Plan de Aseguramiento de la calidad para RiskMap

Fecha de elaboración:

08 de mayo de 2022

Maestro: Edwin Jesús León Bojórquez

Versión 0.2

Miembros del equipo:

Ana Pérez

Lenin Rosas

Pablo Rosas

Control de Documentación

Control de Configuración

Título:	Plan de Aseguramiento de la calidad para RiskMap
Referencia:	https://github.com/PabloR9080/Documentacion-PlanSQA
Autor:	Pablo Rosas
Fecha:	08 de mayo del 2022

Histórico de versiones

Versión	**Fecha**	**Estado**	**Responsable**	**Nombre de archivo**
0.2	18/05/2022	A	Equipo	SQAPlanV0.2.docx
0.2	15/05/2022	B	Equipo	SQAPlanV0.2.docx
0.1	08/05/2022	A	Equipo	SQAPlanV0.1.docx
0.1	08/05/2022	B	Equipo	SQAPlanV0.1.docx

Estado: (B)orrador, (R)evisión, (A)probado

Histórico de cambios

Versión	**Fecha**	**Cambios**
0.2	18/05/2022	Se definieron las herramientas, así como las metodologías a utilizar. Los procesos para el control de código, medios y documentos finales se han especificado en esta versión. Por último, se han agregado las acciones correctivas, así como los métodos de entrenamiento, en el caso necesitar una preparación para los integrantes de la organización.
0.2	15/05/2022	Se realizaron los puntos 10,11 y 12 remarcando los puntos del control de distintos objetos dentro del plan.
0.1	09/05/2022	Creación del repositorio para el control de versiones en Github y vinculación de los documentos referenciados con el repositorio.
0.1	08/05/2022	Se agregan algunas herramientas, en este caso herramientas que serán utilizadas para las pruebas
0.1	09/05/2022	Se realizan los puntos 6-7 para definir cómo se realizarán las auditorías y revisiones, así como también las pruebas que se realizarán a lo largo del plan de aseguramiento de la calidad para el software RiskMap.
0.1	09/05/2022	Se ha realizado la definición del punto 1-5 del documento en donde se describe el propósito del documento actual y los documentos referenciados para le creación de este.

		Por otra parte, se empieza a describir los puntos de gestión, documentación y estándares, prácticas, convenciones y métricas para comenzar a definir el plan de SQA.
0.1	08/05/2022	Se define el proyecto sobre el cual se aplicará el plan de SQA.

Tabla de contenidos

1. Propósito

Con el documento actual se pretende cubrir las actividades, así como otras definiciones del SQA, de igual manera este documento busca ser de utilidad para ser implementado como plan actual del aseguramiento de la calidad para el software RiskMap utilizado para ayudar a la seguridad de las localidades en donde se utilice, generar reportes y que se les dé el seguimiento a través del proceso correcto, generando una mayor percepción de seguridad para todos los segmentos de la población.

Este plan se trabajará durante todo el ciclo de vida ya que se especificarán las estrategias a llevar a cabo para controlar la calidad, las formas de validar, los procesos para verificar que los productos que se generan están de acuerdo con los lineamientos establecidos en el modelo de procesos de software en caso de tener uno.

2. Documentos referenciados

IEEE Std 1028-2008. Standard for Software Reviews and Audits. Este estándar dará las pautas para realizar las revisiones y auditorias sobre los productos de software elegidos para revisión.

IEEE Std 830-1998. Recommended Practice for Software Requirements Specifications Estándar que contiene un formato completo que cubre todas las características que se debe seguir en la especificación de requerimientos.

IEEE Std 1016-2009. Stantard for Information Technology – Systems Design – Software Design Descriptions: establece el contenido que el documento de la descripción del diseño de software debería incluir.

IEEE Std 829-1998. Standard for Software Test Documentation. Establece el contenido y estructura que deben seguir un plan de pruebas.

IEEE Std 1063-1987. Standard for Software User Documentation. Guía para el contenido, consideraciones e información que debe tener la documentación del manual de usuario.

IEEE Std 828-1990. Standard for Software Configuration Managment Plans. Estándar que establece el contenido recomendado que se debería incluir para tener un buen plan de gestión de la configuración de software.

IEEE Std. 1008. Standard for Software Unit Testing. Estándar que se implementará como base para la realización de las pruebas unitarias de los componentes del sistema, para que estas estas pruebas sean de calidad y estén bien organizadas.

ISO/IEC 14764-2006. Software Life Cycle Processes – Maintenance. Estándar dedicado a describir el proceso de mantenimiento y servirá como guía para la creación del Plan de Mantenimiento, además de incluir información para la aplicación del plan, el control, revisiones y evaluación de este proceso.

ISO/IEC 15939-2007. Systems and software engineering – Measurement process. Estándar sobre el que se basará el proceso de medición y se recabarán los datos. Este estándar describe las actividades a llevar a cabo para el proceso de medición, por otra parte, podemos encontrar en este documento el ciclo de planeación, ejecución y evaluación del proceso de medición.

Sánchez, P., (2015), *Pruebas de Software. Fundamentos y Técnicas.* Recuperado el 16/05/2022 https://github.com/PabloR9080/Documentacion-PlanSQA/blob/main/Documentos-Estandares/Pruebas-Software-Fundamentos-Tecnicas.pdf

3. Gestión

3.1 Organización

La actividad de SQA se desglosa en distintas tareas a lo largo del ciclo de vida del software a realizar. El propósito de este equipo de acuerdo con el plan es minimizar los errores que el equipo comete a través de las tareas que son descritas en este plan.

Por lo cual, una organización es requerida para la coordinación en realizar las tareas, generando así una noción sobre las actividades que sea han realizado, dejando pendiente las actividades restantes.

El equipo de SQA es un grupo de personas con mayor importancia sobre el proyecto ya que, el negocio no podría avanzar exitosamente sin este. Es vital la comunicación entre los integrantes de este para que prevalezca el éxito en el proyecto.

3.2 Tareas

3.2.1Tareas generales

Criterio de entrada: ninguno.

Criterio de salida: ninguno.

Tareas

- Revisar y evaluar la calidad de las actividades del proyecto conforme a los lineamientos del aseguramiento de la calidad.
- Planificar todas las tareas definidas en el plan de SQA.
- Mantener contacto y cercanía con el cliente a lo largo del ciclo de vida del proyecto.

3.2.2 Requerimientos

Criterio de entrada: ninguno.

Criterio de salida: documento de especificación de requerimientos y plan de pruebas.

Tareas

1. Asegurar que los stakeholders e integrantes de la compañía estén presentes en los procesos para la educción de los requisitos.
2. Asegurar que los integrantes del equipo reciban una capacitación en el caso de ser necesario.
3. Asegurar que los miembros del equipo que estará involucrado en la fase de requerimientos tengan los conocimientos suficientes para llevar a cabo esta fase.
4. Monitorear la ejecución de las técnicas de elicitación elegidas.
5. Auditar el documento de especificación de requerimientos y plan de pruebas.

6. Asegurarse que se lleven a cabo las revisiones y auditorías necesarias documentadas en el punto 6 del actual plan de SQA.

7. Evaluar los productos generados en la fase de requerimientos para su validación.

3.2.3 Análisis y Diseño

Criterio de entrada: documento de especificación de requerimientos y plan de pruebas.

Criterio de salida: documento de especificación de diseño, documento de diseño detallado.

Tareas

1. Asegurar que los integrantes del equipo reciban una capacitación en el caso de ser necesario.
2. Evaluar y revisar el proceso preliminar de documentar la especificación de diseño.
3. Asegurarse que se lleven a cabo las revisiones y auditorías necesarias documentadas en el punto 6 del actual plan de SQA para la fase.

3.2.4 Implementación

Criterio de entrada: documento de especificación de requerimientos, plan de pruebas y documento de especificación de diseño.

Criterio de salida: código fuente, pruebas unitarias, entorno para despliegue.

Tareas

1. Asegurar que el programador cuenta con los conocimientos suficientes sobre el proyecto y la implementación a realizar.
2. Revisar que el proceso de implementación sea ejecutado de manera correcta siguiendo el documento de especificación de requerimientos.
3. Evaluar la consistencia del proceso con lo que se tiene documentado
4. Realizar revisiones periódicas en el proceso.
5. Auditar el control de cambios en el código fuente o los productos de software que se hayan generado en iteraciones anteriores.

3.2.5 Mantenimiento

Criterio de entrada: peticiones de cambio, usuarios para usar el software y reportar los problemas, plan de pruebas.

Criterio de salida: problemas en el código y cambios para pasar a producción.

Tareas

1. Realizar una revisión posterior a la implementación.
2. Asegurarse que los involucrados en la fase tengan los documentos necesarios para empezar el mantenimiento.
3. Revisar y controlar los cambios que se estén realizando sobre el código fuente.
4. Realizar una verificación antes de proceder con la validación sobre los cambios hechos.
5. Asegurarse que las pruebas sean realizadas conforme al plan de pruebas definido.

3.3 Roles y responsabilidades

A continuación, se presentan las responsabilidades de cada uno de los roles desde la perspectiva de aseguramiento de la calidad:

Rol	Abreviatura	Responsabilidad
Dirección de operaciones	DIOP	Encargado de establecer un programa de calidad para los procesos, revisión y aprobación de planes de aseguramiento de calidad. Además, deberá seleccionar al personal responsable del aseguramiento de la calidad e identificar los factores o metas de calidad a cubrir.
Responsable de la unidad de SQA	RUSQ	Rol responsable de verificar y validar que las actividades sean realizadas acorde a lo establecido en el plan de SQA.
Líder de proyecto	LPR	Aplicar el programa de calidad, identificar las actividades necesarias para cumplir, revisar y aprobar el plan de calidad del proyecto.

		Atiende a los reportes de problemas generados durante verificaciones y auditorias y mantiene los planes asociados al proyecto.
Coordinador de desarrollo	CD	Verificar que los roles AN, PR e IPR cumplan con las actividades designadas de acuerdo al plan de calidad establecido.
Analista	AN	Dar revisión al plan de aseguramiento de calidad del proyecto, así como identificar, implementar y evaluar los factores de calidad definidos en el plan SQA.
Programador	PR	Implementar las especificaciones de diseño de software, revisar y comentar el plan de SQA, verificar los factores de calidad a implementar en el sistema.
Ingeniero de pruebas	IPR	Implementar el programa de calidad definido para el proyecto, verificar y validar cada una de las funcionalidades acordes a la especificación de requisitos.
Administrador de la configuración de software	ACS	Llevar a cabo todas las prácticas relacionadas con la administración de la configuración y control de cambios.

4. Documentación

La implementación de todo proceso de desarrollo trae consigo la elaboración de documentos cuya calidad debe de ser asegurada. Para esto, será importante definir la manera en la que serán revisados o auditados, así como los criterios de revisión o auditoria que deberán aplicarse.

El estándar IEEE 730 – *Standard for Software Quality Assurance Plans* señala que *p*ara asegurar que la implementación de software satisface los requisitos técnicos, se requiere como mínimo la siguiente documentación:

4.1 Especificación de requisitos de software (ERS)

Se detallan todos los requerimientos del sistema, deben de ser claros y no dejar ningún tipo de ambigüedad (cada requisito debe tener una única interpretación), además debe ser redactado en lenguaje informal, debido a que el cliente debe ser capaz de entender gran parte del documento y el equipo de desarrollo debe tener la información suficiente para la creación del sistema. Incluye distintos apartados que permiten tener un panorama completo de cómo se desarrollará el sistema:

- **Introducción**: Describe el contenido que tendrá todo el documento, se pone en perspectiva la problemática que se quiere solucionar e incluye más subsecciones que harán que cualquiera que lea el documento entienda su propósito y del proyecto.
- **Descripción general**: Se da un contexto de todo el sistema y de los factores que se relacionan con el producto como sus interfaces (de usuario, del sistema, de software, hardware y de comunicación), las funcionalidades que tendrá, las características de los usuarios, restricciones. Es importante mencionar que en este apartado no se habla ni de diseño y tampoco se detallan los requisitos.
- **Requisitos específicos**: Se habla a detalle de todos los requerimientos, deben de estar muy bien descritos y que sean entendibles para cualquier persona. Se incluyen los requisitos funcionales, ya sea que estén relacionados según el tipo de usuario, algún objetivo o jerarquía, los relacionados a las interfaces externas y con el rendimiento al que se desea llegar. También se especifican restricciones que deben ser tomadas en cuenta para el diseño y se describe otros atributos del sistema como la seguridad y mantenibilidad.

  Según el estándar *IEEE 830-1998* para la especificación de requerimientos, los requisitos deben ser:

  1. No ambiguos
  2. Completos
  3. Correctos
  4. Consistentes
  5. Clasificados
  6. Verificables
  7. Modificables
  8. Trazables.

- **Apéndices**

- **Índice**

Verificación y Validación

Al final del desarrollo de este documento se deberá generar un reporte del ERS en donde se marquen las deficiencias encontrar y aplicar un plan de corrección, después de crear este plan, se deberá agendar las actividades descritas en este. Además, se deberán seguir las revisiones definidas en el punto *6. Revisiones y auditorías* para el punto correspondiente del documento de especificación de requisitos.

4.2 Descripción de diseño de software (DDS)

Describe las características del sistema en cuanto a las interfaces externas e internas, bases de datos y en general todo lo relacionado con el diseño y que serán dirigidos por los requerimientos del ERS para que estos sean satisfechos. Incluye distintos apartados que sirven para que quede una idea más clara de cómo se va a organizar todo el sistema y las relaciones que habrá entre cada componente.

El estándar IEEE 1016-2009 define distintos puntos de vista de diseño según su uso, algunos de esos puntos de vista son el lógico, las dependencias, de las interfaces, estructuras, interacciones, entre otros. Sin embargo, en un documento de diseño de software debería incluir:

1. **Diseño de la arquitectura del sistema**: Incluye apartados donde se describe los subsistemas y las tareas que deben realizar, así como las interfaces de cada una.
2. **Diseño de los datos:** Se detalla las relaciones entre lo datos, las entidades y sus atributos y además se incluyen diagramas de entidad relación para mayor detalle.
3. **Diseño detallado:** Describe aspectos del desarrollo del sistema, como el rendimiento, espacio en memoria, estándares que se van a seguir.

Verificación y validación

Se realizará una revisión de diseño como lo define el punto 6 de este documento.

El encargado de la revisión deberá asegurarse que el diseño en esta revisión satisface los requerimientos y si la calidad conseguida es buena.

4.3 Planes de verificación y validación de software (PVVS)

Este plan describe el proceso y los criterios que se consideran para poder validar y verificar el software, con el objetivo de tener la seguridad de que el producto cumple con todos los requerimientos establecidos y que estén correctamente desarrollados.

El estándar IEEE 829 establece que un plan de pruebas debería abordar temas como:

1. Ítems para probar
2. Características que serán probadas
3. Características que no serán probadas
4. Enfoque
5. Casos de prueba
6. Tareas
7. Requerimientos de ambiente: herramientas que pueden ser implementadas en las pruebas, tanto de software como hardware y del lugar en el que se harán las actividades.
8. Responsabilidades
9. Personal y entrenamiento
10. Planificación
11. Riesgos y contingencias
12. Aprobación: Se especifica quienes son los responsables para la aprobación y revisión del plan.

4.4 Reportes de verificación y validación de software (RVVS)

Se detallan los resultados que se obtuvieron de las pruebas realizadas de PVVS

Siguiendo el estándar 929, indica que los reportes de pruebas deberán tener la siguiente estructura:

1. Identificador del reporte de la prueba.

2. Resumen: Es el resumen de la evaluación de los ítems de la prueba. Incluye los ítems que fueron probados, su versión o nivel de revisión en el que se encuentra y el ambiente en el que las pruebas fueron hechas.
3. Variaciones.
4. Evaluación exhaustiva.
5. Resumen de los resultados.
6. Evaluación.
7. Resumen de las actividades.
8. Aprobación: se especifica a los responsables de la aprobación del reporte.

4.5 Documentación de usuario

Guías para los usuarios. No están enfocada solo al usuario final, sino que incluye todos los que estén relacionados al sistema, como vendedores o los encargados del mantenimiento. Estos documentos pueden abarcar temas como la instalación, gestión del software y su mantenimiento.

Deben estar completos y no omitir ningún tipo de información que pueda resultar útil para los usuarios. Entre los elementos de esta documentación está las especificaciones de entradas y salidas, instrucciones para entrenamiento con el uso del sistema, descripciones de las limitaciones del sistema, así como los mensajes de error que podrían aparecer en algún momento y las opciones que se tienen para solucionar estos problemas. Dependiendo del tipo de usuario al que vaya dirigido dependerán los temas que incluya.

Verificación y validación

Para el proceso de verificación para este tipo de documentación se deberá realizar una revisión sobre la información que contiene cada documento que caiga en esta categoría. Se realizará una auditoría final sobre los documentos en los que sean necesarios para que puedan pasar al proceso de validación.

El proceso de validación se llevará a cabo en forma de recorrido, asegurando que el usuario que utilizará esta documentación pueda entender sin necesidad de consultar al equipo de soporte.

4.6 Plan de gestión de la configuración del software (PGCS)

Establece las actividades que se deben realizar para la gestión de la configuración. El plan debe incluir temas como:

1. Tareas
2. Métodos: se debe especificar cada método que se implementará para la identificación de los elementos de la configuración de software, para el control e implementación de cambios, las auditorías, los reportes, revisiones y para la elección del personal de mantenimiento.
3. Responsables
4. Calendarios
5. Herramientas y recursos

Verificación y validación

Se deberá realizar una auditoría en ambos casos para asegurarse que el plan cubre todo el cronograma de actividades necesario, asegurarse que los recursos estén dentro del tiempo y el presupuesto establecidos. Por otra parte, se debe asegurar que se cuente con los responsables necesarios para lograr una ejecución eficiente del plan.

La auditoría final debe asegurarse que el plan de la configuración del software sea desarrollado de tal manera que guie y controle múltiples actividades en paralelo, asegurando una buena comunicación en el desarrollo para que de esta forma se realice una buena documentación.

Otra documentación

4.7 Plan de desarrollo de software (PDS)

Describe las actividades técnicas y de gestión que se deben realizar durante el desarrollo de software, además de incluir métodos, herramientas y técnicas para facilitar estas tareas. Se asignan responsabilidades y se crean calendarios para la actividades-

Verificación y validación

Se realizará una verificación que pretende asegurar que todos los requisitos están siendo cubiertos por el plan de desarrollo de software.

Con la validación se deberá lograr la conformidad por parte del cliente de todo lo propuesto en el plan de desarrollo de software, desde el ciclo de vida hasta los productos entregables y las revisiones necesarias para asegurar la correctitud de estos.

4.8 Manual de estándares y procedimientos (MEP)

Este manual incluye toda la información acerca de los estándares y procedimientos que el proyecto considera y debe de cumplir.

Verificación y validación

Se realizará una revisión inicial en la que se revisarán que los estándares y procedimientos satisfagan los requerimientos de desarrollo para asegurar la calidad en cada fase del ciclo de vida del proyecto.

Para la revisión final se revisará que el proyecto haya sido desarrollado conforme al manual de estándares y procedimientos para satisfacer lo que el cliente pide.

4.9 Manual de mantenimiento de software (MMS)

Documentación que especifica las instrucciones del sistema para que se le pueda proveer de mantenimiento. Como mínimo debe incluir:

- Especificaciones del software
- Especificaciones de hardware
- Procedimientos de correctivos
- Instalaciones
- Pruebas de todos los cambios

Verificación

Se realizará una auditoría para asegurar que esté documentado lo necesario sobre el manual de mantenimiento de software para lograr el entendimiento del encargado de realizar el mantenimiento.

5. Estándares, prácticas, convenciones y métricas

5.1 Propósito.

Apartado dedicado a la identificación de los estándares, prácticas, convenciones y métricas que serán implementadas durante el proyecto. Además, se detallará las acciones para asegurar su cumplimiento.

También, es necesario especificar en que fases del ciclo de vida del software se aplicarán.

5.2 Contenido

1. Estándares de documentación
2. Estándares de estructura lógica
3. Estándares de codificación
4. Estándares de comentarios
5. Estándares y pruebas de practicas
6. Métricas de productos y procesos del aseguramiento de la calidad del software seleccionadas

5.2.1 Fase de requerimientos

Fase dedicada al análisis y recopilación de las necesidades del cliente y de los usuarios finales, con el fin de satisfacer sus expectativas del sistema. Al concluir con esta fase se deben los siguientes productos:

- Gestión de la Configuración de Software
- Documento de Especificación de Requisitos
- Plan de Pruebas del Sistema
- Manual de Usuario

Para estos tres documentos se implementarán los estándares:

- IEEE Std 830-1998 para el Documento de Especificación de Requisitos
- IEEE Std. 829-1998 para el Plan de Pruebas del Sistema
- IEEE Std 828-1990 para la gestión de la configuración
- IEEE Std 1063-1987 para el Manual de Usuario.

5.2.2 Fase de análisis y diseño

Fase enfocada en comprensión de los requisitos establecidos en el punto anterior, con el fin de poder crear un modelo que pueda cumplir con lo que el usuario quiere.

Se debe crear una idea completa del sistema, incluyendo comportamiento, datos, de las interfaces y sus procedimientos. Al termino de esta fase, se deben tener los siguientes documentos:

- Plan de Pruebas de Integración.
- Documento de Análisis y Diseño.

Para estos documentos, se hará uso del estándar:

- IEEE Std. 1016-2009

5.2.3 Fase de Construcción

Fase en la que se crean los componentes, siguiendo lo establecido en la fase de Análisis y Diseño. A cada uno de estos componentes se les deberá realizar una prueba unitaria.

Para estas pruebas unitarias, se implementará el siguiente estándar:

- IEEE Std. 1008

5.2.4 Fase de Integración y Pruebas

Fase en la que cada componente se une para formar el sistema completo. Ahora que cada componente está probado individualmente es necesario hacer pruebas a todo el software, con el objetivo de asegurarse que se están cumpliendo con todos lo requerimiento establecidos y que el software es de calidad.

Al terminar esta fase, además de contar con todo el sistema probado, se deben tener completados los siguientes documentos:

a. Manual de Operación
b. Manual de Usuario
c. Manual de Mantenimiento

Para estos documentos, además del estándar anteriormente mencionado para el Manual de Usuario, se utilizará:

- ISO/IEC 14764-2006 para el Manual de Mantenimiento.

5.3 Documentación

Los estándares implementados se mencionan brevemente en la sección 4. Documentación, además de los que se han especificado para los productos durante las fases del ciclo de vida.

Para tener más detalles de los estándares acudir a la sección 2. Documentos referenciados.

5.4 Métricas

Para la recolección de datos de métricas de sobre la calidad de software se implementará el estándar ISO/IEC 15939-2007.

6. Revisiones y auditorías

6.1 Propósito

En el apartado Revisiones y Auditorias se define la forma de realizar revisiones y/o auditorias para los documentos en los que es requerido estas evaluaciones.

Este apartado supone que los procedimientos a realizar para las revisiones y auditorías están conforme al estándar IEEE Std.1028-1998.

De igual manera se indican las acciones necesarias, así como las adicionales para verificar que todo está de acuerdo con lo establecido.

6.2. Requisitos mínimos

Para cubrir de forma aceptable las revisiones y auditorias se debe contar como mínimo con las siguientes revisiones.

6.2.1 Revisión de la especificación de software

Se llevará acabo una revisión sobre la especificación para garantizar que los requisitos establecidos cubren exhaustivamente con los requisitos necesarios para la creación del producto

6.2.2 Revisión del plan de pruebas

Se hará una revisión sobre la estimación, el alcance y la estrategia del plan de pruebas para que el equipo encargado de realizar esta revisión evalúe la consistencia del plan.

Se debe priorizar a que el contenido de este debe cumplir con el objetivo establecido en el documento.

6.2.3 Revisión de diseño detallado

A partir de una revisión se realizará una Revisión Crítica de Diseño (CDR, por sus siglas en ingles) que por su función se determinará si el software procederá a fabricación, demostración y pruebas, así como responde a si los requerimientos de software cumplen los requisitos de costos, tiempo y riesgos.

6.2.4 Revisión del plan de verificación y validación

El propósito de realizar una revisión al plan de verificación y validación es para evaluar el adecuamiento y completitud con respecto a lo definido en los planes de verificación y validación.

Tomando en cuenta esto se deberá realizar una revisión sobre este plan que abarque los puntos mencionados anteriormente.

6.2.5 Auditoria funcional y física

El equipo encargado de SQA se hará cargo de dos auditorias sobre el software para verificar su consistencia con lo especificado en la documentación.

La auditoría funcional en donde se revisará por medio de un recorrido que el software cumpla con establecido en la especificación de requisitos.

La auditoría física por parte del equipo se encargará de revisar la consistencia entre el software y su documentación, así como la preparación para su lanzamiento.

6.2.6 Revisión del plan de administración de la configuración de software

Es un proceso que gestiona, organiza y controla los cambios en los productos del software y otras entidades durante todo el ciclo de vida del software.

Durante el proceso se señalan errores para minimizar la ocurrencia de estos e incrementar la productividad.

Una auditoria verifica que el producto de software satisface los lineamientos necesarios y se entregue lo que es construido en tiempo y forma.

6.2.7 Revisión administrativa

El equipo encargado del SQA realizará revisiones periódicas para asegurar la ejecución de las acciones identificadas en el plan de aseguramiento de la calidad.

Pueden presentarse señalamientos sobre el plan de SQA que deberán ser corregidos en un momento posterior a la revisión.

6.2.8 Revisión Post-Implementación

Después de terminar el proyecto, se realiza está revisión la cual genera las siguientes conclusiones: si los objetivos fueron cumplidos, cómo la eficiencia de realizar el proyecto fue puesta en marcha, las lecciones aprendidas para el futuro y asegurar que la organización haya tenido el mayor beneficio posible del proyecto.

7. Pruebas

En este apartado se describirá la estrategia de pruebas con el que se contará durante todo el ciclo de vida del proyecto RiskMap, además de los criterios para las pruebas. El propósito de las pruebas es el de garantizar que los productos de software generados sean de calidad, evidenciando con las pruebas los defectos del software y validando que el producto esté acorde a lo que se especificó.

Es por ello por lo que, durante la fase de requerimientos, se deberá generar un plan de pruebas cuyo contenido debe de incluir una visión general de las metodologías, cronogramas y recursos para probar el software. En cuanto a la documentación de las pruebas de software, se tomará en cuenta el estándar *IEEE 829-1998 – Standard for Software Test Documentation* que describe la información básica necesaria y resultados de las pruebas de software.

7.1 Proceso de realización de las pruebas:

1. Configurar pruebas

   Como primer paso será necesario planear y preparar todo lo necesario para las pruebas:

   - Plan de pruebas
   - Casos de prueba
   - Procedimientos para realizar las pruebas
   - Herramientas para las pruebas
   - Ambiente de pruebas

2. Realizar pruebas

   Se llevan a cabo las pruebas que fueron planeadas y se obtienen los resultados de las pruebas.

3. Evaluar resultados

   Se verifica si los resultados de los casos de prueba coinciden con los resultados esperados.

4. Reportar defectos

   Se documentan los defectos que fueron encontrados en un Reporte de defectos.

5. Realizar correcciones

   Con base en el Reporte de defectos, se realizan las correcciones pertinentes y se realizan nuevamente las pruebas para asegurar que no se haya introducido otro defecto como efecto secundario.

Las actividades de aseguramiento de calidad en este proceso incluyen la revisión de los siguientes artefactos:

- Plan de pruebas: en este se indican los componentes que serán probados, las características consideradas en la prueba, las tareas a realizar y los riesgos asociados a la prueba.
- Casos de prueba: relacionados directamente con los casos de uso, deben indicar condiciones de ejecución, datos de entrada y datos de salida.
- Procedimiento de pruebas: puede referirse a un caso de prueba en particular o bien aplicarse a varios, define como ejecutar la prueba y en qué condiciones.
- Reporte de incidentes: informa los resultados de la realización de pruebas y debe contener:
  - Entradas
  - Resultados esperados
  - Resultados presentados
  - Anomalías
  - Fecha y hora de ejecución
  - Procedimiento aplicado

- Condiciones de ejecución
- Número de veces que se realizó
- Observaciones

Las revisiones realizadas y las observaciones resultantes deberán de ser documentadas y comunicadas al LPR para su correspondiente atención.

El RUSQ deberá asegurarse de que las observaciones realizadas sean atendidas.

7.2 Pruebas de unidad

Los componentes deberán ser probados por separados para verificar que estos se comporten según fueron diseñados. Para la implementación de las pruebas unitarias se deberá seguir el estándar *IEEE 1008 – Standard for Software Unit Testing*.

Responsables de las pruebas

Los encargados de realizar estas pruebas serán los propios programadores que desarrollaron el componente.

Momentos de aplicación

- Antes de codificar
- Al finalizar la codificación
- Al incorporar cambios en la codificación

Técnicas de prueba

- Caja blanca
- Caja negra
- Regresión

Actividades

- Definir las pruebas unitarias a realizar
- Implementar el código y realizar las pruebas unitarias. Se ejecutan todos los casos de prueba asociados a cada verificación establecida en el plan de pruebas, registrando su resultado.
- Corregir los defectos encontrados
- Repetir las pruebas que detectaron los defectos hasta que se hayan realizado todas las verificaciones establecidas y no se encuentre ningún defecto.

7.3 Pruebas de integración

Una vez probados los componentes de manera individual, probamos la interacción entre ellos para garantizar que funcionan de acuerdo con lo especificado.

Se deberá de elaborar un Plan de Pruebas de Integración durante la fase de Análisis y Diseño. Este plan servirá como guía para su posterior implementación contemplada en la fase de Integración y Pruebas, documentando los resultados en un Reporte de Pruebas de Integración.

El tipo de pruebas de integración a realizarse es la incremental, por lo tanto, conforme los componentes son sometidos a pruebas unitarias deberán de ser probados con el conjunto de componentes a los que ya se la aplicaron las pruebas de integración.

Responsable de las pruebas

Las pruebas son planificadas por los líderes de proyecto en conjunto con el IPR.

Momentos de aplicación

- Al concluir varios componentes o unidades
- Al incorporar cambios a componentes o unidades

Técnicas de prueba

- Caja negra
- Alfa
- Regresión

Actividades

- Planear el enfoque de la prueba
- Programar la entrega de los componentes y el tiempo de ejecución de las pruebas
- Asegurar que los entornos de integración estén configurados y disponibles
- Ejecutar las pruebas definidas
- Registrar y trabajar con los desarrolladores para resolver los defectos identificados

7.4 Pruebas de sistema

Se tiene como objetivo probar los flujos de trabajo que involucran al sistema completo. Son pruebas de integración del sistema de información completo, y permiten probar el sistema en su conjunto y con otros sistemas con los que se relaciona para verificar que las especificaciones funcionales y técnicas se cumplen.

Responsable de las pruebas

La prueba del sistema la realiza el IPR en el producto de software terminado antes de su liberación.

Momentos de aplicación

- Al finalizar el desarrollo
- Al incorporar cambios

Técnicas de prueba

- Alfa
- Rendimiento
- Estrés
- Regresión

Actividades

- Ejecución de las pruebas de sistema definidas
- Registrar y trabajar con los desarrolladores para la resolución de los defectos identificados
- Seguimiento de métricas de prueba
- Resumen de la evaluación de la prueba

8. Reporte de problemas y acciones correctivas

Sección dedicada a la identificación de técnicas (y las responsabilidades de la organización para su ejecución) desarrolladas para la creación de los reportes, el seguimiento y la resolución de problemas.

Es importante mencionar que el alcance de esta actividad no es solamente para la fase de desarrollo de software, sino que también se debe contemplar el proceso de mantenimiento.

8.1 Responsabilidades

Para este plan, el equipo encargado de que se cumplan las acciones correctivas (siguiendo las normas y requerimientos del proyecto), además de darle un adecuado seguimiento hasta que se llegue a una resolución del problema, será el equipo de aseguramiento de calidad, los cuales deben contar con las siguientes características:

- Conocimiento del proceso o producto
- Autoridad para resolver el problema e implementar las acciones correctivas
- Habilidades en las diciplinas técnicas
- Un líder del equipo designado

Cualquier problema que sea encontrado durante el proyecto durante las revisiones de la documentación, en la fase de desarrollo, instalaciones, pruebas y/o mantenimiento debe ser reportado inmediatamente a este equipo.

8.2 Contenido

Para los reportes de problemas o las acciones correctivas, se recomienda que como mínimo contengan:

- **Información general**: información básica acerca del reporte o la acción correctiva.
- **Objetos afectados**: objetos que son afectados por el problema o por la acción correctiva.
- **Tareas**: checklist de las tareas relacionadas al reporte o la acción correctiva.
- **Anexos**: archivos, enlaces o mayor información sobre el problema o la acción correctiva.
- **Análisis del impacto**: vista de cómo el problema o la acción correctiva impacta en otros objetos.
- **Relaciones**: asociaciones entre el reporte del problema o la acción correctiva con cualquier otro objeto de negocio.
- **Seguridad**: responsables quienes tendrán acceso.
- **Historial**: registro de la secuencia de acciones que se han realizado.

9. Herramientas, técnicas y metodologías

Para dar soporte a las actividades realizadas durante el desarrollo e implementación del plan de aseguramiento de calidad se aplicaron técnicas y metodologías que

involucran diferentes herramientas de soporte para el control del versionado, la ejecución de las pruebas de software y herramientas para la administración e implementación del proyecto que se detallan a continuación.

9.1 Herramientas

Para la elección de las herramientas de soporte se consideraron diferentes opciones, analizando las ventajas y desventajas, y seleccionando aquellas que se adecuen más al proyecto. En el caso de los programas a ser usados, se priorizó el uso de software libre dado el presupuesto y las restricciones para el proyecto.

Herramientas para el Control de versiones

Git

Para el control de versiones se decidió hacer uso de un sistema de control de versiones distribuido (SVN) como lo es Git. Este incluye un repositorio central y una serie de repositorios locales en donde cada usuario puede tener una copia completa del proyecto, haciendo que el acceso a la historia de cada uno sea extremadamente rápido. De esta manera, se puede trabajar de forma remota fácilmente sin conexión a internet. La velocidad es otro de los puntos fuertes de Git frente a otros sistemas de control de versiones, ya que necesita menos capacidad de procesamiento y gestión al poder realizar las operaciones en local.

Cabe mencionar que Git es compatible con casi todos los entornos de desarrollo y las herramientas de líneas de comandos de Git se ejecutan en todos los sistemas operativos principales.

Esta herramienta será fundamental para el trabajo en equipo, ya que permite tener un desarrollo en paralelo para un proyecto con acceso compartido sin estar físicamente cerca, así como identificar que usuario y cuando ha realizado cada modificación. Además, podremos comparar las diferentes versiones y restaurarlas en caso de que algo salga mal.

GitHub

GitHub es un servicio de alojamiento de repositorios de software que usa Git y que ha sido pensado para compartir código de una manera fácil y ágil. Esta herramienta nos permite alojar proyectos de código abierto de forma gratuita ofreciéndonos un

seguimiento de errores, búsqueda rápida y una amplia comunidad de desarrolladores alrededor del mundo.

Esta plataforma cuenta también con funciones de organización y gestión de proyectos. Se pueden asignar tareas a individuos o grupos, establecer permisos y roles para los colaboradores y usar la moderación de comentarios para mantener a todos en la tarea.

Herramientas para las Pruebas de Software

JUnit

¿Cuándo utilizar JUnit? Se hará uso de Junit cuando se tenga que realizar pruebas con una única clase.

Permite evaluar el resultado de la ejecución de un método. Con esta herramienta es posible comparar el resultado esperado, con el que realmente se obtuvo después de ejecutar el método.

Mockito

¿Cuándo utilizar Mockito? Cuando las pruebas a una clase utilicen otras clases.

Con Mockito se tiene la posibilidad de simular la respuesta de otro método necesario para ejecutar el método que necesitamos probar. Esto permite un enfoque único en el método que deseamos probar.

Bugzilla

Herramienta que permitirá que los desarrolladores y evaluadores puedan dar seguimiento de los defectos pendientes. Bugzilla permite organizar en múltiples formas los defectos de software, permitiendo el seguimiento de múltiples productos con diferentes versiones, a su vez compuestos de múltiples componentes. Permite además categorizar los defectos de software de acuerdo con su prioridad y gravedad, así como asignarles versiones para su solución.

También permiten anexar comentarios, propuestas de solución, designar a responsables a los que asignar la resolución y el tipo de solución que se aplicó al defecto, todo ello llevando un seguimiento de fechas en las cuáles sucede cada

evento y, si se configura adecuadamente, enviando mensajes de correo a los interesados en el error.

Herramientas de soporte para las revisiones

Las revisiones podrán ser guiadas a través de listas de verificación que deben contener aspectos o características que deben estar presentes en los diferentes artefactos generados durante la realización de un proyecto y deberán de especificar lo siguiente:

- Descripción breve
- Proyecto e identificación del documento
- Fecha de revisión
- Encargado de aplicar la revisión
- Característica verificada y la calificación correspondiente

  Las calificaciones que se pueden asignar son las siguientes:

Completa y correcta (CC)	La característica que se revisa está presente y fue elaborada de forma adecuada
Completa, pero con errores (CE)	La característica que se revisa está presente, pero se necesita corregir algunos aspectos
Incompleta pero correcta (IC)	La característica que se revisa no está completa pero la parte elaborada es correcta
Incompleta y con errores	La característica que se revisa no está completa y la parte elaborada es incorrecta
No considerada	La característica que se revisa no está presente en el artefacto revisado

- Observaciones

Herramientas para la Administración del proyecto

Trello

Utilizamos Trello como software de administración de proyectos ya que esta nos permite organizar, coordinar y gestionar las tareas en tiempo real y con velocidad. Mediante este software asignaremos las actividades a cada uno de los miembros del equipo de trabajo basado en la metodología Kanban.

Consiste en un sistema de tableros, listas y tarjetas que corresponden a las actividades que deben realizarse y en las que es posible añadir imágenes, plazos de entrega, lista de tareas, etiquetas separadas por color y comentarios asociados a cada tarea. Las tarjetas pueden ser editadas y reordenadas, arrastrándolas y soltándolas entre las listas para comunicar el avance de cada una de ellas. De esta manera, una persona que ingrese a Trello estará al tanto del desarrollo de las tareas de un proyecto.

Herramientas para la Implementación de la aplicación

Android Studio

Dado que el software que se desea asegurar con este plan de SQA corresponde a una aplicación móvil para dispositivos Android, usaremos el IDE oficial para desarrollar aplicaciones para este sistema operativo, Android Studio. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece incluso más funciones que aumentan tu productividad cuando desarrollas apps para Android, como las siguientes:

- Sistema de compilación flexible basado en Gradle
- Emulador rápido y cargado de funciones
- Entorno unificado donde puedes desarrollar para todos los dispositivos Android
- Integración con GitHub y plantillas de código para ayudarte a compilar funciones de apps comunes y también incorporar códigos de muestra
- Variedad de marcos de trabajo y herramientas de prueba
- Herramientas de Lint para identificar problemas de rendimiento, usabilidad y compatibilidad de versiones, entre otros

9.2 Técnicas

Las técnicas que se implementarán con el fin de evaluar y mejorar la calidad de software son:

- Pruebas. Para mayor detalle de los momentos en que estas distintas técnicas se aplican y para qué tipo de pruebas se implementan, además de los estándares implementados ver la sección *7. Pruebas.* En resumen, se hará uso de cada una de estas:

    - Alfa: pruebas ejecutadas en las oficinas del desarrollador del producto por un grupo de personas que representan al cliente final, con el fin de registrar errores y problemas de uso.
    - Rendimiento: técnica para medir la velocidad de procesamiento y el tiempo de respuesta del sistema.
    - Estrés: técnica en la cual se seleccionan actividades a probar en un sitio para ejecutarlas en un tiempo determinado desde una ubicación remota.
    - Regresión: pruebas que se realizan cuando un software ha sufrido un cambio, ya sea que se agregó algún modulo nuevo o para solucionar algún problema.
    - Caja negra: técnica que utiliza el análisis de la especificación, funcional y no funcional, sin tener en cuenta la estructura interna del programa.
    - Caja blanca: técnica de en la que se prueba la estructura interna, el diseño y la codificación del software.
- Revisiones: técnica para la detección de errores en el producto de software, puede aplicarse desde etapas tempranas del desarrollo.
- Auditorías: actividad de revisión para verificar el cumplimiento de un sistema establecido y su efectividad o área y mejora o de acción correctiva.
- Uso de métricas según estándar ISO/IEC 15939-2007

9.3 Metodologías

Las técnicas en conjunto con las herramientas mencionadas anteriormente conforman las metodologías a aplicarse para llevar a cabo las diferentes actividades que dan soporte al aseguramiento de la calidad del software.

Con el objetivo de asegurar la calidad que los productos de software generados y controlados con Git y GitHub y la satisfacción de los requerimientos establecidos, se deberá implementar revisiones de los documentos mencionados en el apartado *6. Revisiones* del presente plan por cada versión controlada, así como las auditorias correspondientes para detectar los errores y así puedan presentarse las propuestas de mejora o acciones correctivas que eviten la propagación en futuras versiones.

Además, será muy importante hacer uso de las métricas y pruebas de software, las cuáles deben de estar conforme a lo establecido en el apartado *7. Pruebas* en las que se menciona la aplicación de pruebas a los módulos de manera individual al momento

de ser generados o modificados y la integración incremental de estos con apoyo de las herramientas JUnit y Mockito para la ejecución y Bugzilla para el seguimiento de los defectos encontrados.

Es importante que tanto para las revisiones de los documentos como las revisiones derivadas de las pruebas se considere el uso de las listas de verificación detalladas en las *Herramientas de soporte para las revisiones* y recopilación de evidencia que soporten las actividades realizadas.

10. Control de código

El código fuente, así como los módulos serán desarrollados bajo el uso de un controlador de versiones. Este servirá para el control general de código tanto el etiquetado que llevará los siguientes datos: **nombreDelModulo_equipoAutorizado_apellidoDelEncargado,** representado en el trabajo con ramas. Este etiquetado nos da la noción del pedazo del código, equipo del encargado y el encargado para que durante el tiempo del ciclo de ese código la seguridad sea auditada con el fin de asegurarse que el control se lleve a cabo.

El desarrollo del software indica la creación de código, por lo que, la localización de este será descentralizada, es decir, cada encargado tendrá disponible el código en un repositorio en la nube.

Para llevar a cabo una nueva versión el líder del equipo de desarrollo tendrá que estar de acuerdo que todas las unidades del equipo hayan acabado con lo sus tareas para que posteriormente registre la fecha, el número de versión y describa las adiciones hechas en esta nueva versión, con el fin de proveer copias al cliente en caso de ser necesarios, y también para controlar las versiones del software.

Cada cambio en el código fuente o en un módulo/unidad de software deberá ser documentado mediante los comentarios de las nuevas adiciones, de igual manera deberá ser comunicado a los demás integrantes del equipo para estar atentos a estos cambios.

11. Control de medios

En este punto se abordará todo medio en el que se estarán guardando el proyecto de software.

El control de estos medios se deberá hacer a partir de los puntos que se encuentran posteriormente definidos como "*Acceso no autorizado"* y "*daño o degradación desapercibida".*

Para el caso del sistema RiskMap se hará la documentación del control de medios para llevar a cabo este control sobre el software, así como también, el resguardo de las copias. Los procedimientos por implementar serán descritos a continuación.

El equipo de seguridad será encargado de llevar a cabo el *plan de seguridad del software.* El procedimiento de control empieza por la parte de etiquetado en donde se presentarán los mismos datos que en el control de código, estos son:

- Nombre del medio
- Personal autorizado
- Encargados del control

Lo anterior descrito servirá como el preámbulo del control de medios que, a su vez, nos ayuda para documentación posterior de estos.

11.1 Acceso no autorizado

Se deberá hacer uso de procedimientos para resguardar los medios de accesos no autorizados.

La organización deberá utilizar una contraseña única para cada tipo de usuario dependiendo del rango de acceso ya sea parcial o total. Estas contraseñas deberán ser generadas y entregadas al integrante según el rango que el encargado considere apropiado.

Antes de poder generar estas contraseñas, los encargados de SQA deberán clasificar los medios, asegurar su almacenamiento y definir las restricciones de la siguiente manera:

Software. El software deberá estar almacenado y disponible para su recuperación. El sitio en donde se almacena el software deberá ser un lugar seguro con las características que se mencionan en el punto 11.2. Para el acceso y recuperación de este solo deberá ser permitido para el equipo de desarrollo o quién lo solicite además de controlar el rango de acceso para este.

Copia. La copia del software deberá estar almacenado y disponible para su recuperación. El sitio en donde se almacena la copia del software deberá ser un lugar seguro con las características que se mencionan en el punto 11.2. Para el acceso y recuperación de este solo deberá ser permitido para el equipo de desarrollo o quién lo solicite además de controlar el rango de acceso para este.

11.2 Daño o degradación desapercibida

Para evitar los daños o la degradación de los medios de almacenamiento se hará uso de técnicas adecuadas de la administración de la configuración.

De igual manera, se deberá considerar lugares seguros para su almacenamiento, estos deben contar con las siguientes características:

- Lugar a prueba de fuego
- Con temperaturas internas de hasta 25° Centígrados

El personal de SQA deberá implementar actividades como la revisión periódica de los medios con el fin de documentar el estado de estos.

12. Recolección de registros, mantenimiento y retención

Los siguientes procedimientos están conforme al plan de administración de la configuración del software aprobado por la organización.

Para garantizar la retención e integridad de estos archivos sus accesos serán limitados a el líder del proyecto o responsable del proyecto específico, el cliente para su verificación y validación, por último, los encargados de llevar el control de documentos y la configuración del software.

12.1 Recolección de registros

Los documentos tendrán que cumplir con los siguientes puntos para realizar una recolección para el resguardo de estos:

- Los documentos deberán haber estados validados y verificados con que cumplan los estándares y requerimientos impuestos por el cliente.
- Los documentos resultantes al finalizar el proyecto del software, esto implica la terminación del mantenimiento. Esto con fines de mantener registros históricos.

A continuación, se presenta una lista de los documentos que tendrán que recolectar para su resguardo:

- Manuales
- Diseño especifico
- Plan de pruebas
- Plan de Aseguramiento de la calidad.
- ERS
- Documento de configuración del software

12.2 Mantenimiento de registros.

Los documentos finales o registros estarán almacenados físicamente en un almacén seguro, a prueba de fuego y con seguridad activa en el lugar. Por otro lado, se tendrán los archivos originales de estos documentos almacenados en un repositorio oficial de la organización en la plataforma en la nube GitHub.

Para controlar el acceso sobre estos registros del software, el acceso solo se le dará al líder del desarrollo, el cual será encargado de recuperar el o las partes que el interesado necesita y devolver este registro en caso de finalización de uso.

12.3 Retención de registros.

Los documentos estarán retenidos o almacenados para la recuperación de datos históricos sobre proyectos realizados con el tiempo. Estos documentos son los mismos que se describen para su recolección en el punto 12.1.

En el caso de ser solicitada una destrucción por parte del cliente o la llegada del tiempo límite propuesto como plazo de retención, se llevará a cabo su destrucción del documento y se suspenderá su retención.

13. Entrenamiento

En esta sección se tiene como objetivo el identificar los conocimientos y habilidades necesarias para poder llevar a cabo las actividades de aseguramiento de la calidad de manera efectiva.

Para esto, se deberá definir un Plan de Entrenamiento aplicado a las personas que formarán parte de la unidad de aseguramiento de la calidad. En este se identificarán

las actividades de formación requeridas para lograr una implementación exitosa del Plan de SQA.

A continuación, se presentan habilidades requeridas para realizar las tareas orientadas a las revisiones:

Tarea	Habilidades o conocimientos requeridos
Revisión de código	• Lenguaje de programación aplicado • Framework utilizado en la etapa de implementación
Revisión de la documentación	• Modelo de calidad de referencia • Metodología de desarrollo de software • Guías institucionales de elaboración
Auditoría del proceso de desarrollo	• Proceso de desarrollo de software aplicado • Estándares que fueron tomados como referencia
Auditoría del proceso de Pruebas de Software	• Niveles y técnicas de prueba • Estrategia de pruebas • Herramientas de soporte
Auditoría al proceso de control de cambios	• Administración de la configuración (registro, seguimiento y atención)
Auditoría al proceso de control de riesgos	• Administración de riesgos (identificación, análisis, atención)

En el caso del Plan de Entrenamiento, es fundamental que sea incluya la siguiente información:

- Descripción del personal a ser capacitado
- Objetivos de la capacitación
- Contenido a cubrir en el entrenamiento
- Cantidad estimada de todos los recursos necesarios para llevar a cabo el entrenamiento
- Procedimientos para evaluar la efectividad del entrenamiento y para hacer modificaciones al entrenamiento

- Calendarización de las sesiones de entrenamiento

El personal debe de ser capacitado de acuerdo al objetivo que debe alcanzar, de tal forma que, si su asignación es la generación del Documento de Especificación de Requisitos, conocerá el estándar o estándares de referencia y los lineamientos que debe seguir para generarlo.