Breve manual de diseño de la práctica del dispensador de entradas de teatro

Tabla de contenidos

Versión 1.1.49

19:37 - 19/12/22

1 Introducción

El siguiente documento pretende guiar al estudiante por un diseño adecuado para implementar la práctica de la asignatura.

En un principio parece que las clases `TheaterTicketDispenser` y `UrjcBankServer` hacen todo lo que se pide en la práctica, pero la realidad es que toda la lógica necesaria para la operativa del dispensador está ausente de dichas clases. El diagrama de actividad de la Figura 1 muestra a grandes rasgos la lógica asociada al dispensador de entradas.

Obsérvese que este diagrama no contempla la lógica de bajo nivel asociada a cada operación (seleccionar día, zona o asientos...). Este diagrama tampoco refleja la estructura del teatro, la obra que se presenta…

2 Clases propuestas

La idea tras el diseño que se presenta está en que cada conjunto de operaciones coherente esté en una clase.

La Figura 2 presenta las clases principales que gestionan la operativa. A continuación se resume la responsabilidad de cada clase de este diagrama.

- `TheaterAPP`.- Encargada de iniciar el programa.
- `TheaterManager`.- Clase principal permite gestionar un teatro.
- `DispenseManager`.- Encargada de ocultar la complejidad de la clase `TheaterTicketDispenser` al resto de las clases.
- `DispenseHardware`.- Encargada de ocultar la complejidad de las operaciones con el manejador de la tarjeta de crédito y la impresora del dispensador.
- `Screen`.- Clase base de la que derivan cada una de las diferentes pantallas por las que transcurren las operaciones que realiza el expendedor.
- `TranslatorManager`.- Clase encargada de cargar los diccionarios y traducir las cadenas que se le pasen utilizando alguno de los traductores que contiene.
- `Translator`.- Clase que traduce a un idioma concreto, según el fichero de idioma que haya cargado.
- `Theater`.− Es una clase que da acceso a la información estática del teatro, básicamente a las áreas que contienen las butacas.

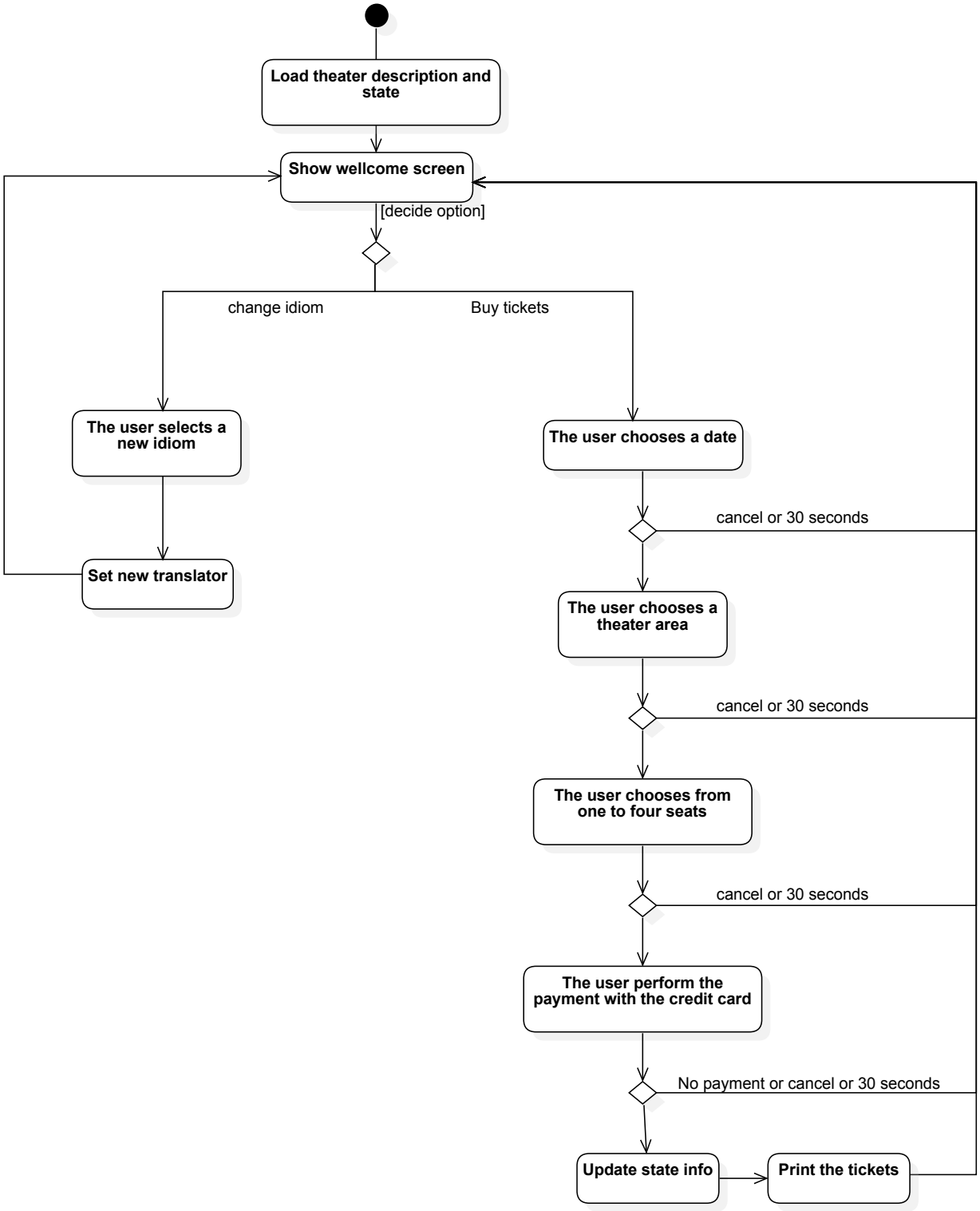

Figura 1.- Diagrama de actividad con la lógica de alto nivel asociada a la práctica.

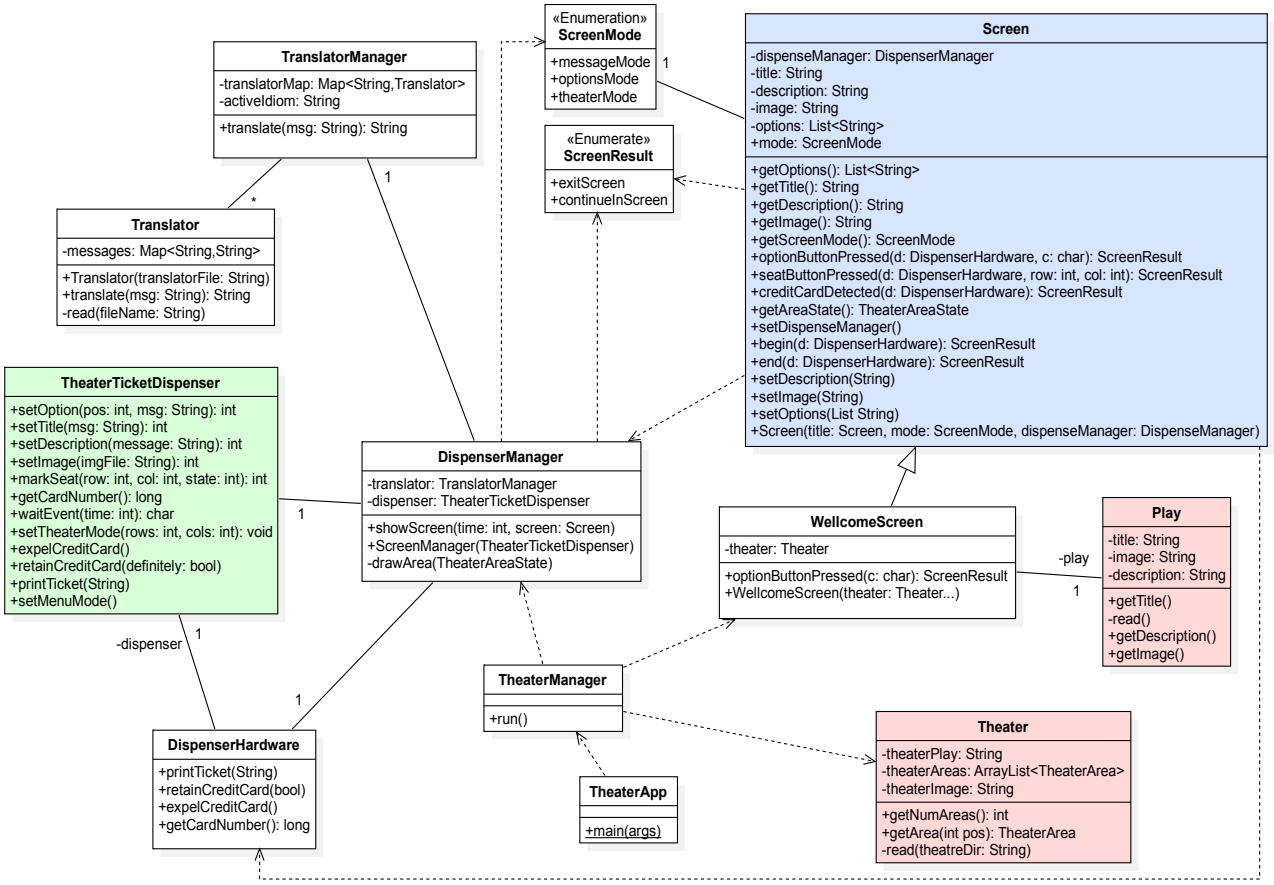

Figura 2.- Diagrama de clases principal.

La Figura 3 presenta las clases que derivan de `Screen` y que se encargan de gestionar la comunicación con el usuario. A continuación se resume la responsabilidad de cada clase de este diagrama.

- `WellcomeScreen`.- Esta pantalla mostrará el título, la descripción y el cartel de la obra. Además, presentará dos opciones: para comprar entradas y cambiar de idioma. Así, esta pantalla puede dar acceso a `DateSelectionScreen` o a `IdiomSelectionScreen`.
- `DateSelectionScreen`.- Permite la selección de la fecha en la que se venderán las siguientes entradas. Esta clase mostrará solo la fecha actual y las 4 siguientes fechas. Esta pantalla dará acceso a `AreaSelectionScreen` y tendrá un botón de cancelar.
- `AreaSelectionScreen`.- Encargada de permitir la selección de la zona del teatro en la que se venderán las siguientes entradas. Esta pantalla mostrará los nombres de las zonas y un dibujo de la zona respecto al teatro. Esta pantalla dará acceso a `SeatSelectionScreen`.
- `SeatSelectionScreen`.- Encargada de permitir la selección del asiento dentro del teatro. Esta pantalla dará acceso a `PaymentScreen` y tendrá un botón de cancelar.

- `PaymentScreen`.- Encargada de gestionar el cobro de la tarjeta de crédito. Presentará un resumen de la compra a pagar. La inserción de la tarjeta de crédito implicará la aceptación de la compra. Además, tendrá un botón de cancelar. Si al realizar el pago se produce un error de comunicación llevará a la pantalla `ErrorScreen`, en otro caso imprimirá los tickets y llevará a `WellcomeScreen`.
    - `IdiomSelectionScreen`.- Encargada de permitir la selección del idioma que se desea usar. Presentará un botón por cada idioma.
    - `ErrorScreen`.- Presenta el mensaje de error “En estos momentos no podemos atenderle” y comprobar cada 10 segundo que se haya restablecido la comunicación con el banco. Cuando se restablezca conducirá a la pantalla `WellcomeScreen`.

La Figura 4 presenta las clases que gestionan la información estática del teatro y de las reservas.

- `TheaterArea`.– Es una clase que gestiona los asientos dentro de un área del teatro.
- `Seat`.- Representa un asiento, su posición y su estado.
- `TheaterState`.– Es una clase que gestiona el estado del teatro en cada fecha.
- `TheaterAreaState`.– Es una clase que gestiona el estado de un área del teatro en una fecha.

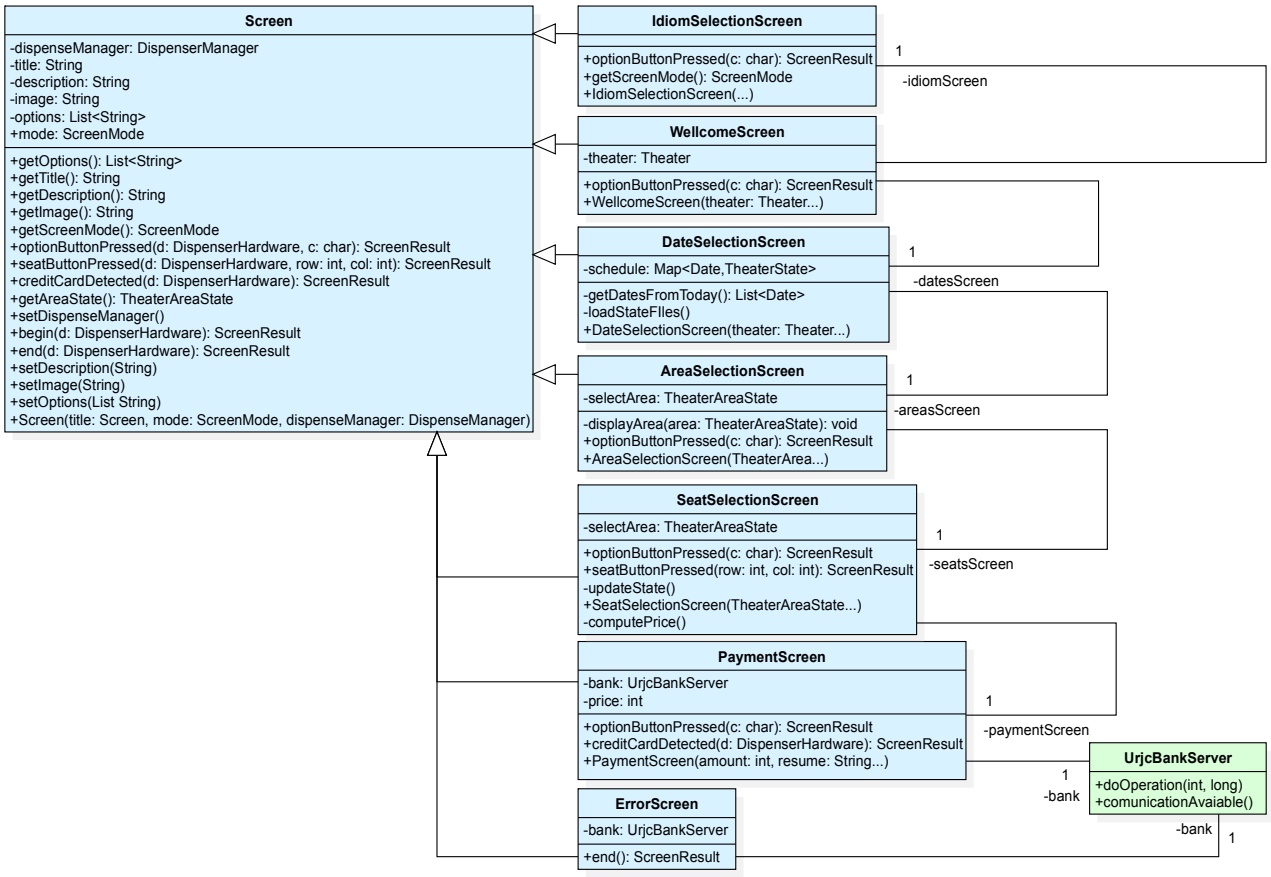

Figura 3.- Diagrama de clases que muestra las diferentes pantallas que presentará la aplicación.

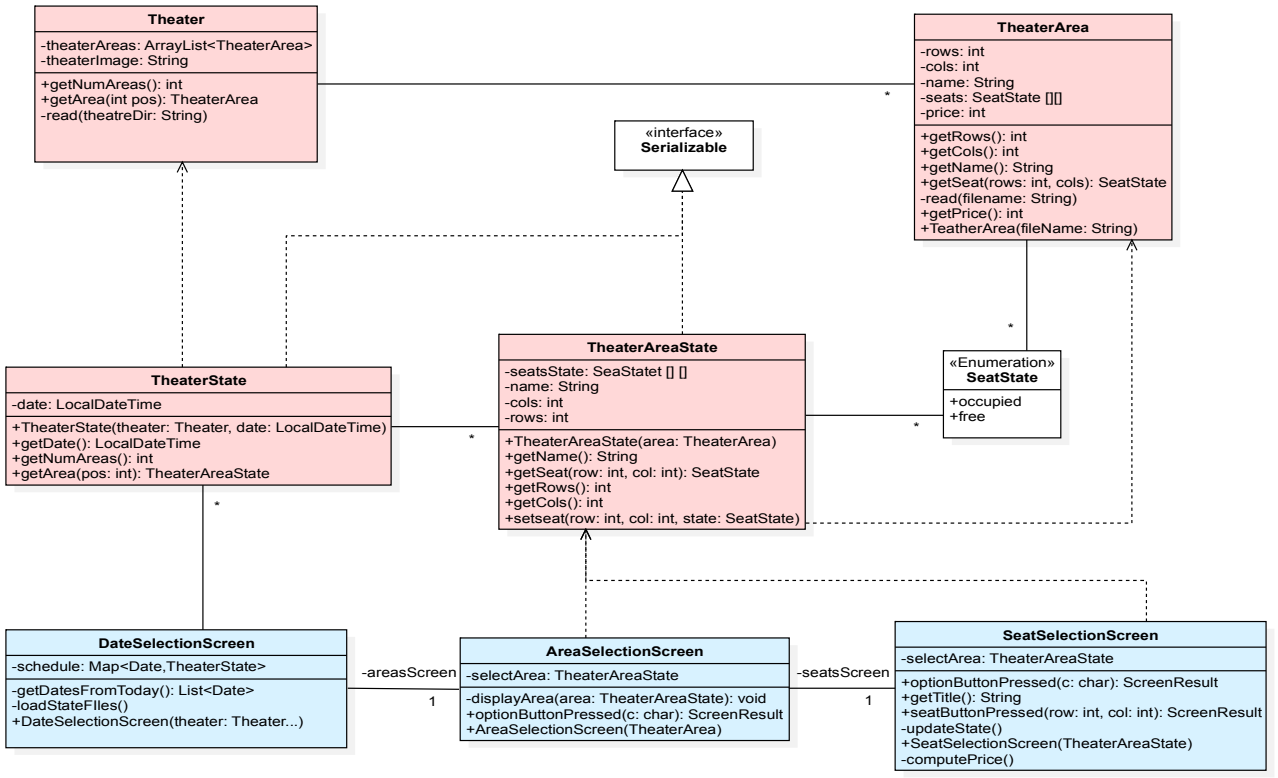

Figura 4.- Diagrama de clases que refleja el almacenamiento de la información sobre la forma del teatro y la agenda de butacas vendidas y por vender.

3 Operativa del dispensador

Partiendo de estas clases, el diagrama de Actividad de la Figura 1 quedaría segmentado en las clases y métodos que se ven el diagrama de la Figura 5.

3.1 Operativa de las clases Screen

Las clases `Screen` permiten gestionar la interacción con el usuario. Cada clase se debería corresponder con una pantalla.

La idea es que programar una clase `Screen` sea muy sencillo. Básicamente se debería:

- Heredar de `Screen`
- Rellenar las propiedades título, opciones, descripción e imagen.
- Programar lo que se desee que ocurra cuando se pulsen los botones que aparezcan correspondientes a las opciones que se habrá rellenado.

La clase base `Screen` es la que se encargaría de responder por defecto a los métodos `getter` usando la información almacenada en las propiedades que tiene. Además, respondería por defecto a las pulsaciones sobre botones no sobrescritos en clases derivadas. Obsérvese el uso del polimorfismo.

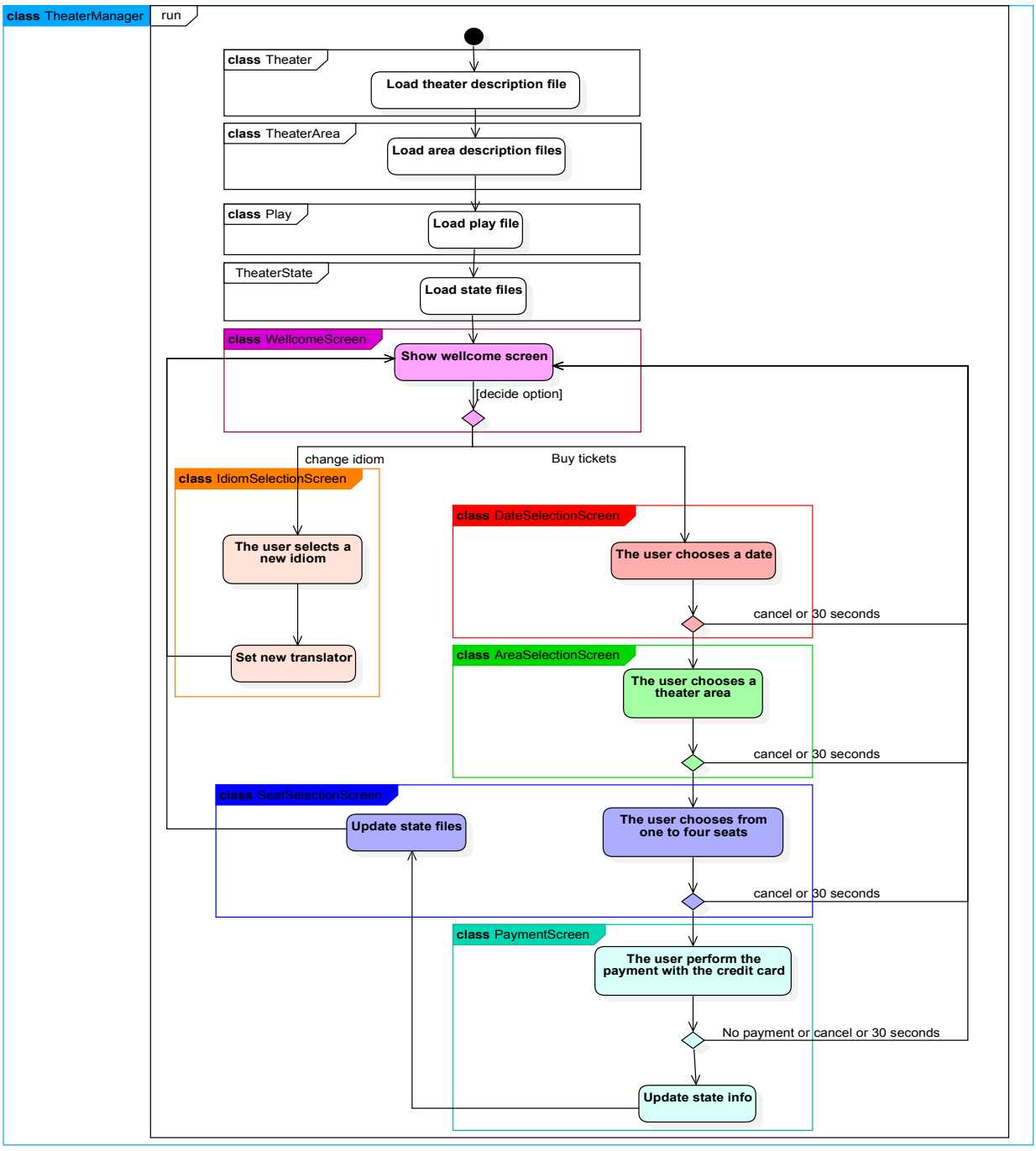

Figura 5.- Diagrama de actividad con la lógica de alto nivel asociada a la práctica indicando la clase en la que se ejecuta cada parte.

Además, estas clases están muy relacionadas con la clase `DispenserManager` que es la clase que las muestras y las controla. `DispenserManager` seguiría el esquema presentado en el diagrama de actividad de la Figura 6.

Básicamente, cuando se le pide a `DispenserManager` que muestre una `Screen` sigue el siguiente esquema:

1. Ejecuta el bloque `begin` del `Screen` que se le pide mostrar.

2. Se queda bloqueado en un `wait` el tiempo que le indiquen.
3. Ejecuta el método de la clase `Screen` correspondiente al evento que reciba del `wait`.
4. Ejecuta el método `end` del `Screen` que se le pide mostrar.

Si en cualquier momento alguno de los métodos devuelve `end` se termina el método de mostrado del `Screen`. Mientras que los métodos que se llaman devuelvan continúe, `DispenserManager` continuará ejecutando los pasos 2 a 4 en bucle.

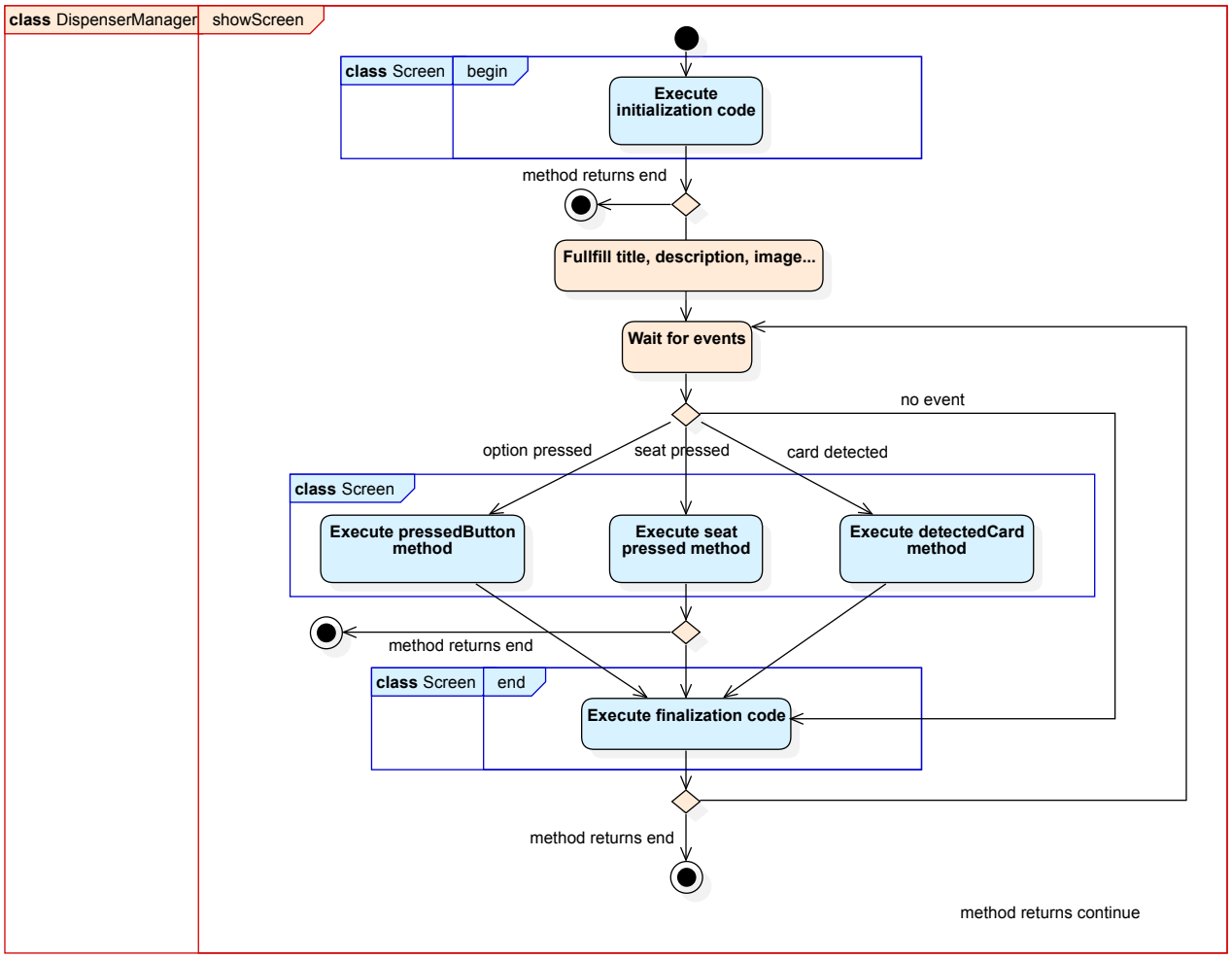

Figura 6.- Diagrama de actividad con la lógica del método showScreen de la clase DispenseManager.

3.2 Operativa del traductor

El diagrama de la Figura 7 muestra el funcionamiento del sistema de traducción. Previamente, el `DispenseManager` debe cargar los ficheros de diccionario en cada uno de los `Translator`.

Cada fichero de diccionario consta de todas las frases que pueden aparecer en el programa en inglés y en otro idioma.

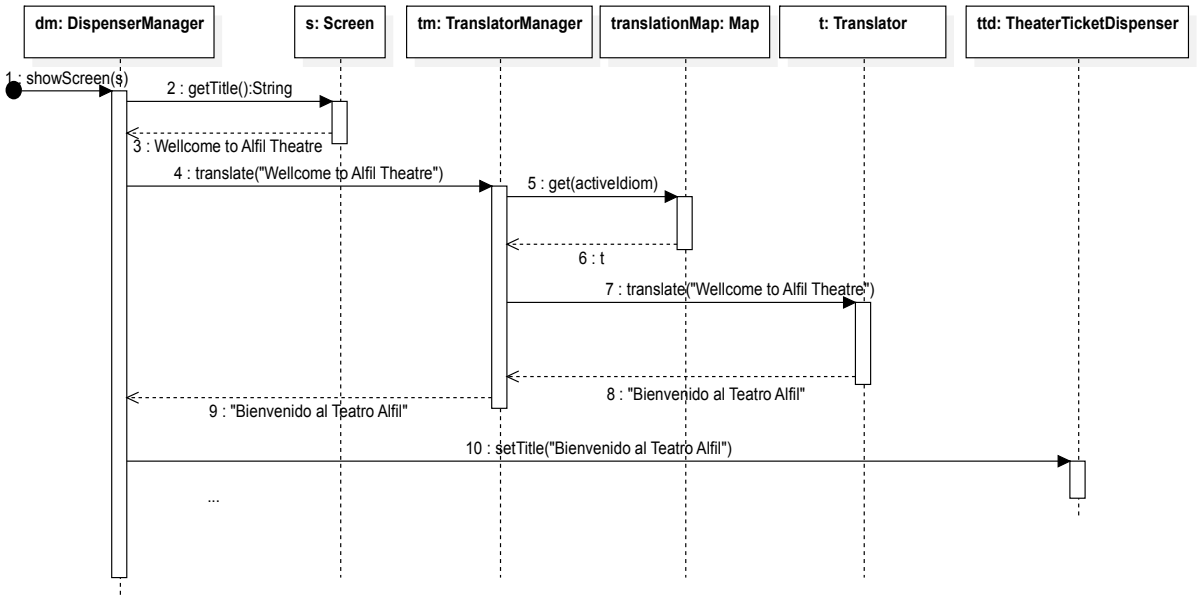

Figura 7.- Diagrama de secuencia que refleja la traducción de una frase.

3.3 Inicio de la aplicación

El diagrama de secuencia de la Figura 8 ilustra el inicio de la aplicación.

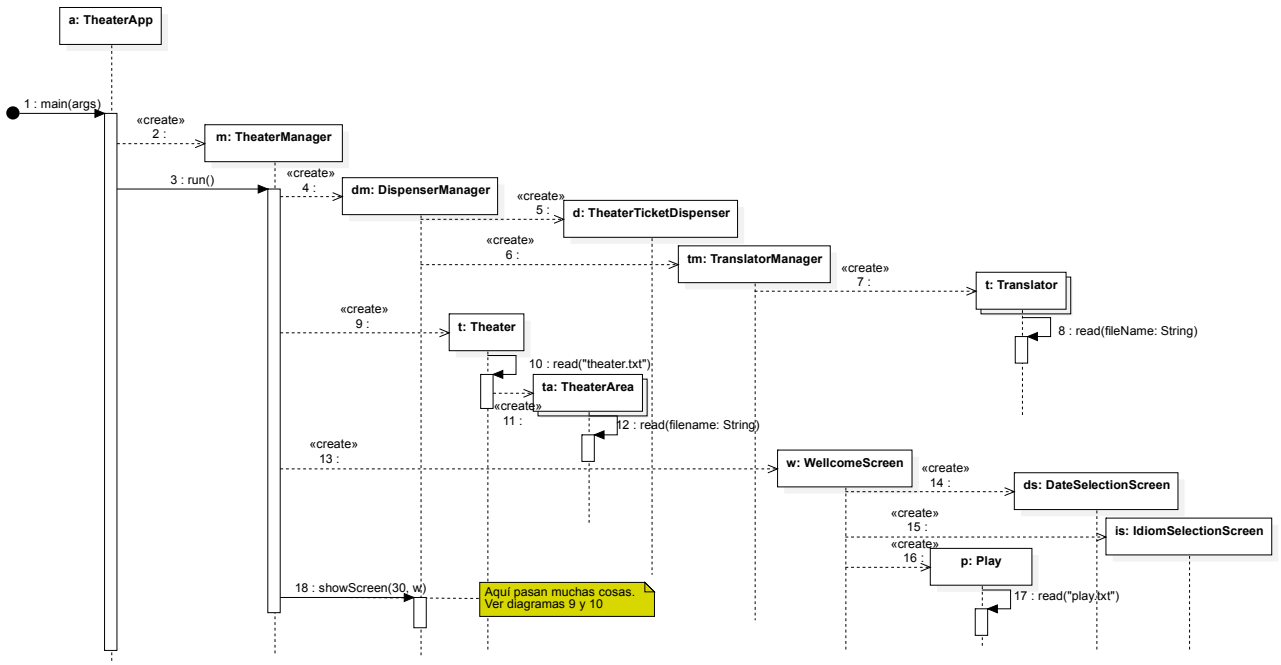

Figura 8.- Diagrama de secuencia que refleja el inicio del programa.

3.4 Operativa de la pantalla WellcomeScreen

El diagrama de la Figura 9 ilustra el mostrado de la pantalla `WellcomeScreen` y el pulsado del botón de elegir una fecha.

El diagrama de la Figura 10 ilustra el mostrado de la pantalla `WellcomeScreen` y la espera de 30 segundos que termina sin que nadie pulse ningún botón.

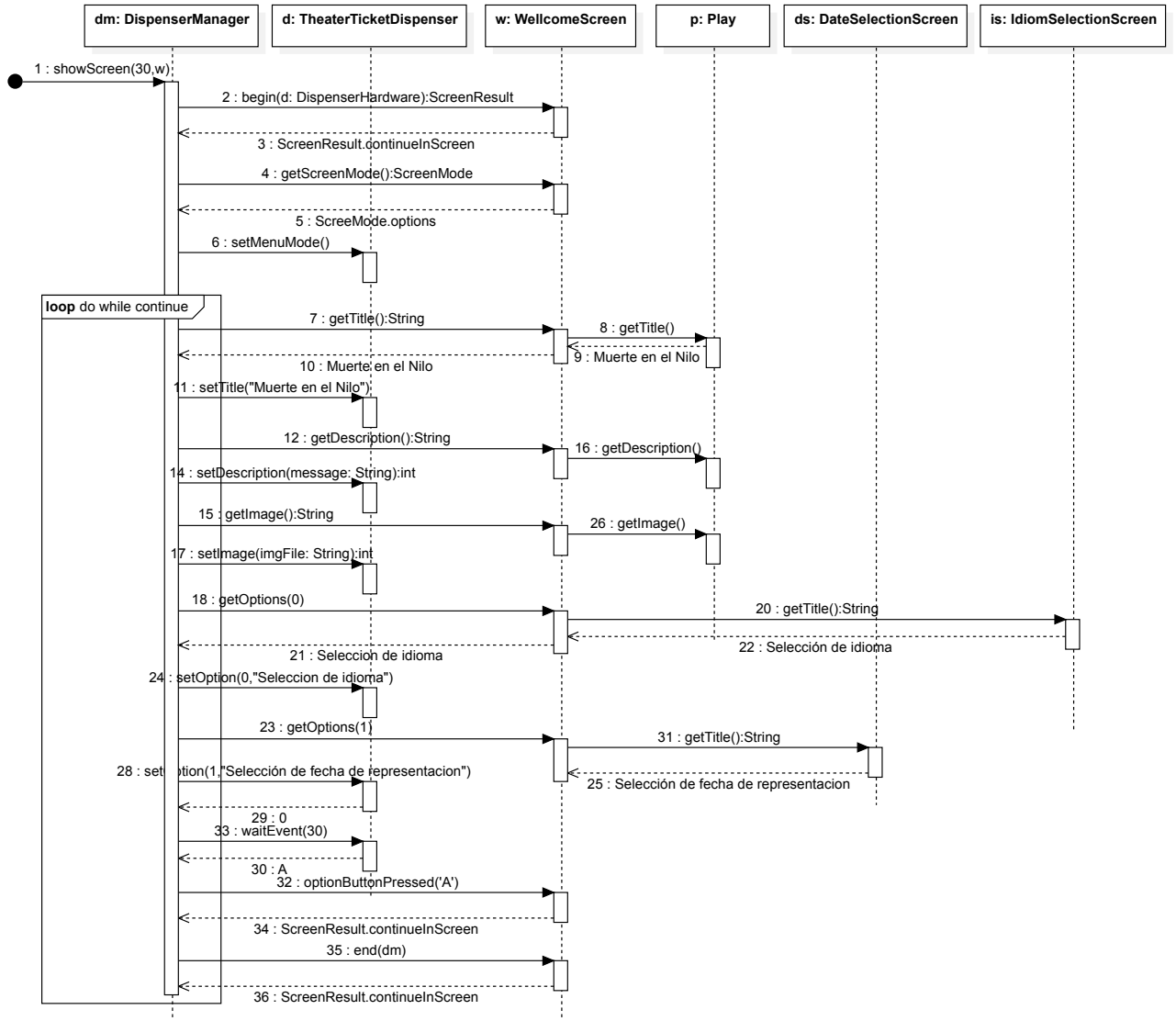

Figura 9.- Diagrama de secuencia que refleja la presentación de la pantalla de Wellcome y la pulsación del botón de seleccionar fecha.

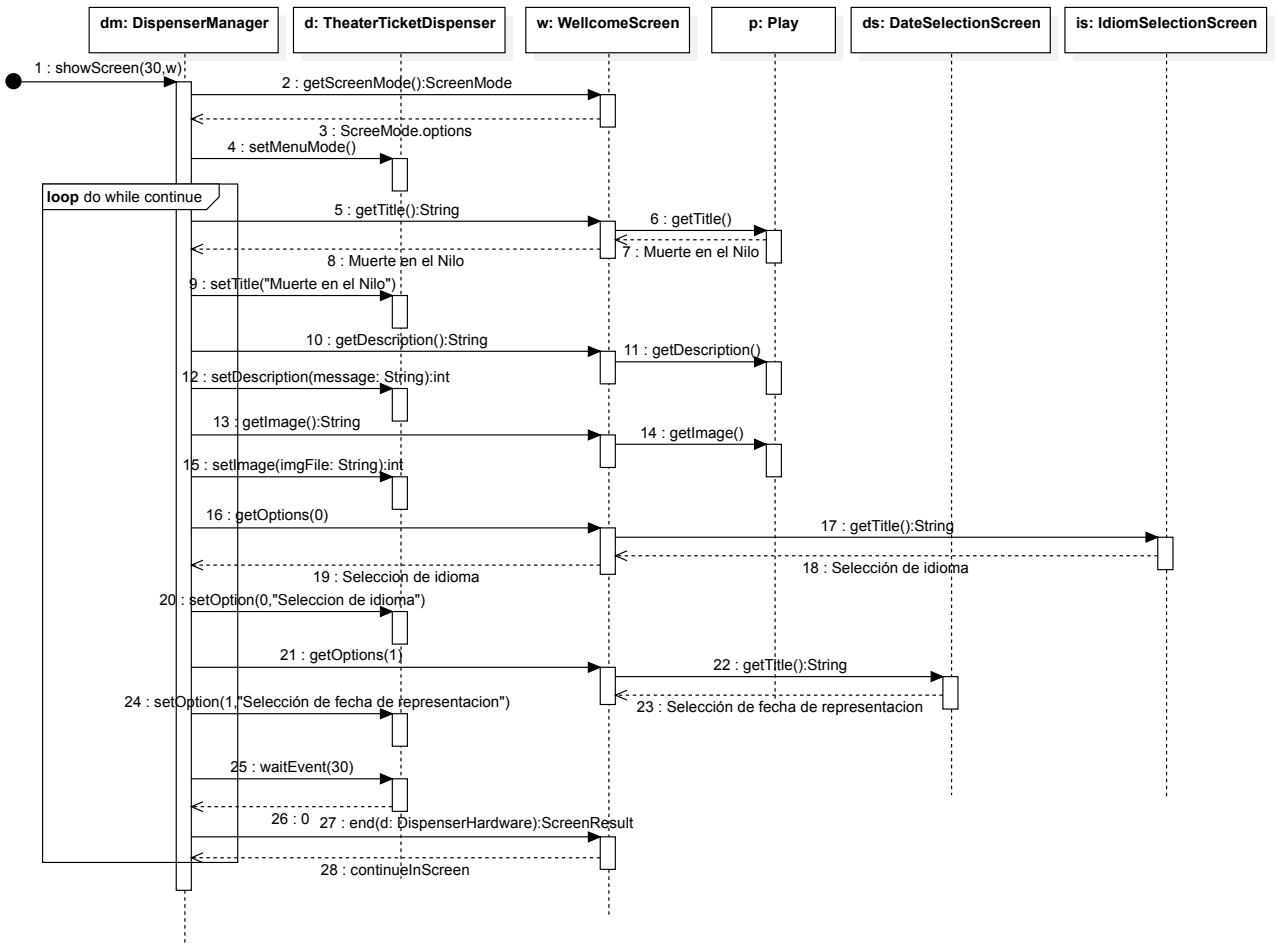

Figura 10.- Diagrama de secuencia que refleja la presentación de la pantalla de Wellcome.

La Figura 11 presenta las operaciones que se desencadenan cuando se pulsa el botón de elegir una fecha de representación en la pantalla de bienvenida.

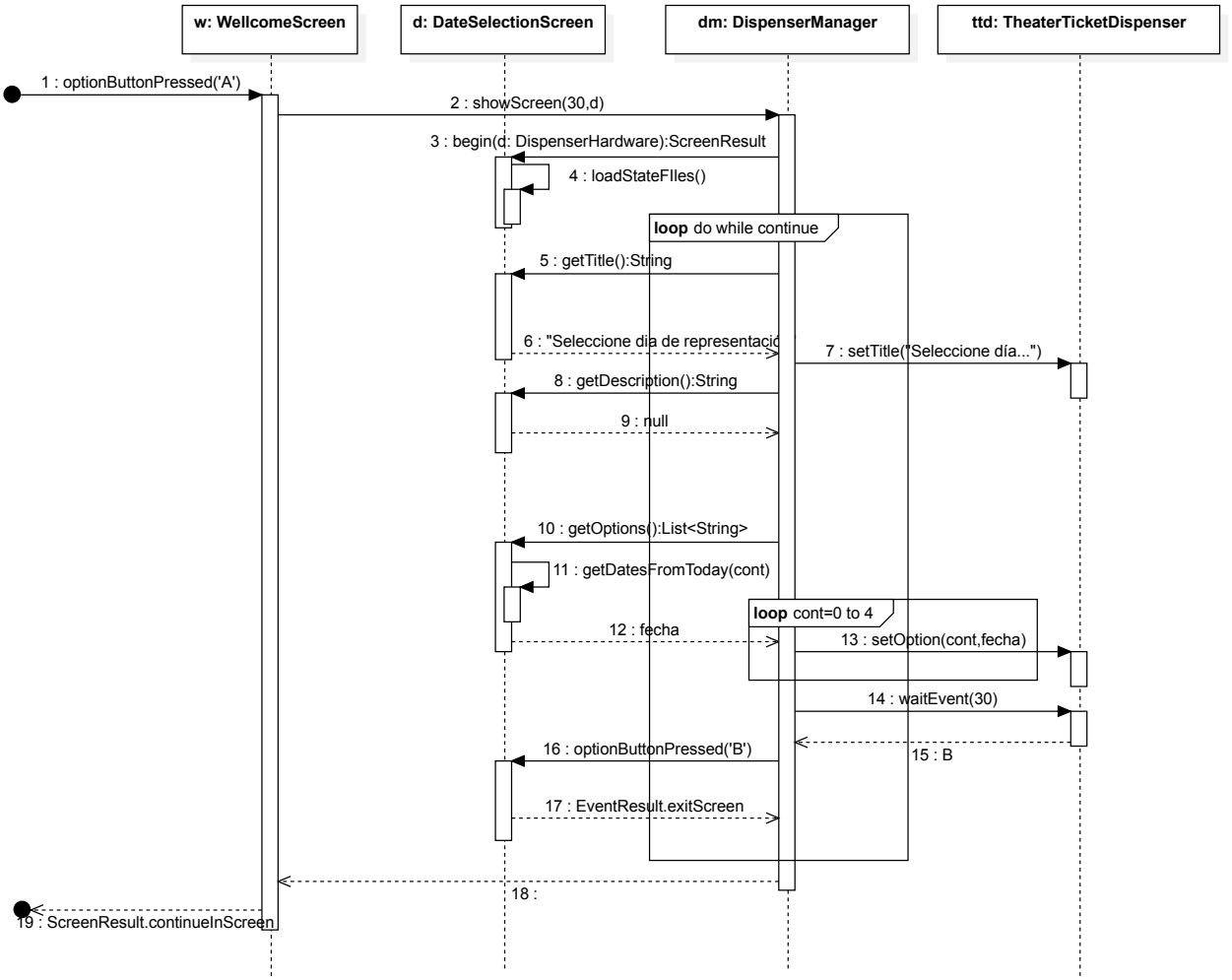

Figura 11.- Diagrama de secuencia que refleja la selección de una fecha para la compra de unas butacas.

3.5 Operativa de la pantalla de selección de asientos y pago

La Figura 12 muestra la operativa que se desencadena al seleccionar un asiento y pulsar el botón de aceptar. Obsérvese que las llamadas al método 13 (`drawArea`) o al método 18 (`seatButtonPressed`) contendrían mucha lógica relativa a marcar la butaca que no se ve reflejado en este diagrama.

Finalmente, la Figura 13 muestra el pago con tarjeta.

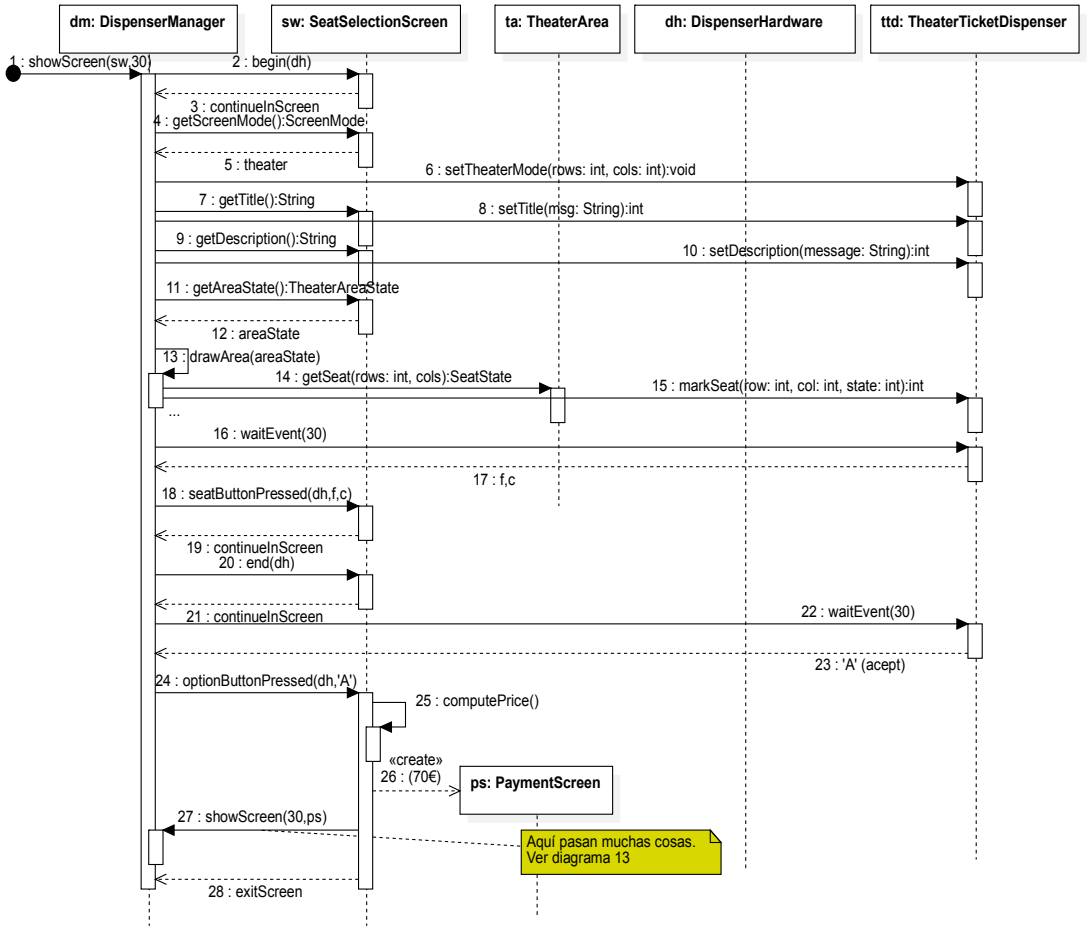

Figura 12.- Diagrama de secuencia que refleja la compra de dos butacas en una sola transacción.

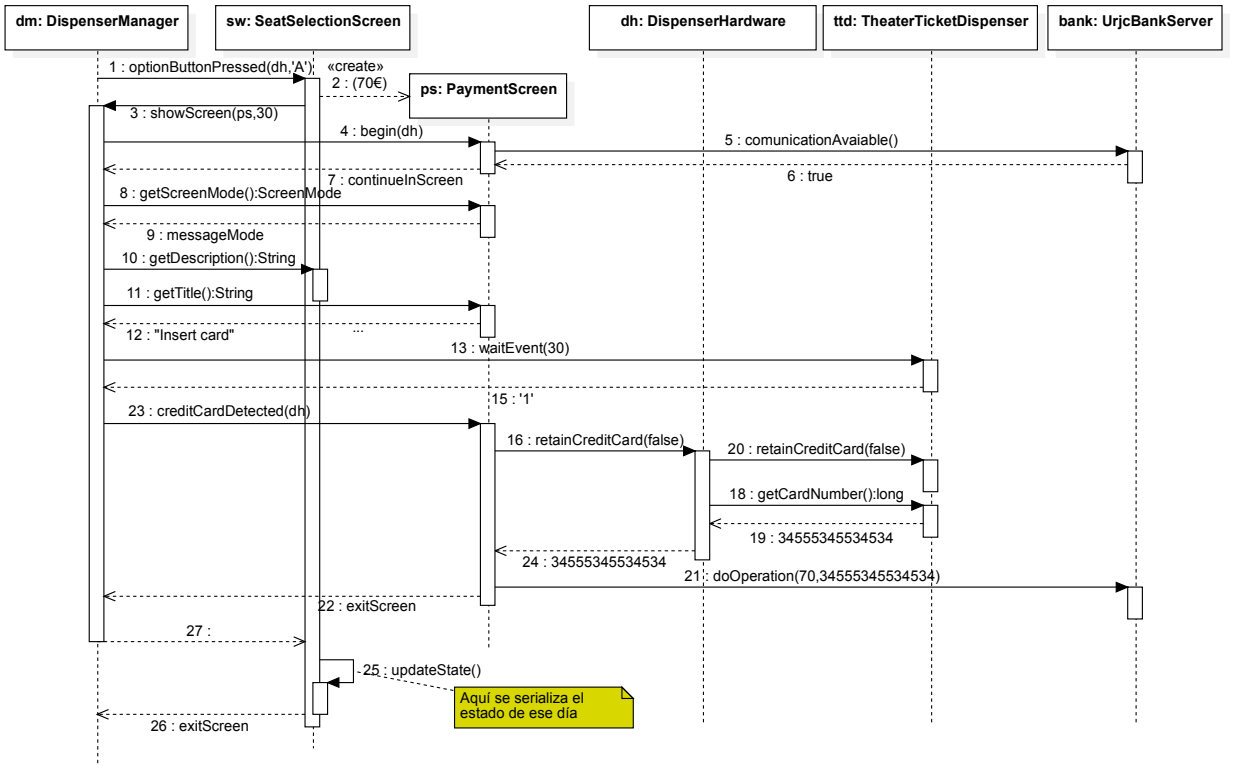

Figura 13.- Diagrama de secuencia que refleja el pago con tarjeta.

4 Bibliotecas utilizadas

Junto con este PDF se proporcionan 3 ficheros JAR y un proyecto de ejemplo que los usa. Los JAR corresponden a las bibliotecas `TheaterTicketDispenser` y `UrjcBankServer` y a la biblioteca `AbsoluteLayout` utilizada por `TheaterTicketDispenser`. El proyecto de ejemplo simplemente está configurado para realizar diferentes operaciones sobre `TheaterTicketDispenser` al pulsar los botones que presenta.

La biblioteca `UrjcBankServer` es un simulador de una API de comunicación con el banco. Dicha API permite realizar las operaciones explicadas en el enunciado de la práctica. Además, se ha añadido una nueva operación necesaria para consultar el balance total de una cuenta corriente.

La biblioteca `TheaterTicketDispenser` es un simulador del dispensador de tickets. Como se puede ver en la Figura 14, dicho simulador está compuesto de 3 ventanas.

- La primera ventana permite seleccionar una de 4 tarjetas de crédito para insertarlas en el dispensador. Al pinchar sobre ellas la tarjeta se introduce en el dispensador.
- La segunda ventana, que es la principal, permite manipular el dispensador. En particular, permite pulsar botones y retirar la tarjeta (pulsando sobre ella).
- Finalmente, cuando se pide imprimir un ticket aparece una nueva ventana que permite ver el resultado.

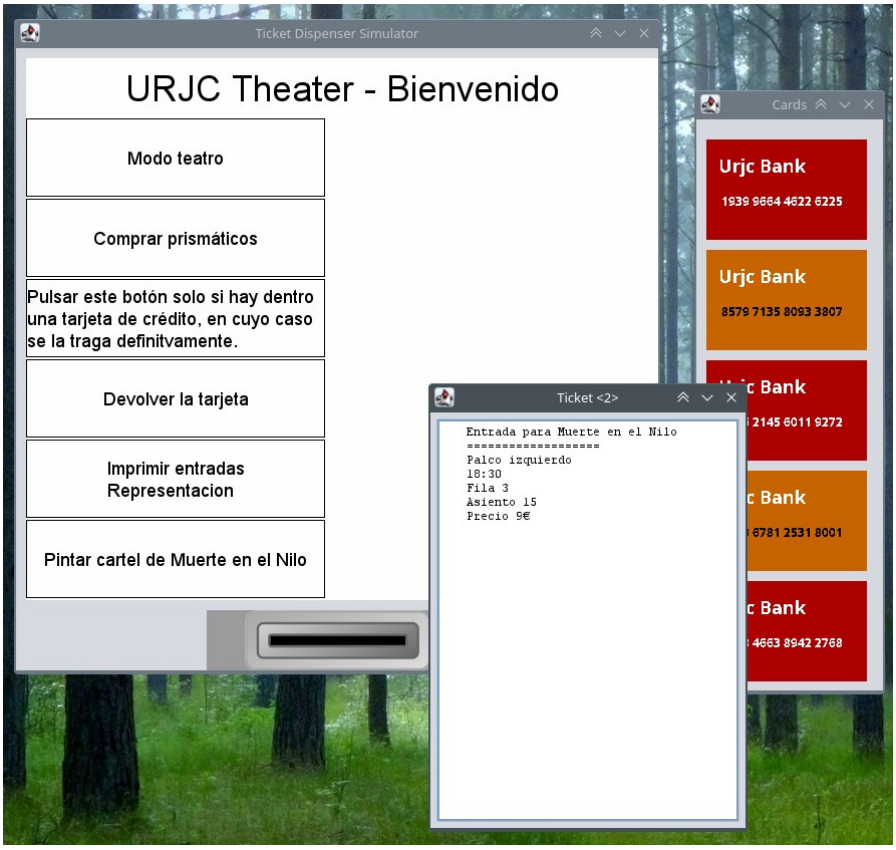

Figura 14.- Posibilidades de la interfaz de usuario con botones configurados al azar a modo de ejemplo.

5 Conclusiones

Este documento presenta un diseño orientativo de la práctica del dispensador de entradas de teatro. Hay muchos aspectos por perfilar que quedan fuera del alcance de este documento para obtener un resultado realista. Dichos aspectos los debe resolver el estudiante en la implementación.

Podéis comenzar a trabajar, aunque el primer día de clase tras la publicación de este documento se discutirán todos los aspectos que se consideren.