

Documento de Diseño Actualizado – BoletaMaster - Grupo Hijos de Iván

1. Introducción

Este documento describe la arquitectura, decisiones de diseño y estructura funcional actualizada del sistema BoletaMaster, tras la incorporación del módulo de Marketplace y la clase Operación, además de las interfaces por consola para los tres tipos de usuario: Administrador, Organizador de Eventos (Promotor) y Cliente.

2. Enfoque general del sistema

El sistema BoletaMaster se diseñó para gestionar de forma integral la venta, transferencia, administración y reembolso de tickets de eventos. Se priorizó la separación por roles con responsabilidades claras: el Administrador supervisa y controla el sistema; el Organizador crea y gestiona eventos; y el Cliente adquiere, vende o transfiere tickets dentro del Marketplace.

3. Nuevos componentes introducidos

3.1 Clase Operación

La clase Operación centraliza la lógica del Marketplace, representando cualquier transacción activa o histórica entre clientes. Cada operación contiene información sobre el ticket involucrado, las partes (vendedor y comprador), la fecha y hora, el estado ("activa" o "finalizada"), y las posibles contraofertas. Permite registrar todas las interacciones comerciales en un log controlado por el Administrador.

3.2 Marketplace

El Marketplace es una extensión funcional del sistema que permite a los clientes publicar ofertas de tickets, comprar tickets de otros usuarios, realizar contraofertas y aceptar o rechazar propuestas. Todas las operaciones son auditadas por el Administrador a través de un registro histórico inmutable ("log_registros"),

garantizando trazabilidad y control sobre las transacciones.

4. Roles principales y sus responsabilidades

4.1 Administrador

- Aprueba o rechaza solicitudes de promotores para ingresar al Marketplace.
- Fija sobrecargos, cuotas fijas e imprime reportes financieros.
- Cancela eventos y autoriza reembolsos.
- Supervisa el log de operaciones (transacciones de tickets).
- Gestiona el registro y aprobación de venues.

4.2 Organizador de Eventos

- Crea y configura eventos, localidades y ofertas.
- Genera tickets simples o por paquete (Temporada, Deluxe).
- Sugiere venues y solicita cancelaciones de eventos.
- Puede participar en el Marketplace como vendedor.

4.3 Cliente

- Compra tickets y paquetes disponibles.
- Publica tickets en venta dentro del Marketplace.
- Realiza contraofertas, acepta o rechaza propuestas.
- Solicita reembolsos y consulta su saldo disponible.
- Se inscribe en el Marketplace para habilitar la compraventa de tickets.

5. Arquitectura y diseño modular

El sistema sigue un diseño modular dividido en capas principales:

- ****Capa de dominio:**** contiene las clases de negocio (Evento, Localidad, Venue, Tickete, Operación, Usuario y subclases).
- ****Capa de persistencia:**** maneja la carga y almacenamiento automático de datos mediante clases Persistencia*.
- ****Capa de presentación:**** implementada con interfaces de consola (una para cada tipo de usuario).
- ****Capa de control:**** integrada dentro de las clases de usuario (Administrador, OrganizadorDeEventos, Cliente) que invocan operaciones sobre los objetos del dominio.

6. Persistencia y manejo de datos

Los datos se cargan automáticamente desde archivos predeterminados al iniciar cada interfaz de usuario, gracias a las clases de persistencia: `PersistenciaUsuarios`, `PersistenciaEvento`, `PersistenciaLocalidad`, `PersistenciaVenu`, `PersistenciaOrganizadores` y `PersistenciaTiquete`. Los cambios (creación, compra o venta de tiquetes, nuevas solicitudes) se guardan de nuevo en los mismos archivos al cerrar sesión.

7. Interfaces de usuario por consola

Cada actor principal cuenta con su propia interfaz basada en consola, con autenticación inicial y menús jerárquicos:

- ****Administrador:**** opciones de aprobación, cancelación, consulta de log y gestión de venues.
- ****Organizador:**** opciones para crear eventos, localidades y tiquetes, y gestionar promociones o solicitudes.
- ****Cliente:**** opciones de compra, venta, contraoferta, inscripción y solicitud de reembolsos.

8. Decisiones de diseño actualizadas

- Se incorporó la clase `Operación` para manejar transacciones y registros históricos.
- Se añadió la estructura de `Marketplace` con log privado en `Administrador`.
- Se fortaleció el principio de encapsulamiento: los registros son privados y sólo accesibles mediante métodos de lectura.
- Se introdujeron `HashMap`s específicos por rol (eventos, solicitudes, clientes inscritos, operaciones activas).
- Las clases ahora incluyen validación de saldo, estado y permisos antes de ejecutar acciones sensibles.

9. Conclusión

El sistema `BoletaMaster` evolucionó hacia una solución más completa y trazable, donde cada actor mantiene sus responsabilidades delimitadas y las transacciones quedan registradas para garantizar seguridad y transparencia. El diseño actual soporta el flujo real de un ecosistema de boletería digital con capacidades

de mercado secundario (Marketplace).