Aseguramiento de la Calidad del Software

Tarea #1

Javier Contreras Muñoz, Randall Delgado Miranda, Sebastián Porras Fallas

September 1,2018

Abstract

Identifying the main quality attributes of a software project as well as their priority is very important because it helps the developer team to focus in what is essential in the project. Also, focusing on the main objectives of a project is a good approach because in most of the time a high quality system will be guaranteed as well as a system that contains only what is essential to satisfy the needs of the users.

The following document will include a chart with each of the six quality attributes established in the ISO-9126 standard. For each quality attribute, three subcharacteristics will be evaluated in priority with its corresponding justification. Also, for each of the chosen subcharacteristics, a metric will be established along with a tool which will facilitate the evaluation of the chosen metric.

Finally, a brief description of each tool will be given as well as the conclusions of the elaborated work.

1 Identificación de los Atributos y su Prioridad

External & Internal quality		
CHARACTERISTIC	SUBCHARACTERISTIC	WEIGHT (High/Medium/Low)
Functionality	Accuracy	High
	Security	Low
	Suitability	High
Reliability	Maturity	High
	Fault Tolerance	High
	Recoverability	High
Usability	Understandability	High
	Learnability	High
	Operability	High
Efficiency	Resource Utilisation	High
	Time Behaviour	High
	Efficiency	High
Maintainability	Testability	High
	Analysability	Medium
	Stability	High
Portability	Adaptability	High
	Co-existence	High
	Installability	High

2 Justificación

- **Functionality**
  - **Accuracy:** El atributo de exactitud define su prioridad como alta debido a que uno de los principales requerimientos de los usuarios es que, de manera automática, se pueda detectar la posición de las

células y que estas sean contabilizadas para cada cuadro. Si la precisión con la que se detectan las células es baja, muy probablemente se realice un conteo erróneo de las células lo cual no es deseable debido a que la información va a ser requerida para posterior investigación.

  - **Security:** El atributo de seguridad define su prioridad como baja debido a que el hecho de que la información sea accesada por terceros es poco probable ya que es un sistema con un contexto muy especifico. Además los datos son pocos sensibles y es otra razón por la cual es baja.
  - **Suitability:** El atributo de adecuación define su prioridad como alta. La razón de esto es que, como el sistema se va a usar en un contexto de medicina e investigación, los usuarios, en este caso los doctores que poseen cierto conocimiento informático, no van a querer funcionalidades extras las cuales no tengan nada que ver con el contexto. Asimismo, los doctores van a querer invertir el mayor tiempo posible en la investigación por lo que incluir funcionalidades fuera de tema podría causar contratiempos.

- **Reliability**

  - **Maturity:** El atributo de madurez se puede definir como la no presencia de fallas en el software. La razón de que su prioridad sea alta es porque la meta es entregar un software totalmente funcional en donde la ocurrencia de fallas sea ninguna o la mínima. Esto debido a que el software va a ser utilizado constantemente por los distintos usuarios. Si este sistema de manera usual empieza a fallar, el trabajo de investigación de los doctores comenzará a sufrir atrasos.
  - **Fault Tolerance:** La tolerancia a fallos se considera de prioridad alta, esto debido a que el cliente como tal necesita no sólo un sistema que cumpla con lo requerido, sino que también sea lo suficientemente confiable para que no presente fallas de manera constante. Dado el

ámbito en que se desenvuelve este software, se espera que, en caso de un problema no deseado, no intervenga en la ejecución y sea lo más transparente posible.

  - **Recoverability:** La recuperabilidad es definida como un atributo de importancia alta, ya que, en caso de que el software sufra algún error inesperado o no considerado por los desarrolladores y corrompa datos, debería haber una forma de volver a un punto atrás en el tiempo para dejar todo a como estaba. Si se puede regresar a datos de estudio guardados (ya sea manual o automáticamente), o simplemente a un estado anterior, la información es mucho más manejable y la confiabilidad aumenta.

- **Usability**

  - **Understandability:** Se define la capacidad de ser entendido como alta ya que los usuarios normalmente al enfrentarse a un software nuevo tienden a rechazarlo. Si se produce un software cuyas funcionalidades son bastante entendibles, entonces, el usuario va a aceptar el software con más facilidad.

  - **Learnability:** La capacidad de ser aprendido va de la mano con la capacidad de ser entendido. Esta la definimos como alta porque si se crea un software cuya curva de aprendizaje sea muy alta, el usuario muy posiblemente vaya a rechazarlo debido a la inversión de tiempo que le debe realizar.

  - **Operability:** El atributo de operabilidad fue definido como alto debido a que los usuarios no van a querer estar desperdiciando su tiempo en funcionalidades las cuales requieran entradas complejas, seguir instrucciones complejas, lo cual en cierta parte, estaría dificultando el uso del mismo sistema. Los usuarios, en este caso los doctores, van a querer maximizar el uso del tiempo en el análisis de los datos y no en dominar la operabilidad del sistema.

- **Efficiency**
  - **Resource Utilization:** Se define la utilización de recursos como alta ya que se considera que el sistema debe de consumir la menor cantidad del tiempo de procesador y memoria por razones de eficiencia. Esta es de suma importancia debido a que se va a realizar el procesamiento de miles de datos, en este caso imágenes.
  - **Time Behavior:** Se define el tiempo de respuesta como alto ya que no se considera que los doctores deban de esperar mucho tiempo después de suministrar un directorio con imágenes ya que el tiempo es vital para el avance de la investigación.
  - **Efficiency:** La eficiencia es identificada como de prioridad alta, ya que para efectos del software actual, se desea que realice sus funcionalidades de análisis y algoritmos con un nivel de rendimiento tal que se use la menor cantidad de entradas para obtener muchas salidas o resultados, tomando en cuenta factores como el tiempo y los recursos disponibles.
- **Maintainability**
  - **Testability:** Se define la capacidad de ser probado como alta ya que el sistema tiene que poseer las capacidades de permitirle al desarrollador probar cada funcionalidad con facilidad con el fin de entregar el sistema en la mejor condición posible.
  - **Analysability:** La analizabilidad resulta ser un atributo de prioridad media, ya que, aunque el software en sí tiene como propósito el análisis de imágenes y uso de patrones de reconocimiento (lo cual suena bastante directo), este atributo busca más bien que el software pueda ser diagnosticado y se identifiquen cambios necesarios o deficiencias presentes, lo cual resulta muy útil pero no indispensable para el software.

- **Stability:** La estabilidad se define como altamente importante, debido a que para un equipo de análisis, se espera que éste tenga la mayor continuidad posible; entre menos fallos presente, mejor. No se debe confundir con que tenga un buen manejo de errores, ya que eso es cubierto por otro atributo, sino que los fallos que se presenten no interrumpan la ejecución y sean manejados lo más transparente posible.

- **Portability**

  - **Adaptability:** Puesto que los usuarios tienen conocimiento informático, no se puede asumir que estos vayan a utilizar el mismo entorno en el cual se desarrolló la aplicación. La razón de que el atributo de adaptabilidad sea alto es que la aplicación debería adaptarse a cualquiera de los entornos de trabajo de los doctores. Si se limita a un solo entorno, existe la posibilidad de que los doctores estén trabajando en otro por razones de comodidad, eficiencia y además puede que tengan información importante la cual no pueda ser trasladada de un entorno a otro

  - **Co-Existence:** EL atributo de co-existencia se considera como alto debido que el sistema de segmentación de células utiliza diversos sistemas independientes a él. Un ejemplo de esto es el componente utilizado para la creación de CSV, Pandas y Keras que es el componente de Deep Learning.

  - **Installability:** La instalabilidad se define como la facilidad o el esfuerzo requerido para instalar un determinado software. La prioridad de este se considero como alto debido a que los usuarios van a querer tener su software funcionando lo más rápido posible. Como se está trabajando bajo un contexto médico, el análisis de los datos es de vital importancia para tomar una decisión sobre si el tratamiento aplicado al tejido celular es efectivo. Por esta razón, se necesita que

el software se encuentre listo y funcionando en el menor tiempo posible.

3 Identificación de las Métricas

Quality in Use Measurement Category			
CHARACTERISTIC	**METRICS**	**REQUIRED LEVEL**	**HERRAMIENTAS**
Accuracy	Cantidad de resultados correctos respecto al umbral establecido	95% de resultados correctos	Estadística
Security	Cantidad de riesgos identificados y su severidad	0-3 riesgos, de severidad baja/media	Fortify
Suitability	El porcentaje de funciones directamente relacionadas con la funcionalidad principal del sistema	90% de las funciones	Reportes de Operador
Maturity	Cantidad de fallas detectadas en un plazo de ejecución determinado	0-3 fallas detectadas en una hora	Stackify
Fault Tolerance	Porcentaje de recuperaciones exitosas del sistema tras n cantidad de fallos	95% de fallas recuperadas exitosamente	Stackify
Recoverability	Porcentaje de datos recuperados con éxito luego de una falla del sistema	90% de recuperación de datos	Backup Navigator
Understandability	Cantidad de letras promedio que tienen las etiquetas de las funciones	10-15 letras aproximadamente	Character Count & Word Count

Learnability	Cantidad de horas que le toma al doctor recorrer el software por completo	1,5 horas máximo	Librería Time de Python
Operability	Cantidad y tipo de funciones ejecutadas por el usuario en cierto tiempo	10 funcionalidades realizadas en una hora, de tipo variado	Pruebas α
Resource Utilization	Cantidad de memoria RAM que utiliza el programa	100 MB de RAM como máximo	Task Manager
Time Behaviour	Cantidad de imágenes procesadas por tiempo definido	50-60 imágenes procesadas en un minuto	Librería Time de Python
Efficiency	Cantidad de datos de entrada necesarios contra sus salidas respectivas	5 entradas para 10 datos de salida por imagen	Task Manager
Testability	Total de pruebas exitosas luego de modificar el software	95% de pruebas exitosas	Selenium
Analysability	Cantidad de lineas comentadas en el código	10 líneas por función implementada	campusMVPLOC
Stability	Tiempo de continuidad del software en ejecución sin interrupciones	2 horas apoximadamente sin presentar fallos	Reportes de los operarios
Adaptability	Cantidad de entornos en los que el sistema puede ejecutar de forma completa	Entre 5 y 10 entornos diferentes para ejecutarse	Reportes de los operarios

Co-Existence	Cantidad de horas para implementar un sistema externo al principal	2-3 horas de implementación	Reportes de los operarios
Installability	Tamaño del descriptor de despliegue (archivos de configuración) en líneas/elementos	200 líneas para archivos de texto, 10 elementos/atributos para archivos XML	CLOC

4 Notas

- Se debe tener en cuenta que algunas herramientas no necesariamente tienen que realizar análisis de software debido a la naturaleza de la métrica

5 Lista de Herramientas

- **Fortify:** Herramienta encargada de encontrar fallos de seguridad.
- **Task Manager:** Funcionalidad incluida en cada sistema operativo para analizar la utilización de recursos.
- **Selenium:** Herramienta encargada de ejecutar una serie de pruebas automáticas.
- **campusMVPLOC:** Herramienta encargada de encontrar el radio de líneas de código con los comentarios.
- **Librería Time:** Con esta herramienta tenemos acceso a funciones como timer o time con la cual podemos medir el tiempo de cada imagen procesada.
- **Reportes de los operarios:** Creación de formularios para que los empleados ingresen cualquier feedback del sistema.

- **Pruebas α:** Repartición de programas dentro del equipo de trabajo para verificar la operabilidad del sistema.
- **Stackify:** Herramienta que permite controlar y monitorear las tasas de error
- **Character Count & Word Count:** Herramienta en línea que permite contar la cantidad de caractéres de un archivo
- **Backup Navigator:** Herramienta que permite el análisis y monitoreo de los procesos de recuperación.
- **Estadística:** A través de la estadística se pueden evaluar numerosa cantidad de datos con el fin de determinar si una muestra específica cumple con determinada condición. Esta también puede ser utilizada para determinar que tan dispersos se encuentran los datos uno de otros (Desviación o Error Estándar)
- **CLOC:** Herramienta que permite contar líneas de código físicas, comentarios y espacios en blanco

6 Conclusiones

En conclusión, la evaluación de los atributos de calidad de un software es de suma importancia ya que con ello se puede establecer los puntos importantes en los que los desarrolladores se van a tener que enfocar durante todo el desarrollo del software. Asimismo, si no se toman en cuenta los diferentes atributos de calidad que pueden estar presentes en un sistema, al final se va a entregar un sistema el cual puede que sea funcional pero con el paso del tiempo se va a ir deteriorando de manera rápida.

Por otro lado, la necesidad de establecer métricas para los distintos atributos de calidad es de gran valor ya con ello se permite cuantificar el comportamiento

del sistema y determinar si este es apto y cumple con las necesidades impuestas por los usuarios.

La calidad en el software debe considerarse como un punto vital en cada proyecto, si bien esto es visto como una pérdida de tiempo para muchas empresas, se debe tener en cuenta que el costo invertido en el proyecto puede llegar a ser mayor que si se hubiera invertido el tiempo necesario en garantizar la calidad del producto.