20 DE MAYO DE 2019

WS-GUARDIAN CLIENT'S AGENT

SOFTWARE PROJECT MANAGEMENT PLAN

PROYECTO DE GRADO

CHRISTIAN GIOVANNY ROJAS DIAZ
PONTIFICIA UNIVERSIDAD JAVERIANA
Colombia, Bogotá D.C.

JUAN DAVID GAMA PEÑA
PONTIFICIA UNIVERSIDAD JAVERIANA
Colombia, Bogotá D.C.

MICHAEL ANDRÉS VARGAS BUITRAGO
PONTIFICIA UNIVERSIDAD JAVERIANA
Colombia, Bogotá D.C.

RIE KANEKO BOJACÁ
PONTIFICIA UNIVERSIDAD JAVERIANA
Colombia, Bogotá D.C.

WILLIAM ALEXANDER MORENO PRIETO
PONTIFICIA UNIVERSIDAD JAVERIANA
Colombia, Bogotá D.C.

Historial de Cambios

Versión	Fecha	No. Sección	Descripción de cambios	Responsable
1.0	1/05/2019	Todos	Creación del documento	Todos
1.1	19/05/2019	5	Se agrego nueva sección	Juan Gama

Prefacio

El presente documento constituye el *Software Project Management Plan* (*SPMP*), donde se aborda la visión y el alcance del producto, mostrando los objetivos generales y específicos. Posteriormente se trata el contexto del proyecto dando a conocer aspectos relevantes del desarrollo junto con la planificación de las tareas según el tiempo de disponibilidad. Se plantean algunos métodos de monitoreo y acciones a tomar frente a posibles cambios que puedan surgir a lo largo del trabajo de grado. Finalmente se realiza la descripción de los entregables y algunos de los procesos de diferentes áreas que participan en el desarrollo del mismo.

Este proyecto busca externalizar la implementación de técnicas de seguridad logrando que el consumo de *Web Services* protegidos por *WS-Guadian* sea más transparente frente al usuario, al solo tener que hacer un consumo normal (sin necesidad de añadir técnicas de seguridad). Nuestro agente se encargará de ver y aplicar las políticas que cada servicio requiera, de esta forma se reducirán los costos que recaen sobre el cliente, al no tener que modificar y adaptar cada aplicación que hace uso de un servicio web protegido por WS-Guardian.

Se adoptó la metodología *SCRUM,* debido a que es la más recomendada para proyectos medianos. Esta metodología ayuda a realizar las diferentes tareas en un corto tiempo, de esta manera se hace más fácil revisar y verificar rápidamente si ocurre algún inconveniente. Para poder cumplir con los objetivos, a cada miembro del grupo se le asignará una tarea diferente, y el progreso se medirá conforme se van cumpliendo las tareas, hasta tener el prototipo del agente que el cuál debe ser entregado a *ITAC S.A* junto con su respectiva documentación.

Este documento es dirigido a *ITAC S.A*, *Pontificia Universidad Javeriana* y a quien le interese el proyecto.

Contenido

Listas de Figuras 4
Listas de Tablas 4
Anexos 4
1. Visión General del Proyecto 5
1.1. Visión del producto 5
1.2. Propósito, Alcance y Objetivos 5
1.2.1. Propósito 5
1.2.2. Alcance 5
1.2.3. Objetivos 6
1.3. Supuestos y Restricciones 6
1.3.1. Supuestos 6
1.3.2. Restricciones 6
1.4. Entregables 7
1.5. Resumen de Calendarización 7
1.5.1. Calendarización 7
2. Glosario 8
3. Contexto del Proyecto 9
3.1. Modelo de Ciclo de Vida 9
3.2. Lenguajes y Herramientas 9
3.2.1. Lenguajes 9
3.2.2. Herramientas 9
3.3. Plan de Aceptación del Producto 10
3.4. Organización del Proyecto y Comunicación 11
3.4.1. Interfaces Externas 11
3.4.2. Organigrama y Descripción de Roles 11
4. Administración de Proyecto 13
4.1. Métodos y Herramientas de Estimación 13
4.2. Inicio del Proyecto 13
4.2.1. Entrenamiento de Personal 13
4.3. Planes de Trabajo del Proyecto 14
4.3.1. Descomposición de Actividades 14
4.3.2. Calendarización 15
5. Proceso Ingeniería de Requerimientos 16
6. Monitoreo y Control de Proyecto 17
6.1. Administración de Requerimientos 17
6.2. Monitoreo y Control de Progreso 17
6.3. Cierre del Proyecto 19
7. Entrega del Producto 19
8. Proceso de Soporte 20
8.1. Ambiente de Trabajo 20
8.2. Análisis y Administración de Riesgos 20
8.3. Administración de Configuración y Documentación 21
8.4. Control de Calidad 23
9. Referencias 25

Listas de Figuras

Figura 1: Organigrama 11
Figura 2: Work Breakdown Structure 15
Figura 3: Colores representativos de las fases 15
Figura 4: Diagrama de Gantt 15
Figura 5: Diagrama Proceso ingeniería de requerimientos 16
Figura 6: Modelo BPMN de Control y Mitigación de Riesgos 20
Figura 7: Diagrama de ramas de desarrollo 22

Listas de Tablas

Tabla 1: Entregables 7
Tabla 2: Glosario 8
Tabla 3: Clasificación y empleo de herramientas 10
Tabla 4: Interfaces externas 11
Tabla 5: Roles y responsabilidades 12

Anexos

Anexo 1: Diagrama BPMN Administración de Requerimientos.png
Anexo 2: Reglas de trabajo.xlsx
Anexo 3: Control y mitigación de riesgos.xlsx
Anexo 4: Diagrama BPMN Revisión de documentos.png
Anexo 5: Pruebas de aceptación.png

1. Visión General del Proyecto

1.1. Visión del producto

WS-Guardian Client's Agent busca hacer más transparente el consumo de *Web Services* (WS), en lo que hace referencia a la seguridad de la información que viaja a través de la red. Logra hacer más segura la comunicación, implementando técnicas de aseguramiento a nivel de mensaje, entre el cliente y el *WS*.

Al lograr hacer más transparente la implementación de técnicas de seguridad, se reducen costos de tiempo y dinero, ya que el cliente no tendrá que hacer la implementación de las técnicas en sus aplicaciones. *WS-Guardian Client's Agent* se integrará con el producto *WS-Guardian* de la empresa *ITAC S.A*, buscando la aceptación y acogida de los usuarios que hacen uso de este servicio (*WS-Guardian)*.

1.2. Propósito, Alcance y Objetivos

1.2.1. Propósito

Actualmente cuando una empresa instala políticas de seguridad sobre sus *WS*, la responsabilidad de implementar las técnicas de seguridad para consumir dichos *WS* queda a cargo del cliente. Esto hace que el implementar estas políticas en sus servicios, sea costoso en materia de tiempo, dinero y esfuerzo, al tener que hacer los cambios en sus aplicaciones clientes de los *WS* con políticas.

Con *WS-Guardian Client's Agent* se quiere reducir los costos del cliente (tiempo, dinero y esfuerzo), al momento de consumir un *WS*. Al lograr automáticamente conocer qué políticas requiere el servicio para ser consumido (políticas que pueden añadirse, modificarse y/o ser removidas desde *WS-Guardian*), y aplicándolas sobre el mensaje, disminuyendo los costos que tiene el cliente al no tener que modificar su sistema.

1.2.2. Alcance

WS-Guardian Client's Agent debe cumplir con los siguientes alcances:

- Comunicación con *WS-Guardian* para conocer las políticas por servicio.
- Un registro de logs con tamaño variable definido por el cliente, con el fin de ayudar a visualizar el estado del agente.
- Una capa de administración para los módulos que implementan técnicas de seguridad.
- Dos módulos de implementación de dos técnicas de seguridad.
- Capacidad para descargar el código en tiempo de ejecución de las políticas que necesite.
- Comunicación con *WS-Guardian* para realizar una inicialización de *WS-Guardian Client's Agent*.

WS-Guardian Client's Agent no contara con:

- Soporte en comunicaciones de tipo *REST*.
- La totalidad de implementaciones para cubrir las políticas de *WS-Guardian*.
- Conexión para centralizar los logs.
- Interfaz gráfica.
- Motor de base de datos.

1.2.3. Objetivos

1.2.3.1. Objetivo General

Diseñar un agente que se ejecute en el lado del consumidor de servicios, el cual proporcionará principios de seguridad como *autorización, confidencialidad, integridad y no repudio* sobre los mensajes emitidos hacia *WS-Guardian*.

1.2.3.2. Objetivos Específicos

- Diseñar el agente para que pueda ser ejecutado en dos o más plataformas.
- Implementar un prototipo del agente para dos o más plataformas.
- Implementar dos técnicas de seguridad dentro del agente.
- Determinar la validez de los requerimientos funcionales y no funcionales por medio del prototipo.
- Demostrar la calidad de software mediante una metodología de pruebas.

1.3. Supuestos y Restricciones

Para realizar el proyecto se tendrán en cuenta los siguientes supuestos y restricciones, los cuales son necesarios para el correcto desarrollo y ejecución de este.

1.3.1. Supuestos

- La empresa *ITAC S.A.* está interesada en el agente.
- El Agente es aceptado y desplegado por *ITAC S.A.* en los clientes interesados.
- No aparecen cambios extremos durante el desarrollo de la aplicación.

1.3.2. Restricciones

- El agente será desarrollado por los miembros del equipo.
- Se debe usar una herramienta de versionamiento.
- Se debe usar una técnica de codificación con el fin de hacer el código entendible y estructurado.
- El cliente cuenta con *Java Runtime Environment* (*JRE*) con el fin de que pueda ejecutar *WS-Guardian Client's Agent*.
- *WS-Guardian Client's Agent* se ejecutará en una máquina que esté conectada a la red donde se ubica *WS-Guardian*.
- *WS-Guardian Client's Agent* se ejecutará en la máquina del cliente.

- Las máquinas son capaces de ejecutar *WS-Guardian Client's Agent*.

1.4. Entregables

El proyecto contará con los siguientes entregables:

Entregable	Descripción	Destinatario	Fecha
PMP	Plan de Administración de Proyecto de Software, documento que describe el contexto, la administración del proyecto y la vista general.	Stakeholder (Ing. Jaime Andrés Pavlich Mariscal)	12/05/2019
SRS versión inicial y final	Especificación de Requerimientos de Software, documento que especifica las funcionalidades y atributos pertenecientes al sistema a desarrollar.	Stakeholder (Ing. Jaime Andrés Pavlich Mariscal)	16/05/2019
SDD versión inicial y final	Documento de Diseño del Software, que establece la arquitectura e interfaces lógicas de usuario que se realizarán en el proyecto.	Stakeholder (Ing. Jaime Andrés Pavlich Mariscal)	16/05/2019
Informe y Documento de Pruebas	Documento que muestra las pruebas y resultados de las mismas a las que fue sometido el prototipo, con el fin de validar el correcto funcionamiento de este, tanto modular como completo asegurando el cumplimiento de los requerimientos del cliente.	Product owner (Ing. Javier Humberto Galindo)	22/11/2019
Prototipo Funcional / Código Fuente	Al finalizar el proyecto, el equipo de trabajo entregará un prototipo funcional acorde a los requerimientos especificados en el documento SRS. Se entregará un código que cumpla con el estándar para que sea mantenible y legible, con el fin de permitir que cualquier ingeniero pueda continuar con el desarrollo del software.	Product owner (Ing. Javier Humberto Galindo)	22/11/2019
Manual de Usuario	Junto al prototipo funcional se entregará un manual de usuario, que debe cumplir con el estándar para que sea entendibles.	Product owner (Ing. Javier Humberto Galindo)	22/11/2019

Tabla 1: Entregables

1.5. Resumen de Calendarización

1.5.1. Calendarización

El proyecto contará con varias fases e hitos que irán distribuidos a lo largo del periodo de desarrollo del proyecto. Cada una de las fases contará con diversos entregables (*sección 4.3.2 Calendarización*), las fases definidas son:

- Fase inicial.
- Fase de elaboración.
- Fase de construcción.
- Fase de validación y cierre.

2. Glosario

Palabra	Descripción
BPMN	Modelo y Notación de Procesos de Negocio.
CEO	Jefe ejecutivo.
Definiciones de probabilidad e impacto	Definición de cada escala de clasificación de impacto. Las definiciones son dadas en materia de costo, tiempo, alcance y calidad, indicando el efecto negativo del riesgo sobre el proyecto.
ITAC S.A	Compañía fabricante de soluciones de seguridad y SOA.
Matriz de probabilidad e impacto	Contiene valores que representan el nivel de impacto que causa un riesgo sobre el proyecto, clasificados en: muy bajo, bajo, moderado, alto y muy alto. Estos datos son asignados a los riesgos detectados como apoyo de administración.
Planning Poker	Método de estimación de Software.
Product Backlog	Lista de requisitos del proyecto.
RBS	Risk Breakdown Structure es la representación jerárquica de riesgos, inicia desde los niveles más altos hasta los más específicos. El nivel 1 divide el riesgo en técnico, externo, infraestructura, cliente, y de administración, y el nivel 2 pasa a niveles más finos como riesgos de requerimientos, diseño, calidad, aplicación, etc.
RUP	Rational Unified Process, metodología de trabajo para desarrollo de software.
SCRUM	Marco de trabajo para desarrollo ágil.
Scrum master	Líder del scrum team (equipo).
SPMP	Planeación del proyecto.
Sprint	Planeación corta.
Sprint Planning	Lista de tareas del sprint.
Stakeholder	Interesado en el proyecto.
TI	Tecnologías de información.
Trello	Plataforma de administración de proyectos.
WS	Web Service (Servicio Web).
WS-Guardian	Plataforma de seguridad desarrollada por ITAC.
WSG-Agent	Ws Guardian Client's Agent, el nombre del software que se va a implementar.
XP	Metodología ágil.

Tabla 2: Glosario

3. Contexto del Proyecto

3.1. Modelo de Ciclo de Vida

Se hará en combinación de *RUP* y *SCRUM*, el detalle de las metodologías, actividades y resultados se encuentran en la Versión Final de la Propuesta, sección 3: Proceso.

Justificación de la elección

Para el proyecto de grado se opta por la metodología *RUP* debido a que en su desarrollo se tiene mucho en cuenta la documentación, con esto se reducen riesgos del proyecto, así mismo define las actividades en cada fase junto con sus resultados.

Por otra parte, una metodología ágil como *SCRUM* determina el marco de trabajo de las actividades además de adaptarse a cambios de acuerdo con las necesidades del negocio. De esta forma se combinan las dos metodologías para llegar a un resultado exitoso.

3.2. Lenguajes y Herramientas

3.2.1. Lenguajes

Dado que ITAC S.A provee soluciones tecnológicas basadas en Java, este es el lenguaje seleccionado para el desarrollo del trabajo de grado.

Java se ejecuta sobre un servidor *JRE* (*Java SE Runtime Environment*), pertenece al paquete de *Java ™ Platform, Standard Edition Runtime Environment (JRE ™).* El servidor *JRE* es un entorno de ejecución específicamente diseñado para implementar Java en entornos de servidor y está disponible para plataformas *Linux*, *Solaris* y *Windows*. [1]

3.2.2. Herramientas

Para el proyecto se hará uso de las siguientes herramientas:

Herramienta	Tipo	Empleo
Netbeans	Desarrollo	*IDE* es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. [2]
Jenkins	Administración prototipo	Servidor de automatización de código abierto, se utiliza para automatizar tareas relacionadas con la creación, prueba y entrega de software, a través de pipelines que son un conjunto de complementos que admite la implementación e integración de canalizaciones de entrega continua. [3]
SonarQube	Calidad código prototipo	Plataforma de código abierto para realizar revisiones automáticas con análisis estático de código para detectar

		errores, deuda técnica y vulnerabilidades de seguridad, buscando una codificación de calidad.[4]
SpiraTeam	Administración proyecto	Solución completa para gestión del ciclo de vida del proyecto, gestiona requisitos, planes, pruebas, tareas, errores y problemas, con una trazabilidad completa desde su inicio hasta su finalización. [5]
Trello	Administración del proyecto	Herramienta visual, flexible y gratuita para gestionar el proyecto y organizar las actividades.[6]
Draw.io	Diseño	Plataforma para crear diagramas de flujo, diagramas de proceso, organigramas, diagramas *UML*, modelos *ER*, diagramas de red [7] y otros necesarios para el diseño del proyecto.
Bizagi modeler	Diseño	Herramienta de *BPM* para crear, optimizar y publicar los diagramas de proceso de trabajo para aumentar la eficiencia y la gobernabilidad de estos en todo el proyecto. [8]
GitHub	Administración prototipo	Es el repositorio del proyecto, contiene todos los archivos del prototipo y almacena el historial de revisiones de cada archivo [9]. Permite mantener la organización e integridad del código en el grupo de trabajo.
Google Drive	Documentación	Plataforma para documentar, almacenar y organizar todo tipo de archivos necesarios para el proyecto, permite la colaboración en línea con otros usuarios para que observen, comenten y editen cualquier archivo o carpeta. [10]
Zotero	Documentación	Herramienta que ayuda a recopilar, organizar y analizar búsquedas de fuentes de información para compartirla de diversas maneras. Permite exportar los datos como referencias formateadas. [11]

Tabla 3: Clasificación y empleo de herramientas

3.3. Plan de Aceptación del Producto

Cada entrega de acuerdo con la *sección 1.4 Entregables*, se debe regir bajo los estándares establecidos y según lo acordado a través de reuniones en las que se realizará seguimiento de cada entregable con el fin de cumplir las expectativas.

Al completar cada hito y al estar de acuerdo los interesados en este caso *ITAC S.A*, se realizarán actas como prueba del cumplimiento y aceptación de dichos entregables.

3.4. Organización del Proyecto y Comunicación

3.4.1. Interfaces Externas

Para el correcto desarrollo del proyecto, es necesario contar con distintos servicios y recursos, que no son propios del grupo de estudiantes. En la *Tabla 4: Interfaces externas* se expone lo dicho anteriormente.

Nombre	Descripción	Responsabilidades	Datos de contacto
ITAC S.A	Empresa con el rol de product owner.	• Proveer herramientas y permisos necesarios para realizar la integración con WS-Guardian.	https://itac.co Tel: 7444822 Carrera 19 No 120 - 71 Bogotá D.C.
Ing. Javier Humberto Galindo, Ph.D	CEO de ITAC (product owner) y director del trabajo de grado.	• Supervisar y orientar en lineamientos de aspectos técnicos durante la realización del proyecto. • Facilitar los requisitos y restricciones para el desarrollo del proyecto.	Correo electrónico: javier.galindo@itac.co
Ing. Jaime Pavlich-Mariscal, Ph.D	Profesor de la asignatura planeación del proyecto final	• Asesorar al grupo de trabajo en todo lo referente a la elaboración de la propuesta de trabajo de grado.	Correo electrónico: jpavlich@javeriana.edu.co Tel: 3208320

Tabla 4: Interfaces externas

3.4.2. Organigrama y Descripción de Roles

Los integrantes del grupo se asignan de acuerdo con *Figura 1: Organigrama,* se especifican roles para tener una mejor distribución de responsabilidades dentro del desarrollo del proyecto. De acuerdo con *SCRUM*, el rol de *Scrum Master* lo tomaría el líder del proyecto y el *Scrum Team* tomará roles para responsabilidades descritas en la *Tabla 5: Roles y responsabilidades*.

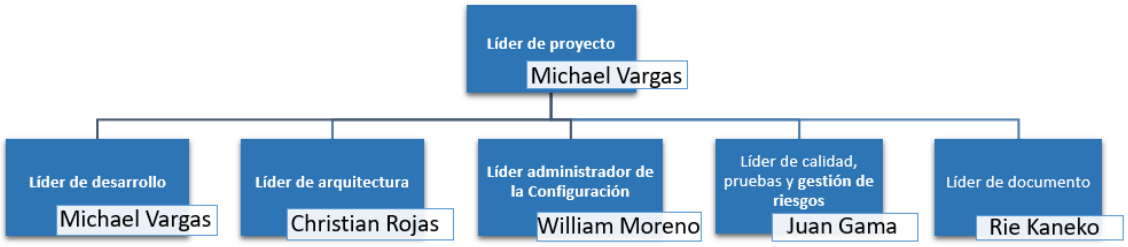

Figura 1: Organigrama

A continuación, se muestran las responsabilidades de cada rol (*Tabla 5: Roles y responsabilidades*):

Rol	Responsabilidades
Líder del proyecto	• Llevar control sobre el desarrollo de actividades propuestas en el cronograma. • Gestionar el proceso *SCRUM* como *Scrum Master* • Eliminar impedimentos. • Verificar la implementación de soluciones. • Asignar tareas.
Líder de Desarrollo	• Participar en los diseños del producto. • Organizar la programación de los *Sprints*. • Participar en la implementación del prototipo, lo que implica asignar cargas de trabajo de acuerdo con la capacidad de desarrollo. • Hacer seguimiento *Sprint Planning* y control de la programación del equipo.
Líder de Administrador de la Configuración	• Llevar control de versionamiento. • Manejar y capacitarse para el uso de las herramientas implicadas. • Establecer la documentación de control. • Manejar el desarrollo y despliegue continuo. • Velar por la calidad del producto incremental.
Líder de arquitectura	• Diseñar modelos necesarios para la construcción de un prototipo con la totalidad de las funciones requeridas y el correcto funcionamiento. • Mejorar de forma continua los diseños, si percibe algún riesgo debe reportarlo al gestor de riesgos.
Líder de Calidad, pruebas y gestión de riesgos.	• Desarrollar el plan de pruebas. • Desarrollar un plan de riegos. • Velar por los estándares de calidad establecidos para el proyecto. • Garantizar la calidad del resultado del producto. • Prestar atención a los riesgos que se presenten o posibles a presentarse, Si es así llevar el control de los mismos.
Líder de documento	• Crear la estructura de los documentos. • Asignar tareas referentes a la documentación • Control y revisión final del contenido documentado.

Tabla 6: Roles y responsabilidades

4. Administración de Proyecto

4.1. Métodos y Herramientas de Estimación

Se utiliza el método de estimación *Planning Poker*. Es una técnica de estimación basada en juicios que se usa en metodologías ágiles para el desarrollo de software, especialmente *SCRUM* y *Extreme Programming*. Se va a calcular a partir de las historias de usuario, siendo esta la unidad de la cual se estiman y desarrollan las funciones de software. Son estimadas en diferentes sesiones por los miembros del equipo de desarrollo [12].

Debido a que la metodología ágil permite que los elementos del *Product Backlog* se actualicen en cualquier momento, se hicieron algunas variaciones. Se realiza la estimación teniendo en cuenta únicamente las tareas que se encuentran en el *Product Backlog* y que aplican para el *sprint* a desarrollar.

4.2. Inicio del Proyecto

4.2.1. Entrenamiento de Personal

- *Seguridad Informática:*

  ITAC S.A proporcionará documentos y presentaciones, para entender las principales técnicas de seguridad que utilizará el producto, y los principios de seguridad que atacan dichas técnicas. También el equipo cuenta con tres personas que tienen conocimiento en áreas de técnicas en seguridad y seguridad informática. Ellos compartirán su conocimiento con el grupo en temas relacionados con ésta área. En caso de no contar con el conocimiento necesario, los expertos de *ITAC S.A* brindarán asesoría.

- *Web Services:*

  Para adquirir conocimientos acerca de seguridad en los WS, el director del trabajo de grado proporcionó al grupo documentos relacionados, en estos se encuentran los principales estándares de seguridad usados en los WS. El grupo cuenta con dos miembros que tienen conocimientos en los WS para aportarlos.

- *SCRUM:*

  Para entender un poco mejor cómo utilizar la metodología ágil, se agendó una capacitación con *Karen Zárate,* asesora de *ITAC S.A*, profesional de Ingeniería de Sistemas y Especialista en Gerencia Informática, con conocimiento y experiencia en dirección de equipos.

- *Java:*

  Este lenguaje de programación es utilizado por todos los miembros del grupo, ya que a lo largo de la etapa académica lo han aprendido y manejado. Es el que se decidió utilizar para el desarrollo del proyecto, ya que no se requiere ninguna capacitación.

- *Java Reflection:*

  Permite manipular e inspeccionar clases e interfaces en tiempo de ejecución, sin conocer los nombres de las clases con las que estamos trabajando. Para utilizar esta *API*, los miembros del grupo deben buscar información y explorar el funcionamiento del código por cuenta propia. Cada uno debe hacer su investigación por sí mismo y después compartir conocimientos de lo que se logró.

- *Java Code Conventions:*

  Usar convenciones *Java* es exigido por el *Product Owner*, también hace parte de calidad de código, ya que con esto el mantenimiento del software es más fácil y entendible, para quienes le darán soporte. El líder de calidad dará una capacitación al equipo sobre las convenciones al programar con *Java*.

- *SpiraTeam*

  Esta herramienta permite la gestión de tareas, desde historias de usuario hasta desarrollo de código. El líder de administración de la configuración quién será capacitado por expertos proporcionados por *ITAC S.A* en la herramienta, se encargará de capacitar al equipo en el uso de esta.

4.3. Planes de Trabajo del Proyecto

4.3.1. Descomposición de Actividades

En la siguiente figura (*Figura 2: Work Breakdown Structure*) se puede observar una estructura de descomposición de tareas *Work Breakdown Structure* en forma de jerarquía. Se encuentran las principales actividades en las que se va a trabajar, durante el desarrollo del proyecto.

En el cuadro verde se ve el título del proyecto. En los cuadros rojos se encuentran las fases en las que se dividió el proyecto y en los cuadros rosados todas las actividades asociadas a sus respectivas fases.

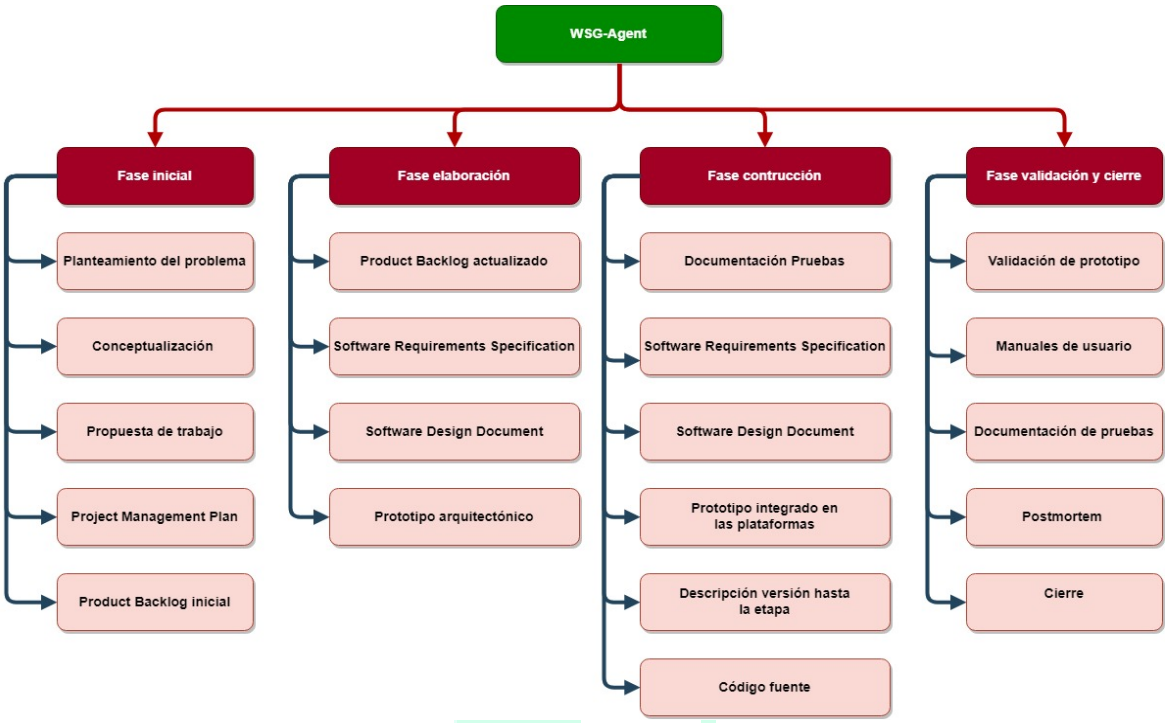

Figura 2: Work Breakdown Structure

4.3.2.Calendarización

En las siguientes figuras: *Figura 3: Colores representativos de las fases y Figura 4: Diagrama de Gantt* se ilustran respectivamente las fases con su convención de color asociada y los entregables, módulos del proyecto y fechas tentativas por semana hasta la entrega del proyecto.

FASE
Inicial
Elaboración
Construcción
Validacion

Figura 3: Colores representativos de las fases

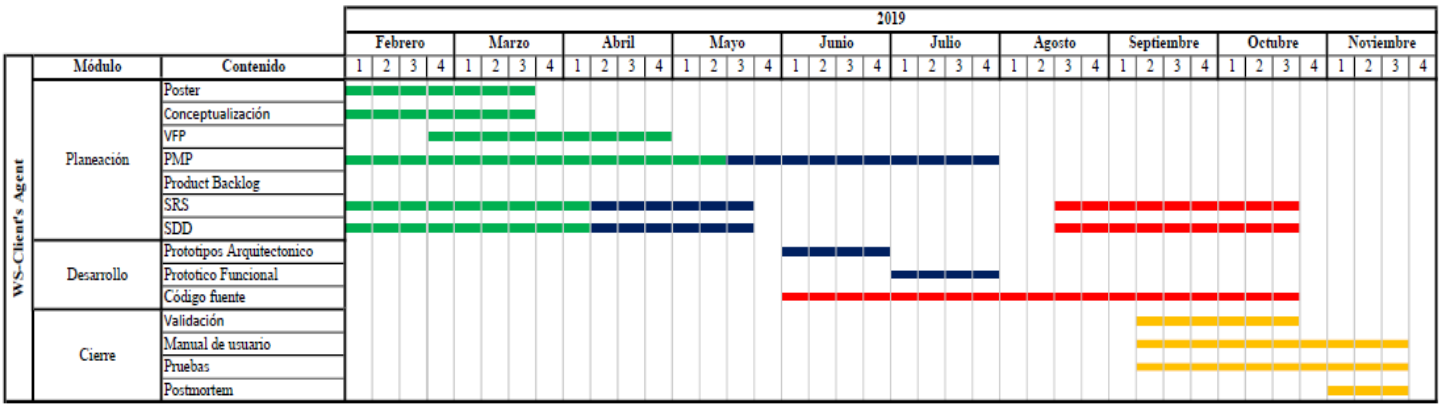

Figura 4: Diagrama de Gantt

5. Proceso Ingeniería de Requerimientos

El proceso de ingeniería de requisitos es fundamental para el desarrollo del proyecto, ya que, si este no se hace de la manera adecuada puede ocasionar problemas al momento de la entrega del software, mediante este proceso se analiza, específica y validan las historias de usuario.

Análisis

El análisis se llevó a cabo en la fase inicial del proyecto, en esta se hicieron varias reuniones con el *Product Owner* y *Stakeholders*, quien explicó al *Scrum Team* cuáles eran las necesidades y el problema existente. A partir de esto se hizo una conceptualización de información relevante para un mayor entendimiento del problema que se requiere resolver, una vez entendido el problema se plantea una posible solución al *Product Owner* para validar que la propuesta coincida con lo requerido, y así poder establecer lineamientos importantes.

Especificación

En esta parte del proceso el equipo utiliza la técnica entrevista abierta para levantamiento de historias de usuario, el *Scrum Team* realiza preguntas al *Product Owner* y *Stakeholders* sobre el sistema existente (*WS-Guardian*), también sobre el sistema a desarrollar (*WSG-Client's Agent*), con esto se logra una mejor comprensión de las necesidades del proyecto. Después de las entrevistas el *Scrum Team* se reúne para definir las historias de usuario con la información recolectada, diferentes miembros escriben las historias según el formato.

Validación

La primera validación se realiza entre el *Scrum Team* quienes revisan las historias y sus especificaciones, para verificar que no tengan inconsistencias, errores y que realmente estén acorde a la solución del problema, así como las necesidades del *Product Owner*. Se realiza una segunda validación con el *Product Owner* y *Stakeholders*, se expone ante ellos las historias de usuario desarrolladas para verificar que estén completas y sea eso lo que se pide.

En la figura X muestra en modo de diagrama el proceso descrito anteriormente.

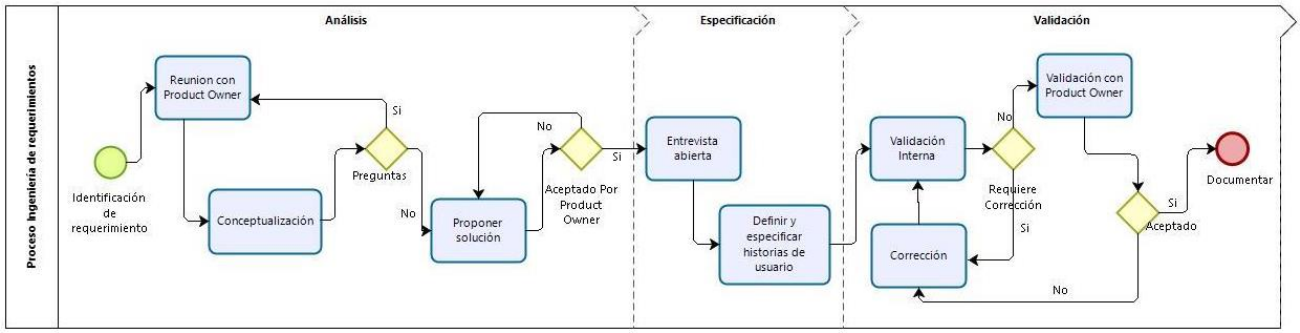

Figura 5: Diagrama Proceso ingeniería de requerimientos

6. Monitoreo y Control de Proyecto

Teniendo en cuenta que en el modelo de ciclo de vida del proyecto se usa una metodología ágil (véase *sección 3.1 Modelo de ciclo de vida*), se requiere constante interacción con el *Product Owner*. Éste, al estar al tanto del progreso del equipo va a tener la oportunidad de darle el visto bueno a lo realizado y si es necesario hacer sugerencias o solicitar cambios en el momento oportuno.

6.1. Administración de Requerimientos

Debido a que los cambios de los requerimientos se pueden presentar en cualquier momento, se maneja un versionamiento con el objetivo de tener trazabilidad por cada uno de ellos. Cuando surge el cambio ya sea por solicitud del *Product Owner* o que el equipo lo vea conveniente, es importante estar preparados y tener una planificación del proceso a realizar, una vez se presente la necesidad de generar estos cambios. En el Anexo 1: Diagrama BPMN Administración de Requerimientos se muestra el proceso que se va a ejecutar en el momento que se requiera algún cambio en los requerimientos.

El proceso puede ser iniciado tanto por el equipo, como del *Product Owner*. En este momento se debe solicitar el cambio de requerimientos y agendar una reunión iniciando así la fase de *Detección*, donde mediante una reunión, se justificará y dará la oportunidad para conocer mejor las razones detrás de este cambio en los requerimientos.

Después de la fase de *Análisis,* el equipo debe esperar la decisión del product owner y así cambiar el proceso dependiendo de la respuesta. Si el cambio es aprobado, el equipo debe realizar un plan de cambio correspondiente a la fase de *Planificación*. Una vez que se realiza el plan, se presenta al *Product Owner* y en el caso de que haya algún inconveniente, el equipo debe replanificar hasta que tenga su aprobación.

Cuando ocurre una modificación en el requerimiento se debe tener en cuenta como mínimo lo siguiente:

- Versionamiento del requerimiento
- Requerimientos asociados
- Tiempo requerido para el cambio

Si no es aceptado el cambio, es responsabilidad del equipo buscar soluciones a los inconvenientes presentados y si es necesario, replantear el cronograma con el fin de cumplir con el requerimiento implicado y afectar en lo menos posible el avance del proyecto.

6.2. Monitoreo y Control de Progreso

Para mantener un buen ritmo de trabajo y tener un estándar entre todos al momento de hacer la revisión del proyecto, se deberán manejar las siguientes medidas y reglas de equipo:

Medidas

Los avances del proyecto se verifican por cada entrega de *Sprint* durante un periodo de 1 a 4 semanas, y divididos (los avances) según las historias de usuario. Las medidas se realizan según los criterios de aceptación que tiene cada historia de usuario.

- **Prioridad de cada sprint:** El impacto que tiene la actividad ante otras actividades y su importancia, si se debe realizar de manera inmediata o no.
- **Complejidad de cada sprint:** La dificultad que tiene la actividad, la cual es calculada haciendo uso de la herramienta *SPIRA*.
- **Tiempo requerido:** El tiempo necesario para terminar la actividad correctamente y en su totalidad (horas de trabajo).

Control del progreso

- **Reuniones internas semanales:** Se habla de los temas a tratar en la semana, se discute el sprint, se revisan los comentarios de los miembros del grupo sobre el trabajo realizado y finalmente se realiza una retroalimentación.
- **Reuniones semanales con el Product Owner:** El equipo realiza una reunión cada 8 o 15 días con el product owner revisando los trabajos realizados, resolviendo dudas y obteniendo retroalimentación del product owner.
- **Reuniones adicionales:** En el caso de que se necesite una reunión adicional con los expertos del tema, se le comunica al *Product Owner* y se agenda la reunión, ya sea para resolver dudas específicas o reforzar conocimientos para completar las tareas.
- **Minuta de reuniones:** Este es un documento donde quedan registradas las reuniones. Aquí se deja constancia de lo que se habló, las tareas a realizar y los puntos para tener en cuenta dentro del proyecto.
- **Herramientas:** Como se mencionó en la *sección 3.2 Lenguajes y Herramientas*, se hace uso de esta plataforma para controlar el progreso, dado que es posible dejar evidencia del plan sobre el que se trabaja el proyecto y los responsables de cada tarea (*Sprint*).

Acciones correctivas

- **Penitencias:** Cuando un integrante del equipo no cumple con la fecha de entrega del trabajo sin una justificación válida, se realiza el llamado de atención respectivo para evitar la repetición de la falla en el futuro.
- **Ajuste en cronograma:** En el caso de que ocurra algún caso inesperado, se debe realizar una reunión interna y ajustar el cronograma para evitar o mitigar el retraso de otras actividades.
- **Análisis de resultados:** Se debe analizar porqué ocurrieron las faltas y hablar con el equipo para no reincidir en ella.

6.3. Cierre del Proyecto

Para la terminación de cada fase, se hará una entrega de análisis *Post-Mortem* teniendo así una visión de lo que se hizo. Con esto se conoce qué aspectos del proyecto y de la organización deben mejorarse, además de resaltar los puntos fuertes del mismo para asegurar la calidad del producto final.

Para cada entrega de los documentos, se hace una revisión interna de corrección de estilo para verificar la cohesión, coherencia, ortografía y gramática, y así, luego presentar al *Product Owner* la versión final. Una vez aceptada la versión del documento, ésta se usa como base de arranque para el proceso de desarrollo del proyecto.

El prototipo a entregar debe pasar por un proceso de pruebas de calidad, donde se revisan tanto los requerimientos funcionales como no funcionales. Debe ser aprobado por el *Product Owner*, teniendo en cuenta su implementación según el objetivo general y los específicos junto con la tabla de requerimientos. Para finalizar se realiza oficialmente la entrega del prototipo junto con su respectiva documentación.

7. Entrega del Producto

La entrega del producto contendrá: tres archivos .JAR donde estará contenido el módulo de administración del agente, y dos módulos de implementación de dos técnicas de seguridad como se me menciona en los objetivos del proyecto (Véase sección 1.2 Propósito, Alcance y Objetivos). Junto a estos archivos se entregará un repositorio de Git, donde estará todo el código fuente del agente con la documentación de este (*SPMP*, *SRS*, *SDD* y documento de pruebas). Para que el proyecto pueda continuar en la empresa *ITAC S.A*, se entrega un manual de usuario donde se especifican las funcionalidades del agente y el procedimiento para su instalación, configuración y administración.

Posterior a la entrega de los artefactos, se realizará una capacitación del equipo de *ITAC S.A*, con el fin de que el proyecto pueda continuarse; adicionando las otras técnicas de seguridad, mostrándoles el código fuente y cómo es el funcionamiento del agente. De esta forma se busca lograr una versión de producción que pueda ser ofrecida e implantada en los clientes de *WS-Guardian*.

8. Proceso de Soporte

En esta sección se describen los procesos que soportan el desarrollo y ejecución del proyecto con planes y medidas de corrección para enfrentar limitaciones, problemas y riesgos.

8.1. Ambiente de Trabajo

Reglas de trabajo

Las reglas de trabajo se pueden ver en el Anexo 2: Reglas de trabajo. En este se detallan una a una las reglas definiendo medidas correctivas y responsables. El proceso y las herramientas de control se detallan en la *sección 6.2. Monitoreo y Control de Progreso*.

8.2. Análisis y Administración de Riesgos

Identificación de riesgos:

Los riesgos identificados se encuentran en el anexo Anexo 3: Control y mitigación de riesgos, en éste se evidencia la descripción, mitigación, probabilidad y nivel de impacto de cada riesgo identificado. Esta plantilla también es usada por el líder de calidad, pruebas y gestión de riesgos; para la gestión y prevención de futuros problemas.

Momentos del proceso de control y mitigación de riesgos:

- Reuniones programadas y extraordinarias (en caso de ser un problema significativo).
- Cada vez que el equipo de trabajo realice una retroalimentación (*feedback*).

Control y mitigación de riesgos

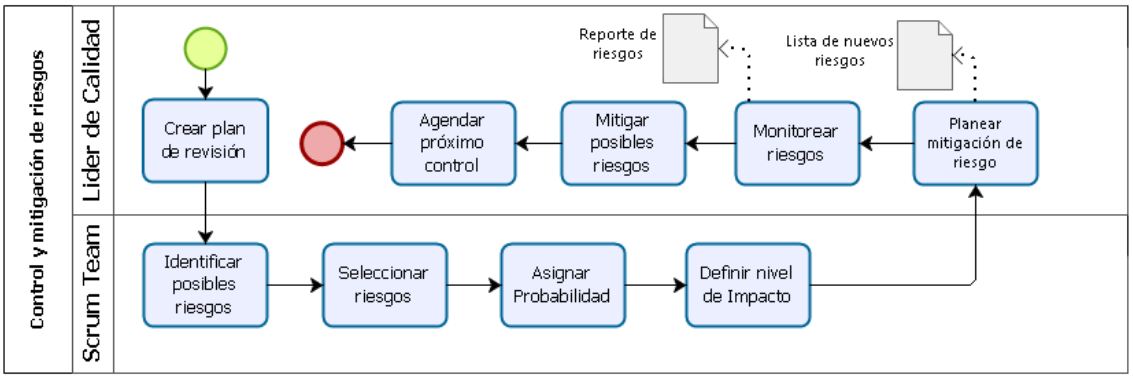

Figura 6: Modelo BPMN de Control y Mitigación de Riesgos

En la *Figura 6: Modelo BPMN de Control y Mitigación de Riesgos* se muestra el proceso que se lleva a cabo cada vez que se hace control de riesgos. El proceso inicia con los *Scrum Team* identificando nuevos riesgos y seleccionando los que más influyen en el desarrollo del proyecto. Si se encontraron nuevos riesgos se les asigna una probabilidad, nivel de impacto y el plan para mitigarlos. Luego, el líder

que está a cargo procede a monitorear la presencia de los riesgos en el último periodo de trabajo.

Según los resultados obtenidos anteriormente, se aplica el plan de mitigación y se reporta en la minuta de control. Éste proceso se hace cada vez que se finalice el ciclo de Sprint y es guiado por el líder de gestión de riesgos.

8.3. Administración de Configuración y Documentación

Spira y *Trello* son las herramientas usadas para administrar la configuración y documentación.

Ítems de configuración

- **Código fuente:** Es la codificación del prototipo desarrollado para la solución propuesta.
- **Documentos de diseño:** Contienen diagramas de diseño y especificaciones de los mismos.
- **Plataformas:** Son los entornos encargados de ejecutar aplicaciones desarrolladas en el lenguaje *Java*.
- **Librerías:** Son algoritmos que solucionan partes importantes en la funcionalidad del agente. Se deben buscar e integrar con el código fuente.
- **Despliegue de servidores:** Es necesario indagar en el despliegue de servidores, para la comunicación por mensajes *http*.
- **Archivos de configuración:** Es la configuración propia del agente, usada para realizar la funcionalidad.
- **Datos de prueba:** Es la información ficticia tomada para realizar las pruebas sobre el prototipo.
- **Manual de Usuario:** Es las guías que especifica cómo usar el aplicativo de forma correcta.
- **Reportes de pruebas:** Son los documentos donde se reflejan resultados y análisis de las pruebas realizadas al aplicativo. Estos se toman como evidencias.

Administración de Desarrollo

La implementación se llevará a cabo con desarrollo continuo, apoyándose en el servidor *Jenkins*, mencionado en la *sección 3.2. Lenguajes y Herramientas*, para la implementación.

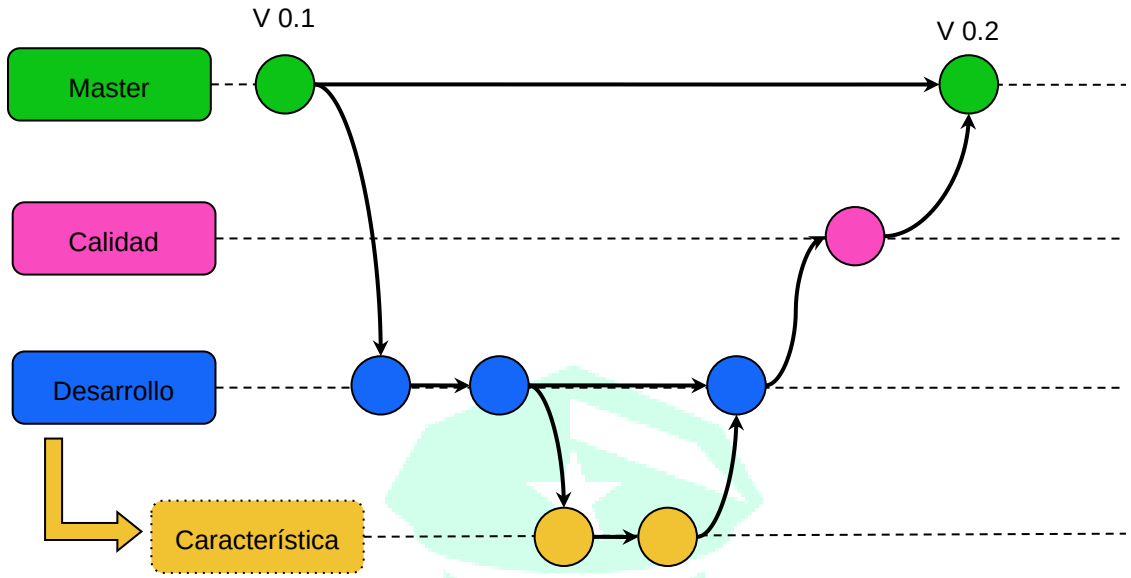

Figura 7: Diagrama de ramas de desarrollo

La *Figura 7: Diagrama de ramas de desarrollo* muestra las ramas de trabajo y el respectivo flujo que se seguirá para la implementación y el aseguramiento de calidad.

Las ramas de administración son:

- **Master:** En esta rama se inicia el versionamiento del producto, para luego enviarlo a desarrollo. En esta rama va a persistir la versión final del código fuente.
- **Calidad:** Es la rama usada para realizar las pruebas de calidad definidas. Una vez que la nueva funcionalidad pasa la fase de pruebas, se hará el *push* en la rama *Master*.
- **Desarrollo:** Sobre esta rama se va a trabajar el proceso de implementación del prototipo. Se divide en subramas *Característica* para separar módulos de trabajo. Una vez se complete una funcionalidad se hace *merge* con la rama *Calidad* para pasar a la siguiente fase.
- **Característica:** Es una subrama de *Desarrollo* contenida de la implementación de una funcionalidad específica. Una vez se completa la codificación se hace *merge* con la rama *Desarrollo*.

Administración de documentos

La documentación se realizará en la herramienta *Google Drive*, la cual almacena y controla el versionamiento con un histórico de cambios. En el momento de su modificación, se debe seguir uno de los siguientes procesos:

- **Cambios internos:** Este proceso es ejecutado en el momento de hacer cambios leves en secciones o partes específicas. El control de versiones es controlado automáticamente por Google Drive.

- **Cambios externos:** Este proceso se ejecuta cuando se van a realizar cambios significativos en la estructura o contenido por parte de los *stakeholders,* generando una nueva versión y actualizando la tabla de historial de cambios. En primera instancia, un miembro del *Scrum Team* manifiesta la intención de cambio al grupo de forma verbal en la reunión más próxima.

  Posteriormente, los miembros del equipo deben dar su valoración o solicitar la intervención de un tercero (Experto o *Product Owner*), para aclarar puntos de posibles cambios. Con base en esa valoración, el *Scrum Master* decidirá si se consulta al experto o se continúa con el cambio.

  Si el cambio no necesita la consulta de un externo, se realizará un proceso de control y mitigación de riesgos, dando como resultado que tan viable es realizar el cambio. Si el cambio es aprobado, es obligatorio que dos miembros del equipo diferentes a el o los que propusieron el cambio, revisen redacción, ortografía, estructura, coherencia y cohesión del cambio a realizar.

8.4. Control de Calidad

El control de calidad se realiza con la ejecución de los siguientes procesos.

Revisión de documentos

El diagrama *BPMN* de este proceso puede consultarse en el Anexo 4: Diagrama BPMN Revisión de documentos. Se realiza cada vez que se elabora un documento para su entrega y está a cargo del líder de documentación.

Inicia cuando el líder asigna las secciones del documento a sus responsables, luego cada miembro identifica sus secciones ubicándolas en su tablero de *Trello*. En caso de que se cumpla el tiempo asignado para la revisión y no se haya terminado, se notifica la tarea como atrasada y se continúa con los otros ítems. Por otro lado, cuando la sección esté lista para revisar, la pareja responsable debe realizar la respectiva revisión y agregar comentarios. Dichos comentarios son tomados por el redactor para ajustar el documento.

Si es necesaria una revisión externa, se hace la solicitud al *Product Owner*, y se reciben los comentarios por el mismo medio. El *Scrum Team* toma los comentarios y ajusta el documento. Luego de esto, el líder genera la versión final del documento y el informe de revisión.

Validación del estándar de codificación

Como bien se sabe, se debe codificar con la técnica *Java Code Conventions* mencionada en la *sección 4.2 Inicio del proyecto*. Ya que la implementación se va a realizar en parejas, la revisión del código estará bajo la responsabilidad de los asignados. Dicho proceso se llevará a cabo con la herramienta SonarQube definida en la *sección 3.2 Herramientas y lenguajes*, la cual muestra las inconsistencias que deben ser atendidas.

Pruebas de aceptación

El diseño BPMN de este proceso se puede apreciar en el Anexo 5: Pruebas de aceptación. Se ejecuta cada vez que se implementa una nueva funcionalidad, para asegurar que cumple correctamente los requisitos. El líder de calidad es el encargado de ejecutar esta etapa.

A lo largo del proceso se define un plan de pruebas y se ejecutan obteniendo resultados. En el caso de que no esté correcta se envía un reporte de los fallos encontrados. Si la implementación está correcta, se genera un informe donde se evidencian las pruebas y resultados que lo respalden.

9. Referencias

[1] «Server JRE (Java SE Runtime Environment) 8 Downloads». [En línea]. Disponible en: https://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154.html. [Accedido: 06-may-2019].

[2] «NetBeans IDE - Overview». [En línea]. Disponible en: https://netbeans.org/features/. [Accedido: 06-may-2019].

[3] «Jenkins User Documentation». [En línea]. Disponible en: https://jenkins.io/doc/. [Accedido: 07-may-2019].

[4] «Open Source Static Code Analysis | SonarQube». [En línea]. Disponible en: https://www.sonarqube.org/about/. [Accedido: 11-may-2019].

[5] «Features of SpiraTeam, Our ALM Suite. Try it Today!» [En línea]. Disponible en: https://www.inflectra.com/SpiraTeam/Highlights.aspx. [Accedido: 11-may-2019].

[6] «Acerca de | ¿Qué es Trello?» [En línea]. Disponible en: https://trello.com/es/about. [Accedido: 11-may-2019].

[7] T. Groß, «draw.io - Diagrams For Everyone, Everywhere», *draw.io*. .

[8] «Software BPMN para el modelamiento de procesos - Descarga gratuita». [En línea]. Disponible en: https://www.bizagi.com/es/productos/bpm-suite/modeler. [Accedido: 11-may-2019].

[9] «About repositories - GitHub Help». [En línea]. Disponible en: https://help.github.com/en/articles/about-repositories. [Accedido: 12-may-2019].

[10] «Google Drive: nuevas funciones y ventajas de Google Cloud Storage». [En línea]. Disponible en: www.google.com/intl/es_ALL/drive/using-drive/. [Accedido: 12-may-2019].

[11] «Zotero | About». [En línea]. Disponible en: https://www.zotero.org/about/. [Accedido: 12-may-2019].

[12] T. H. Viljan Mahnicˇ ∗, «On using planning poker for estimating user stories», p. 10.