Aseguramiento de la Calidad del Software
Tarea #1

André Arroyo Piedra 2015073657
Bryan Vargas 2015011562
Juan Villacis 2016201681

1 de septiembre del 2018

1 Introduccion

Este documento tiene como fin plantear las caracteristicas y atributos de calidad definidos por el ISO-9126, con el fin de asegurar la calidad del producto. Además se incluirán métricas para evaluar tales atributos y se incluira una lista de herramientas para aplicar las metricas.

2 Identificación de los atributos y su prioridad

A continuacion se mostrara un cuadro con las caracteristicas y atributos de calidad con los que debe contar el software, ademas se agregara un nivel de prioridad a cada atributo.

Caracteristicas externas e internas			
Caracteristica	Atributo	Prioridad	Justificacion
Funcionalidad	Exactitud	H	Los resultados tienen que ser precisos ya que en el ambito de salud un resultado inexacto puede tener consecuencias problematicas para los pacientes
	Cumplimiento de Funcionalidad	H	Es importante que cumpla sus funciones para que los usuarios puedan hacer un uso correcto de las imagenes y puedan investigar de una manera mas fiable
	Adecuación	L	Brindarle un adecuado grupo de funcionalidades al usuario es importante para la satisfaccion y comodidad de este a la hora de utilizar el programa
Fiabilidad	Madurez	H	Es importante que el sistema no sufra fallos, ya que un error podria poner en juego la salud de una persona
	Tolerancia a fallos	H	EL sistema no puede generar fallos o al menos si presenta alguno sucede rara vez y se recupera rapidamente

	Capacidad de Recuperación	M	Este software es capaz de recuperarse de un error facilmente sin inconvenientes
Usabilidad	Capacidad de ser entendido	L	Debe ser un programa que sea facil de comprender, sin necesidad de recurrir a las explicaciones de este constantemente
	Capacidad de ser operado	M	No debe tener una complicidad alta para el usuario, debe ser amigable con el usuario y tener una interfaz sencilla y apegada a sus necesidades
	Capacidad de ser aprendido	L	Debe ser un software que no lleve una curva de aprendizaje alta para una persona con conocimientos ya adquiridos sobre el tema de segmentacion de celulas
Eficiencia	Comportamiento temporal	H	Que la capacidad del sistema para completar el procesamiento de la imagen sea rápido y eficaz sin perder calidad en el resultado
	Utilizacion de recursos	L	El programa no requiere muchos recursos para cumplir con su función
Mantenibilidad	Capacidad para ser analizado	M	Si se puede determinar de una manera sencilla los posibles comportamientos que va a tener el software al ser modificado se podrá planificar de una manera mas sencilla los cambios que se deban aplicar
	Estabilidad	M	La importancia de este atributo es que si se cumple la aplicacion tendra una manera constante de comportamiento y esto permite que los usuarios no lidien con situaciones inesperadas
	Capacidad para ser modificado	M	Es importante poder hacer cambios en el software sin causar deterioros en otras partes del sistema
Portabilidad	Adaptabilidad	M	Este Software será capaz de adaptarse a los cambios referentes al entorno donde se use sin necesidad de muchos cambios
	Instalabilidad	M	El programa puede ser instalado en varios entornos de facil manera
	Capacidad para ser reemplazado	L	Será muy sencillo pero tendra la capacidad de reemplazar o ser reemplazado por otros sistemas por esta caracteristica de sencillez

3 Identificación de las métricas

En este cuadro se van a plantear las metricas que se utilizaran en cada atributo para evaluar su calidad y ademas se pondra un valor en el cual se considere aceptable.

Medición de la calidad			
Caracteristica	Atributo	Métrica	Valores aceptables
Funcionalidad	Exactitud	Cantidad de resultados que esten dentro del valor esperado	Este valor se ve determinado por la precision y exactitud del modelo usado para hacer la clasificación
	Cumplimiento de Funcionalidad	Cantidad de errores encontrados en las diferentes funciones del sistema	Los errores no deben de aparecer mas del 20% de las veces que se corra la aplicacion, para cumplir con el principio de Pareto
	Adecuación	Cantidad de funcionalidades principales y extra que tenga el software	Debe tener todas las funcionalidades necesarias para cumplir las tareas propuestas como minimo, ademas puede tener funcionalidades que ayuden a mejorar la experiencia con el usuario
Fiabilidad	Madurez	Cantidad de funciones con manejo de fallos	La totalidad de las funciones críticas deben manejar fallos correctamente
	Tolerancia a fallos	Cantidad de funciones sin fallo alguno	Al menos un 80% de las funciones no deben de tener fallos, esto se debe al principio de Pareto
	Capacidad de Recuperación	Cantidad de errores corregidos correctamente	Un 90% de los errores deben ser corregidos correctamente
Usabilidad	Capacidad de ser entendido	Tiempo que les toma a los usuarios comprender la aplicación	El usuario no debe de tardar mas de 20 minutos para entender el software

	Capacidad de ser operado	Tiempo en que el usuario tarda en hacer una funcion	Deberia ser menor a 10 minutos para preparar el ambiente y menor a un 40% del tiempo actual requerido para hacer la clasificacion
	Capacidad de ser aprendido	Cantidad de errores cometidos despues de haberles dado una explicación previa de la aplicación	Menos de 5 errores en promedio
Eficiencia	Comportamiento temporal	Tiempo requerido para completar el proceso sin afectar la eficacia	El programa es capaz de terminar el proceso en tan solo 10 minutos
	Utilización de recursos	MB de memoria necesarios para ejecutar la aplicacion	La aplicacion no debe ocupar más de 400MB en memoria, lo que equivale al 10% de la memoria principal de un computador de escazos recursos
Mantenibilidad	Capacidad para ser analizado	Lapso de tiempo que se dura en verificar el sistema al ser modificado	Debe ser menor a 8 horas, lo que equivale a un dia de trabajo
	Estabilidad	Lapso de tiempo en el que el sistema se mantiene trabajando de la misma manera sin mostrar errores o cambios inesperados	El sistema debe ser capaz de mantenerse activo durante al menos 20 dias luego de un fallo
	Capacidad para ser modificado	Cantidad de cambios que hay que realizar en el sistema para modificar una funcionalidad o caracteristica	Se debe modificar menos de un 10% de el sistema fuera de esa funcionalidad a cambiar
Portabilidad	Adaptabilidad	Tiempo que soporta el sistema en mantenerse funcionando apesar de los cambios de sortware en el tiempo	EL programa debe servir por más de 10 años

	Instalabilidad	Cantidad de máquinas diferentes donde puede ser instalado	Debe ser posible instalarse en cualquier sistema operativo apartir de windows 7 y en cualquier sistema linux
	Capacidad para ser reemplazado	Cantidad de aplicaciones que podrian cumplir con las mismas funciones	Al menos debe existir una aplicacion para cada función del programa

4 Definición del plan de evaluación de las métricas

4.1 Decripcion de metricas planteadas

A continuación se describiran las metricas mencionadas en la tabla 2.

1. Exactitud: Se correra la aplicacion y se contara cuantas veces estuvo dentro del rango aceptable.

2. Cumplimiento de Funcionalidad: Se correran todas las funcionalidades del sistema y se contabilizara cuantos y en cuales funciones se encuentran errores.

3. AdecuaciónL Se hara un conteo de las funcionalidades que completen de manera correcta el objetivo de la aplicacion

4. Madurez:

   Se colocaran dentro de las funciones parametros incorrectos para evaluar cuantas de estas soportan estos errores.

5. Tolerancia a fallos: Se probara haciendo correr la aplicacion repetidamente, todas las funciones pasaran por este proceso para contabilizar cuantas de estas no presentan ningun error.

6. Capacidad de Recuperacion: Esto sera alcanzado haciendo funcionar el sistema de manera que sea obligado a fallar, así se tomara el tiempo que tarda en recuperarse de los fallos.

7. Capacidad de ser entendido: Se contara el tiempo en el que el usuario tarda en leer la especificacion de como se utiliza la aplicacion y cuanto tiempo dura en hacer varias funciones sin consultar del todo la documentacion

8. Capacidad de ser operado: Se asigna una tarea a un usuario y se toma el tiempo que le tomo realizarla. Esto se repite con varios usuarios y se toma el promedio de todos.

9. Capacidad de de ser aprendido: Esto se logra contando que tantos errores comete un usuario al aprender a utilizar la aplicacion luego de explicarles como funciona. Se ponen varios usuarios a utilizar el sistema y se estima un promedio de errores cometidos entre todos.

10. Comportamiento temporal: Se pondrá la aplicación a prueba ya sea con un cronometro o una aplicacion como por ejemplo TRACKINGTIME y así ver cuanto dura esta en completar su tarea

11. Utilizacion de recursos: En el momento de correr la aplicaion se utilizara una aplicacion por ejemplo HWMonitor, que nos permite ver el estado de temperatura y energia entre otras más y así saber cuanto esta consumiendo esta aplicación en el tema de recursos.

12. Capacidad para ser analizado: Conteo de tiempo de lo que dura un programador en planificar un cambio que se deba aplicar

13. Estabilidad: Esto se puede comprobar simplemente tomando nota de la cantidad de veces que el sistema sufre un fallo en un lapso de tiempo, especialmente medido en semanas o meses.

14. Capacidad para ser modificado: Si se requiere hacer un cambio se contabiliza la cantidad de modificaciones fuera de la funcionalidad a cambiar

15. Adaptabilidad: Se evalua en un lapso de tiempo al sacar la aplicación, que tanto ha sido afectada por los cambios normales al avanzar la tecnologia.

16. Instalabilidad: Se instala la aplicacion en varias maquinas y así se puede saber en cuantas se pudo intalar y poner a funcionar correctamente.

17. Capacidad para ser reemplazado: Se evalua junto con otras aplicaciones similares y se revisa si entre ellas es posible compartir funcionales y asi saber que tan posible es reemplazar o que sea reemplazada por otra.

4.2 Herramientas de evaluación

Herramientas para medir el cumplimiento correcto de las metricas propuestas y verificar la correctitud de entregables:

1. Sonarqube: Detecta errores en el codigo, olores del software (malas practicas), vulnerabilidad en la seguridad y analiza el codigo con todos los posibles caminos para determinar pulgas.

2. Metrics 3: Análisis estático del código fuente de un proyecto (análisis ci-clométrico, etc.)

3. EclEmma Características: Permite saber el porcentaje de cubrimiento decódigo que se realiza en una ejecución o serie de ejecuciones.

4. Junit: Permite diseñar y verificar automáticamente conjuntos de pruebas.

5. Selenium: Automatiza pruebas realizadas desde una GUI.
6. Fortify: Encuentra fallos de seguridad.
7. Manejo de requerimientos: Reqview.
8. Star UML, zohoprojects, VisualParadigm, Visio : Documentación de requerimientos, casos de uso, diagramas de componentes y UML.
9. Encuestas y Observaciones: Con esto se pueden analizar resultados de la utilizacion del software del usuario
10. Observacion: Para medir algunas métricas de calidad se debera emplear en algunos casos la observación

5 Conclusión

A partir de esta informacion se podra analizar, contabilizar y utilizar esta informacion para tener una mejor vision y resultados del software en el que se esta trabajando.