# Atributos de Calidad

Para realizar este trabajo y determinar las métricas para cada atributo y sub características de calidad, el modelo a utilizar es el Modelo ISO/IEC 9126, donde se utilizarán las siguientes métricas para nuestra propuesta:

|  |  |
| --- | --- |
| Característica | Subcaracterística |
| Funcionalidad | Idoneidad |
| Seguridad |
| Fiabilidad | Madurez |
| Tolerancia a fallos |
| Usabilidad | Inteligibilidad |
| Facilidad de aprendizaje |
| Eficiencia | Comportamiento en el tiempo |
| Utilización de recursos |
| Mantenibilidad | Cambiabilidad |
| Estabilidad |
| Portabilidad | Adaptabilidad |

Por cada una de las subcaracterísticas mencionadas en las imágenes anteriores, se realizaron métricas para evaluarlas. A continuación se expondrán algunos atributos de calidad (característica) con sus sub características, además de la métrica que se plantea para evaluarla.

## Funcionalidad

Habilidad del software de realizar las funciones para las que fue creado.

**Idoneidad**

Capacidad del producto de software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuarios especificados.

**Métrica de Calidad Propuesta:**

Idoneidad = Requerimientos realizados sin modificaciones / Requerimientos de los usuarios.

Generalmente la gran mayoría de los requerimientos de los usuarios son realizados, aunque hay un porcentaje que no se realiza tal cual el usuario pretende, generalmente por temas técnicos.

Mientras el resultado sea más cercano al 100% es más óptimo, en cambio, mientras más cercano al 0% es un mal resultado.

**Seguridad**

Capacidad del producto de software para proteger la información y datos de manera que las personas o sistemas no autorizados no puedan leerlos o modificarlos.

**Métrica de Calidad Propuesta:**

Seguridad=(Criterio 1 + Criterio n)/n (Totalidad de Criterios)

Criterio 1 = Manejo de Usuarios y Perfiles.

Criterio 2 = Protocolo de Seguridad.

Criterio 3 = Encriptación interpaginas en ejecución (parámetros encriptados).

Criterio 4 = Encriptación de datos en la BDD.

Criterio 5 = Manejo de interrelación de páginas.

Criterio 6 = Caducidad de sesiones.

Los criterios mencionados anteriormente podría decirse que son un booleano, ya que es tan simple cómo que suman un 0 o un 1, ya que no debería establecer que uno de ellos cumpla en un 50%, ya que si cumple al 50%, simplemente no cumple.

Mientras el resultado sea más cercano al 100% es más óptimo, en cambio, mientras más cercano al 0% es un mal resultado.

El análisis que hemos pronosticado para el sistema en la actualidad da el siguiente resultado:

Criterio 1 = 1

Criterio 2 = 0

Criterio 3 = 1

Criterio 4 = 0

Criterio 5 = 1

Criterio 6 = 1

**Seguridad= 4/ 6= 66%**

## Fiabilidad

Habilidad del software para mantenerse operativo (funcionando) dentro de condiciones normales.

1. **Madurez**

Capacidad del producto de software para evitar fallar como resultado de fallos en el software.

**Métrica de Calidad Propuesta:**

Madurez= Cantidad de usuarios afectados \* cantidad de tiempo sin solución (días) dentro de un año calendario.

**Escala de madurez**

Excelente = Entre 0 y 5.

Muy Buena= Entre 6 y 10.

Buena= Entre 11 y 20.

Deficiente= Entre 21 y 30.

Muy Deficiente= Mayor a 31.

Cómo en todo software se reportan fallos, por algún motivo en específico, generalmente se da por casuísticas no contempladas en las etapas anteriores a la puesta en producción.

La métrica fue realizada en razón de que existen aproximadamente 30 usuarios del sistema.

**ii.** **Tolerancia a Fallos**

Capacidad del software de mantener un nivel especificado de prestaciones en caso de fallos de software o de infringir sus interfaces especificados.

**Métrica de Calidad Propuesta:**

Tolerancia a Fallos=Falla en funcionalidad específica/ Total de funcionalidades.

Tal cual lo mencionado anteriormente, un software siempre está propenso a algún tipo de falla.

Mientras el resultado sea más cercano al 0% es más óptimo, en cambio, mientras más cercano al 100% es un mal resultado.

## Usabilidad

Habilidad del software para que el usuario invierta el mínimo esfuerzo.

1. **Inteligibilidad**

Capacidad del producto de software que permite al usuario entender si el software es adecuado y cómo puede ser usado para unas tareas o condiciones particulares.

**Métrica de Calidad Propuesta:**

Inteligibilidad= Procesos en el sistema / Procesos antes del sistema

Esta métrica se aplica con que los usuarios detectan que lo que hacían antes del sistema, ya sea manualmente o con alguna otra herramienta informática, es capaz de hacer el sistema ahora.

Mientras el resultado sea más cercano al 100% es más óptimo, en cambio, mientras más cercano al 0% es un mal resultado.

**ii. Facilidad de Aprendizaje**

Capacidad del producto de software que permite al usuario aprender sobre su aplicación.

**Métrica de Calidad Propuesta:**

Facilidad de Aprendizaje=Tiempo de entrenamiento (días).

**Escala de facilidad de aprendizaje**

Buena=menor a 3.

Aceptable= mayor a 3 y menor que 6.

Deficiente= mayor a 6.

Se definió esta escala, ya que el sistema no es complejo de aprender, ya que está basado en las metodologías y lenguaje técnico que usan los usuarios, ya sean internos o externos.

## Eficiencia

Habilidad del software para responder a una petición del usuario a una velocidad apropiada.

1. **Comportamiento en el tiempo**

Capacidad del producto de software para proporcionar tiempos de respuesta, tiempos de proceso y potencia apropiados, bajo condiciones determinadas.

**Métrica de Calidad Propuesta:**

Comportamiento en el tiempo= Tiempo mayor de espera en procesos complejos (segundos)/cantidad de usuarios que aplica proceso.

Obviamente los procesos de análisis masivos, son aquellos que más demoran en el sistema.

Cuando hay rondas con alta tasa de participación, las iteraciones en los análisis son un poco más lentas que cuando los participantes son pocos y los análisis también son pocos.

Mientras el resultado sea más cercano al 0 es más óptimo, en cambio, mientras mayor se va haciendo la cifra es peor.

**ii. Utilización de recursos**

Capacidad del producto de software para usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas.

**Métrica de Calidad Propuesta:**

Utilización de recursos=(Uso de Procesador (%) + Uso de Ram(%)) / 2 (operaciones en el servidor).

Tal cual en la métrica anterior, los procesos de análisis masivos, son aquellos que más recursos ocupan en el sistema.

Mientras el resultado sea más cercano al 0% es más óptimo, en cambio, mientras más cercano al 100% es un mal resultado.

## Mantenibilidad

Habilidad del software para que el usuario invierta el mínimo esfuerzo para mantenerlo o mejorarlo.

1. **Cambiabilidad**

Capacidad del producto de software que permite una determinada modificación sea implementada.

**Métrica de Calidad Propuesta:**

Cambiabilidad=Cambios Cumplidos Satisfactoriamente/Total de Cambios Solicitados.

Hay problemas de tipo técnico muchas veces que genera la plataforma de desarrollo en la que se trabaja.

Tiene ciertas particularidades que hay que conocer cómo manejarlas.

Es oportuno señalar que también al estar encasillados en esta plataforma, hay cosas que no se pueden hacer.

Mientras el resultado sea más cercano al 100% es más óptimo, en cambio, mientras más cercano al 0% es un mal resultado.

1. **Estabilidad**

Capacidad del producto de software para evitar efectos inesperados debidos a modificaciones de software.

**Métrica de Calidad Propuesta:**

Estabilidad=Cantidad de instancias para evitar fallos.

Mientras más instancias se dispongan para evitar las fallas, es mejor para el sistema.

No se establece una métrica clara para evitar esto, ya que también es parte de los requerimientos de los usuarios, ya que si no están claros o son ambiguos pueden llevar a un error en el sistema:

**Estabilidad= 3 Niveles.**

1. Evita fallas en la especificación.
2. Evita fallas en la integridad referencial.
3. Evita fallas en la compilación.
4. No evita fallas en tiempo de ejecución.
5. Posee un manejo de errores inferior a un Try –Catch de C#.

**iv. Pruebabilidad**

Capacidad del producto de software que permite que el software modificado sea validado.

**Métrica de Calidad Propuesta:**

Pruebabilidad=Cantidad de instancias de pruebas + documentación del sistema.

Mientras las modificaciones al sistema pasan por más personas o equipos de personas, pueden ser detectados una mayor cantidad de anomalías.

Se consideró que la suma de ambos componentes, es lo mejor para medir esta métrica en el sistema.

## Portabilidad

Habilidad del software para ser transferido de un ambiente a otro y funcionar en este.

1. **Adaptabilidad**

Capacidad del producto de software para ser adaptado a diferentes entornos especificados, sin aplicar acciones o mecanismos distintos de aquellos proporcionados para este propósito por el propio software considerado.

**Métrica de Calidad Propuesta:**

Adaptabilidad=¿Multibrowser?+ ¿Responsivo?+ ¿Móvil?.

Cada uno de los ítems anteriores puede dar tres resultados posibles 0, ½ ó 1 punto.

Esto es si no cumple, cumple a medias o cumple totalmente, respectivamente.

Se debe dejar en claro que la primera opción Multibrowser aplica a las versiones más recientes de los tres browsers más utilizados a nivel mundial, que serían:

Google Chrome, Internet Explorer y Mozilla Firefox.

Mientras el resultado sea más cercano al tres, es un buen indicador, en cambio mientras cercano al cero, es peor el resultado.

El resultado para el sistema sería el siguiente:

**Adaptabilidad= 0,5+ 0 + 0= 0,5.**

# Requerimientos de desempeño

## Estabilidad del Producto

El Sistema debe ser estable, para ello debe evitar efectos inesperados producidos por modificaciones de Software, se cuenta cantidad de instancias para evitar fallos, mientras mayor cantidad de instancias es mejor para el Sistema, en este caso se revisa la integridad referencial de la Base de Datos, se revisa también los try- catch de C#.

## Transacciones

El sistema deberá soportar un mínimo de 40 transacciones concurrentes, aunque se estima que la cantidad de transacciones concurrentes normalmente no superará las 10.

Otras métricas de desempeño a utilizar para SQL Server son uso de memoria y procesador, tráfico de red y actividad de disco.

## Disponibilidad

El sistema deberé estar disponible al menos el 95% del tiempo (up-time), en este aspecto vamos a realizar test constantemente, algunos de ellos son: Web activa (up) o caída (down), Verificación de acceso de los usuarios del Sistema, velocidad de la web, verificación del dominio, verificación de los Servicios web.