Caso Práctico - Product Owner

Marta Muñoz Barrios

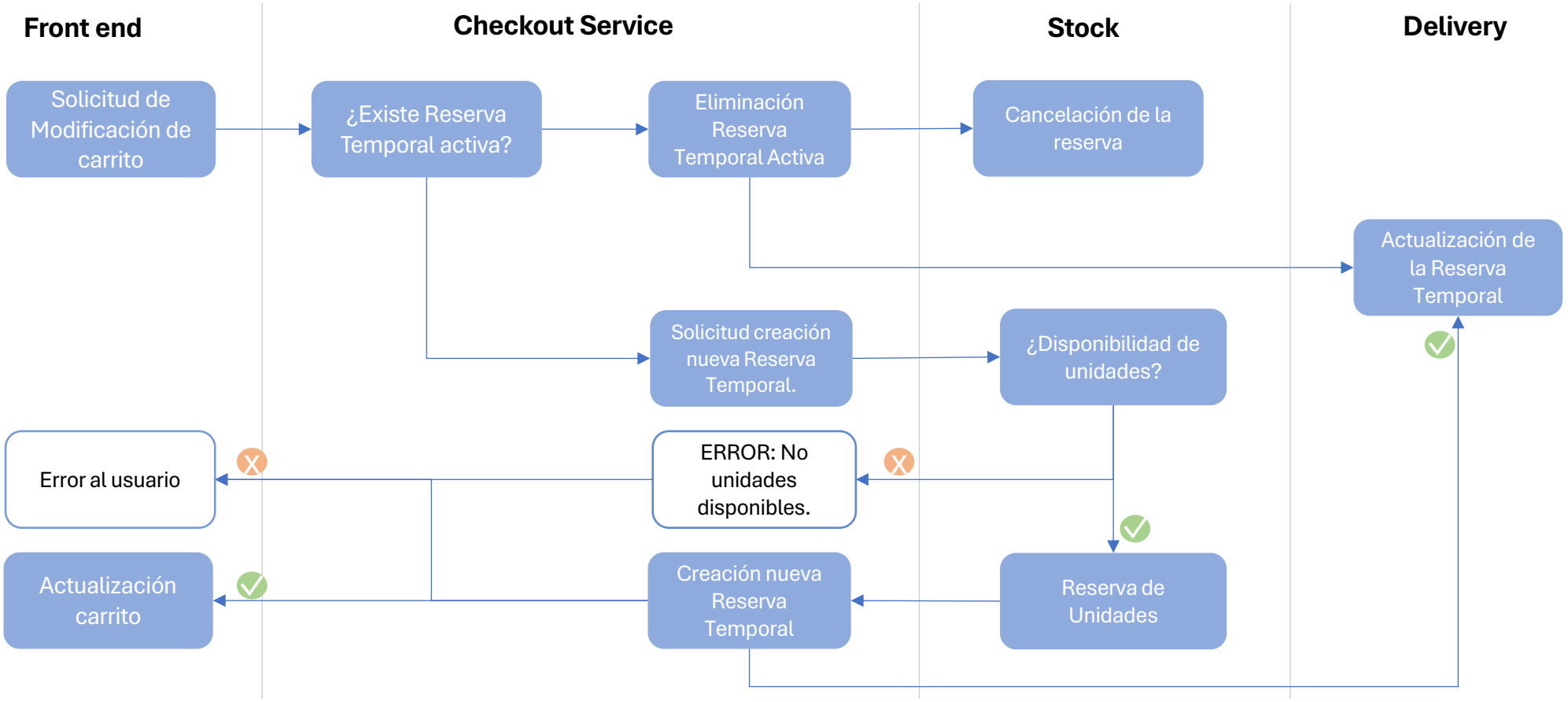
Caso 1 – Modificación de Reservas Temporales

Identificación de KPIs y justificación de negocio

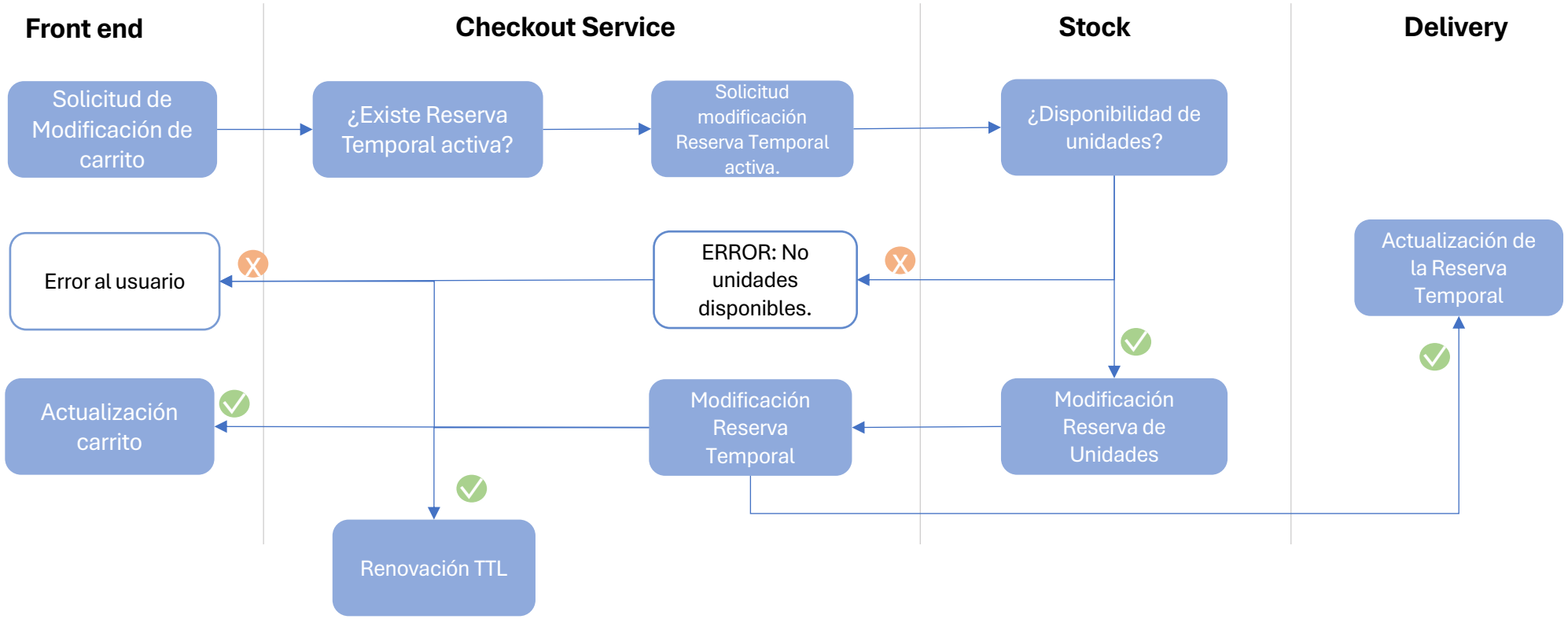
- **Hipótesis realizada:** nueva capacidad propuesta por el equipo de Checkout Service que ha identificado una oportunidad de mejora en esta parte del journey de reservas.
- Se realiza un análisis de los siguientes KPIs para justificar la **criticidad** de la capacidad de modificación de las Reservas Temporales:

KPI
Tasa de error en la reserva
Reservas modificadas para reducir unidades
Reservas modificadas para aumentar unidades
Pedidos perdidos por error de stock
Tiempo medio entre cancelación y nueva reserva
Impacto ingresos de ventas totales

Journey actual de modificación de RT



Propuesta de nuevo Journey



El **TTL** se actualiza tras la respuesta exitosa de updateReservation para evitar **bloqueo excesivo** de stock en caso de error del servicio.

Propuesta de nuevo endpoint

- Se va a crear un **nuevo endpoint en el API de "Reservations"** que gestiona Checkout Service. Suponemos que el API dispone de los endpoints: createReservation, deleteReservation y retrieveReservation.
- El nuevo endpoint será **updateReservation** y contará con los siguientes campos:

request

```
{
"timestamp"
"user_id"
"sku"
"old_quantity"
"new_quantity"
}
```

response (200 OK)

```
{
"reservation_id"
"sku"
"new_quantity"
"old_quantity"
"expires_at"
}
```

Este endpoint deberá ser documentado en el **Swagger del Checkout Service API Reservations**.

Gestión de Stakeholders

NEGOCIO

- Presentación nueva capacidad.
- Impacto en negocio.

AQUITECTURA CS

- Confirmación viabilidad técnica.
- Estimación de esfuerzo.

STOCK

- Consumo nuevo endpoint.
- Implementación lógica de actualización de stock.

ECOMMERCE

- Consumo nuevo endpoint.
- Coordinación mensajes de error.

DELIVERY

- Consumo nuevo endpoint.
- Implementación lógica de actualización.

OBSERVABILITY

- Consumo nuevo endpoint.
- Implementación lógica de actualización.

QA

- Pruebas unitarias.
- Pruebas e2e.
- Pruebas de performance.

SEGURIDAD

- Autenticación de llamadas al API.

Roadmap

Iteración	Objetivo	Items	Valor	KPIs
MVP	**Modificación** de la Reserva Temporal (RT).	Nuevo **endpoint** del API Reservations **updateReservations**	- Reduce fricciones en la experiencia de usuario. - Aumenta conversión de reservas en pedidos. - Evita bloqueo excesivo de stock.	Tasa de conversión de reservas modificadas. Aumento de facturación atribuible al cambio.
		Implementación de comprobación del **estado de la Reserva.**		
		Actualización **TTL** tras éxito del update.		
		Creación lógica de **actualización** de Reserva Temporal que mantenga la coherencia transaccional.		
2ª Iteración	**Eventos asíncronos** para reporting	Publicación de evento de reserva modificada por cola Kafka o en batches.	- Propagación de la información a sistemas de reporting o BI.	Tasa de éxito en el envío de eventos. Retraso en el procesamiento de la información.
3ª Iteración	**Dashboard**, **métricas** y **alarmas** de RTs modificadas.	Creación de métricas asociadas a los nuevos eventos.	- Monitorización del nuevo flujo de modificación de reservas temporales. - Detectar fallos, cuellos de botella y anomalías operativas.	KPIs del estado de salud del servicio: errores, tiempos de respuesta de los endpoints.
		Creación de dashboards que representen las métricas.		
		Nuevas alarmas para los errores del flujo.		

User Story 1: Actualización de la Reserva Temporal y TTL

- Como **Checkout Service** quiero **actualizar la reserva** para mantener coherencia con el resto del flujo de Checkout (Refresh, Check, Confirm).
- Criterios de aceptación:
  - DADA una reserva de estado temporal
    Y validaciones de stock disponible correctas
    CUANDO Checkout Service aplique la modificación
    ENTONCES actualiza se quantity y persiste la reserva.
  - DADO un cliente del API Reservations
    CUANDO la modificación se aplica correctamente
    ENTONCES CS devuelve el evento de modificación de reserva temporal
    Y actualiza el TTL de la reserva temporal.
  - DADO un cliente del API Reservations
    CUANDO el sistema detecta un error transaccional
    ENTONCES el endpoint devuelve un 500 Reservations.Update_failed
    Y la reserva queda sin cambios.

User Story 2: Gestión de stock y validaciones

- Como **Stock**, deseo recibir un **evento** en caso de modificación de la Reserva Temporal para comprobar la disponibilidad de los artículos y modificar la reserva de stock existente.
- Criterios de aceptación:
  - DADA una actualización de la reserva de stock
    CUANDO la reserva tenga el estado “temporal”
    Y las unidades de stock necesarias estén disponibles
    Y no haya errores transaccionales
    ENTONCES se generará un evento de actualización de la reserva temporal.
  - DADA una actualización de la reserva de stock
    CUANDO se compruebe que ya se había creado una reserva para los ítems
    Y la reserva tenga el estado “definitiva”
    ENTONCES se reportará 403 “Reservations.Reserva_definitiva”.
  - DADA una actualización de la reserva de stock
    CUANDO se compruebe que ya se había creado una reserva temporal para los ítems
    Y el número de unidades en stock es menor que el indicado en el carrito
    ENTONCES se reportará 403 “Reservations.Stock_insuficiente”.
  - DADA una actualización de la reserva de stock
    CUANDO se reciba un evento de modificación de reserva temporal
    ENTONCES el evento contendrá el identificador de usuario, el sku y el nuevo número de unidades.

User Story 3: Actualización del frontend

- Como **eCommerce**, deseo recibir un **evento** en caso de modificación del carrito para actualizar la disponibilidad de artículos en la cesta del usuario.
- Criterios de aceptación:
  - DADO QUE un cliente ha modificado las unidades del carrito
    CUANDO ya existía una reserva temporal previamente
    Y hay disponibilidad de las nuevas unidades
    ENTONCES se recibirá un evento con la información de la reserva actualizada.
  - DADO QUE un cliente ha modificado las unidades del carrito
    CUANDO ya existía un carrito previamente
    Y no hay disponibilidad de las nuevas unidades
    ENTONCES se recibirá un mensaje de error.

Caso 2 – Caída de conversión y degradación del rendimiento

Detección

- **KPIs/SLOs y herramientas**
  - Tasa de **conversión** a pedido confirmado por **Marca/Canal** para aislar el problema
    - Dashboards y métricas de negocio (p. ej: Grafana, Prometheus, Looker)
  - **Tasa de errores** 5xx por endpoint
    - Herramientas de monitoring (Grafana, Prometheus), logs para ver el mensaje de error.
  - **Latencia** en refresh y confirm.
    - Herramientas de monitoring (Grafana, Prometheus) o Elastic APM.
  - Saturación en recursos (CPU, conexiones a DB)
    - Dasboards de infraestructura.
  - Reintentos y timeouts
    - Logs de llamadas a las APIs y herramientas de monitoring.

Mitigación

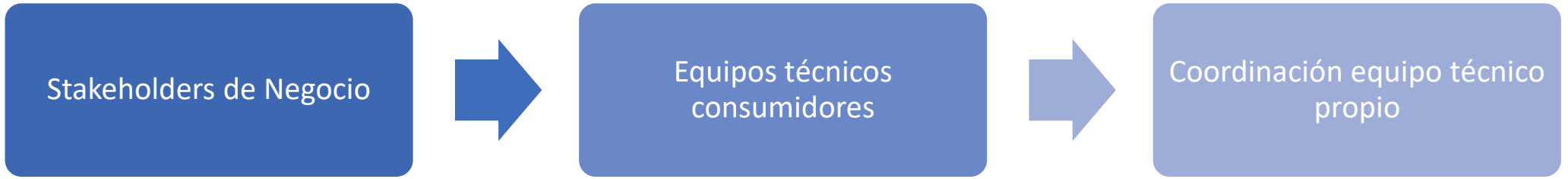
Acciones para recuperar estabilidad

- **Rollback** de despliegues recientes (en caso de haberlos) a versiones estables.
- Establecer mecanismos de **bypass** temporales a los componentes afectados.
- **Escalar** recursos críticos de forma temporal hasta encontrar la causa raíz.
- Documentar todo lo observado para análisis post-mortem.

Evitar futuras regresiones

- **Escalado** proactivo de recursos basado en KPIs de capacidad de la plataforma
- **Dashboards y alertas** en tiempo real que monitoricen los sistemas afectados para detectar problemas antes de que afecten al usuario.
- Revisar los pro**cesos de despliegue, pruebas y monitorización** incorporando los elementos identificados.
- Revisar y ajustar políticas de **timeouts y reintentos** para evitar saturación y errores repetidos.

Coordinación y comunicación



- Comunicación basada en datos procedentes del análisis técnico.
- Diagnosticar el impacto de negocio y criticidad.
- Comunicar el **Plan de Contención** y los avances.
- Gestionar **expectativas** de forma estratégica.

- Identificar posibles impactos en sus sistemas.
- Pausar o reintentar **operaciones que tengan en marcha** según las actuaciones de mitigación.
- Comunicar Plan de Contención y avances.

- Comunicación directa por canal ad hoc.
- Reuniones para alinear actuaciones
- Coordinar el diagnóstico de la **causa raíz**.
- **Priorización de tareas** para no sobrecargar al equipo.
- Reunión post-mortem.