Guía de Apoyo: Análisis Técnico y Ético de Sistemas Monolíticos

✨ Propósito de esta guía

Este documento está diseñado para entregarte un marco de referencia claro y profundo para resolver actividades donde se analiza un sistema monolítico real, se proponen estrategias de microservicios, y se identifican fallas éticas que pueden afectar a los usuarios, a la sociedad y a los equipos de desarrollo.

🤖 Recordando lo básico: Sistemas Monolíticos vs. Microservicios

Sistema Monolítico

Un sistema monolítico es una aplicación “todo en uno”, donde todos los componentes (autenticación, interfaz, lógica de negocio, acceso a base de datos, etc.) están agrupados y desplegados como una sola unidad. Suelen ser difíciles de escalar, mantener y actualizar.

Ventajas:

- Fácil de desarrollar al inicio.
- Menor complejidad en el despliegue.

Desventajas:

- Cambios en una parte afectan a todo el sistema.
- Difícil escalar de forma independiente.
- Riesgo alto en fallos: si falla una parte, puede caer todo.
- Ámbitos de acceso amplios, generando problemas de seguridad.

Ejemplo real: El sistema de gestión interno original de Amazon, que centralizaba todo el manejo de pedidos, pagos y logística, fue reemplazado gradualmente por microservicios a medida que la compañía creció. El enfoque monolítico no permitía escalar por separado cada área.

Arquitectura de Microservicios

La arquitectura de microservicios divide una aplicación en varios servicios pequeños e independientes, cada uno responsable de una funcionalidad específica, comunicándose entre sí generalmente por APIs (REST, gRPC, etc).

Ventajas:

- Escalabilidad independiente.
- Despliegues separados por servicio.
- Más fácil de mantener y actualizar.

Desafíos:

- Mayor complejidad en la comunicación.
- Necesita una buena estrategia de monitoreo y seguridad.

Ejemplo real: Netflix migró de un sistema monolítico a una arquitectura basada en más de 700 microservicios para mejorar su rendimiento, escalabilidad y capacidad de responder a errores localizados sin afectar toda la plataforma.

📈 Profundizando en la migración

Indicadores técnicos de que un sistema monolítico necesita ser dividido:

- Tiempo de respuesta alto por carga de trabajo.
- Muchas fallas por cambios mínimos en el código.
- Equipos de desarrollo que pisan el trabajo de otros.
- Dependencias complejas entre módulos.
- Escasa trazabilidad de errores.

Buenas prácticas al dividir un monolito:

- Identificar dominios funcionales (“bounded contexts”).
- Separar servicios por responsabilidad clara.
- Aplicar principios como SRP (Single Responsibility Principle).
- Diseñar APIs claras para comunicación entre servicios.
- Definir reglas de acceso y seguridad para cada microservicio.

Ejemplo real: LinkedIn reorganizó varios de sus servicios (como perfiles, publicaciones, y mensajería) en microservicios debido a cuellos de botella en el backend monolítico original, mejorando el rendimiento y la entrega continua.

👩‍💻📄 Herramientas de trabajo colaborativo recomendadas

- **Miro / Jamboard:** Para mapear visualmente la arquitectura actual y futura.
- **Trello / Notion / ClickUp:** Para planificar tareas y responsabilidades.
- **Lucidchart / Draw.io:** Para crear diagramas de arquitectura.

⚡ Elementos éticos clave a considerar

Al rediseñar un sistema, no basta con la técnica: también debes evaluar los efectos humanos, sociales y éticos. Aquí algunas áreas clave:

🔒 Privacidad de los datos

- ¿Se accede a datos sensibles sin consentimiento?
- ¿Se almacenan datos innecesarios?
- ¿Hay control de acceso?

Ejemplo real: El escándalo de Cambridge Analytica reveló que millones de datos de usuarios de Facebook fueron recolectados sin consentimiento claro, lo que generó una crisis global de confianza en la privacidad digital.

🚀 Seguridad de la información

- ¿Están cifrados los datos en tránsito y reposo?
- ¿Se auditan los accesos?
- ¿Existen vulnerabilidades conocidas no corregidas?

Ejemplo real: La filtración de Equifax en 2017 expuso datos financieros de 147 millones de personas debido a una vulnerabilidad conocida no parchada en su sistema monolítico.

🌎 Impacto social y ambiental

- ¿El sistema afecta negativamente a ciertos grupos sociales?
- ¿Fomenta el uso responsable de recursos?

Ejemplo real: Algoritmos de crédito que discriminan por ubicación o historial social, como ocurrió en Apple Card, donde usuarios denunciaron disparidades en los límites de crédito otorgados entre hombres y mujeres.

👮 Equidad y no discriminación

- ¿El sistema tiene sesgos (por ejemplo, género, edad, ubicación)?
- ¿Todos los usuarios acceden a las mismas funcionalidades?

Ejemplo real: El sistema de contratación automatizada de Amazon (descontinuado) discriminaba a mujeres al aprender de datos históricos con sesgos de género.

📰 Transparencia y responsabilidad

- ¿El usuario sabe qué datos entrega y para qué?
- ¿El equipo asume responsabilidad si hay fallos?

Ejemplo real: Zoom fue criticado en 2020 por no explicar claramente que compartía datos con Facebook, lo que llevó a cambios en su política de privacidad.

🚀 Ejemplo de división de un sistema monolítico

Sistema monolítico: Plataforma educativa

Módulos integrados:

- Autenticación
- Publicación de contenidos
- Evaluación online
- Chat entre estudiantes
- Análisis de datos de rendimiento

Posible división en microservicios:

1. **Servicio de autenticación**
2. **Servicio de contenidos educativos**
3. **Servicio de evaluaciones**
4. **Servicio de mensajería**
5. **Servicio de análisis y reportes**

Ejemplo real: Moodle, un LMS ampliamente usado, permite ahora el desarrollo modular y la integración con servicios externos (como plugins y APIs), superando limitaciones de su arquitectura original más monolítica.

💭 Preguntas orientadoras para tu trabajo en equipo

- ¿Cuál es el principal fallo del sistema original?
- ¿Qué parte podría separarse en un microservicio independiente?
- ¿Qué error ético se cometió o se podría cometer?
- ¿Qué medidas técnicas y éticas pueden evitarlo?
- ¿Qué herramienta usarán para planificar y colaborar?

:)
