Universidad de Buenos Aires
Facultad de Ciencias Económicas

El Proceso de Quality Assurance

AR Tema extractado del libro "**Análisis Funcional de Sistemas y Tecnologías de la Información**" de Aníbal M. Mazza Fraquelli - ISBN 978-987-26981-3-3

---

1. Presentación del Tema

El Proceso de QA (Quality Assurance – Aseguramiento de Calidad) en el desarrollo de software constituye un conjunto estructurado de actividades orientadas a garantizar que un sistema informático cumpla con los requisitos funcionales y no funcionales definidos por la organización. En el contexto de las Tecnologías de la Información (TI), QA no es una etapa aislada, sino un proceso transversal que acompaña todo el ciclo de vida del desarrollo.

Desde la perspectiva administrativa, el proceso de QA es un mecanismo formal de control preventivo que permite reducir riesgos, optimizar recursos, mejorar la eficiencia operativa y asegurar la alineación entre soluciones tecnológicas y objetivos estratégicos del negocio.

En organizaciones digitalizadas, donde los sistemas de información sostienen procesos críticos —finanzas, logística, recursos humanos, comercio electrónico, gestión documental—, un proceso de QA estructurado impacta directamente en:

- Continuidad operativa.
- Satisfacción del cliente.

- Cumplimiento normativo.
- Gestión de riesgos.
- Eficiencia presupuestaria.
- Gobernanza de TI.

El proceso típico de QA en el desarrollo de software incluye seis etapas fundamentales:

1. Planificación de QA.
2. Definición de Requisitos.
3. Diseño de Casos de Prueba.
4. Ejecución de Pruebas.
5. Revisión y Análisis de Resultados.
6. Cierre de QA.

Cada una de estas etapas cumple una función específica dentro del ecosistema tecnológico y organizacional.

---

2. Desarrollo

2.1 Planificación de QA

La planificación de QA es la etapa inicial en la que se definen las estrategias, objetivos, alcance y enfoques para garantizar la calidad del software.

En esta fase se establecen:

- Estrategia de pruebas.
- Alcance del aseguramiento.
- Criterios de aceptación.
- Métricas de calidad.
- Recursos necesarios.
- Cronograma de actividades.
- Herramientas a utilizar.

Desde la perspectiva de TI, la planificación debe considerar:

- Arquitectura del sistema.
- Integraciones con otros sistemas.
- Riesgos técnicos.
- Requisitos regulatorios.
- Impacto en infraestructura.

Conceptos relevantes:

- Test Plan (Plan de Pruebas).
- Risk-Based Testing (Pruebas Basadas en Riesgo).
- Service Level Agreement – SLA (Acuerdo de Nivel de Servicio).
- Key Performance Indicators – KPI (Indicadores Clave de Desempeño).

Ejemplo:

En el desarrollo de un sistema bancario digital, la planificación de QA deberá priorizar:

- Seguridad.
- Rendimiento.
- Integridad de datos.
- Cumplimiento normativo.

Desde la administración, esta etapa permite asignar recursos de forma racional y anticipar costos asociados a la calidad.

2.2 Definición de Requisitos

La calidad del software depende directamente de la calidad de los requisitos. Si los requisitos son ambiguos o incompletos, el producto final será defectuoso, aunque el código esté correctamente implementado.

En esta etapa se busca asegurar que los requisitos sean:

- Claros.
- Completos.

- Consistentes.
- Medibles.
- Verificables.

Se distinguen dos grandes tipos de requisitos:

1. Requisitos funcionales (qué debe hacer el sistema).
2. Requisitos no funcionales (cómo debe comportarse el sistema).

Requisitos no funcionales incluyen:

- Performance (rendimiento).
- Security (seguridad).
- Usability (usabilidad).
- Scalability (escalabilidad).
- Reliability (confiabilidad).

Ejemplo:

Requisito funcional:

> El sistema debe permitir registrar una orden de compra.

Requisito no funcional:

> El registro debe completarse en menos de 2 segundos.

Desde la administración, la correcta definición de requisitos reduce:

- Retrabajos.
- Desviaciones presupuestarias.
- Conflictos contractuales.
- Riesgos de incumplimiento.

El QA participa activamente validando la coherencia y verificabilidad de los requisitos antes de que el desarrollo avance.

2.3 Diseño de Casos de Prueba

El diseño de casos de prueba consiste en crear escenarios estructurados que permitan verificar si el software cumple con los requisitos definidos.

Un caso de prueba incluye:

- Identificador.
- Descripción.
- Precondiciones.
- Pasos de ejecución.
- Datos de entrada.
- Resultado esperado.

Conceptos asociados:

- Test Case (Caso de Prueba).
- Test Scenario (Escenario de Prueba).
- Test Suite (Conjunto de Pruebas).
- Traceability Matrix (Matriz de Trazabilidad).

La matriz de trazabilidad permite vincular cada requisito con uno o más casos de prueba, garantizando cobertura completa.

Ejemplo:

Requisito:

> El sistema debe calcular correctamente el IVA.

Caso de prueba:

- Ingresar monto base.
- Aplicar tasa correspondiente.
- Validar resultado final.

Desde la mirada administrativa, el diseño de casos de prueba permite:

- Medir cobertura.
- Estimar esfuerzo.

- Controlar cumplimiento contractual.
- Asegurar transparencia en auditorías.

2.4 Ejecución de Pruebas

La ejecución de pruebas es la fase operativa donde se aplican los casos diseñados.

Puede incluir:

- Pruebas manuales.
- Pruebas automatizadas.
- Pruebas de integración.
- Pruebas de regresión.
- Pruebas de carga.
- Pruebas de seguridad.
- User Acceptance Testing (UAT – Pruebas de Aceptación de Usuario).

Automatización:

El uso de herramientas de automatización permite:

- Reducir tiempo.
- Aumentar precisión.
- Repetir pruebas consistentemente.
- Integrarse con Continuous Integration (CI – Integración Continua).

Ejemplo:

En una plataforma de comercio electrónico, se ejecutan pruebas de carga para verificar que el sistema soporte 10.000 usuarios simultáneos.

Desde la administración, esta etapa impacta en:

- Control de calidad.
- Validación contractual.
- Mitigación de riesgos.

- Cumplimiento de SLA.

2.5 Revisión y Análisis de Resultados

Una vez ejecutadas las pruebas, se analizan los resultados y se documentan los defectos.

Elementos clave:

- Defect Report (Reporte de Defectos).
- Severity (Severidad).
- Priority (Prioridad).
- Root Cause Analysis (Análisis de Causa Raíz).
- Metrics Dashboard (Panel de Métricas).

Severidad indica impacto técnico.

Prioridad indica urgencia de corrección.

Ejemplo:

Un error que impide iniciar sesión tiene alta severidad y alta prioridad.

En cambio, un error estético puede tener baja severidad y baja prioridad.

Desde la administración, esta etapa permite:

- Evaluar estado del proyecto.
- Analizar tendencias de defectos.
- Tomar decisiones sobre liberación.
- Ajustar recursos.

El análisis estructurado evita decisiones basadas en intuición y fortalece la gobernanza de TI.

2.6 Cierre de QA

El cierre de QA implica validar que:

- Todos los defectos críticos fueron resueltos.
- Se cumplió la cobertura planificada.

- Se alcanzaron los criterios de aceptación.
- Se documentaron resultados finales.
- Se emitió informe de calidad.

En esta fase se define si el software está listo para:

- Deploy (Despliegue).
- Release (Liberación).
- Go-Live (Puesta en Producción).

Desde la perspectiva organizacional, el cierre de QA reduce riesgos asociados al lanzamiento.

Ejemplo:

Antes de liberar un sistema de gestión contable, QA verifica que:

- No existan defectos críticos abiertos.
- Se hayan ejecutado pruebas de regresión.
- Se hayan validado cálculos financieros.

El cierre formal protege a la organización frente a fallas en producción.

2.7 Integración del Proceso de QA con la Gestión Organizacional

El proceso de QA se integra con:

- Gestión de proyectos.
- Gestión de riesgos.
- Gestión de cambios.
- Auditoría interna.
- Gobierno corporativo.

Indicadores relevantes:

- Defect Density (Densidad de Defectos).
- Test Coverage (Cobertura de Pruebas).

- Defect Leakage (Fugas de Defectos).
- MTTR (Mean Time To Repair – Tiempo Medio de Resolución).

Desde la administración, estos indicadores permiten:

- Evaluar desempeño del equipo.
- Tomar decisiones estratégicas.
- Optimizar presupuesto.
- Mejorar procesos.

QA no es un proceso aislado; es parte del sistema de control organizacional.

---

3. Conclusión

El proceso de QA en el desarrollo de software constituye un marco estructurado y sistemático que garantiza la calidad tecnológica desde la planificación hasta la liberación del producto.

Las etapas de planificación, definición de requisitos, diseño de casos de prueba, ejecución, análisis y cierre no solo aseguran calidad técnica, sino que protegen a la organización frente a riesgos operativos, financieros y reputacionales.

Desde la perspectiva de Tecnologías de la Información y de la administración, QA representa:

- Un mecanismo preventivo.
- Un sistema de control interno.
- Un instrumento de gestión de riesgos.
- Un soporte a la gobernanza de TI.
- Un habilitador de mejora continua.

En organizaciones donde los sistemas de información son críticos, el proceso de QA no es una opción operativa, sino una condición estratégica para la sostenibilidad y competitividad empresarial.

Preguntas de autoevaluación

1. Explique por qué la planificación de QA es fundamental para la gestión eficiente de recursos en un proyecto de software.
2. ¿Cómo contribuye la definición clara de requisitos a la reducción de costos?
3. Analice la importancia de la matriz de trazabilidad dentro del proceso de QA.
4. ¿Qué indicadores permiten evaluar la efectividad del proceso de QA?
5. Desde la perspectiva administrativa, ¿por qué el cierre formal de QA reduce riesgos organizacionales?

Material de Clases

Compilado por **Aníbal M. Mazza Fraquelli** Doctor de la Universidad de Buenos Aires para el uso de sus clases en la Facultad de Ciencias Económicas de la Universidad de Buenos Aires.

Contenidos de esta página

Los contenidos **aquí incluidos integran desarrollos y escritos propios del autor, así como materiales de terceros (documentos, textos, fragmentos, conceptos, imágenes, esquemas, definiciones u otros recursos),** los cuales son utilizados a título ilustrativo, explicativo o formativo, respetando la normativa vigente en materia de derechos de autor y citando las fuentes cuando corresponde.
La selección, organización, adaptación pedagógica y contextualización de los contenidos constituye un trabajo original del autor, orientado a facilitar los procesos de enseñanza y aprendizaje.
Este material no persigue fines comerciales y su reproducción, total o parcial, queda limitada al ámbito educativo, debiendo preservarse siempre la mención de la autoría y las fuentes originales.

Autorización de uso

Se permite la reproducción, comunicación pública, distribución y utilización total o parcial de los contenidos de su material, en formato físico o digital, con fines exclusivamente educativos, académicos o de divulgación, siempre que se respete la integridad del contenido y se incluya la correspondiente referencia a la fuente y a la autoría.
Las ideas, opiniones e interpretaciones contenidas en este material corresponden exclusivamente al autor.
Queda expresamente excluido cualquier uso con fines comerciales.