



Elaboración de diagramas de comportamiento

Objetivos

- Identificar los tipos de diagramas de comportamiento que se pueden construir en UML.
- Interpretar y elaborar diagramas de casos de uso.
- Interpretar diagramas de interacción.
- Elaborar diagramas de interacción sencillos.
- Interpretar diagramas de estados.
- Diseñar diagramas de estados sencillos.
- Interpretar diagramas de actividades.
- Elaborar diagramas de actividades sencillos.

Contenidos

- 6.1. Diagramas de comportamiento
- 6.2. Diagramas de casos de uso
- 6.3. Diagramas de interacción
- 6.4. Diagramas de estados
- 6.5. Diagramas de actividades

Introducción

En la Unidad 5, se vio que el lenguaje UML se utiliza para elaborar diferentes tipos de diagramas que son necesarios para realizar las tareas de análisis y diseño de una aplicación informática. Entre ellos, se estudiaron en detalle los diagramas de clases, los cuales son diagramas estructurales que se usan para representar la parte estática del sistema. Paralelamente, esta unidad se dedicará a los **diagramas de comportamiento**, es decir, aquellos que reflejan la parte dinámica del sistema.

En primer lugar, se presentan los *diagramas de casos de uso*, que son fundamentales en todo el proceso de desarrollo del software y reflejan las funciones encomendadas al sistema desde el punto de vista de quienes utilizan la aplicación. Después, se caracterizan los *diagramas de interacción* (diagramas de secuencia y de colaboración), cuya función es informar con detalle sobre el comportamiento de cada caso de uso. Por último, se estudian los *diagramas de estados* y los *de actividades*.

6.1. Diagramas de comportamiento

Los diagramas de comportamiento UML sirven para visualizar, especificar, construir y documentar los aspectos dinámicos de un sistema. La Tabla 6.1 recoge los siete tipos de diagramas de comportamiento disponibles y la utilidad de cada uno de ellos.

Tabla 6.1. Tipos de diagramas de comportamiento que se pueden construir en lenguaje UML y su uso

Diagrama	Uso
Diagramas de casos de uso	Describen las funcionalidades del sistema desde el punto de vista de la persona usuaria. Son fundamentales, pues se usan como punto de partida para los demás diagramas.
Diagramas de actividades	Se emplean para especificar paso a paso una operación compleja. Sirven para modelar el flujo de un caso de uso o entre casos de uso.
Diagramas de estados	Sirven para modelar el comportamiento de un objeto dirigido por eventos que provocan cambios de estado o transiciones.
Diagramas de secuencia	Muestran la secuencia cronológica de mensajes entre objetos durante un escenario de un caso de uso.
Diagramas de colaboración	Muestran un conjunto de objetos, los enlaces entre ellos y los mensajes que se reciben y se envían entre ellos.
Diagramas de tiempos	Representan el comportamiento de distintos objetos en un determinado periodo de tiempo.
Diagrama global de interacciones	Aportan una visión general de la interacción en el sistema.

A lo largo de esta unidad, se estudiarán los diagramas de comportamiento de mayor relevancia en la construcción de un sistema.

6.2. Diagramas de casos de uso

Los diagramas de casos de uso son fundamentales en el proceso de desarrollo de software orientado a objetos. De hecho, una de las características más importantes de la metodología RUP (proceso unificado de Rational), descrita en el Apartado 1.9.1, es que el proceso RUP está dirigido por los casos de uso. Así, durante la tarea de análisis, la captura de los requisitos se realiza a través de los diagramas de casos de uso, y se descubren y definen clases a medida que se leen las descripciones de los casos de uso. Durante las tareas de diseño y programación, quienes desarrollan el software crean modelos para dar respuesta a los requisitos incluidos en los diagramas de casos de uso. Finalmente, en la tarea de pruebas se garantiza que la aplicación implementa correctamente los casos de uso.

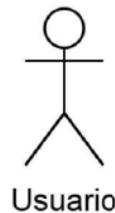
Se puede afirmar que los casos de uso sirven de guía en un conjunto de actividades de desarrollo:

- Creación y validación de la arquitectura del sistema.
- Definición de procedimientos y casos de prueba.
- Planificación de iteraciones.
- Generación de la documentación de usuario.
- Despliegue del sistema.

Una característica muy relevante de los casos de uso es que proporcionan trazabilidad al sistema, esto es, por medio de ellos, se puede determinar qué parte del sistema se ha creado a partir de cada caso de uso: qué parte del diagrama de clases, qué parte del programa, qué casos de prueba, etc. De esta manera, se puede detectar al realizar un cambio en un caso de uso, qué clases y componentes es necesario modificar.

Los principales elementos de estos diagramas son dos:

1. Los **casos de uso** describen, bajo la forma de acciones y reacciones, el comportamiento de un sistema desde la perspectiva de la persona que lo usa. Son descripciones de la funcionalidad del sistema independientes de su arquitectura. Como se señaló en el Apartado 5.2.1, los casos de uso se representan mediante una elipse con borde continuo. En el interior de la elipse o a su lado, se debe escribir el nombre del caso de uso.
2. Los **actores** se determinan observando las personas usuarias directas del sistema, las personas responsables del uso o mantenimiento del sistema y otros sistemas que interactúan con el sistema en cuestión. Hay que tener en cuenta que una misma persona puede desempeñar varios papeles distintos y, por lo tanto, se representaría por medio de distintos actores. Cada actor lleva asociado un nombre que describe el papel que desempeña. Los actores se representan como se muestra en la Figura 6.1, escribiéndose el nombre del actor en la parte inferior.



Usuario

Figura 6.1. Símbolo que representa a un actor en un diagrama de casos de uso con su nombre debajo.

Un diagrama de casos de uso para una aplicación ofrece un panorama de todos los casos de uso y sus relaciones. Por lo tanto, en él, se representa toda la funcionalidad que proporciona el sistema. Todos los casos de uso se colocan en el interior de un recuadro que representa al sistema que se está construyendo, de manera que los actores quedan fuera de este recuadro, pues se encuentran fuera del sistema.

Entre los elementos integrantes de un diagrama de casos de uso (casos de uso y actores) se pueden establecer varios tipos de relaciones:

- **Relación de comunicación:** este tipo de relaciones se usa para indicar qué actor o actores intervienen en cada caso de uso. Se representa mediante una línea que une el actor con el caso de uso.
- **Relación de uso o inclusión:** se crea este tipo de relación cuando se dispone de varios casos de uso con una serie de actividades comunes. En estos casos, se crea un caso de uso que incluye la actividad o actividades duplicadas y, luego, se indica que los otros casos de uso incluyen o hacen uso de este nuevo caso de uso que se crea. Esta relación de inclusión se representa mediante una línea con trazado discontinuo y una punta de flecha hacia el caso de uso con la actividad o actividades duplicadas. Esta línea lleva la leyenda o el estereotipo «*include*».
- **Relación de extensión:** este tipo de relación se emplea cuando un caso de uso origen (A) amplía o extiende la funcionalidad del caso de uso destino (B), es decir, incluye algunos pasos adicionales con respecto al caso de uso destino. Esta relación se representa mediante una línea con trazado discontinuo y una punta de flecha hacia el caso de uso destino. Esta línea lleva la leyenda o el estereotipo «*extend*». El caso de uso A no es imprescindible que ocurra y, cuando ocurre, realiza funciones adicionales a las del caso de uso B.

A modo de ejemplo, en una entidad bancaria en la que los clientes pueden realizar diferentes operaciones con sus cuentas, como transferencias, ingresos y extracciones de dinero, se dispondría de tres casos de uso correspondientes a esas tres funcionalidades (transferencia, ingreso y extracción). Para poder realizar las operaciones de transferencia y extracción, los clientes se tienen que identificar. Por lo tanto, se podría crear un caso de uso que incluyera esa parte común a esos dos casos de uso, que se puede llamar *identificación*. En el caso de las transferencias, también se pueden realizar por internet, por lo que habría que crear el caso de uso *transferencia por internet*, que incluiría algunos pasos adicionales con respecto al caso de uso transferencia. El diagrama de casos de uso quedaría como el que se muestra en la Figura 6.2.

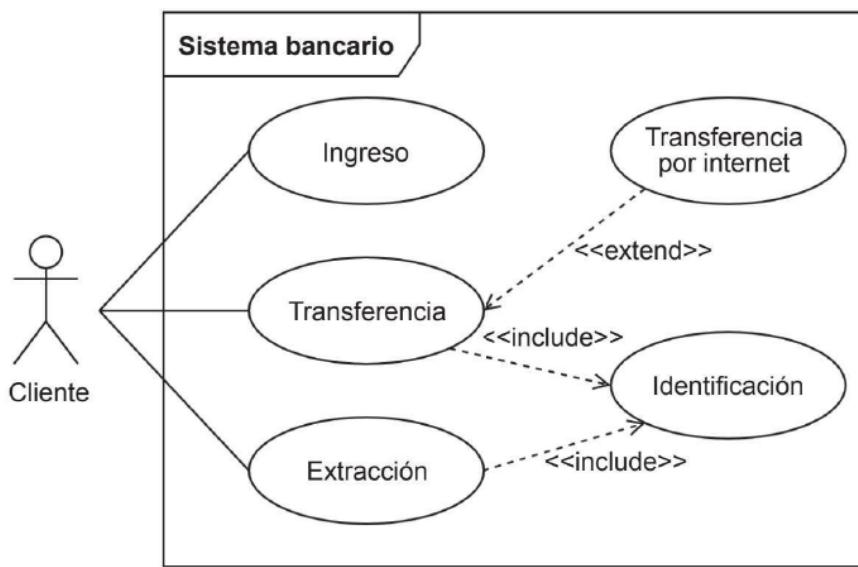


Figura 6.2. Diagrama de casos de uso para un sistema bancario en el que las y los clientes pueden realizar ingresos, extracciones o transferencias de dinero.

Un caso de uso debe ser simple, inteligible, claro y conciso. Lo habitual es que haya pocos actores asociados a cada caso de uso. Hay una serie de preguntas clave que es preciso plantearse al crear un diagrama de casos de uso:

- ¿Cuáles son las tareas del actor?
- ¿Qué información crea, guarda, modifica, destruye o lee el actor?
- ¿Debe el actor notificar al sistema los cambios externos?
- ¿Debe el sistema notificar al actor los cambios internos?

Además de obtener el diagrama de casos de uso para una aplicación, es necesario describir para cada caso de uso lo siguiente:

- Los actores involucrados.
- La precondición, es decir, el estado inicial del que parte el caso de uso.
- El flujo básico, es decir, el flujo de eventos ordenados por los que pasa el caso de uso.
- Los caminos alternativos, es decir, las posibles desviaciones con respecto al flujo básico.
- La poscondición, esto es, el estado final al que llega el caso de uso.

Los casos de uso permiten comprobar la **verificación** y **validación del sistema**. En cuanto al primer aspecto (verificación), permiten determinar que el sistema se ha desarrollado correctamente, es decir, que las tareas se llevan a cabo de manera adecuada. En lo que respecta a la validación, permiten comprobar si el sistema es realmente el que la persona usuaria desea. O dicho de otro modo, la validación consiste en determinar si el sistema hace lo que quiere la persona que lo va a utilizar, mientras que la verificación consiste en determinar

si lo que hace el sistema lo hace bien, es decir, si lo hace como la persona usuaria quiere que se haga.

El modelo de casos de uso consta de un diagrama de casos de uso y la descripción de todos los casos de uso. Cuando se termina de elaborar el modelo, se presenta a las personas que van a usar la aplicación y a los clientes para que lo validen, es decir, para que determinen si cubre sus necesidades y si les ofrece la funcionalidad deseada.

Actividad resuelta 6.1

Diagrama de casos de uso y descripciones de los casos de uso

Se desea crear un programa con una agenda de antiguos compañeros y compañeras de estudios con los que se suele realizar una cena anual para mantener la amistad. Por cada una y uno de ellos, se desea registrar en la agenda su nombre completo (nombre y dos apellidos), número de teléfono y correo electrónico. No puede haber dos excompañeros o dos excompañeras con el mismo nombre.

Las operaciones que deberá realizar la aplicación son las que se indican a continuación:

- *Agregar excompañero/a*: para esta operación, se deberán suministrar todos los datos indicados anteriormente.
- *Consultar datos de un/a excompañero/a*: se proporciona su nombre completo y se mostrarán su número de teléfono y correo electrónico.
- *Agregar cena*: para esta operación, se deberá suministrar la fecha de la cena, el lugar en el que se va a celebrar y el nombre de la persona que la organiza.
- *Eliminar cena*: para esta, se deberá suministrar el año de la cena que se desea eliminar.
- *Consultar cena*: se suministrará el año de la cena y se mostrarán sus datos (fecha, lugar y nombre del organizador).
- *Agregar asistente*: para esta operación, es necesario indicar el año de la cena y el nombre del nuevo asistente.
- *Eliminar asistente*: para esta, habrá que suministrar el año de la cena y el nombre del asistente que al final no va a acudir a la cena.
- *Consultar asistentes*: se deberá indicar el año de la cena y se mostrarán los nombres de los asistentes a esta.

Asimismo, deben tenerse en cuenta las siguientes restricciones:

- No puede haber dos excompañeros o excompañeras con el mismo nombre completo (nombre y dos apellidos), por lo que antes de agregar una persona, se deberá comprobar que no están ya almacenados sus datos.
- No puede haber dos cenas el mismo año. Se deberá realizar la comprobación pertinente a la hora de agregar una cena.
- La persona encargada de organizar una cena debe ser una previamente agregada al programa. Esto se deberá comprobar al agregar una nueva cena.
- Todos los asistentes a las cenas deben ser antiguos compañeros y compañeras ya agregados al programa. Esto se deberá comprobar al agregar un nuevo asistente.

A continuación, se detallan cada uno de los casos de uso considerando las restricciones indicadas:

- *Agregar excompañero/a*: en este caso de uso, se solicita al usuario o usuaria la introducción del nombre completo, número de teléfono y correo electrónico del excompañero o excompañera. En caso de que haya uno o una con ese mismo nombre, se muestra un mensaje de error; en caso contrario, se añade el antiguo compañero o compañera.
- *Consultar excompañero/a*: se solicita a la persona usuaria la introducción del nombre completo del excompañero o excompañera cuyos datos se desean ver. En caso de que no haya ninguno o ninguna con ese nombre, se muestra un mensaje de error; en caso contrario, se muestran el número de teléfono y el correo electrónico del antiguo compañero o compañera.
- *Agregar cena*: se solicita la fecha de la cena. Si ya hay cena ese año, se mostrará un mensaje de error; si no la hubiera, la aplicación pedirá al usuario que ingrese el lugar de la cena y el nombre de la persona que la organiza. Si el nombre de esta no se corresponde con el de un excompañero o excompañera existente, aparecerá un mensaje de error; en el caso contrario, se guardan los datos de la cena.
- *Eliminar cena*: se pide el año de la cena que se desea eliminar. Si no hubiera una cena ese año, se mostrará un mensaje de error; en caso contrario, se eliminará la cena.
- *Consultar cena*: para este caso de uso, se solicita el año de la cena. En caso de que no haya ninguna cena ese año, aparecerá un mensaje de error; en caso contrario, se mostrarán la fecha de la cena, el lugar donde se realizará y el nombre de la persona organizadora.
- *Agregar asistente*: se pide que se indique el año de la cena. Si no hay cena en ese año, se mostrará un mensaje de error; en el caso contrario, se preguntará el nombre del nuevo asistente a la cena. En caso de que el nombre del asistente no coincida con alguno de los y los excompañeros agregados, saldrá un mensaje de error. En caso de que ya esté anotado ese asistente para la cena, se indicará mediante un mensaje; en caso contrario, se añadirá dicho asistente a la cena.
- *Eliminar asistente*: de nuevo, se solicitará el año de la cena. Si ese año no hay ninguna cena, aparecerá un mensaje de error; y, en caso contrario, se solicitará el nombre del asistente a la cena. Si esa persona no figura como asistente a la cena, se mostrará el pertinente mensaje de error; en el caso de que sí figure como asistente, se eliminará ese asistente de la cena.
- *Consultar asistentes*: en este caso de uso, se pide el año de la cena. En caso de que no haya ninguna cena ese año, se mostrará el correspondiente mensaje de error, y en caso contrario, por cada uno de los asistentes a la cena, se mostrará su nombre.

Se pide realizar el diagrama de casos de uso y la descripción de los casos de uso *Agregar excompañero/a*, *Consultar excompañero/a*, *Agregar cena*, *Eliminar cena* y *Agregar asistente*.

Solución

En la Figura 6.3, se muestra el modelo de casos de uso de la aplicación.

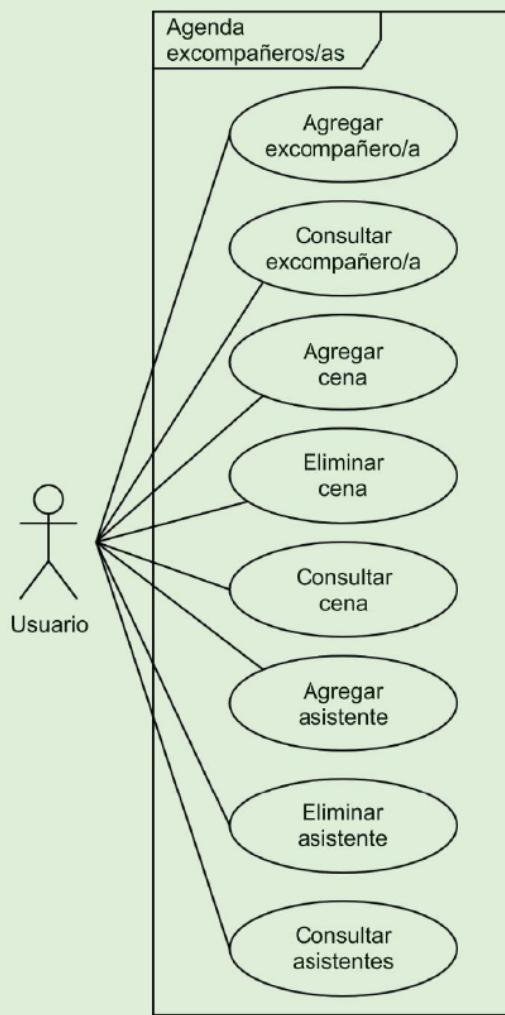


Figura 6.3. Diagrama de casos de uso para un sistema en el que se desean almacenar datos de antiguos compañeros y compañeras con los que se celebran cenas anualmente.

La descripción de los casos de uso solicitados sería como se expone a continuación.

Caso de uso Agregar excompañero/a

- *Actores involucrados:* persona usuaria.
- *Precondición:* la agenda tiene que estar creada.
- *Flujo básico:*
 1. La persona usuaria solicita agregar un nuevo excompañero o excompañera.
 2. Se pide a la persona usuaria de la aplicación la introducción del nombre completo, teléfono y correo electrónico del nuevo excompañero o excompañera.
 3. Tras la introducción de los datos, el sistema comprueba que no haya una persona con el mismo nombre y, si es así, lo añade a la agenda y muestra un mensaje de confirmación en pantalla.
 4. Se vuelve a solicitar al usuario o usuaria la selección de una operación.

■ Caminos alternativos:

- Si en el paso 3 se detecta que ya existe un excompañero o excompañera con ese nombre, se muestra un mensaje que el usuario debe aceptar y se pasa al paso 4 anterior.
- **Poscondición:** la agenda dispone de un excompañero o excompañera más o del mismo de número de antiguas y antiguos compañeros si la persona ya estaba en la agenda.

Caso de uso *Consultar excompañero/a*

■ Actores involucrados: persona usuaria.**■ Precondición:** la agenda tiene que estar creada y deben existir excompañeras o excompañeros agregados.**■ Flujo básico:**

1. El usuario o usuaria solicita consultar los datos de un antiguo compañero o compañera.
2. Se pide a la persona usuaria la introducción del nombre completo del excompañero o excompañera (nombre y apellidos) que se busca.
3. El sistema comprueba si hay alguna persona con ese nombre y muestra sus datos: teléfono y correo electrónico.
4. Se vuelve a solicitar a la persona usuaria la selección de una operación.

■ Caminos alternativos:

- Si en el paso 3 anterior se detecta que no existe ninguna persona con ese nombre, se muestra un mensaje que el usuario o usuaria debe aceptar y se pasa al paso 4.

■ Poscondición: el estado del sistema no cambia.

Caso de uso *Agregar cena*

■ Actores involucrados: persona usuaria.**■ Precondición:** la agenda tiene que estar creada y debe existir algún excompañero agregado o excompañera agregada.**■ Flujo básico:**

1. El usuario o la usuaria solicita introducir una nueva cena.
2. Se pide a la persona usuaria la introducción de la fecha y el lugar de la cena.
3. El sistema comprueba que no haya ninguna cena ese mismo año.
4. Se pide a la persona usuaria la introducción del nombre de la persona organizadora de la cena.
5. El sistema comprueba que haya alguna persona con el nombre del organizador de la cena y si la hay, se añaden los datos de la cena y se muestra un mensaje de confirmación en la pantalla.
6. Se vuelve a solicitar al usuario o usuaria que seleccione una operación.

 Caminos alternativos:

- Si en el paso 3 se detecta que ya existe una cena ese año, se muestra un mensaje que el usuario o usuaria debe aceptar y se pasa al paso 6.
 - Si en el paso 5 se detecta que no existe ninguna persona con el nombre de la persona organizadora, se muestra un mensaje que la persona usuaria debe aceptar y se pasa al paso 6.
-  Poscondición: la agenda dispone de una cena más o del mismo de número de cenas si esta no se ha podido añadir.

 Caso de uso *Eliminar cena* Actores involucrados: persona usuaria. Precondición: la agenda tiene que estar creada y deben existir cenas. Flujo básico:

1. La persona usuaria solicita eliminar una cena.
2. Se pide al usuario o usuaria la introducción del año de la cena que se desea eliminar.
3. El sistema comprueba que haya una cena ese año y, si es así, se elimina la cena y se muestra un mensaje de confirmación en la pantalla.
4. Se vuelve a solicitar al usuario o usuaria la selección de una operación.

 Caminos alternativos:

- Si en el paso 3 se detecta que no existe ninguna cena ese año, se muestra un mensaje que el usuario o usuaria debe aceptar y se pasa al paso 4.

 Poscondición: la agenda dispone de una cena menos o del mismo de número de cenas si esta no se ha podido eliminar. Caso de uso *Agregar asistente* Actores involucrados: persona usuaria. Precondición: la agenda tiene que estar creada y deben existir cenas y excompañeras y excompañeros agregados. Flujo básico:

1. El usuario o usuaria solicita agregar un asistente a una cena.
2. Se pide al usuario o usuaria la introducción del año de la cena.
3. El sistema comprueba que haya una cena ese año.
4. Se solicita entonces a la persona usuaria la introducción del nombre de un excompañero o una excompañera.
5. El sistema comprueba que haya alguna persona con ese nombre.
6. El sistema comprueba que esa persona no figure ya entre los asistentes a la cena.

7. El sistema añade al excompañero o excompañera como asistente a la cena y muestra un mensaje de confirmación.

8. Se vuelve a solicitar a la persona usuaria la selección de una operación.

■ **Caminos alternativos:**

- Si en el paso 3 se detecta que no existe ninguna cena ese año, se muestra un mensaje que la persona usuaria debe aceptar y se pasa al paso 8.

- Si en el paso 5 se detecta que no existe ninguna persona con ese nombre, se muestra un mensaje que el usuario o usuaria debe aceptar y se pasa al paso 8.

- Si en el paso 6 se detecta que esa persona ya figura entre los asistentes a la cena, se muestra un mensaje que el usuario o usuaria debe aceptar y se pasa al paso 8.

■ **Poscondición:** en la cena indicada figura un asistente más o los mismos en caso de que no se haya podido añadir un asistente a la cena.

Actividad propuesta 6.1

Descripciones de casos de uso

A partir de la Actividad resuelta 6.1, proporciona la descripción de los casos de uso *Consultar cena*, *Eliminar asistente* y *Consultar asistentes*.

6.2.1. Herramientas para la elaboración de diagramas de casos de uso

En este apartado, se verá cómo se crean los diagramas de casos de uso mediante la herramienta diagrams.net y el módulo Papyrus SysML de Eclipse.

Elaboración de diagramas de casos de uso con diagrams.net

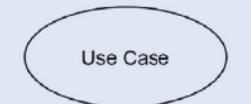
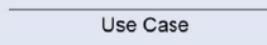
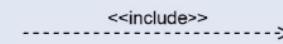
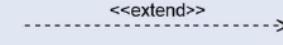
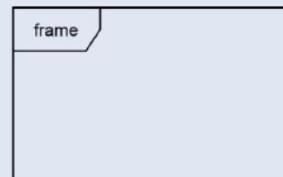
Cabe recordar aquí lo que se aprendió en el Apartado 5.6.1. En dicho apartado, se explicó cómo se generan diagramas de clases con la herramienta diagrams.net, a la que se puede acceder a través de la página web <https://app.diagrams.net/>.

Una vez dentro de la aplicación, tras indicar que se desea crear un nuevo diagrama y la ubicación donde se quiere almacenar, aparece una pantalla como la de la Figura 5.34.

A modo de ilustración, aquí se creará el diagrama de casos de uso para el sistema bancario de la Figura 6.2. En la Tabla 6.2 se muestran los elementos de las paletas de diagrams.net que se pueden utilizar en un diagrama.

Para crear un caso de uso, hay que hacer clic con el botón izquierdo del ratón sobre el ícono correspondiente a un caso de uso en la paleta UML. Ese ícono aparece entonces en el área de edición y se puede escribir en él su nombre. Para los actores, se debe hacer clic sobre el ícono del actor y asignarle su nombre.

Tabla 6.2. Elementos que forman parte de los diagramas de casos de uso con su icono y paleta correspondiente en diagrams.net

Elemento	Icono	Paleta
Caso de uso	 	UML
Actor		UML
Relación de comunicación	 Association / Connector / Instance Specification / Property / Connector End	UML 2.5
Relación de inclusión	 Include	UML 2.5
Relación de extensión	 Extend	UML 2.5
Sistema		UML

Una vez creados los actores y los casos de uso, hay que establecer las relaciones entre ellos. Para las relaciones de comunicación, se utiliza el ícono *Association/Connector* de la paleta *UML 2.5* que se muestra en la Tabla 6.2. Se puede cambiar el grosor de la línea, seleccionándola y reduciendo su número de puntos en la pestaña *Style* de las propiedades de la derecha (véase Figura 5.34).

Para establecer las relaciones de inclusión y de extensión, se emplean los elementos *Include* y *Extend* que aparecen al final de la paleta *UML 2.5*. El diagrama de casos de uso quedará como el que se muestra en la Figura 6.4.

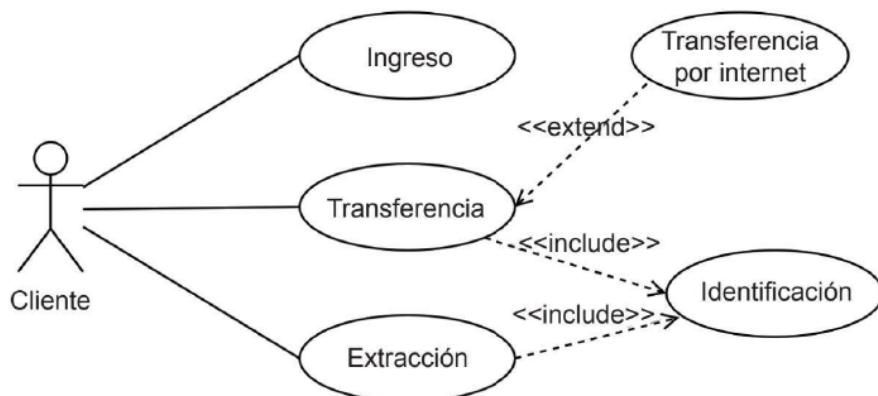


Figura 6.4. Diagrama de casos de uso para una entidad bancaria creado con diagrams.net.

Ya solo faltaría indicar los límites del sistema, pues debe recordarse que los actores no forman parte del sistema en un diagrama de casos de uso. A tal fin, se hace uso del elemento *Frame* de la paleta UML, como se muestra en la Tabla 6.2. Se debe clicar en este elemento, escribir el nombre que se quiere dar al sistema donde pone *frame* y mover el marco al lugar del área de edición. Allí se dimensionará este marco lo necesario para que abarque todo el diagrama excepto los actores. De esta manera, se habrá conseguido un diagrama de casos de uso completo como el de la Figura 6.2.

■■■ Elaboración de diagramas de casos de uso con Papyrus

En el Apartado 5.6.2. se vio que, para crear diagramas UML con el módulo Papyrus SysML de Eclipse, es necesario crear un proyecto por medio de la opción de menú *File* → → *New* → *Papyrus Project*. Al activar en la ventana emergente (véase Figura 5.42) la casilla de verificación correspondiente a *Use Case Diagram*, se indica que se quiere crear un diagrama de casos de uso.

En el caso de querer añadir un diagrama de casos de uso a un proyecto ya existente, una vez abierto este, se pueden añadir nuevos diagramas activando la opción del menú *Papyrus* → *New Diagram* y eligiendo el tipo de diagrama de que se trate.

En este ejemplo, se va a crear el mismo diagrama de casos de uso para un sistema bancario que se ha creado con diagrams.net. En la Tabla 6.3 se recogen los elementos de la paleta que se van a incluir en el diagrama y la sección de la paleta en la que se encuentra cada elemento.

Tabla 6.3. Elementos que forman parte de los diagramas de casos de uso con el símbolo correspondiente en Papyrus SysML y la sección de la paleta donde se encuentra

Elemento	Símbolo	Sección de la paleta
Caso de uso	oval Use Case	Nodes
Actor	stickman Actor	Nodes

Elemento	Símbolo	Sección de la paleta
Relación de comunicación		Links
Relación de inclusión		Links
Relación de extensión		Links
Sistema		Nodes

En primer lugar, se colocará en el diagrama el elemento que señala los límites del sistema. Para ello, se utilizará el elemento *Subject* de la sección *Nodes* de la paleta. Una vez seleccionado este elemento, y al hacer clic sobre el área de edición, aparecerá una ventana como la de la Figura 6.5, en la que habrá que indicar el tipo de elemento que se desea crear. Se debe elegir el tipo *Component*.

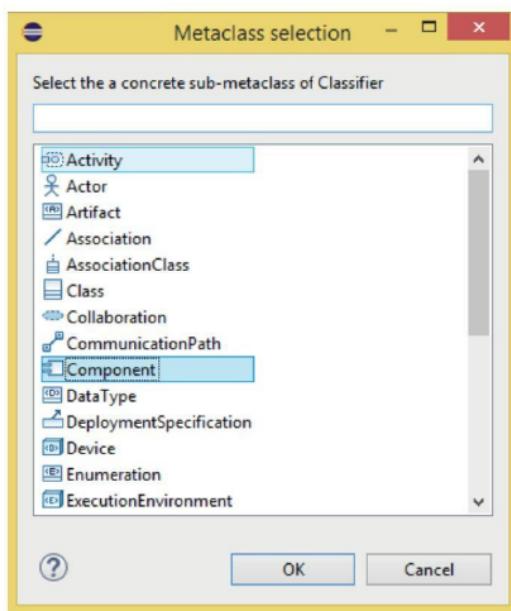


Figura 6.5. Ventana en la que se indica el tipo de elemento que se desea crear cuando se establecen los límites del sistema con el elemento *Subject* de la paleta. Se debe elegir el elemento *Component*.

A continuación, aparecerá dicho componente en el área de edición. En ese momento, se debe asignar un nombre al sistema y luego se podrá redimensionar para que abarque todo el diagrama, exceptuando el actor *Cliente*.

Cuando se añaden elementos al diagrama, se deben de colocar todos ellos, excepto el actor, dentro de los límites del sistema. Para crear un caso de uso, hay que clicar sobre el ícono correspondiente en la sección *Nodes* de la paleta. Luego, se debe hacer clic en el área delimitada por el sistema y se escribe el nombre del caso de uso. En el caso de los actores, se debe pulsar sobre el ícono del actor en la misma sección de la paleta y asignarle su nombre.

Una vez creados los actores y los casos de uso, se deben establecer las relaciones entre ellos. Para las relaciones de comunicación, se hará uso del ícono *Association* de la sección *Links* de la paleta que se muestra en la Tabla 6.3. Solo hará falta pulsar sobre este ícono, luego hacer clic en el actor y el correspondiente caso de uso en el área de edición.

Si se quiere establecer relaciones de inclusión y de extensión, se utilizarán los elementos *Include* y *Extend* que aparecen en la sección *Links* de la paleta.

De esta manera, se habrá conseguido un diagrama de casos de uso completo como el de la Figura 6.6.

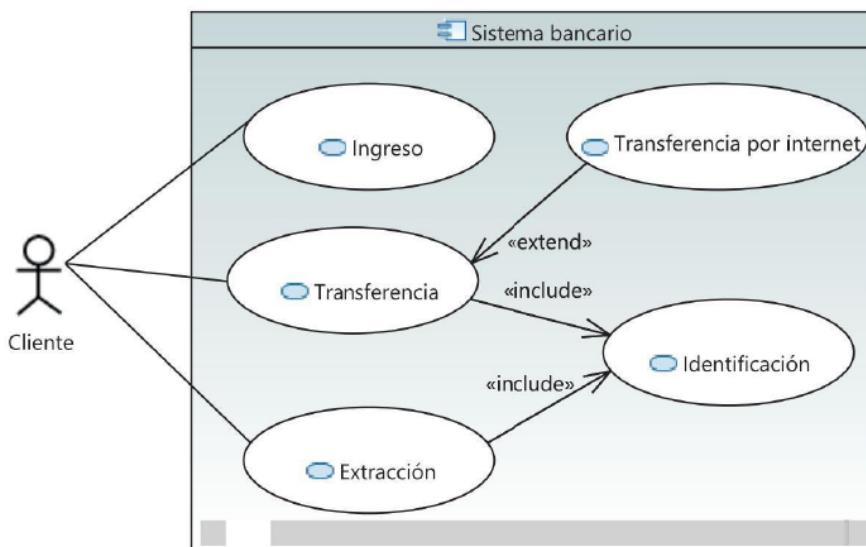


Figura 6.6. Diagrama de casos de uso para una entidad bancaria creado con Papyrus SysML en Eclipse.

6.3. Diagramas de interacción

Los diagramas de interacción muestran un conjunto de objetos y los mensajes que se intercambian entre ellos para describir el comportamiento del sistema.

Argot técnico



El nombre de **diagramas de interacción** que se asigna a estos diagramas en UML obedece a que estos diagramas muestran de qué manera interactúan los distintos objetos que forman parte de una aplicación para llevar a cabo las tareas encomendadas a ella. Hay que tener presente que una aplicación orientada a objetos se ejecuta por medio del envío de mensajes entre los diferentes objetos que la componen y que un mensaje es una solicitud enviada desde un objeto a otro o a él mismo para que ejecute un método.

Aunque en la última versión de UML se distinguen cuatro tipos de diagramas de interacción, los más relevantes son los **diagramas de secuencia** y los **de colaboración**, que se van a estudiar a continuación.

6.3.1. Diagramas de secuencia

Los diagramas de secuencia muestran la secuencia cronológica de los mensajes intercambiados entre objetos durante un escenario de un caso de uso, por lo que es posible afirmar que detallan el flujo de control para llevar a cabo lo especificado en un escenario de un caso de uso.

Argot técnico



Lo relevante en un **diagrama de secuencia** es precisamente la secuencia cronológica o el orden en el que se envían los mensajes para ejecutar lo especificado en un escenario de un caso de uso; de ahí su nombre.

En un diagrama de secuencia, para cada actor u objeto que interviene, se crea una línea vertical con trazo discontinuo llamada **línea de vida**, en la parte superior de la cual se dibuja un rectángulo que representa a una clase, o un actor, según el caso. En el interior del rectángulo se debe escribir el signo de puntuación dos puntos (:) y el nombre de la clase, que representa un objeto de dicha clase.

El tiempo transcurrido se representa de arriba hacia abajo. Los **mensajes** se representan mediante flechas de una clase a otra clase, indicando que la clase de la que parte la flecha envía un mensaje a la clase en la que acaba la flecha. Por medio de dichas flechas, se solicita la ejecución del método correspondiente de la clase receptora del mensaje.

Sobre las líneas de vida se pueden representar opcionalmente rectángulos en vertical que representan el tiempo durante el cual está activo un método, es decir, el tiempo durante el que este se está ejecutando o bien está esperando la finalización de otro método llamado desde él.

En el ejemplo de la Figura 6.7 se muestra un diagrama de secuencia en el que un objeto de la clase A envía un mensaje a un objeto de la clase B solicitando la ejecución del método *m1()*. A su vez, este objeto, necesita, para ejecutar el método *m1*, enviar dos mensajes a un objeto de la clase C, el *m2()* y el *m3()*, en este orden.

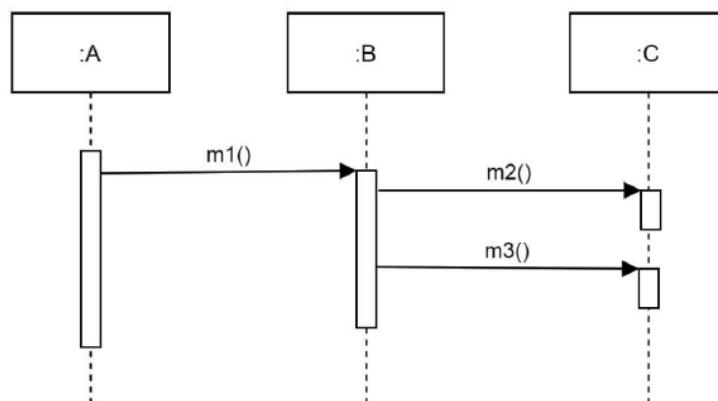


Figura 6.7. Diagrama de secuencia de ejemplo con tres objetos que se intercambian mensajes para realizar lo especificado en un supuesto escenario de un caso de uso.

Como paso previo a la elaboración de los diagramas de secuencia para cada caso de uso, es necesario identificar las clases de análisis. Por ejemplo, para la aplicación de la agenda de excompañeros de la Actividad resuelta 6.1, para la que se especificaron los casos de uso, se pueden identificar las siguientes clases:

- Una clase de interfaz, que se llama *Ventana*.
- Una clase de control, que se llama *Control*.
- Dos clases de entidad, llamadas *Excompañero* y *Cena*.

Con estas cuatro clases, es posible crear el diagrama de clases que se representa en la Figura 6.8, en el que se ha relacionado la clase de control con las clases de entidad *Excompañero* y *Cena*. Para cada cena se necesita conocer los excompañeros y excompañeras que han asistido y la persona que la ha organizado. No se han indicado métodos para las clases, los cuales se irán añadiendo a las clases a medida que se vayan descubriendo en los diagramas de secuencia. Cada mensaje enviado a una clase en un diagrama de secuencia será un método de la clase receptora en el diagrama de clases.

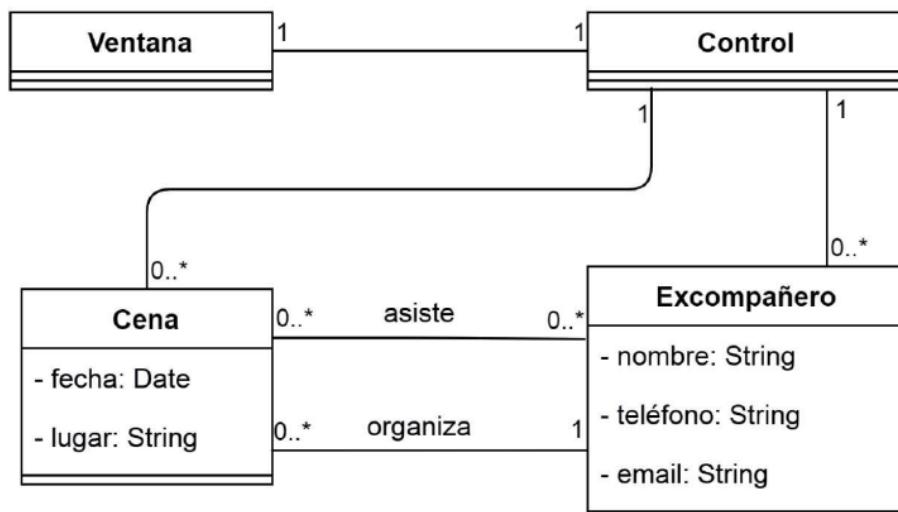


Figura 6.8. Diagrama de clases de análisis con las clases de interfaz, de control y de entidad identificadas en la aplicación para una agenda de antiguos compañeros y compañeras de estudios.

Actividad resuelta 6.2

Diagramas de secuencia

Se desea crear los diagramas de secuencia correspondientes a varios casos de uso de la agenda de excompañeros y excompañeras de la Actividad resuelta 6.1. Concretamente, se van a elaborar los diagramas de secuencia para los casos de uso *Agregar excompañero/a*, *Consultar excompañero/a*, *Agregar cena*, *Eliminar cena* y *Agregar asistente*.

Todo caso de uso es iniciado por un actor (persona usuaria) que solicita realizar la operación correspondiente. La ventana recoge los datos introducidos por la persona usuaria y solicita a la clase de control que realice la operación correspondiente.

Solución

El caso de uso *Agregar excompañero/a* se inicia en el momento que el usuario o usuaria solicita agregar un/a excompañero/a, la ventana obtiene los datos del excompañero/a (nombre, teléfono y correo electrónico) y pide a la clase de control que agregue una persona con esos datos. La clase de control comprueba si ya existe algún excompañero/a con ese nombre comparando el nombre del excompañero/a que se desea añadir con los nombres de las personas ya agregadas. En caso de que ya exista una persona con ese nombre, finaliza el caso de uso; en caso contrario, se crea un objeto de la clase *Excompañero* a partir de los datos introducidos por la persona usuaria. En la Figura 6.9 se puede observar el correspondiente diagrama de secuencia.

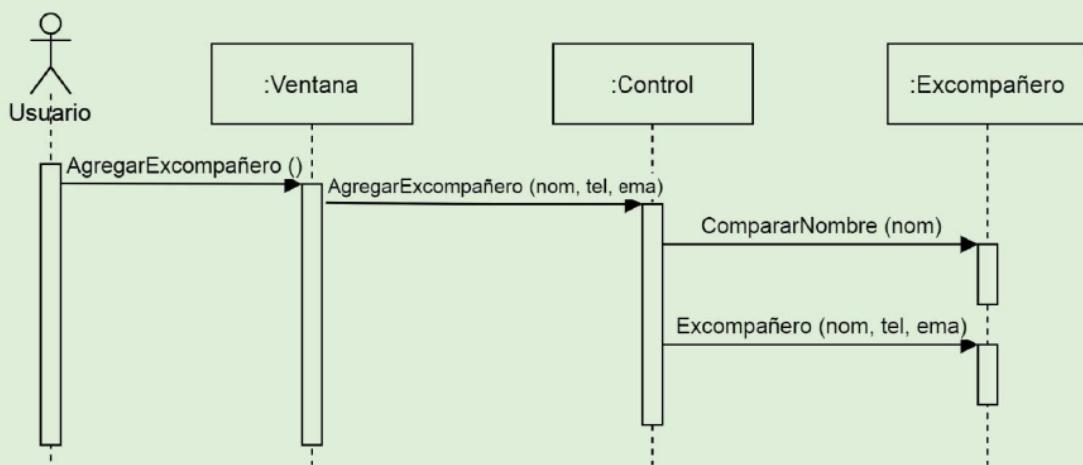


Figura 6.9. Diagrama de secuencia del caso de uso Agregar excompañero/a.

El caso de uso *Consultar excompañero/a* (véase Figura 6.10) comienza cuando la persona usuaria solicita consultar los datos de un/a excompañero/a, la ventana obtiene el nombre de este o esta, y pide a la clase de control que busque una persona con ese nombre. La clase de control comprueba si existe alguna persona con ese nombre comparando el nombre del excompañero/a que se desea consultar con los nombres de los excompañeros/as existentes. En caso de que no exista una persona con ese nombre, finaliza el caso de uso; y en caso contrario, se muestran los datos de ese antiguo compañero o compañera (teléfono y correo electrónico).

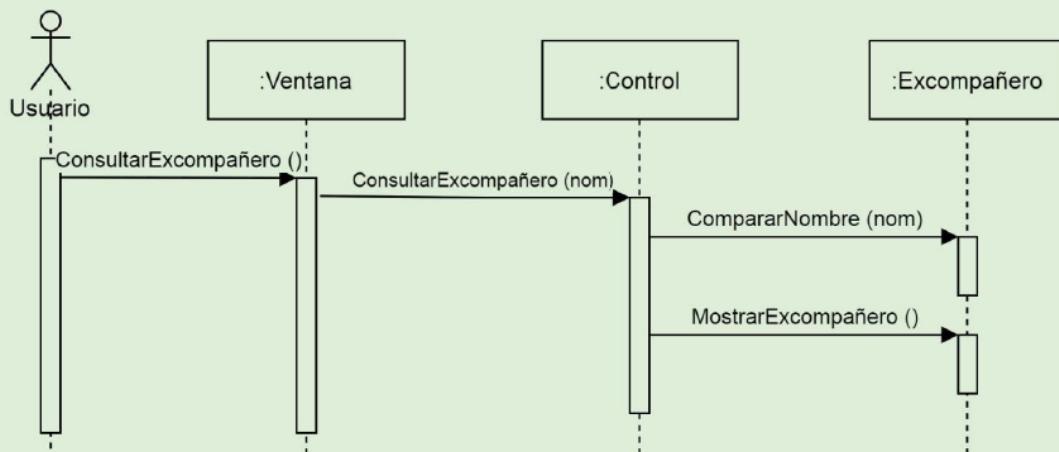


Figura 6.10. Diagrama de secuencia del caso de uso Consultar excompañero/a.

El caso de uso *Agregar cena* (Figura 6.11) se inicia cuando el usuario o usuaria solicita agregar una cena. La ventana obtiene entonces la fecha y el lugar de la cena y pide a la clase de control que agregue una cena con esos datos. Se comprueba si ya existe alguna cena para el año indicado, comparando el año de la cena que se desea añadir con los años de las cenas existentes. En caso de que se encuentre una cena para el año indicado, finaliza el caso de uso; en caso contrario, la ventana solicita la introducción del nombre de la persona que organiza la cena y se busca un excompañero o excompañera con el nombre de la persona organizadora, comparando este nombre con los de las personas ya agregadas. En caso de que no exista ningún excompañero o excompañera con ese nombre finaliza el caso de uso; en caso contrario, se crea un objeto de la clase *Cena* a partir de los datos introducidos por la persona usuaria.

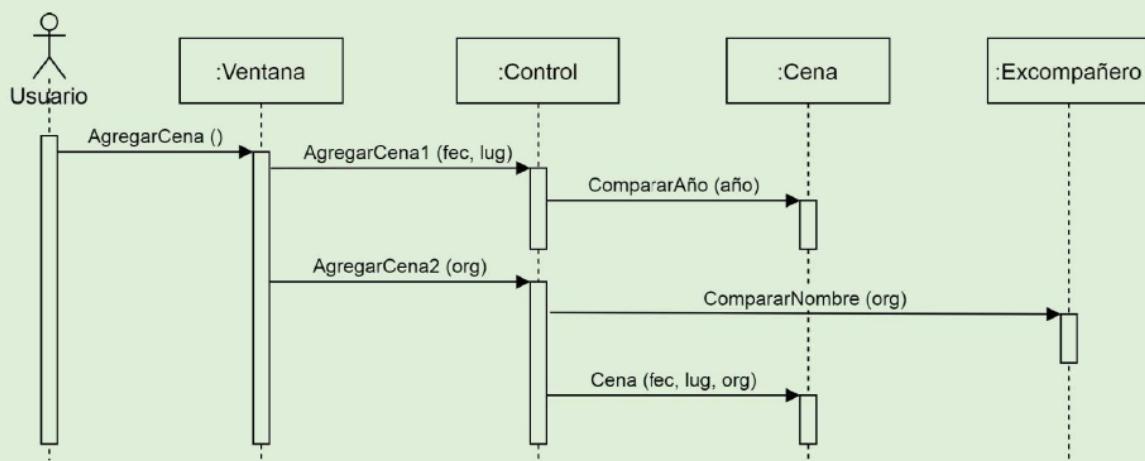


Figura 6.11. Diagrama de secuencia del caso de uso *Agregar cena*.

El caso de uso *Eliminar cena* (Figura 6.12) comienza solicitándose eliminar una cena. La ventana obtiene el año de la cena que se desea eliminar y pide a la clase de control que busque una cena para ese año. La clase de control comprueba si existe alguna cena para el año indicado comparando el año de la cena que se desea eliminar con los años de las cenas existentes. En caso de que no exista, finaliza el caso de uso; en caso de que sí exista, la clase de control solicita la eliminación de la cena enviando un mensaje a la clase *Cena*.

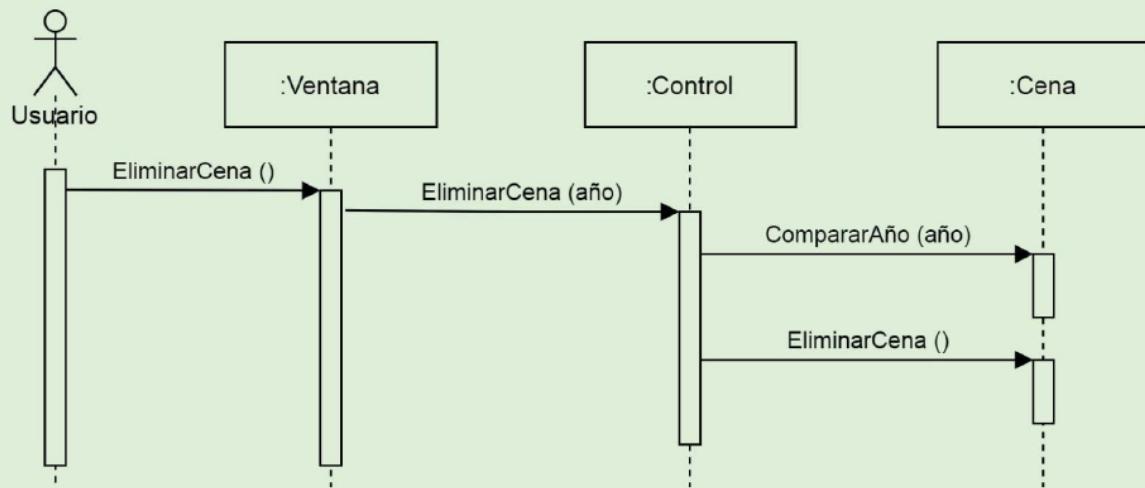


Figura 6.12. Diagrama de secuencia del caso de uso *Eliminar cena*.

El caso de uso Agregar asistente (Figura 6.13) se inicia en el momento en que la persona usuaria solicita agregar un asistente a una cena, la ventana obtiene el año de la cena y pide a la clase de control que consulte si hay una cena para ese año. La clase de control comprueba si esta existe, comparando el año de la cena que se desea consultar con los años de las cenas existentes. En caso de que no haya ninguna cena para ese año, finaliza el caso de uso; en el caso contrario, la ventana obtiene el nombre del asistente a la cena que se desea agregar y pide a la clase de control agregar dicho asistente. Esta clase de control solicita buscar dicho asistente entre las personas agregadas. Si no lo encuentra, finaliza el caso de uso; en caso contrario, lo busca entre los asistentes a la cena. En caso de que ya figure, finaliza el caso de uso; y, en caso contrario, se agrega dicho asistente a la cena.

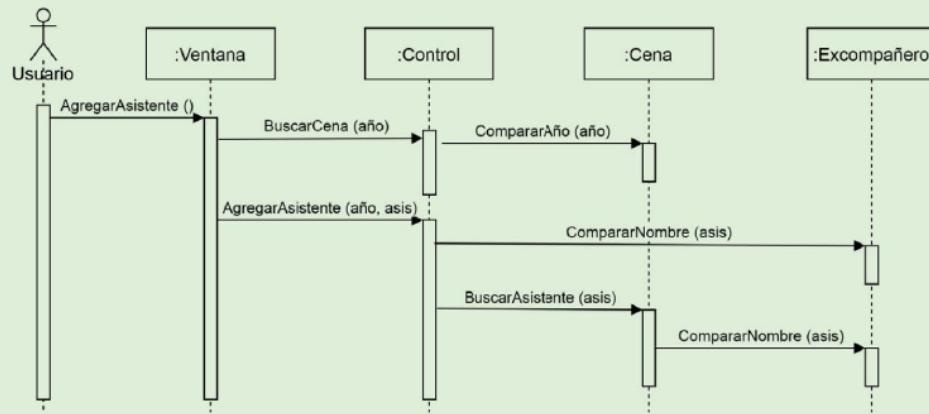


Figura 6.13. Diagrama de secuencia correspondiente al caso de uso Agregar asistente.

Una vez elaborados los diagramas de secuencia, se puede ir refinando o ampliando el diagrama de clases de la Figura 6.8, de manera que cada mensaje que recibe una clase se corresponde con un método de dicha clase. Por tanto, a partir de los diagramas de secuencia especificados en esta actividad, se generaría el diagrama de clases que se muestra en la Figura 6.14.

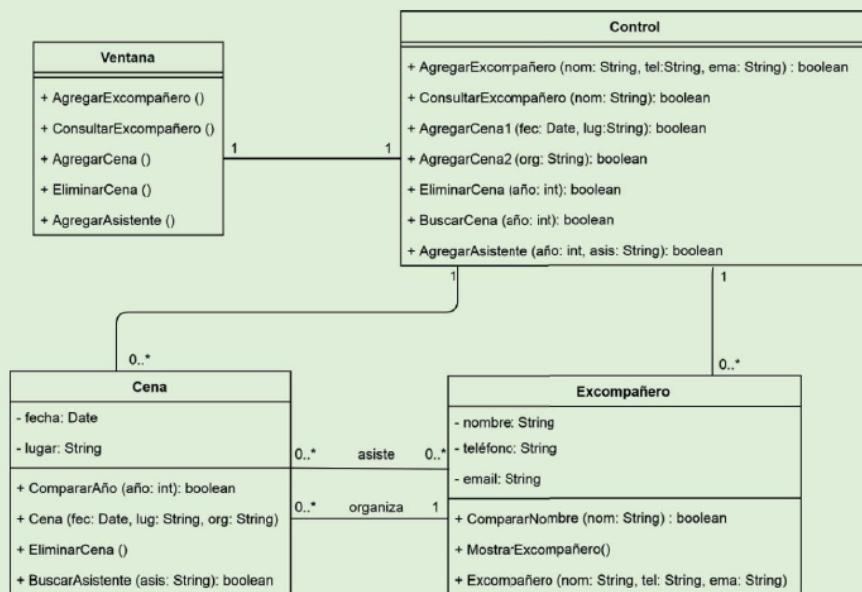


Figura 6.14. Diagrama de clases ampliado con los métodos encontrados en los diagramas de secuencia correspondientes a los casos de uso Agregar excompanionero/a, Consultar excompanionero/a, Agregar cena, Eliminar cena y Agregar asistente.

Actividad propuesta 6.2

Diagramas de secuencia

A partir de la Actividad resuelta 6.2, crea los diagramas de secuencia correspondientes a los casos de uso *Consultar cena*, *Eliminar asistente* y *Consultar asistentes*. Después, amplía el diagrama de clases de la Figura 6.14, incluyendo los métodos descubiertos en los diagramas de secuencia propuestos.

Actividad resuelta 6.3

Diagrama de casos de uso, diagramas de secuencia y diagrama de clases

Se desea crear una aplicación para la gestión de un videoclub a partir de la siguiente descripción:

1. Para cada película es preciso almacenar en el sistema su código, título, año de producción, género al que pertenece y nombre del país de producción. De la introducción de estos datos en el sistema se encarga el dependiente del videoclub.
2. Los clientes del videoclub alquilan películas usando una máquina dispensadora de películas. De cada cliente se guarda la siguiente información en el videoclub: NIF, nombre, dirección completa, número de teléfono, dirección de correo electrónico, número de socio y fecha de alta. El dependiente del videoclub también se encarga de dar de alta a los clientes en el sistema.
3. Cuando un cliente quiere alquilar una película, selecciona la película en la máquina dispensadora y el sistema comprueba si el cliente ha recibido una sanción porque, en tal caso, no se le permitirá alquilar la película. En caso contrario, se comprobará si hay existencias suficientes de esa película y, en caso afirmativo, se disminuye el stock de la película y se registra la fecha del alquiler.
4. Cuando un cliente realiza la devolución de una película, se aumenta el stock de la película. Será necesario registrar asimismo la fecha de devolución de la película, así como la sanción impuesta en caso de que la devolución se efectúe con posterioridad a la fecha convenida.

Solución

El primer paso es crear el diagrama de casos de uso, que se muestra en la Figura 6.15.

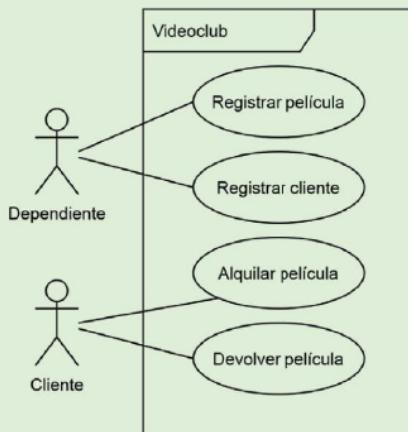


Figura 6.15. Diagrama de casos de uso para un videoclub, que incluye los dos actores que intervienen en la aplicación (cliente y dependiente) y cuatro casos de uso (registrar película, registrar cliente, alquilar película y devolver película).

A continuación, se elaboran los diagramas de secuencia para los casos de uso *Alquilar película* y *Devolver película*, pero antes se tienen que identificar las clases de análisis, que son las siguientes:

- Una clase de interfaz, que se puede llamar *Ventana*.
- Una clase de control, que se puede llamar *Videoclub*.
- Tres clases de entidad, llamadas *Película*, *Cliente* y *Alquiler*.

El caso de uso *Alquilar película* (véase la Figura 6.16) se inicia cuando el cliente solicita alquilar una película, la ventana obtiene la identificación del cliente (su NIF) y la de la película que desea alquilar (su código) y le pide a la clase *Videoclub* que proceda al alquiler de dicha película. Esta comprueba si el cliente tiene impuesta alguna sanción, buscando el cliente por su NIF. Si ha recibido alguna sanción, se muestra un mensaje de error y finaliza el caso de uso; en caso contrario, se busca la película que se desea alquilar y una vez encontrada, se comprueba si hay ejemplares suficientes de esa película. En caso de que no lo haya, se muestra un mensaje de error y finaliza el caso de uso; en caso contrario, se disminuye el stock de dicha película y se procede a registrar el alquiler.

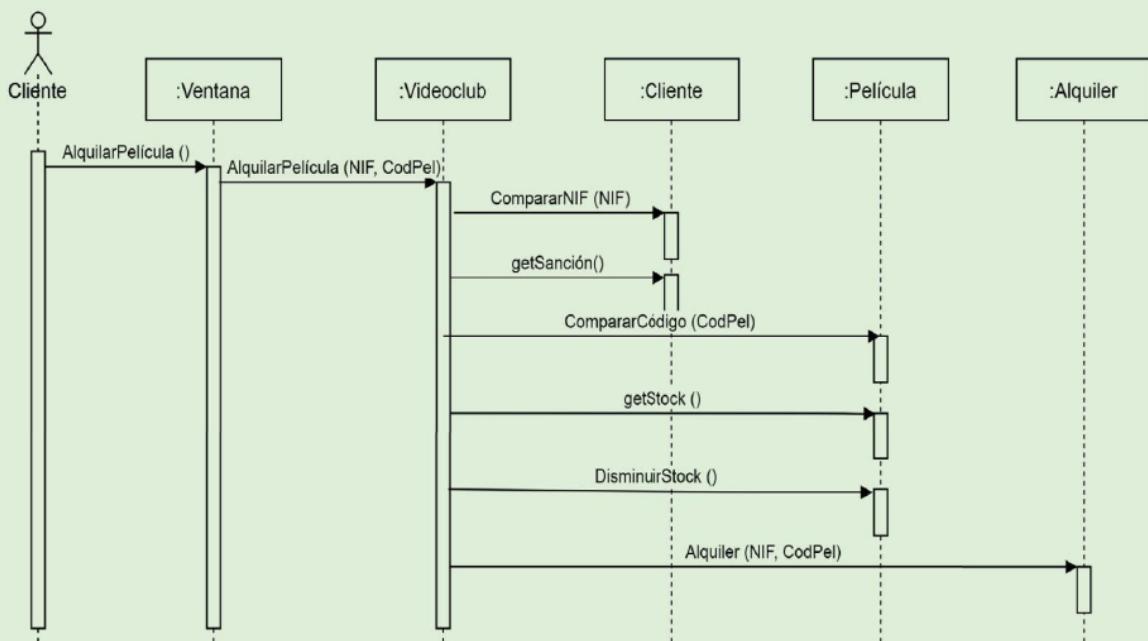


Figura 6.16. Diagrama de secuencia correspondiente al caso de uso *Alquilar película*.

El caso de uso *Devolver película* (Figura 6.17) comienza cuando el cliente solicita devolver una película. Para ello, la ventana obtiene la identificación del cliente (su NIF) y la de la película que desea devolver (su código) y le pide a la clase *Videoclub* que proceda a la devolución de dicha película. Al efecto de registrar la devolución, se busca la película por su código y se aumenta su stock. Posteriormente, se registra la devolución apuntando la fecha de esta y se comprueba si hay que imponer una sanción al cliente por devolución fuera de plazo. Si es así, se registra dicha sanción para el cliente correspondiente.

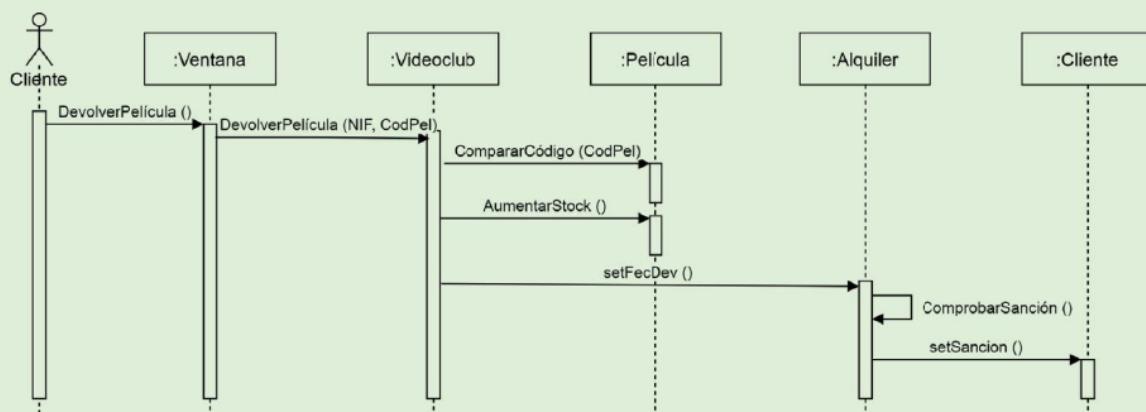


Figura 6.17. Diagrama de secuencia del caso de uso Devolver película.

A partir de las clases de análisis indicadas anteriormente y los métodos detectados en los dos diagramas de secuencia anteriores, se deduce el diagrama de clases que se observa en la Figura 6.18.

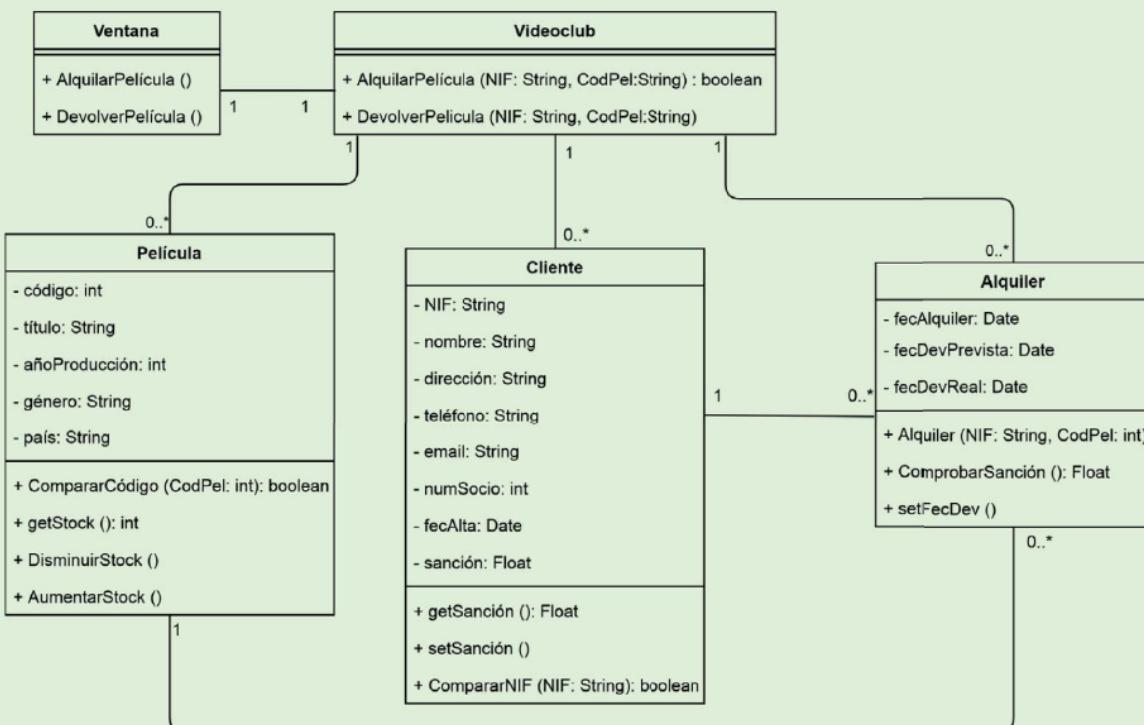


Figura 6.18. Diagrama de clases para un videoclub.

6.3.2. Diagramas de colaboración

Los diagramas de colaboración son otro tipo de diagramas de interacción. Estos diagramas muestran la misma información que los diagramas de secuencia, pero en los diagramas de colaboración no se concede una especial relevancia al orden (la secuencia) en que se envían los mensajes, sino que se da mayor importancia a visualizar las clases que colaboran y cómo estas se relacionan mediante el envío de mensajes.



Recuerda

Lo relevante en un **diagrama de colaboración** no es el orden en el que se envían los mensajes para ejecutar lo especificado en un escenario de un caso de uso, sino la visualización de los objetos que colaboran en una aplicación o en parte de ella por medio del envío de mensajes. De hecho, los mensajes y colaboraciones que se muestran en un diagrama de colaboración no tienen por qué estar restringidos a un determinado caso de uso, sino que pueden mostrarse las colaboraciones necesarias para la ejecución de varios o incluso todos los casos de uso de la aplicación.

En un diagrama de colaboración intervienen los siguientes elementos:

- **Objetos, clases o actores:** se representan de igual modo que en los diagramas de secuencia.
- **Enlaces:** las clases que interactúan entre sí mediante el envío de algún mensaje se unen mediante una línea que representa un enlace entre las clases.
- **Mensajes:** se representan al lado del enlace entre las dos clases que se intercambian el mensaje, que se representa mediante una flecha, al lado de la cual se coloca un número y el mensaje en cuestión. El número hace referencia al orden del mensaje (número de secuencia) dentro del diagrama de colaboración.

Así, el diagrama de colaboración que corresponde al diagrama de secuencia de la Figura 6.7 es el que se muestra en la Figura 6.19. En este diagrama de colaboración, delante de cada mensaje se ha colocado un número que hace referencia al orden en el que se envía cada mensaje.

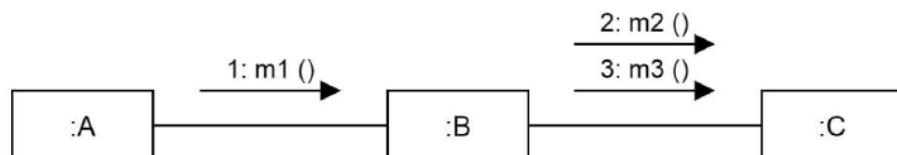


Figura 6.19. Diagrama de colaboración de ejemplo correspondiente al diagrama de secuencia de la Figura 6.7.

En las Figuras 6.20 y 6.21, se muestran, respectivamente, los diagramas de colaboración que corresponden a los diagramas de secuencia del caso de uso Agregar excompañero/a (Figura 6.9) y del caso de uso Agregar cena (Figura 6.11).

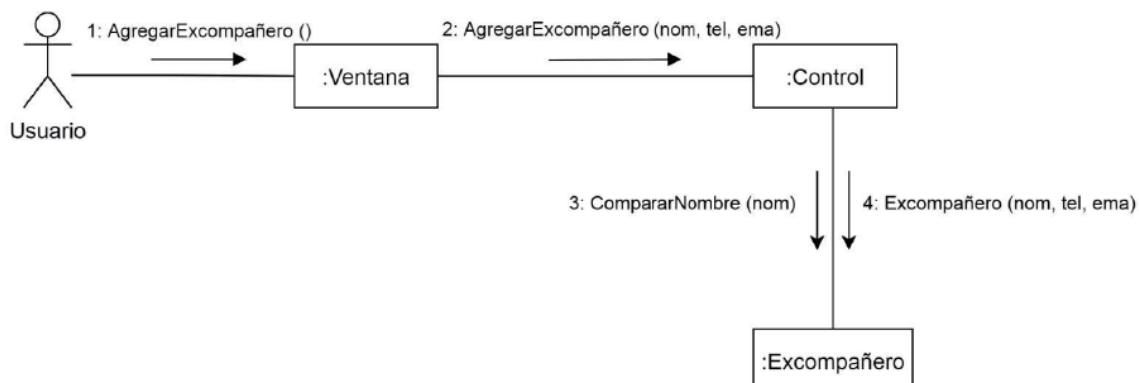


Figura 6.20. Diagrama de colaboración de ejemplo correspondiente al diagrama de secuencia de la Figura 6.9.

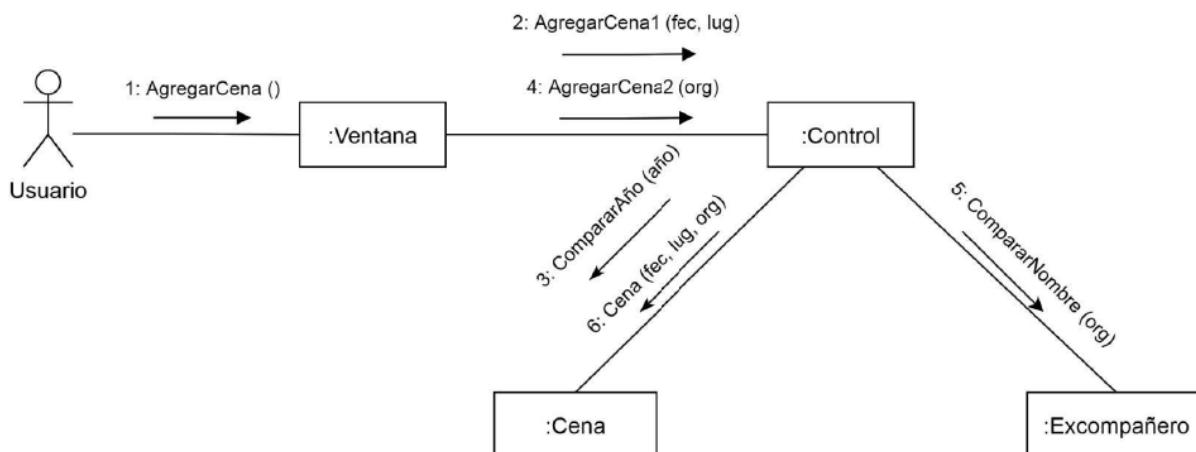


Figura 6.21. Diagrama de colaboración de ejemplo correspondiente al diagrama de secuencia de la Figura 6.11.

6.3.3. Herramientas para la elaboración de diagramas de interacción

En este apartado, se verá cómo se pueden emplear las herramientas diagrams.net y el módulo Papyrus SysML de Eclipse para la creación de diagramas de interacción.

Elaboración de diagramas de secuencia con diagrams.net

Como ya se señaló antes, a la herramienta *diagrams.net* se accede por medio de la página web <https://app.diagrams.net/>.

Una vez cargada la aplicación y tras indicar que se desea crear un nuevo diagrama y la ubicación donde se desea almacenar este, aparecerá una pantalla como la de la Figura 5.34 de la Unidad 5.

En el ejemplo que se va a describir aquí, en primer lugar, se va a crear el diagrama de secuencia para la agenda de antiguos compañeros y compañeras correspondiente al caso de uso *Agregar excompañero/a* de la Figura 6.9.

Para crear un diagrama de secuencia, hay que hacer clic sobre el ícono correspondiente a cada elemento en la paleta UML. Dichos elementos se recogen en la Tabla 6.4.

Tabla 6.4. Elementos que forman parte de los diagramas de secuencia, su ícono correspondiente en diagrams.net y la paleta que le corresponde

Elemento	Ícono	Paleta
Línea de vida para un actor		UML

Elemento	Icono	Paleta
Línea de vida para un objeto	A simple rectangular box labeled ':Object'.	UML
Mensaje enviado de un objeto a otro	A UML message arrow pointing from one lifeline to another, labeled 'dispatch' at the source end and 'return' at the target end.	UML
Mensaje enviado de un objeto a sí mismo	A UML self-call arrow pointing back to the same lifeline, labeled 'self call'.	UML

Para comenzar a elaborar el diagrama, habrá que colocar las líneas de vida correspondientes al actor *Usuario* y a las clases *Ventana*, *Control* y *Excompañero*. Se clica sobre el icono correspondiente de la paleta *UML* y se le asigna un nombre al actor o a la clase. Posteriormente, habrá que representar los mensajes.

Para cada uno de ellos, se debe pinchar sobre el icono que le corresponde (véase Tabla 6.4) y se colocará en el lugar correspondiente del diagrama. La flecha discontinua de vuelta de un mensaje que se envía de un objeto a otro se puede eliminar y se debe sustituir el texto *dispatch*, haciendo doble clic sobre él, por el nombre del método correspondiente con sus parámetros, si es el caso.

Se puede alargar o acortar el rectángulo vertical que representa el tiempo durante el cual un método está activo, seleccionando dicho rectángulo y haciendo clic con el botón izquierdo del ratón sobre su extremo superior o inferior. De igual modo, también es posible acortar o alargar la línea discontinua que representa la línea de vida de un objeto o de un actor.

Siguiendo estas indicaciones, resulta muy sencillo elaborar un diagrama de secuencia como el de la Figura 6.9.

Elaboración de diagramas de secuencia con Papyrus

En el siguiente ejemplo, con el fin de crear de manera adecuada los diagramas de secuencia de la agenda de excompañeros y excompañeras, en primer lugar, se crea el diagrama de casos de uso correspondiente a esta aplicación, que es el que se mostró

en la Figura 6.3, si bien solo se van a incluir los casos de uso *Agregar excompañero/a*, *Consultar excompañero/a*, *Agregar cena*, *Eliminar cena* y *Agregar asistente*. Después, se creará también el diagrama de clases de la Figura 6.14.

Para ello, se va a crear un nuevo proyecto con el módulo Papyrus SysML de Eclipse. Hay que indicar que en este caso se desea crear un diagrama de casos de uso, un diagrama de clases y un diagrama secuencia. Por consiguiente, se deberán activar en la pantalla como la de la Figura 5.42 de la Unidad 5, las casillas de verificación correspondientes a *Use Case Diagram*, *Class Diagram* y *Sequence Diagram*.

En primer lugar, se generará el diagrama de casos de uso de la Figura 6.22 siguiendo las instrucciones recogidas en el Apartado 6.2.1. Se muestra el diagrama resultante en dicha figura.

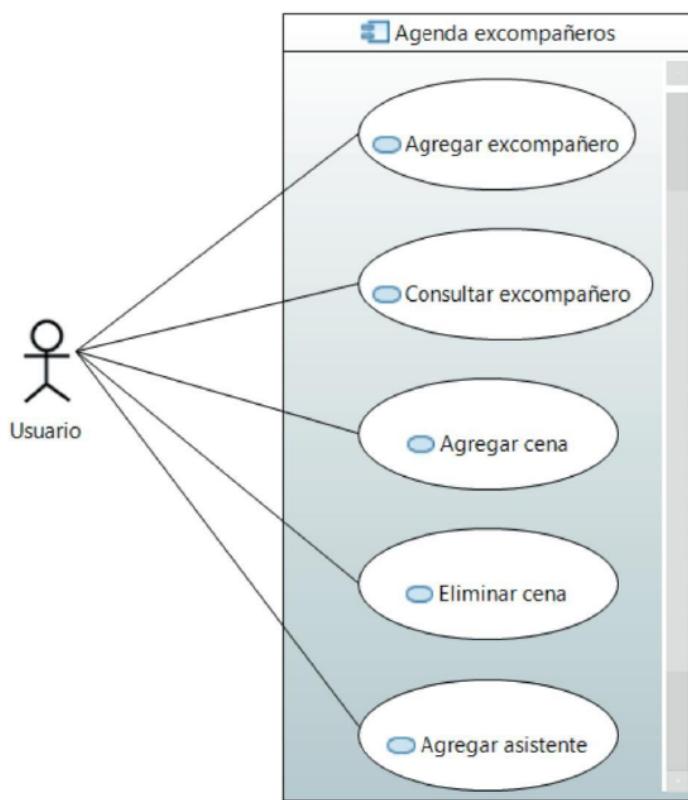


Figura 6.22. Diagrama de casos de uso para la aplicación de una agenda de antiguos compañeros y compañeras que incluye los casos de uso Agregar excompañero/a, Consultar excompañero/a, Agregar cena, Eliminar cena y Agregar asistente.

El siguiente paso es crear el diagrama de clases de la Figura 6.14. Para ello, se puede consultar el Apartado 5.6.2, en el que se indicó cómo crear diagramas de clases con el módulo Papyrus SysML de Eclipse, si bien no se añadieron métodos a las clases. Dicha tarea es lo que se debe hacer ahora.

Para añadir métodos a una clase incluida en un diagrama, se debe pulsar en el botón de la sección *Owned operation* del área de propiedades (véase la Figura 5.44). Es necesario indicar por cada método, su nombre, tipo de valor que devuelve, visibilidad y parámetros. En el caso de la clase *Ventana*, para cada método solo es necesario indicar su nombre en

Name. Una vez creados los métodos, para que se puedan visualizar en el diagrama, es necesario, como en el caso de los atributos, arrastrarlos desde el explorador del modelo.

Si el método requiere parámetros, es necesario, por cada uno de ellos, clicar en el botón de la sección Owned parameter. Así, para el método AgregarExcompañero de la clase Control, se deben añadir tres parámetros. Para el primer parámetro, llamado nom, habrá que indicar su nombre en Name, que es un parámetro de entrada (valor in) en Direction y que su tipo es String, como se puede observar en la Figura 6.23.

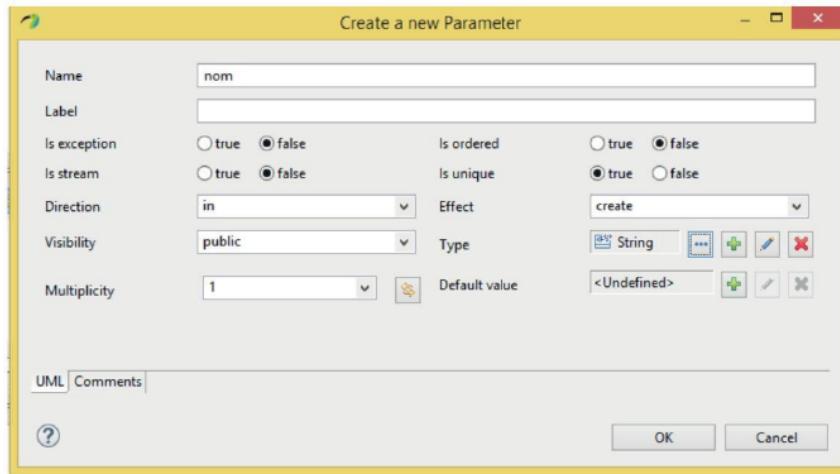


Figura 6.23. Propiedades del parámetro nom del método AgregarExcompañero de la clase Control. El nombre, dirección y tipo de dato son las más importantes.

Para el método AgregarExcompañero, se deben añadir de igual modo los otros dos parámetros: tel y ema. Como esta función devuelve un valor booleano que indica si el excompañero o la excompañera se ha podido agregar a la agenda, se debe incluir un «cuarto parámetro» que determine el tipo del valor de retorno. Por este motivo, no se asignará ningún nombre al parámetro en la casilla Name, pero se indicará en Direction el valor return y se elegirá el tipo de dato Boolean (véase la Figura 6.24).

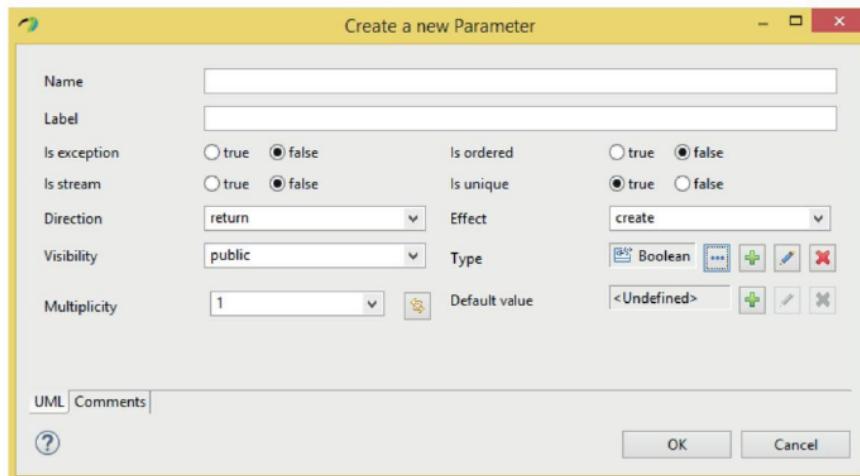


Figura 6.24. Ventana en la que se muestran las propiedades para el valor de retorno del método AgregarExcompañero de la clase Control. No es necesario asignarle nombre, pero se debe indicar en Direction el valor return y se debe indicar su tipo de dato, Boolean.

Una vez incluidos los tres parámetros para el método *AgregarExcompañero*, e indicado el tipo de dato del valor de retorno, las propiedades de este método son las que se muestran en la Figura 6.25.

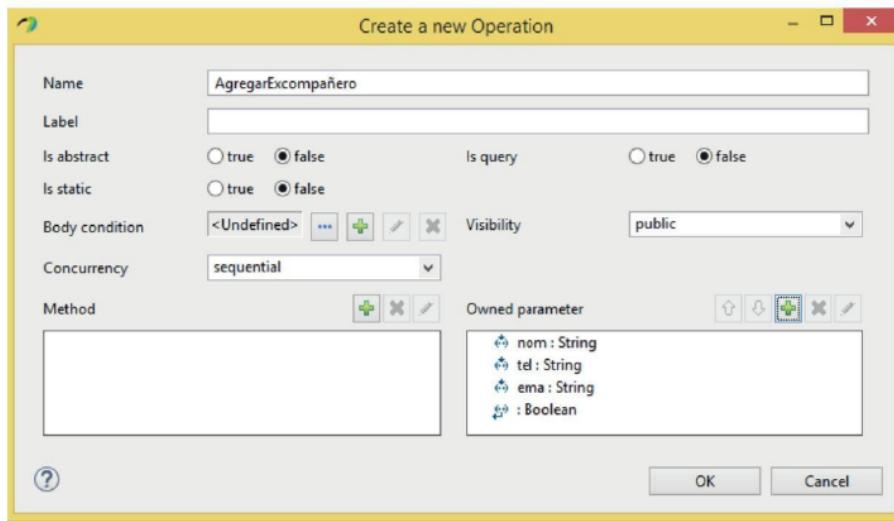


Figura 6.25. Propiedades para el método AgregarExcompañero de la clase Control: su nombre, visibilidad, sus tres parámetros y el tipo del valor de retorno.

Se deberá obrar de igual modo para los demás métodos de esta clase y para el resto de clases del diagrama, obteniéndose al final el diagrama de clases de la Figura 6.26.

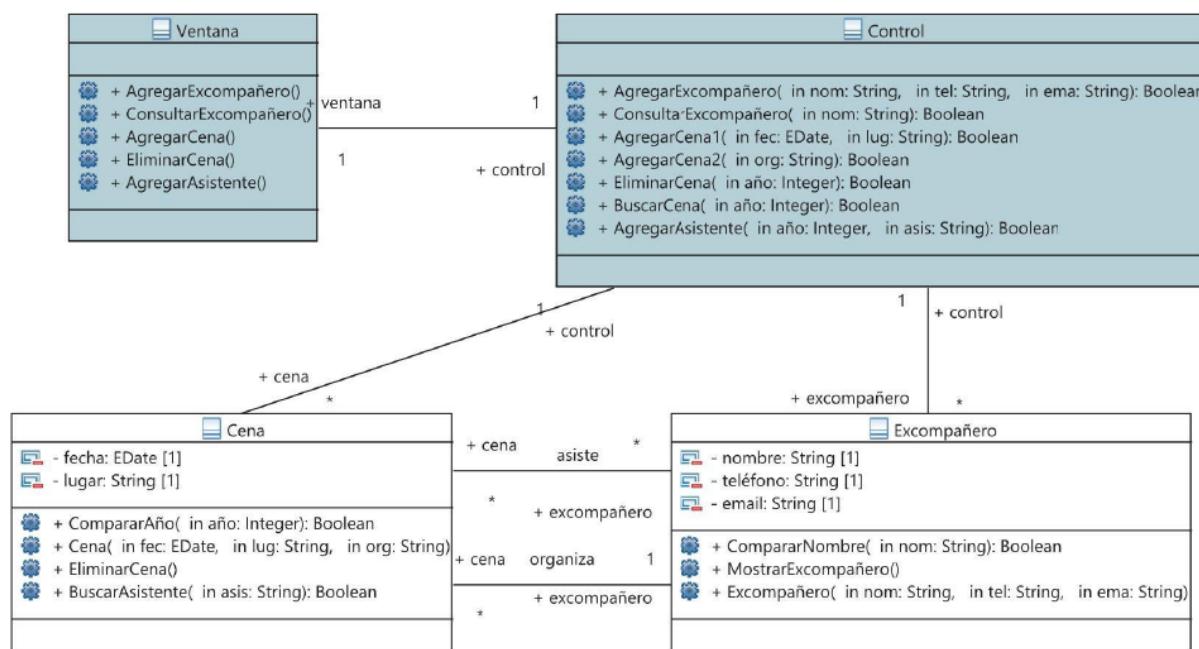


Figura 6.26. Diagrama de clases creado con Papyrus SysML para los casos de uso Agregar excompañero/a, Consultar excompañero/a, Agregar cena, Eliminar cena y Agregar asistente de la agenda de antiguos compañeros y compañeras.

Seguidamente, se explicará cómo crear diagramas de secuencia con Papyrus, tomando como ejemplo el diagrama de secuencia para el caso de uso *Agregar cena* de la Figura 6.11.

Con este fin, en primer lugar, hay que colocarse en la pestaña de Eclipse correspondiente al diagrama de secuencia. Una vez situados en esa pestaña, lo primero que hay que hacer es colocar las líneas de vida correspondientes al actor *Usuario* y a las clases *Ventana*, *Control*, *Cena* y *Excompañero*. Para ello, se arrastra el elemento correspondiente (el actor y las clases) desde el explorador del modelo al área de edición. Se obtiene un resultado como el que se observa en la Figura 6.27.

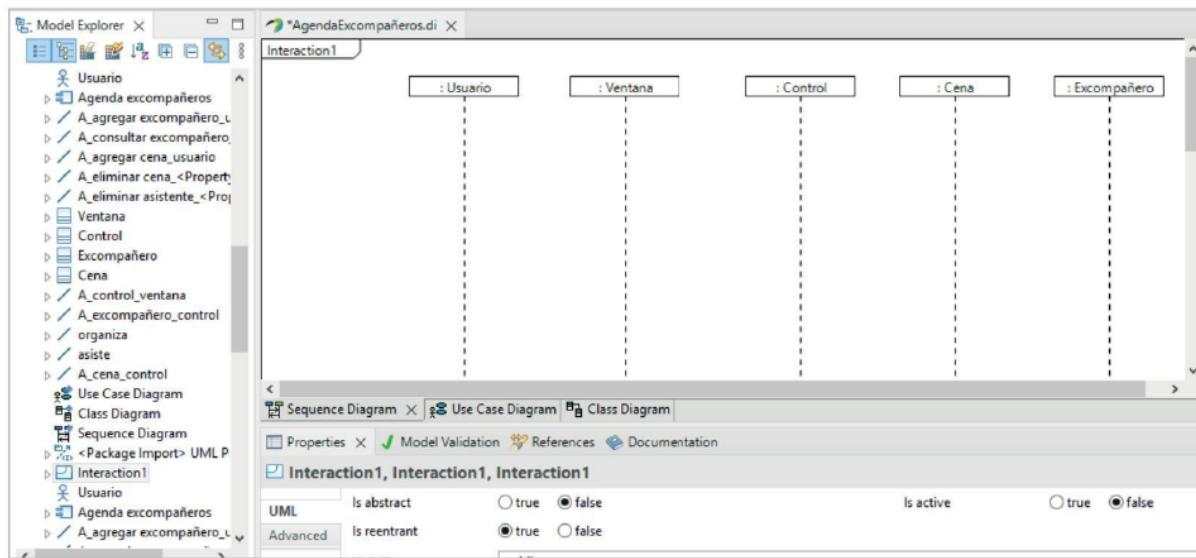


Figura 6.27. Líneas de vida necesarias para el diagrama de secuencia correspondiente al caso de uso Agregar cena. Para colocar las líneas de vida, se debe arrastrar el elemento correspondiente de cada línea de vida (actor o clase) desde el explorador del modelo al área de edición del diagrama.

Para añadir los mensajes, se hace clic sobre el elemento **Message Sync** de la sección **Edges** de la paleta (Figura 6.28) situada a la derecha del área de edición.

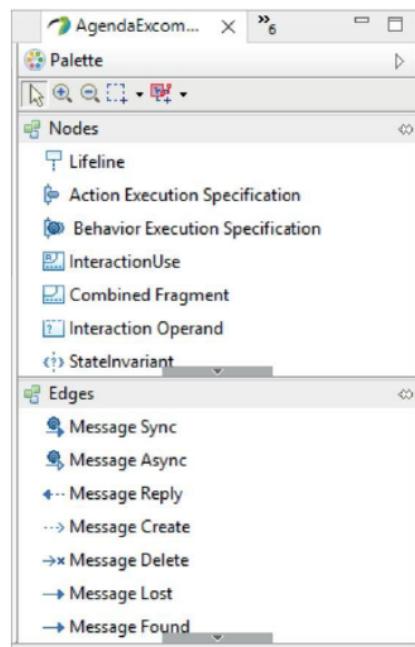


Figura 6.28. Paleta de Papyrus para la elaboración de diagramas de secuencia.

Luego, se debe arrastrar el ratón desde la línea de vida del emisor a la del receptor, y se asigna un nombre por defecto al mensaje enviado y al mensaje de retorno (Figura 6.29).

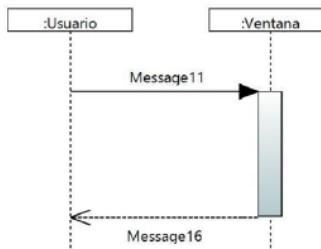


Figura 6.29. Por cada mensaje que se envía en un diagrama de secuencia, Papyrus SysML crea dos mensajes: el mensaje enviado, que en este caso se llama Message11, y el mensaje de retorno, llamado Message16.

Al seleccionar el mensaje de envío *Message11*, es posible conocer sus propiedades en la sección *Properties*, tal y como se muestra en la Figura 6.30.

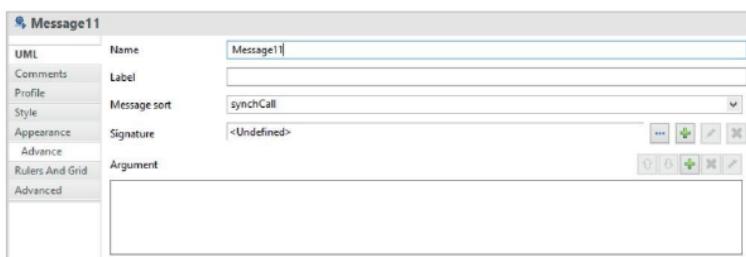


Figura 6.30. En la sección Properties, tras seleccionar un mensaje, se pueden ver y modificar sus propiedades (su nombre, etiqueta, el método al que se llama, etc.).

En el campo *Signature* (Figura 6.30), se debe indicar el método del objeto receptor que ha de ejecutarse en respuesta al mensaje. Para ello, se hace clic en el botón que presenta los tres puntos y se elige el método *AgregarCena()* de la clase *Ventana*, como se muestra en la Figura 6.31. Al realizar esta operación, el nombre del mensaje enviado es sustituido por el método seleccionado.

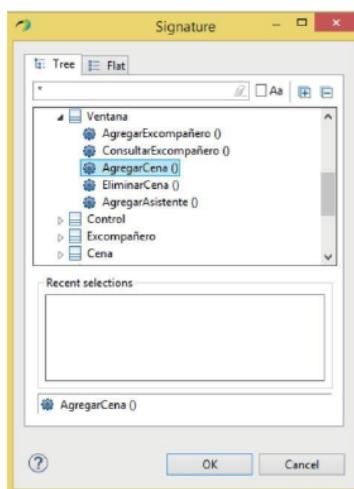


Figura 6.31. Por cada mensaje que se envía en un diagrama de secuencia en Papyrus SysML, se debe seleccionar el método de la clase receptora que se debe ejecutar en respuesta a ese mensaje, asignando un valor al campo *Signature* del mensaje enviado.

Aunque los mensajes de retorno no sean necesarios en este caso, no se pueden eliminar del modelo porque entonces Papyrus SysML borra el rectángulo vertical que representa el tiempo durante el cual el método está activo. Lo que se puede hacer es seleccionar el mensaje de retorno y dejar en blanco el nombre del mensaje (campo Name de las propiedades del mensaje). Tras hacer esto, para el primer mensaje del diagrama de secuencia, el resultado es el que se muestra en la Figura 6.32.

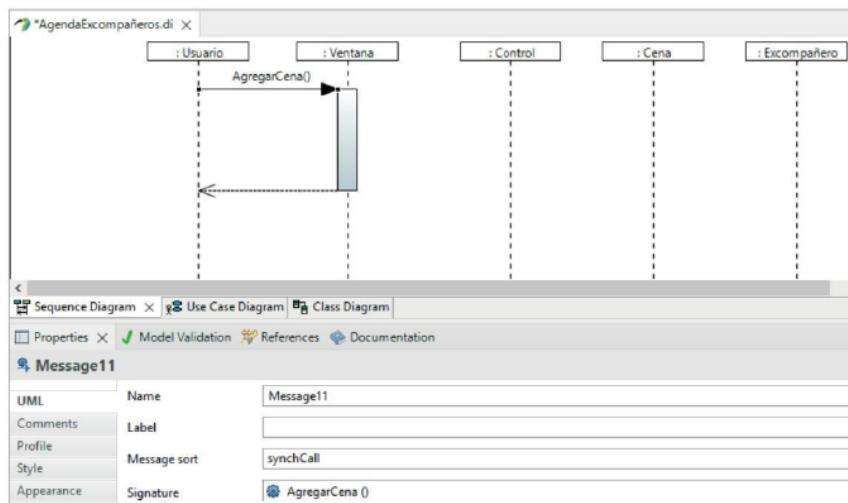


Figura 6.32. Diagrama de secuencia correspondiente al caso de uso Agregar cena, en el que se muestra el primer mensaje, que es enviado desde el actor Usuario a la clase Ventana y solicita la ejecución del método AgregarCena().

Se debe realizar la misma operación para crear el resto del diagrama de secuencia. Conviene que se alarguen o acorten los rectángulos verticales que representan el tiempo de activación de cada mensaje antes de añadir posteriores mensajes. Para ello, basta con arrastrar el extremo inferior de cada rectángulo. Al final se obtiene un diagrama de secuencia como el que se muestra en la Figura 6.33.

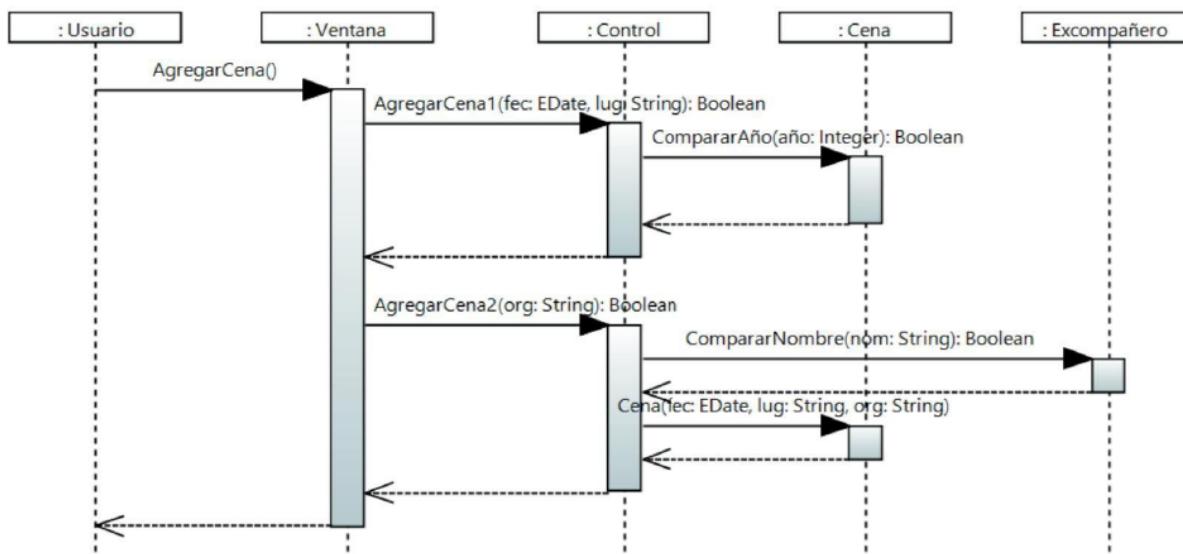


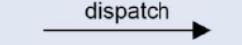
Figura 6.33. Diagrama de secuencia para el caso de uso Agregar cena de la agenda de excompañeros y excompañeras realizado con el módulo Papyrus SysML de Eclipse.

Elaboración de diagramas de colaboración con diagrams.net

En este apartado, se aprenderá cómo se puede crear un diagrama de colaboración con diagrams.net. En este caso, se realizará el correspondiente al caso de uso Agregar ex-compañero/a de la agenda de antiguos compañeros y compañeras (Figura 6.20).

Para crear un diagrama de colaboración, se pincha sobre el ícono correspondiente a cada elemento en la paleta *UML*, de acuerdo con la Tabla 6.5.

Tabla 6.5. Diferentes elementos que forman parte de los diagramas de colaboración, su símbolo correspondiente en diagrams.net y la paleta que le corresponde

Elemento	Símbolo	Paleta
Actor	 Actor	UML
Clase u objeto	 Object	UML
Enlace	 Association / Connector / Instance Specification / Property / Connector End	UML
Mensaje	 Message	UML

Se comenzará a elaborar el diagrama colocando en el área de edición el actor *Usuario* y las clases *Ventana*, *Control* y *Excompañero*. Una vez hecho esto, se unirán mediante un enlace el actor *Usuario* y la clase *Ventana*.

Luego, se añade el mensaje *AgregarExcompañero()* enviado desde *Usuario* a *Ventana*. Tras colocar la flecha en el área de edición, se sustituye el texto *dispatch* por *AgregarExcompañero()*.

Siguiendo estas indicaciones, resulta muy sencillo elaborar el diagrama de colaboración de la Figura 6.20.

Elaboración de diagramas de colaboración con Papyrus

En este ejercicio, se va a añadir al proyecto en el que antes se ha creado el diagrama de secuencia para el caso de uso *Agregar cena* (véase Figura 6.11) el correspondiente diagrama de colaboración, que se mostró en la Figura 6.21. A tal efecto, una vez abierto el proyecto en Eclipse, seleccionándolo desde el explorador del modelo, se activa en el menú contextual la opción *New Diagram* → *Communication Diagram* y se le da un nombre al diagrama de colaboración.

Para comenzar, será necesario colocar en el área de edición los rectángulos que representan los objetos o clases. En Papyrus, los actores se representan también con rectángulos, como los objetos y las clases. Para ello, se arrastran desde el explorador del modelo los elementos del modelo correspondientes: el actor *Usuario* y las clases *Ventana*, *Control*, *Cena* y *Excompañero*. Una vez hecho esto, la única manera de establecer enlaces entre las clases es mediante el envío de mensajes entre ellas.

Se empieza haciendo clic sobre el elemento *Message* del área *Edges* de la paleta (véase Figura 6.34).

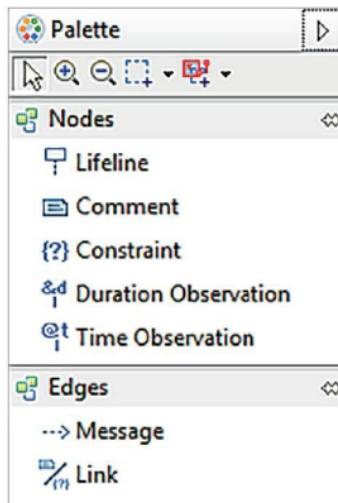


Figura 6.34. Paleta de Papyrus para la elaboración de diagramas de colaboración.

A continuación, se clica sobre la clase que envía el mensaje, en primer lugar, y luego sobre la clase receptora. En este caso, se comenzará con el mensaje *AgregarCena()* enviado desde el actor *Usuario* a la clase *Ventana*. Aparecerá entonces un mensaje con un nombre por defecto que habrá que modificar. En las propiedades del mensaje, en el campo *Signature*, se seleccionará el mensaje de la clase receptora (*Ventana*) correspondiente: en este caso, *AgregarCena()*. A diferencia de lo que ocurría al hacer esto en los diagramas de secuencia, no se modifica el texto del mensaje, por lo que habrá que escribirlo en la propiedad *Name*, como se muestra en la Figura 6.35. Debe recordarse que, para los diagramas de colaboración, se debe poner antes del nombre del mensaje un número, que indica el orden en el que se envía el mensaje en el contexto de este diagrama. Las propiedades para el primer mensaje del diagrama se mostrarán como se observa en la Figura 6.35.

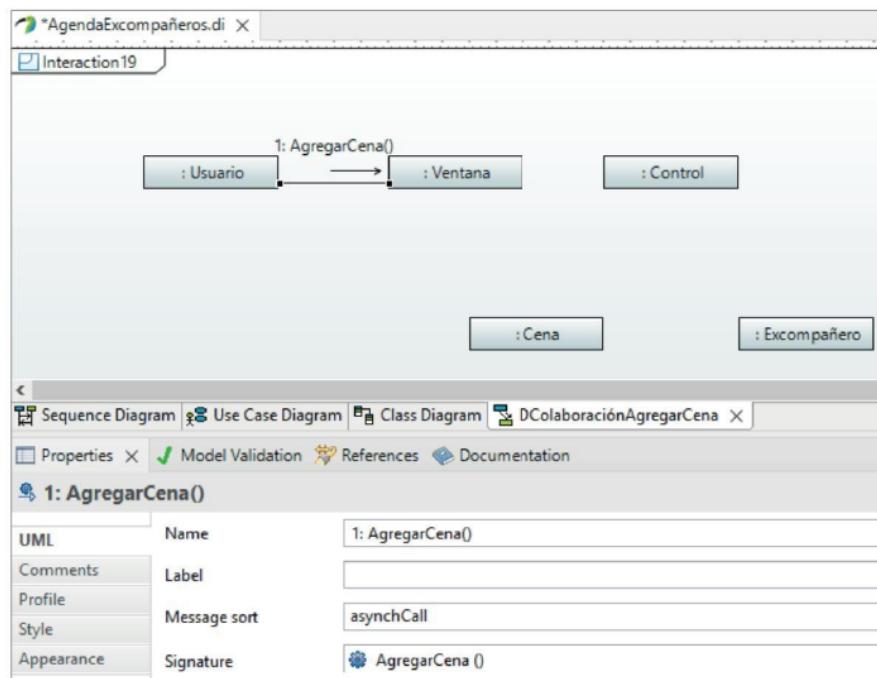


Figura 6.35. Propiedades del mensaje AgregarCena(), que es el primero del diagrama de colaboración correspondiente al caso de uso Agregar cena de la agenda de excompañeros y excompañeras.

Después, se establecerán los restantes mensajes entre las clases del diagrama de colaboración de igual modo y se conseguirá al final un diagrama de colaboración como el que se muestra en la Figura 6.36.

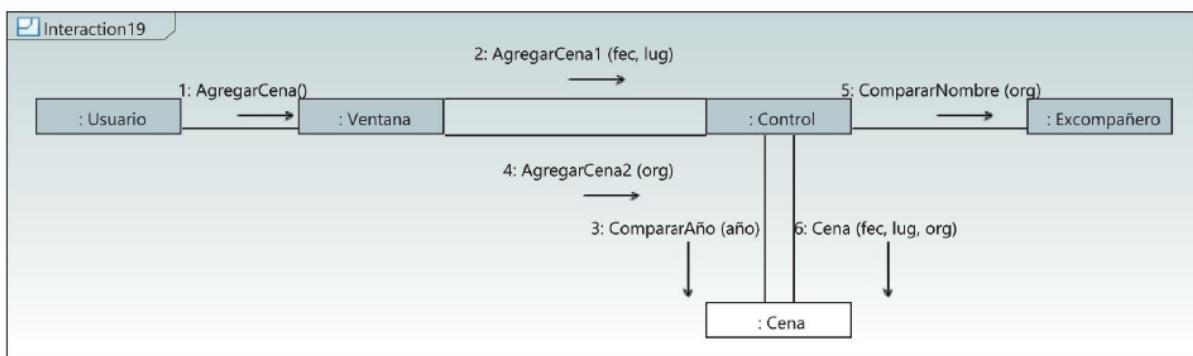


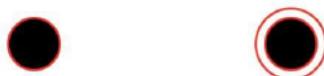
Figura 6.36. Diagrama de colaboración para el caso de uso Agregar cena de la agenda de antiguos compañeros y compañeras realizado con el módulo Papyrus SysML de Eclipse.

6.4. Diagramas de estados

Los diagramas de estados se usan para describir el comportamiento en el tiempo de un objeto particular o de una interacción. Concretamente, muestran los estados (situaciones) por los que puede pasar un objeto y las transiciones o cambios de estado que experimenta como consecuencia de los eventos que van sucediendo. Este tipo de diagramas se usan normalmente para mostrar el comportamiento de objetos relevantes.

Seguidamente, se detallan los elementos de los que consta un diagrama de estados y la manera de representarlos:

- **Estado:** un estado es cada una de las situaciones por las que puede pasar un objeto, tiempo durante el cual realiza cierta actividad o espera la ocurrencia de algún evento. Un estado se representa mediante un rectángulo con esquinas redondeadas. No obstante, hay dos estados especiales durante los cuales el objeto no realiza ninguna actividad; estos se simbolizan de manera distinta (véase la Figura 6.37).
 1. El **estado inicial**, que se representa mediante un círculo relleno de color negro.
 2. El **estado final** o último estado por el que pasa un objeto, que se representa como se muestra en la Figura 6.37. No es necesario que en todos los casos haya un estado final, y también hay que tener en cuenta que podría haber más de uno.



Estado inicial Estado final

Figura 6.37. Representación de los estados inicial y final en un diagrama de estados.

- **Evento:** es algo que acontece durante la vida de un objeto y que desencadena el paso de este a otro estado, es decir, los cambios de estado de los objetos vienen determinados por la ocurrencia de determinados eventos. Un evento puede ser el cumplimiento de una determinada condición, la recepción de un mensaje, el transcurso de cierto periodo de tiempo, etcétera.
 - **Transición:** es el paso de un estado a otro como consecuencia de la ocurrencia de un evento. Se representa como una flecha que parte del estado origen y llega al estado destino y sobre esta flecha se debe indicar el nombre del evento que desencadena el cambio de estado.

A modo de ejemplo, se va a mostrar a continuación un diagrama de estados que representa los estados por los que puede pasar la puerta de un garaje a lo largo de su vida (Figura 6.38).

Para el manejo de esta puerta, se dispone de un mando a distancia con dos botones: uno para solicitar su apertura y otro para solicitar su cierre. Así, del estado inicial, la puerta pasa al estado *Cerrada*. Si en este estado se solicita su cierre, seguirá en el mismo estado, mientras que si alguien solicita su apertura, pasará al estado *Abriendo*. Encontrándose en este estado, cuando se detecte que la puerta ya se ha abierto completamente, pasará al estado *Abierta*. Si en el estado *Abriendo* se solicita su apertura no cambiará de estado, mientras que si se solicita su cierre, pasará al estado *Cerrando*. Si en este estado se solicita su cierre, continuará en el mismo estado, pero si se solicita su apertura, pasará al estado *Abriendo*. Si encontrándose en el estado *Cerrando*, se detecta que la puerta ya se ha cerrado completamente, pasará al estado *Cerrada*. En este caso, no se necesita la existencia de un estado final.

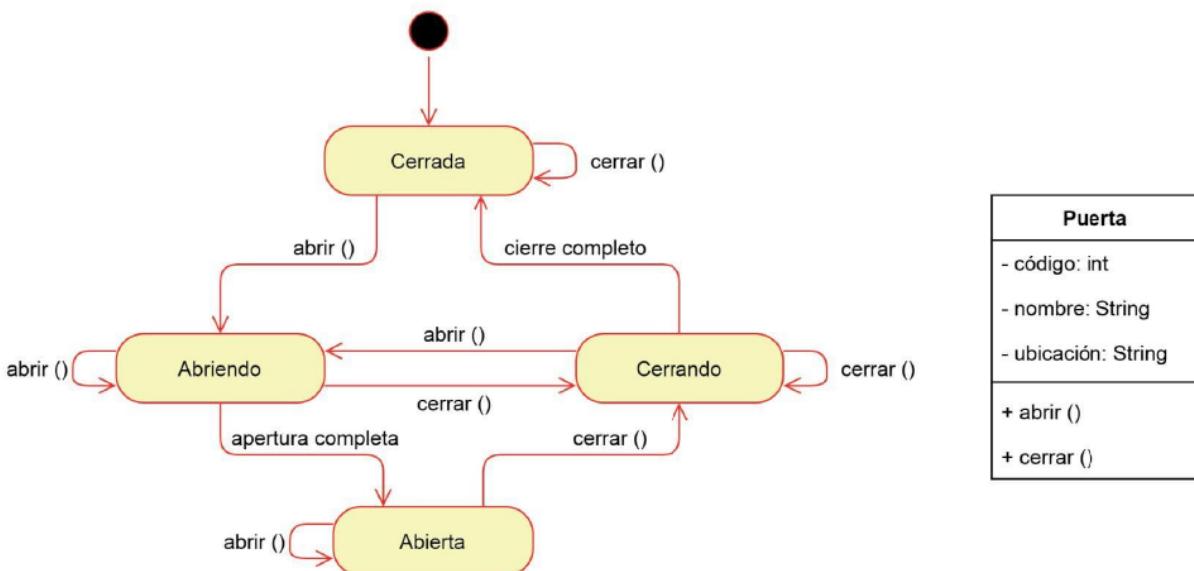


Figura 6.38. Diagrama de estados que representa los estados por los que puede pasar una puerta de garaje.

Sobre las flechas que representan las transiciones, se pueden incluir no solo el evento que origina la transición, sino también los dos elementos que se indican a continuación:

1. Una **condición de guarda**, que se evalúa cuando ocurre el evento, de forma que si la condición es verdadera, se produce la transición, mientras que si es falsa, no se da la transición. Esta condición se debe especificar entre corchetes después del nombre del evento.
2. Una **acción**, que es una operación que se lleva a cabo si se produce la transición y se corresponde con algún método de la clase para la que se está creando el diagrama. También puede ser una solicitud para la ejecución de algún método por parte de otra clase. Se debe escribir esta acción después del evento, con la condición asociada, si es el caso, y después del símbolo de barra (/).

Si se incluyen en el evento una condición de guarda y una acción, el texto que se debe colocar encima de la flecha correspondiente a la transición puede tener alguna de las dos siguientes formas, dependiendo de si la acción se corresponde con un método de la propia clase o de otra clase, respectivamente:

evento [condición] / método

evento [condición] / objeto.método

Para ilustrar lo que se acaba de explicar, aquí se va a realizar el diagrama de estados correspondiente a la clase Ventana de la agenda de excompañeros y excompañeras, reflejando únicamente los estados por los que pasa esta clase en relación con los casos de uso Agregar excompañero/a y Agregar cena.

La ventana parte de un estado llamado *En espera* hasta que el usuario o usuaria selecciona una de las opciones del menú.

En el momento en que se selecciona la opción de menú Agregar excompañero/a, pasa al estado Comprobando excompañero/a con el fin de determinar si el antiguo compañero o compañera que se pretende añadir ya está en la agenda. Si esta persona ya está agregada, se muestra un mensaje de error y se vuelve al estado de espera. En cambio, si esa persona no está en la agenda, se crea este excompañero o excompañera y se vuelve al estado de espera.

Si se selecciona la opción de menú Agregar cena, pasa al estado Comprobando cena con el fin de determinar si ya hay una cena ese año. En este caso, se muestra un mensaje de error y se vuelve al estado de espera. En caso de que no haya cena ese año, se comprueba si el nombre de la persona que organiza la cena es correcto, es decir, si ya existe en la lista de antiguos compañeros y compañeras. Si no existe, se muestra un mensaje de error, y si existe, se agrega la cena a la lista de cenas y se vuelve al estado de espera.

Si en estado de espera se selecciona la opción de menú Salir, se pasa al estado final.

El diagrama de estados del ejemplo anterior se representaría como se muestra en la Figura 6.39.

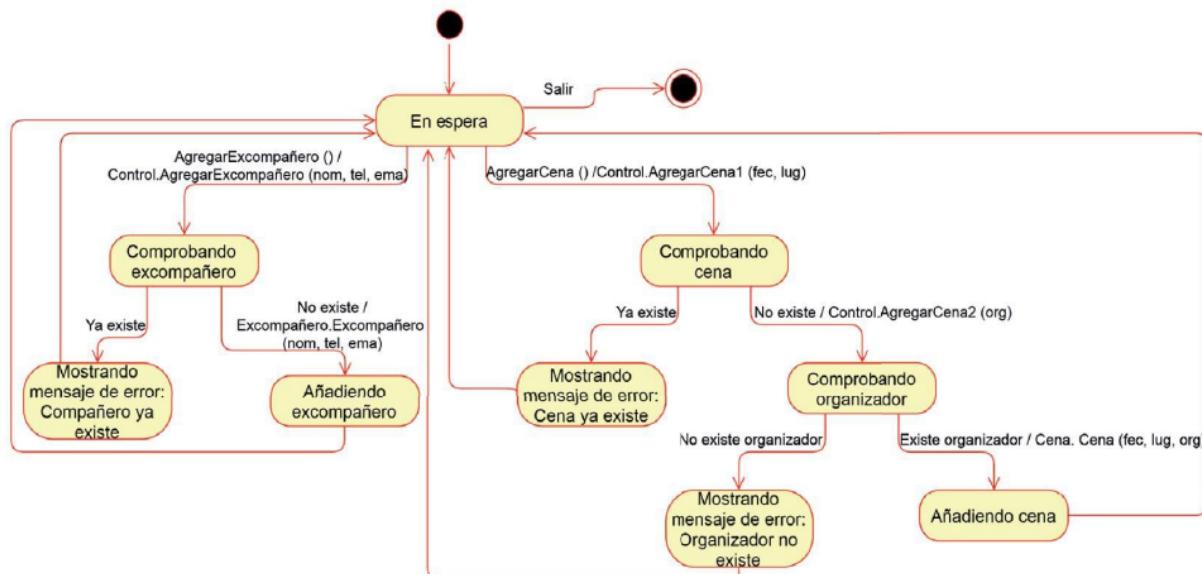


Figura 6.39. Diagrama de estados que representa los estados por los que pasa la clase Ventana en la agenda de excompañeros y excompañeras en relación con los casos de uso Agregar excompañero/a y Agregar cena.

Actividad propuesta 6.3

Diagrama de estados

Realiza el diagrama de estados correspondiente a la clase Ventana de la agenda de antiguos compañeros y compañeras, reflejando únicamente los estados por los que pasa esta clase en relación con los casos de uso Consultar excompañero/a y Eliminar cena.

Actividad resuelta 6.4

Diagrama de estados

Crea un diagrama de estados que refleje el funcionamiento de un microondas que solo dispone de dos botones: uno para ponerlo en funcionamiento y otro para abrir la puerta. El botón de funcionamiento solo tiene efecto cuando la puerta del horno está cerrada, en cuyo caso se calienta la comida durante un tiempo determinado, tras el cual finaliza el calentamiento. Se puede interrumpir el proceso pulsando el botón de apertura de la puerta. Si el microondas está en funcionamiento o la puerta está abierta, se enciende la luz del interior del microondas.

En relación con el funcionamiento del microondas, se pueden aplicar los siguientes métodos: *encender()*, *apagar()*, *encenderLuz()* y *apagarLuz()*.

Solución

Se detectan tres estados: *Apagado con puerta cerrada*, *Apagado con puerta abierta* y *En funcionamiento*. El estado inicial es el primero de los tres.

Si mientras el microondas está apagado con la puerta cerrada, se pulsa el botón de funcionamiento, se deben encender el microondas y su luz y se debe pasar al estado *En funcionamiento*. Si cuando está apagado con la puerta cerrada se pulsa el botón de apertura, se debe encender la luz y pasar al estado *Apagado con puerta abierta*.

Si mientras está apagado con la puerta abierta, se pulsa el botón de funcionamiento ON o el botón de apertura de la puerta, no se produce ningún cambio de estado. Si se cierra la puerta, se debe apagar la luz y se pasa al estado *Apagado con puerta cerrada*.

Si mientras se encuentra en funcionamiento el microondas, se pulsa el botón de funcionamiento ON, no se producirá ningún cambio de estado. Sin embargo, si se pulsa el botón de apertura, se debe apagar el microondas y pasar al estado *Apagado con puerta abierta*. Si, en funcionamiento, expira el tiempo de calentamiento, se debe apagar el microondas y la luz y se debe pasar al estado *Apagado con puerta cerrada*.

En la Figura 6.40 se muestra el diagrama que refleja las situaciones antes descritas.

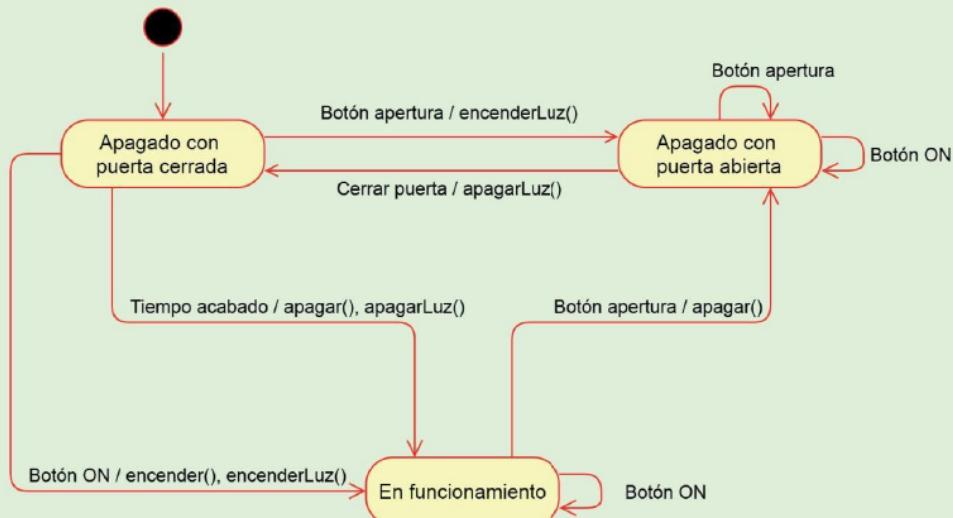


Figura 6.40. Diagrama de estados que representa el funcionamiento de un microondas con dos botones: el de funcionamiento (ON) y el de apertura de la puerta. Este microondas dispone de una luz que debe permanecer encendida mientras está en funcionamiento o si la puerta está abierta.

6.5. Diagramas de actividades

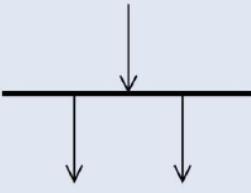
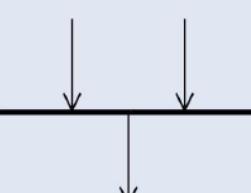
Los diagramas de actividades se usan para mostrar el flujo de control de las acciones que se llevan a cabo en una parte del sistema. En concreto, se pueden usar los diagramas de actividades para:

- Especificar el comportamiento de un método complejo.
- Especificar el comportamiento de un caso de uso.
- Describir el comportamiento de un proceso de negocio o un flujo de trabajo entre las personas usuarias y el sistema.
- Especificar el comportamiento de un objeto o de un conjunto de objetos.
- Definir estados complejos.

Los elementos que se pueden incluir en un diagrama de actividades se recogen en la Tabla 6.6. Como se puede observar, por cada elemento, se indica su nombre, el símbolo que lo representa y una descripción del elemento.

Tabla 6.6. Elementos de un diagrama de actividades

Elemento	Símbolo	Descripción
Nodo de inicio		Representa el inicio del diagrama de actividad.
Nodo de fin		Representa el final del flujo de actividades. Puede que no haya ningún nodo de fin, uno o varios.
Actividad o acción		Representa cada una de las acciones individuales que se ejecutan a lo largo del flujo de trabajo.
Flujo de control		Establece el flujo de control entre las acciones. Después de la acción de la que parte la flecha, se ejecuta la acción a la que llega la flecha.
Bifurcación		El flujo de control se desvía a alguna de las ramas etiquetadas con condiciones. En función de la condición que se cumple, el flujo se dirige por un camino u otro. El rombo de decisión solo puede tener una entrada y puede tener dos o más salidas.
Fusión		El flujo de control de varios caminos se junta en uno solo. El rombo de fusión puede tener varias entradas pero sola una salida.

Elemento	Símbolo	Descripción
División		Un flujo de control se separa en dos o más flujos de control concurrentes. Se representa llegando un flujo a una barra de sincronización y saliendo varios. Sirve para indicar que, a partir de determinado punto, se van a ejecutar varias actividades simultáneamente.
Unión		Varios flujos de control concurrentes se unen en un único flujo de control. Se representa llegando a una barra de sincronización varios flujos y saliendo uno solo. Sirve para indicar que a partir de determinado punto finalizan su ejecución varias actividades simultáneas.

Se muestra a continuación el diagrama de actividades correspondiente al caso de uso *Agregar cena* de la agenda de excompañeros (Figura 6.41). Como se puede observar en el diagrama de secuencia correspondiente a este caso de uso (Figura 6.11), antes de agregar la cena, hay que realizar dos comprobaciones: que no haya una cena ese mismo año y que la persona que organiza la cena se encuentre en la lista de excompañeros y excompañeras. Si se cumplen estas dos condiciones, se procede a añadir la cena a la lista de cenas y, en caso contrario, se muestra el correspondiente mensaje de error.

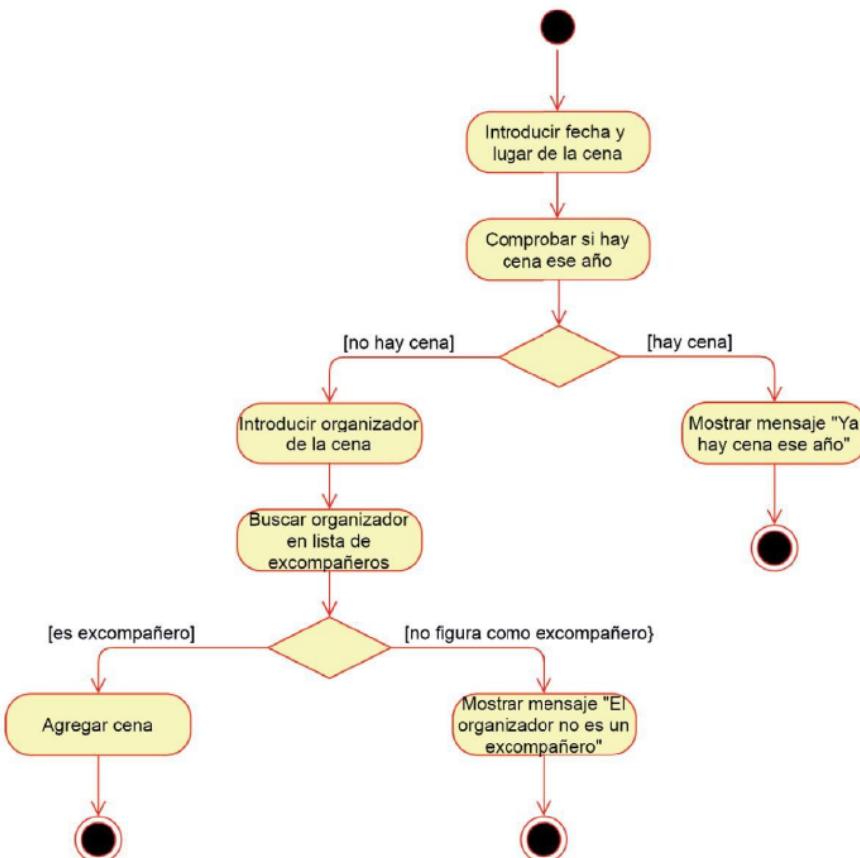
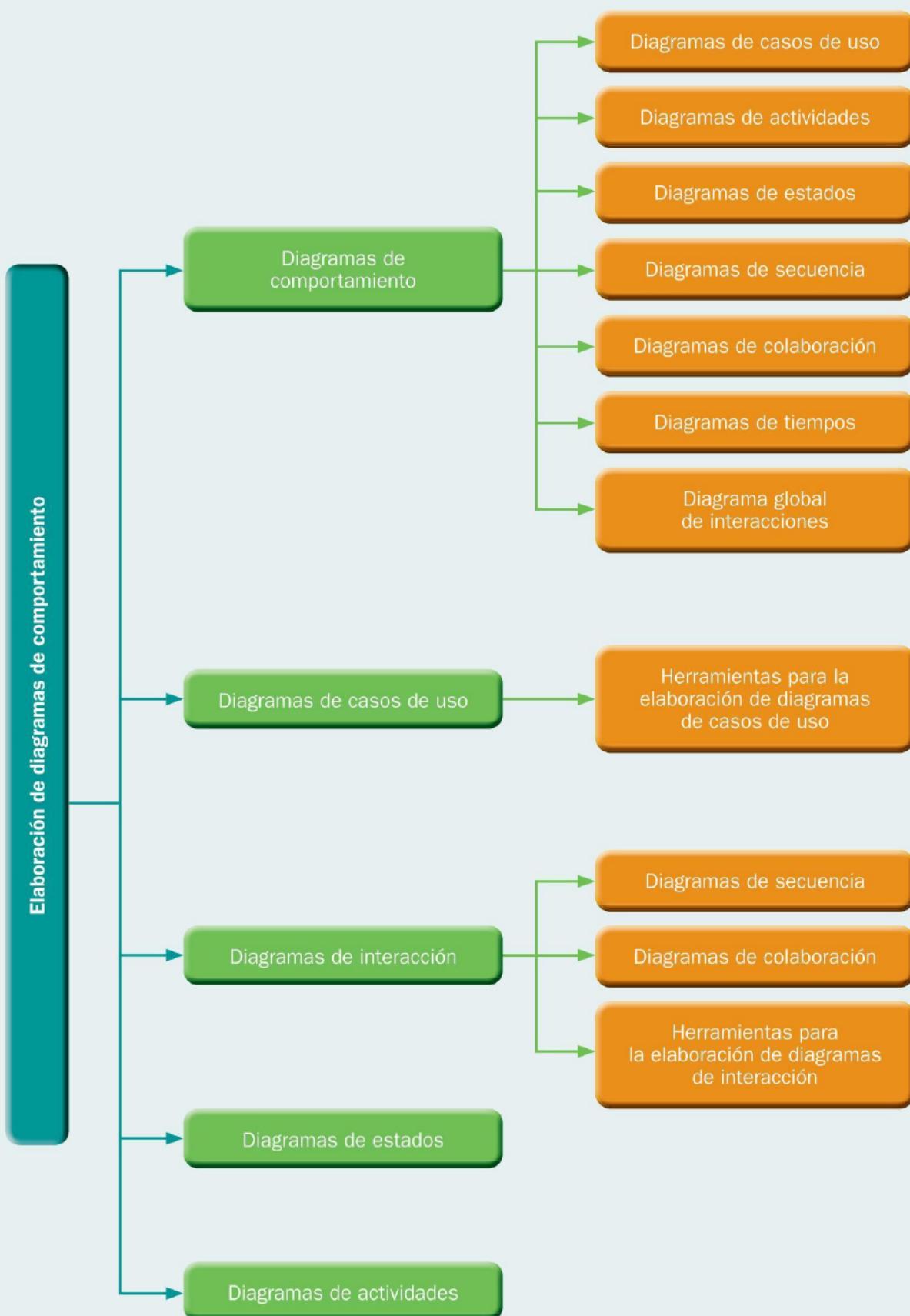


Figura 6.41. Diagrama de actividades que muestra el flujo de control para el caso de uso Agregar cena.



Actividades de comprobación

- 6.1. ¿Qué diagramas de los especificados a continuación se deben crear antes en el proceso de desarrollo de software orientado a objetos?**
- a) Diagramas de secuencia.
 - b) Diagramas de clases.
 - c) Diagramas de colaboración.
 - d) Diagramas de casos de uso.
- 6.2. ¿Qué relación entre casos de uso indica que varios de ellos hacen uso de otro caso de uso de forma que este último incluye actividades comunes a los otros?**
- a) Relación de uso o inclusión.
 - b) Relación de comunicación.
 - c) Relación de extensión.
 - d) Ninguna de las respuestas anteriores es correcta.
- 6.3. ¿Qué apartado de la descripción de un caso de uso indica la secuencia ordenada de pasos por los que pasa el caso de uso?**
- a) La precondición.
 - b) La poscondición.
 - c) Los caminos alternativos.
 - d) El flujo básico.
- 6.4. ¿En qué tipos de diagramas se establecen enlaces entre las clases que se envían mensajes y se muestran los mensajes intercambiados, precediendo a cada mensaje un número que hace referencia al orden del mensaje dentro del diagrama?**
- a) Diagramas de colaboración.
 - b) Diagramas de secuencia.
 - c) Diagramas de casos de uso.
 - d) Diagramas de estados.
- 6.5. En un diagrama de secuencia, si una clase recibe un mensaje solicitando la ejecución de un método:**
- a) Ese método debe estar alojado en la clase que recibe el mensaje.
 - b) Ese método debe estar alojado en la clase que envía el mensaje.
 - c) Ese método debe estar alojado en la clase que recibe el mensaje y en la clase que envía el mensaje.
 - d) Ninguna de las respuestas anteriores es correcta.
- 6.6. Los diagramas de actividades se usan para representar:**
- a) El comportamiento de varios objetos.
 - b) La secuencia de mensajes que se intercambian los objetos involucrados en la realización de un caso de uso.
 - c) El comportamiento de un método o de uno o varios casos de uso.
 - d) Ninguna de las respuestas anteriores es correcta.

6.7. En un diagrama de estados:

- a) Debe haber un único estado inicial y un único estado final.
- b) Debe haber un único estado inicial y puede no haber estado final.
- c) Puede no haber estado inicial y debe haber un único estado final.
- d) Puede no haber ni estado inicial ni estado final.

6.8. Los diagramas UML que sirven para especificar el flujo de control de las acciones que se deben realizar para ejecutar un método son:

- a) Los diagramas de colaboración.
- b) Los diagramas de secuencia.
- c) Los diagramas de estados.
- d) Los diagramas de actividades.

6.9. En un diagrama de estados, un cambio de estado recibe el nombre de _____ y lo que desencadena un cambio de estado se llama _____ :

- a) evento/transición.
- b) acción/transición.
- c) transición/evento.
- d) evento/acción.

6.10. ¿Qué diagramas se podrían emplear para mostrar los mensajes intercambiados entre varios objetos en varios casos de uso?

- a) Los diagramas de secuencia.
- b) Los diagramas de colaboración.
- c) Los diagramas de actividades.
- d) Los diagramas de casos de uso.

6.11. ¿En qué diagrama se muestran todos los actores que intervienen en una aplicación?

- a) En un diagrama de secuencia.
- b) En un diagrama de colaboración.
- c) En un diagrama de actividades.
- d) En un diagrama de casos de uso.

6.12. ¿En qué diagramas se pueden detectar métodos para incorporarlos a las clases que forman parte del diagrama de clases?

- a) En los diagramas de secuencia.
- b) En los diagramas de actividades.
- c) En los diagramas de casos de uso.
- d) Ninguna de las respuestas anteriores es correcta.

6.13. ¿Es posible en un diagrama de actividades representar la ejecución concurrente o simultánea de varias tareas a partir de un determinado momento?

- a) No, no es posible.
- b) Sí, para lo que se debe usar un elemento llamado *bifurcación*.
- c) Sí, para lo que se debe usar un elemento llamado *división*.
- d) Sí, para lo que se debe usar un elemento llamado *fusión*.

Actividades de aplicación

6.14. Crea el diagrama de casos de uso que refleje la gestión de los pedidos de una empresa que vende una serie de productos de acuerdo con la siguiente descripción:

- Cuando un cliente llama por teléfono o se persona en la empresa para realizar un pedido, le atiende una vendedora, la cual, para poder registrar el pedido, en primer lugar, le solicita al cliente sus datos personales, para más tarde tomar los datos del pedido en cuestión. La vendedora acordará con el cliente la forma de pago.
- Los clientes también pueden consultar en cualquier momento el estado de sus pedidos, información que será proporcionada por la vendedora.
- Por otro lado, una vez que la empresa dispone de los productos solicitados por el cliente, la persona responsable de los envíos se encarga de gestionar su envío al domicilio del cliente.

6.15. Crea el diagrama de casos de uso para una librería *online* de acuerdo con la siguiente descripción y sin emplear relaciones «*include*» ni «*extend*»:

- Cualquier persona usuaria puede buscar los libros que desee aunque no esté registrada. Sin embargo, antes de comprar libros, esta persona debe registrarse suministrando sus datos personales, preferencias literarias, un nombre de usuario y una contraseña. Al finalizar el registro, se le envía a dicha persona un correo electrónico confirmándole su registro.
- Cualquier usuaria o usuario registrado puede pedir los libros que considere, añadiéndolos al carrito de la compra. Cuando la persona usuaria selecciona la opción *Pedir libros*, se le solicita un nombre de usuario y contraseña y, si estos son correctos, se le muestran los datos de los libros en pantalla y su dirección de envío, que puede ser modificada. Una vez confirmados estos datos, debe proceder al pago mediante tarjeta bancaria, introduciendo los datos de la tarjeta. Si estos datos son correctos, se procede al cobro y se le muestra un número de pedido para posteriores consultas del estado del pedido.
- Cualquier usuaria o usuario registrado puede consultar el estado de su pedido, para lo que debe introducir el nombre de usuario y contraseña y un número de pedido.

6.16. Realiza la descripción del caso de uso *Registro* para la librería *online* de la Actividad 6.15.

6.17. Realiza la descripción del caso de uso *Pedir libros* para la librería *online* de la Actividad 6.15.

6.18. Modifica el diagrama de casos de uso de la Actividad 6.15, teniendo en cuenta que si al pedir un libro, un usuario o usuaria no está registrado, se le permite hacerlo en ese momento. No obstante, sigue habiendo la opción de registrarse de manera independiente.

6.19. Modifica el diagrama de casos de uso de la Actividad 6.18, incorporando alguna relación «*include*».

- 6.20.** Crea el diagrama de casos de uso para una academia de acuerdo con la siguiente descripción:
- Esta academia imparte diferentes cursillos a lo largo del año. Cada cursillo tiene asignado un código, un nombre, una duración en horas, una fecha de inicio, una fecha de fin y se imparte ciertos días de la semana en un horario determinado. De la introducción de los datos de cada cursillo se encarga la secretaría de la academia.
 - Además, para realizar cada cursillo es necesario disponer de una determinada titulación académica. Por cada una de estas, se registra un código y un nombre (graduado/a en ESO, bachillerato, ingeniero/a, etc.).
 - Las y los alumnos se pueden apuntar en la academia en cualquier momento a lo largo del año. El registro de un alumno o alumna en la academia implica anotar los siguientes datos: NIF, nombre y apellidos, dirección, teléfono, correo electrónico y titulación de mayor nivel que posee. La secretaría se encarga de introducir estos datos en el sistema.
 - Cuando un alumno o una alumna se desea matricular en un determinado cursillo, se lo solicita a la secretaría. El sistema comprobará que su titulación se corresponda con la requerida para realizar el cursillo y, en caso de que así sea, se matriculará al alumno o alumna asignándole un número de matrícula por cada cursillo en el que se matricule.
 - Es necesario registrar en el sistema, por cada cursillo en el que se matricula un alumno o alumna, si él o ella ha asistido al 80 % de las clases o no y su calificación (apto o no apto). Al finalizar el cursillo, la secretaría se encargará de generar los títulos para quienes tengan la calificación de apto.
- 6.21.** Crea el diagrama de secuencia para el caso de uso de la Actividad 6.20 relacionado con la matrícula del alumnado en un cursillo.
- 6.22.** Crea el diagrama de clases que refleje todas las clases con sus atributos y los métodos detectados en el diagrama de secuencia de la Actividad 6.21.
- 6.23.** Crea un diagrama de actividades para el caso de uso *Consultar excompañero/a*, cuyo diagrama de secuencia se muestra en la Figura 6.10.
- 6.24.** Crea un diagrama de actividades para el caso de uso *Agregar asistente*, cuyo diagrama de secuencia se muestra en la Figura 6.13.
- 6.25.** Crea un diagrama de estados que refleje los estados por los que puede pasar un mensaje de correo electrónico teniendo en cuenta lo siguiente: inicialmente el mensaje se encuentra en estado no leído y, desde esta situación, o bien se puede eliminar, con lo que pasaría a la papelera, o bien se puede leer, en cuyo caso pasaría a la situación de leído. Desde esta situación, también se puede eliminar, por lo que pasaría a la papelera. Estando en la papelera, el mensaje se puede eliminar definitivamente. Además, si han transcurrido 30 días desde que se envió a la papelera, también es eliminado de manera definitiva por el servidor de correo.

En la Figura 6.42, se muestra la clase *Email* con los mensajes que provocan un cambio de estado: *leer()*, *eliminar()* y *eliminarDefinitivamente()*.

Email
- fecha
- emisor
- asunto
- contenido
+ leer()
+ eliminar()
+ eliminarDefinitivamente()

Figura 6.42. Clase Email con métodos para leer el mensaje, eliminarlo para enviarlo a la papelera y eliminarlo de manera definitiva.

- 6.26.** Crea un diagrama de estados que refleje los estados por los que puede pasar un préstamo solicitado a una entidad bancaria. Se debe tener en cuenta que el préstamo comienza en estado solicitado. En algún momento se procede a su estudio, como resultado del cual, puede ser rechazado o aceptado por la entidad bancaria. Se debe comunicar este estado al cliente, que puede formalizarlo o rechazarlo. En caso de que lo formalice, a partir de ese momento, cada cierto tiempo, el cliente procederá a su pago parcial o a su amortización (pago por completo), momento en el cual el préstamo quedará pagado. En la Figura 6.43, se muestra la clase Préstamo con los mensajes que provocan un cambio de estado: *estudiar()*, *aceptarBanco()*, *rechazarBanco()*, *comunicar()*, *formalizar()*, *rechazarCliente()*, *pagar()* y *amortizar()*.

Préstamo
- código
- importe
- plazo
- cuota
+ estudiar()
+ aceptarBanco()
+ rechazarBanco()
+ comunicar()
+ formalizar()
+ rechazarCliente()
+ pagar()
+ amortizar()

Figura 6.43. Clase Préstamo con métodos para estudiar el préstamo, su aceptación o rechazo por parte del banco, su comunicación al cliente, su formalización, su rechazo por parte del cliente, su pago parcial y su amortización total.

Actividades de ampliación

- 6.27. A veces, en los diagramas de secuencia, es necesario incluir mensajes que se han de enviar solo en caso de que se cumpla una determinada condición, de manera similar a una estructura alternativa simple de programación. Busca información sobre cómo se puede representar esta situación en un diagrama de secuencia.
- 6.28. También existe la posibilidad de representar en los diagramas de secuencia estructuras alternativas dobles o múltiples. Averigua cómo se pueden representar dichas situaciones en un diagrama de secuencia.
- 6.29. En ocasiones, es necesario incluir en los diagramas de secuencia mensajes que se han de enviar varias veces mientras se cumpla una determinada condición, de manera similar a una estructura repetitiva *while*. Busca información sobre cómo se puede reflejar esta situación en un diagrama de secuencia.
- 6.30. Modifica el diagrama de secuencia correspondiente al caso de uso *Agregar excompañero/a* de la Figura 6.9 para detallarlo más, empleando una estructura alternativa simple y una estructura repetitiva.
- 6.31. Modifica el diagrama de secuencia correspondiente al caso de uso *Agregar cena* de la Figura 6.11 con el fin de detallarlo más. Para ello, haz uso de estructuras alternativas y repetitivas.
- 6.32. En los diagramas de estados, se representa un estado mediante un rectángulo con el nombre del estado en su interior, pero también existe la posibilidad de representar los estados con mayor detalle indicando para cada estado la acción de entrada (*entry*), la acción de salida (*exit*) y la acción interna (*do*). Busca información sobre esta representación de los estados y explica en qué consiste cada uno de los elementos indicados (*entry*, *exit* y *do*).

Enlaces web de interés

-  **DiagramasUML** - <https://diagramasuml.com/>
(Tutorial sobre diagramas UML con numerosos ejemplos)
-  **UML-diagrams.org** - <https://www.uml-diagrams.org/>
(Sitio web en inglés con información sobre todos los diagramas UML)
-  **Diagrams.net** - <https://www.diagrams.net/>
(Sitio web sobre la herramienta de modelado de código abierto app.diagrams.net)