ADMINISTRACIÓN
Y NEGOCIOS

inacap

# Evaluación 3 ingeniería de Software

NOMBRE: Robert Zúñiga, Benjamín Marrian, Juan Cárdenas, Christofer Paredes, Cristóbal Vargas, Cristian Mora, Bárbara Valenzuela, Vicente Rebolledo
CARRERA: Ingenieria en informatica
ASIGNATURA: Ingeniería de  Software
PROFESOR: José Alfredo Jara Vergara
FECHA: 19/11/2025

# Índice

# Introducción

El presente informe corresponde a la **Evaluación 3 de Ingeniería de Software: Prototipo** , cuyo objetivo principal es detallar y formalizar las estrategias de **Desarrollo del Proyecto y Aseguramiento de Calidad** para el Sistema de Gestión de Flota de Buses. Este documento establece los cimientos técnicos y operativos para garantizar que el sistema cumpla con los más altos estándares de calidad, seguridad y mantenibilidad.

# Etapa 1: Desarrollo del Proyecto y Aseguramiento de Calidad

**Paso 1.1. Normas, estándares y pruebas**

**1. Normas y Estándares Incorporados**

| Norma / Standard | Descripción | Motivo de Incorporación |
|---|---|---|
| **ISO/IEC 25010** | Modelo calidad de producto de software | Garantizar que el sistema cumpla con características de calidad como funcionalidad, confiabilidad, usabilidad y mantenibilidad. |
| **ISO/IEC 27001** | Gestión de la seguridad de la información. | Asegurar la confidencialidad, integridad y disponibilidad de los datos de flota, conductores y costos. |
| **IEEE 829-2008** | Estándar para documentación de pruebas de software. | Estandarizar los formatos y procesos de las pruebas para facilitar su trazabilidad y reproducibilidad |

| Norma / Standard | Descripción | Motivo de Incorporación |
|---|---|---|
| **MVC (Modelo-Vista-Controlador)** | Patrón de arquitectura de software. | Su trazabilidad y reproducibilidad, separa la lógica de negocio, la interfaz de usuario y el control de flujo, para facilitar el mantenimiento y la escalabilidad. |
| **W3C Web Standards** | Estándares web (HTML5, CSS3, JavaScript). | Garantizar compatibilidad entre navegadores y dispositivos, y mejorar la accesibilidad. |
| **NCh-ISO 8000** | Calidad de datos. | Asegurar que los datos registrados (combustible, rutas, mantención) sean |

| | | |
|---|---|---|
| | | precisos y consistentes. |

**2.- Plan de Pruebas**

| Tipo de Prueba | Objetivo | Criterios de aceptación | Ejemplo de Caso de Prueba |
|---|---|---|---|
| **Pruebas Unitarias** | Verificar ViajeForm.save en views.py asigna y guarda coordenadas de origen/destino. | Asigna lat/long desde lugar_origen/lugar_destino. Persiste cuando commit=True, No borra otros campos del Viaje. | Crear ViajeForm con lugar_origen y lugar_destino, llamar save(commit=True) y comprobar coordenadas y que instance.pk existe |
| **Pruebas de Integración** | Verificar que al crear un viaje en viajes/models.py se pueda asociar un CostosViaje de costos/models.py y que CostosViaje.save() calcule correctamente el costo total. | Viaje crea instancias referenciando Bus (flota/models.py), Conductor y Lugar (core/models.py).<br><br>CostosViaje se asocia mediante OneToOneField con viaje.costos.<br><br>CostosViaje.save() calcula costo_total automáticamente (combustible + mantenimiento + peajes + otros_costos).<br><br>Eliminar Viaje elimina su CostosViaje en cascada. | Crear Viaje con bus/conductor/lugares válidos, crear CostosViaje asociado, llamar save() y comprobar que costo_total y que viaje.costos existe; al eliminar el viaje verificar que también se elimina su CostosViaje. |
| **Pruebas de Sistema** | Validar el comportamiento completo del sistema. | Cumplimiento de todos los requerimientos funcionales y no funcionales. | Flujo completo: Login → Crear Viaje → Generar Reporte. |
| **Pruebas de Usabilidad** | Evaluar la facilidad de uso por parte de conductores y administradores. | Curva de aprendizaje < 2 horas. | Conductor completa registro de viaje en < 3 min. |
| **Pruebas de Seguridad** | Verificar autenticación, | Acceso denegado sin credenciales válidas | Intentar acceder a reportes sin rol de |

| Tipo de Prueba | Objetivo | Criterios de aceptación | Ejemplo de Caso de Prueba |
|---|---|---|---|
| **Pruebas Unitarias** | Verificar ViajeForm.save en views.py asigna y guarda coordenadas de origen/destino. | Asigna lat/long desde lugar_origen/lugar_destino. Persiste cuando commit=True, No borra otros campos del Viaje. | Crear ViajeForm con lugar_origen y lugar_destino, llamar save(commit=True) y comprobar coordenadas y que instance.pk existe |
| **Pruebas de Integración** | Verificar que al crear un viaje en viajes/models.py se pueda asociar un CostosViaje de costos/models.py y que CostosViaje.save() calcule correctamente el costo total. | Viaje crea instancias referenciando Bus (flota/models.py), Conductor y Lugar (core/models.py).<br><br>CostosViaje se asocia mediante OneToOneField con viaje.costos.<br><br>CostosViaje.save() calcula costo_total automáticamente (combustible + mantenimiento + peajes + otros_costos).<br><br>Eliminar Viaje elimina su CostosViaje en cascada. | Crear Viaje con bus/conductor/lugares válidos, crear CostosViaje asociado, llamar save() y comprobar que costo_total y que viaje.costos existe; al eliminar el viaje verificar que también se elimina su CostosViaje. |
| **Pruebas de Sistema** | Validar el comportamiento completo del sistema. | Cumplimiento de todos los requerimientos funcionales y no funcionales. | Flujo completo: Login → Crear Viaje → Generar Reporte. |
| **Pruebas de Usabilidad** | Evaluar la facilidad de uso por parte de conductores y administradores. | Curva de aprendizaje < 2 horas. | Conductor completa registro de viaje en < 3 min. |
| | autorización y cifrado. | | administrador. |
| **Pruebas de Rendimiento** | Evaluar tiempos de respuesta bajo carga. | Respuesta < 3 segundos con 20 vehículos simultáneos. | Simular 20 viajes activos al mismo tiempo. |

### 1.2 Plan de implementación

#### A) Gestión de disponibilidad

**Objetivo:** 98% de disponibilidad mensual.

**Estrategias:**
Servidores en cluster con balanceo de carga.
Monitoreo continuo con herramientas como Nagios o Zabbix.
Respaldos automáticos diarios de la base de datos.
Mantenimiento programado en horarios de baja demanda (ej: domingos 02:00 - 04:00 AM).

#### B) Gestión de continuidad

**Procedimientos Proactivos:**

Acciones preventivas basadas en los puntos definidos en el informe.

**Réplica de base de datos en sitio secundario:**

- **Cómo:** Configuración de replicación **MySQL tipo Master-Slave**, asegurando que los datos del sistema de buses (tablas de viajes, mantenciones, choferes) se copien automáticamente.
- **Dónde:** En una instancia secundaria de MySQL Server ubicada en un servidor de respaldo, separada del servidor de producción.
- **Cuándo:** Sincronización continua (tiempo real) para asegurar que el "Slave" tenga siempre los últimos registros de venta de pasajes o encomiendas.

**Capacitación a personal en procedimientos de contingencia:**

- **Cómo:** Simulacros de despliegue del proyecto Django y activación del entorno virtual (`source venv/bin/activate`) en el servidor de respaldo.
- **Dónde:** En el entorno de pruebas (Staging) que replica la configuración del servidor de producción.
- **Cuándo:** Trimestralmente, o cada vez que se actualice la versión de Django o librerías críticas en el `requirements.txt`.

**Revisiones periódicas de infraestructura:**

- **Cómo:** Monitoreo de los servicios clave (servicio MySQL, servidor Gunicorn/Uvicorn) y espacio en disco disponible para logs. Uso de herramientas como Nagios o Zabbix.
- **Dónde:** Dashboard de monitoreo centralizado y revisión de logs de errores de Django (`django.request`).

- **Cuándo:** Revisión automatizada 24/7 con alertas, y auditoría manual de logs de acceso cada lunes a las 09:00 AM.

**Procedimientos Reactivos:**

**Acciones correctivas basadas en los puntos definidos en el informe.**

- **Plan de recuperación ante desastres (DRP) con restauración de respaldos:**
  - **Cómo: Ejecución de scripts de restauración de MySQL (`mysql < backup.sql`) y redespliegue del contenedor o servicio Django si los archivos estáticos o de media (fotos de buses) se corrompieron.**
  - **Dónde: En el servidor de contingencia designado en el plan de arquitectura.**
  - **Cuándo: Inmediatamente tras confirmar que el servidor principal no tiene reparación rápida (tiempo de caída > 15 minutos).**
- **Comunicación inmediata a usuarios vía correo o SMS:**
  - **Cómo: Disparo de notificaciones masivas a la lista de correos de "Administradores de Flota" y "Conductores" informando la indisponibilidad.**
  - **Dónde: A través de una plataforma externa de mailing (para no depender del servidor caído) o servicio de SMS gateway configurado.**
  - **Cuándo: Máximo 15 minutos después de detectada la incidencia crítica.**
- **Escalación automática de incidentes al equipo técnico:**
  - **Cómo: Registro automático de ticket con etiqueta "CRITICAL" en Jira o Zendesk cuando el health-check del servidor Django responde con error 500 o timeout.**
  - **Dónde: Plataforma de gestión de tickets del equipo de desarrollo.**
  - **Cuándo: Al instante (tiempo real) mediante integración con las herramientas de monitoreo.**

### 1.3 Plan de Mantención

#### a. Gestión de Configuración

Se utilizará un Repositorio Git (GitLab o GitHub) para control de versiones.
Cada componente (código, BD, documentación) tendrá un identific único (ej: `v1.2.3`).
Se documentarán todas las configuraciones de servidores y servicios en un registro de configuración.

**b. Gestión de Cambios**

Los cambios se solicitarán mediante RFC (Request for Change).

Serán evaluados por un Comité de Cambio (jefe de proyecto, representante del cliente).

Se priorizará según impacto y urgencia.

Cambios críticos requieren respaldo previo y ventana de mantenimiento.

**c. Gestión de Incidentes**

Los incidentes se registran en un sistema de tickets (ej: Jira, Zendesk).
Clasificación por prioridad:
Crítico: Sistema no funciona > Resolución en < 4 horas.
Alto: Funcionalidad clave afectada > Resolución en < 8 horas.
Medio: Problema no impide operación > Resolución en < 48 horas.
Se realizará un informe mensual de incidentes para identificar tendencias.

## 1.4 Plan de auditoría

**ALCANCE DE LA AUDITORÍA**

Sistema de Gestión de Flota de Buses - Módulos: Gestión de Viajes, Control de Combustible, Mantenimiento Vehicular, Reportes Financieros

| Norma ISO | Aplicación Específica en el Proyecto | Frecuencia Auditoría |
|---|---|---|
| ISO 27001 | Sistema de Gestión de Seguridad de la Información (SGSI) | Anual (Externa) |
| ISO 27002 | Controles de seguridad para datos de conductores y flota | Trimestral (Interna) |
| ISO 27005 | Gestión de riesgos de seguridad de información | Semestral |
| ISO 27017 | Controles de seguridad para cloud computing (si aplica) | Trimestral |
| ISO 27031 | Preparación para continuidad del negocio | Semestral |
| ISO 25010 | Calidad del producto software | Mensual |

**CRONOGRAMA DETALLADO DE AUDITORÍAS**

**AUDITORÍAS MENSUALES**

Enfoque: Calidad del Software y Controles Básicos de Seguridad

Checklist Mensual:
Verificación de integridad de respaldos automáticos
Revisión de logs de acceso al sistema
Validación de actualizaciones de seguridad aplicadas
Verificación de cumplimiento de tiempos de respuesta (< 3 segundos)
Revisión de incidentes de seguridad del mes
Validación de cifrado de datos en tránsito (TLS 1.2+)

**AUDITORÍAS TRIMESTRALES**

Enfoque: ISO 27002 - Controles de Seguridad Específicos

Áreas a Auditar (por trimestre):

Q1 - Controles de Acceso (ISO 27002: A.9)
Revisión de políticas de autenticación por roles
Verificación de revocación de acceso de ex-empleados
Validación de fuerza de contraseñas
Auditoría de asignación de privilegios

Q2 - Criptografía (ISO 27002: A.10)
Verificación de cifrado de base de datos
Validación de certificados SSL/TLS
Revisión de políticas de cifrado de respaldos
Auditoría de gestión de claves criptográficas

Q3 - Seguridad Operacional (ISO 27002: A.12)
Revisión de procedimientos de respaldo
Validación de protección contra malware
Verificación de logs de eventos de seguridad
Auditoría de gestión de vulnerabilidades

Q4 - Aspectos Legales (ISO 27002: A.18)
Cumplimiento Ley 19.628 Protección Datos Personales
Cumplimiento Ley 20.584 Derechos Usuarios
Revisión de contratos con proveedores
Verificación de políticas de privacidad

**AUDITORÍAS SEMESTRALES**

Enfoque: ISO 27005 - Gestión de Riesgos y Continuidad

Actividades:
Análisis de riesgos de seguridad actualizado
Revisión del plan de continuidad del negocio
Simulación de incidentes de seguridad
Evaluación de controles de recuperación ante desastres

**AUDITORÍA ANUAL**

Enfoque: ISO 27001 - Certificación del SGSI

Actividades:
Auditoría completa por entidad certificadora externa
Revisión de toda la documentación del SGSI
Evaluación de efectividad de todos los controles
Renovación del certificado ISO 27001

## SEGURIDAD FÍSICA Y AMBIENTAL (ISO 27002: A.11)

Control de acceso a servidores locales
Protección contra incendios e inundaciones
Registro de entradas/salidas del centro de datos
Monitoreo ambiental (temperatura, humedad)

## SEGURIDAD DE RECURSOS HUMANOS (ISO 27002: A.7)

Acuerdos de confidencialidad firmados
Capacitación en seguridad para nuevos empleados
Procedimientos de terminación de acceso
Concientización periódica en seguridad

## GESTIÓN DE INCIDENTES (ISO 27002: A.16)

Procedimiento documentado para reportar incidentes
Equipo de respuesta a incidentes designado
Registro y análisis de todos los incidentes
Mejoras implementadas basadas en lecciones aprendidas

## GESTIÓN DE ACTIVOS (ISO 27002: A.8)

Inventario actualizado de todos los activos de información
Clasificación de datos (públicos, internos, confidenciales)
Etiquetado adecuado de información sensible
Políticas de manejo de medios de almacenamiento

**METRÍAS Y KPI'S DE AUDITORÍA**

| Metrica | Objetivo | Frecuencia Medición |
|---|---|---|
| Tiempo de respuesta a incidentes | < 4 horas para críticos | Mensual |
| Cobertura de respaldos | 100% de datos críticos | Semanal |
| Cumplimiento de políticas | > 95% | Mensual |
| Vulnerabilidades críticas resueltas | < 7 días | Mensual |
| Tasa de éxito en autenticación | > 99% | Mensual |

## DOCUMENTACIÓN REQUERIDA

Política de Seguridad de la Información:**Gestión de Seguridad de la Información (SGSI)**, y requisito clave de **ISO 27001**

Procedimiento de Gestión de Incidentes:**Gestión de Incidentes** (ISO 27002: A.16) y la clasificación de incidentes por prioridad en la **Gestión de Incidentes** interna.

Plan de Continuidad del Negocio: I**SO 27031**

Inventario de Activos de Información: **Gestión de Activos** (ISO 27002: A.8)

Registros de Auditorías anteriores: **ISO 27001, ISO 27002** e **ISO 25010**

Informes de No Conformidades y Acciones Correctivas: **Auditoría Anual ISO 27001** y las revisiones de **ISO 25010 y el ciclo de Mejora Continua (PDCA).**

## RESPONSABLES DE AUDITORÍA

Auditor Líder: Jefe de Proyecto TI
Auditor de Seguridad: Especialista en Ciberseguridad
Auditor de Calidad: Líder de Aseguramiento de Calidad
Auditores Externos: Empresa certificadora ISO 27001

**1.5 Mejora continua**

Se adoptará el modelo de mejora continua de ITIL, basado en el ciclo PDCA (Plan-Do-Check-Act):

**Planificar**:Identificar áreas de mejora a partir de reportes de incidentes, feedback de usuarios y métricas de rendimiento.

**Ejecutar**:Implementar mejoras en versiones menores (ej: v1.1, v1.2).

**Verificar**:Medir el impacto de las mejoras mediante KPIs (ej: tiempo de respuesta, satisfacción de usuarios).

**Actuar**:Si la mejora es exitosa, se integra al sistema de forma permanente. Si no, se ajusta y reevalúa.

**Ciclos de Mejora Continua**

Mejoras Mensuales:

Revisión de métricas de rendimiento (tiempos de respuesta, disponibilidad).

Análisis de incidentes y propuestas de corrección.

Actualizaciones menores de seguridad y usabilidad.

Mejoras Trimestrales:

Análisis de feedback de usuarios (conductores, administradores).

Revisión de reportes de auditoría interna.

Implementación de nuevas funcionalidades menores.

Mejoras Semestrales:

Evaluación de cumplimiento de objetivos de calidad (ISO 25010).

Revisión de arquitectura y escalabilidad.

Actualización de estándares de seguridad (ISO 27002).

**Ejemplo de Mejora Continua Mensual**

| Mes | Actividad de Mejora | Responsable |
|---|---|---|
| Enero | Optimización de consultas de base de datos para reportes | Equipo Desarrollo |
| Febrero | Rediseño de interfaz móvil basado en feedback de conductores | UX/UI |
| Marzo | Actualización de políticas de cifrado y respaldo | Seguridad TI |

# Conclusión

El plan de aseguramiento de calidad y gestión operativa desarrollado en este informe establece un marco robusto y completo para el ciclo de vida del Sistema de Gestión de Flota de Buses. La integración de estándares como **ISO/IEC 25010** y **W3C Web Standards** garantiza que el software no solo sea funcional, sino también confiable, usable y compatible. La adopción de un **Plan de Pruebas** que incluye cobertura de código superior al 80% y criterios estrictos de rendimiento minimiza los riesgos de fallos en producción.

# Bibliografía


**Axelos.** (2019). *ITIL Foundation: ITIL 4 Edition*. TSO (The Stationery Office).

**Biblioteca del Congreso Nacional de Chile.** (1999). *Ley 19.628: Sobre protección de la vida privada*. Ministerio de Secretaría General de la Presidencia. Recuperado de https://www.bcn.cl/leychile

**Biblioteca del Congreso Nacional de Chile.** (2012). *Ley 20.584: Regula los derechos y deberes que tienen las personas en relación con acciones vinculadas a su atención de salud*. Ministerio de Salud. Recuperado de https://www.bcn.cl/leychile

**Django Software Foundation.** (2025). *Django Documentation: Models and Views*. Recuperado de https://docs.djangoproject.com/

**IEEE.** (2008). *IEEE Standard for Software and System Test Documentation (IEEE Std 829-2008)*. IEEE Computer Society.

**Instituto Nacional de Normalización.** (2012). *NCh-ISO 27001: Tecnología de la información - Técnicas de seguridad - Sistemas de gestión de seguridad de la información - Requisitos*. INN Chile.

**International Organization for Standardization.** (2011). *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models (ISO/IEC 25010:2011)*.

**International Organization for Standardization.** (2013). *Information technology — Security techniques — Code of practice for information security controls (ISO/IEC 27002:2013)*.

**International Organization for Standardization.** (2018). *Information technology — Security techniques — Information security risk management (ISO/IEC 27005:2018)*.

**Pressman, R. S., & Maxim, B. R.** (2020). *Ingeniería de software: Un enfoque práctico* (9a ed.). McGraw-Hill Interamericana.

**Sommerville, I.** (2016). *Software Engineering* (10a ed.). Pearson Education.

**World Wide Web Consortium (W3C).** (2014). *HTML5: A vocabulary and associated APIs for HTML and XHTML*. Recuperado de https://www.w3.org/TR/html5/