**Requisitos No-Funcionales**

Compatibilidad de plataformas, significa que un producto es capaz de funcionar bien en varias plataformas. Una aplicación se vuelve compatible entre plataforma gracias a las herramientas que usamos en el desarrollo de la aplicación. Este requisito se debe definir en la etapa de diseño. Por ejemplo, deseo implementar una aplicación basada en web, que será un sistema CRM para respaldar mi negocio. Ahora digo que usaré PHP como lenguaje de scripting y MySQL como servidor de bases de datos.

La aplicación desarrollada es compatible con plataformas para Windows y Linux, ya que tanto estos son compatibles con PHP como MYSQL. Pero si hubiera optado por ASP.NET y MS SQL Server para el desarrollo, entonces habría necesitado una plataforma Windows para hospedarlo, no se puede ejecutar en la plataforma Linux

La privacidad, que usualmente puede ser considerado un requisito fundamental de un sistema debido a su gran importancia, pero no siempre es así, la privacidad puede ser un requisito secundario por ejemplo la aplicación de música Spotify, su funcionalidad principal es la reproducción de música, pero al tener funcionalidades sociales el usuario quiere tener control sobre quién ve lo que está escuchando, algo que de por si se puede ver como no fundamental, pero que para el usuario es de gran importancia. Este requisito se debe tener en cuenta en la etapa de desarrollo.

La calidad como requisito no funcional se define como la no inferioridad o superioridad del producto final. Esto quiere decir que el producto que se diseñó satisface las expectativas del cliente, en esta satisfacción de la cliente esta, el obvio funcionamiento del producto, cómo se compara con los competidores en el mercado, entre otras visiones subjetivas del cliente sobre el producto. La calidad es un atributo perceptual, condicional y algo subjetivo y puede ser entendido de manera diferente por diferentes personas. Este requisito se debe definir en la etapa de análisis con vistas hacia la operación y el mantenimiento.

La legibilidad es la capacidad de que un programador lea el código de otro, y lo entienda fácilmente ya sea para su uso o expansión. La legibilidad es importante porque los programadores pasan la mayor parte de su tiempo leyendo, tratando de comprender y modificar el código fuente existente, en lugar de escribir nuevo código fuente. El código ilegible a menudo conduce a errores, ineficiencias y código duplicado. Un ejemplo básico es una empresa de desarrollo de software, donde un programador le entrega un código a otro para que este continúe su desarrollo, que se complicaría si el código es ilegible. Este requisito se debe definir en la etapa de desarrollo.

La confiabilidad es la probabilidad de que el software funcione sin fallas durante un período de tiempo específico en un entorno específico. La alta complejidad del software es el principal factor que contribuye a los problemas de confiabilidad del software. Este requisito no funcional es de vital importancia, una confiabilidad del sistema alta provoca gran satisfacción de uso por parte del usuario, por ejemplo un software de compra debería estar listo para funcionar un largo tiempo, debido a que una pausa por arreglos seria una gran perdida de dinero para la empresa. Este requisito debe estar presente en la etapa de operación.

La capacidad de recuperación es la capacidad de proporcionar y mantener un nivel de servicio aceptable frente a fallas y desafíos para la operación normal ”.La recuperación sobre fallas es un punto fundamental. Este requisito debe implementar en la etapa de desarrollo.

El tiempo de respuesta, es el tiempo transcurrido entre una consulta en un sistema y la respuesta a esa consulta. Es utilizado como una medida del rendimiento del sistema, por ejemplo el tiempo que le toma a un sistema realizar una consulta a una base de datos y mostrarlos de forma ordenada. Este requisito se debe tener en cuenta en el diseño y se debe implementar en la operación.

La limitación de recursos, se refiere a las limitaciones de los insumos disponibles para ejecutar un programa o sistema como por ejemplo memoria, procesador, espacio en el disco. Este requisito se debe tener en cuenta en la etapa de análisis.

La reusabilidad es la capacidad de los elementos de software de poder utilizarse para la construcción de diferentes aplicaciones, por ejemplo en lugar de empezar desde cero con cada nueva aplicación, un programador consultará las bibliotecas de componentes existentes para ver si se pueden utilizar o son adecuados como puntos de partida para el diseño de una nueva aplicación. Este requisito se debe tener en cuenta en la etapa de desarrollo.

**Webgrafía**

<https://fabrictoken.io/tokengen-3-6-adapting-to-the-new-metamask-privacy-setting/>

<https://en.wikipedia.org/wiki/Non-functional_requirement>

Ambler, Scott. "Technical (Non-Functional) Requirements: An Agile Introduction". Agile Modelling. Ambysoft Inc. Retrieved 5 October 2018.

<https://www.wikihow.com/Improve-the-Readability-of-Your-Software-Code>

<https://dev.to/silviobuss/resilience-pattern-for-java-microservices-the-circuit-breaker-b2g>

<https://www.techopedia.com/definition/9181/response-time>