**Universidad Autónoma de Yucatán**

**Facultad de Matemáticas**

**Licenciatura en Ingeniería de Software**

**Plan de SQA**

**Maestro:**

Edwin Jesús León Bojórquez

**Alumnos:**

Castrejón Cervantes Rodrigo Alejandro

Cetina Escalante Fernán Enrique

Andrade Ac Edwin Alonso

Pliego Vega Daniel

Gómez Gómez Horacio Jahir

**Fecha de entrega:**

25 de mayo de 2023

**Índice de contenidos**

[**Control de la configuración 1**](#_fcr91gpzjrtz)

[**1. Introducción 3**](#_5dixbockz0q6)

[1.1 Propósito 3](#_j1mbd9ss8jrz)

[1.2 Alcance 3](#_qzlcexxogbbx)

[1.3 Productos y procesos cubiertos en el Plan 4](#_fdd9gl3flpnc)

[**2. Referencias, definiciones y acrónimos 4**](#_rd4yn58n6d6y)

[2.1 Referencias específicas 4](#_c8xgvik1pos9)

[2.2 Definiciones 5](#_ix4p594zrur8)

[2.3 Abreviaciones y acrónimos 6](#_i0z4mffnjhy5)

[**3. Gestión 8**](#_m5cu3hd0jl5i)

[3.1 Organización 8](#_cndyc49tmaiw)

[3.2 Responsabilidades Generales 8](#_j4lcrarjkvre)

[3.3 Tareas 9](#_fva6w7x84fbn)

[3.4 Responsabilidades Específicas 11](#_3hrjbgfamzm8)

[**4. Documentación 13**](#_m6jjqt1rlk91)

[4.1 Requerimientos de Documentación 13](#_1r0oels531tr)

[4.1.1 Especificación de Requerimientos de Software (SRS) 13](#_a41b02sovgym)

[4.1.2 Descripción del Diseño de Software (SDD) 13](#_krfricer5xb9)

[4.1.3 Documentación del Usuario 14](#_2agkrtpxbmxv)

[4.1.4 Plan de Gestión de la Configuración de Software (SCMP) 14](#_f9zlrb4x5g32)

[4.2 Otros documentos de importancia 14](#_nl176sdso8dl)

[4.2.1 Plan de Desarrollo de Software (SDP) 14](#_1nlrw82aoj37)

[4.2.2 Manual de Estándares y Procedimientos (SPM) 14](#_rb1bonocecrc)

[4.2.3 Plan Gestión del Proyecto de Software (SPMP) 15](#_4s8ggt7bod6h)

[4.2.1 Manual de Mantenimiento de Software (SMM) 15](#_ld7lja2foe5b)

[**5. Estándares, prácticas, convenciones y métricas 15**](#_53gcg693dmv5)

[5.1 Estándares y prácticas de pruebas: 15](#_85hwke5syr41)

[5.2 Métricas: 16](#_v7ip5dnr4mew)

[**6. Revisiones y auditorías 18**](#_o6f2tilzw613)

[6.1 Auditoría Funcional 18](#_xae0h9fjitv5)

[6.2 Otras 18](#_i0gkxqyvq0if)

[**7. Pruebas 18**](#_p9q65w6a0pi)

[**8. Informe de problemas y medidas correctivas 19**](#_shvt78h6xri8)

[**9. Herramientas, técnicas y metodologías 19**](#_uxxdvo92mnn5)

[9.1 Herramientas 19](#_kwkaqfjmdsjc)

[**10. Control de código 20**](#_momlhaawoki7)

[**11. Control de medios 20**](#_4794ouv030iy)

[**12. Control de registros de recolección, mantenimiento y retención 21**](#_nbwopq251wyj)

[**13. Capacitación continua 21**](#_x6g7ex6whjnp)

[**14. Gestión de riesgos 22**](#_tuxl7zq8oy7i)

[**15. Bibliografía 24**](#_a7g460it8lk6)

[**16. Anexos 27**](#_kmewgw46bpbg)

[Anexo A.1 – Plantillas de SCM para SCI’s con base en la información básica propuesta por la guía dentro del estándar IEEE 1042-1987 y de acuerdo con los requerimientos esenciales planteados por el estándar IEEE 828-1983. 27](#_uhwnbaoy9c8)

[Anexo A.2 – Tabla 1 – Ejemplo de relaciones y cronograma de revisiones y auditorías requeridas. 29](#_i1br20y0yyc3)

[Anexo A.3 – Flujos de datos importantes de las fases de pruebas unitarias de software planteado dentro del estándar IEEE 1008 - 1987. 30](#_14cn4iwbii8i)

[Anexo A.4 - Acciones correctivas 31](#_mdnc0gf8hizf)

[Anexo A.5 - Matriz de elementos de información planteado en el estándar IEEE/EIA 12207.1 - 1997 32](#_1ldipc2afna6)

# Control de la configuración

Para la gestión de configuración de este Plan de Aseguramiento de la Calidad de Software, se hace uso de la estructura planteada dentro del Plan de Gestión de la Configuración de Software en su anexo Anexo A.1 – Plantillas de SCM para SCI’s con base en la información básica propuesta por la guía dentro del estándar IEEE 1042-1987 y de acuerdo con los requerimientos esenciales planteados por el estándar IEEE 828-1983.

**1.1 Tabla de Identificación del Elemento de la Configuración de Software**

|  |  |
| --- | --- |
| **Título del producto de software** | Plan de Aseguramiento de la Calidad de Software |
| **Localización de referencia** | <https://github.com/Daniel-Pliego/SQA> |
| **Autor** | Edwin Alonso Andrade Ac |
| **Fecha de creación** | 10 de mayo de 2023 |

**1.2 Tabla de Historial de Versiones**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Versión del SCI** | **Fecha de elaboración** | **Estado de integración del cambio** | **Responsable del cambio** | **Identificador de la versión** |
| 1.1 | 24/05/2023 | EN REVISIÓN | Equipo | SQA\_SQAP\_V1.1 |
| 1.0 | 10/05/2023 | ACEPTADO | Equipo | SQA\_SQAP\_V1.0 |

**1.3 Tabla de Historial de Cambios**

|  |  |  |
| --- | --- | --- |
| **Versión del SCI** | **Fecha de la Solicitud de Cambios de Software** | **Descripción de cambios** |
| 1.1 | 24/05/2023 | Modificación de secciones con información faltante, especificación de estándares y referencia a documentos específicos para más información. |
| 1.0 | 10/05/2023 | Versión inicial del documento del Plan de Aseguramiento de la Calidad de Software: |

# 1. Introducción

El presente documento actúa como implementación del estándar IEEE 730.1-1995 para satisfacer con las necesidades para el aseguramiento de la calidad de software dentro de su uso a lo largo del ciclo de vida del proyecto de software de la aplicación web “Pharma Plus”.

El software de Pharma Plus actúa como una plataforma que permite el acceso sencillo a la comparación de información entre precios de medicamentos, los precios históricos de los mismos y la compra de estos de acuerdo con los servicios de entrega disponibles de la farmacia que los vende; su implementación se realizaría dentro de la web para una mayor disponibilidad dentro de su acceso; la importancia de esta herramienta se encuentra dentro de las consideraciones o impacto social que tendría dentro de aquellos usuarios que tengan de necesidad el consumo de medicamentos de forma frecuente, hablándoles una herramienta que facilite la toma de decisiones de manera informada y adecuada dentro de la compra de medicamentos, siendo de esta forma, un elemento de importancia dentro del manejo económico de los consumidores de medicamentos.

## 1.1 Propósito

El propósito del software de “Phama Plus” es ofrecer una herramienta que permita el acceso a la información de los precios de medicamentos ofrecidos por parte de distintas farmacias, de forma que se realice una comparación entre los precios que estos tengan para recomendar al usuario aquel que sea relativamente más baratos entre los que pueda estar buscando, ofreciendo la posibilidad de hacer necesario una única plataforma en donde se pueda hacer la compra de múltiples medicamentos de múltiples farmacias y, los servicios de entrega que serían ofrecidos de acuerdo a la farmacia a la que se haga la compra de estos.

Así, este documento será utilizado como base para la aplicación de acciones del aseguramiento de la calidad de software, dictando los procesos e instrucciones de trabajo necesarios a realizar dentro de los Elementos de Configuración que serán utilizados dentro del ciclo de vida del desarrollo y mantenimiento del proyecto a realizar, siendo estos los mencionados dentro de la subsección **2.3 Productos y procesos cubiertos en el Plan**.

## 1.2 Alcance

El siguiente documento está basado en el estándar IEEE 730.1-1995, Guía para la Planeación del Aseguramiento de la Calidad de Software, con las subclausulas o secciones descritas con base a la Cláusula 3 dentro de esta guía para satisfacer el estándar IEEE 730-2002, siendo necesario para la aplicación estandarizada de las acciones propuestas de acuerdo con los roles mencionados dentro de este plan.

## 1.3 Productos y procesos cubiertos en el Plan

* Documento de Control de Interfaz
* Plan de Gestión de la Configuración de Software
* Descripción del Diseño de Software
* Plan de Desarrollo de Software
* Manual de Mantenimiento de Software
* Plan de la Gestión de la Configuración de Software
* Manual de Estándares y Procedimientos
* Plan de Aseguramiento de la Calidad de Software
* Especificación de Requerimientos de Software
* Documentación del Usuario.
* Plan de Pruebas.
* Plan de Pruebas Unitarias.

# 2. Referencias, definiciones y acrónimos

## 2.1 Referencias específicas

Este plan será utilizado en conjunto con las siguientes publicaciones:

IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology (ANSI)

IEEE Std 730-2002, IEEE Standard for Software Quality Assurance Plans.

IEEE Std 730.1-1989, IEEE Standard for Software Quality Assurance Plans (ANSI).

IEEE Std 828-1990, IEEE Standard for Software Configuration Management Plans (ANSI).

IEEE Std 829-1983 (Reaff 1991), IEEE Standard for Software Test Documentation (ANSI).

IEEE Std 830-1993, IEEE Recommended Practice for Software Requirements Specifications (ANSI).

IEEE Std 1008-1987 (Reaff 1993), IEEE Standard for Software Unit Testing (ANSI).

IEEE Std 1012-1986 (Reaff 1992), IEEE Standard for Software Verification and Validation Plans (ANSI).

IEEE Std 1012 - 2016, IEEE Standard for System and Software Verification and Validation.

IEEE Std 1016-1987 (Reaff 1993), IEEE Recommended Practice for Software Design Descriptions (ANSI).

IEEE Std 1028-1988, IEEE Standard for Software Reviews and Audits (ANSI).

IEEE Std 1058.1-1987 (Reaff 1993), IEEE Standard for Software Project Management Plans (ANSI).

IEEE Std 1063-2001, IEEE Standard for Software User Documentation.

IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes.

IEEE Std 1219-1993, IEEE Standard for Software Maintenance (ANSI).

ISO 9001:2015, Sistemas de gestión de la calidad - Requisitos.

IEEE/EIA 12207.1 - 1997, Industry Implementation of International Standard ISO/IEC 12207 : 1995 (ISO/IEC 12207) Standard for Information Technology— Software life cycle processes— Life cycle data.

ISO/IEC/IEEE 16326:2019, Systems and software engineering — Life cycle processes — Project management.

MIL-STD-498, Software Development and Documentation.

ISO 19475:2021, Document management — Minimum requirements for the storage of documents.

ISO/IEC/IEEE 12207:2008, System and Software Engineering - Software life cycle processes.

IEEE 29119: Software testing

ISO/IEC 27001: Information security management systems

ISO/IEC 27005 Information technology — Security techniques — Information security risk management

## 2.2 Definiciones

Las definiciones listadas a continuación son utilizadas para establecer un significado más apropiado dentro del contexto de este documento; otras definiciones pueden ser encontradas dentro de los estándares de IEEE 610.12-1990 e IEEE 730-2002 ubicados dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

**Convenciones:** Pautas de aceptación para prescribir una disciplina, ofreciendo un enfoque uniforme para proveer consistencia dentro de un elemento de software como lo serían patrones uniformes o maneras en la que debería ser acomodados datos de cierto tipo.

**Guía:** Documentos en los cuales se ofrecen enfoques alternativos y buenas prácticas sin mencionar una recomendación de manera estricta.

**Práctica:** Enfoque recomendado para realizar una disciplina prescrita de una manera uniforme dentro de un ciclo de vida de un proyecto de software.

**Estándares:** Requerimientos mandatorios utilizados para prescribir un enfoque disciplinado y uniforme al desarrollo, mantenimiento y funcionamiento u operación de software.

**Técnicas:** Técnicas y procedimientos de gerenciales o de gestión para alcanzar un objetivo especificado.

**Herramientas de software:** Programas de computadora utilizados para apoyar dentro del desarrollo, análisis o mantenimiento de un programa o su documentación.

**Metodología:** Serie de técnicas o métodos de manera comprensiva e integrada para la creación de la teoría de sistemas generales sobre la manera en la que debe ser realizado una aproximación basada en ideas,

## 2.3 Abreviaciones y acrónimos

Las siguientes abreviaciones y acrónimos acomodados en forma alfabética son utilizados a lo largo del documento:

**CCB** Junta de Control de la Configuración

**CDR** Revisión de Diseño Crítico

**CI**  Elemento de Configuración

**CM** Gestión de la Configuración

**COTS** Componente Comercial Salido del Estante

**ICD**  Documento de Control de Interfaz

**QA**  Aseguramiento de la Calidad

**SCM**  Gestión de la Configuración de Software

**SCMP** Plan de Gestión de la Configuración de Software

**SCMPR**  Revisión de Plan de Gestión de la Configuración de Software

**SDD**  Descripción del Diseño de Software

**SDP** Plan de Desarrollo de Software

**SMM**  Manual de Mantenimiento de Software

**SPMP**  Plan de la Gestión de la Configuración de Software

**SPM** Manual de Estándares y Procedimientos

**SQA** Aseguramiento de la Calidad de Software

**SQAP**  Plan de Aseguramiento de la Calidad de Software

**SRR**  Revisión de Requerimientos de Software

**SRS**  Especificación de Requerimientos de Software

**SVVP** Plan de Validación y Verificación de Software

**SVVPR**  Reporte del Plan de Validación y Verificación de Software

**SVVR** Reporte de Validación y Verificación de Software

**TQM**  Gestión de la Calidad Completa

**UDR**  Revisión de la Documentación del Usuario

**V&V** Verificación y Validación.

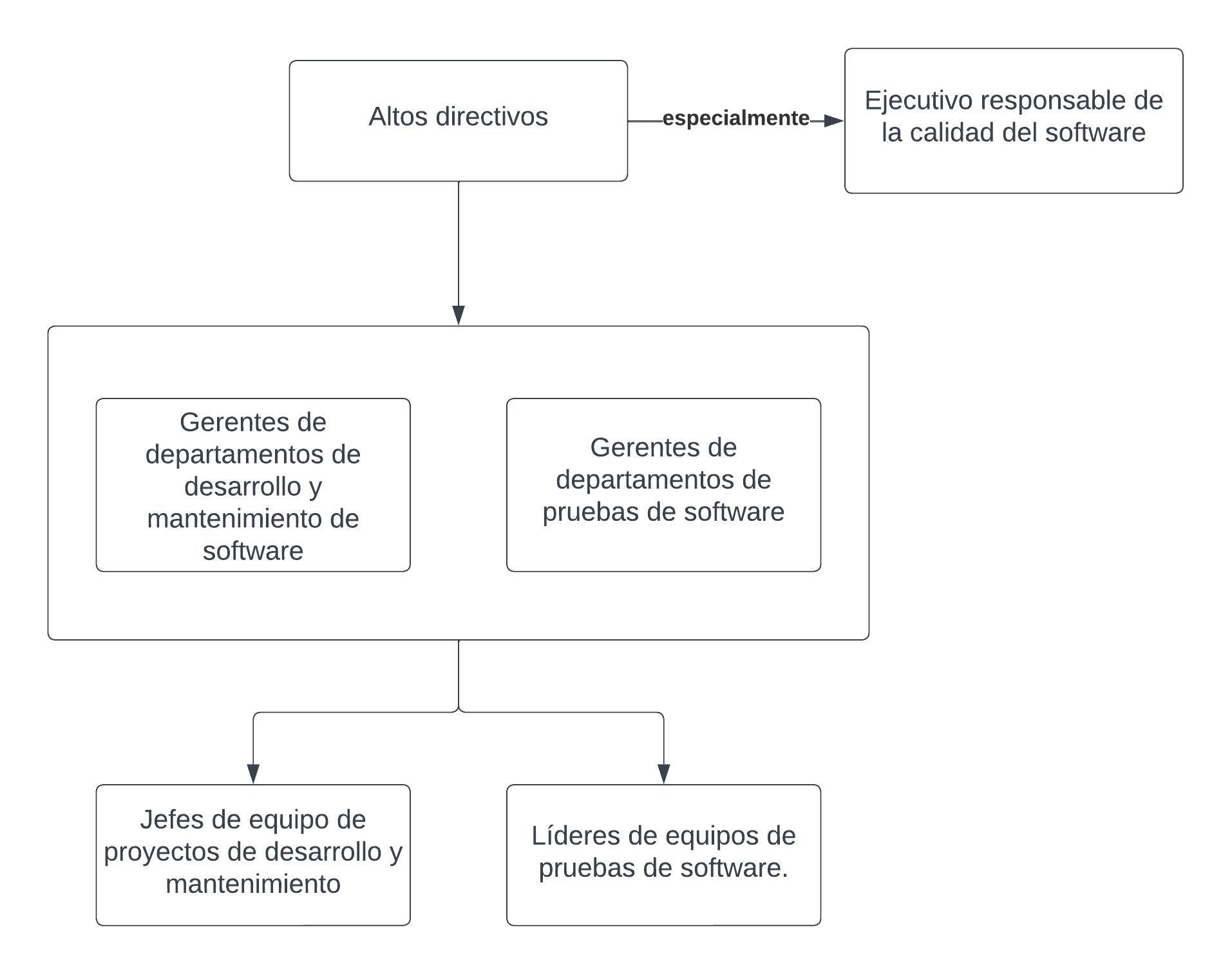
# 3. Gestión

## 3.1 Organización

Se presenta la estructura organizativa que influye en la calidad del software, incluyendo una descripción de cada elemento de la organización junto con sus respectivas responsabilidades.

La estructura organizativa que afecta la calidad del software se compone de los siguientes elementos principales:

*Ilustración 3.1.A – Gráfica de los elementos de la organización.*



## 3.2 Responsabilidades Generales

* Departamento de Desarrollo de Software:
  + Diseñar, desarrollar e implementar el software de acuerdo con los requisitos establecidos.
  + Realizar pruebas unitarias y de sistema.
  + Mantener la documentación técnica actualizada.
  + Realizar revisiones técnicas y de diseño.
* Equipo de Control de Calidad:
  + Definir y mantener los estándares de calidad del software.
  + Realizar revisiones formales del código y las documentaciones asociadas.
  + Ejecutar pruebas de calidad, incluyendo pruebas de funcionalidad, rendimiento y seguridad.
  + Registrar y hacer seguimiento de las incidencias y defectos encontrados.
  + Coordinar con el equipo de desarrollo para la resolución de defectos.
* Equipo de Pruebas:
  + Diseñar y ejecutar casos de prueba funcionales y no funcionales.
  + Realizar pruebas de regresión para garantizar la estabilidad del software.
  + Automatizar pruebas recurrentes para mejorar la eficiencia.
  + Generar informes de pruebas y analizar los resultados.
  + Colaborar con el equipo de desarrollo y control de calidad en la resolución de defectos.
* Equipo de Gestión de Proyectos:
  + Planificar y coordinar las actividades relacionadas con el desarrollo del software.
  + Establecer plazos y asignar recursos para las tareas del proyecto.
  + Realizar un seguimiento regular del progreso del proyecto.
  + Identificar y gestionar los riesgos asociados al desarrollo del software.
  + Coordinar la comunicación entre los diferentes equipos involucrados.

Es importante destacar que el equipo de Control de Calidad y el equipo de Pruebas operan de manera independiente del Departamento de Desarrollo de Software. Ambos equipos reportan directamente a la Gerencia de Proyectos, que tiene la autoridad y la responsabilidad final sobre la calidad del software.

Esta estructura organizativa permite una separación clara de roles y responsabilidades, asegurando que se realicen actividades de SQA de manera independiente y objetiva. Además, se fomenta la colaboración entre los diferentes equipos para garantizar la calidad del software en todas las etapas del desarrollo.

## 3.3 Tareas

El Plan de Aseguramiento de Calidad de Software abarca las siguientes etapas del ciclo de vida del software:

|  |  |  |
| --- | --- | --- |
| **Etapa** | **Tarea** | **Descripción** |
| **Requisitos y Análisis** | Tarea 1 | Definición y documentación de los requisitos del software. |
| Tarea 2 | Análisis de requisitos para evaluar su completitud y coherencia. |
| Tarea 3 | Revisión de los requisitos por parte del equipo de control de calidad. |
| **Diseño** | Tarea 4 | Diseño arquitectónico del software. |
| Tarea 5 | Diseño detallado de componentes y módulos. |
| Tarea 6 | Revisión del diseño por parte del equipo de control de calidad. |
| **Implementación y Pruebas Unitarias** | Tarea 7 | Codificación del software según el diseño establecido. |
| Tarea 8 | Realización de pruebas unitarias para verificar la funcionalidad de los componentes. |
| Tarea 9 | Revisión del código por parte del equipo de control de calidad. |
| **Integración y Pruebas de Integración** | Tarea 10 | Integración de los componentes y módulos desarrollados. |
| Tarea 11 | Ejecución de pruebas de integración para validar la interoperabilidad y el correcto funcionamiento del software. |
| Tarea 12 | Revisión de los resultados de las pruebas de integración por parte del equipo de control de calidad. |
| **Pruebas del Sistema** | Tarea 13 | Diseño y ejecución de casos de prueba funcionales y no funcionales en el entorno del sistema. |
| Tarea 14 | Evaluación de la usabilidad, rendimiento y seguridad del software. |
| Tarea 15 | Revisión de los informes de pruebas y seguimiento de las acciones correctivas necesarias. |
| Tarea 16 | Documentar el manual de operación y el manual del usuario. |
| Tarea 17 | Verificar el manual de operación y el manual del usuario. |
| **Documentación y Entrega** | Tarea 18 | Elaboración de la documentación técnica del software. |
| Tarea 19 | Preparación del paquete de entrega final del software. |
| Tarea 20 | Revisión de la documentación por parte del equipo de control de calidad. |

Las tareas mencionadas anteriormente se llevarán a cabo en la secuencia presentada. Cada tarea incluirá actividades específicas de aseguramiento de calidad del software, como revisiones, pruebas y evaluaciones. Estas actividades serán realizadas por los equipos correspondientes, y sus resultados serán revisados y evaluados en los puntos de control planificados.

## 3.4 Responsabilidades Específicas

Las responsabilidades organizativas específicas para cada tarea son las siguientes:

|  |  |  |
| --- | --- | --- |
| **Etapa** | **Tarea** | **Responsables** |
| **Requisitos y Análisis** | Tarea 1 | Departamento de Desarrollo de Software |
| Tarea 2 | Equipo de Control de Calidad |
| Tarea 3 | Equipo de Control de Calidad |
| **Diseño** | Tarea 4 | Departamento de Desarrollo de Software |
| Tarea 5 | Departamento de Desarrollo de Software |
| Tarea 6 | Equipo de Control de Calidad |
| **Implementación y Pruebas Unitarias** | Tarea 7 | Departamento de Desarrollo de Software |
| Tarea 8 | Equipo de Pruebas |
| Tarea 9 | Equipo de Control de Calidad |
| **Integración y Pruebas de Integración** | Tarea 10 | Departamento de Desarrollo de Software |
| Tarea 11 | Equipo de Pruebas |
| Tarea 12 | Equipo de Control de Calidad |
| **Pruebas del Sistema** | Tarea 13 | Equipo de Pruebas |
| Tarea 14 | Equipo de Pruebas |
| Tarea 15 | Equipo de Control de Calidad |
| Tarea 16 | Departamento de Desarrollo de Software |
| Tarea 17 | Equipo de Control de Calidad |
| **Documentación y Entrega** | Tarea 18 | Departamento de Desarrollo de Software |
| Tarea 19 | Departamento de Desarrollo de Software |
| Tarea 20 | Equipo de Control de Calidad |

Esta asignación de responsabilidades garantiza la participación de los elementos organizativos pertinentes en cada etapa del proceso de desarrollo y aseguramiento de la calidad del software. Cada equipo tiene roles específicos y contribuye con su experiencia y conocimiento para asegurar que se cumplan los estándares de calidad establecidos y se logren los objetivos del proyecto.

* Equipo de desarrollo: Será responsable de llevar a cabo las actividades de revisión de requisitos, inspección de diseño, pruebas de unidad, pruebas de integración y revisión de código. También serán responsables de corregir los defectos identificados durante estas actividades y de colaborar con el equipo de pruebas en la resolución de problemas.
* Equipo de pruebas: Será responsable de planificar y ejecutar las pruebas de aceptación, así como de realizar pruebas de integración adicionales en colaboración con el equipo de desarrollo. También se encargará del seguimiento de defectos y de reportar los resultados de las pruebas a los interesados pertinentes.
* Equipo de gestión de la calidad: Será responsable de supervisar y coordinar todas las actividades de aseguramiento de calidad del software. Esto incluye establecer y mantener los estándares y procesos de calidad, proporcionar orientación y capacitación sobre SQA, y asegurar que se cumplan los requisitos de calidad en todos los proyectos. Además, la posición de gestión responsable de la calidad del software retendrá la autoridad y responsabilidad global para garantizar la calidad del software en toda la organización.

# 4. Documentación

Dentro de este SQAP, se hace la identificación de aquellos documentos que serán preparados durante el desarrollo, la verificación y validación, el uso y el mantenimiento de software, al igual que aquellas revisiones y auditorías específicas identificadas que se encuentren asociadas en forma de criterios requeridos para cada documento mencionado haciendo su mención al apartado correspondiente dentro de la sección de **7. Revisiones y Auditorías.**

## 4.1 Requerimientos de Documentación

La documentación siguiente conformaría aquellos documentos necesarios para cumplir con el mínimo establecido dentro del estándar IEEE 730-1989, estándar que se encuentra ubicado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

### 4.1.1 Especificación de Requerimientos de Software (SRS)

El SRS deberá describir de manera clara y precisa los requerimientos esenciales (funciones, rendimientos, restricciones de diseño y atributos) del software y de las interfaces externas, siendo que cada uno de los requerimientos puedan ser definidos de forma que permitan una verificación objetiva y una validación prescrita por métodos formalizados; este deberá cumplir con el estándar IEEE 830-1993 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SRS\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control)

### 4.1.2 Descripción del Diseño de Software (SDD)

El SDD deberá proporcionar una idea clara sobre la manera en la que el software será estructurado para satisfacer los requerimientos planteados dentro del documento de SRS, describiendo los componentes y subcomponentes del diseño de software, incluyendo tanto bases de datos como interfaces internas, preparándose inicialmente como un SDD preliminar o *Top-Level SDD* y de manera subsecuente ser expandido para producir el SDD detallado; este deberá cumplir con el estándar IEEE 1016-1987 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SDD\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control).

### 4.1.3 Documentación del Usuario

La documentación del usuario deberá especificar y describir los datos requeridos, entradas de control, secuencias de entradas, opciones, limitaciones del programa y otras actividades o elementos necesarios para una ejecución exitosa del software, incluyendo pero no limitándose a la descripción de identificación de mensajes de error para la posterior implementación de acciones correctivas; estos deben cumplir con el estándar IEEE Std 1063-2001 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar la carpeta de Documentación de Usuario dentro del repositorio del proyecto para los diversos productos que caen en esta categoría.

### 4.1.4 Plan de Gestión de la Configuración de Software (SCMP)

El SCMP deberá documentar los métodos que serán utilizados para la identificación de elementos de software, el control e implementación de cambios y el registro y reporte del estado de la implementación de cambios, describiendo las tareas, metodología y herramientas necesarias para asegurar que los procedimientos SCM sean los más adecuados y que estos sean documentados e implementados de manera correcta; este deberá cumplir con los estándares IEEE 828-1990 e IEEE 1042-1987 que pueden ser consultados dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SCM\_V1.1 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control).

## 4.2 Otros documentos de importancia

Otros documentos de importancia tomados en cuenta son los siguientes:

### 4.2.1 Plan de Desarrollo de Software (SDP)

El SDP puede ser utilizado como un documento de planeación de alto nivel que dirija un proyecto, como un elemento de apoyo dentro de una agrupación de planes, siendo que incluya la descripción del desarrollo de software, la organización, responsabilidades e interfaces del desarrollo de software; el proceso para la gestión del desarrollo de software, métodos técnicos, herramientas y técnicas que puedan servir de apoyo dentro del desarrollo; la asignación de responsabilidades de cada actividad de software, entre otros. Este debe cumplir con el estándar IEEE ISO/IEC/IEEE 16326:2019 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SDP\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control).

### 4.2.2 Manual de Estándares y Procedimientos (SPM)

El SPM debe proveer detalles sobre los estándares y procedimientos que deben de seguirse, siendo que como mínimo, debe incluir la información descrita en la sección **6. Estándares, prácticas, convenciones y métricas**; este debe cumplir con el estándar IEEE ISO/IEC/IEEE 16326:2019 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SPM\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control).

### 4.2.3 Plan Gestión del Proyecto de Software (SPMP)

El SPMP debe identificar todas las actividades técnicas y de gerencia asociadas a una instancia del desarrollo de software, incluyendo elementos pero no limitándose a:

* Proceso para la gestión del desarrollo de software.
* Calendarización e interrelaciones entre actividades.
* Procesos de actividades para mejoras.
* Actividades para el despliegue de metas.
* Lista de derivables
* Plan(es) de la gestión de subcontratistas.

Este debe cumplir con el estándar 1058.1-1987 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SPMP\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control).

### 4.2.1 Manual de Mantenimiento de Software (SMM)

El SDP debe contener las instrucciones para el soporte del producto y mantenimiento de software, como lo serían los procedimientos para la corrección de defectos, la instalación de mejoras y la prueba de todos los cambios, siendo que incluya información descrita en la sección **9. Informe de problemas y medidas correctivas**; este debe cumplir con el estándar IEEE Std 1219-1992 que puede ser consultado dentro del repositorio en la carpeta de [estándares usados para SQAP](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares/Usados%20para%20SQAP).

Para una descripción más detallada, revisar el documento de SQA\_SMM\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control).

# 5. Estándares, prácticas, convenciones y métricas

Esta sección establece los estándares, prácticas, convenciones y métricas que guiarán la fase de pruebas del proceso de SQA. Con esto, se pretende garantizar la realización de pruebas exhaustivas y la identificación oportuna de los defectos.

### 5.1 Estándares y prácticas de pruebas:

*Tabla 5.1.A. Listado de estándares que se usarán para la elaboración y verificación de los productos de la etapa de pruebas.*

|  |  |  |
| --- | --- | --- |
| **Producto** | **Tipo** | **Estándar** |
| Plan de Pruebas de Integración | Plantilla | Estándar: MIL-STD-498  Plantilla: DI-IPSC-81438A |
| Reporte de pruebas de integración | Plantilla | Estándar: MIL-STD-498  Plantilla: DI-IPSC-81440A |
| Registro de rastreo | Plantilla | ANSI/IEEE 1058-1998 |
| Manual de operación | Plantilla | Estándar: MIL-STD-498  Plantilla: DI-IPSC-81444A |
| Manual de operación | Verificación | IEEE 1012-2016 |
| Plan de pruebas de sistema | Plantilla | Estándar: MIL-STD-498  Plantilla: DI-IPSC-81438A |
| Reporte de pruebas de sistema | Plantilla | Estándar: MIL-STD-498  Plantilla: DI-IPSC-81440A |
| Manual de usuario | Plantilla | Estándar: MIL-STD-498  Plantilla: ​​DI-IPSC-81443A |
| Manual de usuario | Verificación | IEEE 1012-2016 |
| Revisión técnica y auditoría de software | Plantilla | MIL-STD-1521 |
| Reporte de actividades | Plantilla | ANSI/IEEE 1058-1998 |

### 5.2 Métricas:

Se establecen las siguientes métricas adaptadas de Dustin et al. (1999, lib. Automated Software Testing) con el objetivo de seguir buenas prácticas para la fase de pruebas en la organización.

*Tabla 5.2.A – Métricas de pruebas adaptadas de Dustin et al. (1999, lib. Automated Software Testing).*

|  |  |  |
| --- | --- | --- |
| **Nombre de la Métrica** | **Descripción** | **Categoría** |
| Cobertura de Pruebas | Número total de procedimientos de prueba/número total de requisitos de prueba. La métrica de Cobertura de Pruebas indicará la cobertura planificada de las pruebas. | Cobertura |
| Análisis de Cobertura del Sistema | Mide la cantidad de cobertura a nivel de la interfaz del sistema. | Cobertura |
| Estado de Ejecución del Procedimiento de Prueba | Número de procedimientos de prueba ejecutados/número total de procedimientos de prueba. Esta métrica de Ejecución del Procedimiento de Prueba indicará la cantidad de esfuerzo de prueba pendiente. | Progreso |
| Tasa de Descubrimiento de Errores | Número total de defectos encontrados/número de procedimientos de prueba ejecutados. La métrica de Tasa de Descubrimiento de Errores utiliza el mismo cálculo que la métrica de densidad de defectos. Se utiliza para analizar y respaldar una decisión racional de lanzamiento del producto. | Progreso |
| Antigüedad de Defectos | Fecha en que se abrió el defecto versus fecha en que se solucionó el defecto. La métrica de Antigüedad de Defectos proporciona una indicación del tiempo de respuesta para solucionar el defecto. | Progreso |
| Retesteo de Solución de Defectos | Fecha en que se solucionó y lanzó el defecto en una nueva versión versus fecha en que se volvió a probar el defecto. La métrica de Retesteo de Solución de Defectos proporciona una idea de si el equipo de pruebas está volviendo a probar las soluciones lo suficientemente rápido como para obtener una métrica de progreso precisa. | Progreso |
| Proporción de Calidad Actual | Número de procedimientos de prueba ejecutados con éxito (sin defectos) versus el número total de procedimientos de prueba. La métrica de Proporción de Calidad Actual proporciona indicaciones sobre la cantidad de funcionalidad que se ha demostrado con éxito. | Calidad |
| Calidad de las Soluciones | Número total de defectos reabiertos/número total de defectos solucionados. Esta métrica de Calidad de las Soluciones proporcionará indicaciones sobre problemas de desarrollo. | Calidad |
| Proporción de funcionalidad previamente funcional versus nuevos errores introducidos | La métrica de Calidad de las Soluciones realizará un seguimiento de la frecuencia con la que la funcionalidad previamente funcional se ve afectada negativamente por las correcciones de software. | Calidad |
| Informes de Problemas | Número de Informes de Problemas de Software desglosados por prioridad. La medida de Informes de Problemas Resueltos cuenta la cantidad de problemas de software informados, listados por prioridad. | Calidad |
| Efectividad de las Pruebas | La efectividad de las pruebas debe evaluarse estadísticamente para determinar qué tan bien los datos de prueba han expuesto los defectos presentes en el producto. | Calidad |
| Eficiencia de las Pruebas | Número de pruebas requeridas / número de errores del sistema | Calidad |

# 6. Revisiones y auditorías

De acuerdo con la tabla 1 – “Ejemplo de relaciones y cronograma de revisiones y auditorías requeridas” (anexo A.2) del estándar IEEE 730.11995, la única auditoría o revisión perteneciente a esta fase es la de Auditoría Funcional para la Documentación de pruebas.

## 6.1 Auditoría Funcional

La auditoría funcional compara el software construido (incluyendo sus formas ejecutables y la documentación disponible) con los requisitos de software establecidos en el SRS. Su propósito es asegurar que el código abordó todos y solo los requisitos y capacidades funcionales documentadas establecidas en el SRS.

El equipo de control de calidad será el encargado de realizar la auditoría funcional y deberá de documentar sus resultados en un reporte de auditoría funcional, el cual, según el estándar IEEE 730.1-1995, identifica todas las discrepancias encontradas, así como un plan y un cronograma para su resolución. Las auditorías no se consideran completas hasta que todas las discrepancias hayan sido resueltas.

*Tabla 6.1.A – Ejemplo de reporte de auditoría funcional tomando como base lo establecido en el estándar IEEE 730.1-1995.*

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre: | | | Fecha: |
| **Requerimiento** | **Discrepancia encontrada** | **Plan para su resolución** | **Cronograma para su resolución** |

## 6.2 Otras

Para las UDR, se tomaron en cuenta los siguientes documentos del usuario: Manual de Operación y Manual de Usuario. Con el fin de asegurar que esta documentación se pruebe junto con el software, se seguirán los procedimientos establecidos en los siguientes planes:

* Plan de Pruebas de Integración.
* Plan de Pruebas de Sistema.

Además, se realizarán pruebas de sistema para validar que el producto de software concuerde con la documentación. Se compararán las funcionalidades con la documentación y se verificarán los manuales de operación y de usuario utilizando el método establecido en el estándar IEEE 1012-2016.

# 7. Pruebas

* Pruebas de integración: Se centran en probar la interacción entre diferentes componentes o módulos de un sistema. Su objetivo es garantizar que el sistema funcione como una entidad coherente y que los componentes individuales se integren de manera adecuada. Se aplicarán las pruebas siguiendo el Plan de Pruebas de Integración, el cual se basa en la plantilla DE-IPSC-81438A con uso del estándar IEEE 1008-1987. Adicionalmente, se seguirá el procedimiento planteado en *End-To-End Integration Testing Design* (Tsai et al., 2001).
* Pruebas de sistema: Son un conjunto de pruebas que se realizan para evaluar el sistema completo; evalúan el comportamiento global del sistema en diferentes escenarios, asegurando que todas las funcionalidades operen de manera adecuada. Estas pruebas se llevan a cabo después de las pruebas de integración y tienen como objetivo verificar si el sistema cumple con los requisitos funcionales y no funcionales establecidos. Las pruebas se aplicarán siguiendo el Plan de Pruebas de Sistemas, el cual se basa en la plantilla DE-IPSC-81438A con uso del estándar 1012 - 2017. Adicionalmente, se tomará en cuenta el método planteado en *Software system testing method based on formal model* (Zhang et al., 2017).

Para el flujo de datos se puede considerar la gráfica del anexo A.3 para el planteamiento adecuado de entradas y salidas de datos.

Para un detallado de las acciones anteriores, se puede consultar el documento de SQA\_STP\_V1.0 dentro del repositorio del proyecto en la carpeta de [control de versiones](https://github.com/Daniel-Pliego/SQA/tree/main/Version%20Control) para un análisis descriptivo de las acciones realizadas.

# 8. Informe de problemas y medidas correctivas

Para asegurar que los problemas se encuentren documentados, validados; que su resolución llega a término con base a lo aprobado por el cliente y se brinda la retroalimentación necesaria al equipo de desarrollo, se usará como plantilla la sección 5.17 Acciones correctivas, del Estándar MIL-STD-498. El contenido de dicha sección se encuentra disponible en el Anexo A.4.

# 9. Herramientas, técnicas y metodologías

### 9.1 Herramientas

*Tabla 9.1.A – Herramientas que se utilizarán en la etapa de pruebas.*

|  |  |  |
| --- | --- | --- |
| **Categoría** | **Herramienta** | **Descripción** |
| Administración de pruebas | SonarQube | Es una herramienta de análisis estático de código fuente que se utiliza para evaluar la calidad y seguridad del software.  SonarQube se usará para identificar y corregir problemas de código, como errores y vulnerabilidades lógicas. |
| Jira | Es una herramienta de gestión de proyectos y seguimiento de problemas que se utiliza para organizar, rastrear y gestionar el trabajo en equipo.  Jira se usará para gestionar y realizar un seguimiento de los casos de prueba, registrar y priorizar los errores encontrados durante las pruebas, asignar tareas a los miembros del equipo de pruebas y rastrear el progreso de las pruebas. |
| Automatización de pruebas | Selenium | Es una herramienta de automatización de pruebas que se utiliza para simular interacciones de usuarios y validar el funcionamiento de aplicaciones web.  Selenium se utilizará para escribir y ejecutar casos de pruebas automatizadas, interactuar con elementos de la interfaz de usuario y verificar resultados esperados. |
| Generación de datos de pruebas | Faker | Es una biblioteca de generación de datos falsos.  Faker se usará para generar datos de prueba aleatorios y realistas, como nombres, direcciones, números de teléfono, correos electrónicos, entre otros. |
| Análisis de datos | Microsoft Excel | Es una herramienta de software que permite crear y manipular hojas de cálculo.  Excel se utilizará como una herramienta para recopilar, organizar y procesar datos relevantes para evaluar los resultados de las pruebas. |

# 10. Control de código

Los métodos y recursos utilizados para mantener, almacenar, proteger y documentar versiones controladas del software, durante todas las fases del ciclo de vida del software son especificados en el documento: SQA\_CC\_ V1.0 dentro del repositorio de la organización y que fue realizado con base al estándar ISO/IEC 27001.

# 11. Control de medios

Para el control de medios dentro de la información para aquellos productos de computadora y documentación cuenten con la protección al acceso no autorizado, daños inadvertidos o la degradación de estos a lo largo de las fases del ciclo de vida del software, tomando en cuenta los tres puntos de guía principales planteados por el estándar IEEE/EIA 12207.1 - 1997 sobre las pautas que debe cumplir la información obtenida durante el ciclo de vida de un software:

* Seguro y privado: La información debe ser segura y privada, existiendo un control al acceso de información.
* Protegida: La información debe ser protegida para la persistencia de información en forma de respaldos para evitar la pérdida o el daño hacia la misma.
* Exacta: La información debe ser exacta al cumplir con ser correcta y adecuada.

Así, para asegurar la protección y acceso de estos, se hace uso de la matriz de elementos de información planteados por el mismo estándar para la identificación de los elementos de información que el estándar IEEE/EIA 12207 requiere y/o recomienda producir como parte de alguna actividad, tarea o dentro del proceso de ciclo de vida como tal. El contenido de dicha matriz se encuentra disponible en el anexo A.5

# 12. Control de registros de recolección, mantenimiento y retención

Para asegurar el control de la información documentada y para garantizar la calidad de los productos finales se puede consultar el documento SQA\_DC\_V1.0 elaborado de acuerdo con la norma ISO 9001. La norma exige procedimientos de acceso a los documentos y control de cambios para proteger la integridad de los datos que se utilizan para crear productos que cumplan las expectativas de los clientes.

De esta forma podemos garantizar que los documentos elaborados durante la fase de pruebas como lo serían: manual de usuario, plan de pruebas, manual de operación, pruebas de integración, entre otros; se encuentran almacenados de forma segura.

# 13. Capacitación continua

Las necesidades descritas en la parte inferior son cubiertas por el plan de capacitación que fue realizado con base al estándar ISO 9001:2015. El cual se encuentra especificado en el documento: SQA\_CoCap\_V1.0 dentro del repositorio de la organización. En este documento se mencionan y describen las actividades, responsables, modo de realización y momentos donde se aplica cada etapa del plan

* Tecnologías y herramientas de prueba: Proporcionar capacitación regular sobre las últimas tecnologías y herramientas utilizadas en el proceso de prueba de software. Esto incluye herramientas de automatización de pruebas, herramientas de gestión de defectos y herramientas de monitoreo de rendimiento, acorde a lo especificado en el estándar IEEE 829.
* Metodologías de desarrollo y estándares de la industria: Mantener al equipo actualizado sobre las metodologías de desarrollo de software más recientes, como Agile, Scrum, DevOps, etc. Además, asegurar que estén al tanto de los estándares y mejores prácticas de la industria en cuanto a calidad y pruebas de software, acorde a lo establecido en los estándares ISO/IEC IEEE 12207, ISO/IEEC 15288, así como el estándar propio de la metodología ágil empleada, como Scrum, Kanban, Lean, entre otros.
* Tendencias y avances en SQA: Fomentar la exploración de las últimas tendencias y avances en el campo del SQA. Esto puede incluir áreas como pruebas basadas en inteligencia artificial (IA), pruebas de rendimiento en la nube, pruebas de seguridad y pruebas móviles. Proporcionar recursos y oportunidades para que el equipo se mantenga actualizado sobre estos temas, acorde a lo planteado en el estándar ISO/IEC IEEE 29119.
* Habilidades de comunicación y colaboración: La capacitación continua no se limita solo a aspectos técnicos. Es importante mejorar las habilidades de comunicación y colaboración del equipo de SQA. Esto puede incluir talleres o cursos sobre habilidades de comunicación efectiva, trabajo en equipo, resolución de conflictos y gestión de proyectos, sugiriendo utilizar CMMI como modelo a seguir.
* Aseguramiento de la calidad en el ciclo de vida del software: Proporcionar capacitación sobre cómo integrar eficazmente el aseguramiento de la calidad en todas las etapas del ciclo de vida del software, desde la especificación y diseño hasta el despliegue y mantenimiento. Esto implica comprender cómo identificar y prevenir problemas de calidad en cada fase y cómo trabajar de manera efectiva con otros equipos, como desarrollo y gestión de proyectos, siguiendo lo propuesto en los estándares ISO/IEC 25010 e ISO/IEC 12207.
* Actualizaciones en regulaciones y normativas: Proporcionar capacitación sobre las últimas actualizaciones en regulaciones y normativas que afecten al proceso de desarrollo y prueba de software. Esto asegurará que el equipo esté al tanto de los requisitos legales y pueda cumplir con las normativas aplicables. conforme a lo indicado en el estándar ISO/IEC 27001.

# 14. Gestión de riesgos

Se identificaron un par de áreas de riesgo técnico con respecto a la fase de pruebas. Usando el estándar IEEE 730.1-1995, se determinó la inclusión de la siguiente información para cada área de riesgo:

1. Descripción
2. Impacto
3. Monitoreo
4. Mitigación

Es por ello que en el documento SQA\_RM\_V1.0 disponible en el repositorio de la institución, que fue creado con base en el estándar ISO/IEC 27005 se definen las actividades a realizar por cada actividad identificada en cada una de las áreas.

# 15. Bibliografía

"IEEE Standard Glossary of Software Engineering Terminology," in IEEE Std 610.12-1990 , vol., no., pp.1-84, 31 Dec. 1990, doi: 10.1109/IEEESTD.1990.101064."IEEE Standard Glossary of Software Engineering Terminology," in IEEE Std 610.12-1990 , vol., no., pp.1-84, 31 Dec. 1990, doi: 10.1109/IEEESTD.1990.101064.

"IEEE Standard for Software Quality Assurance Plans," in IEEE Std 730-2002 (Revision of IEEE Std 730-1998) , vol., no., pp.1-10, 23 Sept. 2002, doi: 10.1109/IEEESTD.2002.94130.

"IEEE Guide for Software Quality Assurance Planning," in IEEE Std 730.1-1995 , vol., no., pp.1-20, 10 April 1996, doi: 10.1109/IEEESTD.1996.80817.

"IEEE Standard for Software Configuration Management Plans," in IEEE Std 828-1990 , vol., no., pp.1-19, 6 Nov. 1990, doi: 10.1109/IEEESTD.1990.101063.

"IEEE Standard for Software Test Documentation," in IEEE Std 829-1983 , vol., no., pp.1-48, 18 Feb. 1983, doi: 10.1109/IEEESTD.1983.81615.

"IEEE Recommended Practice for Software Requirements Specifications," in IEEE Std 830-1993 , vol., no., pp.1-32, 8 April 1994, doi: 10.1109/IEEESTD.1994.121431.

"IEEE Standard for Software Unit Testing," in ANSI/IEEE Std 1008-1987 , vol., no., pp.1-28, 30 Nov. 1986, doi: 10.1109/IEEESTD.1986.81001.

"IEEE Standard for Software Verification and Validation Plans," in IEEE Std 1012-1986 , vol., no., pp.1-87, 14 Nov. 1986.

"IEEE Recommended Practice for Software Design Descriptions," in IEEE Std 1016-1987 , vol., no., pp.1-16, 13 July 1987, doi: 10.1109/IEEESTD.1987.122643.

"IEEE Standard Software Reviews and Audits," in IEEE Std 1028-1988 , vol., no., pp.1-36, 30 June 1989, doi: 10.1109/IEEESTD.1989.81607.

"IEEE Standard for Software Project Management Plans," in IEEE Std 1058.1-1987 , vol., no., pp.1-28, 31 Aug. 1988, doi: 10.1109/IEEESTD.1988.121942.

"IEEE Standard for Software User Documentation," in IEEE Std 1063-2001, vol., no., pp.1-24, 20 Dec. 2001, doi: 10.1109/IEEESTD.2001.93368.

"IEEE Standard for Developing Software Life Cycle Processes," in IEEE Std 1074-1995 , vol., no., pp.1-106, 26 April 1996, doi: 10.1109/IEEESTD.1996.79663.

"IEEE Standard for Software Maintenance," in IEEE Std 1219-1993 , vol., no., pp.1-45, 2 June 1993, doi: 10.1109/IEEESTD.1993.115570.

Online Browsing Platform. (2015). *ISO 9001:2015(es) Sistemas de gestión de la calidad*. Recuperado 5 de marzo de 2023, de<https://www.iso.org/obp/ui/#iso:std:iso:9001:ed-5:v1:es>

"ISO/IEC/IEEE International Standard - Systems and software engineering - Life cycle processes - Project management," in ISO/IEC/IEEE 16326:2019(E) , vol., no., pp.1-42, 13 Dec. 2019, doi: 10.1109/IEEESTD.2019.8932690.

W. T. Tsai, Xiaoying Bai, R. Paul, Weiguang Shao and V. Agarwal, "End-to-end integration testing design," 25th Annual International Computer Software and Applications Conference. COMPSAC 2001, Chicago, IL, USA, 2001, pp. 166-171, doi: 10.1109/CMPSAC.2001.960613.

Zhang, W., Liu, W., & Wei, B. (2017). Software system testing method based on formal model. https://doi.org/10.1109/icccbda.2017.7951948

Dustin, E., Rashka, J., & Paul, J. (1999). Automated Software Testing: Introduction, Management, and Performance. Addison-Wesley Professional.

Washington, N. M. C. F. O. S., DC. (1994). *Military Standard: Software Development and Documentation*. <https://doi.org/10.21236/ada291266>

Online Browsing Platform. (2021). *ISO 19475:2021 Document management — Minimum requirements for the storage of documents*. Recuperado 5 de marzo de 2023, de<https://www.iso.org/standard/76865.html>

"ISO/IEC/IEEE International Standard - Systems and software engineering -- Software life cycle processes," in IEEE STD 12207-2008 , vol., no., pp.1-138, 31 Jan. 2008, doi: 10.1109/IEEESTD.2008.4475826.

“IEEE 29119 Software testing” in IEEE 29119, vol., no., pp.1-15, doi: 10.1109/IEEESTD.2013.6588537

“ISO/IEC 27001 Sistema de gestión de seguridad de información” in ISO/IEC 27001, vol., no., pp.1-10,

<https://dgsei.edomex.gob.mx/sites/dgsei.edomex.gob.mx/files/manuales/Norma%20ISO%2027001-2013.pdf>

“ISO/IEC 27005 Information technology — Security techniques — Information security risk management” in ISO/IEC 27005, vol., no., pp.1-60, <https://www.amnafzar.net/files/1/ISO%2027000/ISO%20IEC%2027005-2018.pdf>

"IEEE Standard for System, Software, and Hardware Verification and Validation," in IEEE Std 1012-2016 (Revision of IEEE Std 1012-2012/ Incorporates IEEE Std 1012-2016/Cor1-2017) , vol., no., pp.1-260, 29 Sept. 2017, doi: 10.1109/IEEESTD.2017.8055462.

"IEEE/EIA Guide - Industry Implementation of International Standard ISO/IEC 12207 : 1995 (ISO/IEC 12207) - Software Life Cycle Processes - Life Cycle Data," in IEEE/EIA 12207.1-1997 , vol., no., pp.1-38, 27 April 1998, doi: 10.1109/IEEESTD.1998.86377.

# 16. Anexos

Los estándares utilizados se pueden encontrar en la siguiente carpeta: [Carpeta de estándares](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Estandares)

Las plantillas utilizadas se pueden encontrar en la siguiente carpeta: [Carpeta de plantillas](https://github.com/Daniel-Pliego/SQA/tree/main/Recursos/Plantillas)

## Anexo A.1 – Plantillas de SCM para SCI’s con base en la información básica propuesta por la guía dentro del estándar IEEE 1042-1987 y de acuerdo con los requerimientos esenciales planteados por el estándar IEEE 828-1983.

**Tabla de Identificación del Elemento de la Configuración de Software**

|  |  |
| --- | --- |
| **Título del producto de software** | NOMBRE DEL SCI |
| **Localización de referencia** | LOCALIZACIÓN FÍSICA O DIGITAL DONDE SE ENCUENTRA |
| **Autor** | PERSONA QUE HIZO EL MANEJO DE LA CONFIGURACIÓN INICIAL DEL SCI |
| **Fecha de creación** | FECHA EN LA CUAL SE HIZO LA ELABORACIÓN DEL MANEJO DE LA CONFIGURACIÓN DEL SCI |

**Tabla de Historial de Versiones**

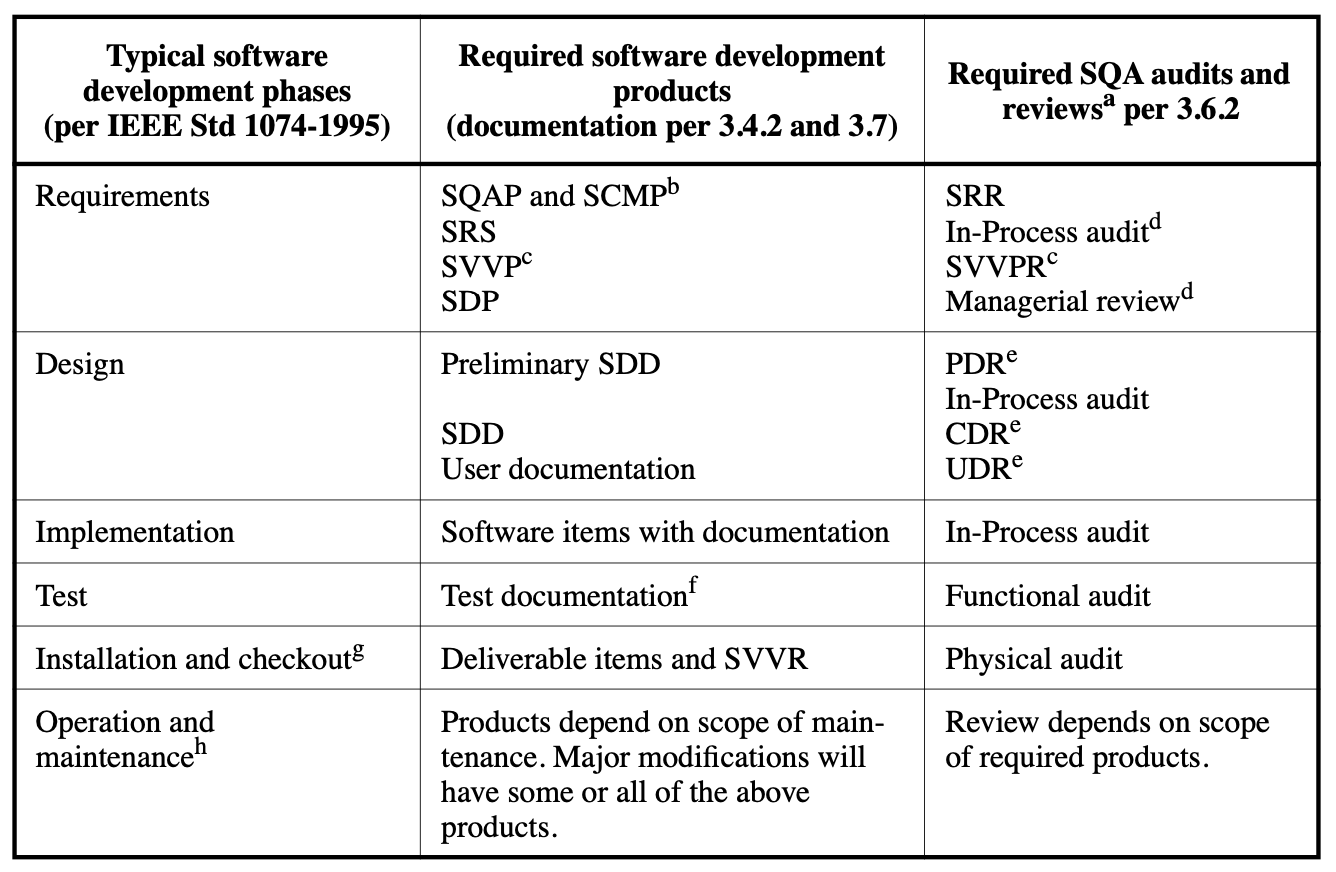
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Versión del SCI** | **Fecha de elaboración** | **Estado de integración del cambio** | **Responsable del cambio** | **Identificador de la versión** |
| VB.VI | dd/mm/aaaa | ACEPTADO / EN REVISIÓN | EQUIPO / PERSONAL QUE INTEGRA EL CAMBIO | IDENTIFICADOR |

**Tabla de Historial de Cambios**

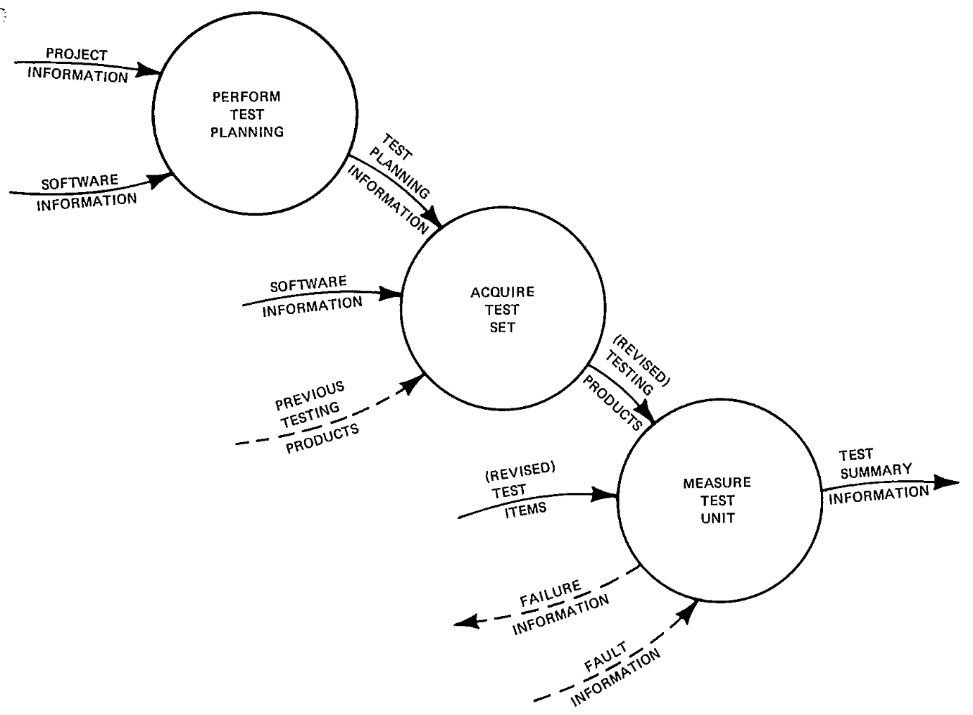
|  |  |  |
| --- | --- | --- |
| **Versión del SCI** | **Fecha de la Solicitud de Cambios de Software** | **Descripción de cambios** |
| VB.VI | dd/mm/aaaa | RESUMEN SENCILLO DE LOS CAMBIOS IMPLEMENTADOS DENTRO DE LA VERSIÓN |

Donde VB se refiere al número de la versión base del SCI y VI se refiere al número de versión intermedia del SCI.

## Anexo A.2 – Tabla 1 – Ejemplo de relaciones y cronograma de revisiones y auditorías requeridas.



## Anexo A.3 – Flujos de datos importantes de las fases de pruebas unitarias de software planteado dentro del estándar IEEE 1008 - 1987.

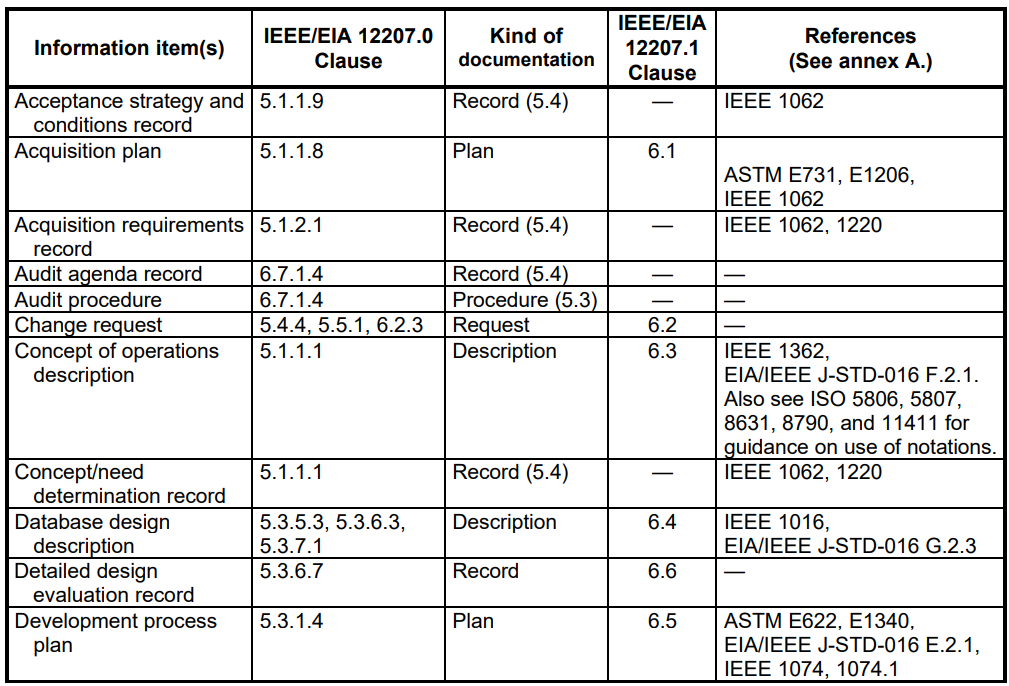


## 

## Anexo A.4 - Acciones correctivas

## 

## Anexo A.5 - Matriz de elementos de información planteado en el estándar IEEE/EIA 12207.1 - 1997



s

