Benemérita Universidad Autónoma de Puebla

Ingeniería en Tecnologías de la Información

Control de Calidad de Software

Software Quality Assurance Plan

Otoño 2021

Contenido

[Propósito 3](#_Toc178097731)

[Gestión 3](#_Toc178097732)

[Organización 3](#_Toc178097733)

[Herramientas y tecnologías 3](#_Toc178097734)

[Métricas de Calidad 4](#_Toc178097735)

[Actividades 4](#_Toc178097736)

# Propósito

Este documento establece un plan de aseguramiento de la calidad con el objetivo de garantizar que el proyecto se entregue cumpliendo con los atributos de calidad identificados: usabilidad, funcionalidad y eficiencia. A través de este plan, se busca fomentar la colaboración entre todos los miembros del equipo, establecer canales de comunicación efectivos con las partes interesadas y definir procesos para la identificación y resolución de problemas. El objetivo final es asegurar que el producto de software cumpla con los requisitos funcionales y no funcionales establecidos en la documentación del proyecto.(para que, cuando se maneja, que proyecto)

# Referencias

(para los efectos de tal cosa vease ese documento)

# Gestión

Este documento establece un plan de aseguramiento de la calidad con el objetivo de garantizar que el proyecto se entregue cumpliendo con los atributos de calidad identificados: usabilidad, funcionalidad y eficiencia. A través de este plan, se busca fomentar la colaboración entre todos los miembros del equipo, establecer canales de comunicación efectivos con las partes interesadas y definir procesos para la identificación y resolución de problemas. El objetivo final es asegurar que el producto de software cumpla con los requisitos funcionales y no funcionales establecidos en la documentación del proyecto. (como se gestionara el proyecto, podemos mencionar jira y la metodología como marco de trabajo, quien va a designar los roles fijos o intercambiables, se hará una planificación del proyecto y quien la hará product owner a que formatos nos debemos apegar, aquí van las buenas practicas de manera general)

# Documentación

(que es lo que lleva que documentación se va a entregar, cada modulo que se termine que documentación se entregará, requerimientos mínimos de documentación, ver las secciones, tareas responsabilidades, organización,)

# Especificiaciones

(como se van a documentar los requerimientos, diagramas, etc. Como debe estar definido sujeto, verbo o predicado, como voy a aceptar un requerimiento nivel de clarificación si no se tiene esa métrica que tengo que hacer)

**En estos documentos poner quien valida el documento**

# SDD

Cuales son las reglas para implementar ese documento, como representare esa arquitectura, cuantas inspecciones, cual es el mínimo de revisiones **Si vamos a usar scrum no podemos pedir que se tenga hecho la base de datos,** podemos pedir que cada que se agregue algo se deba someter a pruebas o incrementos **no actualizaciones**

# SVVP

Poner como vamos a documentar las pruebas, que formato, como diseñarlas, cuantas pruebas se van a hacer, un modulo no se puede considerar aprobado si no pasa tantas pruebas

# Documentación del usuario

Formato de la documentación, como algoritmo o por caso de uso o una ayuda como Excel.

# SCMP

Establecer la línea base, como se hará un cambio, cuando.

# Other

Si se va a usar xp, manual de estándares y procedimientos, planificación del software, plan del mantenimiento

# Organización

La estructura organizacional del proyecto es un componente esencial para garantizar la calidad del software. Las responsabilidades están distribuidas entre diversos equipos que interactúan para implementar efectivamente las actividades de calidad. El equipo de desarrollo es responsable de crear el software, realizar pruebas unitarias e integrales, y participar en revisiones de código para asegurar su consistencia. Por otro lado, el equipo de aseguramiento de calidad diseña y ejecuta los casos de prueba, analiza los resultados y gestiona los defectos que se identifican. El gestor del proyecto supervisa el progreso, asigna recursos y coordina a los equipos para cumplir con los objetivos planteados.

## Herramientas y tecnologías

Para garantizar una comunicación efectiva y una colaboración fluida entre los miembros del equipo, se utilizarán las siguientes herramientas:

* **Jira:** Esta plataforma servirá como el núcleo de nuestra gestión de proyectos, permitiendo un seguimiento detallado de las tareas, desde su creación hasta su finalización. A través de los tableros Kanban, visualizaremos el flujo de trabajo y podremos identificar rápidamente cualquier cuello de botella. Además, las incidencias secundarias nos permitirán vincular tareas relacionadas y mejorar la trazabilidad.
* **Google Drive:** Utilizaremos Google Drive para almacenar y compartir de forma segura toda la documentación del proyecto, incluyendo informes, platillas y estadísticas. La estructura de carpetas y los permisos de acceso garantizarán que la información sea accesible.
* **Microsoft Teams:** Esta herramienta será nuestro espacio de trabajo virtual, donde se llevarán a cabo las reuniones diarias (Daily Standup), se compartirán anuncios y se discutirán temas relevantes. Los canales específicos permitirán organizar la comunicación y facilitar la búsqueda de información.
* **GitHub:** Como plataforma de desarrollo, GitHub nos permitirá gestionar el código fuente de manera eficiente, colaborando en la creación de nuevas funcionalidades y corrigiendo errores. Las ramas de desarrollo y las revisiones de código garantizan la calidad del software.

# Métricas de Calidad

**SOLO AQUÍ PUEDO HABLAR DE USABILIDAD, etc., que entendemos por cosas como intuitividad, como medir y cuando va a ser aceptable**

(también van las métricas del producto, complejidad algorítmica, numero mínimo de pruebas, cual es el nivel de errores, que se entiende, como se va a medir, cuando es aceptable. **Aquí se detallan las buenas prácticas, como garantizar que se están ocupando,** (hay herramientas, podemos buscarlas))

Para evaluar la efectividad de las herramientas y el progreso del proyecto, se utilizarán diversas métricas de calidad. El ciclo de vida de los defectos medirá el tiempo promedio necesario para resolver un defecto. La cobertura de código reflejará el porcentaje del código que ha sido cubierto por pruebas unitarias, mientras que la densidad de defectos evaluará el número de errores por línea de código. Finalmente, la velocidad de desarrollo será medida por el número de historias de usuario completadas en cada iteración, lo que permitirá analizar el ritmo de trabajo y ajuste en función de los objetivos del proyecto.

# Actividades

Este plan abarca las fases clave del ciclo de vida del software, asegurando que se realicen las actividades más importantes para garantizar la calidad del producto final:

* **Planificación del proyecto:** Definir objetivos, criterios de aceptación, métricas de calidad, y asignar recursos y roles. Establecer un plan de gestión de riesgos.
* **Análisis de requisitos:** Validar y revisar los requisitos funcionales y no funcionales, asegurando su trazabilidad y verificabilidad.
* **Diseño del software:** Aprobar la arquitectura del sistema, validando la escalabilidad y mantenibilidad, así como las interfaces y dependencias entre módulos.
* **Desarrollo:** Implementar el código según estándares de calidad, realizar revisiones de código y pruebas unitarias.
* **Pruebas:** Ejecutar pruebas unitarias, de integración, de sistema y de aceptación, documentando resultados y gestionando defectos.
* **Implementación y despliegue:** Validar los criterios de aceptación, asegurar entornos de calidad y realizar un despliegue controlado.
* **Mantenimiento:** Corregir defectos post-despliegue, implementar mejoras y monitorear el rendimiento.

**Actividades de calidad a realizarse**

Las tareas de calidad cubren la evaluación, seguimiento y corrección de los productos generados durante el desarrollo del software. Las actividades incluyen:

1. **Revisar cada producto:**
   * Realizar revisiones detalladas de cada entregable producido en cada fase del ciclo de vida del software.
   * Asegurar que los productos cumplan con los estándares de calidad establecidos.
2. **Revisar el ajuste al proceso:**
   * Verificar que las actividades realizadas en el proyecto sigan los procedimientos definidos y estén alineadas con los estándares de calidad.
   * Documentar cualquier desviación respecto al plan inicial.
3. **Realizar Revisión Técnica Formal (RTF):**
   * Programar y ejecutar RTF en puntos clave del desarrollo para evaluar el cumplimiento de los requisitos técnicos y de calidad.
   * Las RTF incluyen revisiones de código, diseño, pruebas y documentación.
4. **Asegurar que las desviaciones son documentadas:**
   * Documentar todas las desviaciones identificadas durante el ciclo de vida del software.
   * Informar a los responsables de las desviaciones y hacer seguimiento hasta su resolución.
   * Implementar acciones correctivas para resolver las desviaciones, asegurando que no afecten la calidad final del producto.

# Revisiones y auditorias

Aquí va el plan de comunicación, como, cada cuando, con que tecnologías, las inspecciones se planifican, las auditorias no

Empezamos a definir cada cosa, como voy a hacer una revisión de mis requerimientos, como se hicieron las revisiones del diseño, como elegir al mejor.

De donde viene ese requerimiento, lo cumple el software, **Todo se revisa**

# Post mortem

Es referente a cosas que ya se revisaron

# Pruebas

Todas las pruebas que no estén metidas en el plan de validación, si todas están en el manual, se meten ninguna

# Problemas correctivos y reportes

Como nos vamos a comunicar

# Herramientas y tecnologías

**Aquí ya se definen en como github, jira, teams**

# Mantenimiento

Como guardar versiones anteriores, que va a pasar

# Entrenamiento

Que necesitamos que la gente sepa y como

# Manejo de riesgos

Que se va a hacer, se puede hacer referencia al plan de riesgos (obvio hacer el plan de riesgos en este va la lista de los riesgos)