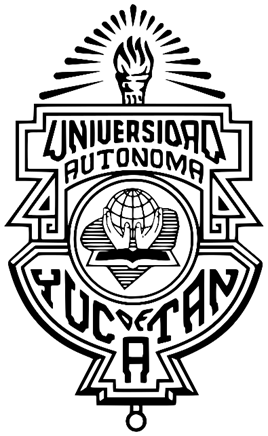
Plan de SQA



Alumnos

Álvarez Vázquez Jesús Miguel

Avila Pacheco David de Jesús

Gonzalez Herrera Miguel Humberto

Echeverria Leon Eduardo Leonel

Chan Zurita Mario Jesús

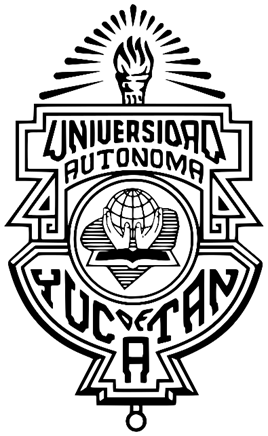
Edwin de Jesús León Bojórquez

Universidad Autónoma de Yucatán

Facultad de Matemáticas

Licenciatura en Ingeniería de Software

Aseguramiento de la Calidad de Software



CopyPaste

Plan de aseguramiento de la calidad

Aseguramiento de la calidad

Documento del Plan de SQA

Versión 2.0

**Autores**

Álvarez Vázquez Jesús Miguel

Avila Pacheco David de Jesús

Gonzalez Herrera Miguel Humberto

Echeverria Leon Eduardo Leonel

Chan Zurita Mario Jesús

**Fecha de finalización**

29 de mayo de 2022

**Persona(s) que aprobaron el documento**

Comité de revisión de diseño formal (FDR):

Álvarez Vázquez Jesús Miguel

Avila Pacheco David de Jesús

Gonzalez Herrera Miguel Humberto

Echeverria Leon Eduardo Leonel

Chan Zurita Mario Jesús

**Fecha de aprobación**

29 de mayo de 2022

**Firma(s) del auto(es) y persona(s) que lo aprobaron**

Álvarez Vázquez Jesús Miguel

Avila Pacheco David de Jesús

Gonzalez Herrera Miguel Humberto

Echeverria Leon Eduardo Leonel

Chan Zurita Mario Jesús

**Lista de versiones y revisiones anteriores**

| **Fecha** | **Versión** | **Autor** |
| --- | --- | --- |
| 05/05/2022 | 1.0 | Equipo de SQA |
| 12/05/2022 | 1.1 | Equipo de SQA |
| 15/05/2022 | 1.2 | Equipo de SQA |
| 26/05/2022 | 1.3 | Equipo de SQA |
| 28/05/2022 | 1.4 | Equipo de SQA |
| 29/05/2022 | 2.0 | Equipo de SQA |

| **Fecha** | **Revisión** | **Autor** |
| --- | --- | --- |
| 05/05/2022 | 1.0 | Comité de revisión de diseño formal (FDR) |
| 12/05/2022 | 1.1 | Comité de revisión de diseño formal (FDR) |
| 15/05/2022 | 1.2 | Comité de revisión de diseño formal (FDR) |
| 26/05/2022 | 1.3 | Comité de revisión de diseño formal (FDR) |
| 28/05/2022 | 1.4 | Comité de revisión de diseño formal (FDR) |
| 29/05/2022 | 2.0 | Comité de revisión de diseño formal (FDR) |

**Lista de circulación**

El documento será subido a un repositorio, en el cual solo los miembros de este podrán obtenerlo y visualizarlo (requiriendo del correo electrónico de cada miembro para dicho procedimiento).

**Restricciones de confidencialidad**

No se puede tomar fotografías, capturas y realizar copias del documento.

Solamente el personal autorizado puede realizar cambios en el documento y con la condición de contar con la autorización aprobada para hacerlo.

**Índice**

[**Propósito**](#_heading=h.30j0zll) **7**

[**Alcance**](#_heading=h.6aif4m5c7zo0) **7**

[**Documentos referenciados**](#_heading=h.vl55b9skllts) **7**

[**Gestión**](#_heading=h.3znysh7) **8**

[Organización](#_heading=h.2et92p0) 8

[Actividades](#_heading=h.tyjcwt) 11

[**Documentación**](#_heading=h.1t3h5sf) **14**

[Especificación de requisitos de software (ERS)](#_heading=h.dzbfqe1dn6su) 15

[Documento de análisis y diseño (DAD)](#_heading=h.gjxbv2cm8gsz) 15

[Configuración de software (CS)](#_heading=h.et9bmrxgz2o) 16

[Manual de usuario (MU)](#_heading=h.7e8k3m8xyrxb) 16

[Manual de operación (MO)](#_heading=h.ibiutv12nawo) 16

[Manual de mantenimiento (MM)](#_heading=h.myj6ozzf2ygo) 17

[Registro de rastreo (RR)](#_heading=h.uch3aawp5b7g) 17

[Plan de pruebas de sistema (PPS)](#_heading=h.7igoclye4pbr) 18

[Plan de pruebas de integración (PPI)](#_heading=h.pcoycn6ph5hr) 18

[**Estándares, prácticas, convenciones y métricas.**](#_heading=h.mbfd9t8toxlt) **18**

[Fase de requerimientos](#_heading=h.goa38hp6qq1e) 18

[Fase de análisis y diseño](#_heading=h.7r6ku8xmg87r) 20

[Fase de construcción](#_heading=h.7bhscdopcv3) 22

[Fase de integración y pruebas](#_heading=h.u92w4ug4z2v4) 24

[Fase de cierre](#_heading=h.ush5j7y543bb) 25

[**Revisiones y auditorías**](#_heading=h.ghhihnqncu70) **27**

[Objetivo](#_heading=h.1ksv4uv) 27

[Revisión de Especificación de requisitos de software (RERS)](#_heading=h.2jxsxqh) 28

[Revisión del documento de análisis y diseño (RDAD)](#_heading=h.z337ya) 28

[Revisión de la configuración de software (RCS)](#_heading=h.rq64fxd0hths) 28

[Revisión del manual de usuario (RMU)](#_heading=h.dv1xabv3f8sq) 28

[Revisión del manual de operación (RMO)](#_heading=h.cxfhkwgfvvh5) 29

[Revisión del manual de mantenimiento (RMM)](#_heading=h.in135c7q2x0h) 29

[Revisión del registro de rastreo (RRR)](#_heading=h.71344iwpgfb4) 29

[Revisión del plan de pruebas de sistema (RPPS)](#_heading=h.od3al3pp7pt7) 29

[Revisión del plan de pruebas de integración (RPPI)](#_heading=h.ynk09vz6llkl) 30

[Auditoría Funcional](#_heading=h.g5hw3i3rk951) 31

[Auditoría Física](#_heading=h.o5w1sy19pid6) 31

[Auditorías internas al proceso](#_heading=h.dl6xbxkge3k) 32

[**Pruebas**](#_heading=h.1ci93xb) **32**

[**Informe de problemas y acción correctiva**](#_heading=h.3whwml4) **33**

[Reporte del proceso de auditorías](#_heading=h.2bn6wsx) 33

[Presentación del reporte del proceso auditoría](#_heading=h.qsh70q) 33

[Procedimiento de escalamiento para la Resolución de no ocurrencia en el Proceso de Reporte de Auditoría](#_heading=h.3as4poj) 33

[**Técnicas, herramientas y metodologías**](#_heading=h.1pxezwc) **34**

[Herramientas](#_heading=h.49x2ik5) 34

[Técnicas](#_heading=h.2p2csry) 34

[Metodologías](#_heading=h.147n2zr) 34

[**Control del código**](#_heading=h.3o7alnk) **35**

[**Control de medios**](#_heading=h.23ckvvd) **36**

[Medio de almacenamiento](#_heading=h.ihv636) 36

[Proceso de copias de seguridad](#_heading=h.sq0al6gwxmkx) 36

[Puntos de control](#_heading=h.zae2dsnpmyku) 36

[Desgaste de los medios](#_heading=h.v5pqieh2vxdc) 37

[Vulneración de los medios](#_heading=h.ukcm4qndtf9s) 37

[**Recopilación, mantenimiento y retención de registros**](#_heading=h.41mghml) **37**

[**Procedimiento de control de la documentación**](#_heading=h.uiqslnndt2go) **37**

[**Entrenamiento**](#_heading=h.2grqrue) **38**

[**Anexos**](#_heading=h.woa7ibmyws20) **40**

[Anexo A](#_heading=h.lmvlis5rhvgp) 40

[Anexo B](#_heading=h.kwh91ii28t65) 48

[Anexo C](#_heading=h.mlasqw53agvb) 49

[Anexo D](#_heading=h.pfxltsmon2tj) 51

[Anexo E](#_heading=h.l1rlsc5bila9) 51

[Anexo F](#_heading=h.5vciwubywrnm) 51

[Anexo G](#_heading=h.em5nqcxwuz0) 51

[Anexo H](#_heading=h.awao5m27vssy) 52

[Anexo I](#_heading=h.rsevrxvzaiqq) 53

[Anexo J](#_heading=h.lufklq4vf590) 54

# 

# Propósito

El objetivo de este plan es establecer las actividades necesarias para garantizar la calidad del software “Hexanary”. Hexanary es una calculadora de programador la cual será desarrollada para plataformas móviles con sistema operativo android.

Para cada actividad se indicarán los atributos de calidad relevantes, los métodos de evaluación y los responsables. Garantizando el cumplimiento de los estándares y de las metodologías, se brindarán los elementos de apoyo a la gestión del proyecto para realizar verificaciones sobre la adecuación al proceso y así detectar desvíos que puedan resultar en acciones correctivas en etapas tempranas.

# 

# Alcance

El plan de SQA está dirigido a la unidad de SQA y al equipo de desarrollo de software.

El alcance del plan abarca las fases del ciclo de vida relacionadas con: elaboración, construcción, evaluación y despliegue.

Los productos cuya calidad deberá ser asegurada son:

* Especificación de requisitos de software
* Documento de análisis y diseño
* Configuración de software
* Manual de usuario
* Manual de operación
* Manual de mantenimiento
* Registro de rastreo
* Plan de pruebas del sistema
* Plan de pruebas de integración

# Documentos referenciados

[B1] IEEE Std 730.1-1995, IEEE Guide for Software Quality Assurance Planning.

[B2] IEEE Std 730-2002, IEEE Software Quality Assurance Plans.

[B3] IEEE Std 730-2014, IEEE Software Quality Assurance Processes.

[B4] IEEE Std 1028-2008, IEEE Standard for Software. Reviews and Audits.

[B5] IEEE Std 1008-1987, IEEE Standard for Software. Unit Testing.

[B6] ISO/IEC/IEEE 29119-2013, Software Testing Standard.

[B7] ISO 10007-2017, Guidelines for configuration management.

[B8] ANSI / IEEE–STD 830-1998 Guide for Software Requirements Specifications.

[B9] ANSI / IEEE – STD 1016-2009 Recommended Practice for Software Design Descriptions.

[B10] ANSI / IEEE – STD 1063-2001 Standard for Software User Documentation.

[B11] ISO 9000-2015, Quality management systems.

[B12] ISO/IEC 14598-1998, Product evaluation.

[B13] ISO/IEC 19505-2:2012, Object Management Group Unified Modeling Language.

[B14] ISO / IEC / IEEE 12207: 2017, Software life cycle processes.

[B15] IEEE 1012-1986, Standard for Software Verification and Validation Plans.

[B16] ISO/IEC 25000-2005, System and Software Quality Requirements and Evaluation.

[B17] IEEE 829-2008, Standard for software Test Documentation.

[B18] ISO 14764-2006, Software Life Cycle Processes: maintenance.

[B19] IEEE 1219-1998, Standard for Software Maintenance.

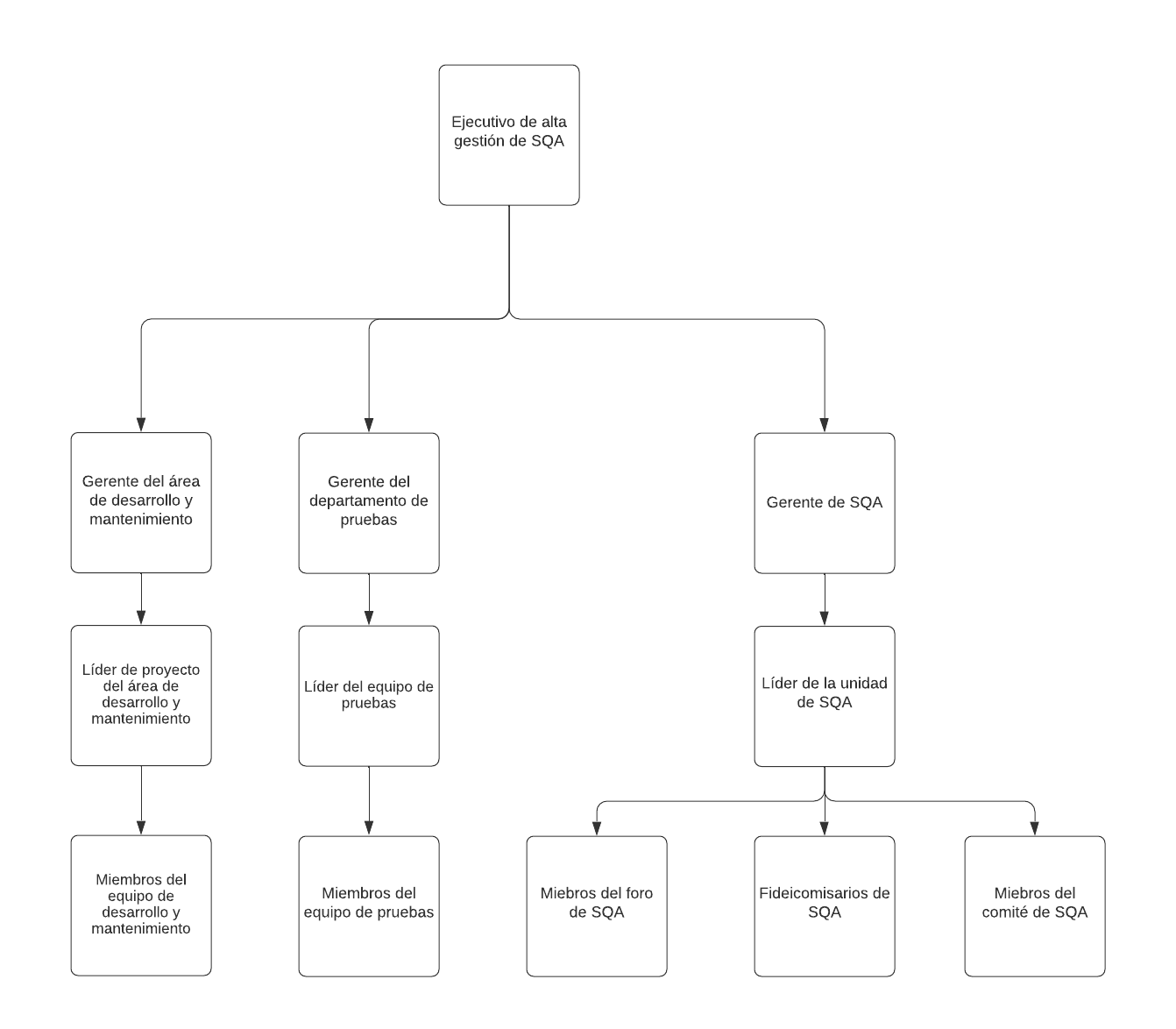
# Gestión

La gestión del proyecto está a cargo del administrador del proyecto, sin embargo será monitoreada tanto por este, como por el responsable de SQA. Se intenta controlar que las actividades se ajusten al plan propuesto y minimizar posibles desviaciones

## Organización

La línea directa de trabajo seguida se basa en las 5 fases del desarrollo de software:

* Requerimientos.
* Análisis y diseño.
* Construcción.
* Integración y pruebas.
* Cierre.



La estructura del equipo de trabajo será de la siguiente manera:

| Rol | Descripción | Responsable |
| --- | --- | --- |
| Ejecutivo de alta gestión. | Ejecutivo encargado de administrar y dirigir a todos los departamentos involucrados. | Jesús Álvarez |
| Gerente del área de desarrollo y mantenimiento. | Encargado de organizar y dirigir a los miembros del departamento de desarrollo y mantenimiento, así como representar el departamento ante alta gestión. | David Ávila |
| Líder del proyecto del área de desarrollo y mantenimiento. | Encargado de representar y supervisar al equipo de desarrollo y mantenimiento. | Eduardo Echeverría |
| Miembros del equipo de desarrollo y mantenimiento. | Encargados de realizar actividades de mantenimiento de software y de SQA, tal como mejorar las perspectivas de lograr los requisitos funcionales y de gestión.  Encargados de asegurar que el software se encuentre ajustado a los requisitos técnicos funcionales. | Mario Chan  David Ávila  Jesús Álvarez  Eduardo Echeverría  Miguel González |
| Gerente del departamento de pruebas | Encargado de organizar y dirigir a los miembros del departamento de pruebas, así como representar el departamento ante alta gestión. | Eduardo Echeverría |
| Líder del equipo de pruebas. | Encargado de representar y supervisar al equipo de pruebas. | Mario Chan |
| Miembros del equipo de pruebas. | Encargados de aplicar los estándares de calidad apropiados, así como aplicar las estrategias de testeo necesarias. | Mario Chan  David Ávila  Jesús Álvarez  Eduardo Echeverría  Miguel González |
| Gerente de SQA. | Encargado de la preparación de los planes de desarrollo del sistema de SQA, así como del control general de la implementación del programa anual de actividades de SQA, es el encargado de la presentación y defensa de los problemas de SQA a la alta gestión. | Miguel Gonzáles |
| Líder de la unidad de SQA. | Encargado de representar y supervisar al equipo de la unidad de SQA. | Mario Chan |
| Fideicomisarios de SQA. | Encargados de auxiliar al gerente de la unidad en la preparación de instrucciones y recopilación de datos para el cálculo de métricas.  Encargados de los reportes de fallas e incumplimientos relacionados con la calidad del software.  Encargados del programa de capacitación de la unidad de SQA. | Mario Chan  David Ávila  Jesús Álvarez  Eduardo Echeverría  Miguel González |
| Miembros del comité de SQA. | Los miembros del comité permanente se encargan del control de cambios del software así como de las acciones correctivas procedimientos herramientas de desarrollo de métodos y métricas de calidad.  Los miembros del comité ad hoc se encargan de la solución de fallas de software y la elaboración de métricas para un proceso o producto en específico. | Mario Chan  David Ávila  Jesús Álvarez  Eduardo Echeverría  Miguel González |
| Miembros del foro de SQA. | Encargados de proporcionar apoyo a los integrantes de las demás áreas de la unidad de SQA. | Mario Chan  David Ávila  Jesús Álvarez  Eduardo Echeverría  Miguel González |

*Tabla 1. Roles del equipo*

## Actividades

| **Descripción** | **Entradas** | **Salidas** | **Rol y responsabilidades** | |
| --- | --- | --- | --- | --- |
| A1. Realización de la fase de requerimientos | | | | |
| A1.1. Verificar la Especificación de Requerimientos | Especificación de requerimientos | Reporte de verificación | Gerente del área de desarrollo y mantenimiento:   * Encargado de dirigir la definición y verificación de requerimientos.   Líder del proyecto del área de desarrollo y mantenimiento:   * Encargado de supervisar al equipo de desarrollo y mantenimiento.   Miembros del equipo de desarrollo y mantenimiento:   * Encargados de definir y verificar los requerimientos. | |
| A1.2. Validar la Especificación de Requerimientos | Especificación de requerimientos | Reporte de validación | Gerente del departamento de pruebas:   * Encargado dirigir las pruebas con relación a la especificación de requerimientos.   Líder del equipo de pruebas:   * Encargado de supervisar al equipo de pruebas.   Miembros del equipo de pruebas:   * Encargados de realizar las pruebas funcionales. | |
| A1.3. Verificar el Plan de Pruebas de Sistema | Plan de pruebas de sistema | Reporte de verificación | Gerente del departamento de pruebas:   * Encargado de la definición del plan de pruebas.   Líder del equipo de pruebas:   * Encargado de supervisar al equipo de pruebas.   Miembros del equipo de pruebas:   * Encargado de la verificación del plan de pruebas. | |
| A1.4. Verificar el Manual de Usuario | Manual de usuario | Reporte de verificación | Gerente de SQA:   * Encargado de la definición del manual de usuario.   Líder del equipo de la unidad de SQA:   * Encargado de supervisar al comité de SQA.   Miembros del comité de SQA:   * Encargados de la verificación del manual de usuario. | |
| A2. Realización de la fase de Análisis y Diseño | | | | |
| A2.1. Verificar el Análisis y Diseño y el Registro de Rastreo | Documento de análisis y diseño  Reporte de rastreo | Reporte de verificación | Gerente de SQA:   * Encargado de controlar y gestionar a la unidad SQA encargada del diseño del sistema, así como del Registro de Rastreo.   Líder de unidad de SQA:   * Representante de la unidad SQA encargada del diseño y del Registro de Rastreo.   Miembros del comité de SQA:   * Encargados del análisis y resolución de conflictos que puedan presentarse durante el análisis del diseño.   Miembros del foro de SQA:   * Encargados del análisis de casos de diseños erróneos y su correspondiente propuesta de mejora.   Fideicomisarios de SQA:   * Encargados de monitorear el procedimiento aplicado durante la fase de diseño. | |
| A2.2. Validar el Análisis y Diseño | Documento de análisis y diseño | Reporte de validación | Gerente del área de desarrollo y mantenimiento:   * Encargado de definir la estructura del documento de análisis y diseño.   Líder del proyecto del área de desarrollo y mantenimiento:   * Encargado de supervisar al equipo de desarrollo y mantenimiento.   Miembros del equipo de desarrollo y mantenimiento:   * Encargados de verificar el documento de análisis y diseño. | |
| A2.3. Verificar el Plan de Pruebas de Integración | Plan de pruebas de integración | Reporte de verificación | Gerente del departamento de pruebas:   * Encargado dirigir las pruebas con relación a al análisis y diseño   Líder del equipo de pruebas:   * Encargado de supervisar las pruebas realizadas   Miembros del equipo de pruebas:   * Encargados de realizar las pruebas funcionales | |
| A3. Realización de la fase de construcción | | | | |
| A3.1. Verificar el Registro de Rastreo | Registro de rastreo | Reporte de verificación | Gerente del área de desarrollo y mantenimiento:   * Encargado de definir la estructura del documento de registro de rastreo.   Líder del proyecto del área de desarrollo y mantenimiento:   * Encargado de supervisar al equipo de desarrollo y mantenimiento.   Miembros del equipo de desarrollo y mantenimiento:   * Encargados de verificar el documento de registro de rastreo. | |
| A4. Realización de la fase de integración y pruebas | | | | |
| A4.1. Verificar el Manual de Operación | Manual de operación | Reporte de verificación | Gerente de pruebas:   * Encargado de definir la estructura del documento de Manual de Operación.   Líder de pruebas:   * Encargado de supervisar al equipo asignado para la construcción del manual. | |
| A4.2. Verificar el Manual de Usuario | Manual de usuario | Reporte de verificación | Gerente de pruebas líder de SQA:   * Encargado de definir la estructura del Manual de usuario. | |
| A5. Realización de la fase de cierre | | | | |
| A5.1. Verificar el Manual de Mantenimiento | Manual de mantenimiento | Reporte de verificación | Gerente de SQA:   * Encargado de controlar al equipo encargado de construir el Manual de Mantenimiento.   Líder de SQA:   * Representante del equipo de SQA, encargado de supervisar el desarrollo del Manual de Mantenimiento.   Comité de SQA:   * Encargados de actualizar procedimientos que requieran cambio dentro del Manual de Mantenimiento.   Foros:   * Encargados de definir métricas de calidad para el Manual de Mantenimiento. | |

*Tabla 2. Actividades a realizar para el plan de SQA*

# Documentación

Para asegurar que la implementación de la aplicación satisface los requerimientos definidos es necesaria la elaboración de los siguientes documentos:

## Especificación de requisitos de software (ERS)

El documento ERS debe describir de forma clara y precisa cada requerimiento esencial (funciones, rendimiento, restricciones de diseño y atributos) del software y de interfaces externas.

Para su elaboración se usará como referencia el estándar IEEE 830-1998 “Recommended Practices for Software Requirements Specifications” que describe el contenido y cualidades de un buen documento de especificación de requerimientos de software.

La verificación del documento de especificación de requerimientos será con base en una “Revisión de Diseño Formal” que describe un proceso o reunión durante la cual se presenta un producto de trabajo, o un conjunto de productos de trabajo a las partes interesadas para comentarios o aprobación y estas tienen que estar aprobadas para pasar a la siguiente fase del proyecto de desarrollo de software. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

La validación del documento de especificación de requerimientos será con base en “Pruebas Funcionales” que se llevan a cabo mediante la ejecución de los casos de prueba registrados a través del programa de prueba. El plan de pruebas será elaborado siguiendo el estándar IEEE 829-2008 “Software and System Test Documentation” que describe el contenido y si los productos de desarrollo de una actividad determinada se ajustan a los requisitos de esa actividad y si el sistema y/o el software satisfacen su uso previsto y las necesidades.

## Documento de análisis y diseño (DAD)

El DAD debe representar cómo el software se estructurará para satisfacer los requisitos del ERS. El DAD debe describir los componentes y subcomponentes del diseño del software, incluidas las bases de datos y las interfaces internas.

Para su elaboración se usará como referencia el estándar IEEE 1016-1998 “Recommended Practice for Software Design Descriptions” que describe el contenido de la información necesaria y las recomendaciones para una organización de descripciones de diseño de software.

La verificación del documento de análisis y diseño será con base en una “Revisión de Diseño Formal” que describe un proceso o reunión durante la cual se presenta un producto de trabajo, o un conjunto de productos de trabajo a las partes interesadas para comentarios o aprobación y estas tienen que estar aprobadas para pasar a la siguiente fase del proyecto de desarrollo de software. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

La validación del documento de análisis y diseño será con base en “Pruebas Funcionales” que se llevan a cabo mediante la ejecución de los casos de prueba registrados a través del programa de prueba. El plan de pruebas será elaborado siguiendo el estándar IEEE 829-2008 “Software and System Test Documentation” que describe el contenido y si los productos de desarrollo de una actividad determinada se ajustan a los requisitos de esa actividad y si el sistema y/o el software satisfacen su uso previsto y las necesidades.

## Configuración de software (CS)

La CS debe documentar los métodos que se utilizarán para identificar elementos de software, controlar e implementar cambios, y registrar y reportar el estado de implementación del cambio.

Para su elaboración se usarán como referencia los estándares IEEE 828-1990 “Standard for Software Configuration Management Plans” que describe los contenidos mínimos requeridos de un Plan de Gestión de Configuración de Software y define las actividades específicas que se abordarán y sus requisitos para cualquier parte del ciclo de vida de un producto de software, y IEEE 1042-1987 (Reaff 1993) “Guide to Software Configuration Management” que proporciona información sobre los diversos factores que deben tenerse en cuenta y sugerencias y ejemplos detallados de planes para la configuración de software.

La verificación del documento de configuración de software será con base en una “Revisión de Diseño Formal” que describe un proceso o reunión durante la cual se presenta un producto de trabajo, o un conjunto de productos de trabajo a las partes interesadas para comentarios o aprobación y estas tienen que estar aprobadas para pasar a la siguiente fase del proyecto de desarrollo de software. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Manual de usuario (MU)

El MU debe especificar y describir los datos requeridos y las entradas de control, las secuencias de entrada, las opciones, las limitaciones del programa y otras actividades o elementos necesarios para la ejecución exitosa del software. Se identificarán todos los mensajes de error y se describirán las acciones correctivas.

Para su elaboración se usará como referencia el estándar IEEE 1063-1987 “Standard for Software User Documentation” que describe los requisitos mínimos sobre la estructura, el contenido de la información y aborda las consideraciones editoriales y estilísticas sólo cuando afectan la estructura y el contenido para el manual de usuario.

La verificación del documento de manual de usuario será con base en una revisión por pares de tipo “Inspección” que su objetivo es detectar errores y desviaciones de los estándares y es de carácter correctiva. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Manual de operación (MO)

El MO documento electrónico o impreso que contenga la información indispensable para la instalación y administración del software, así como el ambiente de operación. Éste deberá ser redactado en términos comprensibles al personal responsable de la operación.

Para su elaboración se usará la plantilla proporciona por el estándar IEEE 730.1-1995 “Guide for Software Quality Assurance Planning” que describe los enfoques de las buenas prácticas de garantía de la calidad del software en apoyo de IEEE Std 730-1989, estándar IEEE para planes de garantía de la calidad del software.

Plantilla del estándar IEEE 730.1-1995:

a) Operating instructions that include

1) An introduction

2) Run schedules

3) Setup requirements/procedures

4) Run control procedures

5) Error procedures

6) Security procedures

7) Distribution procedures

8) Backup and recovery procedures

9) Restart procedures

10) Termination procedures

11) Tutorial and practice procedures

b) Specifications for the system, including environmental requirements

c) Input/output specifications

d) Auditing controls

La verificación del documento de manual de operación será con base en una “Revisión de Diseño Formal” que describe un proceso o reunión durante la cual se presenta un producto de trabajo, o un conjunto de productos de trabajo a las partes interesadas para comentarios o aprobación y estas tienen que estar aprobadas para pasar a la siguiente fase del proyecto de desarrollo de software. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Manual de mantenimiento (MM)

El MM debe contener instrucciones para el soporte y mantenimiento del producto de software, tales como procedimientos para corregir defectos, instalación de mejoras y prueba de todos los cambios. Todo el hardware y las especificaciones de configuración del software requeridas para mantener el software deben describirse en detalle.

Para su elaboración se usará como referencia el estándar IEEE 1219-1998 “Software Maintenance” que describe el proceso para administrar y ejecutar actividades de mantenimiento de software y el contenido de un manual de mantenimiento.

La verificación del documento de manual de mantenimiento será con base en una revisión por pares de tipo “Inspección” que su objetivo es detectar errores y desviaciones de los estándares y es de carácter correctiva. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Registro de rastreo (RR)

El RR debe contener información sobre el entorno operativo cuando un componente de software no funciona como debería, esto con relación entre requerimientos, elementos análisis y diseño, componentes y planes de pruebas.

Para su elaboración se usará la plantilla “Reporte de rastreo” (ver Anexo A) que describe el contenido y formato del reporte de rastreo.

La verificación del documento de registro de rastreo será con base en una revisión por pares de tipo “Inspección” que su objetivo es detectar errores y desviaciones de los estándares y es de carácter correctiva. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Plan de pruebas de sistema (PPS)

El PPS debe contener las pruebas requeridas para el cumplimiento de los requerimientos especificados.

Para su elaboración se usará como referencia el estándar IEEE 829-2008 “Software and System Test Documentation” que describe el contenido y si los productos de desarrollo de una actividad determinada se ajustan a los requisitos de esa actividad y si el sistema y/o el software satisfacen su uso previsto y las necesidades.

La verificación del documento de plan de pruebas de sistema será con base en una revisión por pares de tipo “Inspección” que su objetivo es detectar errores y desviaciones de los estándares y es de carácter correctiva. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Plan de pruebas de integración (PPI)

El PPI debe describir el orden de integración de los componentes o subsistemas, guiado, por la parte arquitectónica del análisis y diseño y las pruebas que se aplicarán para verificar la interacción entre los componentes.

Para su elaboración se usará como referencia el estándar IEEE 829-2008 “Software and System Test Documentation” que describe el contenido y si los productos de desarrollo de una actividad determinada se ajustan a los requisitos de esa actividad y si el sistema y/o el software satisfacen su uso previsto y las necesidades.

La verificación del documento de plan de pruebas de integración será con base en una revisión por pares de tipo “Inspección” que su objetivo es detectar errores y desviaciones de los estándares y es de carácter correctiva. El plan de revisión será elaborado siguiendo el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

# Estándares, prácticas, convenciones y métricas.

Para cada fase del proceso de desarrollo se requieren métricas, prácticas, convenciones y métricas. A continuación, se listan y describen para cada una de las fases.

## Fase de requerimientos

Para la fase de análisis, durante la cual se realiza un proceso de descubrimiento, refinamiento, modelado y especificación de los requisitos del sistema, se requieren seguir estándares, convenciones, métricas y prácticas con el objetivo de asegurar la calidad del documento de Especificación de Requisitos (ERS).

***Estándares***

Los estándares a emplear durante esta fase son:

* **Estándar IEEE 830-1998 “** **Guide for Software Requirements Specifications”**, el cual es utilizado principalmente como una guía y plantilla sobre la información, estructura y organización que deberá contener el documento de Especificación de Requisitos.
* **Estándar ISO / IEC / IEEE 12207-2017 “Software life cycle processes”,** el cual se enfoca en el proceso del ciclo de vida del software, pero aplicado para esta fase en específico define las tareas a realizar, tales como la definición de las necesidades del sistema y de las partes interesadas (stakeholders) y el análisis del negocio (para la obtención de los requisitos).
* **Estándar IEEE 1028-2008 “Software Reviews and Audits”,** donde se especifican los elementos a inspeccionar (para este caso, el documento de especificación de requisitos) y los responsables de realizar la actividades de revisión y auditoría.
* **Estándar IEEE 1012-1986 “Standard for Software Verification and Validation Plans”,** es un estándar de proceso que aborda todos los procesos del ciclo de vida del sistema, software y hardware, incluidos los grupos de procesos acuerdo, habilitación de proyectos organizacionales, proyecto, técnico, implementación de software, soporte de software y reutilización de software.

***Prácticas***

* **Preferir requisitos ejecutables sobre documentación estática.** En la manera posible, especificar los requisitos en forma de “pruebas de usuario” que puedan ejecutarse y diseñar para futuras pruebas de desarrollo ejecutables, en lugar de documentación “estática” no ejecutable.
* **Trazabilidad.** Tanto para los requisitos como para el documento en sí se debe considerar el atributo de calidad de trazabilidad, por lo cual debe verse reflejada en una matriz que permite analizar el impacto que tendría en el sistema un cambio en alguno de los requisitos. Para dicha práctica se colocará en cada requisito la referencia que tiene con otros requisitos (referencias cruzadas); asimismo, el requisito también deberá contar con la referencia de origen (sea persona u otro documento) y ser trazable con otros documentos incluyendo información con otros documentos con los cuales comparte relación.
* **Empleo de técnicas para la obtención de requisitos.** Todo el equipo debe tener un entendimiento básico de una técnica de modelado, incluyendo los clientes. Por ejemplo, tomar unos momentos para explicar qué es una tarjeta CRC y por qué se utiliza, todo esto con el propósito de ayudar a facilitar en el proceso de obtención de requisitos.
* **Adoptar la terminología de los clientes.** No se debe insistir en que los clientes serán capaces de entender la terminología técnica del desarrollo, por lo tanto, es con la terminología del cliente que se deberá emplear para el modelado del sistema. Un artefacto útil para consolidar está práctica es elaborar y mantener un glosario de términos de negocio.
* **Resolución de conflictos a través de la negociación.** Para asegurar que todas las partes interesadas en el proyecto estén satisfechas y tengan constancias de las dificultades del proyecto, el analista de requisitos deberá tener conocimientos de las técnicas para la resolución de conflictos, particularmente enfocado a dos en concreto: la **colaboración** consiste en llegar en un acuerdo mutuo de todas las partes para satisfacer sus necesidades; **comprometerse** es una estrategia donde cada parte hace concesiones para alcanzar una satisfacción mínima en los interesados.

***Convenciones***

* **Control de versiones.** Para identificar las versiones del conjunto de requisitos aprobados se emplea la plantilla para el control de la documentación (ver Anexo B) donde se asignan los cambios realizados entre la anterior versión y la nueva.
* **Requisitos en una única ubicación.** Los requisitos se mantendrán en una única ubicación para su fácil acceso y localización. La documentación referente a los requisitos del sistema se encontrará en un repositorio privado de Github (con control de acceso a usuarios del equipo de desarrollo) y contarán con un respaldo en drive.

***Métricas***

Las métricas tienen como principal objetivo el verificar si los requisitos del sistema y si la Especificación de Requisitos del Sistema (ERS) cumplen con los atributos de calidad propuestos, con la finalidad de que el software se desarrolle acorde a su especificación y cumple las necesidades de los clientes. Se tomarán como criterios de evaluación la completitud del documento, si se detectan ambigüedades y contradicciones en los requisitos, si el documento cuenta con todas las secciones indicadas en el estándar IEEE 830, entre otros criterios definidos en los instrumentos de evaluación. Para la evaluación del documento de Especificación de Requisitos se emplearán listas de verificación donde se evalúan la trazabilidad, atributos de calidad, estructura y organización, completitud y claridad del documento (ver Anexo C).

## Fase de análisis y diseño

Durante el proceso de análisis y diseño se utiliza la información recolectada en la etapa de requerimientos del producto. La principal tarea de la etapa de diseño es desarrollar un modelo para las especificaciones del producto o componentes del sistema. Los estándares, convenciones, métricas y prácticas a emplear para asegurar la calidad de este proceso se listan y describen a continuación:

***Estándares***

Los estándares a usar en esta fase son:

* + **Estándar IEEE 1016-2009 “Recommended Practice for Software Design Descriptions”,** el cual describe los diseños de software y establece el contenido de la información y la organización de una descripción de diseño de software.
  + **Estándar ISO / IEC / IEEE 12207-2017 “Software life cycle processes”,**  el cual se enfoca en el proceso del ciclo de vida del software, pero aplicado para esta fase en específico define las tareas a realizar, tales como la definición de los módulos del sistema, la arquitectura del sistema, y definición de servicios.
  + **Estándar ISO/IEC 19505-2:2012 “Object Management Group”,** Ofrece una definición formal de un metamodelo común basado en MOF que especifica la sintaxis abstracta de UML, además de una explicación detallada de la semántica de cada concepto de modelado UML y una especificación de los elementos de notación legibles por humanos para representar los conceptos de modelado UML individuales, así como reglas para combinarlos en una variedad de diferentes tipos de diagramas correspondientes a diferentes aspectos de los sistemas modelados.
  + **Estándar IEEE 1028-2008 “Software Reviews and Audits”,** donde se especifican los elementos a inspeccionar (para las descripciones de diseño y diagramas del sistema) y los responsables de realizar las actividades de inspección.
  + **Estándar IEEE 1012-1986 “Standard for Software Verification and Validation Plans”,** es un estándar de proceso que aborda todos los procesos del ciclo de vida del sistema, software y hardware, incluidos los grupos de procesos acuerdo, habilitación de proyectos organizacionales, proyecto, técnico, implementación de software, soporte de software y reutilización de software.

***Prácticas***

* **Uso de patrones arquitectónicos:** Debido a la alta demanda de software, y a los cortos tiempos de desarrollo impuestos por el mercado, el uso de patrones arquitectónicos ofrece un esquema a problemáticas comunes.
* **Empleo de diagramas de UML.** Mantener el diseño simple y útil, realizando únicamente lo necesario para ayudar a entender lo que se desarrollará, apoyándose de diagramas de UML.

***Convenciones***

* **Uso de UML: Libro “ABOUT THE UNIFIED MODELING LANGUAGE SPECIFICATION”:** ayuda a capturar mediante un conjunto de símbolos y diagramas a comunicar la idea de un sistema (software orientado a objetos), a quien esté involucrado en su proceso de desarrollo sirviendo.

***Métricas***

Las principales métricas usadas para evaluar un diseño de software están orientadas a objetos, puesto que UML está centrada en este paradigma, pero no exclusivo de esta.

* **DIT (Depth of Inheritance Tree):** La profundidad de una clase dentro de la jerarquía de herencia es la longitud máxima desde el nodo de clase hasta la raíz del árbol y se mide por el número de clases antepasadas.
* **NOC (Number Of Children):** es el número de subclases inmediatas subordinadas a una clase en la jerarquía.
* **CBO (Coupling Between Object Classes):** es un recuento del número de otras clases a las que se acopla una clase.
* **LCOM (Lack of Cohesion of Methods):** mide el grado de similitud de los métodos por variables de entrada de datos o atributos (propiedades estructurales de las clases).

## Fase de construcción

La fase de construcción consiste en utilizar los modelos creados durante la etapa de análisis y diseño para crear los componentes del sistema. Los principales productos generados de esta fase son el código fuente del sistema, la respectiva documentación y los diferentes componentes construidos durante esta fase. Los estándares, convenciones, métricas y prácticas a emplear para asegurar la calidad de este proceso se listan y describen a continuación:

***Estándares***

* **Estándar ISO / IEC / IEEE 12207-2017 “Software life cycle processes”,** el cual se enfoca en el proceso del ciclo de vida del software, pero aplicado para esta fase en específico define las tareas a realizar, tales como la definición de las herramientas a emplear durante la construcción del software, definición de la codificación del programa (elegir el lenguaje de programación adecuado para el programa, la estructura de datos).
* **Estándar ISO/IEC 14598-2007 “Software product evaluation”,** es actualmente usado como base metodológica para la evaluación del producto software. Establece un marco de trabajo para evaluar la calidad de los productos de software proporcionando, además, métricas y requisitos para los procesos de evaluación de los mismos.
* **Estándar ISO/IEC 25000-2005 “System and Software Quality Requirements and Evaluation”,** proporciona una guía para el uso de las series de estándares internacionales llamados requisitos y Evaluación de Calidad de Productos Software (SQuaRE). Establece criterios para la especificación de requisitos de calidad de productos software, sus métricas y su evaluación, e incluye un modelo de calidad para unificar las definiciones de calidad de los clientes con los atributos en el proceso de desarrollo.
* **Estándar IEEE 1028-2008 “Software Reviews and Audits”,** donde se especifican los elementos a inspeccionar (para este caso, el código y documentación relacionada al mismo) y los responsables de realizar las actividades de revisión y auditoría.
* **Estándar IEEE 1012-1986 “Standard for Software Verification and Validation Plans”,** es un estándar de proceso que aborda todos los procesos del ciclo de vida del sistema, software y hardware, incluidos los grupos de procesos acuerdo, habilitación de proyectos organizacionales, proyecto, técnico, implementación de software, soporte de software y reutilización de software.

***Prácticas***

* **Emplear módulos y librerías para optimizar y facilitar la construcción del software.** La disponibilidad de módulos y librerías testeadas y optimizadas permiten facilitar el proceso de construcción del software, lo cual se traduce en ahorrar tiempo en el desarrollo. Las IDE actuales disponen de librerías, por lo que pueden ser empleadas como herramientas para agilizar el desarrollo.
* **Estandarizar las reglas del desarrollo.** Al momento de comenzar con la construcción del software, es requerido definir la forma de trabajar: La forma de llamar y definir las funciones, las variables, el nombre de los ficheros, atributos; en un buen código se distinguen fácilmente estos elementos. Un código que no sigue alguna normalización resulta más complicado de mantener.
* **Comentar código.** Para facilitar las modificaciones y mantenimiento, se debe redactar sentencias simples y concisas. Cuanto más corta sea, menos errores se producirán y más fácil será localizarlos y solventarlos. Los comentarios deberán ser evitados dentro de métodos o funciones (los comentarios deben explicar el “qué” y el “cómo”).
* **Documentación del código construido.** Documentar durante la etapa de construcción permite facilitar el entendimiento del desarrollo y funcionamiento del software al personal del proyecto y futuras personas que trabajen en él. En la documentación no solamente será incluida información referente al código, sino también especificaciones del software y herramientas que se emplean durante la construcción.

***Convenciones***

* **Convenciones de código para el lenguaje de programación JAVA.**  Para el manejo del lenguaje de programación JAVA se requieren seguir convenciones, mismas que se encuentran especificadas en el documento anexo (ver Anexo D).
* **Convenciones de código para el lenguaje de programación Kotlin.**  Para el manejo del lenguaje de programación Kotlin se requieren seguir convenciones, mismas que se encuentran especificadas en el documento anexo (Ver Anexo E).

***Métricas***

* **Densidad de errores en el código.** Para obtener el valor de la densidad de errores dentro del código se requiere conocer el número de errores entre el número de miles de líneas de código. Aunque con esta métrica no se toma en consideración si dichos errores son graves o no, únicamente se obtiene la densidad de errores.
* **Número de errores en el código.** Una métrica importante para el cálculo de otras. Esta métrica puede obtenerse gracias a herramientas de entorno de desarrollo u otras de análisis de código (herramientas tales como Raygun y Sentry).
* **Densidad de error de código ponderado.** Métrica que requiere los errores de código ponderados detectados (de las fuentes de datos para esta métrica son las mismas que para números de errores detectados en el código) entre el número de miles de líneas de código. Esta métrica considera la ponderación de los errores detectados (si los errores son graves, tendrán mayor peso, y en caso contrario el valor será menor).

## Fase de integración y pruebas

El principal objetivo de esta etapa consiste en asegurar que los componentes individuales que integran al sistema o producto, cumplen con los requisitos de la especificación creada durante la etapa de diseño. Los estándares, convenciones, métricas y prácticas a emplear para asegurar la calidad de este proceso se listan y describen a continuación:

***Estándares***

Los estándares a usar en esta fase son:

* **Estándar IEEE 829-2008 “Standard for software Test Documentation”,** el objetivo del estándar es proporcionar un conjunto estandarizado de documentos para la documentación de las pruebas de software. Estos documentos se encuentran clasificados en 8 tipos que pueden usarse en las 3 etapas distintas de las pruebas de software:

1. Preparación de pruebas.
2. Ejecución de las pruebas.
3. Término de las pruebas.

Estos documentos son:

1. Plan de pruebas.
2. Especificación del diseño de pruebas.
3. Especificación de casos de prueba.
4. Procedimientos de prueba.
5. Reporte de transmisión de ítems de pruebas.
6. Log de pruebas.
7. Reporte de incidentes de pruebas.
8. Reporte de las pruebas.

* **Estándar ISO/IEC/IEEE 29119-2022 “Software testing”,** el cual facilita el uso de otros estándares al presentar conceptos y vocabulario en los que se basan estos estándares, además de proporcionar ejemplos de su aplicación en la práctica. De esta forma, es informativo y proporciona un punto de partida, un contexto y una guía para las otras partes. Este consta de 5 partes:

1. Definiciones y vocabulario
2. Proceso de prueba
3. Documentación de prueba
4. Técnicas de prueba
5. Pruebas dirigidas por palabras clave

* **Estándar IEEE 1028-2008 “Software Reviews and Audits”,** donde se especifican los elementos a inspeccionar (para este caso, la documentación de pruebas) y los responsables de realizar las actividades de revisión y auditoría.
* **Estándar IEEE 1012-1986 “Standard for Software Verification and Validation Plans”,** es un estándar de proceso que aborda todos los procesos del ciclo de vida del sistema, software y hardware, incluidos los grupos de procesos acuerdo, habilitación de proyectos organizacionales, proyecto, técnico, implementación de software, soporte de software y reutilización de software.

***Prácticas***

* **Las pruebas deben validar una única cosa.** Las buenas pruebas validan una cosa y solo una cosa, lo que significa que, por lo general, las pruebas deberán validar un solo caso de uso. El propósito con está práctica es diseñar pruebas más simples y comprensibles, mejorando de esta manera la capacidad de mantenimiento y la depuración. Las pruebas que validan más de una cosa pueden volverse fácilmente complejas y su mantenimiento requiere mucho tiempo.
* **En cada prueba realizar una única afirmación.** Para cada prueba recomienda, en lo más posible, requerir una afirmación; esto con el objetivo de centrarse en validar solo lo que se necesita para el caso de uso que se está probando.
* **Las pruebas deben poder ejecutarse en cualquier máquina, en cualquier orden, sin que se afecten entre sí.** Si es posible, las pruebas no deben depender de factores ambientales o del estado global o externo. Las pruebas que tienen estas dependencias son más difíciles de ejecutar y, por lo general, inestables, lo que las hace más difíciles de depurar y corregir, y terminan costando más tiempo del que ahorran

***Convenciones***

* **Convención para nombrar pruebas unitarias.** Para asignar nombre a las pruebas unitarias se deben seguir una serie de estrategias y convenciones, estas se encuentran especificadas en el documento “Convención para Nombres de Pruebas Unitarias” (ver Anexo F).

***Métricas***

* **Amplitud de las pruebas.** Proporciona una indicación de cuantos requisitos (del número total de ellos) se han probado.
* **Profundidad de las pruebas.** Es una medida del porcentaje de los caminos básicos independientes probados en relación al número total de estos caminos en el programa.
* **Madurez a las pruebas.** Será usado como indicador del buen desempeño del flujo de pruebas, lo cual comprende los casos de pruebas que han obtenido resultados satisfactorios (no únicamente enfocarse en la completitud de los casos de prueba).
* **Densidad de defectos.** La métrica ofrece una medida sobre la proporción de defectos con respecto a la cantidad de elementos de especificación. La métrica permite valorar la integridad y madurez del software analizado.

## 

## 

## Fase de cierre

Durante esta etapa se da soporte y mantenimiento al producto y/o sistema liberado, con lo cual se da seguimiento a los comentarios de usuarios y clientes. Los estándares, convenciones, métricas y prácticas a emplear para asegurar la calidad de este proceso se listan y describen a continuación:

***Estándares***

* **Estándar ISO / IEC / IEEE 12207-2017 “Software life cycle processes”,**  el cual se enfoca en el proceso del ciclo de vida del software, pero aplicado para esta fase en específico define las tareas a realizar, tales como las modificaciones después de la entrega para corregir errores, mejorar el rendimiento u otros atributos y tareas para adaptar al sistema a un nuevo entorno modificado.
* **Estándar ISO 14764-2006 “Software Life Cycle Processes: maintenance”,** presenta los requerimientos para el proceso de mantenimiento del software, contiene las actividades y tareas del mantenedor, proporciona una guía que explica cómo llevar a cabo el proceso de mantenimiento y establece definiciones para los distintos tipos de mantenimiento. La guía es aplicable a la planificación, ejecución y control, mantenimiento, revisión y evaluación del proceso de mantenimiento.
* **Estándar IEEE 1219-1998 “Standard for Software Maintenance”,** es el único estándar que íntegramente se ocupa del proceso de mantenimiento del software. Describe un proceso iterativo para la gestión y ejecución de las actividades del proceso. Aunque sólo menciona las fases de desarrollo y de producción de un producto de software, éstas cubren todo su ciclo de vida, cualquiera que sea su tamaño o complejidad.
* **Estándar IEEE 1028-2008 “Software Reviews and Audits”,** donde se especifican los elementos a inspeccionar (para este caso, la documentación de liberación y mantenimiento) y los responsables de realizar las actividades de revisión y auditoría.
* **Estándar IEEE 1012-1986 “Standard for Software Verification and Validation Plans”,** es un estándar de proceso que aborda todos los procesos del ciclo de vida del sistema, software y hardware, incluidos los grupos de procesos acuerdo, habilitación de proyectos organizacionales, proyecto, técnico, implementación de software, soporte de software y reutilización de software.

***Prácticas***

* **Actualizaciones de mantenimiento.** Al aplicar las actualizaciones de mantenimiento de software, empleando un enfoque científico, es decir, se debe realizar una prueba de “control” sin aplicar ninguna actualización de mantenimiento. Después de eso, aplicar una única actualización de mantenimiento y volver a ejecutar el conjunto de pruebas.
* **Supervisar las nuevas versiones de mantenimiento de software.** Es importante prestar atención a las fechas de finalización del mantenimiento de las versiones instaladas en la pila de programas informáticos. Las versiones que ya no reciben ninguna actualización de mantenimiento deberían poner en marcha un proyecto de actualización para mantener a los usuarios al día.
* **Empleo de herramientas para el análisis de datos.** Con la ayuda de un software se realizará la identificación de fallos, historial de las revisiones y el comportamiento de los equipos como aires acondicionados, calefacción, extintores, etc. Esto es indispensable para llevar un seguimiento del trabajo que se realiza. Con Protecnus es posible gestionar y controlar en tiempo real la información registrada por los dispositivos móviles de los técnicos.

***Convenciones***

* No hay convenciones

***Métricas***

* ***Planned maintenance percentage (PPC):*** Esta métrica representa el porcentaje de tiempo dedicado a las actividades de mantenimiento planificadas frente a las no planificadas.
* ***Overall Equipment Effectiveness (OEE):*** OEE es la medida de la productividad de un equipo. Brinda datos informados sobre la eficacia de los procesos de mantenimiento de la organización en función de factores como la calidad, el rendimiento y la disponibilidad del equipo.
* ***Mean time to repair (MTTR):*** MTTR es la medida de la mantenibilidad de los elementos reparables. El reloj MTTR comienza a funcionar cuando comienzan las reparaciones y continúa hasta que se restablecen las operaciones. Esto incluye el tiempo de reparación, el período de prueba y el regreso a la condición de funcionamiento normal.
* ***Mean time between failure (MTBF):*** MTBF es la medida del tiempo previsto entre una avería y la siguiente durante el funcionamiento normal. En esencia, MTBF indica la vida útil esperada de un equipo específico. Un MTBF más alto significa que la pieza (o producto) que compró funcionará por más tiempo antes de que falle.
* ***Preventive maintenance compliance (PMC):*** El cumplimiento de PM se define como el porcentaje del trabajo preventivo programado y completado en un tiempo determinado.

# Revisiones y auditorías

## Objetivo

Definir las revisiones y auditorías técnicas y de gestión que se realizarán; asimismo, se especifica el cómo serán llevadas a cabo dichas revisiones y auditorías. Las revisiones y auditorías de los resultados del desarrollo se realizan a medida que se terminan cada una de las fases del ciclo de vida de desarrollo de software, con el fin de:

* Conocer el progreso alcanzado en el desarrollo.
* Evaluar el ajuste a los requerimientos del sistema.
* Evaluar la eficiencia en el trabajo

## Revisión de Especificación de requisitos de software (RERS)

Se aplicará una “Revisión de Diseño Formal” para verificar la claridad de la redacción de la especificación de requerimientos y su consistencia con la Descripción del producto y con el estándar requerido. Adicionalmente, se revisará que los requerimientos sean completos y no ambiguos o contradictorios. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Diseño Formal, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del documento de análisis y diseño (RDAD)

Se aplicará una “Revisión de Diseño Formal” para verificar claridad de la documentación del Análisis y Diseño, su factibilidad y la consistencia con el estándar de documentación requerido en el Proceso Específico. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Diseño Formal, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión de la configuración de software (RCS)

Se aplicará una “Revisión de Diseño Formal” para verificar la claridad de la redacción del documento de configuración de software y (introducir descripción de qué se hará en la revisión). Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Diseño Formal, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del manual de usuario (RMU)

Se aplicará una revisión por pares de tipo “Inspección” para verificar consistencia del Manual de Usuario con la Especificación de Requerimientos y con el estándar de documentación requerido en el Proceso Específico. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Inspección, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del manual de operación (RMO)

Se aplicará una “Revisión de Diseño Formal” para verificar consistencia del Manual de Operación con el Software y con el estándar de documentación requerido en el Proceso Específico. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Diseño Formal, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del manual de mantenimiento (RMM)

Se aplicará una revisión por pares de tipo “Inspección” para verificar consistencia del Manual de Mantenimiento con la Configuración de Software y con el estándar de documentación requerido en el Proceso Específico. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Inspección, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del registro de rastreo (RRR)

Se aplicará una revisión por pares de tipo “Inspección” para verificar que el Registro de Rastreo contenga las relaciones adecuadas entre los requerimientos y los elementos de Análisis y Diseño. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Inspección, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del plan de pruebas de sistema (RPPS)

Se aplicará una revisión por pares de tipo “Inspección” para verificar consistencia del Plan de Pruebas de Sistema con la Especificación de Requerimientos y con el estándar de documentación requerido en el Proceso Espeífico. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Inspección, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

## Revisión del plan de pruebas de integración (RPPI)

Se aplicará una revisión por pares de tipo “Inspección” para verificar consistencia del Plan de Pruebas de Integración con el Análisis y Diseño y con el estándar de documentación requerido en el Proceso Específico. Los defectos encontrados se documentan en un Reporte de Verificación.

La descripción del proceso para la ejecución de la revisión será definida en el Plan de Revisión de Inspección, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe cómo llevar a cabo una revisión y los requisitos mínimos aceptables para las revisiones sistemáticas de software.

Las revisiones se realizaron con base a las actividades mencionadas anteriormente, están ordenadas de tal manera:

| **Verificación** | **Actividad** | **Producto** |
| --- | --- | --- |
| Ver1 | A1.1 | Especificación de requerimientos |
| Ver2 | A1.3 | Plan de pruebas de sistema |
| Ver3 | A1.4 | Manual de usuario |
| Ver4 | A2.1 | Documento de análisis y diseño y Registro de rastreo |
| Ver5 | A2.3 | Plan de pruebas de integración |
| Ver6 | A3.1 | Registro de rastreo |
| Ver7 | A4.1 | Manual de operación |
| Ver8 | A4.2 | Manual de usuario |
| Ver9 | A5.1 | Manual de mantenimiento |

*Tabla 3. Tabla de relación de verificaciones y actividades y su orden*

Para el caso de auditorías a realizar, se toman a consideración las siguientes:

| **Auditoría** | **Descripción** |
| --- | --- |
| Auditoría funcional | Esta auditoría se realiza previa a la liberación del software, para verificar que todos los requerimientos especificados en el documento de requerimientos fueron cumplidos. |
| Auditoría física | Esta revisión se realiza para verificar que el software y la documentación son consistentes y están aptos para la liberación. |
| Auditorías internas al proceso | Estas auditorías son para verificar la consistencia: del código versus el documento de diseño, especificaciones de interfaz, implementaciones de diseño versus requerimientos funcionales, requerimientos funcionales versus descripciones de testeo. |

*Tabla 4. Tabla de las auditorías a realizar dentro del Plan SQA*

## Auditoría Funcional

Se aplicará una Auditoría Funcional antes de la liberación del software, con el objetivo de verificar que las funcionalidades que han sido probadas de un componente de configuración han cubierto los requisitos mencionados en el documento de Especificación de Requisitos del Software, es decir, verificar que el producto y la documentación cumplen con los requisitos de funcionalidad, desempeño y calidad. Asimismo, se verificará la trazabilidad de los requisitos del documento ERS y los requisitos cubiertos por el sistema.

La descripción del proceso para la ejecución de la auditoría será definida en el Plan de Auditorías, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe la principal información que debe contener, tal como el propósito y alcance de la auditoría, antecedentes de la organización auditada, los productos de software a auditar y criterios a considerar en la verificación.

## Auditoría Física

Se aplicará una Auditoría Física, con la cual se verificará si los elementos de configuración integrados cumplen con la especificación técnica que los define. Esto implica que todos los componentes requeridos están presentes, que corresponden con la versión correcta y con la información contenida en los reportes. La auditoría tiene el propósito de verificar si el software y la documentación están listos para su liberación.

La descripción del proceso para la ejecución de la auditoría será definida en el Plan de Auditorías, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe la principal información que debe contener, tal como el propósito y alcance de la auditoría, antecedentes de la organización auditada, los productos de software a auditar y criterios a considerar en la verificación.

## Auditorías internas al proceso

Se aplicarán auditorías internas al proceso para verificar la consistencia del código con el documento de diseño, especificaciones de interfaz, implementaciones de diseño con los requisitos funcionales y los requisitos funcionales con las descripciones de testeo.

La descripción del proceso para la ejecución de la auditoría será definida en el Plan de Auditorías, que será elaborado con base en el estándar IEEE 1028-2008 “Software Reviews and Audits” que describe la principal información que debe contener, tal como el propósito y alcance de la auditoría, antecedentes de la organización auditada, los productos de software a auditar y criterios a considerar en la verificación.

# Pruebas

Para la construcción del documento de pruebas se usará el estándar ISO/IEC/IEEE 29119-3 - 2013 “Software Testing Standard” que proporciona una plantilla para la construcción de la documentación de pruebas.

Las descripción de los procesos para las pruebas de integración estarán definidas y descritas en el documento “Plan de pruebas de integración”. Para su posterior verificación/validación, los procedimientos estarán en el documento “Plan de verificación y validación”.

Para las pruebas de aceptación, no es necesario ningún plan, puesto que son realizadas por los usuarios finales, quienes validaron que el producto entregado cumpla con sus expectativas.

Tipos de pruebas:

* Pruebas de integración: Pruebas realizadas para la adición de módulos/componentes de software a un subsistema más complejo.
* Pruebas de aceptación: Pruebas realizadas por el cliente/usuario final con el objetivo de validar el funcionamiento y satisfacción.
* Pruebas de caja negra: Pruebas realizadas en la que se prueba la funcionalidad de una aplicación ignorando la parte interna de dicha aplicación.

Todo personal que desarrolle algún componente del proyecto realizará las siguientes actividades de pruebas necesarias del software y estas son responsables de:

* Realizar las pruebas Unitarias.
* Realizar las pruebas de acuerdo al plan de pruebas del software.
* Anotar los resultados de las pruebas.
* Recomendar acciones correctivas si se encontraron defectos en los módulos probados.

# Informe de problemas y acción correctiva

En esta sección se describe el reporte y control del sistema utilizado por el personal de calidad para registrar y analizar las discrepancias encontradas, así como para monitorear la implementación de las acciones correctivas. Los formatos utilizados para realizar los reportes se describen más adelante en esta sección.

## Reporte del proceso de auditorías

El personal de calidad reportará el resultado de las auditorías y las recomendaciones proporcionadas mediante el formato “Reporte de auditorías” (Ver Anexo G). Este reporte se usa para asegurarse que el proceso:

* Se está siguiendo de manera correcta y se está trabajando de forma efectiva
* Se está siguiendo, pero no se está trabajando de manera efectiva
* No se está siguiendo.

## Presentación del reporte del proceso auditoría

El proceso de Reporte de Auditorías está dirigido hacia el Administrador del Proyecto el cual utilizará los reportes de las siguientes maneras:

* Para saber si los procesos de desarrollo son acatados y si son efectivos para el cumplimiento de las metas del proyecto. Cuando sea necesario el administrador del proyecto puede iniciar cambios a los procesos, mediante los procedimientos establecidos, para que los procesos queden estables.
* Para indicar el acuerdo, desacuerdo, o el aplazamiento de las recomendaciones hechas en el Proceso de Reporte de Auditoría. En caso de que el administrador del proyecto indique desacuerdo con las recomendaciones registradas en el proceso de reporte de auditoría, la disposición final de recomendaciones del informe se hace por el patrocinador del proyecto.

## Procedimiento de escalamiento para la Resolución de no ocurrencia en el Proceso de Reporte de Auditoría

Al encontrarse un problema de calidad en algún elemento de trabajo ya sea documento, código o producto de software se tendrá que realizar lo siguiente:

* Primero se tratará con el creador de ese elemento:

1. Si existen problemas de desacuerdos en la resolución del problema, el personal de calidad tendrá que notificar al administrador del proyecto para que este tome cartas en el asunto y de una solución al problema;

2. Si el administrador del proyecto no da una solución, se escalará el problema al patrocinador del proyecto y él tomará la decisión final.

El personal de calidad conservará el registro original de las conclusiones tomadas y la resolución posterior en sus expedientes de auditoría.

Para el caso de las herramientas de software, se tomará la plantilla para el formato de Reporte de Evaluación de Herramientas de Software (ver Anexo H).

# Técnicas, herramientas y metodologías

## Herramientas

Para esta sección se incluyen utilidades del sistema operativo, Debuggeadores, documentos de ayuda, checklist , analizadores de estructuras, analizadores de código, auditorías de estándares, monitoreo de rendimiento, software de desarrollo, matrices de seguimiento de software, pruebas de generadores de casos. A continuación, se mencionan las herramientas a ser empleadas y las condiciones en las cuales serán utilizadas:

* Lenguajes de programación: Java, Kotlin, Swift, XML.
* Herramientas para la elaboración de diagramas UML: Jude, DB Designer, Star UML.
* Herramientas para el trabajo de bases de datos: MYSQLite.
* Herramientas para elaborar Casos de Uso: Jude, Star UML.
* Herramientas para procesamiento de texto: Microsoft Word.
* Herramientas de apoyo: Internet, Excel, Photoshop, Figma, Github.
* Herramientas de desarrollo: Android Studio, Visual Studio Code.

## Técnicas

En las técnicas se incluirá la revisión de uso de estándares, inspecciones de software, rastreo de requerimientos, verificación y validación de diseño y requerimientos, mediciones y evaluaciones de fiabilidad, análisis de lógica de negocio.

* Estándares: Codificación de Lenguajes, UML, Diseño de BD Lógicas y Físicas
* Programación en pares.

Para las técnicas que ayudan a la evaluación de la calidad son:

* ANSI / IEEE – STD 830 Guide for Software Requirements Specifications
* ANSI / IEEE – STD 1016 Recommended Practice for Software Design Descriptions
* ANSI / IEEE – STD 1008 Standard for Software Unit Testing
* ANSI / IEEE – STD 1063 Standard for Software User Documentation
* ANSI / IEEE – STD 1028 Standard for Software Reviews and Audits

## Metodologías

Las metodologías de Aseguramiento de Calidad serán conjuntos integrados de técnicas, de entre los anteriores. Como principal metodología se emplearán:

* Programación orientada a objetos
* Programación orientada a eventos.

# Control del código

Los objetos de configuración (SCI) que se controlarán respecto a la sección de control de código será:

* Código de software
  + Código fuente.
  + Objetos de código.
  + Java Archive(Jar).
  + Prototipos de software
* Archivos de información
  + Scripts de pruebas

Para las tareas de la gestión de configuración del software, se usará el grupo de cambio de software de control, para ello se utilizará una plantilla propia para la gestión de los cambios del versionamiento del software (Ver Anexo I).

***Medios físicos de almacenamiento y mantenimiento.***

Se tendrá un control de las ubicaciones físicas de las versiones del código, para ello se necesitará de medios de almacenamiento, esto será completado gracias a discos duros (HDD) o discos de estados solido (SSD) dedicados al apartado de versiones, en estos se almacenan los respaldos de las versiones pasadas, las copias de respaldo de versiones se realizan cada 72 horas, posteriormente se eliminan las versiones antiguas de los equipos de trabajo.

***Procedimientos para la distribución de una copia.***

Se deberá haber proporcionado una cuenta de Github al personal competente del control del repositorio para tener acceso a los objetos de configuración permitidas por los encargados competentes de la seguridad del proyecto.

Se solicitará autorización al líder de equipo de desarrollo o encargados de las versiones de código para el acceso a una versión específica del proyecto.

***Procedimientos para implementar una nueva versión.***

Para la implementación de nuevos cambios, el procedimiento será:

1. Generar una petición de cambio del software, siguiendo la plantilla proporcionada (Ver anexo J).
2. Realizar estimaciones del esfuerzo, costos, tiempos que implica el cambio en el estado actual del proyecto.
3. Revisar los documentos que se verán afectados y llevar un control aproximado de los cambios.
4. Realizar valoraciones para la implementación.
5. Autorizar o rechazar el cambio.
6. Realizar los cambios previstos.

***Aseguramiento del proceso***

Para asegurar el cumplimiento del proceso de almacenamiento, se llevarán a cabo revisiones cada inicio de semana, esto con el fin de revisar que el control de acceso a los documentos de código, documentos, y versionamiento sean adecuados.

Para asegurar el cumplimento del proceso de solicitud de cambios, la entrega del formato de solicitud de cambio deberá ser hecho al inicio de cada semana, esto con el fin de poder resolver cualquier incongruencia, falta de información, o error en el formato. Con esto se asegura que exista suficiente tiempo para verificar la calidad del contenido del formato y contar con todos los recursos necesarios para realizar el proceso de cambio.

# Control de medios

La organización responsable por esta tarea es la organización de desarrollo, con la supervisión de la organización de la SQA. Se debería asegurar que:

* Está garantizado el almacenamiento y recuperación de software.
* El software está accesible únicamente para aquellos que lo necesitan.
* Se controla el entorno para que no se degrade el medio físico en el que se almacena el software.
* Se almacenan copias del software crítico y del código en línea base fuera de las instalaciones de la organización.

## Medio de almacenamiento

El medio del programa de computadora se define como aquellos medios sobre los cuales los datos son almacenados. Se utilizarán los siguientes medios:

* Los discos duros como dispositivos primarios.
* Los CD ́s y dispositivos USBs como almacenamiento secundario, para guardar las copias de seguridad.
* La documentación respectiva sobre el desarrollo de software (papel).
* Github como medio de respaldo para el código fuente y documentación.

## Proceso de copias de seguridad

Las copias de seguridad serán realizadas a la finalización de cada sesión de trabajo, registrándose la fecha y hora de copia de seguridad.

## Puntos de control

Para el acceso no autorizado, se debe asignar cuentas privilegiadas, cada usuario que interactúa con el software tendrá su propia cuenta de acuerdo al cargo que desempeñe y para el manejo del programa se debe contar con una cuenta de correo electrónico en Github. Se cuenta con la integridad de la Base de Datos para la protección de los datos. Se realizará una revisión periódica del software con el fin de que funcione óptimamente

## Desgaste de los medios

El uso de medios electrónicos como medios USB, discos duros (electromecánicos), o CD son propensos a daños físicos que afectan el rendimiento y la integridad de los datos resguardados. Para resguardar los medios de posibles daños o desgastes por el tiempo, el uso adecuado de una gestión de configuración, provee de una minimización de los efectos adversos. El resguardo de lugares sin electricidad estática, o propensos a generar polvo otorgan un grado de seguridad al mantener este tipo de dispositivos, lugares seguros contra fuegos, y sin humedad. Adicional a estas medidas, es recomendable la asignación de fechas específicas para revisiones programadas con el fin de asegurar el lugar de resguardo de los medios, y verificar el estado de los dispositivos usados.

## Vulneración de los medios

El uso de herramientas como Github, como respaldo en la nube conlleva a un reto adicional contra el acceso no autorizado al repositorio, por lo que las medidas de seguridad que han de emplearse han de consistir en la validación de cada usuario. La asignación de un depósito privado, asignación de cuentas empresariales con grados de acceso para el uso exclusivo del repositorio, las conexiones a través de llaves ssh son algunas recomendaciones para mantener la integridad y confidencialidad de los datos respaldados.

# Recopilación, mantenimiento y retención de registros

Los documentos que son requeridos son los siguientes:

* Plan de garantía de calidad del software.
* Especificación de requerimientos del software.
* Descripción del diseño del software.
* Plan de verificación y validación del software.
* Documentación del usuario.

Para el mantenimiento de los registros del software, será realizado por versiones sucesivas de actualizaciones de las mismas, para esto se lleva un registro de actualizaciones de la documentación. Los documentos verificados y validados, deben ser documentados en libros impresos, con tres copias de cada documento y almacenado en lugares diferentes y ambiente adecuados como Github y en disco duros. La retención de registros se realizará en cada finalización de las fases del ciclo de vida de desarrollo de software y según los puntos de verificación y validación.

# Procedimiento de control de la documentación

Se aplicará el ISO 9000:2015 para el control de documentación, el cual consiste en asegurar que la información documentada se encuentre disponible para su uso, dónde y cuándo se necesite, así como proteger la confidencialidad de la misma.

Los documentos se encontrarán alojados en un repositorio privado en GitHub al cual estará disponible las 24 horas del día los 7 días de la semana (exceptuando el mantenimiento del servicio o caídas del mismo) únicamente para el personal con la autorización necesaria.

De manera complementaria los documentos serán almacenados en discos duros a manera de respaldo de emergencia, en caso de que haya alguna alteración o pérdida del acceso a los documentos subidos en GitHub.

El formato para el control de documentación se puede consultar en el Anexo B.

# Entrenamiento

Existen diferentes disciplinas bajo las cuales los miembros del equipo de desarrollo de software tienen opción para formarse. En esta sección a través de la Tabla 11, se indican los conocimientos o habilidades deseables para realizar efectivamente las tareas orientadas a aseguramiento de calidad.

Es recomendable que las personas designadas a la unidad de aseguramiento de calidad tengan conocimiento de las herramientas utilizadas por el grupo de desarrolladores y preferentemente que hayan participado en el desarrollo de al menos un proyecto.

| **Tarea** | **Habilidades Requeridas** | **Tipo de Revisión** |
| --- | --- | --- |
| Revisión de código | * Lenguaje de programación aplicado * Marcos de trabajo aplicados | Revisión por pares |
| Revisión de documentación | * Modelo de calidad de referencia * Metodología de desarrollo de software * Guías institucionales de elaboración | Revisión técnica formal  Inspecciones |
| Auditoría del Proceso de Desarrollo | * Proceso de desarrollo de software aplicado * Estándares tomados como referencia | Auditoría del proceso realizado en sus diferentes fases |
| Pruebas de Software | * Niveles y técnicas de pruebas * Estrategia de pruebas * Herramientas de soporte a la realización de pruebas | Auditoría del proceso de pruebas |
| Administración | * Administración de proyectos * Técnica de valor ganado * Recopilación y análisis de métricas | Auditoría del proceso administrativo |
| Control de Cambios | * Administración de la configuración (registro, control de cambios, seguimiento y atención) | Auditoría al proceso de control de cambios |
| Control de Riesgos | * Administración de riesgos * Auditoría al proceso (identificación, análisis, control de riesgos, atención) | Auditoría al proceso de control de riesgos |

*Tabla 5. Habilidades requeridas para realizar las revisiones*

El plan de entrenamiento aplicado a las personas que formarán parte de la unidad de aseguramiento de calidad, como se mencionó previamente, debe iniciar con los desarrolladores. La transmisión de conocimiento hacia el grupo de desarrolladores pretende que realicen las diferentes fases del proceso de construcción de software de forma sistemática y alineada a un modelo de calidad.

Cada desarrollador júnior trabajará en conjunto con un desarrollador sénior quien será el responsable de conducir su entrenamiento de acuerdo a las actividades que le sean asignadas, cabe aclarar que no se aplica un plan de entrenamiento general, más bien se prepara al personal de acuerdo al objetivo que debe alcanzar, de tal forma que si su asignación es la generación del documento de especificación de requerimientos, conocerá el estándar o estándares de referencia y los lineamientos que debe seguir para generarlo.

Una vez que un desarrollador adquiere experiencia y consolida sus conocimientos su participación en el grupo de aseguramiento de calidad debe ser más efectiva.

# Anexos

## Anexo A

Definición de la plantilla para el Registro de Rastreo.

**Registro de rastreo**

*[Nombre del proyecto]*

Registro de rastreo

*Versión [1.0]*

*Este documento es la plantilla base para elaborar el documento Registro de Rastreo.*

*Los textos que aparecen entre paréntesis rectos son explicaciones de qué debe contener cada sección. Dichos textos se deben seleccionar y sustituir por el contenido que corresponda. Para actualizar la tabla de Contenidos, haga clic con el botón derecho del ratón sobre cualquier línea del contenido de la misma y seleccione Actualizar campos, en el cuadro que aparece seleccione Actualizar toda la tabla y haga clic en el botón Aceptar.*

*LAS SECCIONES PARA LAS QUE NO TENGA DATOS NO DEBE BORRARLAS; EN DICHOS CASOS MÁRQUESE CON “N/A” O “SIN DATOS”]*

Historial de revisiones

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| [dd/mm/aaaa] | [x.x] | [detalles] | [nombre] |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Contenido**

1. Introducción.

1.1. Propósito.

1.2. Generalidades.

1.3. Objetivos.

1.4. Referencias.

2. Matrices de trazabilidad.

2.1. Especificación de Requerimientos X Modelo de Casos de Uso.

2.2. Modelo de Casos de Uso X Modelo de Análisis.

2.3. Modelo de Casos de Uso X Modelo de Diseño.

2.4. Modelo de Diseño X Modelo de Implementación.

2.5. Modelo de Implementación X Plan de Verificación.

2.5.1. Componentes del modelo de implementación X Casos de prueba unitarias.

2.5.2. Subsistemas del modelo de implementación X Casos de prueba de integración.

2.5.3. Requerimientos del sistema X Casos de pruebas del sistema.

1. Introducción

1.1. Propósito

*El propósito de este documento es obtener un* ***registro de rastreo*** *de todos los productos (internos, externos, intermedios y finales) del proceso de desarrollo de software.*

1.2. Generalidades

*Consiste en una representación gráfica de las relaciones entre dos o más productos del proceso de desarrollo, generalmente identificadas en las intersecciones de líneas verticales y horizontales.*

*Por ejemplo, para representar la relación entre los requisitos y el diseño de un componente del software.*

1.3. Objetivos

·  *Seguimiento de la trazabilidad entre los elementos de diferentes niveles de abstracción.*

*· Poder medir el impacto que el proyecto tendrá al momento de realizar un cambio.*

*· Facilitar el manejo de cambios, testing y mantenimiento del sistema.*

1.4. Referencias

*Para obtener en detalle la trazabilidad entre los modelos de Análisis, Diseño, Implementación y Distribución, consulte los documentos Descripción de Arquitectura [(DSARQGXvY.doc)] y Modelo de Implementación [(IMOOMIMGXvy.DOC)].*

*A diferencia de dichos documentos, con el presente se pretende tener un registro “liviano”, mas ágil, accesible para consultar y mas fácil de mantener.*

2. Matrices de trazabilidad

2.1. Especificación de Requerimientos X Modelo de Casos de Uso

*[Matriz que muestra la correspondencia entre los requerimientos funcionales*

*de la Especificación de Requerimientos y los casos de uso del Modelo de*

*Casos de Uso.*

*La numeración de los requerimientos como de los casos de uso deben coincidir con la de la Especificación de Requerimientos y Modelo de Casos de Uso respectivamente.*

*Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]*

***Ejemplo****: El CU01 satisface al REQF1*

| **REQ / CU** | CU01 | CU02 | CU03 | CU04 | ….. |
| --- | --- | --- | --- | --- | --- |
| REQF1 | X |  |  |  |  |
| REQF2 |  | X | X |  |  |
| REQF3 |  |  |  | X |  |
| REQF4 |  |  |  |  |  |
| REQF4 |  |  |  |  |  |
| ….. |  |  |  |  |  |

2.2. Modelo de Casos de Uso X Modelo de Análisis

*[****OPCIONAL****: solo si realizaron el Modelo de Análisis.*

*Matriz que muestra la correspondencia entre Casos de uso del Modelo de*

*Casos de Uso y clases y paquetes especificados en el Modelo de Análisis.*

*La numeración de los casos de uso como la de las clases y paquetes deben coincidir con la del Modelo de Casos de Uso y Modelo de Análisis respectivamente.*

*Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]*

| **CU / Clase-PAQ** | Clase1 | Clase2 | … | PAQ1 | ….. |
| --- | --- | --- | --- | --- | --- |
| CU01 | X |  |  |  |  |
| CU02 |  | X | X |  |  |
| CU03 |  |  |  | X |  |
| CU04 |  |  |  |  |  |
| CU05 |  |  |  |  |  |
| ….. |  |  |  |  |  |

2.3. Modelo de Casos de Uso X Modelo de Diseño

*[Matriz que muestra la correspondencia entre Casos de Uso del Modelo de*

*Casos de Uso y clases y subsistemas especificados en el Modelo de Diseño.*

*La numeración de los casos de uso como la de las clases y subsistemas debe coincidir con la del Modelo de Casos de Uso y Modelo de Diseño respectivamente.*

*Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]*

| **CU / Clase-SubSist** | Clase1 | Clase2 | … | SubSist1 | ….. |
| --- | --- | --- | --- | --- | --- |
| CU01 | X |  |  |  |  |
| CU02 |  | X | X |  |  |
| CU03 |  |  |  | X |  |
| CU04 |  |  |  |  |  |
| CU05 |  |  |  |  |  |
| ….. |  |  |  |  |  |

2.4. Modelo de Diseño X Modelo de Implementación

*[Matriz que muestra la correspondencia entre clases y subsistemas en el*

*Modelo de Diseño y componentes, subsistemas e interfaces en el Modelo de*

*Implementación.*

*La numeración de las clases y subsistemas como la de las clases, subsistemas e interfaces deben coincidir con la del Modelo de Diseño y Modelo de Implementación respectivamente.*

*Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]*

| **Clase-SubSist / Comp-SubSist-Interf** | Comp1 | Comp2 | SubSist1 | Interf1 | ….. |
| --- | --- | --- | --- | --- | --- |
| Clase1 | X |  |  |  |  |
| Clase2 |  | X | X |  |  |
| .. |  |  |  | X |  |
| SubSist1 |  |  |  |  |  |
| SubSist2 |  |  |  |  |  |
| ….. |  |  |  |  |  |

2.5. Modelo de Implementación X Plan de Verificación

2.5.1. Componentes del modelo de implementación X Casos de prueba unitarias

*[Matriz que muestra la correspondencia entre los componentes del Modelo de Implementación con los casos de prueba unitaria del Plan de Verificación.*

*La numeración de los componentes como la de los casos de prueba unitaria, deben coincidir con la del Modelo de Implementación y el Plan de Verificación respectivamente.*

*Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]*

| **CompImp / CPU** | CPU1 | CPU2 | CPU3 | CPU4 | ….. |
| --- | --- | --- | --- | --- | --- |
| CompImp1 | X |  |  |  |  |
| CompImp2 |  | X | X |  |  |
| CompImp3 |  |  |  | X |  |
| CompImp4 |  |  |  |  |  |
| CompImp5 |  |  |  |  |  |
| ….. |  |  |  |  |  |

2.5.2. Subsistemas del modelo de implementación X Casos de prueba de integración

[Matriz que muestra la correspondencia entre los subsistemas del Modelo de Implementación con los casos de prueba de integración del Plan de Verificación.

La numeración de los subsistemas como la de los casos de prueba de integración, deben coincidir con la del Modelo de Implementación y el Plan de Verificación respectivamente.

Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]

| **REQ / CPI** | CPI1 | CPI2 | CPI3 | CPI4 | ….. |
| --- | --- | --- | --- | --- | --- |
| SubSist1 | X |  |  |  |  |
| SubSist2 |  | X | X |  |  |
| SubSist3 |  |  |  | X |  |
| SubSist4 |  |  |  |  |  |
| SubSist5 |  |  |  |  |  |
| ….. |  |  |  |  |  |

2.5.3. Requerimientos del sistema X Casos de pruebas del sistema.

[Matriz que muestra la correspondencia entre los requerimientos del sistema con los casos de prueba del sistema Plan de Verificación]

La numeración de los requerimientos como la de los casos de prueba del sistema, deben coincidir con la del Modelo de Implementación y el Plan de Verificación respectivamente.

Si la matriz es muy grande, se puede realizar en Excel y en lugar de ella dejar un link al nuevo archivo.]

| **REQ / CPS** | CPS1 | CPS1 | CPS1 | CPS1 | ….. |
| --- | --- | --- | --- | --- | --- |
| REQF1 | X |  |  |  |  |
| REQF2 |  | X | X |  |  |
| REQF3 |  |  |  | X |  |
| ……. |  |  |  |  |  |
| REQNF1 |  |  |  |  |  |
| REQNF1 |  |  |  |  |  |
| …….. |  |  |  |  |  |

## Anexo B

Plantilla para el control de documentación.

**Autores**

**Fecha de finalización**

**Persona(s) que aprobaron el documento**

**Fecha de aprobación**

**Firma(s) del auto(es) y persona(s) que lo aprobaron**

**Descripciones de los cambios introducidos en la nueva versión**

**Lista de versiones y revisiones anteriores**

## Anexo C

Plantilla para la verificación del documento de Especificación de Requisitos.

| **Estructura del documento de especificación de requisitos** | | | |
| --- | --- | --- | --- |
| **ID** | **Criterio** | **¿Lo cumple? (Sí/No)** | **Comentarios** |
| 1 | ¿Todos los requisitos están escritos en un nivel consistente y apropiado de detalle? |  |  |
| 2 | ¿Falta alguna sección dentro del documento acorde a lo especificado por la plantilla del estándar 830 IEEE? |  |  |
| 3 | ¿Están todas las figuras, tablas y diagramas etiquetados y referenciados? |  |  |
| 4 | ¿Se clasifican los diferentes tipos de requisitos? |  |  |
| 5 | ¿Están definidos todos los términos y unidades de medida? |  |  |
| **Completitud y exactitud del documento** | | | |
| 6 | ¿Se han especificado todas las tareas que realizará el sistema? |  |  |
| 7 | ¿Los requisitos abarcan todas las necesidades conocidas del cliente o del sistema? |  |  |
| 8 | ¿Están definidas todas las interfaces de comunicación, software y hardware externo? |  |  |
| 9 | ¿Se incluye la prioridad de implementación de cada requisito? |  |  |
| 10 | ¿Está documentado el comportamiento esperado para todas las condiciones de error anticipadas? |  |  |
| 11 | ¿Se han definido las interfaces externas, como por ejemplo usuarios o hardware? |  |  |
| **Claridad y no ambigüedad** | | | |
| 12 | ¿Los requisitos se han especificado de forma clara para que si se entregan a un grupo independiente para la implementación, dicho grupo sea capaz de entenderlo? |  |  |
| 13 | ¿Los requisitos están especificados de forma concisa, de modo que evitan la posibilidad de hacer múltiples interpretaciones de ellos? |  |  |
| 14 | ¿Todos los requisitos evitan conflictos con otros requisitos? |  |  |
| 15 | ¿Cada requisito está escrito en un lenguaje coherente, claro y conciso? |  |  |
| **Trazabilidad y organización** | | | |
| 16 | ¿Son correctas todas las referencias cruzadas internas a otros requisitos? |  |  |
| 17 | ¿Cada requisito está identificado de manera única y correcta? |  |  |
| 18 | ¿Se puede rastrear cada requisito hasta su origen? |  |  |
| 19 | ¿Los requisitos están organizados de forma lógica y accesible? |  |  |
| **Atributos de calidad** | | | |
| 20 | ¿Se especifican correctamente todos los objetivos de usabilidad, rendimiento, seguridad y protección? |  |  |
| 21 | ¿Todos los requisitos de calidad son medibles? |  |  |

## Anexo D

Para consultar las convenciones de programación de JAVA, checar el documento en el repositorio “Anexo D Convenciones de programación JAVA”.

## Anexo E

Para consultar la guía de convenciones Kotlin, checar el documento en el repositorio “Anexo E Guía de convenciones Kotlin”.

## Anexo F

Para consultar las convenciones de nombres para pruebas unitarias, checar el documento en el repositorio “Anexo F Convenciones de nombres para pruebas unitarias”.

## Anexo G

Formato del reporte del proceso de auditorías.

| REPORTE DEL PROCESO DE AUDITORÍAS | |
| --- | --- |
| Identificador de seguimiento: | |
| Auditor: | Fecha de reporte: |
| Equipo: | Nombre del proyecto: |
| Fecha de la auditoría: | |
| Proceso/procedimiento auditado: | |
| Checklist de auditoría usado (agregar): | |
| Resultado de la auditoría: (Marque uno solo)  \_\_\_\_\_ Proceso/Procedimiento Aceptable  \_\_\_\_\_ Proceso/Procedimiento Condicionalmente aceptable  \_\_\_\_\_ Proceso/Procedimiento Inaceptable  (Sujeto a la finalización satisfactoria de los puntos mencionados arriba)  Notas de Condiciones:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  TITULO: ENCARGADO: FECHA DE VENCIMIENTO:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
|
| Acciones correctivas:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Disposición: (Marque el correspondiente)  \_\_\_\_ Aprobado  \_\_\_\_ Cancelado  \_\_\_\_ Aplazado | |
| Administrador del proyecto: Fecha:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Firma del encargado de SQA: Fecha: | |

## Anexo H

Formato de Reporte de Evaluación de Herramientas de Software.

| **EVALUACIÓN DE HERRAMIENTAS DE SOFTWARE** |
| --- |
| SQA: Fecha de evaluación: |
| Herramienta de software evaluada: |
| Métodos o criterios usados en esta evaluación: |
| Resultados de la evaluación: |
| Acciones correctivas recomendadas: |
| Acciones correctivas tomadas: |

## Anexo I

Tabla para el control de versiones y revisiones de SCI.

| **Fecha** | **Versión** | **Autor** |
| --- | --- | --- |
| dd/mm/yyyy | El formato de versión será A.B.C donde:   * A es valor de la versión base. * B es el valor de versión de parches. * C es el valor de versión de parches menores. |  |

| **Fecha** | **Revisión** | **Autor** |
| --- | --- | --- |
| dd/mm/aaaa | Revisión # |  |

## Anexo J

Plantilla de solicitud de cambios.

(1) Principios de cambio

* El iniciador
* La fecha de presentación del SCR
* El carácter del cambio
* Los objetivos
* La contribución esperada al proyecto/sistema
* La urgencia de la actuación

(2) Cambiar detalles

* Descripción del cambio propuesto
* Una lista de los SCI que se cambiarán
* Efecto esperado en otras LME
* Efecto esperado en las interfaces con otros sistemas de software y firmware de hardware
* Retrasos esperados en los cronogramas de finalización del desarrollo y perturbaciones esperadas en los servicios a los clientes

(3) Cambio de calendario y estimación de recursos

* Calendario de implementación
* Recursos profesionales requeridos estimados
* Otros recursos necesarios
* Costo total estimado del cambio solicitado