TEC | Tecnológico de Costa Rica

Aseguramiento de la Calidad Software
IC-6831

Estándar ISO-9126. Atributos y métricas.

Profesor:

M. Sc. Saúl Calderón Ramírez

Estudiantes:

Carlos M. Girón Alas - 2014113159

Julián J. Méndez Oconitrillo - 2014121700

Daniel A. Troyo Garro - 2014073396

20 de agosto 2016.

Índice

1. Calidad Externa e Interna

A continuación, la tabla con las características, sus atributos o subcaracterísticas, el peso que tienen en la aplicación y la justificación por dicha elección.

Característica	**Subcaracterísticas**	**Peso (Alto, Medio, Bajo)**	**Justificación**
Funcionalidad	Precisión	Alto	El programa requiere del correcto posicionamiento de los jugadores puesto que una diferencia en la asignación de su área podría cambiar el informe final.
	Seguridad	Media	El único mecanismo de seguridad tiene como propósito que ningún agente externo distinto al DT y la junta. Por lo que un mecanismo de Log In estándar cumplirá el propósito.
	Idoneidad	Media	El software tiene que cumplir con las necesidades específicas de los clientes proporcionando el conjunto adecuado de funciones que maneje el procesamiento del video.
Confiabilidad	Madurez	Media	El programa tiene que tener la capacidad de poder evadir errores a nivel de lectura del video, lo cual puede ser dado por ángulos distintos o diferencias en los frames de estos por fallas propias y poder dar un resultado no tan desviado.
	Tolerancia a Fallos	Alta	Debido a que el programa podrá recibir una variedad y cantidad bastante alta de videos, debe ser capaz de manejar posibles fallos que ocurran por el formato de los mismos y no ver alterado bruscamente su desempeño, mostrado en este caso por la duración del procesamiento del video y generación del informe.
	Recuperabilidad	Alta	Errores de este tipo podrían ocurrir en la etapa de decodificación del video a frames, donde en ese caso se cancelará la operación; pero si ocurre en la lectura de un frame específico, debería ser capaz de ignorarlo y continuar procesando el resto, mostrando del error en el informe final.
Usabilidad	Operabilidad	Media	Aunque la aplicación inicial del software es bastante básica, tiene que permitir que el personal técnico y administrativo (DT y Junta) del Boca Juniors pueda interactuar con ella de manera eficaz y rápida, por lo que tiene que ser lo más operable posible.
	Entendibilidad	Alta	Dado que estamos en la primera iteración del proyecto y que se debe de entregar un PoC, es bastante importante que los usuarios comprendan las aplicaciones de este tipo de programas a diversos escenarios y condiciones de uso, por lo que la entendibilidad es algo de prioridad elevada.
	Conformidad de la usabilidad	Baja	El cumplimiento de estándares de calidad y de convenciones permitirá una mejor usabilidad del producto, aunque si la operabilidad y la entendibilidad tengan valores elevados en sus métricas, dichos estándares pueden ser levemente ignorados.

Eficiencia	Comportamiento sobre el tiempo	Alto	El tiempo es una variable sumamente importante en este tipo de aplicación, por lo que la duración que toma el procesado del video y generado de los informes tendrá una prioridad alta debido al impacto que tiene sobre el concepto del usuario sobre el programa.
	Utilización de recursos	Alto	Al igual que el tiempo, el consumo general de recursos es algo que afecta intensamente al concepto general de usuario sobre la aplicación, por lo que es uno de los atributos de calidad más importantes de todos.
	Conformidad de la eficiencia	Media	El seguimiento de convenciones y estándares relacionados a la eficiencia es importante puesto que permite que se disminuyan problemas con los dos atributos de calidad de esta misma categoría, por lo que tiene una prioridad media.
Mantenibilidad	Analizabilidad	Alta	El diagnóstico de deficiencias y causas de fallos en este tipo de programas es importante dados que suelen ser aplicaciones de muy pocas funcionalidades, por lo que la rápida detección de vulnerabilidades y fallas disminuiría considerablemente el tiempo de mantenimiento en detección y solución de errores.
	Modificabilidad	Alta	Ya que en este contexto el usuario pide una primera iteración del programa y que es más que todo un PoC, es posible que el cliente pida más cambios y funcionabilidades en el futuro, por lo que el producto de software debe mantener un atributo de calidad respecto a la modificabilidad con prioridad alta ya que los futuros cambios son muy probables..
	Estabilidad	Alta	Al ser un programa con una única funcionalidad por el momento, es importante que el sistema se mantenga estable ante futuros cambios y que no muestre un comportamiento inesperado, ya que esto podría afectar negativamente el concepto del usuario sobre el sistema.
Portabilidad	Coexistencia	Media	La coexistencia de este producto de software con otros programas independientes con consumo de recursos compartidos es algo que podría afectar a la eficiencia del programa, pero no a niveles drásticos, por lo que se le asigna una prioridad media.
	Instalabilidad	Baja	Ya que el producto de software será algo que no será instalado en muchas terminales de trabajo y es algo que hasta podría ser realizado mediante una aplicación web, la instalabilidad no es algo tan importante.
	Reemplazabilidad	Baja	Puesto que la variedad de este tipo de software específicos para este contexto es muy baja, este atributo de calidad no tiene mucha relevancia por el momento.

2. Métricas

La siguiente selección de métricas para cada atributo de calidad se realiza con el objetivo de alcanzar la mayor medición posible del estado actual de calidad del producto de software a desarrollar en el contexto dado de la tarea. Dado que algunas métricas no pueden ser obtenidas mediante una herramienta de software, se pondrá la fórmula requerida para obtenerla en la columna de Herramienta o Procedimiento. Todas las métricas expuestas fueron obtenidas de los reportes técnicos **ISO/IEC TR 9126-2 Software Engineering - Product Quality: External Metrics y ISO/IEC TR 9126-3 Product Quality: Internal Metrics.** De modo adaptativo para el contexto del proyecto o a la herramienta de software utilizada, puede que se realice ciertas modificaciones al proceso de obtención o variables requeridas.

Para la categoría de *Funcionalidad* se cuenta con tres atributos de calidad. El primero es el de la *precisión*, que será medido con con la **Exactitud Computacional** debido a que esta métrica **interna** cuenta el número de funciones que han implementado los requerimientos de precisión y la compara con las que tienen que hacerlo, pudiendo ver qué tanto han sido cumplidos los requerimientos de precisión. Se puede normalizar en el rango [0, 1] por lo que se establece un valor 0.7 como mínimo valor de calidad. Al ser un programa de análisis de video y detección de jugadores que luego se asignará en sectores, es importante que el mayor número de funciones cumpla estándares de precisión. La sub característica de la *seguridad* se ve relacionada al sistema de autenticación que debe implementar el programa para que sólo el director técnico y los miembros de la Junta Directiva del Boca Juniors la puedan utilizar. Se utilizará la métrica **interna** de **Controlabilidad del acceso**, la cual tiene como propósito ver qué tan controlable es el acceso al sistema y su método consiste en contar el número de requerimientos de control de acceso al sistema implementados frente al número de requerimientos de control de acceso a implementar según especificación. Al igual que la métrica anterior, es un promedio por lo que se ubica en el rango [0,1] y se le establece un valor mínimo de 0.9 ya que en este contexto el acceso al sistema es algo importante. Como sub característica final de esta categoría viene la *idoneidad*, que se le asignó la métrica **interna** de **Funcionalidad adecuada**, que consiste en analizar el número de funciones que son idóneas y han sido implementadas con problemas presentados para una tarea específica frente al número de funciones idóneas implementadas para tal tarea. Es un promedio y al igual que los otros dos valores se encuentran en el rango [0, 1]. Se le asigna un valor mínimo de 0.7.

La categoría de *Confiablilidad* cuenta con varios atributos que son importantes para predecir la posible satisfacción del producto de software ante las necesidades del cliente. La primera es la *madurez*, la cual en este contexto y al ser una primera iteración del mismo debe tener la capacidad de poder ser medida bajo una métrica **interna** como la de **Detección de Fallas**, que es algo muy importante en este tipo de producto de software debido a que sólo presenta un disminuido número de funcionalidades que deben de mostrar el mejor resultado y esta métrica logra detectar cuántas fallas han sido detectadas comparando el número de fallas detectadas en una revisión frente al número de fallas estimadas en una fase determinada. El valor mínimo debe ser menor a 0, puesto que al ser un promedio entre fallas detectadas y fallas estimadas es deseable que las fallas detectadas no sean mayores a las estimadas. Respecto a la *tolerancia a fallos*, es un atributo sumamente importante en este tipo de productos de software donde puede entrar una diversidad de videos con ángulos distintos y el sistema tiene que tener funciones a prueba de fallos que logren manejar casos no esperados. La métrica *interna* para este atributo es el **evitado de operación incorrecta**, que trata de saber qué tantas funciones implementadas con capacidad de evitar operaciones incorrectas. Al igual que la métrica anterior, se tratará que no supere el valor de 1 ya que esto significaría que hay más patrones de operación incorrecta considerados que el número de funciones que manejan o evaden este tipo de patrones. Finalmente, el atributo de *recuperabilidad* es también importante en este proyecto ya que pueden ocurrir errores al leer frames y el sistema tiene que ser capaz de recuperarse de estos errores así como de otros posibles tipos de errores. La métrica **externa** de **disponibilidad** determina qué tanto tiempo está disponible el sistema por un periodo de tiempo, lo cual en nuestro caso, debe ser un valor alto o al menos que siempre esté disponible cuando los miembros del club vayan a utilizarlo. El valor mínimo para esta métrica es de 0.9, lo que indica una muy alta disponibilidad.

Ahora, respecto a la categoría de usabilidad, cuyas métricas deben ser usadas para predecir el entendimiento y facilidad de uso que tendrá el usuario con el producto de software, se escogieron tres atributos primarios para este contexto, los cuales son la entendibilidad, operabilidad y conformidad de la usabilidad. El primero de estos atributos comprende la comprensión del usuario, en este caso los miembros del Boca Juniors, de las funciones y procesos del sistema. La *entendibilidad* es medida por la comprensión que se tiene de las distintas funcionalidades del programa y el uso que se les puede dar. Es medida por la métrica **externa Completitud de la Descripción**, que puede ser realizada mediante entrevistas a usuarios tras probar el sistema y ver qué proporciones de las funciones del sistema son comprendidas por ellos. Al ser una proporción, se ubica en rango [0,1] donde el valor mínimo es 0.6. Ya que el sistema será usado por el

DT del grupo, sería lo ideal que entienda al menos el funcionamiento del 60 % de lo que ofrece el producto de software. El atributo de *operabilidad* define qué tan operable es el sistema para el usuario y en este contexto la mediremos mediante el **chequeo de validez de las entrada**, métrica **interna** que trata de evaluar la cantidad de ítems que son validados. El método de esta métrica consiste en una comparación de los ítems validados frente a los que podrían ser validados. Lo ideal sería que fuera lo más cercano a 1 y dado que es un sistema que requiere de suficientes validaciones para no caer en errores no previstos. El valor mínimo asignado a esta métrica es de 0.85. El último atributo de esta categoría es la *conformidad de usabilidad*, la cual está relacionada con la utilización de estándares y cumplimiento de convenciones. La métrica **interna** utilizada para este atributo lleva el mismo nombre **(conformidad de usabilidad)**. Determina qué tan conforme se encuentra el producto de software según estándares relacionados a la usabilidad. El método consiste en comparar el número de ítems correctamente implementados que siguen estándares de usabilidad confirmados frente al total de ítems que siguen estándares de usabilidad. Al ser un promedio, tal y como la mayoría de métricas usadas en este proyecto, se mantiene en un valor [0,1]. Dado que se estableció al inicio del proyecto el seguimiento de procesos de calidad, lo ideal sería que el valor mínimo fuera de 0.8.

Pasando a la categoría de eficiencia, se escogieron los únicos tres atributos presentados en el estándar ISO-9126. El primero de estos es el *comportamiento sobre el tiempo*, atributo realmente importante en este sistema ya que sería lo ideal que el procesamiento del video y generación del informe tome el menor tiempo posible, aumentando la satisfacción del cliente final. La métrica **externa** para este tipo de atributo sería el **tiempo de respuesta**, que es el tiempo que se toma para realizar una tarea específica desde que se ordena la acción hasta que el sistema muestra un resultado. Lo ideal sería que fuera lo más bajo posible, dado que sería ideal que el usuario espere lo menos posible. Se estima que por un video de 10 minutos tarde máximo 180 segundos. Puede variar dependiendo de más variables como el formato del video o ángulo del mismo. Como segundo atributo y muy similar al primero, viene el *consumo de recursos*, que mediremos mediante el **consumo máximo de memoria**, el cual se espera que no supere los 300 MB. Al ser una métrica **externa**, un alto consumo de memoria afectará negativamente al usuario al ser más exigente el producto de software. La *conformidad de la eficiencia* es una métrica **interna** que asegurará que los ítems que requieren ser implementados siguiendo estándares de eficiencia lo sean implementados al calcular un porcentaje de estos comparando los implementados frente a los que tienen que ser implementados, por lo que se espera que sea un valor mínimo de 0.9. Se recuerda que al ser un promedio está en el rango normalizado [0,1].

Pasamos al campo de la mantenibilidad donde **todas las métricas fueron escogidas como internas** al ser un área más enfocada a los desarrolladores. El primero de los atributos de esta categoría corresponde a la *analizabilidad*, que establece qué tanto esfuerzo debe realizar el desarrollador para detectar posibles causas de error como de comportamientos no deseados del producto de software. Al igual que todos los atributos de esta categoría, sería ideal que tuviera un valor positivo ya que ayudaría a reducir el tiempo y esfuerzo dado al mantenimiento. La métrica usada para este atributo es el **registro de actividad**, que permite saber qué tan completo es el registro del estado del sistema mediante la comparación de las entradas en el registro de actividad frente al número de entradas requeridas en el registro de actividad. Con un valor mínimo de 0.8, es un promedio al igual que la mayoría de las métricas ubicado en el rango [0,1]. El siguiente atributo es la *modificabilidad*, que establece qué tan flexible es el producto al implementar una nueva función. Medido por la métrica de **registrabilidad del cambio**, dado en un promedio y con valor mínimo de 0.8. Pretende evaluar qué tan adecuados son los registros de cambios en el código con líneas de comentarios. Por último, el atributo de *estabilidad* será medido por el **impacto al cambio**, que tendrá un valor mínimo de 0.9 y tiene como propósito ver qué tantos cambios adversos han ocurrido en comparación al número de cambios totales. El contexto no afecta tanto a esta categoría, ya que son atributos que suelen ser muy generales en el desarrollo de software e independientes del contexto del producto de software.

En cuanto a portabilidad, las métricas escogidas fueron **dos internas** (coexistencia disponible y continuo uso de información) y una **externa** (facilidad de instalación). La categoría de portabilidad está compuesta por tres atributos para este proyecto: la coexistencia, la instalabilidad y la reemplazabilidad. La *instalabilidad* es medida mediante la métrica externa, **facilidad de instalación**. Esta métrica mide qué tan fácil o difícil puede ser para un usuario la instalación del software en un sistema. Ya que el software podría ser web, y se ha hecho mención de personal de mantenimiento por defecto este atributo no tiene gran prioridad. Bastaría con ver que tal es el proceso para unos pocos usuarios para saber si el proceso de instalación necesita cambios. La métrica **coexistencia disponible** se usa para medir el atributo de *coexistencia*. Este se usa para saber si el software en desarrollo puede interactuar con otros software presentes en los sistemas de los clientes. En la especificación no se hace mención de otros software relacionados. Si fuera el caso que hay alguno, o se agrega uno, bastaría con estandarizar los datos de salida del software para que los demás puedan utilizar esos datos. Finalmente para el atributo de *reemplazabilidad* se usa la métrica de **Continuo uso de información**, la cual mide que tanta de la información usada por software viejos se puede utilizar en el nuevo software a desarrollar. Nuevamente no

se hace mención de software similar existente previo a este, pero si lo hubiese se podría intentar agregar compatibilidad a los estándares de datos usados previamente para permitir que hasta cierto punto no sea necesario el uso del software viejo.

3. Atributos y sus métricas

Característica	Subcaracterísticas	Métrica	Nivel mínimo requerido	Herramienta o Procedimiento
Funcionalidad	Precisión	Exactitud Computacional <**Interna**>	0.7	Realizar pruebas con JUnit para asegurarnos que haya consistencia de datos a lo largo de las versiones.
	Seguridad	Controlabilidad del acceso <**Interna**>	0.9	Constantemente realizar pruebas con Fortify, para mantener altos niveles de seguridad.
	Idoneidad	Funcionalidad adecuada <**Interna**>	0.7	Crear pruebas en JUnit que cubran todos los casos de uso.
Confiabilidad	Madurez	Detección de fallos <**Interna**>	No mayor a 1	Se usaría JUnit para buscar posibles fallos cada vez que se actualice parte del sistema.
	Tolerancia a Fallos	Evitado de Operación incorrecta <**Interna**>	No mayor a 1	Configurar pruebas en JUnit para asegurar que no se puedan usar entradas que sean incorrectas para el sistema.
	Recuperabilidad	Disponibilidad <**Externa**>	0.95	Se aplica el cálculo del promedio manual con el tiempo disponible en un lapso de 1 mes tras release. Se podrían crear pruebas de JUnit para constantemente cerciorarse que el sistema siga disponible.
Usabilidad	Operabilidad	Chequeo de la validez de input <**Interna**>	0.85	Cerciorarse de que se validen todas las posibles entradas de video que podría recibir el sistema mediante JUnit.

	Entendibilidad	Completitud de la descripción <**Externa**>	0.8	Permitir al usuario probar el sistema y posteriormente llenar una encuesta mediante Google Forms para posteriormente analizar los datos.
	Conformidad de la usabilidad	Conformidad de la usabilidad <**Interna**>	0.8	Seguir fórmula de la métrica. Esta se puede medir con pruebas de JUnit.
Eficiencia	Comportamiento sobre el tiempo	Tiempo de respuesta <**Externa**>	180 segundos máximo por 10 minutos de video	Se utilizarán librerías del lenguaje en que se desarrolle la aplicación para validar que el tiempo de respuesta cumpla el mínimo. Las pruebas se pueden hacer con JUnit.
	Utilización de recursos	Consumo máximo de memoria <**Externa**>	300MB	Se correrían todas las pruebas que se tengan de JUnit en diferentes sistemas para asegurarse que el consumo de memoria no exceda los 200 MB.
	Conformidad de la eficiencia	Conformidad de la eficiencia <**Interna**>	0.9	Tras cada iteración del sistema se harían pruebas, posiblemente con JUnit, para cerciorarse que no se han violado los estándares y regulaciones.
Mantenibilidad	Analizabilidad	Registro de la actividad <**Interna**>	0.8	Se guardarán, de forma consistente, registros de la actividad del sistema en algún lugar externo al sistema, posiblemente una base de datos o un software gestor de proyectos. Posteriormente el cálculo se puede hacer manual.
	Modificabilidad	Registrabilidad del cambio <**Interna**>	0.8	Se utilizará Sonarqube para asegurarse que, cuando hayan cambios, hayan suficientes líneas de comentarios explicando el cambio.

	Estabilidad	Impacto del cambio <**Interna**>	0.9	Se realizarán todas las pruebas que se tengan de JUnit para analizar el impacto que tuvo un cambio en el sistema.
Portabilidad	Coexistencia	Coexistencia disponible <**Interna**>	0.7	Se analizarían los demás sistemas del cliente para permitir que los datos de salida del sistema sean compatibles con los otros. Se podría utilizar una herramienta automatizada de pruebas compatible con el lenguaje y arquitectura del sistema a reemplazar.
	Instalabilidad	Facilidad de instalación <**Externa**>	0.9	No requiere ninguna herramienta de software ya que se puede obtener al simplemente observar el proceso de instalación de una muestra de usuarios y aplicar estadísticas o tomar los tiempos, tomando promedios u otro cálculo manual. Se podría obtener estos datos mediante Google Forms.
	Reemplazabilidad	Continuo uso de información <**Interna**>	0.9	Por el momento no es de gran importancia. Mediante este proceso se analizaría los datos de entrada que usaba el sistema a reemplazar, de esta forma se agregaría la posibilidad de usar los mismos datos de entrada que usaba el sistema anterior, pero se desconoce de un sistema previo que utilice esta tecnología. Se podría utilizar una herramienta automatizada de pruebas compatible con el lenguaje y arquitectura del sistema a reemplazar.

Referencias

ISO/IEC, *TR 9126-2 Software Engineering - Product Quality: External Metrics.*

ISO/IEC, *TR 9126-3 Product Quality: Internal Metrics..*