CALIDAD

Antes de iniciar las actividades de aseguramiento de la calidad del software es importante definir que es la Calidad del Software:

> "Concordancia con los requisitos funcionales y de desempeño explícitamente establecidos, estándares de desarrollo explícitamente documentados y características implícitas que se espera de todo software desarrollado profesionalmente."

La calidad se refiere a características mensurables y cuando una entidad se examina en base a estas existen dos tipos de calidad:

- CALIDAD DE DISEÑO: requisitos, especificaciones y diseño del sistema.
- CALIDAD DE CONCORDANCIA: se refiere a si la implementación sigue el diseño y el sistema resultante satisface sus requisitos y metas de desempeño.

CONTROL DE CALIDAD

El control de la variación es la clave para un producto de calidad. En el contexto de software se lucha por controlar la variación en el proceso genérico que se aplica, por ejemplo: de un proyecto a otro se quiere minimizar la diferencia entre los recursos predichos y los realmente utilizados, o se busca minimizar el número de defectos de una liberación a otra.
El control de calidad es una serie de inspecciones, revisiones y pruebas empleadas a lo largo del proceso de software para garantizar que cada producto de trabajo satisfaga los requisitos que le se le han asignado. Incluye un bucle de retroalimentación que permite afinar el proceso cuando los productos de trabajo no satisfacen sus especificaciones.

COSTO DE LA CALIDAD

Incluye todos los costos que genera la búsqueda de calidad o que demanda el desarrollo de las actividades relacionadas con la calidad. Estos de dividen en:

- Costos de Prevención: planificación, revisiones técnicas formales, equipo de prueba y entrenamiento.
- Costos de Evaluación: inspección en el proceso y entre procesos, calibración y mantenimiento de equipos y prueba.
- Costos de fallas: son aquellos que desaparecerían sino aparecieran defectos antes de enviar un producto a los clientes. Se subdividen en:
  - Costo de Fallas Internas: aparecen cuando se detectan defectos en el producto antes del envío. Incluyen reelaboración, reparación y análisis en modo de fallas.
  - Costo de Fallas Externas: se asocian a defectos detectados después del envío del producto al cliente. Ejemplo: resolución de quejas, devolución y reemplazo del producto, soporte de ayuda en línea y trabajo de garantía.

GESTIÓN DE CALIDAD

La Gestión de Calidad es una actividad que se aplica a lo largo de todo el proceso de software, y abarca:

1. Un proceso de Garantía de la calidad del software (SQA)
2. Tareas específicas de aseguramiento y control de calidad (que incluyen revisiones técnicas formales y una estrategia de pruebas de varios niveles)
3. Prácticas efectivas de ingeniería de software (métodos y herramientas)
4. Control de todos los productos de trabajo del software y los cambios que generan
5. Un procedimiento para garantizar la concordancia con los estándares de desarrollo del software (cuando sea aplicable)
6. Mecanismos de medición e informe

Garantía de la Calidad del Software (SQA)

Es un patrón sistemático y planificado de acciones que se requiere para garantizar alta calidad en el software. Es un conjunto de funciones de auditoría e información que evalúan la efectividad y que tan completas son las actividades de control de calidad.

La garantía de calidad involucra a:

- Los ingenieros de software que realizan el trabajo técnico (revisiones y pruebas)
- Un grupo SQA que tiene la responsabilidad de planificar, supervisar, guardar registros, analizar y reportar la garantía de calidad.

El SEI (Software Engineering Institute) recomienda un conjunto de actividades a ser realizadas por un grupo SQA independiente:

a) Preparar un Plan SQA para un proyecto ▪ identifica las evaluaciones, auditorías y revisiones a realizar, los estándares aplicables al proyecto, los procedimientos para el informe y seguimiento de errores, documentos a producir y la retroalimentación proporcionada al equipo de proyecto .
b) Participar en el desarrollo de la descripción del proceso de software del proyecto. ▪ revisa que el proceso concuerde con las políticas organizacionales, estándares internos y externos (ej.: ISO 9001) y otras partes del Plan de Proyecto del Software.
c) Revisar las actividades de ingeniería del software para verificar que se ajusten al proceso software definido ▪ identifica, documenta y sigue las desviaciones del proceso y verifica que se hayan hecho las correcciones.
d) Audita productos de trabajo de software seleccionados para verificar que se ajusten con los definidos como parte del proceso de software.
e) Garantiza que las desviaciones en el trabajo del software y en los productos de trabajo estén documentadas y se manejen de acuerdo con el procedimiento establecido.
f) Registrar cualquier falta de ajuste y lo informa al gestor ejecutivo

Además de estas actividades, el grupo SQA coordina el control y la gestión del cambio y ayuda a recopilar y analizar métricas de software.

Revisiones

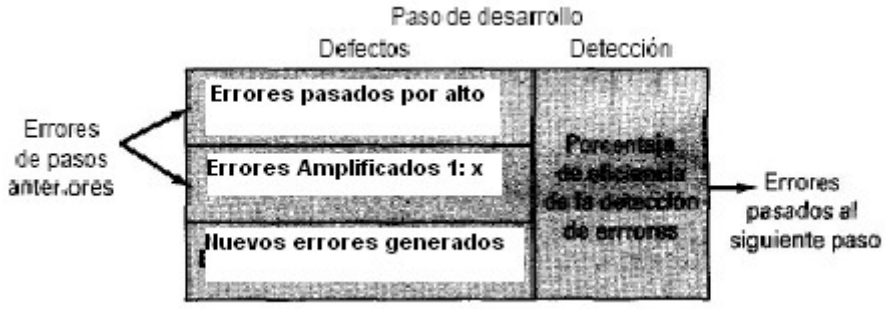
Las revisiones se aplican en varios puntos durante la ingeniería del software y sirven para descubrir errores y defectos que luego pueden eliminarse. Existen muchos tipos, de los cuales se consideran el medio más efectivo para descubrir errores y mejorar la calidad del SW la llamada Revisión Técnica Formal o Inspección de código.

Revisiones Técnicas Formales (RTF)

Es una actividad de control de calidad del software lleva a cabo por una Junta de Revisión que incluye recorridos, inspecciones, revisiones cíclicas y otro pequeño grupo de evaluaciones técnicas de software. Una RTF sólo tendrá éxito si se planifica, controla y atiende apropiadamente.
Su objetivo principal es descubrir los errores durante el proceso, de modo que no se conviertan en defectos después de liberar el software. Otros objetivos son:

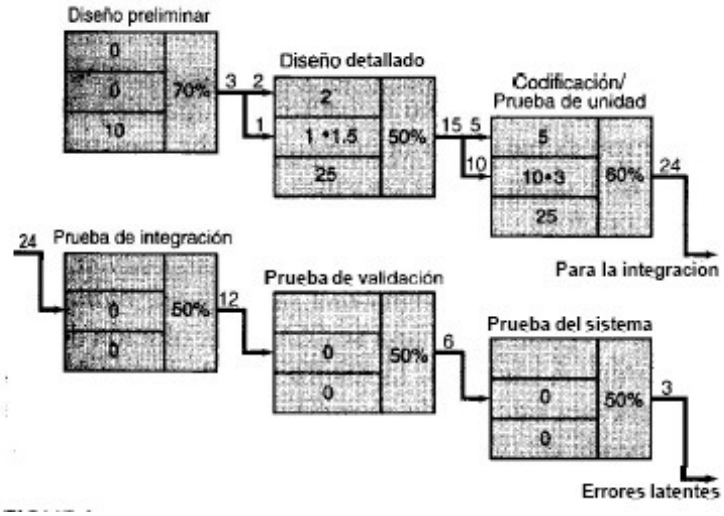
a) Descubrir errores en la función, lógica o implementación en cualquier representación del software.
b) Verificar que el software en revisión satisface sus requisitos.
c) Garantizar que el software se ha representado de acuerdo con los estándares predefinidos.
d) Lograr software desarrollado en una manera uniforme
e) Hacer proyectos más manejables.

El beneficio de las RTF es el descubrimiento temprano de errores, evitando que los mismos se propaguen al paso siguiente en el proceso de software y reduciendo sustancialmente los costos de las actividades subsecuentes.
Para ilustrar la generación y detección de errores se puede usar un Modelo de Amplificación de Defectos según el siguiente esquema:



En este modelo, un recuadro representa un paso de desarrollo de software. Durante un paso, **nuevos errores** se pueden generar de manera inadvertida. La revisión puede fallar en descubrir errores generados de manera reciente y errores de pasos anteriores, lo que resulta en un cierto número de **errores que se pasan por alto**. Algunos de los **errores pasados por alto desde pasos anteriores se amplifican** con el trabajo actual (factor de amplificación x). Las subdivisiones de los recuadros representan estas características y el porcentaje de eficiencia de la detección de errores, una minuciosidad de la revisión.

Ejemplo de generación y detección de errores durante el diseño preliminar, el diseño de detalles y los pasos de codificación:



Junta de revisión

Cualquier reunión de revisión debe acogerse a las siguientes restricciones:

- Deben involucrarse para la revisión (normalmente) entre tres y cinco personas;
- Se debe preparar con anticipación, pero sin que requiera más de dos horas de trabajo a cada persona;
- La duración de la reunión de revisión debe ser menor de dos horas.

Es por esto que cada RTF centra su atención en un producto de trabajo, es decir, en una parte específica y pequeña del software total, con lo que la probabilidad de descubrir errores es mayor. Ejemplo: una porción de una especificación de requisitos, un diseño detallado del módulo, un listado del código fuente de un módulo).

El proceso es el siguiente:

a) El individuo que ha desarrollado el producto -el productor- informa al jefe del proyecto de que el producto está terminado y que se requiere una revisión.
b) El jefe del proyecto contacta con un jefe de revisión, que evalúa la disponibilidad del producto, genera copias del material del producto y las distribuye a dos o tres revisores para que se preparen por adelantado.
c) Cada revisor estará entre una y dos horas revisando el producto, tomando notas y también familiarizándose con el trabajo.
d) De forma concurrente, también el jefe de revisión revisa el producto y establece una agenda para la reunión de revisión que, normalmente, queda convocada para el día siguiente.
e) La reunión de revisión es llevada a cabo por el jefe de revisión, los revisores y el productor. Uno de los revisores toma el papel de

registrador de forma escrita) de todos los sucesos importantes que se produzcan durante la revisión.

f) La RTF comienza con una explicación de la agenda y una breve introducción a cargo del productor. Entonces el productor procede con el «recorrido de inspección>>del producto, explicando el material, mientras que los revisores exponen los problemas que descubrieron antes de la junta. Cuando se descubren problemas o errores válidos, el registrador los va anotando.

g) Al final de la revisión, todos los participantes en la RTF deben decidir si:
   - Aceptan el producto sin posteriores modificaciones;
   - Rechazan el producto debido a los serios errores encontrados (una vez corregidos, ha de hacerse otra revisión).
   - Aceptan el producto provisionalmente (se han encontrado errores menores que deben ser corregidos, pero sin necesidad de otra posterior revisión).

h) Una vez tomada la decisión, todos los asistentes llenan un documento de finalización en el que indican su participación en la RTF y su conformidad con los hallazgos y conclusiones del equipo de revisión.

Informe de la revisión y conservación de registros

En base a las anotaciones del Registrador se genera, al finalizar la junta, una Lista de problemas de revisión y se llena un Informe Resumen de la RTF.

El Informe Resumen de la RTF consta de una sola página (con posibles anexos) que responde a tres preguntas:

1. ¿Qué fue revisado?
2. ¿Quién lo revisó?
3. ¿Cuáles fueron los hallazgos y conclusiones?

Este se adjunta al registro histórico del proyecto y puede ser enviada al jefe del proyecto y a otras partes interesadas.

La Lista de problemas de revisión sirve para dos propósitos:(1) identificar áreas problemáticas dentro del producto y (2) servir como lista de comprobación de puntos de acción que guíe al productor para hacer las correcciones. Normalmente se adjunta al informe resumen.

Es importante establecer un procedimiento de seguimiento que asegure los problemas son corregidos adecuadamente. Un enfoque consiste en asignar la responsabilidad del seguimiento al jefe de revisión.

Directrices de la revisión

Se deben establecer de antemano directrices para conducir las revisiones técnicas formales, distribuyéndolas después entre los revisores para ser consensuadas y,finalmente, seguidas.

Conjunto mínimo de directrices para las revisiones técnicas formales:

- *Revisar el producto, no al productor:* Una RTF involucra gente y egos. Conducida adecuadamente, todos los participantes deben tener

un sentimiento agradable de estar cumpliendo su deber. Se deben señalar los errores educadamente; el tono de la reunión debe ser distendido y constructivo.

- *Establecer una agenda y respetarla*: El jefe de revisión tiene la responsabilidad de mantener el plan de la reunión y no vacilar en llamar la atención de la gente cuando se empiece a divagar.
- *Limitar el debate y la impugnación:* cuando no existe consenso sobre el impacto de un problema planteado por un revisor, este debe ser registrado y dejado su tratamiento para después.
- *Enunciar áreas de problemas, sin intentar resolver todos los que se hayan señalado:* una revisión no es una sesión para resolver problemas, esto debe posponerse para después de la junta de revisión.
- *Tomar notas:* es bueno que el registrador tome notas en un pizarrón, de modo que las palabras y prioridades puedan ser evaluadas por los otros revisores mientras se registra la información.
- *Limitar el número de participantes e insistir en la preparación anticipada:* el número de miembros del equipo de revisión debe ser el mínimo necesario y estos deben prepararse por anticipado. El jefe de revisión debe solicitar comentarios escritos.
- *Desarrollar una lista de verificación para cada producto que tenga la probabilidad de ser revisado:* esta ayuda al jefe de revisión a estructurar la junta y a los revisores a enfocarse en los problemas importantes.
- *Asignar recursos y programar las RTF: las revisiones son efectivas si se programan como parte del proceso de desarrollo, incluyendo el tiempo necesario para realizar las modificaciones que resultan de la RTF.*
- *Realizar un entrenamiento significativo de los revisores: el entrenamiento debe ser formal y enfocarse tanto a los problemas del proceso, como al lado psicológico y humano de as revisiones.*
- *Analizar las revisiones previas:* para descubrir problemas en el proceso de revisión mismo

Ya que son muchas las variables involucradas que inciden en una revisión provechosa, cada organización debe experimentar para determinar que enfoque funciona mejor en un contexto local.

Revisiones basadas en muestras

Las revisiones implican recursos y tiempo que son limitados en el proyecto, de manera tal que si el proyecto se encuentra retrasado suele disminuirse el tiempo dedicado a la revisión, lo que conduce a una falta de calidad. Una solución es utilizar *revisiones basadas en muestras.*

Este proceso de revisión se basa en tomar muestras de todos los productos de trabajo con el objeto de cuantificar aquellos que sean más proclives al error y concentrar los recursos disponibles en revisar dichos productos.

Pasos:

1. Inspeccionar una fracción *a1* de cada producto de trabajo de software *i*. Registrar el nº de fallas *fi* encontradas dentro de *ai*.
2. Desarrollar una estimación bruta del nº de fallas del producto calculando: *fi*1/ai*.
3. Ordenar los productos de trabajo en forma descendente de acuerdo con la estimación bruta del número de fallas en cada uno.
4. Enfocar los recursos de revisión disponibles en aquellos productos de trabajo con el mayor nº estimado de fallas.

La fracción con la que se hizo el muestreo debe ser representativa del producto de trabajo como un todo y ser lo suficientemente grande de tal manera que sea significativa para los revisores que realicen el muestreo.

Garantía de la Calidad Estadística del Software

La garantía de la calidad estadística refleja una tendencia creciente en la industria de adoptar un enfoque más cuantitativo acerca de la calidad. Implica los siguientes pasos:

1. Se agrupa y se clasifica la información sobre los defectos del software.
2. Se intenta encontrar la causa subyacente de cada defecto. Por ejemplo:

- Especificación incompleta o errónea (EIE).
- Mala interpretación de la comunicación del cliente (MCC).
- Desviación deliberada de la especificación (DDE).
- Incumplimiento de los estándares de programación (IEP).
- Error en la representación de los datos (ERD).
- Interfaz de componente inconsistente (IMI).
- Error en la lógica de diseño (ELD).
- Prueba incompleta o errónea (PIE).
- Documentación imprecisa o incompleta (DII).
- Error en la traducción del diseño al lenguaje de programación. (TLP)
- Interfaz hombre-computadora ambigua o inconsistente (IHC)

3. Mediante el principio de Pareto (el 80 % de los defectos se pueden encontrar en el 20% de todas las posibles causas), se aísla el 20 % (los vitales). Ejemplo: en la siguiente tabla observamos que EIE, MCC y ERD son las causas vitales que explican el 53 % de todos los defectos para un proyecto xxx.

	Total		Grave		Moderado		Leve	
Error	No.	%	No.	%	No.	%	No.	%
IEE	205	22%	34	27%	68	18%	103	24%
MCC	156	17%	12	9%	68	18%	76	17%
DDE	48	5%	1	1%	24	6%	23	5%
IEP	25	3%	0	0%	15	4%	10	2%
ERD	130	14%	26	20%	68	18%	36	8%
IME	58	6%	9	7%	18	5%	31	7%
ELD	45	5%	14	11%	12	3%	19	4%
PIE	95	10%	12	9%	35	9%	48	11%
DII	36	4%	2	2%	20	5%	14	3%
TLP	60	6%	15	12%	19	5%	26	6%
IHM	28	3%	3	2%	17	4%	8	2%
VAR	56	6%	0	0%	15	4%	41	9%
Totales	942	100%	128	100%	379	100%	435	100%

TABLA 8.1. Recolección de datos para la SQA estadística

4. Una vez que se han identificado los defectos vitales, se actúa para corregir los problemas que han producido los defectos. Conforme sus causas se corrigen, nuevas causas ocuparán la parte superior de la clasificación y deberán ser corregidas. Para el ejemplo anterior, para corregir MCC, el desarrollador de software puede implementar técnicas que faciliten la recopilación de requisitos para mejorar la calidad de la comunicación y las especificaciones del cliente.

Seis sigma

El término “seis sigma” se deriva de seis desviaciones estándar- 3.4 instancias (defectos) por millón de ocurrencias-.

La estrategia seis sigma es una metodología rigurosa y disciplinada que utiliza análisis de datos y estadístico para medir y mejorar el desempeño operativo de una compañía al identificar y eliminar defectos en la fabricación y los procesos relacionados con el servicio.

Los pasos que propone el método DMAMC (definir, medir, analizar, mejorar y controlar) son:

CENTRALES

1. Definir los requisitos del cliente, entregables y metas del proyecto por medio de métodos bien definidos de comunicación con el cliente.

2. Medir el proceso existente y su salida para determinar el desempeño de calidad actual (recopilación de métricas de defecto)
3. Analizar las métricas de defecto y determinar las causas poco vitales.

ADICIONALES (si el proceso existente está en marcha y requiere mejoría)

4. Mejorar el proceso eliminando las causas originales de los defectos.
5. Controlar el proceso para garantizar que el trabajo futuro no vuelva a introducir las causas de defectos.

Una variación del método anterior llamada DMADV añade dos pasos centrales para cuando se está desarrollando un proceso de software en lugar de mejorarlo:

- Diseñar el proceso para evitar las causas originales de defectos y satisfacer los requerimientos del cliente.
- Verificar que el modelo de proceso, de hecho, evitará los defectos y satisfará al cliente.

Fiabilidad del Software

La fiabilidad del software, a diferencia de otros factores de calidad, se puede medir, dirigir y estimar mediante datos históricos o de desarrollo.
La fiabilidad del software se define en términos estadísticos como la probabilidad de operación libre de fallas de un programa de computadora en un entorno determinado y durante un tiempo específico.
Una falla es cualquier falta de concordancia con los requisitos del software.

Medidas de fiabilidad y disponibilidad

Todavía se debate sobre la relación entre los conceptos de la fiabilidad del hardware y su aplicabilidad al software, pues la mayoría de los modelos de fiabilidad relativos al hardware están más orientados a fallas debido al uso que a fallas debidas a defectos de diseño, mientras que todas las fallas de software se originan por problemas de diseño. Sin embargo existen algunos conceptos que se aplican a ambos sistemas:

Considerando un sistema basado en computadora, una sencilla medida de la fiabilidad es el tiempo medio entre fallas (TMEF), donde;
TMEF = TMDF + TMDR
Las siglas TMDF y TMDR corresponden a tiempo medio de fallo y tiempo medio de reparación, respectivamente.

Otra medida es la de Disponibilidad, que es la probabilidad de que un programa opere de acuerdo con los requisitos en un momento dado, y se define como:
Disponibilidad = [TMDF/ (TMDF + TMDR)] x 100 %

La medida de fiabilidad TMEF es igualmente sensible al TMDF que al TMDR, mientras que la medida de disponibilidad es algo más sensible al TMDR, una medida indirecta de la facilidad de mantenimiento del SW.

Seguridad del software
Este enfoque es similar al análisis de riesgos, siendo la principal diferencia su énfasis en los conflictos tecnológicos más que en los tópicos relacionados con el proyecto.

La seguridad del software es una actividad de garantía de calidad del software que se centra en la identificación y evaluación de los riesgos potenciales que pueden producir un impacto negativo en el software y hacer que falle el sistema completo. Si se pueden identificar pronto los riesgos en el proceso de ingeniería del software podrán especificarse las características del diseño del software que permitan eliminar o controlar los riesgos potenciales.
Inicialmente, se identifican los riesgos y se clasifican por su importancia y su grado de riesgo.
Cuando se han identificado estos riesgos del sistema, se utilizan técnicas de análisis (como el análisis del árbol de fallos, la lógica de tiempo real o los modelos de redes de Petri) para asignar su gravedad y su probabilidad de ocurrencia. Para que sea efectivo, se tiene que analizar el software en el contexto del sistema completo.
Una vez que se han identificado y analizado los riesgos, se pueden especificar los requisitos relacionados con la seguridad para el software a través de una lista de eventos no deseables y las respuestas del sistema a estos eventos.

Aunque la fiabilidad y la seguridad del software están bastante relacionadas, es importante entender la sutil diferencia que existe entre ellas. La fiabilidad del software utiliza el análisis estadístico para determinar la probabilidad de que pueda ocurrir un fallo del software.
Sin embargo, la ocurrencia de un fallo no lleva necesariamente a un riesgo o a un accidente. La seguridad del software examina los modos según los cuales los fallos producen condiciones que pueden llevar a accidentes. Es decir, los fallos se evalúan en el contexto de un completo sistema basado en computadora.

Estándares de Calidad ISO 900

Un sistema de Garantía de Calidad se puede definir como la estructura organizacional, responsabilidades, procedimientos, procesos y recursos para implementar la gestión de la calidad.
Para obtener el registro ISO 9001:2000, una organización de software debe establecer políticas y procedimientos para satisfacer los 20 requisitos que define el estándar para un sistema eficiente de garantía de la calidad, y además demostrar que se siguen dichas políticas y procedimientos, para lo cual la organización es sometida a auditorias periódicas.
Como el estándar es aplicable a todas las disciplinas de ingeniería, se ha desarrollados un conjunto de directrices ISO 9000-3 que ayudan interpretar el estándar para emplearlo en el proceso de SW.

Plan SQA

El plan de SQA es desarrollado por un grupo de SQA (o por el equipo de desarrollo sino existe grupo SQA) y sirve como plantilla para las actividades de SQA instituidas para cada proyecto de software.
El IEEE ha recomendado un estándar para los planes de SQA cuya estructura identifica:

1. El propósito y el alcance del documento
2. Una descripción de todos los productos de trabajo de ingeniería del software.
3. Todos los estándares y prácticas aplicables que se aprovechan durante el proceso de software.
4. Acciones y tareas de SQA y su ubicación a través del proceso de software.
5. Herramientas y métodos que soportan las acciones y tareas de SQA.
6. Procedimientos de gestión de configuración de software para gestionar el cambio.
7. Métodos para ensamblar, salvaguardar y mantener todos los registros relacionados con el SQA.
8. Papeles y responsabilidades en la organización relativas a la calidad de producto.

RESUMEN

La garantía de calidad del software es una actividad protectora- que incluye tanto control como aseguramiento de la calidad- que se aplica a cada paso del proceso del software. La SQA comprende procedimientos para la aplicación eficaz de métodos y herramientas, revisiones técnicas formales, técnicas y estrategias de prueba, procedimientos de control de cambios, procedimientos para garantizar la concordancia con los estándares y mecanismos de medición y reporte.
La SQA es complicada por la compleja naturaleza de la calidad del software -un atributo de los programas de computadora que se define como «concordancia con los requisitos definidos explícita e implícitamente»-.Pero cuando se considera de forma más general, la calidad del software engloba muchos factores de proceso y de producto diferentes con sus métricas asociadas.
Las revisiones del software son una de las actividades más importantes de la SQA. Las revisiones sirven como filtros durante todas las actividades de ingeniería del software, eliminando defectos mientras que son relativamente poco costosos de encontrar y corregir. La revisión técnica formal es una revisión específica (junta) que se ha mostrado extremadamente efectiva en el descubrimiento de errores.
Para llevar a cabo adecuadamente una garantía de calidad del software, se deben recopilar, evaluar y distribuir todos los datos relacionados con el proceso de ingeniería del software. La SQA estadística ayuda a mejorar la calidad del producto y el proceso de software en si mismo. Los modelos de fiabilidad del software amplían las medidas, permitiendo extrapolar los datos recogidos sobre los defectos, a predicciones de tasas de fallo y de fiabilidad.
Resumiendo, recordemos las palabras de Dunn y Ullman]: “El aseguramiento de la calidad del software es el mapeo o guía de los preceptos de gestión y las

disciplinas de diseño de la garantía de calidad en el espacio gerencial y tecnológico aplicable a la ingeniería del software". La capacidad para garantizar la calidad es la medida de una disciplina de ingeniería madura. Cuando se sigue de forma adecuada esa guía anteriormente mencionada, lo que se obtiene la madurez de la ingeniería del software.