Descripción breve

Estudio inicial de la arquitectura a seguir para el proyecto ‘Entrecine’

Proyecto Entrecine

v009

Claudio López Ardura; Diego Menéndez Díaz; David Santamaría Martin; Juan Manuel Sánchez Gómez; Alejandro Fernández Fernández; Miguel Ángel González Manchado

Contenido

[1 Planteamiento del Problema 2](#_Toc353013890)

[2 Metodología usada 2](#_Toc353013891)

[3 Identificación de los Stakeholders 2](#_Toc353013892)

[3.1 Empleados de taquilla 2](#_Toc353013893)

[3.2 Usuario de aplicación web 2](#_Toc353013894)

[3.3 Administrador del sistema 2](#_Toc353013895)

[3.4 Directivos 3](#_Toc353013896)

[3.5 Banco 3](#_Toc353013897)

[4 Atributos de Calidad 3](#_Toc353013898)

[5 Relación entre Stakeholders y Atributos de Calidad 3](#_Toc353013899)

[6 Descripción de la solución 4](#_Toc353013900)

[6.1 Negocio de las solución 5](#_Toc353013901)

[6.2 Funcionamiento del Servidor Web 5](#_Toc353013902)

[6.3 Funcionamiento del Sistema Central 5](#_Toc353013903)

[6.4 Funcionamiento del Sistema de Administración 6](#_Toc353013904)

[6.5 Funcionamiento del Sistema de Base de Datos 6](#_Toc353013905)

[7 Escenarios de Calidad 6](#_Toc353013906)

[8 Vistas 9](#_Toc353013907)

[8.1 System View 9](#_Toc353013908)

[8.1.1 Presentación principal 9](#_Toc353013909)

[8.1.2 Catálogo de elementos 10](#_Toc353013910)

[8.1.3 Diagrama Contextual 13](#_Toc353013911)

[8.1.4 Guía de puntos de variación 13](#_Toc353013912)

[8.1.5 Justificación de las decisiones 13](#_Toc353013913)

[8.2 WebServer View 13](#_Toc353013914)

[8.2.1 Presentación principal 14](#_Toc353013915)

[8.2.2 Catálogo de elementos 14](#_Toc353013916)

[8.2.3 Diagrama contextual 16](#_Toc353013917)

[8.2.4 Guía de puntos de variación 17](#_Toc353013918)

[8.2.5 Justificación de las decisiones 17](#_Toc353013919)

[8.3 Central System View 17](#_Toc353013920)

[8.3.1 Presentación principal 18](#_Toc353013921)

[8.3.2 Catálogo de elementos 18](#_Toc353013922)

[8.3.3 Diagrama contextual 25](#_Toc353013923)

[8.3.4 Guía de puntos de variación 25](#_Toc353013924)

[8.3.5 Justificación de las decisiones 25](#_Toc353013925)

[8.4 Administration System View 26](#_Toc353013926)

[8.4.1 Presentación principal 26](#_Toc353013927)

[8.4.2 Catálogo de elementos 26](#_Toc353013928)

[8.4.3 Diagrama contextual 29](#_Toc353013929)

[8.4.4 Guía de puntos de variación 29](#_Toc353013930)

[8.4.5 Justificación de las decisiones 29](#_Toc353013931)

[8.5 Database System View 30](#_Toc353013932)

[8.5.1 Presentación principal 30](#_Toc353013933)

[8.5.2 Catálogo de elementos 30](#_Toc353013934)

[8.5.3 Diagrama contextual 33](#_Toc353013935)

[8.5.4 Guía de puntos de variación 34](#_Toc353013936)

[8.5.5 Justificación de las decisiones 34](#_Toc353013937)

[8.6 PaymentGateway Adapter View 34](#_Toc353013938)

[8.6.1 Presentación principal 35](#_Toc353013939)

[8.6.2 Catálogo de elementos 35](#_Toc353013940)

[8.6.3 Diagrama contextual 36](#_Toc353013941)

[8.6.4 Guía de puntos de variación 36](#_Toc353013942)

[8.6.5 Justificación de las decisiones 36](#_Toc353013943)

[8.7 TPV Adapter View 36](#_Toc353013944)

[8.7.1 Presentación principal 37](#_Toc353013945)

[8.7.2 Catálogo de elementos 37](#_Toc353013946)

[8.7.3 Diagrama contextual 38](#_Toc353013947)

[8.7.4 Guía de puntos de variación 38](#_Toc353013948)

[8.7.5 Justificación de las decisiones 38](#_Toc353013949)

[8.8 Package Model View 38](#_Toc353013950)

[8.8.1 Presentación principal 39](#_Toc353013951)

[8.8.2 Catálogo de elementos 40](#_Toc353013952)

[8.8.3 Diagrama contextual 41](#_Toc353013953)

[8.8.4 Guía de puntos de variación 41](#_Toc353013954)

[8.8.5 Justificación de las decisiones 41](#_Toc353013955)

[8.9 Deployment View 41](#_Toc353013956)

[8.9.1 Presentación principal 42](#_Toc353013957)

[8.9.2 Catálogo de elementos 42](#_Toc353013958)

[8.9.3 Diagrama contextual 43](#_Toc353013959)

[8.9.4 Guía de puntos de variación 43](#_Toc353013960)

[8.9.5 Justificación de las decisiones 43](#_Toc353013961)

[9 Bibliografía 44](#_Toc353013962)

[10 Índice de figuras 44](#_Toc353013963)

[11 Glosario 44](#_Toc353013964)

[12 Índice de palabras 45](#_Toc353013965)

# Planteamiento del Problema

Se necesita construir un sistema que permita gestionar automáticamente la venta de entradas de un cine. La venta de las entradas se podrá hacer vía web o desde taquilla.

# Metodología usada

Se va a realizar un estudio de arquitectura siguiendo el método de ADD (Atribute-Driven Design) y la norma del SEI (ANSI/IEEE 1471, 2000).

# Identificación de los Stakeholders

El sistema tiene una serie de interesados que incurrirán en las decisiones del diseño.

|  |  |
| --- | --- |
| Código | Stakeholder |
| ST-01 | Empleados de taquilla |
| ST-02 | Usuario de aplicación web |
| ST-03 | Administrador del sistema |
| ST-04 | Directivos del cine |
| ST-05 | Banco |

## Empleados de taquilla

Trabajadores del cine. Se encargan de atender a los clientes desde la taquilla, vendiendo las entradas en metálico o por tarjeta de crédito. Sus intereses son:

* Control y accesos rápidos entre sesiones y películas para despachar a los clientes en el menor tiempo posible.

## Usuario de aplicación web

Son los clientes propiamente dichos. Aunque tenga palabra directa en las decisiones sobre la arquitectura, es necesario tenerlos en cuenta porque forman parte una parte esencial del sistema. Los intereses que comúnmente se asocian con los clientes son:

* Fácil acceso a la cartelera del cine, sin navegaciones innecesarias ni información de difícil acceso.
* Facilidad en el pago mediante tarjeta de crédito.
* Control sobre sus datos registrados en el sistema Entrecine.

## Administrador del sistema

Es el trabajador del cine que se encarga de mantener actualizados los datos sobre las películas y sesiones, así como los precios. También tiene acceso a los datos de los usuarios y al control de las transacciones bancarias. Sus intereses son:

* Manejo *CRUD* sobre todos los datos del sistema.
* Configuración de los parámetros de la aplicación.

## Directivos

Se trata del equipo directivo del cine. Son responsables de los presupuestos y toman las decisiones que comprometen fondos de dicho presupuesto. Sus intereses son:

* Coste del desarrollo de la aplicación bajo.
* Tiempo de desarrollo de la aplicación corto.

## Banco

Aunque no actúe directamente sobre el sistema, es necesario tener relación con él porque es el que proporciona los métodos para los pagos vía tarjeta de crédito. Sus intereses son:

* Ofrece una pasarela de pago con la que es sistema debe comunicarse para la realización de los pagos.

# Atributos de Calidad

Para el sistema se ha definido los siguientes atributos de calidad:

|  |  |  |
| --- | --- | --- |
| Código | Descripción | Tipo |
| AT-01 | Venta online de entradas 24x7 | Disponibilidad |
| AT-02 | Actualización de cartelera y precios inmediata | Modificabilidad |
| AT-03 | Robustez de la aplicación en horas putas | Rendimiento |
| AT-04 | Concurrencia de ventas | Rendimiento |
| AT-05 | Realización rápida de ventas desde la TPV | Rendimiento |
| AT-06 | Conexión segura con la pasarela de pago | Seguridad |
| AT-07 | Protección de datos de los usuarios | Seguridad |
| AT-08 | Funcionalidad de la página web en los navegadores más comunes | Adaptabilidad |
| AT-09 | Fácil manejo de la web y de la compra de entradas | Usabilidad |
| AT-10 | Administración completa y de manejo sencillo | Usabilidad |
| AT-11 | Comprobaciones de que el comprobante de compra se corresponda con las ventas | Testabilidad |
| AT-12 | Tiempo de desarrollo corto | Time to market |
| AT-13 | Coste de desarrollo bajo | Coste-Beneficio |

# Relación entre Stakeholders y Atributos de Calidad

Los stakeholder y los atributos de calidad están relacionados directamente, siendo los intereses de los stakeholders la principal razón de inclusión de atributos de calidad que en un principio pudieron no tenerse en cuenta. Esta tabla muestra la relación entre ambos:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ST-01 | ST-02 | ST-03 | ST-04 | ST-05 |
| AT-01 |  | **X** |  | **X** |  |
| AT-02 |  | **X** | **X** |  |  |
| AT-03 |  | **X** |  |  |  |
| AT-04 | **X** | **X** |  |  |  |
| AT-05 | **X** | **X** |  |  |  |
| AT-06 |  | **X** |  |  | **X** |
| AT-07 |  | **X** | **X** |  |  |
| AT-08 |  | **X** |  |  |  |
| AT-09 |  | **X** |  |  |  |
| AT-10 |  |  | **X** |  |  |
| AT-11 | **X** | **X** |  |  |  |
| AT-12 |  |  |  | **X** |  |
| AT-13 |  |  |  | **X** |  |

# Descripción de la solución

El modelo de negocio de la aplicación estará constituido por un servicio web desde el cual un usuario puede ver la cartelera del cine y sus horarios, además de poder comprar entradas para las sesiones. También es posible comprar las entradas desde la taquilla del cine, cuyas TPVs estarán conectadas también con el sistema.

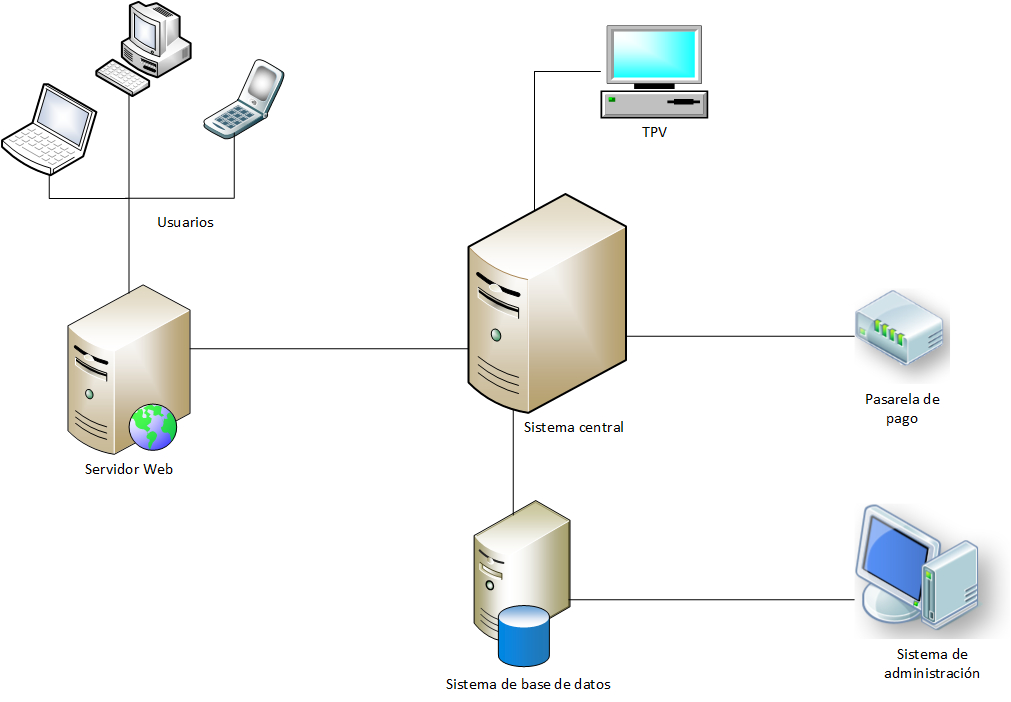


Figura 1. Presentación del modelo de negocio

## Negocio de las solución

Existen 4 sistemas:

* Servidor Web
  + Ofrece la página web a los usuarios.
  + Es posible conectarse a ella desde cualquier tipo de dispositivo.
  + Permite comprar entradas para cualquier sesión de las películas que se proyecten en ese día y los 7 siguientes.
  + Permite registrarse para no tener que rellenar los datos de pago cada vez que se compre una entrada.
* Sistema central
  + Se encarga de gestionar todas las ventas que se produzcan en el sistema, ya sean por web o por TPV.
  + Realiza los pagos de la compra de entradas a través de una pasarela de pago.
  + Gestiona los usuarios en sesión.
* Sistema de administración
  + Modifica cartelera de la base de datos.
  + Visualiza los datos de las ventas y los usuarios.
* Sistema de base de datos
  + Guarda los datos de todas las ventas realizadas en el sistema.
  + Guarda la información de los usuarios.
  + Protege los datos de los usuarios y las ventas mediante encriptaciones.
  + Almacena la cartelera.

## Funcionamiento del Servidor Web

Recibe las peticiones de los usuarios para mostrar la página web permitiéndoles navegar por toda la cartelera. Cuando un usuario quiere comprar una entra, selecciona la sesión que le interesa. Rellena los datos sobre la tarjeta de crédito y envía la petición al servidor. Este traslada la petición de la compra al sistema central. Si todo finaliza correctamente el usuario obtiene el comprobante de la entrada, el cual tendrá que imprimir obligatoriamente.

Un usuario puede registrarse en el sistema rellenado un formulario. El servidor valida el formulario indicando los errores en caso de haberlos. Una vez validado envía la petición de registro al sistema central que finalizará el registro.

Un usuario registrado puede realizar compras sin tener que volver a rellenar los datos. Para acceder a esta funcionalidad primer hay que loguearse en el sistema, petición que el servidor traslada al sistema central.

La primera vez que se arranca un servidor web y se conecta al sistema central se suscribe a las actualizaciones de la cartelera en el sistema central. Cuando se produce una, el servidor solicita al sistema central la nueva cartelera.

## Funcionamiento del Sistema Central

El sistema central gestiona las ventas. Recibe las peticiones de la venta de entradas desde los servidores web y las TPVs. Los pagos con tarjeta se llevan a cabo a través de una pasarela de pago.

En caso de que la venta sea online, obtiene el comprobante de la entrada para enviárselo al usuario. Este comprobante es escaneado en una TPV y se pide al sistema central que lo valide para cancelar la entrada. Todos los comprobantes creados se envían para que se guarden en el sistema de base de datos.

Todos las ventas, ya sean por tarjeta de crédito pagadas en metálico desde la TPV se enviar para ser guardadas en el sistema de base de datos.

Si recibe una petición de registro comprueba que no exista ya un usuario con el mismo identificador y lo envía a guardar. Gestiona los inicio de sesión de los usuarios validando la contraseña. Si un usuario lleva más de 30 minutos sin actividad cierra automáticamente la sesión.

Tanto de un TPV como de un servidor puede recibir la petición de obtener la cartelera. Cuando esto ocurre transforma los datos de la cartelera, tras obtenerlos del sistema de base de datos, en un formato que el servidor/TPV pueda interpretar. Para que un servidor reciba la cartelera primero tuvo que pedirle al sistema central que lo suscribiese a las actualizaciones, momento en el que la petición se reenvía al sistema de bases de datos.

## Funcionamiento del Sistema de Administración

La administración se encarga de enviar las modificaciones a la cartelera. Primero obtiene del sistema de base de datos la cartelera antigua y sobre ella el administrador realiza las modificaciones y añadidos pertinentes.

Tras un periodo de tiempo o cuando la cantidad de modificaciones a alcanzado cierto nivel, el sistema de administración envía las actualizaciones al sistema de base de datos. El administrador puede cancelar las actualizaciones antes de que estas se envíen.

Adicionalmente el administrador puede acceder al registro de usuarios y ventas para poder visualizar las preferencias de los mismos.

## Funcionamiento del Sistema de Base de Datos

El sistema de base de datos almacena la información de todo el sistema. También se encarga de encriptar los datos de los usuarios y de las ventas para protegerlos.

Recibe peticiones para almacenar comprobantes, usuarios y ventas del sistema central, los cuales encripta antes de guardarlo en la base de datos. También recibe peticiones para enviar datos almacenados en la base de datos, desencriptándolos antes de enviarlos.

Del sistema central recibe los servidores web que quieren suscribirse a las actualizaciones de la cartelera. Cuando el administrador modifica algo en ella, el sistema de base de datos avisa a todos los servidores suscriptores de que necesitan actualizar su cartelera.

# Escenarios de Calidad

A partir de toda la información anterior se definen los escenarios de calidad del sistema. La tabla siguiente muestra los escenarios y los atributos con los que están relacionados.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Escenario Nº | Fuente del estímulo | Estímulo | Entorno | Artefacto | Respuesta | Medición de la respuesta | Atributo de calidad asociado |
| 1 | Página web | Atención a peticiones | Explotación | Servidor | Atención 24x7 | Denegaciones de servicio < 1% | **AT-01** |
| 2 | Cambio de cartelera | Actualización de la cartelera | Explotación | Servidor, Sistema de Administración | Actualización de la cartelera y precios | Ventas en cartelera desactualizada: 0 | **AT-02** |
| 3 | Hora punta de ventas | Múltiples accesos al servicio web | Explotación | Servidor, Sistema central | Activación de servidores espejo | Ralentizaciones y errores < 5% | **AT-03** |
| 4 | Venta de entrada | Acceso concurrente a misma butaca | Explotación | Servidor, Sistema central | Venta de la butaca a un solo usuario | Errores en ventas: 0 | **AT-04** |
| 5 | TPV | Venta de entradas en taquilla | Explotación | TPV, Sistema central | Cliente despacho rápidamente | Tiempo de venta en taquilla < 30s | **AT-05** |
| 6 | Venta de entrada | Pago con tarjeta de crédito | Explotación | Sistema central | Transacción segura con el banco | Cargos inválidos: 0 | **AT-06** |
| 7 | Acceso a datos del usuario | Acceso a información | Explotación | Servidor | Acceso por contraseña | Los usuarios solo pueden acceder a sus datos | **AT-07** |
| 8 | Uso de página web | Envío de página web | Explotación | Servidor | Página web visualizada correctamente | Debe ser usable en IE, Firefox, Chrome… | **AT-08** |
| 9 | Página web | Acceso a la cartelera y ventas | Explotación | Servidor | Información del cine fácilmente accesible | Perdidas de clientes < 5% | **AT-09** |
| 10 | Sistema de administración | Acceso y cambios en el sistema | Explotación | Sistema de administración, sistema de persistencia | Actualizaciones rápidas | Tiempo en actualizar cartelera < 5min/película | **AT-10** |
| 11 | TPV | Comprobante de compra online | Explotación | TPV, Sistema central | Validación y entrega de entrada | Errores de comprobación: 0 | **AT-11** |
| 12 | Sistema en desarrollo | Sistema en desarrollo | Desarrollo | Sistema | Tiempo bajo de desarrollo | Time to market < 2 meses | **AT-12** |
| 13 | Sistema en desarrollo | Sistema en desarrollo | Desarrollo | Sistema | Coste bajo de producción | Coste de desarrollo: 0€ | **AT-13** |

# Vistas

En este apartado se realizara la descripción de las vistas del sistema y de sus subsistemas, así como la vista de despliegue del sistema y su empaquetamiento para la implementación.

|  |  |  |
| --- | --- | --- |
| Vistas | Referencia | Escenarios |
| System |  | 2,5,6 |
| WebServer |  | 1,3,7,8,9 |
| Central System |  | 3,4,5,6,11 |
| Administration System |  | 2,10 |
| Database System |  | 2,7,10 |
| PaymentGateway Adapter |  | 6 |
| TPV Adapter |  | 5,6,11 |
| Package Model |  | 12,13 |
| Deployment |  |  |

## System View

La vista del sistema describe los 4 subsistemas, *WebServer, Central System, Database System y Administration System,* y las interfaces con las que se comunican.

### Presentación principal



Figura 2. System View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Vista | Propiedades |
| WebServer | Subsistema encargado de interactuar con el cliente. |
| Central System | Subsistema que gestiona todo el sistema de ventas. |
| Administration System | Subsistema dedicado a la gestión de sistema. |
| Database System | Subsistema que protege los datos y los almacena. |

#### Relaciones

*WebServer* interactúa con *Central System* para obtener los datos necesarios para el usuario y realizar compras.

*Central System* se comunica con *Database System* para obtener y salvar toda la información procesada.

*Administration System* se encarga de modificar los datos con lo que trabaja *Database System.*

#### Interfaces/Puertos

##### WebServer

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| HttpRequest | Proveída | Web Service | Recibe las peticiones del navegador del cliente. |
| Register | Requerida | RMI | Solicita la creación de un nuevo usuario registrado en el sistema. |
| Login | Requerida | RMI | Solicita el inició de sesión de un usuario. |
| RequestBillboard | Requerida | RMI | Solicita la actualización de la cartelera. Esta actualización se solicitará cuando se realice una modificación en la cartelera. |
| Payment | Requerida | RMI | Solicita la realización de la compra de una entrada y del pago con tarjeta de crédito a través de la pasarela de pago. |
| ReceiveTicket | Requerida | RMI | Solicita la obtención del comprobante de la compra de la entrada vía web. |
| Connect | Requerida | RMI | Solicita la conexión del servidor con las actualizaciones de la cartelera. |

##### Central System

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades | |
| Register | Proveída | RMI | Reciba la creación de un nuevo usuario en el sistema. | |
| Login | Proveída | RMI | Recibe la solicitud de inicio de sesión. Si el usuario se mantiene inactivo más de 30 minutos se cierra la sesión. | |
| RequestBillboard | Proveída | RMI | Recibe una petición de actualización de la cartelera. Si la cartelera ha sido actualizada se envía toda la nueva información. | |
| Payment | Proveída | RMI | Recibe la solicitud de compra de una entrada y del pago a través de la pasarela de pago.  Cuando se intente comprar una entrada de una butaca ya vendida se impedirá la venta.  Si el pago es realizado por un usuario registrado se asocia la venta al mismo. | |
| ReceiveTicket | Proveída | RMI | | Recibe la solicitud de la entrega del comprobante de compra online. |
| Connect | Proveída | RMI | | Recibe la solicitud de conexión con las actualizaciones de la cartelera. |
| SaveUser | Requerida | RMI | | Solicita el almacenamiento del usuario recién registrado y creado. |
| LoadUser | Requerida | RMI | | Solicita la carga de los datos de un usuario que inicia sesión. |
| GetTicket | Requerida | RMI | | Solicita la obtención del comprobante y la venta de entrada del código escaneado. |
| SaveTicket | Requerida | RMI | | Solicita el almacenamiento del comprobante recién creado y asociado a una venta online. |
| SaveSale | Requerida | RMI | | Solicita el almacenamiento en la base de datos de una venta realizada. |
| LoadBillboard | Requerida | RMI | | Solicita la carga de la cartelera. |
| ConnectBillboard | Requerida | RMI | | Solicita la conexión de un servidor que haya realizado con las actualizaciones de la cartelera. |
| TPVRequest | Proveída | RMI | | Recibe las solicitudes de la TPV para la realización de los pagos, obtención de cartelera y cancelaciones de comprobantes. |
| DoPayment | Requerida | RMI | | Solicita la realización de un pago de tarjeta de crédito a través de la pasarela de pago. |

##### Administration System

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetUsers | Requerida | Method Call | Solicita la obtención de toda la información de los usuarios. |
| GetSales | Requerida | Method Call | Solicita la obtención de las entradas vendidas |
| GetRawBillboard | Requerida | Method Call | Solicita la cartelera en un formato editable. |
| UpdateBillboard | Requerida | Method Call | Actualiza la cartelera con los datos nuevos o con los antiguos modificados. |

##### Database System

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| SaveUser | Proveída | RMI | Recibe la solicitud del almacenamiento de un nuevo usuario en la base de datos.  Antes de guardarlo encripta los datos para protegerlos. |
| LoadUser | Proveída | RMI | Recibe la solicitud de carga de los datos de un usuario de la base de datos.  Al obtener los datos los desencripta para que puedan ser entendidos. |
| GetTicket | Proveída | RMI | Recibe la solicitud de obtención del comprobante de una compra online. |
| SaveTicket | Proveída | RMI | Recibe la solicitud del almacenamiento del comprobante de una venta online. |
| SaveSale | Proveída | RMI | Recibe la solicitud del almacenamiento de una nueva venta.  Antes de guarda los datos los encripta para proteger las ventas. |
| LoadBillboard | Proveída | RMI | Recibe la solicitud de carga la información de la cartelera de la base de datos. |
| ConnectBillboard | Proveída | RMI | Recibe la solicitud de conectar un servidor con las actualizaciones de la cartelera |
| GetUsers | Proveída | Method Call | Recibe la solicitud de obtención de la información de todos los usuarios.  Cuando los obtiene los desencripta. |
| GetSales | Proveída | Method Call | Recibe la solicitud de obtención de información sobre las entradas vendidas.  Cuando obtiene las ventas las desencripta. |
| GetRawBillboard | Proveída | Method Call | Recibe la solicitud de obtención de la cartelera en formato editable. |
| UpdateBillboard | Proveída | Method Call | Recibe la solicitud de modificación de la cartelera. Cuando esta actualización ocurre se notifica a los servidores conectados la actualización. |

#### Comportamiento

##### WebServer

Se encarga de atender las peticiones de los usuarios a través de la web. Tiene una copia de la cartelera en una base de datos nativa, que se actualiza en tiempo real.

Cuando el usuario quiere comprar una entrada se conecta al sistema central para llevar a la venta, proveyendo al usuario la interfaz para la realización de un pago con tarjeta de crédito.

Al finalizar una venta obliga a la impresión del comprobante de la entrada comprada.

##### Central System

Es el sistema que rige todas las ventas. Tanto los servidores como las TPVs acceden a él a través de un adaptador que permiten adaptar numerosas peticiones al subsistema aliviando la carga de trabajo gestionándola con un pool de accesos.

Comprueba que las entradas vendidas no sean sobre una misma butaca.

Gestiona los accesos de los usuarios registrados y crea usuarios nuevos cuando es necesario.

Gestiona las peticiones de los servidores y TPVs para la actualización/obtención de la cartelera.

Se conecta con la pasarela de la pago para la realización de pagos con tarjeta de crédito con el banco.

##### Administration System

Es la interfaz de administración del sistema. Actúa directamente con *Database System* para modificar los parámetros de la cartelera, así como para obtener la información de las entradas vendidas y de los usuarios.

##### Database System

Es un sistema dedicado a proteger toda la información de la base de datos. Encripta ´y desencripta la información de los usuarios y las ventas para trabajar con ellas desde el resto de sistemas.

Contiene la cartelera en la forma primitiva, de forma que la el tamaño de los datos que tiene sea mínimo, y se encarga de transformarla para enviarla a los otros sistemas.

### Diagrama Contextual

N/A

### Guía de puntos de variación

N/A

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 2 | **AT-02** | Las interfaces ***“Connect “y*** ***“ConnectBillboard”*** permiten mantener actualizada la cartelera en tiempo real una vez un *WebServer* se haya conectado a *Central System.* |
| 5 | **AT-05** | Las interfaces de acceso de la TPV permiten que varias puedan conectarse a *Central System* y poder realizar las ventas y cancelaciones de entradas. |
| 6 | **AT-06** | La interfaz ***“GatewayPayment”*** permite conectarsecon la pasarela de pago suministrada por el banco para la realización de pagos con tarjeta de crédito. |

## WebServer View

La vista de *WebServer* muestra el primer nivel de descripción de los componentes de *WebServer.*

### Presentación principal



Figura 3. WebServer View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| PresentacionManager | Es la capa de presentación del servidor web. |
| ServerLogicManager | Capa lógica del servidor |
| ServerBillboardManager | Encargado de actualizar la base de datos con la cartelera |
| ServerBillboardDB | Base de datos nativa con la cartelera |

#### Relaciones

*PresentationManager* se comunica con *ServerLogicManager* para pasarle las interacciones del usuario que necesiten de cómputos lógicos, y para obtener la cartelera.

*ServerLogicManager* pide a *ServerBillboardDB* la cartelera.

*ServerBillboardManager* se encarga de actualizar la cartelera almacenada en *ServerBillboardDB*.

#### Interfaces/Puertos

##### PresentationManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| HttpRequest | Proveída | WebService | Ver 8.1.2.3.1. |
| UserAction | Puerto (Out) | Method Call | Solicita computar las acciones del usuario que necesiten llevar a cabo de manera separada a la capa de presentación. |
| ObtainBillboard | Puerto (In-Out) | Method Call | Recoge bajo demanda la cartelera del cine. |

##### ServerLogicManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| Register | Requerida | RMI | Ver 8.1.2.3.1. |
| Login | Requerida | RMI | Ver 8.1.2.3.1. |
| ServerPayment | Requerida | RMI | Ver 8.1.2.3.1. |
| ObtainTicket | Requerida | RMI | Ver 8.1.2.3.1. |
| GetBillboard | Puerto (In-Out) | DAO (ODBC) | Solicita a la base de datos la obtención de la cartelera actualizada.  Si la no se han producido cambios no pasa nada. |
| UserAction | Puerto (In) | Method Call | Realiza las acciones del usuario necesite. |

##### ServerBillboardManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| RequestBillboard | Requerida | RMI | Ver 8.1.2.3.1. |
| Connect | Requerida | RMI | Ver 8.1.2.3.1. |
| SetBillboard | Puerto (Out) | DAO (ODBC) | Solicita modificar la base de dato en cuando se produce una actualización de la base de datos general. |

##### ServerBillboardDB

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetBillboard | Acceso DB | DAO (ODBC) | Devuelve los datos actualizados de la cartelera. SI no se han producido cambios desde el último acceso no se producen cambios. |
| SetBillboard | Acceso DB | DAO (ODBC) | Añade e introduce las modificaciones de la base de datos que se hayan producido. |

#### Comportamiento

##### PresentationManager

Este módulo se encarga de la presentación del que verá el usuario en su navegador.

En función del navegador usado, el sistema devolverá a través de la interfaz *HttpRequest*  la versión de la página que mejor se ajuste al navegador.

Para mostrar la página, necesita cargar la cartelera (*ObtainBillboard*) y con ella trabaja para que el usuario pueda interactuar con las distintas entidades que las conforman.

Cuando un usuario realiza una acción, como por ejemplo registrarse, traslada la petición con la interfaz *UserAction.*

##### ServerLogicManager

Se encarga de la lógica realizada por el servidor.

Cuando la capa de presentación necesita la cartelera (*ObtainBillboard*) pide, mediante un *DAO* a la base de datos toda la información que necesite (*GetBillboard*). De la cartelera obtenida coge solo los datos de los próximos 7 días, los cuales envías a *PresentationManager* para que se lo muestre al usuario.

Cuando recibe acciones del usuario desde *PresentationManager* (*UserAction*)*,* puede desencadenar la necesidad de realizar acciones ajenas al módulo. Cuando se trata de una acción de registro, comprueba que los datos sean válidos y los envía al siguiente subsistema. Lo mismo pasa con *Login,* que preara los datos para un inicio de sesión; *ServerPayment,* validando que el número de tarjeta de crédito sea correcto; y *ObtainTicket*, que se desencadena tras *ServerPayment* y obliga a imprimir el comprobante tras obtenerlo.

##### ServerBillboardManager

Este módulo mantiene la base de datos al día.

Mediante la interfaz *Connect* se registra en el sistema de subscripciones de la actualización de la cartelera, permitiendo que esta se actualice cuando se realiza algún cambio.

Cuando se produce una actualización el módulo realiza una petición (*RequestBillboard*) para obtener la información necesaria de cartelera y la almacena en mediante una *DAO*  en la base de datos del servidor (*SetBillboard*)*.*

##### ServerBillboardDB

Gestiona el almacenamiento de datos de cartelera en una base de datos nativa.

Permite modificar y añadir elementos a la cartelera (*SetBillboard*) y recuperar los datos de la misma (*GetBillboard*).

### Diagrama contextual

Esta vista es el desarrollo del subsistema *WebServer.* (Ver 8.1.1)

### Guía de puntos de variación

*PresentationManager* y *ServerLogicManager* pueden estar implementados mediante un Modelo Vista-Controlador y actuando como un módulo único, pero con una clara separación de las capas.

El *ServerBillboardManager* debería esta implementado por un patrón *Observer* que permita estar suscrito a la actualización de la cartelera, utilizando un *Listener* como elemento de paso por la interfaz *Connect*  y que al ser notificado cause la llamada de *RequestBillboard.*

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 1 | **AT-01** | El subsistema *WebServer* está pensado de manera que puedan existir múltiples servidores en el sistema y en caso de que una fallo haya otro para atender las peticiones de los usuarios |
| 3 | **AT-03** | Debido a la independencia de este subsistema frente al sistema, en caso de necesidad, se pueden añadir servidores para poder soportar una mayor carga de trabajo. Además debido a la disposición de la base de datos nativa, no necesita de acceder al resto del sistema para mostrarla al usuario, por lo que disminuye también los cuellos de botella. |
| 7 | **AT-07** | Un usuario en sesión tiene acceso a sus datos, pero estos se encuentran en el servidor, de manera que solo ese usuario en esa sesión puede acceder a ellos. En caso de que una sesión se encuentre inactiva mucho rato esta se cierra, protegiendo los datos del usuario. |
| 8 | **AT-08** | Los navegadores actuales son muy heterogéneos, puesto que no es lo mismo acceder desde un ordenador que desde un dispositivo móvil, por lo que manteniendo una capa de presentación que sea capaz de identificar el navegador y responder en consecuencia, permitiendo una mejor visualización en una mayor gama de dispositivos. |
| 9 | **AT-09** | En la capa de presentación debe tenerse en cuenta la usabilidad del sistema, por lo que manteniéndola aislada en un módulo es posible modificarla en función de la necesidad sin tener que modificar otros módulos. |

## Central System View

La vista de *Central System* muestra el primer nivel de descripción de los componentes de *Central System.*

### Presentación principal



Figura 4. Central System View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| WebServerGateway | Balancea la carga de trabajo de las peticiones recibidas de los servidores para distribuirla en el sistema. |
| TPVGateway | Balancea la carga de trabajo de las peticiones recibidas desde las TPV para distribuirla por el sistema. |
| PaymentManager | Gestiona las venta de entradas en el sistema |
| TicketManager | Crea los comprobantes de venta de entradas y realiza sus validaciones. |
| CreditCardManager | Maneja las ventas realizadas con tarjeta de crédito. |
| CashManager | Maneja las ventas en metálico. |
| UserManager | Gestiona la información de los usuarios. |
| BillboardManager | Adapta las peticiones de la cartelera a las peticiones que se realicen. |
| SalesManager | Adapta las ventas a para que sean enviadas a la base de datos como. |

#### Relaciones

*WebServerGateway* pasa los registros y login de los usuarios a *UserManager* para que los trate. Para realizar los pagos con tarjeta de crédito se relaciona con *PaymentManager,* y también para obtener los comprobantes de las entradas vendidas. Para obtener la cartelera primero se conecta a las suscripciones y luego solicita la cartelera, ambas a través de *BillboardManager.*

*TPVGateway* se relaciona con *PaymentManager*  para llevar a cabo los pagos con tarjeta de crédito y para registrar las ventas en metálico, así como para cancelar entradas de cine. Se conecta con *BillboardManager* para obtener la cartelera.

*PaymentManager* envía los datos de las ventas a través de *SalesManager* para que sean guardadas. Pasa las peticiones de pagos con tarjeta de crédito a *CreditCardManager*, y las ventas metálico a *CashManager. CreditCardManager* también recibe las peticiones de obtención de comprobantes de *PaymentManager.*

*CreditCardManager* envía a *PaymentManager*  las peticiones para conectarse a la pasarela de pago.

*TicketManager* recibe las solicitudes para realizar cancelaciones de entradas vendidas a través de *PaymentManager*, y la creación de los comprobantes de *CreditCardManager.* Además envía a *PaymentManager* los mensajes para guardar u obtener los comprobantes de la base de datos.

#### Interfaces/Puertos

##### WebServerGateway

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| RequestBillboard | Proveída | RMI | Ver 8.1.2.3.2. |
| Register | Proveída | RMI | Ver 8.1.2.3.2. |
| Login | Proveída | RMI | Ver 8.1.2.3.2. |
| Payment | Proveída | RMI | Ver 8.1.2.3.2. |
| ReceiveTicket | Proveída | RMI | Ver 8.1.2.3.2. |
| Connect | Proveída | RMI | Ver 8.1.2.3.2. |
| RegisterUser | Puerto (Out) | Message queue  Transaction | Envía el mensaje de registro de un nuevo usuario en el sistema. |
| LoginUser | Puerto (In-Out) | Message queue  Transaction | Solicita el inicio de sesión de un usuario registrado. |
| ServerPayment | Puerto (Out) | Message queue  Transaction | Envía el mensaje para la realización de un pago a través de la pasarela de pago. |
| ObtainTicket | Puerto (In-Out) | Message queue  Transaction | Solicita la obtención del comprobante de la entrada comprada para imprimirlo. |
| ConnectServer | Puerto (Out) | Message queue  Transaction | Solicita la conexión del servidor con las actualizaciones de la cartelera. |
| GetServer Billboard | Puerto(In-Out) | Message queue  Transaction | Solicita la obtención de la cartelera. |

##### TPVGateway

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| TPVRequest | Proveída |  | Ver 8.1.2.3.2. |
| CashPayment | Puerto (Out) | Message queue  Transaction | Envía el mensaje de la realización de un pago en metálico, bloqueando la butaca donde de la entrada vendida. |
| CreditCard Payment | Puerto (Out) | Message queue  Transaction | Solicita el pago a través de la pasarela de pago, bloqueando la butaca donde de la entrada vendida. |
| CheckTicket | Puerto(In-Out) | Message queue  Transaction | Solicita la validación del comprobante de una entrada vendida online. |
| GetTPVBillboard | Puerto(In-Out) | Message queue  Transaction | Solicita la obtención de la información que usará la TPV para mostrar la cartelera. |

##### PaymentManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| DoPayment | Proveída |  | Ver 8.1.2.3.2. |
| ServerPayment | Puerto (In) | Method Call | Recibe la petición del WebServer para realizar un pago a través de la pasarela de pago. |
| ObtainTicket | Puerto (In) | Method Call | Recibe la petición para enviar el comprobante de la venta de una entrada. |
| CheckTicket | Puerto (In-Out) | Method Call | Recibe la petición para validar el comprobante de una entrada vendida. |
| CreditCard Payment | Puerto (In) | Method Call | Recibe la petición de la TPV para realizar el pago de una entrada a través de la pasarela de pago. |
| CashPayment | Puerto (In) | Method Call | Recibe la realización de una venta en metálico hecha a través de la TPV. |
| CreditCardSale | Puerto (Out) | Method Call | Envía el resultado y los datos de una venta hecha a través de la pasarela de pago. |
| CashSale | Puerto (Out) | Method Call | Envía los datos de una venta en metálico hecha a través de TPV. |
| SaveTicket | Puerto (Out) | Method Call | Envía la solicitud de almacenamiento del comprobante de una entrada recién vendida. |
| GetTicket | Puerto (In-Out) | Method Call | Envía la solicitud de obtener el comprobante de la entrada que se quiere validar desde la TPV. |

##### TicketManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| SaveTicket | Puerto (Out) | Method Call | Envía la solicitud para guardar el comprobante de una venta recién hecha. |
| GetSaleTicket | Puerto (In-Out) | Method Call | Recibe la petición para la creación y envío del comporbante para una entrada vendida online. |
| GetTicket | Puerto (In-Out) | Method Call | Envía la petición para recibir el comporbante de la venta de una entrada para validarla. |
| CheckTicket | Puerto (In-Out) | Method Call | Recibe la petición de la TPV para la validar el comprobante de una entrada. |

##### CreditCardManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| ServerPayment | Puerto (In) | Method Call | Recibe la petición de *WebServer* para realizar un pago a través de la pasarela de pago. |
| ObtainTicket | Puerto (In-Out) | Method Call | Recibe la petición para enviar el comporbante de la venta de una entrada. |
| CreditCard Payment | Puerto (In) | Method Call | Recibe la petición de la TPV para realizar el pago de una entrada a través de la pasarela de pago. |
| GetSaleTicket | Puerto (In-Out) | Method Call | Envía la petición para obtener el comprobante de una entrada vendida desde el servidor. |
| CreditCardSale | Puerto (Out) | Method Call | Envía el resultado y los datos de una venta hecha a través de la pasarela de pago. |
| Payment | Puerto (Out) | Method Call | Lleva a cabo el pago de la venta de una entrada con tarjeta de crédito a través de la pasarela de pago. |

##### CashManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| CashPayment | Puerto (In) | Method Call | Recibe la realización de una venta en metálico hecha a través de la TPV. |
| CashSale | Puerto (Out) | Method Call | Envía los datos de una venta en metálico hecha a través de TPV. |

##### UserManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| SaveUser | Requerida |  | Ver 8.1.2.3.2. |
| LoadUser | Requerida |  | Ver 8.1.2.3.2. |
| RegisterUser | Puerto (In) | Method Call | Recibe la petición y crea un nuevo usuario a través de los datos de registro. |
| LoginUser | Puerto (In-Out) | Method Call | Reciba la petición de obtener la información del usuario que inicia sesión mediante su identificador.  Cierra la sesión si el usuario ha estado inactivo tras un periodo de tiempo. |

##### BillboardManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| ConnectBillboard | Requerida | RMI | Ver 8.1.2.3.2. |
| LoadBillboard | Requerida | RMI | Ver 8.1.2.3.2. |
| ConnectServer | Puerto (In) | Method Call | Recibe la petición de *WebServer* para conectarse a las actualizaciones de la cartelera. |
| GetServer Billboard | Puerto (In-Out) | Method Call | Recibe la petición del servidor para obtener la cartelera.  Tras recibir la cartelera la transforma a un formato que el servidor puede interpretar. |
| GetTPVBillboard | Puerto (In-Out) | Method Call | Recibe la petición de la TPV para obtener la cartelera.  Tras recibir la cartelera la transforma a un formato que la TPV puede interpretar. |

##### SalesManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| SaveSale | Requerida |  | Ver 8.1.2.3.2. |
| CreditCardSale | Puerto (In) | Method Call | Recibe la petición para guardar una venta hecha a través de la pasarela de pago.  Transforma la venta en un formato compatible con la base de datos. |
| CashSale | Puerto (In) | Method Call | Recibe la petición para guardar una venta en metálico hecha a través de la TPV.  Transforma la venta en un formato compatible con la base de datos. |

#### Comportamiento

##### WebServerGateway

Este módulo se encarga de balancear el trabajo que recibe el subsistema por parte de los servidores. También sirve como el adaptador que encapsula el acceso a *Central System.*

Un usuario puede registrarse en el sistema (*Register*) o loguearse (*Login*) y el módulo se encarga de dirigir las llamadas a al *UserManager*  en función de la necesidad (*RegisterUser* o *LoginUser*).

Cuando un servidor ser conecta por primera vez a *Central System* necesita suscribirse a las actualizaciones de la cartelera, para ello primero pasa por este módulo (*Connect*) que le redirecciona a *BillboardManager* para cumplirlo. En caso de que ya esté conectado y necesite actualizar su cartelera (*ServerRequestBillboard*) el módulo trae lo datos (*GetServerBillboard*) y se lo pasa.

Cuando se lleva a cabo la compra de una entrada (*Payment*) el módulo conecta con *PaymentManager* (*ServerPayment*) y realiza el pago, tras lo cual pide el comprobante de la venta (*ObtainTicket*). Cuando el servidor necesita el comprobante lo pide (*ReceiveTicket*) y lo recibe.

##### TPVGateway

Este módulo balance la carga de trabajo de las TPV con *Central System*  y encapsula los accesos al mismo. Las peticiones de las TPVs (*TPVRequest*) se adaptan a unas peticiones entendidas por el subsistema.

Cuando una TPV se conecta recibe la cartelera a través de *GetTPVBillboard.*

Si se realiza la compra de una entrada por tarjeta de crédito el módulo adapta la petición para enviar *PaymentManager* a través de *CreditCardManager.* En caso de que sea una venta en metálico usa el puerto *CashManager* para enviar la petición.

Las cancelaciones de entradas desde una TPV se adaptan con el puerto *CheckTicket* permitiendo obtener la validación de la misma.

##### PaymentManager

*PaymentManager*  se encarga de controlar todas las operaciones relacionadas con la venta de entradas.

Cuando un servidor realiza la compra de una entrada online (*ServerPayment*) traslada la petición a *CreditCardManager.* Después de la venta, también se pedirá el comprobante de entrada (*ObtainTicket*) y también reenviará esta petición a *CreditCardManager.*

Desde una TPV pueden venir las ventas en metálico (*CashPayment*) las cuales son enviadas a *CashManager.* Las ventas realizadas con tarjeta de crédito (*CreditCardPayment*) y son enviadas a *CreditCardManager.* Cuando se quiere cancelar una entrada (*CheckTicket*) esta se válida mediante *TicketManager.*

Desde el módulo *TicketManager*  recibe las peticiones para guardar el comprobante de una entrada vendida recientemente (*SaveTicket*) y las peticiones que necesitan obtener los comprobantes de ventas ya realizadas para validar una cancelación.

Cuando una venta mediante pasarela de pago ha sido terminada *CreditCardManager* al módulo enviar la petición para guardar en base de datos la venta (*CreditCardSale*). Lo mismo ocurre con *CashManager* y las ventas en metálico realizadas (*CashSale*).

A través de *DoPayment* se conecta con el adaptador de la pasarela de pago para realizar los pagos mediante tarjeta de crédito.

##### TicketManager

Se encarga de crear y validar los comprobantes asociados a las entradas vendidas a través del servicio web.

Cuando desde una TPV llega la petición de cancelar una entrada (*CheckTicket*) el módulo obtiene el comprobante asociado a la entrada (*GetTicket*) para luego validarlo en caso de que sea correcto o denegarlo en caso de que no lo sea.

Cuando se hace una venta online se le pide al módulo crear un comprobante para dicha venta (*GetSaleTicket*). Una vez creado se contesta a la petición con la versión para imprimir y se guarda asociada a la venta en la base de datos (*SaveTicket*).

##### CreditCardManager

Es el módulo encargado de manejar las ventas con tarjeta de crédito y conectarlas con la pasarela de pago para llevar a cabo el pago propiamente dicho. También obtiene y envías los comprobantes para las ventas online y envía a guardar las ventas.

Cuando recibe la petición de llevar a cabo un pago con tarjeta de crédito, ya sea desde un servidor (*ServerPayment*) o una TPV (*CreditCardPayment*) maneja las peticiones y se conecta con la pasarela de pago (*Payment*) para llevar a cabo el pago.

En caso de que se una petición desde un servidor, obtiene el comprobante de la venta (GetSaleTicket) que se pedirá posterior desde el servidor (*ObtainTicket*).

Sea cual sea el tipo de petición que haya realizado el pago, esta será enviada en la petición para ser guardada en la base de datos (*CreditCardSale*) en un formato que incluirá la butaca en de la entrada.

##### CashManager

Maneja las ventas pagadas en metálico desde la TPV y las envía a guardar a la base de datos.

La venta de una entrada en metálico es convertida por el módulo a un formato que reconocerá la base de datos, incluyendo en ella la butaca asociada a la entrada.

##### UserManager

Este módulo se encarga de la creación de usuario y del control de las sesiones abiertas por los mismos.

Cuando un usuario nuevo se registra en el sistema (*RegisterUser*) crea un nuevo usuario a partir de los datos recibidos y los envía para ser guardado en el sistema (*SaveUser*).

Un usuario puede loguearse en el sistema para acceder a sus datos (*LoginUser*) y el módulo se encarga de obtener los datos (*LoadUser*) y si la contraseña es correcta retornarlos al usuario.

##### BillboardManager

Se encarga de obtener y distribuir la cartelera en los servidores y TPV, adaptándolos a los formatos que soporten. También se encarga de redirigir las suscripciones a la actualización de la cartelera de los servidores.

Cuando un servidor se quiere suscribir a las actualizaciones de la cartelera (*ConnectServer*) el módulo reenvía la petición al módulo de la base de datos que se encarga de llevar a cabo las actualizaciones (*ConnectBillboard*).

Cuando un servidor necesita actualizar su cartelera realiza la petición al módulo (*GetServerBillboard*) este la adapta la información de la cartelera a un formato interpretables por el servidor. Lo mismo ocurre cuando la petición viene por una TPV (*GetTPVBillboard*).

Cuando el módulo necesita obtener la cartelera de la base de datos realiza una petición para ello (*LoadBillboard*) y la almacena internamente para disminuir la carga de trabajo del servidor, llamando obteniendo solo la cartelera cuando se produce una actualización en la misma.

##### SalesManager

*SalesManager* adapta los resultados de las ventas por tarjeta de crédito (*CreditCardSale*) y en metálico (*CashSale*) a un formato interpretables por la base de datos de manera que la butaca continúe asociada a la venta y los envía para ser guardados en la base de datos (*SaveSale*).

### Diagrama contextual

Esta vista es el desarrollo del subsistema *Central System.* (Ver 8.1.1)

### Guía de puntos de variación

En *WebServerGateway* a pesar de que S*erverPayment* y *ObtainTicket* son 2 puertos distintos, ambos deben usarse en una misma transacción para evitar que si la primera termina y la segunda falla, no se quede una venta online sin comprobante.

Tanto *WebServerGateway* como *TPVGateway* están presentados como un módulo cada uno, pero podrían estar implementados separando las distintas funcionalidades que los componen. Por ejemplo, un módulo dedicado a balancear la carga de trabajo de los registros y logueos de usuarios tendrá menos trabajo que el módulo que solicita las actualizaciones de la cartelera cuando estás se produce, por lo que es conveniente que el balanceo se produzca por separado y que uno no depende del otro.

*TicketManager* se encuentra dentro de *PaymentManager* porque está relacionado directamente con las ventas y cancelaciones de entradas, pero esto es una decisión de diseño y podría estar presentado o implementado por separado.

*BillboardManager* podría almacenar en una pequeña base de datos una copia de cartelera para que cuando se lleven a cabo las actualizaciones no tengan que estar obteniendo todo el rato la misma cartelera.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 3 | **AT-03** | El balanceador de carga de *WebServerGateway* permite manejar y controlar grandes cargas de trabajo de manera que el subsistema sufra menos en horas puntas, dando prioridad a las ventas frente a otras acciones. |
| 4 | **AT-04** | Los módulos *CreditCardManager* y *CashManager* asocian a la venta la butaca de la entrada que se está comprando, de manera que solo la primera venta que llegué tendrá dicha butaca asociada, impidiendo que otra se quede con ella. |
| 5 | **AT-05** | El módulo *TPVManager* se encarga de balancear el trabajo de las ventas hechas por las TPVs de manera que las ventas se produzcan rápidamente, permitiendo además ventas más rápidas aun cuando son en metálico porque *CashManager* solo recibe las peticiones desde un módulo. |
| 6 | **AT-06** | Ya sea a través de *WebServerGateway* o *TPVManager* los pagos hechos con tarjeta d crédito a través de la pasarela de pago se realiza de manera transaccional, por lo que permite una mayor seguridad, impidiendo que si se un error cause algún daño en el comprador.  El subsistema usa un adaptador para poder conectar con la pasarela de pago ofrecida por el banco, reforzando la seguridad al no ser un elemento integrado forzosamente en el sistema. |
| 11 | **AT-11** | Al hacer la compra de una entrada a través de la web, se crea el comprobante asociado a la misma, de manera que cuando dicho comprobante es presentado en un TPV se pueda validar y entregar la entrada al comprador.  Cuando se produce la compra, esta se realiza de manera transaccional incluyendo el pago a través de la pasarela de pago y la creación y obtención del comprobante de la venta, de manera que no haya ventas sin comprobante. |

## Administration System View

La vista de *Administration System* muestra el primer nivel de descripción de los componentes de *Administration System.*

### Presentación principal



Figura 5. Administration System View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| AdminPresentation | Es la capa de presentación de la aplicación del administrador. |
| AdminLogicManager | Capa lógica del administrador. |
| CommitManager | Encargado de llevar a cabo las actualizaciones de la cartelera sin sobrecargar el sistema. |
| InfoReader | Obtiene la información del sistema que necesite el administrador |

#### Relaciones

*AdminPresentation* pide a *AdminLogicManager* la información que necesite en un momento determinado. InfoReader recibe la petición desde AdminLogicManager y la lleva a cabo, tras lo cual devuelve los datos recogidos.

*AdminPresentation* lleva a cabo unos cambios en la cartelera que son guardados y se lo comunica a *AdminLogicManager*. Este pasa lo cambios a *CommitManager* que lleva a cabo la petición de actualizar la cartelera tras un periodo de tiempo.

#### Interfaces/Puertos

##### AdminPresentation

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| EditBillboard | Puerto (Out) | Method Call | Solicita la salvaguarda de las modificaciones hechas en la cartelera.  También puede solicitar la cancelación de una actualización. |
| Billboard | Puerto (In-Out) | Method Call | Pide y recoge los datos de la última versión de la cartelera en un formato editable. |
| Sales | Puerto (In-Out) | Method Call | Pide y recoge los datos de todas las entradas vendidas en un ámbito determinado por el administrador. |
| Users | Puerto (In-Out) | Method Call | Pide y recoge los datos de todos los usuarios. |

##### AdminLogicManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| UpdateBillboard | Requerida | Method Call | Ver 8.1.2.3.3. |
| GetRawBillboard | Requerida | Method Call | Ver 8.1.2.3.3. |
| GetSales | Requerida | Method Call | Ver 8.1.2.3.3. |
| GetUsers | Requerida | Method Call | Ver 8.1.2.3.3. |
| EditBillboard | Puerto (In) | Method Call | Recibe los peticione para la realización de cambios en la cartelera y los pasa a *CommitManager* |
| Billboard | Puerto (In-Out) | Method Call | Recibe la petición de obtención de la cartelera y la traslada a *InfoReader.* Cuando contesta, devuelve los datos. |
| Sales | Puerto (In-Out) | Method Call | Recibe la petición de los datos de ventas y los parámetros con los que se buscan.  Pasa la petición a *InfoReader* y devuelve los datos recogidos cuando contesta. |
| Users | Puerto (In-Out) | Method Call | Recibe la petición de obtener todos los datos de los usuarios.  Pasa la petición a *InfoReader* y devuelve los datos recogidos cuando contesta. |

##### CommitManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| UpdateBillboard | Puerto (Out) | Method Call | Actualiza los datos del sistema con modificados de la cartelera que tenga guardados tras un periodo de tiempo |
| EditBillboard | Puerto (In) | Method Call | Recibe la solicitud de salvaguardar los datos de una actualización de cartelera.  Los guarda en memoria secundaria hasta que se haga una actualización.  Puede recibir la cancelación de la actualización de los datos que se encuentren en memoria secundaria. |

##### InfoReader

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| GetRawBillboard | Requerida | Method Call | Ver 8.1.2.3.3. |
| GetSales | Requerida | Method Call | Ver 8.1.2.3.3. |
| GetUsers | Requerida | Method Call | Ver 8.1.2.3.3. |
| Billboard | Puerto (In-Out) | Method Call | Recibe la petición de obtención los datos de la última versión de la cartelera en un formato editable.  Cuando los obtiene los devuelve. |
| Sales | Puerto (In-Out) | Method Call | Recibe la petición de obtener los datos de las ventas de entradas en un ámbito.  Cuando recibe los datos los devuelve a quien los pidió. |
| Users | Puerto (In-Out) | Method Call | Recibe la petición para obtener la información de todos los usuarios.  Cuando la recibe la devuelve. |

#### Comportamiento

##### AdminPresentation

Es el módulo encargado de proporcionar la interfaz gráfica con la que trabaja el administrador.

Cuando el administrador quiere ver las ventas de una determinada entrada de la cartelera deberá indicar los parámetros por los que se busca (*Sales*).

También puede visualizar los usuarios registrado en el sistema (*Users*).

Para realizar una modificación de la cartelera, primero es necesario obtenerla (*Billboard).* Desde ella se pueden realizar los cambios pertinentes y después se pueden guardar para que se actualicen en el sistema (*EditBillboard*).

##### AdminLogicManager

Se encarga de encapsular las peticiones del administrador para que las lleven a cabo los módulos que los componen.

Las interfaces *Billboard, Sales* y *Users*  reciben las peticiones de datos y las trasladan a *InfoReader* cuyas peticiones internas se llevan a través de las interfaces *GetRawBillboard, GetSales* y *GetUsers.*

Cuando llega una petición de editar la cartelera por la interfaz *EditBilboard* la pasa a *CommitManager.* Cuando *CommitManager* realiza una actualización, pasa a través de la interfaz *UpdateBillboard.*

##### CommitManager

Este módulo se encarga de gestionar las actualizaciones de la cartelera de manera que no se produzcan sobrecargas en el sistema por hacer demasiados cambios en poco tiempo.

Cuando recibe la cartelera modificada (*EditBillboard*) la almacene en una memoria secundaria. Además comprueba que las horas de las sesiones estén separadas por al menos 30 minutos entre ellas.

Cuando ha pasado un período determinado de tiempo o la cantidad de información alcanza cierto nivel, lleva a cabo la actualización de la cartelera en el sistema mediante una transacción para evitar que se produzcan problemas por el medio y deje a medio actualizar la cartelera.

Si la actualización no se ha producido es posible borrar todos los cambios producidos a través del puerto *EditBillboard*.

##### InfoReader

Es el encargado de llevar a cabo las peticiones para obtener la información y datos que el administrador necesite.

Los puertos *GetRawBillboard y Billboard* se encargan de pedir, recibir y enviar los datos relacionados con la cartelera del sistema central al administrador.

Los puertos *GetSales* y *Sales* hacen lo mismo con las ventas de entradas, pero a través de unos parámetros indicados por el administrador para recibir las ventas de un ámbito de la cartelera.

Los puertos *GetUsers* y *Users* funcionan de igual manera pero con la información de los usuarios.

### Diagrama contextual

Esta vista es el desarrollo del subsistema *Administration System.* (Ver 8.1.1)

### Guía de puntos de variación

El módulo AdminPresentation de puede implementarse de muchas maneras, por lo que no existen restricciones al respeto, siempre y cuando cumpla lo requerimientos para conectarse y mostrar la información.

CommitManager puede tener una pequeña base de datos donde almacenar los cambios antes de trasladarlos, o puede almacenarlos en memoria.

InfoReader está presentado como un solo módulo, pero puede estar implementado con una división que separe los 3 tipos de peticiones de manera que los cambios en uno no afecten a los otros.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 2 | **AT-02** | Las actualizaciones que se lleven a cabo se hacen de manera transaccional, de manera que un usuario solo tendrá la última versión válida de la cartelera. Además los cambios que se haga en la cartelera se hacen en grupos gracias al *CommitManager*, por lo que el sistema no tiene que actualizar los servidores cada vez que se hagan pequeños cambios. |
| 10 | **AT-10** | La interfaz se presenta de manera que permite realizar cambios en la cartelera en la cartelera. Cuando uno de esos cambios se produce se guarda en *CommitManager,* pero no realiza la actualización inmediatamente, por lo que el administrador puede deshacer los cambios antes de que afecten a todo el sistema. |

## Database System View

La vista de *Database System* muestra el primer nivel de descripción de los componentes de *Database System.*

### Presentación principal



Figura 6. Database System View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| UserDBManager | Módulo gestor de encriptación y almacenamiento de usuarios. |
| SalesDBManager | Módulo gestor de encriptación y almacenamiento de venta de entradas. |
| BillboardDBManager | Módulo gestor de almacenamiento de la cartelera. |
| GeneralDB | Base de datos general que contiene todos los datos del sistema. |

#### Relaciones

*UserDBManger*  se conecta con *GeneralDB* para almacenar los usuarios tras encriptar sus datos y para obtenerlos y desencriptarlos para su uso en otras partes del sistema.

*SalesDBManager* está relacionada de forma similar con *GeneralDB* pero añadiendo la gestión de los comprobantes de venta.

*BillboardDBManager* está relacionada con *GeneralDB* para la manipulación y obtención de la cartelera.

#### Interfaces/Puertos

##### UserDBManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| SaveUser | Proveída | RMI | Ver 8.1.2.3.4. |
| LoadUser | Proveída | RMI | Ver 8.1.2.3.4. |
| GetUsers | Proveída | Method Call | Ver 8.1.2.3.4. |
| Save EncryptedUser | Puerto (Out) | DAO (ODBC) | Guarda los datos encriptados de un nuevo usuario. |
| Get EncryptedUser | Puerto (In-Out) | DAO (ODBC) | Carga los datos del usuario buscado por identificador de inicio de sesión para desencriptarlo y enviarlo. |

##### SalesDBManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| SaveSale | Proveída | RMI | Ver 8.1.2.3.4. |
| SaveTicket | Proveída | RMI | Ver 8.1.2.3.4. |
| GetTicket | Proveída | RMI | Ver 8.1.2.3.4. |
| GetSales | Proveída | Method Call | Ver 8.1.2.3.4. |
| Save EncryptedSale | Puerto (Out) | DAO (ODBC) | Guarda los datos encriptados de un nueva entrada vendida e indicando la butaca elegida y su tipo (tarjeta o metálico). |
| Get EncryptedSales | Puerto (In-Out) | DAO (ODBC) | Carga todas las ventas de entradas realizadas para desencriptarlas y enviarlas. |
| Save EncryptedTicket | Puerto (Out) | DAO (ODBC) | Guarda encriptado el comprobante de venta online asociado a una determinada entrada. |
| Get EncryptedTicket | Puerto (In-Out) | DAO (ODBC) | Obtiene el comprobante asociado a un determinada venta lo desencripta y lo envía |

##### BillboardDBManager

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| ConnectBillboard | Proveída | RMI | Ver 8.1.2.3.4. |
| LoadBillboard | Proveída | RMI | Ver 8.1.2.3.4. |
| UpdateBillboard | Proveída | Method Call | Ver 8.1.2.3.4. |
| GetRawBillboard | Proveída | Method Call | Ver 8.1.2.3.4. |
| SaveBillboard | Puerto (Out) | DAO (ODBC) | Guarda los datos nuevos y modificados de la cartelera en la base de datos. |
| GetBillboard | Puerto (In-Out) | DAO (ODBC) | Carga la cartelera para enviarla. |

##### GeneralDB

|  |  |  |  |
| --- | --- | --- | --- |
| Interface | Tipo | Tecnología | Propiedades |
| Save EncryptedUser | Acceso DB | DAO (ODBC) | Escribe en la base de datos un nuevo usuario. |
| Get EncryptedUser | Acceso DB | DAO (ODBC) | Obtiene los datos encriptados del usuario buscado. |
| Save EncryptedSale | Acceso DB | DAO (ODBC) | Escribe en la base de datos una nueva venta de entrada. |
| Get EncryptedSales | Acceso DB | DAO (ODBC) | Obtiene los datos encriptados de todas las ventas. |
| Save EncryptedTicket | Acceso DB | DAO (ODBC) | Escribe en la base de datos el comprobante de la venta de una entrada. |
| Get EncryptedTicket | Acceso DB | DAO (ODBC) | Obtiene los datos encriptados del comprobante asociado a la entrada vendida. |
| SaveBillboard | Acceso DB | DAO (ODBC) | Escribe en la base de datos la cartelera con las últimas modificaciones. |
| GetBillboard | Acceso DB | DAO (ODBC) | Obtiene todos los datos de la cartelera. |

#### Comportamiento

##### UserDBManager

Este módulo se encarga de la encriptación y desencriptación de la información sobre los usuarios de la base de datos.

Cuando recibe los datos de un nuevo usuario registrado en el sistema (*SaveUser*) encripta los datos para luego almacenarlos en la base de datos a través de un *DAO* (*SaveEncryptedUser*).

Cuando necesita acceder a los datos de un usuario que acaba de iniciar sesión (*LoadUser*) primero los carga de la base de datos (*GetEncryptedUser*), luego los desencripta y lo envía a quien los pidió.

De manera similar ocurre cuando el administrador del sistema pide los datos de los usuarios (*GetUsers*), pero cargando todos los usuarios y desencriptandolos antes de enviarlos.

##### SalesDBManager

Este módulo se encarga de encriptar las ventas de entradas y los comprobantes asociadas a estas.

Cuando se realiza un venta en el sistema es enviada a este módulo para almacenarla en la base de datos (*SaveSale*), donde se encripta y se manda a guardar (*SaveEncryptedSale*). Si la venta ha sido a través del servicio Web, paralelamente se creó un comprobante asociado a la entrada, que también es enviado a este módulo (*SaveTicket*) para encriptarlo y guardarlo en la base de datos (*SaveEncryptedTicket*).

Si se realiza la cancelación desde una TPV se requiere conocer entrada a la que está asociada (*GetTicket*) por lo que se busca (*GetEncryptedTicket*) en la base de datos y se desencripta antes de enviarla.

Cuando un administrador necesita comprobar las entradas vendidas (*GetSales*) el módulo obtiene todas las ventas según los parámetros del administrador (*GetEncryptedSales*).

##### BillboardDBManager

Este módulo se encarga de gestionar los datos y actualizaciones de la cartelera en el sistema.

Cuando un servidor se suscribe (*ConnectBillboard*) permite al módulo indicarle cuando se producen actualizaciones.

Cuando un servidor necesita obtener la nueva cartelera (*LoadBilboard*) se conecta a la base de datos (*GetBillboard*) para obtener la última versión de la cartelera guardada.

De forma similar ocurre cuando un administrador accede a los datos de la cartelera (*GetRawBillboard*) pero en este caso lo envía en un formato editable.

Cuando el administrador añade o modifica algo de la cartelera (*UpdateBillboard*) se realiza la actualización de la base de datos (*SaveBillboard*) y se desencadena la actualización de los servidores.

##### GeneralDB

Es módulo es la base de datos general del sistema. En ella se almacena toda la información usada por el sistema.

Guarda (*SaveEncryptedUser)* y recupera (*GetEncryptedUser*) la información encriptada de los usuarios registrados y.

Guarda (SaveEncryptedSale) la información de las entradas vendidas y recupera todas las ventas según los parámetros buscados (*GetEncryptedSales*).

Almacena los comprobantes de una entrada vendida a través de la Web (*SaveEncryptedTicket*) y recupera el comprobante para cancelar la entrada (*GetEncryptedTicket*).

Guarda las modificaciones y añadidos a la cartelera hechas por un administrador (*SaveBillboard*) y recupera toda la cartelera de la base de datos (*GetBillboard*).

### Diagrama contextual

Esta vista es el desarrollo del subsistema *Database System.* (Ver 8.1.1)

### Guía de puntos de variación

*GeneralDB* puede estar separada en varias bases de datos, pero deberían estar interrelacionadas, puesto que las entradas vendidas a un usuario registrado deberían estar asociadas a él.

*BillboardDBManager* implementa la otra parte del patrón *Observer*  donde se suscriben los servidores para recibir actualizaciones (Ver 8.2.4) metiendo los *Listener* de los servidores en una lista que se recorrería para notificarles que deberían hacer la petición de actualizar la cartelera.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 2 | **AT-02** | El módulo *BillboardDBManager* está pensando para que cuando se haga un cambio en la cartelera este cambio se transmita a los servidores y estos obtengan la última versión de la cartelera. |
| 7 | **AT-07** | Las ventas y los usuarios son encriptados para proteger los datos según las leyes de protección de datos de manera que solo los elementos internos del sistema sepan cómo se encripten/desencripten. |
| 10 | **AT-10** | Cuando se realiza una actualización de la cartelera de parte del administrador, esta se cambia en la base de datos rápidamente. |

## PaymentGateway Adapter View

La vista del Adaptador de la pasarela de pago muestra la forma en la que el subsistema se comunica con ella para realizar los pagos con tarjeta de crédito.

### Presentación principal



Figura 7. PaymentGateway Adapter View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| IPaymentGateway | Encapsula los métodos de pago de las pasarelas de pago de manera que solo se comunique mediante un método de pago. |
| BankGatewayPayment | Clase que contiene los métodos específicos de pasarela de pago y que lleva a cabo el pago propiamente dicho a través del método especificado por su interfaz. |
| DoPayment | Esta clase encapsula el pago de la entrada vendida, indicándole sobre qué clase debe realizarse el pago mediante un *set.* |
| CreditCardSale | Contiene toda la información relacionada con la venta, como puede el número de tarjeta de crédito, cantidad a pagar… |

#### Relaciones

N/A

#### Interfaces/Puertos

N/A

#### Comportamiento

Ver *Propiedades* en 8.6.2.1.

### Diagrama contextual



Figura 8. Contexto de PaymentGateway Adapter

### Guía de puntos de variación

La pasarela de pago es ofrecida por el banco, por lo que es conveniente mantener separada la pasarela de todo el sistema, de manera que si esta cambia pueda modificarse adaptándose a la interfaz IPaymentGateway para esa pasarela, de manera que solo haya que cambiarla, mediante el método *setPaymentGateway,* en la clase *DoPayment* del subsistema.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 6 | **AT-06** | El adaptador permite llevar a cabo una transacción segura con el banco y también permite modificar la pasarela de pago en caso de que esta sea necesaria cambiarla sin complicar la implementación. |

## TPV Adapter View

La vista del Adaptador de la TPV muestra la forma en la que esta pasa los mensajes de la realización de las operaciones al subsistema.

### Presentación principal



Figura 9. TPV Adapter View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| ITPVEvent | Encapsula la operación de establecer el subsistema en la TPV para usar sus métodos en las circunstancias que se requieran. |
| TPVEvent | Implementa las funcionalidades concretas de una TPV para usando los métodos de *TPVActions* para realizar las operaciones. |
| ITPVActions | Encapsula las operaciones de la TPV reconocidas por el sistema. |
| TPVActions | Esta clase es la implementación concreta de *ITPVActions* para el subsistema. |
| CashSale | Contiene la información de una entrada pagada en metálico. |
| CreditCardSale | Contiene toda la información relacionada con la venta con tarjeta de crédito, como puede el número de tarjeta de crédito, cantidad a pagar… |
| Ticket | Contiene los datos de un comprobante que se requiere validar en el sistema para canelar la entrada. |

#### Relaciones

N/A

#### Interfaces/Puertos

N/A

#### Comportamiento

Ver *Propiedades* en 8.7.2.1.

### Diagrama contextual



Figura 10. Contexto TPV Adapter

### Guía de puntos de variación

Las operaciones de la TPV pueden variar de un modelo a otro, por lo que es posible que un mismo cine haya varias implementaciones de *TPVActions* que se conecten con el sistema.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 5 | **AT-05** | La adaptación de las operaciones de la TPV al sistema permite una mayor velocidad ya que se focalizan en las funcionalidades del sistema que necesitan cumplir. |
| 6 | **AT-06** | La TPV permite realizar pagos con tarjeta de crédito seguros a través de la pasarela de pago con la que interacciona el subsistema *Central System*. |
| 11 | **AT-11** | Una de las funcionalidades del sistema que se adaptan de la TPV es la validación de los comprobantes de ventas online, de manera que el adaptador también permite validarlas a través de *Central System.* |

## Package Model View

La vista de paquete muestra información importante para el equipo de desarrollo del sistema: el empaquetamiento de los componentes.

### Presentación principal



Figura 11. Package Model View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Disposición de los elementos |
| WebServer | Package principal del subsistema encargado de atender las peticiones de los clientes.   1. PresentationManager – Component 2. ServerLogicManager – Component 3. ServerBillboardManager – Component 4. ServerBillboardDB – Component |
| Central System | Package principal del subsistema central de ventas.   1. WebServerGateway – Component 2. TPVGateway – Component 3. PaymentManager –Package    1. TicketManager – Component    2. CashManager – Component    3. CreditCardManager 4. UserManager – Component 5. BillboardManager – Component 6. SalesManager – Component 7. DoPayment – Class 8. TPVActions –Class |
| TPV Adapter | Disposición de los elementos del adaptador de la TPV.   1. TPVEvent – Class 2. ITPVActions –Interface 3. ITPVEvent –Interface 4. CashSale – Class 5. CreditCardSale –Class 6. Ticket – Interface |
| PaymentGateway Adapter | Disposición de los elementos del adaptador de la pasarela de pago.   1. BankPaymentGateway –Class 2. IPaymentGateway 3. CreditCardSale –Class |
| Administration System | Package principal del subsistema de administración.   1. AdminPresentation – Component 2. AdminLogicManager – Package    1. CommitManager – Component    2. InfoReader –Component |
| Database System | Package principal del subsistema de gestión de bases de datos.   1. SalesDBManager – Component 2. UsersDBManager – Component 3. BillboardDBManager – Component 4. GeneralDB – Component |
| Elemento | Disposición de los elementos |

#### Relaciones

Las relaciones de jerarquía están documentadas en el diagrama y en 8.8.2.1.

#### Interfaces/Puertos

N/A

#### Comportamiento

N/A

### Diagrama contextual

N/A

### Guía de puntos de variación

N/A

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 12 | **AT-12** | La partición en paquetes permite una cantidad de paralelismo importante en el desarrollo, lo que permitirá un desarrollo en un periodo menor de 2 meses. |
| 13 | **AT-13** | Mediante un solo equipo de estudiantes ávidos de ganar experiencia en el terreno de la arquitectura, el coste de desarrollo del prototipo del sistema será de 0 €. |

## Deployment View

La vista de Deployment ofrece una vista de los módulos en ejecución y sobre los equipos previstos.

### Presentación principal



Figura 12. Deployment View

### Catálogo de elementos

#### Elementos

|  |  |
| --- | --- |
| Elemento | Propiedades |
| HTTPServer | Ordenador donde se instalará el servidor *Tomcat* y el subsistema *WebServer* para atender las peticiones de los usuarios. |
| WebServer | Subsistema *WebServer* ver 8.2. |
| CentralSystemServer | Sistema de virtualización de máquinas con Virtual Box |
| CentralFrontEnd | Frontal con balanceador de carga los accesos desde el subsistema *WebServer* y del adaptador de la TPV.  Se ejecutará sobre máquinas virtuales *JVM*. |
| CentralBackEnd | Back-end de *Central System* donde se realizan todas las operaciones de negocio del sistema. |
| Database | Maquina física donde se instala la base de datos y el sistema de acceso a ella. |
| DatabaseFrontEnd | Frontal donde se reciben todos los accesos a la base de datos que permiten proteger los datos y realizar algunas acciones adicionales.  Se ejecuta sobre una *JVM* de la máquina física. |
| DatabaseBackEnd | Base de datos donde se guardará toda la información del sistema.  La base de datos será *MySQL* para reducir costes y se ejecutara sobre la máquina física. |
| Administrator | Ordenador o terminal donde se instalará la aplicación de administración del sistema.  No tiene ningún requisito a excepción de que pueda conectarse a la base de datos y pueda ejecutar la máquina virtual de java. |
| AdministrationSystem | Subsistema Administration System ver 8.4. |

#### Relaciones

*HTTPServer* transmite las peticiones de los usuarios a *CentralSystemServer* a través de RMI.

*CentralFrontEnd* se comunica con *CentralBackEnd* mediante colas de mansajes para balancear la carga de trabajo.

*CentralSystemServer* realiza las peticiones de *CentralBackEnd* con *RMI* para enviárselas a *Database.*

Dentro de *Database*, su *FrontEnd* se comunica con la base de datos propiamente dicha mediante *ODBC* y *DAOs.*

La máquina *Administrator* se comunica con *Database* para modificar los datos mediante llamadas a métodos.

#### Interfaces/Puertos

N/A

#### Comportamiento

N/A

### Diagrama contextual

N/A

### Guía de puntos de variación

Las partes del sistema pensadas para funcionar en máquinas virtuales Virtual Box, el sistema operativo donde se instalen no influye en el despliegue, siempre y cuando se sigan las directrices de la vista.

Aunque *Administrator* y *Database* están en 2 máquinas reales distintas, podrían ejecutarse en la misma a través desde 2 *JVM* distintas.

### Justificación de las decisiones

Las decisiones que han llevado a este diseño son:

|  |  |  |
| --- | --- | --- |
| Escenario | Atributo de calidad | Justificación |
| 1 | **AT-01** | El subsistema *WebServer* se puede desplegar en varios servidores de manera que si uno de ellos cae puede haber otros para ocupar su lugar, o se pueden añadir rápidamente. |
| 3 | **AT-03** | Los subsistemas están pensados para funcionar independientes, por lo que es posible añadir nuevos elementos replicados que permiten soportar mayor carga de trabajo sin tener que hacer adaptaciones sobre la marcha. |

# Bibliografía

**ANSI/IEEE 1471. 2000.** *Recommended Practice for Architectural Description of Software-Intensive Systems.* s.l. : ANSI/IEEE, 2000. ANSI/IEEE Std 1471-2000.

**Clements, Paul, y otros. 2000.** *A practical method for documenting software architectures*. [En línea] Carnegie Mellon University, 2000. http://www-2.cs.cmu.edu/afs/cs/project/able/ftp/icse03-dsa/submitted.pdf.

**Clements, Paul, y otros. 2010.** *Documenting Software Architecture.* Boston : Pearson Education, Inc., 2010. ISBN-13: 978-0-321-55268-6.

**Juan Fuente, Aquilino Adolfo. 2013.** *Arquitectura de Software*. Universidad de Oviedo, 2013.

# Índice de figuras

[Figura 1. Presentación del modelo de negocio 3](#_Toc353005996)

[Figura 2. System View 8](#_Toc353005997)

[Figura 3. WebServer View 13](#_Toc353005998)

[Figura 4. Central System View 17](#_Toc353005999)

[Figura 5. Administration System View 25](#_Toc353006000)

[Figura 6. Database System View 29](#_Toc353006001)

[Figura 7. PaymentGateway Adapter View 34](#_Toc353006002)

[Figura 8. Contexto de PaymentGateway Adapter 35](#_Toc353006003)

[Figura 9. TPV Adapter View 36](#_Toc353006004)

[Figura 10. Contexto TPV Adapter 37](#_Toc353006005)

[Figura 11. Package Model View 39](#_Toc353006006)

[Figura 12. Deployment View 41](#_Toc353006007)

# Glosario

|  |  |
| --- | --- |
| ADD | Es un método de modelado de arquitecturas basado en la identificación de los atributos de calidad y la creación de escenarios que esos atributos deben cumplir. |
| Artefacto | Es un elemento, en AAD es el elemento que está estimulado o afectado por el estímulo. |
| Atribute-Driven Design | Ver *ADD.* |
| Atributo de Calidad | Un atributo de calidad es un concepto complejo relacionado con los requisitos no funcionales que debe tener un modelo de arquitectura. |
| CRUD | Acrónimo de Crear, Obtener, Actualizar y Borrar (del inglés: **C**reate, **R**ead, **U**pdate and **D**elete). Es usado para referirse a las funciones básicas en bases de datos . |
| DAO | Acrónimo de Objeto de Acceso a Datos (del inglés **D**ata **A**ccess **O**bject), es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos. |
| Escenario de Calidad | Es una representación de un requisito no funcional establecido para un determinado atributo de calidad. |
| Interesado | Persona, institución o grupo de personas con intereses específicos en una arquitectura. |
| JVM | Máquina Virtual del Java (del inglés **J**ava **V**irtual **M**achine). |
| Method Call | Solicitud de uso de una operación de un elemento. |
| MySQL | Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario distribuido bajo licencia pública. |
| ODBC | **O**pen **D**ataBase **C**onnectivity es un estándar de acceso a las bases de datos. |
| RMI | **R**emote **M**ethod **I**nvocationes un mecanismo ofrecido por Java para invocar un método de manera remota. |
| SEI | **S**oftware **E**ngineering **I**nstitute (SEI) es un instituto federal estadounidense de investigación y desarrollo. |
| Stakeholder | Ver *Interesado.* |
| TPV | Acrónimo de **T**erminalde **P**unto de **V**enta. |

# Índice de palabras

ADD, 1

Artefacto, 7

Atribute-Driven Design, 1

Atributo de Calidad, 2, 7, 13, 17, 26, 30, 36, 38, 40, 44, 46

CRUD, 1

DAO, 15, 16, 33, 34, 35, 46

Escenario de Calidad, 6

Interesado, 1

JVM, 46

Method Call, 11, 12, 15, 20, 21, 22, 27, 28, 29, 33, 34

MySQL, 46

ODBC, 15, 16, 33, 34, 35, 46

RMI, 10, 11, 12, 15, 19, 22, 33, 34, 46

SEI, 1

Stakeholder, 1, 2

TPV, 2, 3, 4, 5, 7, 8, 11, 12, 13, 18, 20, 21, 22, 23, 24, 25, 26, 35, 39, 40, 43, 45