM's y equipos de proyectos similares Consultores y proveedores tecnologicos
Planificación de la Respuesta a Riesgos	Definir respuestas a riesgos Planificar ejecucion de respuestas	Welcom Risk	Sponsor y Usuarios en general PM y equipo de proyecto PM's y equipos de proyectos similares Consultores y proveedores tecnologicos Archivos históricos de proyectos
Supervisión y Control de Riesgos	Verificar ocurrencia de riesgos Supervisar ejecucion de respuestas Verificar efectividad de respuestas Verificar aparición nuevos riesgos	Welcom Risk	Sponsor y Usuarios en general PM y equipo de proyecto

Ilustración 13:Plan de gestión de riesgos

1. Identificación de riesgos: En esta actividad se procede a crear/actualizar la lista de riesgos posibles a los cuales el proyecto se encuentra vulnerable. Esta actividad se realiza mediante el análisis de probabilidades de supuestos y lluvia de ideas. Para esta actividad se optó por crear una lista de riesgos la cual se verificará y se caracterizaran los riesgos con el fin de llevar un conteo general de los riesgos presentes en el proyecto.

2. Análisis de riesgos: En esta actividad se consideran los riesgos identificados y se realiza un juicio de la probabilidad de ocurrencia y de la gravedad de que estos ocurran. El análisis de riesgos es posible dividirlo en un análisis cualitativo y/o cuantitativo teniendo cada uno sus respectivas ventajas. Entre las cuales encontramos:

    a. Análisis cualitativo: consisten en priorizar los riesgos evaluando su probabilidad de ocurrencia y urgencia, es decir, se tiene en cuenta la prioridad que tiene el riesgo según su impacto y probabilidad de que este suceda. El análisis cualitativo permite a los directores del proyecto enfocarse en gestionar los riesgos de alta prioridad, por lo cual se convierte en un método de análisis rápido y económico para establecer prioridades.
    b. Análisis cuantitativo: Es un análisis donde se mira numéricamente el efecto de los riesgos identificados. Esta información cuantitativa ayuda al proceso de toma de decisión que busca controlar la

incertidumbre en el proyecto. En este tipo de análisis se cuantifican los efectos que posee la ocurrencia de un riesgo en el presupuesto y en el cronograma.

3. Planificación de la respuesta a riesgos: En esta actividad se establecen las estrategias para manejar los riesgos seleccionados en el análisis de riesgos a través de acciones que buscan minimizar el efecto global que tiene el riesgo en el proyecto. De acuerdo con lo explicado al inicio de este apartado, se plantean dos tipos de estrategias para los riesgos y las oportunidades que se perciben en el proyecto:
    a. Riesgo

OPCION DE MANEJO	DESCRIPCION	TIPO DE RESPUESTA
Evitar el Riesgo	Medida encaminada a eliminar la actividad que genera el riesgo previniendo su materialización.	**Plan de Mejoramiento**
Reducir el Riesgo	Medidas encaminadas a disminuir tanto la probabilidad (medidas de prevención) como el impacto (medidas de protección). Se consigue mediante la optimización de los procedimientos y la implementación de controles.	
Compartir o Transferir el Riesgo	Medidas que reducen el efecto de un riesgo, a través del traspaso de las pérdidas a otras organizaciones.	
** **Aceptar el Riesgo**	Asumir (aceptar) la presencia de un riesgo mínimo o residual después de que el riesgo se ha reducido o transferido a otras organizaciones.	**Plan de Contingencia**

Tabla 7 Descripción de riesgos

    b. Oportunidades:
        i. Explotar: Se busca asegurar la oportunidad y obtener los mayores beneficios de esta oportunidad
        ii. Mejorar: Se busca aumentar las posibilidades y/o impacto positivo de una oportunidad en el proyecto
        iii. Compartir: Asignar una parte o en su totalidad la oportunidad a un tercero capacitado para asegurar la obtención de la oportunidad
        iv. Aceptar: Se aprovecha la oportunidad si esta se presenta durante el desarrollo del proyecto

4. Planificación de la respuesta de los riesgos: Esta actividad tiene como fin la creación de planes para combatir un riesgo. El PMBOK sugiere la implementación de varios planes y cooperación entre estos mismos con el fin de crear un plan general para la respuesta del riesgo, hasta el punto de solucionar este o transferirlo. La estrategia fundamentar es realizar la evaluación de los riesgos identificados de manera individual y generar una respuesta a cada uno, teniendo en cuenta la prioridad asignada.

5. Monitoreo del riesgo: Esta actividad implica el seguimiento continuo del riesgo durante el desarrollo del proyecto, con el fin de tomar acciones

correctivas y resolver los riesgos en cualquier momento del proyecto. Esta actividad se inicia en conjunto con el propio desarrollo del proyecto. Es importante tener en cuenta que es en esta fase de la gestión donde se mira la efectividad de las estrategias tomadas para los riesgos encontrados. También se busca identificar nuevos riesgos.

Medición de riesgo

A partir de la metodología propuesta por el PMBOK, como primera instancia el grupo Active realizo la respectiva reunión con el fin de obtener la identificación de los riesgos, en esta reunión el equipo de trabajo planteo todos los riesgos que se pudieron identificar, como se puede evidenciar en el *Anexo: Control de riesgos*, que es una hoja de cálculo con los respectivos valores asignados de probabilidad de ocurrencia y el impacto que cada uno de los riesgos de los cuales se generó la prioridad para cada uno de estos. De acuerdo con lo anterior, el grupo Active asigno los diferentes valores de probabilidad e impacto para los riesgos de la siguiente manera:

- Probabilidad de ocurrencia: Para el cálculo de la probabilidad de ocurrencia se establecieron 5 categorías según la probabilidad de ocurrencia:

TALBA DE PROBABILIDAD		
PROBABILIDAD	DESCRIPTOR	DESCRIPCIÓN
1	Raro.	El evento puede ocurrir solo en circunstancias excepcionales.
2	Improbable.	El evento puede ocurrir en algún momento.
3	Posible.	El evento podría ocurrir en algún momento.
4	Probable.	El evento probablemente ocurrirá en la mayoría de las circunstancias.
5	Casi seguro.	Se espera que el evento ocurra en la mayoría de las circunstancias.

Tabla 8 Calificación de impacto en un riesgo

Tabla 1: Calificación de impacto

- Impacto: para el cálculo del impacto se tuvo en cuenta el retraso en tiempo que implicaba en el cronograma del proyecto, el impacto que tendría en la calidad del proyecto. Dicho lo anterior se proponen

TABLA DE IMPACTO		
RIESGO	DESCRIPTOR	DESCRIPCIÓN
1	Insignificante.	Se el hecho llega a presentarse, tendría consecuencias o efectos mínimos sobre el proyecto.
2	Menor.	Se el hecho llegara a presentarse, tendría bajo impacto o efecto sobre el proyecto.
3	Moderado.	Si el hecho llegara a presentarse, tendría medianas consecuencias o efectos sobre el proyecto.
4	Mayor.	Si el hecho llegara a presentarse, tendría altas consecuencias o efectos sobre la entidad.
5	Catastrófico.	Si el hecho llegara a presentarse, tendría desastrosas consecuencias o efectos sobre el proyecto.

Tabla 9Tabla de impacto de riesgo

De acuerdo con lo anterior el *Anexo: Control de riesgos*, en la pestaña de "Matriz de calor de riesgos", se puede identificar amenazas/oportunidades y evidenciar la importancia de cada riesgo tomando como referencia la *Tabla 8: Matriz de calificación de riesgos*

Teniendo en cuenta las clasificaciones anteriores de probabilidades de ocurrencia y la clasificación de impacto se presenta la *Tabla 10 Matriz de calificación, evaluación y respuesta a los riesgos*

PROBABILIDAD	IMPACTO				
	Insignificante (1)	Menor (2)	Moderado (3)	Mayor (4)	Catastrófico (5)
Raro (1)	B (1)	B (2)	M (3)	A (4)	A (5)
Improbable (2)	B (2)	B (4)	M (6)	A (8)	E (10)
Posible (3)	B (3)	M (6)	A (9)	E (12)	E (15)
Probable (4)	M (4)	A (8)	A (12)	E (16)	E (20)
Casi Seguro (5)	A (5)	A (10)	E (15)	E (20)	E (25)

B: Zona de Riesgo Bajo: Asumir el Riesgo.
M: Zona de Riesgo Moderado: Reducir o Asumir el Riesgo.
A: Zona de Riesgo Alto: Evitar, Reducir, Compartir o Transferir el Riesgo.
E: Zona de Riesgo Extremo: Evitar, Reducir, Compartir o Transferir el Riesgo.

Tabla 10 Matriz de calificación, evaluación y respuesta a los riesgos

10.3. Administración de configuración y documentación

Para el desarrollo del proyecto, se estableció que la metodología que se manejaron para el control de versiones será por medio del versionamiento semántico, que consiste en que cada uno de los documentos de trabajo tiene

asignado una numeración referente a la modificación que se realizó y a su relevancia en el documento.

Las tablas siguientes corresponden a los ítems de configuración, los cuales serán utilizados a lo largo del proyecto. Para esto, el grupo Active se basó en lo anteriormente dicho para que cuando se realicen modificaciones a los documentos fuera necesaria la aprobación de todos los integrantes y solamente será modificado por el responsable del documento.

Nombre	Software Project Management Plan (SPMP)
Descripción	Documento inicial solicitado por el Product Owner cuya finalidad es mostrar al usuario las herramienta, metodologías, estándares y ambientes de trabajo que serán usados por el equipo de trabajo Active para la elaboración del producto final.
Control de cambios	Natalia Otero y Carlos Camacho son los únicos autorizados a realizar cambios del documento y para eso se utilizará la herramienta Google drive y el versionamiento semántico. Al iniciar el proyecto, se estableció que para estos documentos de entrega al Product Owner se manejará el major, minor y patch correspondientes a la entrega a la que se refiere (major), la modificación del documento al finalizar cada Sprint (minor) y las correcciones de coherencia y ortografía por parte de los responsables (patch). Para que en el documento se realice alguna modificación de carácter menor debe cumplir con lo dicho en la sección *9 Entrega de producto*
Versión Final General	1.8.3 Versión final del documento SPMP
Almacenamiento	Se mantendrán todas las versiones en Google drive, con necesidad de descarga para los responsables y de solo lectura para los demás integrantes del equipo.

Tabla 11Versionamiento SPMP

Nombre	Casos de Uso (CU)
Descripción	Conjunto de diagramas que representan una serie de funcionalidades objetivo que el grupo Active espera que tenga el aplicativo CampusFinder. Estos diagramas cuentan tanto con los procesos generales que se trabajarán como con el flujo de las funcionalidades del mismo.
Control de cambios	A pesar de que la elaboración sea por parte de la totalidad del grupo Active, el responsable de la virtualización de lo que se haya decidido en cuanto a los CU será por parte de Andrés García, quien será el único autorizado a manejar el versionamiento por cada uno de ellos.
Versión Final General	1.11.2
Almacenamiento	Aunque solo Andrés se encarga de virtualizar los casos de uso, se almacenan tanto las fotos borrador de dichos CU como los ya virtualizados en una carpeta compartida en GoogleDrive.

Tabla 12Versionamiento casos de uso

Nombre	Diagramas
Descripción	Conjunto de diagramas correspondientes a los anexos de apoyo al documento SPMP los cuales corresponden a: Diagrama de Gantt, Diagramas BPMN de modelos de ciclo de vida y control de calidad, diseño del logo y el organigrama. Todos estos diagramas se hacen con el fin de facilitar la comprensión del documento.
Control de cambios	La totalidad de los diagramas, aunque son pensados por el grupo Active hay un responsable por cada uno, el cuál es el único autorizado a modificarlo antes de ser presentado como anexo terminado a los Scrum Master, los cuales establecerán si deben ser modificados por el responsable o si son aceptados de acuerdo con los estándares propuestos en la sección *10.5 Control de calidad*
Versión Final General	1.5.2
Almacenamiento	Todos los diagramas serán guardados en una carpeta llamada anexos la cual se encuentra en la carpeta del proyecto del Drive compartido con el grupo Active.

Tabla 13Versionamiento diagramas

10.4. Métricas y proceso de medición

En la siguiente sección se mostrará el proceso por medio el cual se harán las mediciones de calidad del proyecto y del producto, el tiempo que tome cada actividad y como se hará la medición del manejo de riesgos y el control de proceso del trabajo.
Estas medidas fueron tomadas por los métodos y herramientas de estimación tomados en la sección *7.1 Métodos y herramientas de estimación.*

10.4.1. Calidad de documento

La calidad del producto se mide con respecto a la *Sección 6.3 plan de aceptación del producto*. Los integrantes del equipo deben procurar ejecutar trabajo de calidad que cumpla con lo requerido. Esto es que todo elemento del documento sea coherente, tenga buena redacción, buena ortografía y tenga el menor número de errores posible. Adicionalmente, cada sección tendrá sus condiciones de calidad que deben ser cumplidas.

10.4.2. Calidad de software

Al igual que lo descrito para la calidad del documento, el software seguirá lo estipulado en la *Sección 6.3: Plan de Aceptación del Producto*. Entonces la medición a seguir se puede hacer según la funcionalidad del producto y número de líneas de código escritas. Adicionalmente, la calidad se medirá durante cada iteración del proyecto. Esto es que durante cada sprint se hará un análisis de calidad del producto y se hará una retroalimentación donde se

puede definir que hay a favor y que debemos corregir durante el proceso. De esto podemos medir el número de problemas que resultan del diseño.

10.4.3. Medición de tiempo

Para llevar un manejo adecuado del tiempo, se medirá la relación entre las estimaciones de tiempo y el tiempo real que tomo la ejecución de una actividad. Scrum nos ofrece una opción de medir esto mediante el Product Backlog y los mecanismos de sprint que son las reuniones de inicio y de cierre de sprint. El Backlog nos otorga una visualización del trabajo efectuado contra el tiempo y lo compara con el estimado.

10.4.4. Medida para el control de avance de proyecto

Para medir el avance del proyecto se usarán diagramas de Gantt utilizando la herramienta Microsoft Office Project. Adicionalmente hay especificaciones de sobre las fechas y duraciones de actividades en la *Sección 8.3.2: Calendarización* y especificación de las actividades de monitoreo a seguir en la *Sección 9.1: Monitoreo y control de progreso.*

10.4.5. Medida de calidad total del proyecto

La medición de los riesgos involucra cinco actividades que se deben ejecutar durante el desarrollo del proyecto. Este consta de la identificación de riesgos, el análisis de riesgos, la planificación de los riesgos y el control y monitoreo de los riesgos. De aquí podemos cuantificar el impacto que tiene cada riesgo en el proyecto y con esta información determinar cómo enfrentarse a los riesgos. La administración de riesgos se ve en detalle en la *Sección 10.2: Análisis de Riesgos.*

Para determinar la calidad total del proyecto, cada una de las actividades realizadas por cada integrante del equipo Active Se le medirá bajo los lineamientos encontrados en la sección 8*.2.2 Control de Progreso.*
La calidad general del proyecto se mide que cumpla con los estándares de calidad que se encuentran en la *Sección 6.3: Plan de Aceptación del Producto* y se medirá que tan acorde es a los objetivos del proyecto. Adicionalmente cada sección del documento realizada tendrá una valoración numérica asociada. Las valoraciones numéricas posibles son:

Nota	Descripción
0	No se entregó la sección
1-2	La sección carece de coherencia con el contenido requerido
2-3	Sección casi completa, pero contiene errores ortográficos y de redacción
3-4	Sección carece de referencias que sustenten su contenido
4-5	Sección cumple con los estándares de calidad definidos en la *Sección 11.5: Control de calidad* y se considera como completa

Tabla 14Calificación de calidad

A continuación, se mostrará la tabla de medición de procesos:

Proceso	¿quién?	¿Cómo?	¿Cuándo?	Herramientas
Calidad de documento	Departamento de calidad	Los documentos que los integrantes entregan pasan por revisión antes de ser aceptados.	Todos los días y se hará una revisión general cada jueves	Google drive
Calidad del software	Equipo de desarrollo	Se realizarán pruebas unitarias cada vez que un componente se integre. La programación se hace en pares.	Al finalizar un avance de software.	Lenguajes de programación, herramientas como hosting y servidor
Tiempo	Cada miembro del grupo Active	Las actividades se dividen en sprints. Aquellas actividades que se decidan para un sprint deben ser culminadas en el sprint.	Todos los días	Google drive
Calidad de diseño	Líder de diseño	Al momento de realizar un diseño, el líder de diseño deberá aprobar dicho diseño. Este debe ser acorde a los requerimientos del proyecto.	Al finalizar un diseño	Google drive

Calidad de proceso	Scrum Masters	Como indica el rol, los Scrum Masters deben asegurar que el proceso de Scrum se ejecute y se mantenga.	Al inicio y final de cada sprint	Google drive
Riesgos	Los integrantes de Manejo de Riesgos	Se hará medición de los problemas que surjan durante el desarrollo de los sprints.	Revisión constante	Excel

Tabla 15Procesos de medición

10.5. Control de calidad

El grupo Active quiere asegurarse que se le entregue al cliente un producto que sea capaz de satisfacer todas sus necesidades, y por este motivo, se establecieron procedimientos con los cuales los integrantes del grupo verificaran que durante el desarrollo del producto no se presenten defectos o inconsistencias en los documentos o en el producto.

Para el proceso de evaluación, un integrante de cada uno de los departamentos se reunirá para ejecutar el proceso de control de calidad siguiendo las pautas establecidas para los diferentes tipos de archivos; y al finalizar la evaluación el Scrum master examinará el documento con las correcciones que se hagan y decidirá si cumple con lo que se espera o si requiere de una nueva revisión.

A continuación, se explicará en detalle el proceso y los elementos a evaluar dependiendo del elemento que se esté revisando:

- Calidad de los documentos:
  - ✓ Coherencia: El contenido del documento debe ser consistente y debe seguir una línea de ideas sin ramificarse más de lo que sea necesario para que se entiendan los contenidos.
  - ✓ Legibilidad: el documento debe poder leerse sin dificultad y para esto hay que verificar que tenga una buena sintaxis. Es importante verificar la coherencia del documento antes de entrar a evaluar la legibilidad ya que es necesario entender el contenido que el documento trabaja para poder tener un contexto en el cual se verifique si la sintaxis es a adecuada.
  - ✓ Ortografía: Se ha de revisar la ortografía del documento para verificar que las palabras que se usen estén bien escritas y la puntuación del documento debe permitir la lectura del mismo con fluidez.
  - ✓ Referencias: Todos los elementos que requieran de citas bibliográficas y todas las fuentes de información que se hayan utilizado como base teórica

para la realización del documento han de ser referenciadas al final de documento bajo la norma APA.

- ✓ Estilo (de escritura): Para la presentación de los documentos se ha de usar un lenguaje formal y claro que sea conciso y consistente. La utilización de términos técnicos ha de

Al finalizar la evaluación de estos componentes, se realizará una comparación entre el documento actual y su última versión aprobada con el objetivo de identificar discrepancias antes de que se apruebe la entrega actual como versión más reciente del documento.

- Calidad del código:
  - ✓ Formato: El equipo de desarrollo definirá el formato que tendrán los archivos que tengan código, y como ha de cambiar el formato dependiendo del lenguaje con el que se trabaje.
  - ✓ Comentarios: Todos los archivos de código han de tener comentarios donde sea apropiado (nombrado de alguna función o variable, estado de un segmento, etc.) que expliquen rápidamente la funcionalidad del segmento de código que se está evaluando.
  - ✓ Funcionalidad: se verificará si el código funciona tal como se propone.

  Al finalizar la evaluación del código, el scrum master y el líder del grupo desarrollador han de evaluar si el código cumple con lo que se pide (tanto a nivel de código escrito como a nivel de funcionalidad en los prototipos), y de ser asiste subirá al repositorio el código con un comentario donde se indique que se aprobó por el scrum master

- Calidad de requerimientos:
  En el caso de la calidad de los requerimientos, se especifica el proceso con los cuales se evalúan en el documento *SRS - Software Requirements Specification.*

- Calidad de diseño y diagramas:
  - ✓ Coherencia: El contenido del elemento debe ser consistente y debe seguir una línea de ideas sin ramificarse más de lo que sea necesario para que se entiendan los contenidos.
  - ✓ Legibilidad: El elemento y su contenido deben ser fáciles de entender a primera vista.
  - ✓ Ortografía: Se ha de revisar la ortografía del elemento para verificar que las palabras que se usen estén bien escritas y tengan correcta puntuación.
  - ✓ Referencias: Todos los elementos que requieran de citas bibliográficas y todas las fuentes de información que se hayan utilizado como base teórica han de ser referenciadas.

Los diagramas y los elementos de diseño se deben realizar de acuerdo con los estándares correspondientes al tipo de diagrama que se vaya a evaluar (EJ: diagrama de casos de uso según los principios de diseño de UML). Todos los

integrantes analizaran los diagramas y votación se determinará si cumple con su objetivo. Se planea realizar de esta manera para que todos los integrantes tengan idea de que está sucediendo en los diferentes procesos que se están desarrollando al mismo tiempo.

- Calidad de procesos:

La evaluación de la calidad de procesos es un proceso constante que se realiza durante el desarrollo del proyecto. El scrum master se encargará de aprobar todo lo que se considere pertinente a la entrega final del proyecto, lo que implica que evaluará y registrará todos los cambios. Esos resultados y registros se expresarán en el reporte gerencial.

11. Anexos

Se encuentran presentes en la carpeta de anexos.

Bibliografía

(n.d.). Retrieved from W3.CSS Home: https://www.w3schools.com/w3css/

Administracion de Requerimientos. (n.d.). Retrieved 8 25, 2017, from http://clases3gingsof.wikifoundry.com/page/Administraci%C3%B3n+de+Requerimientos

Agile Alliance. (n.d.). *What is Extreme Programming (XP)? | Agile Alliance*. Retrieved from Agile Alliance: https://www.agilealliance.org/glossary/xp/#q=~(filters~(postType~(~'post~'aa_book~'aa_event_session~'aa_experience_report~'aa_glossary~'aa_research_paper~'aa_video)~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1)

Amazon Web Services. (n.d.). Retrieved from https://aws.amazon.com/es/nosql/

Bizmanualz. (n.d.). *Document Manager Job Description*. Retrieved 8 17, 2017, from https://www.bizmanualz.com/job-descriptions/aerospace-job-descriptions/document-manager-job-description

Bustamante Barajas Juan Manuel, C. O. (2015, Junio 28). *Gestión de Proyectos de software SCG-1009 SC9A*. Retrieved Agosto 21, 2017, from https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnwyMDE1MDZncHN1bGxvYXJvZG9sZm98Z3g6M2ViMGY2ZGJmNzY0MjljYQ

Ciclo de vida en "V" - INGENIERIA DE SOFTWARE. (n.d.). (Weebly) Retrieved 8 20, 2017, from http://ingsoftware.weebly.com/ciclo-de-vida-en-v.html

International Scrum Institute. (n.d.). *Scrum Roles – The Scrum Team - International Scrum Institute*. (International Scrum Institute) Retrieved 8 16, 2017, from http://www.scrum-institute.org/Scrum_Roles_The_Scrum_Team.php

Introduction to HTML. (n.d.). Retrieved from https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML

ISQTB. (n.d.). *What is Spiral model- advantages, disadvantages and when to use it?* Retrieved 8 19, 2017, from http://istqbexamcertification.com/what-is-spiral-model-advantages-disadvantages-and-when-to-use-it/

iZenBridge. (n.d.). *What Is Configuration Management?* Retrieved 8 19, 2017, from https://www.izenbridge.com/blog/what-is-configuration-management-a-software-management-study/

Project Management Institute Staff. (2013). *A Guide to the Projecto Management Body of Knowledge(PMBOK Guide)-Fifth Edition.* Newtown square,Pa: Project Management Institute Inc.

Scrum Alliance. (n.d.). *Scrum Roles Demystified - Scrum Alliance*. (Scrum Alliance) Retrieved 8 16, 2017, from https://www.scrumalliance.org/agile-resources/scrum-roles-demystified

Scrum.Org. (n.d.). *The Scrum Guide™*. Retrieved 8 21, 2017, from https://www.scrumguides.org/scrum-guide.html

SCRUMstudy. (2016). *A Guide to the Scrum Body of Knowledge (SBOK™ Guide) – 2016 edition* . VMEdu Inc.

Software Engineering Institute. (n.d.). *Duties, Skills, & Knowledge of a Software Architect*. Retrieved 8 19, 2017, from https://www.sei.cmu.edu/architecture/research/previousresearch/duties.cfm

SOMMERVILLE, I. (2011). *Software Engineering (Ninth Edition).* Adison Wesley.

The Interantional Scrum Institute. (n.d.). *Scrum Revealed.*

Tusalario. (2017, 08). *Tusalario.org/Colombia*. Retrieved from http://www.tusalario.org/colombia/Portada/tusalario/compara-tu-salario#/

Unified Process for Education. (n.d.). *Role: Configuration Manager*. Retrieved 8 19, 2017, from http://www.upedu.org/process/workers/wk_cmmgr.htm

Weese, S. (2009, 4 9). *Identify Projecto Assuptions and Constraints | Learning Tree*. Retrieved 4 9, 2009, from https://blog.learningtree.com/identifying-projecto-assumptions-and-constraints/