“La recuperación y consolidación de la economía peruana" en Perú”

UNIVERSIDAD PERUANA LOS ANDES
FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

ARQUITECTURA DE SOFTWARE

Integrantes:

Ramirez Yarasca Edwin

Retamozo Ataucusi Josue Nehemias

Sanchez Romero Aldair

Docente:

FERNANDEZ BEJARANO RAUL ENRIQUE

Ciclo:

8 B”1”

HUANCAYO – PERÚ

2025

Actividad 2: Diseño de una Arquitectura Basada en Estándares

Enunciado: En equipos, diseñen la arquitectura de un sistema de software para una biblioteca digital. Deben aplicar al menos dos estándares internacionales relevantes y documentar el diseño utilizando plantillas basadas en ISO/IEC/IEEE 42010. El entregable debe incluir:

- Objetivos del sistema.
- Stakeholders y sus preocupaciones.
- Vistas arquitectónicas (lógica, desarrollo, procesos, física).
- Relación con los estándares aplicados.

Solución esperada: Un documento arquitectónico estructurado que evidencie el uso de estándares en la definición de vistas, decisiones arquitectónicas y atributos de calidad. Se valorará la claridad, coherencia y alineación con los estándares seleccionados.

Actividad 2: Diseño de Arquitectura de una Biblioteca Digital

Objetivo del Sistema

El objetivo del sistema es permitir la gestión digital de documentos, principalmente libros y artículos científicos, de forma centralizada y accesible para los usuarios, con un enfoque en la gestión eficiente de los recursos, la búsqueda de contenidos, la visualización y la descarga de documentos. Además, debe incluir características como la asignación de permisos, la integración con otras plataformas y la capacidad de soportar múltiples formatos (PDF, ePub, audio, etc.).

Estándares aplicados al diseño arquitectónico

1. ISO/IEC/IEEE 42010: Arquitectura de Software Este estándar es esencial para definir cómo debe organizarse la arquitectura de un sistema software y cómo los diversos componentes deben interactuar entre sí. Permite estructurar el sistema en componentes modulares, con una descripción clara de las vistas arquitectónicas, decisiones y preocupaciones clave. 2. ISO/IEC 25010: Calidad del Software Este estándar proporciona un conjunto de atributos de calidad que deben ser considerados al desarrollar el sistema. La aplicación de este estándar ayudará a garantizar que el sistema de la Biblioteca Digital cumpla con los requisitos no funcionales esenciales.

Vistas Arquitectónicas (ISO/IEC/IEEE 42010)

Vista Lógica: Módulos principales: Frontend, Backend, Base de Datos e Integraciones. Se recomienda el uso de microservicios para separación de responsabilidades. • Vista de Desarrollo: Código dividido en repositorios específicos para cada servicio (gestión de libros, usuarios, pagos). APIs bien definidas. • Vista de Procesos: Flujo de usuario desde autenticación hasta búsqueda, lectura y pagos en caso de libros premium. • Vista Física: Infraestructura en la nube con contenedores (Docker/Kubernetes), balanceadores de carga y bases de datos distribuidas.

Estándar	Componente del Sistema	Relación
ISO/IEC/IEEE 42010	Módulos (Documentos, Seguridad, Autenticación, Pagos)	Define la necesidad de una arquitectura modular y documentada que facilite la expansión y el mantenimiento.
ISO/IEC 25010	Gestión de documentos, Autenticación, Seguridad	Establece los atributos de calidad como la seguridad, usabilidad, y rendimiento, que guían el diseño de los módulos y las interacciones entre ellos.

Conclusión

El diseño arquitectónico propuesto para la Biblioteca Digital sigue principios sólidos de modularidad y microservicios, asegurando que el sistema sea escalable, seguro y fácil de mantener. La aplicación de los estándares ISO/IEC/IEEE 42010 y ISO/IEC 25010 garantiza que el sistema sea de alta calidad, ofreciendo una excelente experiencia de usuario y robustez operativa.

Actividad 3: Evaluación de Calidad Arquitectónica

Enunciado: Desarrolla una matriz de evaluación de calidad para un sistema de software existente o propuesto, utilizando el estándar ISO/IEC 25010. Evalúa atributos como usabilidad, rendimiento, seguridad y mantenibilidad. Presenta los resultados en un informe comparativo entre el estado actual y el estado ideal según el estándar.

Solución esperada: Una matriz con criterios de evaluación, puntuaciones y análisis crítico. El estudiante debe identificar brechas y proponer mejoras arquitectónicas para cumplir con los estándares de calidad. Se espera una reflexión sobre cómo los estándares guían el diseño y evaluación de software.

Matriz de Evaluación de Calidad Arquitectónica - ISO/IEC 25010

1. Introducción

Sistema Evaluado: E-Commerce Platform "TechStore"

- **Dominio:** Plataforma de comercio electrónico
- **Usuarios:** 50,000+ usuarios activos
- **Tecnología:** Arquitectura de microservicios, React frontend, Node.js backend
- **Base de datos:** PostgreSQL, Redis para caché

Objetivo de la Evaluación

Evaluar la calidad arquitectónica actual del sistema TechStore utilizando el estándar ISO/IEC 25010 e identificar brechas para proponer mejoras que eleven la calidad del software.

2. Metodología de Evaluación

Escala de Puntuación

- **1-2:** Deficiente - Requiere mejoras críticas inmediatas
- **3-4:** Básico - Cumple requisitos mínimos, necesita mejoras
- **5-6:** Satisfactorio - Cumple estándares aceptables
- **7-8:** Bueno - Supera expectativas básicas
- **9-10:** Excelente - Estado óptimo según estándares

Criterios de Evaluación

Cada característica se evalúa mediante métricas específicas, evidencias documentadas y pruebas técnicas.

3. Matriz de Evaluación de Calidad

Característica	Actual	Objetivo	Brecha
Funcionalidad	6.2	8.5	2.3
Confiabilidad	5.7	8.7	3.0
Usabilidad	6.8	8.8	2.0
Eficiencia	5.3	8.3	3.0
Mantenibilidad	6.5	8.5	2.0
Seguridad	4.8	9.0	4.2
Compatibilidad	7.0	8.5	1.5
Portabilidad	6.3	8.0	1.7

4. Análisis de Resultados

Puntuación General del Sistema

- **Puntuación Actual Promedio:** 6.08/10
- **Puntuación Objetivo Promedio:** 8.52/10
- **Brecha General:** 2.44 puntos

Características Críticas (Mayor Brecha)

1. **Seguridad (Brecha: 4.2)** - Requiere atención inmediata
2. **Confiabilidad (Brecha: 3.0)** - Impacta directamente la experiencia del usuario
3. **Eficiencia (Brecha: 3.0)** - Afecta la escalabilidad del negocio

Características en Mejor Estado

1. **Compatibilidad (Brecha: 1.5)** - Relativamente bien implementada
2. **Portabilidad (Brecha: 1.7)** - Arquitectura moderna facilita portabilidad

5. Identificación de Brechas Críticas

5.1 Brechas de Seguridad

Problemas Identificados:

- Encriptación inconsistente de datos sensibles
- Falta de auditoría completa deacciones críticas
- Autenticación de dos factores no obligatoria
- Validación de entrada incompleta

Impacto: Alto riesgo de brechas de seguridad y incumplimiento regulatorio

5.2 Brechas de Confiabilidad

Problemas Identificados:

- Tiempo medio entre fallas muy bajo (7 días)
- Disponibilidad por debajo de estándares industriales
- Recuperación manual en muchos escenarios
- Tiempo de recuperación excesivo

Impacto: Pérdida de confianza del usuario y revenue

5.3 Brechas de Eficiencia

Problemas Identificados:

- Tiempos de respuesta superiores a expectativas modernas
- Utilización alta de recursos en picos
- Capacidad limitada de usuarios concurrentes

Impacto: Limitación del crecimiento y experiencia de usuario degradada

6. Propuestas de Mejora Arquitectónica

6.1 Mejoras en Seguridad (Prioridad: Crítica)

Arquitectura de Seguridad por Capas:

- Implementar Web Application Firewall (WAF)
- Establecer Zero Trust Architecture
- Implementar Service Mesh con mTLS

Medidas Específicas:

- Migrar a OAuth 2.0 + OpenID Connect
- Implementar Hardware Security Modules (HSM)
- Establecer Security Information and Event Management (SIEM)
- Implementar Data Loss Prevention (DLP)

Timeline: 6 meses **Inversión Estimada:** $150,000 **ROI Esperado:** Reducción 90% incidentes

seguridad

6.2 Mejoras en Confiabilidad (Prioridad: Alta)

Arquitectura de Alta Disponibilidad:
- Implementar patrón Circuit Breaker
- Establecer redundancia multi-zona
- Implementar chaos engineering

Medidas Específicas:
- Migrar a arquitectura event-driven
- Implementar health checks avanzados
- Establecer automated rollback
- Implementar distributed tracing

Timeline: 4 meses **Inversión Estimada:** $100,000 **ROI Esperado:** Reducción 75% downtime

6.3 Mejoras en Eficiencia (Prioridad: Alta)

Arquitectura de Alto Rendimiento:

- Implementar CDN global
- Establecer caching multinivel
- Optimizar queries de base de datos

Medidas Específicas:
- Implementar lazy loading y code splitting
- Establecer database sharding
- Implementar async processing con queues
- Optimizar bundle sizes y assets

Timeline: 3 meses **Inversión Estimada:** $75,000 **ROI Esperado:** Mejora 50% performance

6.4 Mejoras en Mantenibilidad (Prioridad: Media)

Arquitectura Limpia y Modular:
- Refactorizar hacia Domain-Driven Design
- Implementar patrones SOLID consistentemente
- Establecer documentación automática

Medidas Específicas:
- Implementar architectural decision records (ADRs)
- Establecer automated testing pyramid
- Implementar static code analysis
- Crear component library reutilizable

Timeline: 5 meses **Inversión Estimada:** $80,000 **ROI Esperado:** Reducción 40% tiempo

desarrollo

7. Plan de Implementación

Fase 1: Fundamentos (Meses 1-2)

- Implementación medidas seguridad críticas
- Establecimiento monitoring y alertas
- Mejoras básicas de performance

Fase 2: Estabilización (Meses 3-4)

- Implementación alta disponibilidad
- Optimización rendimiento avanzada
- Automatización testing

Fase 3: Optimización (Meses 5-6)

- Refactoring arquitectónico
- Implementación patrones avanzados
- Documentación y knowledge transfer

8. Métricas de Seguimiento

KPIs de Calidad

- **Seguridad:** Número de vulnerabilidades críticas (Objetivo: 0)
- **Confiabilidad:** Uptime mensual (Objetivo: >99.5%)
- **Eficiencia:** Tiempo respuesta p95 (Objetivo: <1s)
- **Mantenibilidad:** Tiempo promedio desarrollo feature (Objetivo: <3 días)

Proceso de Monitoreo

- Evaluación mensual de métricas
- Revisión trimestral de arquitectura
- Audit anual de cumplimiento ISO/IEC 25010

9. Reflexión sobre el Estándar ISO/IEC 25010

Valor del Estándar en el Diseño

El estándar ISO/IEC 25010 proporciona un framework sistemático para evaluar la calidad del software que va más allá de la funcionalidad básica. Su aplicación en este ejercicio reveló:

Beneficios Clave:

- **Visión Holística:** Aborda todos los aspectos críticos de la calidad, no solo la funcionalidad
- **Objetividad:** Proporciona criterios medibles y comparables
- **Orientación Estratégica:** Ayuda a priorizar inversiones en mejoras arquitectónicas
- **Comunicación:** Facilita discusiones técnicas con stakeholders no técnicos

Guía para el Diseño Arquitectónico

El estándar influye en las decisiones arquitectónicas de múltiples formas:

1. **Decisiones Tempranas:** Considera atributos de calidad desde el diseño inicial
2. **Trade-offs Informados:** Ayuda a evaluar compromisos entre diferentes características
3. **Validación Continua:** Proporciona base para testing y validación arquitectónica
4. **Evolución Guiada:** Orienta la evolución y refactoring del sistema

Limitaciones y Consideraciones

- **Contextualización:** Requiere adaptación al dominio específico
- **Subjetividad:** Algunas métricas pueden tener elementos subjetivos
- **Overhead:** La evaluación completa requiere tiempo y recursos significativos
- **Dinamismo:** Necesita actualizaciones regulares para mantener relevancia

10. Conclusiones

La evaluación mediante ISO/IEC 25010 revela que el sistema TechStore tiene una base sólida pero requiere mejoras significativas, especialmente en seguridad y confiabilidad. Las brechas identificadas representan tanto riesgos como oportunidades de mejora que, al ser abordadas sistemáticamente, pueden elevar significativamente la calidad general del sistema.

La inversión total estimada de $405,000 distribuida en 6 meses puede resultar en un sistema que no solo cumple con los estándares internacionales, sino que también proporciona una plataforma sólida para el crecimiento futuro del negocio.

El uso del estándar ISO/IEC 25010 como guía demostró ser invaluable para identificar aspectos de calidad que podrían haber sido pasados por alto en una evaluación menos sistemática, confirmando su valor como herramienta fundamental en la ingeniería de software moderna.