FUNDACIÓN ESCUELA TECNOLÓGICA DE NEIVA
JESÚS OVIEDO PÉREZ

Programa de Formación:	Ingeniería de Software
Tema:	Métricas y estándares de Calidad
Objetivo alcanzar:	El estudiante Procesa información teórica acerca de la calidad del software, para planificar un sistema de evaluación del software con predisposición al trabajo colaborativo. Diseña y construye estándares y medidas para garantizar la calidad de los procesos de software tomando como base estándares ISO y NTP 9001, 15504, 9126, 12119 y el modelo CMMI.
Duración de la Guía:	1 HORAS

Un poco de historia sobre la calidad en el desarrollo de software y sus principales estándares

A principios de la década de 1980, se dieron dos eventos que son el pistoletazo de salida para la generación de modelos de calidad o estándares de calidad en el desarrollo de software.

Antes, cada empresa que ofrecía productos de software tenía sus propios métodos de trabajo que en general eran considerados como información confidencial ya que representaban una ventaja estratégica.

Estoy hablando de unas cuantas marcas esencialmente dirigidas para grandes corporativos, como Sperry-Burroughs, IBM y Hewlett-Packard y para pequeñas empresas, como Digital Research. Era la época en que nombres desconocidos surgían como plaga, una gran cantidad de nuevas marcas llegaban al mercado y de las cuáles solamente dos prosperaron hasta llegar a dominar el mercado de cómputo personal y luego también el cómputo corporativo: Microsoft y Apple.

Paralelamente empresas alineadas a estas marcas ofreciendo productos adicionales especializados llegarían también a ser globales, como Oracle.

Otro factor fue que la introducción del cómputo personal a la vida cotidiana y el incremento del uso de software para acelerar los procesos empresariales comenzaron a impulsar una demanda insaciable por aplicaciones y soluciones para todo tipo de actividad.

Durante la Segunda Guerra Mundial el ejército estadounidense utilizó las primeras computadoras y el primer software en el desarrollo de la bomba atómica y después del final de la guerra continuó aplicándolos a otro tipo de armamento, como eran tan novedosos de operar implicaban la participación de los mejores científicos y técnicos de la época siendo actividades exclusivas del gobierno debido a sus costos tan altos.

Sin embargo, fue el desarrollo del transistor, primero y luego del circuito integrado lo que aceleró la utilización del software, y es así como se da el **primer evento** de los dos que mencioné al principio: El ejército de Estados Unidos de Norteamérica aprendió después de ganar la guerra que la tecnología tendría una influencia cada vez más decisiva para triunfar.

El uso de tecnología como el radar y los cohetes demostraron que la técnica avanzada daba una ventaja estratégica sobre el enemigo, esencial para la victoria.

Por lo tanto, ese aprendizaje lo transformó en métodos y procedimientos para aplicarla en cualquier tipo de armamento y consecuentemente el uso de software y circuitos integrados no fueron excepción.

En 1982 durante la guerra de las Malvinas, entre Inglaterra y Argentina, un misil tierra-aire destinado a derribar aviones salió de un destructor. El misil perseguía al avión guiado por el calor de sus motores, pero debido a una falla en el software embebido en el circuito integrado del misil, este confundió a su tobera de salida como la fuente de calor más caliente, regresando al barco de dónde había sido lanzado provocándole daños importantes.

Este evento marcó un hito para el Departamento de Defensa de los Estados Unidos de Norteamérica: “debido a la utilización cada vez más intensiva del software en el armamento, si no está bien diseñado, construido y probado, nuestras armas pueden actuar en nuestra contra, es imprescindible contar con un método de trabajo que garantice una alta calidad de los productos de software, ya sea embebido en circuitos integrados o corriendo en algún hardware”.

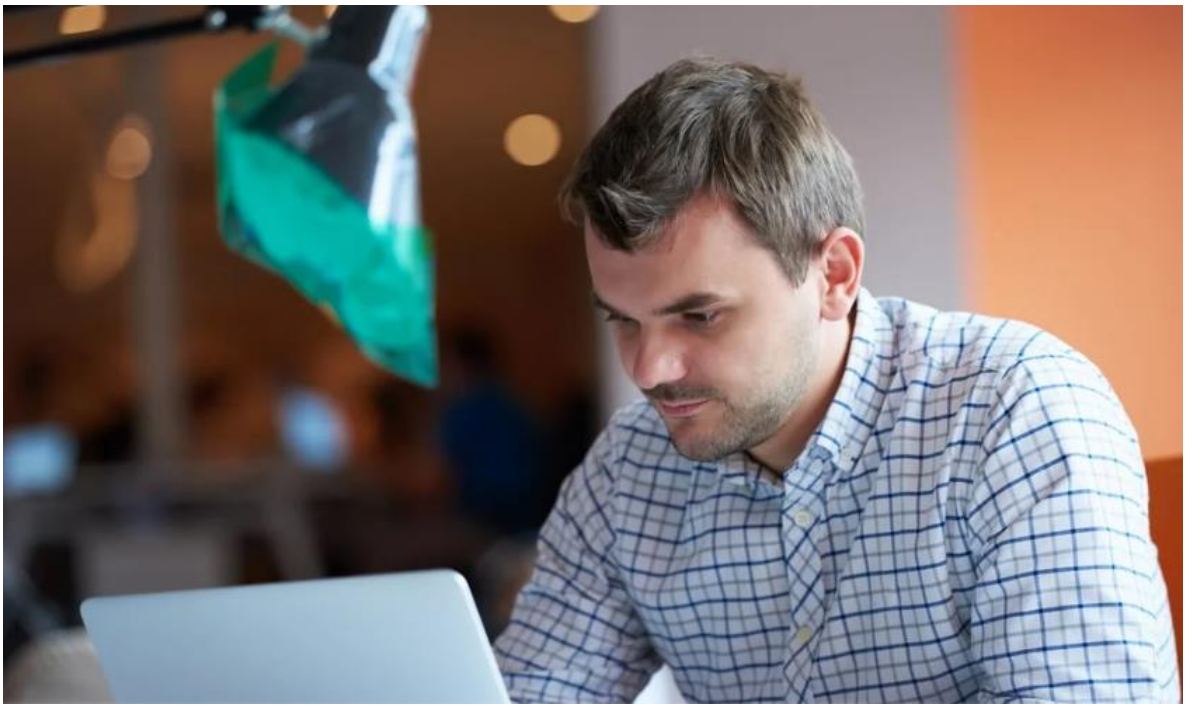
El **segundo evento** se dio cuando Margaret Thatcher asumió como primer ministro de Inglaterra, encontrando un país en caos organizacional, operativo, social y financiero. En 1980 una de las primeras cosas que hizo fue exigir a su gabinete de tecnología un modelo de uso y diseño de productos de TIC (Tecnologías de la Información y Telecomunicaciones) para obtener productos confiables y de alta calidad que al utilizarlos permitieran al gobierno altos niveles de desempeño en beneficio del ciudadano.

Para solucionar su problema, el departamento de defensa de los Estados Unidos acudió al Instituto de Ingeniería de Software de la universidad Carnegie-Mellon, para encontrar un método de control de proyectos de software que satisficiera su demanda, publicando en noviembre de 1986 el modelo CMM (Capability Maturity Model).

Por su parte, el gobierno británico acudió a la OGC (Office of Government Commerce), hoy inexistente ya que ha sido fusionada en diversos ministerios para encontrar la solución, primero en la norma BS15000 (*British Standard*) y después promoviendo la creación, a través de la ISO (*International Organization for Standardization*), de una norma de uso

internacional para asegurar que cualquier proveedor certificado en ella ofreciera un nivel adecuado de calidad en sus productos de software, publicando en julio de 1995 la primera versión de la norma ISO 12207 para el Ciclo de Vida del Software.

Otros organismos han venido trabajando y publicando estándares y normas en este mismo sentido, como el IEEE (Institute of Electrical and Electronics Engineers), pero no han tenido tanta divulgación, sin embargo, sin importar la norma o modelo, todas buscan lo mismo: que la empresa logre productos de software de alta calidad.



Calidad en el desarrollo de software: ingeniería y fabricación

Hablemos sobre las características que tienen en común todos los modelos y normas de desarrollo de software, considerando cómo se administra exitosamente un proyecto de desarrollo de software.

Lo primero que hay que advertir, es que un proyecto de desarrollo de software no es un proyecto "inspiracional", no depende de la **creatividad** del desarrollador. Este mito es el primero que deben descartar los propios desarrolladores.

El desarrollo de software es un **proyecto de ingeniería de diseño** en donde la complejidad del proyecto depende de la sofisticación de la tecnología utilizada y de la cantidad de gente participante y demandan las habilidades de administración de un jefe de proyecto, similar a como es el proyecto de diseño/desarrollo de un automóvil, un avión, un puente o un electrodoméstico, y es que el software es un producto tan real como estos, aunque no sea tangible.

Así mismo es un **proyecto de construcción de productos,** que demanda la utilización de técnicas de control de la producción como se aplica a cualquier fábrica.

Por lo tanto, el desarrollo de software simultáneamente es un proyecto de ingeniería y una fábrica de productos.

En un reciente estudio del consorcio Stripe, se reporta que existen en el mundo 18 millones de desarrolladores de software que en promedio trabajan 41.1 horas por semana y de las cuales 17.3 son de "deuda técnica", es decir el software desarrollado no es lo más eficiente posible y tiene mal código. Estamos hablando de que el 42% del tiempo de un desarrollador no es utilizado para generar código de alta calidad.

En el mismo informe se anota que el mercado global de desarrollo de software es de 3 millones de millones de dólares y que la ineficiencia evaluada representa un total de 300,000 millones de dólares anuales de pérdida del PIB mundial, debida solamente a la ineficiencia del desarrollador. Esto también se traduce en que solamente un 22% de los proyectos de software en el mundo terminan exitosos plenos, un 60% con éxito parcial y un 18% en fracaso.

Tipo de proyecto.	Características.	Porcentaje Global.
Éxito pleno. **Cumple con todas las características.**	En plazo. En presupuesto. Con la funcionalidad solicitada.	22%
Éxito parcial. **Cumple con una o todas las características.**	Con retraso. Con mayor presupuesto. Con funcionalidad parcial.	60%
Fracaso.	Abandonado.	18%

Que el 78% de los proyectos no sean plenamente exitosos sucede precisamente por considerar que el desarrollo de software no es un proyecto de ingeniería.

Es necesario que las empresas que desarrollan software comprendan lo que ya explicamos antes: que son una empresa de proyectos de ingeniería, por ejemplo, una empresa que construye plantas industriales y por lo tanto debe incluir prácticas de administración de proyectos como tal, pero que además es una empresa de desarrollo de productos, por ejemplo, una fábrica de muebles y por lo tanto requiere de técnicas de control de calidad de productos como tal.

Entonces, en conclusión: **un modelo o norma en calidad en desarrollo de software es una propuesta de procesos y prácticas de ingeniería de proyectos y de procesos y prácticas de control de calidad de productos.**

Ciclo de vida del desarrollo de software

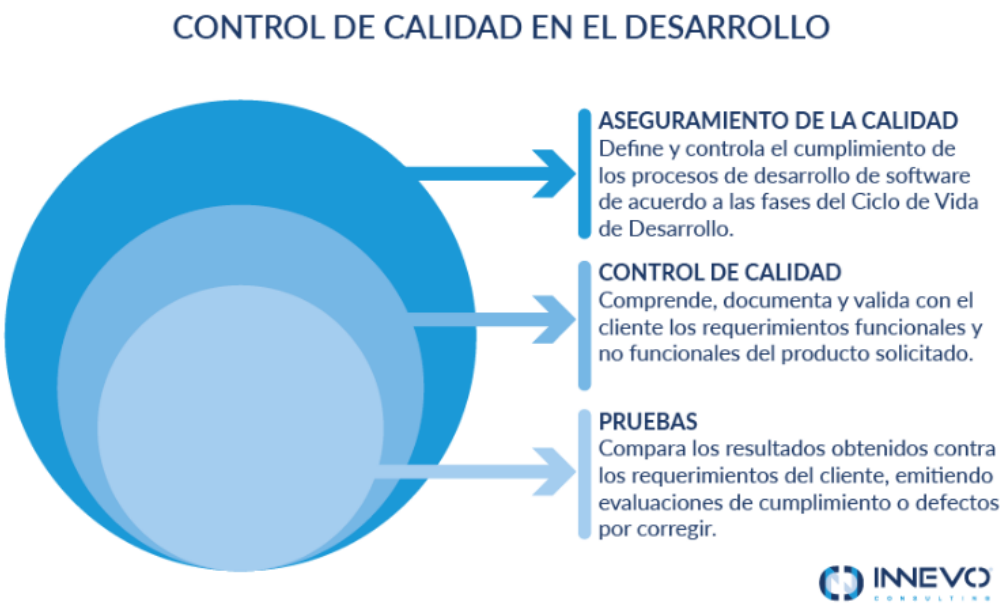
Para abordar el desarrollo del software como proyecto de ingeniería, los modelos o normas de calidad se basan en el control del Ciclo de Vida del Desarrollo contemplando genéricamente las principales fases de cualquier proyecto de software:

1. Planeación
2. Análisis
3. Diseño
4. Desarrollo
5. Pruebas y liberación
6. Despliegue

Control de calidad del desarrollo de software

Cuando se aborda el desarrollo del software como fabricación de un producto los modelos o normas de calidad se enfocan en el control de calidad de fabricación de este, contemplando genéricamente las principales características de una eficiente línea de producción:

- **Aseguramiento de la calidad:** Define y controla el cumplimiento de los procesos de desarrollo de software de acuerdo a las fases del Ciclo de Vida de Desarrollo.
- **Control de calidad:** Comprende, documenta y valida con el cliente los requerimientos funcionales y no funcionales del producto solicitado.
- **Pruebas:** Compara los resultados obtenidos contra los requerimientos del cliente, emitiendo evaluaciones de cumplimiento o defectos por corregir.



¿Cuál es el mejor modelo o norma para el desarrollo de software? ¿Cómo elegir la ideal para tu empresa?

Todos los modelos o normas de calidad en software contienen el aspecto de la ingeniería y de la calidad del desarrollo, por lo tanto, la pregunta siguiente es: ¿cuál es el mejor? La respuesta que te voy a dar espero no la tomes de Perogrullo: cualquiera. Debido a que tienen el mismo objetivo,

sus diferencias esenciales son sólo la manera en cómo ordenan o estructuran los procesos necesarios para lograrlo.

CMMI facilita la evolución de la empresa porque está estructurado en niveles de capacidad, técnicamente le llama niveles de madurez. Mientras más alto el nivel de madurez la empresa tiene mayor capacidad de control de sus procesos. Por lo tanto, ofrece un camino paulatino hacia la alta calidad.

Las normas ISO exigen la implementación de todos los procesos de la norma en una sola acción, no hay desempeño gradual.

Ambos requieren una auditoría de certificación (en CMMI se llaman evaluaciones Benchmark, pero esencialmente es lo mismo) realizada por una persona capacitada y autorizada por el organismo propietario del modelo, para el caso de las normas ISO es la misma ISO y para CMMI es la ISACA, quien avala que la empresa revisada cumple con los requerimientos, procesos y prácticas exigidos por la norma.

Relacionado: *Modelo CMMI Para Desarrollo*

Entonces surge una pregunta nueva en base a la respuesta anterior: **¿Qué norma me conviene?** Aquí sí hay una respuesta diferenciada. El criterio de selección que debes utilizar es el siguiente en orden de prioridad según la respuesta que tengas a las siguientes preguntas.

¿Qué norma prefiere mi cliente?

- Si conoces la respuesta, esa es la norma en que debes certificar tu empresa.
- Si a tu cliente le es indiferente cuál norma, entonces debes hacerte la siguiente pregunta.

¿Qué norma prefiere mi mercado?

- Si conoces la respuesta, esa es la norma en que debes certificar tu empresa.
- Si no hay preferencia por alguna norma, entonces debes hacerte la siguiente pregunta.

¿En qué norma está certificada mi competencia?

- Si conoces la respuesta, en esa norma debes certificar a tu empresa.
- Si a todas las preguntas no hay respuesta concreta, entonces debes hacerte la siguiente y final pregunta:

¿Tengo tiempo para certificarme? Es decir, ¿es un tema prioritario?

- Si la respuesta es no, entonces busca certificar a tu empresa en CMMI nivel 3. Se puede, requiere de una estrategia especial, pero es perfectamente lograble
- Si la respuesta es sí, entonces certifica a tu empresa en nivel 2 y luego aborda los siguientes niveles de madurez, hasta llegar al nivel 5.

Pero falta una pregunta que seguramente ya te hiciste: **¿Qué es más caro, CMMI o ISO?** La respuesta es, CMMI. ¿Qué tanto más caro? En promedio un 25%. **¿Es mucho dinero?** No, el costo total representa en promedio el salario que pagas mes/hombre en tu empresa por 08 meses.

La última pregunta que quizá te estés haciendo, **¿cuál es la norma más aceptada en el mercado?** Por mucho, es decir mayoritariamente, CMMI. Para que puedas compararlo, de cada certificación en ISO hay 5 certificaciones en CMMI. El gobierno chino exige a las empresas de desarrollo de software que deseen venderle servicios que estén certificadas en CMMI. Eso te da una idea de en lo que se está convirtiendo este modelo.

Sin embargo, la norma que elijas será de beneficio para tu empresa. Lograrás distinguir tu marca por la calidad de sus productos y por la satisfacción de tu cliente. Si tu empresa entrega software de alta calidad a sus clientes ¿crees que les importa con qué norme lo logra? Evidentemente no. Lo único que valora es que tiene un **proveedor de excelencia**. Realmente lo negativo es que no tengas certificada a tu empresa y no en qué norma está certificada.

¿Cuándo piensas abordar una certificación? ¿Cuándo tu cliente te presente quejas frecuentes por los problemas de calidad en tus productos o de inmediato para aventajar a tu competencia?