Instituto Politécnico Nacional

Escuela Superior de Cómputo

Ingeniería en software

Integrantes:

Contla Mota Luis Andrés

Escárcega Hernández Steven Arturo

Gómez Sánchez Héctor Rodrigo

García Mayorga Rodrigo

MediWeb – Sistema Médico

Autores: Contla Mota Luis Andrés, Escárcega Hernández Steven Arturo, Gómez Sánchez Héctor Rodrigo, García Mayorga Rodrigo
Afiliación institucional: Instituto Politécnico Nacional, ESCOM

Ejercicio 1: Análisis detallado de la calidad del proceso y del producto

El análisis de calidad para el proyecto "Sistema Médico" debe considerar tanto los procesos como el producto final:

Calidad del Proceso:

1. Gestión del Proyecto:

- El repositorio muestra una estructura organizada usando C como lenguaje principal
- La actualización reciente (25 de junio de 2025) indica actividad continua
- La gestión de issues está habilitada, permitiendo el seguimiento de problemas

2. Desarrollo:

- Se debe evaluar la implementación de metodologías ágiles o tradicionales
- Verificar la existencia de documentación del proceso de desarrollo
- Revisar la frecuencia y calidad de los commits

3. Pruebas:

- Evaluar la existencia de pruebas unitarias, de integración y de sistema
- Verificar la cobertura de código por pruebas automatizadas
- Comprobar la documentación de casos de prueba

Calidad del Producto:

1. Funcionalidad:

- Evaluar si el sistema cumple con los requerimientos establecidos
- Verificar la integridad funcional y la adecuación a las necesidades médicas
- Comprobar la seguridad del sistema, crucial para datos médicos

2. Usabilidad:

- Analizar la interfaz de usuario y su facilidad de aprendizaje
- Verificar la accesibilidad para diferentes tipos de usuarios
- Evaluar la protección contra errores de usuario

3. Eficiencia:

- Medir los tiempos de respuesta y uso de recursos
- Verificar el comportamiento bajo carga de trabajo elevada
- Analizar la escalabilidad del sistema

4. Mantenibilidad:

- Evaluar la modularidad del código
- Verificar la facilidad de modificación y reutilización
- Analizar la posibilidad de realizar pruebas efectivas

5. Confiabilidad:

- Evaluar la madurez del sistema y su tolerancia a fallos
- Verificar la capacidad de recuperación ante situaciones problemáticas
- Analizar la disponibilidad del sistema

Ejercicio 2: Lista de verificación basada en ISO/IEC 25010 y resultados de la auditoría

Lista de Verificación (Checklist) basada en ISO/IEC 25010:

1. Adecuación Funcional

- [x] ¿El sistema permite el registro completo de pacientes?
- [x] ¿Se pueden gestionar citas médicas eficientemente?
- [x] ¿El sistema maneja correctamente los historiales médicos?
- [x] ¿Existe un sistema de alertas para medicamentos e interacciones?
- [x] ¿Se generan reportes médicos completos y precisos?

2. Eficiencia de Desempeño

- [x] ¿Los tiempos de respuesta son adecuados (<2 segundos)?
- [x] ¿El sistema mantiene su rendimiento con múltiples usuarios simultáneos?

- [x] ¿Se optimiza el uso de recursos del servidor?
- [x] ¿Las consultas a la base de datos están optimizadas?
- [x] ¿La aplicación escala correctamente con el volumen de datos?

3. Compatibilidad

- [x] ¿El sistema se integra con otros sistemas hospitalarios?
- [x] ¿Es compatible con estándares médicos electrónicos (HL7, DICOM)?
- [x] ¿Coexiste adecuadamente con otros sistemas en el mismo entorno?
- [x] ¿La exportación/importación de datos funciona correctamente?
- [x] ¿Es compatible con diferentes navegadores y dispositivos?

4. Usabilidad

- [x] ¿La interfaz es intuitiva para personal médico?
- [x] ¿Existe ayuda contextual para usuarios?
- [x] ¿El sistema es accesible para personas con discapacidades?
- [x] ¿Se proporciona retroalimentación clara de las acciones?
- [x] ¿Los mensajes de error son descriptivos y útiles?

5. Fiabilidad

- [x] ¿El sistema tiene mecanismos de respaldo automático?
- [x] ¿Existen procedimientos de recuperación ante fallos?
- [x] ¿El sistema está disponible 24/7?
- [x] ¿Se monitorea el estado del sistema constantemente?
- [x] ¿Se realizan pruebas de estrés regularmente?

6. Seguridad

- [x] ¿Se implementa autenticación de doble factor?
- [x] ¿Los datos sensibles están cifrados?
- [x] ¿Se mantiene un registro detallado de accesos y cambios?
- [x] ¿Cumple con regulaciones médicas (HIPAA, GDPR)?
- [x] ¿Se realizan auditorías de seguridad periódicas?

7. Mantenibilidad

- [x] ¿El código sigue estándares y convenciones?
- [x] ¿Existe documentación técnica actualizada?
- [x] ¿El código está modularizado adecuadamente?
- [x] ¿Se utilizan patrones de diseño apropiados?
- [x] ¿Existen pruebas automatizadas con buena cobertura?

8. Portabilidad

- [x] ¿El sistema puede desplegarse en diferentes entornos?
- [x] ¿La instalación es sencilla y documentada?
- [x] ¿Se puede migrar fácilmente a nuevas versiones?
- [x] ¿Es adaptable a diferentes configuraciones de hardware?
- [x] ¿Puede reemplazar eficientemente sistemas anteriores?

Resultados de la Auditoría (Simulados):

Tras aplicar la lista de verificación, se identificaron las siguientes fortalezas y debilidades:

Fortalezas:

- Alta adecuación funcional para la gestión de pacientes y citas
- Buena seguridad en autenticación y cifrado de datos
- Interfaces de usuario intuitivas para el personal médico
- Documentación completa del sistema

Debilidades:

- Rendimiento subóptimo con cargas elevadas de usuarios
- Integración limitada con sistemas externos
- Falta de pruebas automatizadas completas
- Problemas de portabilidad entre diferentes navegadores

Ejercicio 3: Autoevaluación de madurez del equipo, justificación del nivel y plan de mejora

Autoevaluación de Madurez del Equipo

Utilizando el modelo de madurez CMMI (Capability Maturity Model Integration), evaluamos al equipo de desarrollo:

Nivel de Madurez Estimado: 3 - Definido

Justificación:

1. Procesos Establecidos:

- **El equipo ha establecido procesos estándar para el desarrollo**
- **Existe documentación de los procesos principales**
- **Se siguen metodologías de desarrollo consistentes**

2. Gestión de Proyectos:

- **Se utilizan herramientas de gestión de proyectos**
- **Los roles y responsabilidades están claramente definidos**
- **Se realiza seguimiento sistemático del progreso**

3. Calidad:

- **Se han establecido estándares de código**
- **Existen revisiones de código, aunque no son completamente sistemáticas**
- **Se realizan pruebas, pero la automatización es parcial**

4. No alcanza Nivel 4 porque:

- **Falta medición cuantitativa sistemática de los procesos**
- **La gestión de la calidad no está completamente integrada**
- **Las mejoras de proceso son reactivas, no proactivas**

Plan de Mejora Propuesto:

Corto Plazo (3 meses):

1. Implementar revisiones de código obligatorias

- **Establecer estándares de revisión**
- **Integrar herramientas automatizadas de análisis de código**
- **Capacitar al equipo en mejores prácticas de revisión**

2. Mejorar la cobertura de pruebas

- Implementar pruebas unitarias para módulos críticos
- Establecer objetivos mínimos de cobertura (>70%)
- Integrar pruebas en el pipeline de CI/CD

3. Documentar procesos clave

- Crear wiki técnica accesible para todo el equipo
- Estandarizar la documentación de APIs
- Establecer plantillas para documentación de requisitos

Mediano Plazo (6-12 meses):

1. Implementar métricas de proceso

- Definir KPIs relevantes para el proceso de desarrollo
- Implementar herramientas de monitoreo continuo
- Establecer reuniones periódicas de análisis de métricas

2. Mejorar la gestión de requisitos

- Implementar trazabilidad completa de requisitos
- Mejorar el proceso de validación con stakeholders
- Automatizar pruebas de aceptación

3. Fortalecer la seguridad

- Realizar auditorías de seguridad periódicas
- Implementar análisis de vulnerabilidades automatizado
- Capacitar al equipo en desarrollo seguro

Largo Plazo (1-2 años):

1. Avanzar hacia CMMI nivel 4

- Implementar gestión cuantitativa de procesos
- Establecer objetivos de calidad medibles
- Implementar mejora continua basada en datos

2. Optimizar la eficiencia del equipo

- Automatizar tareas repetitivas
- Implementar DevOps completo
- Reducir el tiempo de ciclo de desarrollo

3. Expandir capacidades técnicas

- Capacitar en nuevas tecnologías relevantes
- Implementar arquitectura orientada a microservicios
- Mejorar la escalabilidad del sistema

Conclusiones Generales

La aplicación de estándares de calidad como ISO/IEC 25010 y modelos de madurez como CMMI resulta fundamental en el desarrollo profesional de software, especialmente en sistemas críticos como los médicos. A través de este análisis, podemos concluir:

1. Los estándares de calidad proporcionan un marco objetivo para evaluar tanto el proceso como el producto, facilitando la identificación de fortalezas y debilidades de manera sistemática.

2. La calidad no es un estado final sino un proceso continuo que requiere evaluación y mejora constante. El plan de mejora propuesto refleja esta filosofía, estableciendo metas graduales y medibles.

3. La madurez del equipo de desarrollo impacta directamente en la calidad del producto. Un equipo con procesos más maduros tiene mayor capacidad para predecir resultados, gestionar riesgos y entregar software confiable.

4. El balance entre diferentes atributos de calidad es esencial. En sistemas médicos, por ejemplo, la seguridad y confiabilidad pueden tener prioridad sobre la eficiencia en ciertas funcionalidades.

5. La automatización de pruebas y revisiones es clave para mantener altos estándares de calidad de manera sostenible, especialmente cuando el sistema crece en complejidad.

6. La documentación adecuada facilita no solo el mantenimiento sino también la transferencia de conocimiento y la incorporación de nuevos miembros al equipo.

En la ingeniería de software profesional, la aplicación sistemática de estos estándares y modelos no debe verse como una carga administrativa adicional, sino como una inversión que reduce costos a largo plazo, minimiza riesgos y genera confianza en los usuarios finales del sistema, elementos particularmente críticos en un sistema médico como el desarrollado en este proyecto.