

---

**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

**UNIDAD DE POSGRADO**

**Facultad de Ingeniería de Sistemas e Informática – FISI**

**Maestría en Ingeniería de Sistemas e Informática Mención en Ingeniería de Software**



**CURSO: GESTIÓN DE LA CALIDAD DEL SOFTWARE**

**Docente: REMBRANDT UBALDE**

**GRUPO N.º 1:**

Heber Hualpa Canales.

Melissa Rodriguez Sandoval.

Ronald Ticona Humpiri.

Sihomara Ochoa Cisneros.

Jhonathan Pauca Joya.

Lima, Octubre de 2025

---

Mapeo problemas detectados → ISO/IEC 25010

| Problema Técnico  | Característica   | Observación   |
|---|--|---|
| Consultas y reportes tardan hasta <b>10 s</b> (Tasa de respuesta >8s) | Performance efficiency                                     | Prioridad alta — afecta adopción y percepción.      |
| Código con alto acoplamiento y ciclomática promedio <b>25</b>         | Maintainability  | Requiere refactor + métricas estáticas              |
| Interfaz confusa y poco uniforme (Satisfacción: <b>62%</b> )          | Usability  | UX redesign + pruebas de usabilidad.                |
| Falla de sincronización móvil (Error rate: <b>5%</b> )                | Reliability  | Revisar cola/sincronización y manejo de reintentos. |
| Falta de métricas y QA formal, documentación deficiente               | Maintainability / Testability ( <b>IEEE 730/ISO12207</b> ) | Implantar CI/CD, trazabilidad y QA.                 |

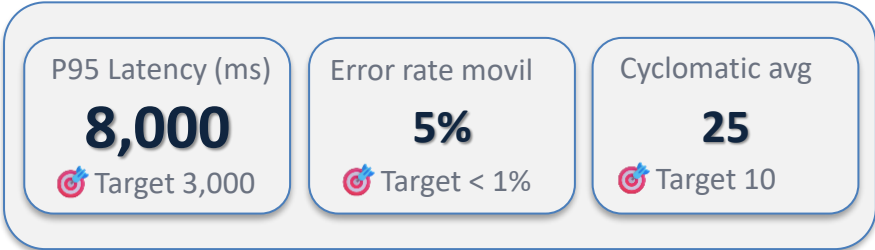
**Problema:** Latencias en reportes (hasta 10s), deuda técnica elevada (ciclomática 25), sync móvil con 5% error y satisfacción usuario 62%.

**Solución propuesta (alto nivel)**  
CI/CD + SonarQube, observabilidad (Prometheus), peer review, pruebas autom. (JUnit, Selenium, k6), optimizaciones DB & caching, endurecer sync móvil.

Beneficios clave:

- P95 latency ↓ a ≤3s
- Error rate móvil <1%
- Mantenibilidad: ciclomática media ≤10
- Satisfacción usuario ≥80%
- Menor tiempo de despliegue y rollback más seguros

KPI clave: P95 latency, error rate, coverage, code smells críticos.



GRUPO 01

[LINK DOCUMENTO](#)

# Acciones Técnicas Propuestas — SILSU

| Estándar / Dimensión                                 | Prácticas y Configuraciones Clave  |
|--|--|
| ISO/IEC 25010 – Instrumentación y observabilidad     | Integrar Micrometer en Spring Boot con Prometheus + Grafana para métricas de JVM, requests y base de datos. Instrumentar Angular con Lighthouse y RUM básico (TTFB, métricas de usuario real).   |
| ISO/IEC 25010 – Performance y eficiencia             | Definir escenarios de carga (picos y concurrencia real), ejecutar pruebas con k6. Analizar P50/P95/P99, optimizar SQL con EXPLAIN e índices, aplicar paginación y cache (Redis).Optimizar configuración de Tomcat/JVM, heap y GC tuning. |
| ISO/IEC 25010 – Mantenibilidad y calidad del código  | Integrar SonarQube con quality gates; refactor guiado por hotspots (alta complejidad ciclomática). Implementar revisiones de código obligatorias (MRs) con checklist formal: 1-2 aprobadores   |
| ISO 29119 – QA y pruebas funcionales                 | Crear suite de tests unitarios (JUnit 5) e integración con cobertura ≥70% (objetivo 80%). Automatizar pruebas de UI con Selenium y pruebas de regresión en CI.   |
| ISO/IEC 25010 – Confiabilidad / Sincronización móvil | Robustecer lógica de sincronización (idempotencia, retries exponenciales, colas persistentes). Pruebas de integración móviles y simulación de condiciones offline.   |
| ISO/IEC 27001 – Seguridad y dependencias             | Integrar escaneo de dependencias (Snyk, Dependabot, GitLab Security) y análisis SAST (CodeQL / Semgrep).   |
| IEEE 730 – Procesos y documentación                  | Establecer plan de control de calidad: roles, artefactos, criterios de entrada/salida. Documentar arquitectura, contratos API (OpenAPI), runbooks y playbooks de incidentes.   |
| ISO/IEC 25010 – CI/CD y entrega continua             | Diseñar pipeline Jenkins: build → tests (unit + static) → sonar → deploy staging → e2e tests → approval → deploy producción.   |
| ISO/IEC 25010 – Disponibilidad y monitoreo           | Definir alertas proactivas: • P95 response > 4s• HTTP error rate >1% (5m)• DB connections >80%• Sync error rate >1%.   |

# Propuesta de métricas y Herramientas — SILSU

| Métrica                               | Fórmula / cómo medir                     | Herramienta              | Frecuencia              | Valor actual  | Objetivo (6 meses)               | Responsable    |
|---------------------------------------|--|--------------------------|-------------------------|---------------|----------------------------------|----------------|
| Tiempo de respuesta (P50/P95)         | Latency histograma (ms)                  | Prometheus + APM         | 5m / reportes semanales | P95 > 8000 ms | P50 ≤ 800 ms; P95 ≤ 3000 ms      | DevOps/Backend |
| Throughput (req/s)                    | Requests por segundo                     | Prometheus/nginx/tomcat  | 5m                      | —             | Mantener sin degradación en peak | DevOps         |
| Error rate HTTP                       | errores 5xx / total                      | Prometheus               | 5m                      | —             | <0.5%                            | Backend        |
| Tasa de fallo de sincronización móvil | Fallos sincronización / intentos         | Logs + métricas mobile   | Diario                  | 5%            | <1%                              | Mobile team    |
| Complejidad ciclomática media         | Media por archivo / módulo (Sonar)       | SonarQube                | En cada PR              | ~25           | ≤10 en módulos críticos          | Dev leads      |
| Code smells críticos / bloqueantes    | Conteo                                   | SonarQube                | En cada PR              | Alta          | 0 nuevos; reducir existentes 80% | Devs           |
| Cobertura de tests (unit)             | Lines covered / total                    | JaCoCo                   | En cada PR              | Baja          | ≥70%                             | Devs           |
| Tiempo medio de recuperación (MTTR)   | Tiempo entre incident start y resolución | Incidents DB             | Por incidente           | —             | <60 min (SLA)                    | SRE/On-call    |
| Disponibilidad                        | Uptime (%)                               | Prometheus/uptime checks | Diario                  | —             | ≥99.5%                           | Infra          |
| Satisfacción de usuario (SUS/NPS)     | Encuesta                                 | Survey                   | Trimestral              | 62%           | ≥80%                             | Product/UX     |
| Tiempo de despliegue (lead time)      | PR->prod                                 | GitLab/Jenkins metrics   | Mensual                 | —             | Reducir 30%                      | DevOps         |

## Mejoras Tecnológicas — SILSU

| Etapas  | Acción técnica  | Resultado esperado  |
|---|---|---|
| <b>Análisis de dominio (Domain Decomposition)</b>     | Identificar bounded contexts según dominios funcionales: licenciamiento, indicadores, supervisión, autenticación, reportes.                       | Base para modularización según el modelo de negocio SUNEDU.   |
| <b>Extracción progresiva de módulos</b>               | Migrar gradualmente componentes de alto cambio (por ejemplo, reportes e indicadores) a microservicios RESTful.                                    | Reducción del acoplamiento y de los tiempos de despliegue.  |
| <b>Definición de contratos estables (API Gateway)</b> | Diseñar APIs estandarizadas con OpenAPI/Swagger, gestionadas vía Kong o Spring Cloud Gateway.   | Facilita interoperabilidad con entidades externas, también facilita la gestión de autenticación, enrutamiento y límites de tráfico, evitando caídas por sobrecarga. |
| <b>Desacoplamiento de datos</b>                       | Cada microservicio gestiona su propia base de datos Independencia transaccional y aislamiento lógico (PostgreSQL schemas o instancias separadas). | de fallos.  |
| <b>Comunicación asíncrona</b>                         | Incorporar RabbitMQ/Kafka para eventos entre servicios (ej. registro de licencias → actualización de indicadores).                                | Mejora la resiliencia y reduce la latencia percibida.   |
| <b>Observabilidad unificada</b>                       | Exponer métricas con Micrometer, centralizar logs en ELK o Grafana Loki.  | Visibilidad transversal de desempeño y errores.   |
| <b>Automatización CI/CD modular</b>                   | Pipelines independientes en GitLab/Jenkins por microservicio con SonarQube, pruebas unitarias y despliegue en Docker.                             | Incrementa frecuencia de releases y confiabilidad de despliegues.   |

1.Repo GitLab con ramas: feature -> develop -> release -> main.

2. Jenkins detecta PR/MR y ejecuta:

- build
- unit tests (JaCoCo report)
- SonarQube analysis (quality gate)
- security scans (Snyk)
- artefact store (Docker image)

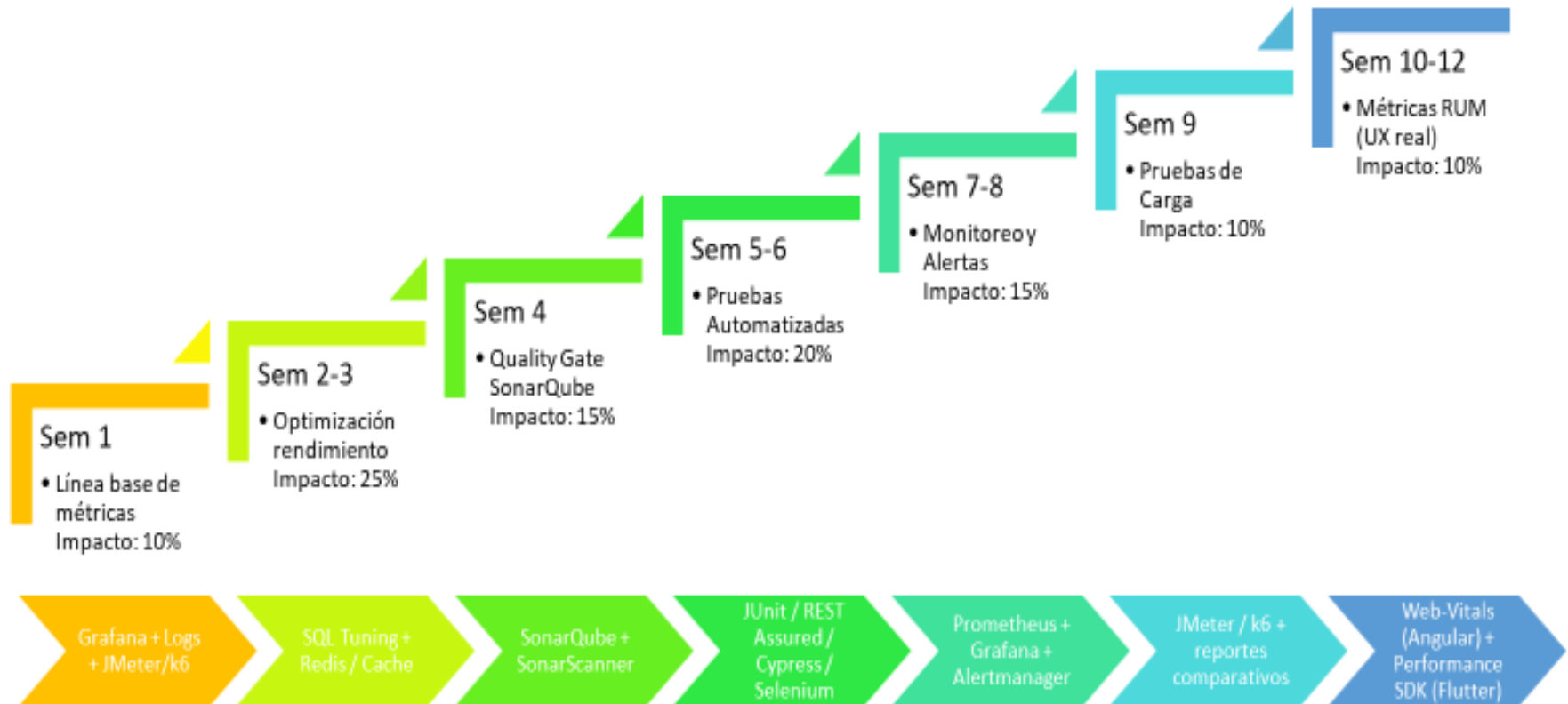
3. Tras pasar quality gate:

- deploy a staging (docker-compose)
- ejecutar e2e tests (Selenium)
- análisis de performance en staging (k6 smoke)

4.Manual approval -> deploy production automatizado.

5.Monitoring: Prometheus (app, DB, exporters). Grafana dashboards y gestor de reglas de alerta.

## Roadmap de Implementación (3 Meses) - SILSU



# IMPACTO ESTRATÉGICO Y GOBERNANZA DE TI

## Beneficios de la Confiabilidad (ISO 25010)

**Credibilidad Institucional:** Reducción del Error Rate (5% → < 1%) previene pérdida de datos sensibles y errores en el proceso de Licenciamiento

**Transparencia:** El sistema toma decisiones (Licencias) basadas en datos íntegros y oportunos

## Beneficios del Proceso (ISO 12207 / IEEE 730)



**Auditoría de Calidad:** La evidencia formal generada por el Plan SQA (IEEE 730) facilita la auditoría por parte de Órganos de Control (CGR)

**Sostenibilidad:** La formalización del proceso reduce la dependencia de personas y los costos de mantenimiento correctivo a largo plazo

## Conclusión Final

El Plan de Mejora es una **inversión en la gobernanza de TI** que asegura la calidad del servicio público, maximizando el valor público de la SUNEDU ante las universidades y la ciudadanía.

¡Gracias!  
Preguntas y Respuestas



# Nuestro Equipo



**Jhonathan Pauca**

Me dedico a innovar soluciones que generen valor a través de liderar proyectos y emprendimientos comercialmente viables y que contribuyen al bienestar social y al desarrollo sostenible.



**Melissa Rodriguez**

Me dedico a analizar y desarrollar soluciones que le den valor a las áreas de negocio.



**Heber Hualpa**

Me dedico a soluciones de infraestructura



**Sihomara Ochoa**

Desarrolladora full stack especializada en soluciones digitales y análisis de datos para la industria y la minería.



**Ronald Ticona**

Senior de Proyectos  
Mine to Mill |  
Fragmentación &  
conminución |  
ML/Modelado predictivo  
(Python/R) + analítica  
tiempo real (PI System,