The Scenario Problem

Objective

Ensure environment stability and prevent incidents while rebuilding trust between teams and establishing effective collaboration channels.

Problem analysis

- Mocks in staging and development environments are not synchronized with changes in the actual production service.
- This misalignment causes repeated incidents in both environments.
- There is frustration within the team due to unreliable development environments, and loss of trust between teams is affecting collaboration.
- The technical person responsible for the other team has a difficult reputation, which complicates communication.

1: Technical diagnosis and incident documentation

- Internal team review to gather objective data.
- Document recent incidents in staging and production environments, collecting concrete examples of incidents and their business impact.
- Identify specific cases where mock inconsistencies caused failures, recording which changes triggered the problems and which mock components were out of sync.
- Quantify the impact of each incident.
- The purpose is to create objective, data-backed documentation for technical discussions, not to assign blame.

2: Technical solutions proposal

- Implement an automated notification system that alerts our team when the other team makes relevant changes in production or staging.
- Establish an additional step in the other team's CI/CD pipeline that generates automatic notifications when changes that could affect service integration are detected.
- Create self-managed mocks based on the current behavior of the production service if the other team cannot keep them updated.
- Configure cross-integration tests in staging to detect broken changes before they reach production.

3: Collaborative meeting approach

- Approach conversations collaboratively, not accusatorily.

- Focus on shared goals of stability and reliability for both teams.
- Present concrete examples and data instead of emotions or frustrations.
- Propose solutions that reduce the burden for both teams.

4: Contingency plan

- While working toward a definitive solution, our team will develop and maintain its own mocks based on the current behavior of the production service. This will allow us to not depend on the other squad for stable development and testing environments.
- We will implement a periodic update process where we monitor changes in production and update our mocks accordingly to maintain synchronization.

5: Escalation (if necessary)

- If after attempting collaborative solutions and presenting concrete proposals no satisfactory resolution is achieved, we will escalate the matter to the management of both teams. We will present the collected incident documentation, estimated economic impact, and proposed solutions that were not implemented.
- We will seek mediation from senior leaders to establish mandatory communication and synchronization processes between teams, prioritizing product stability over individual preferences.

Expected responses and counter-responses

Response 1: "This is not our problem. You should adapt to our changes."
My response: "We understand that services evolve. What we've observed is that changes in production are not reflected in the mocks. We would appreciate establishing a mechanism to keep mocks updated after relevant changes that affect integration between systems."

Response 2: "We don't have time to maintain updated mocks."
My response: "I understand the workload pressure. That's why I'm proposing solutions that minimize effort for your team. We could implement an automated notification system that alerts us when you make changes, so we can take care of updating the mocks."

Response 3: "Our changes shouldn't affect you."
My response: "According to our incident reports, these specific changes in the service broke our integration. We could establish a simple communication protocol for changes that might affect integration between systems."

Response 4: “You should build better error handling.”
My response: “We are improving our resilience, but accurate mocks would help us prepare for changes proactively rather than reactively, especially for changes that affect integrations between services.”

Long-term prevention

- Establish a clear communication protocol for changes that may affect integrations between services.
- Implement automated notifications in deployment pipelines for relevant changes.
- Create shared documentation and open communication channels.
- Schedule regular synchronization meetings between teams.
- Develop automated validation processes for environment consistency.

Key principles

- Maintain professional relationships despite difficulties.
- Focus on data and business impact, not personal frustrations.
- Propose mutually beneficial solutions.
- Document everything for clarity and accountability.
- Escalate through appropriate channels only after attempting collaborative solutions.

Spanish version

Objetivo

Garantizar la estabilidad de los entornos y evitar incidentes, recuperando la confianza entre equipos y estableciendo canales de colaboración efectivos.

Análisis del problema

- Los mocks en entornos de staging y desarrollo no están sincronizados con los cambios del servicio real en producción.

- Esta desalineación causa incidentes repetidos en ambos ambientes.

- Existe frustración en el equipo debido a entornos de desarrollo poco confiables y hay pérdida de confianza entre equipos que afecta la colaboración.

- La persona técnica responsable del otro equipo tiene una reputación difícil, lo que complica la comunicación.

1: Diagnostico técnico y documentación de incidentes

- Revisión interna con el equipo para recopilar datos objetivos.
- Documentar incidentes recientes en entornos de staging y producción, recolectando ejemplos concretos de incidentes y su impacto en el negocio.
- Identificar casos específicos donde inconsistencias en los mocks causaron fallos, registrando qué cambios desencadenaron los problemas y qué componentes de los mocks estaban desincronizados.
- Cuantificar el impacto de cada incidente.
- El propósito es crear documentación objetiva respaldada por datos para discusiones técnicas, no para asignar culpas.

2: Propuesta de soluciones técnicas

- Implementar un sistema de notificaciones automatizadas que alerte a nuestro equipo cuando el otro equipo realice cambios relevantes en producción o staging.
- Establecer un paso adicional en el pipeline de CI/CD del otro equipo que genere una notificación automática cuando se detecten cambios que puedan afectar la integración entre servicios.
- Crear mocks auto-gestionados basados en el comportamiento actual del servicio en producción si el otro equipo no puede mantenerlos actualizados.
- Configurar pruebas de integración cruzada en staging para detectar cambios rotos antes de que lleguen a producción.

3: Enfoque de reunión colaborativa

- Abordar las conversaciones de manera colaborativa, no acusatoria.
- Enfocarse en objetivos compartidos de estabilidad y confiabilidad para ambos equipos.
- Presentar ejemplos concretos y datos en lugar de emociones o frustraciones.
- Proponer soluciones que reduzcan la carga para ambos equipos.

4: Plan de contingencia

- Mientras se resuelve la situación de manera definitiva, nuestro equipo desarrollará y mantendrá sus propios mocks basados en el comportamiento actual del servicio en producción. Esto nos permitirá no depender del otro squad para tener entornos de desarrollo y testing estables.
- Implementaremos un proceso de actualización periódica donde monitorearemos los cambios en producción y actualizaremos nuestros mocks de acuerdo con los cambios para mantener la sincronización.

5: Escalación (si es necesario)

- Si después de intentar soluciones colaborativas y presentar propuestas concretas no se logra una resolucion satisfactoria, escalaremos el tema a la gerencia de ambos equipos. Presentaremos la documentacion recopilada de incidentes, el impacto economico estimado y las soluciones propuestas que no fueron implementadas.
- Buscaremos la mediacion de lideres superiores para establecer procesos obligatorios de comunicacion y sincronizacion entre equipos, priorizando la estabilidad del producto sobre preferencias individuales.

Respuestas esperadas y contrarespuestas

Respuesta 1: "Este no es nuestro problema. Ustedes deberían adaptarse a nuestros cambios."
Mi respuesta: "Entendemos que los servicios evolucionan. Lo que hemos observado es que los cambios en producción no se reflejan en los mocks. Agradeceríamos establecer un mecanismo para mantener los mocks actualizados después de cambios relevantes que afecten la integración entre sistemas."

Respuesta 2: "No tenemos tiempo para mantener mocks actualizados."
Mi respuesta: "Entiendo la presión de la carga de trabajo. Por eso estoy proponiendo soluciones que minimicen el esfuerzo para su equipo. Podríamos implementar un sistema de notificaciones automatizadas que nos alerte cuando hagan cambios, así nosotros nos encargamos de actualizar los mocks."

Respuesta 3: "Nuestros cambios no deberían afectarlos."
Mi respuesta: "Según nuestros reportes de incidentes, estos cambios específicos en el servicio rompieron nuestra integración. Podríamos establecer un protocolo simple de comunicación para cambios que puedan afectar la integración entre sistemas."

Respuesta 4: "Deberían construir mejor manejo de errores."
Mi respuesta: "Estamos mejorando nuestra resiliencia, pero mocks precisos nos ayudarían a prepararnos para cambios de manera proactiva en lugar de reactiva, especialmente para cambios que afecten integraciones entre servicios."

Prevencion a largo plazo

- Establecer un protocolo de comunicación claro para cambios que puedan afectar integraciones entre servicios.
- Implementar notificaciones automatizadas en los pipelines de deployment para cambios relevantes.
- Crear documentación compartida y canales de comunicación abiertos.
- Programar reuniones de sincronización regulares entre equipos.
- Desarrollar procesos de validación automatizada para la consistencia de entornos.

Principios clave

- Mantener relaciones profesionales a pesar de las dificultades.
- Enfocarse en datos e impacto de negocio, no en frustraciones personales.
- Proponer soluciones mutuamente beneficiosas. Documentar todo para claridad y rendición de cuentas.
- Escalar a través de canales apropiados solo después de intentar soluciones colaborativas.