

Diagramas de COMPORTAMIENTO

Diagramas de comportamiento

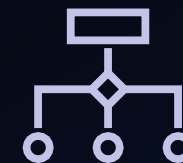
Diagrama de comportamiento: expresa las secuencias de estados por los que pasa un objeto a lo largo de su vida en respuesta a eventos.

Al mostrar diferentes estados de un proceso plasma de forma gráfica los procesos a programar. Se usan para visualizar y especificar, a la vez que documentar, aspectos dinámicos de un sistema.

Cuando hablamos de desarrollo de software, estos aspectos dinámicos pueden ser mensajes que se generan, acciones de entrada de datos, eventos, etc...

Estos diagramas pueden ser muy simples o muy complejos en función del proceso que están representando. Hacen referencia a objetos, ya que se emplean mucho en la programación orientada a objetos. Un objeto puede ser cualquier entidad con un determinado estado y comportamiento.

Se suele utilizar para su desarrollo la notación UML (Unified Modeling Language), un estándar de lenguaje para el modelado de sistemas de software, recoge un conjunto de símbolos apropiados para cada tipo de estado o acción.



Diagramas de flujo



Diagramas de flujo

El diagrama de flujo o también diagrama de actividades es **una manera de representar gráficamente un algoritmo o un proceso** de alguna naturaleza, a través de una serie de pasos estructurados y vinculados que permiten su revisión como un todo.

Emplea, una serie determinada de **figuras geométricas que representan cada paso** puntual del proceso que está siendo evaluado. Estas formas definidas de antemano se conectan entre sí a través de flechas y líneas que marcan la dirección del flujo y establecen el recorrido del proceso, como si de un **mapa** se tratara.

Dispone de un **inicio y un fin claro**.

Diagramas de flujo








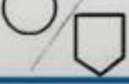

~~El diagrama de flujo o también diagrama de actividades es~~ **una manera de representar gráficamente un algoritmo o un proceso** ~~de alguna naturaleza, a través de una serie de pasos estructurados y vinculados que permiten su revisión como un todo.~~

~~Emplea, una serie determinada de~~ **figuras geométricas que representan cada paso** ~~puntual del proceso que está siendo evaluado. Estas formas definidas de antemano se conectan entre sí a través de flechas y líneas que marcan la dirección del flujo y establecen el recorrido del proceso, como si de un~~ **mapa** ~~se tratara.~~

~~Dispone de un~~ **inicio y un fin claro.**

Simbología y su función

LOS + UTILIZADOS

Nombre	Símbolo	Función
Inicio/Final		Representa el inicio o fin de un proceso
Proceso		Representa la actividad llevada a cabo
Entrada/Salida		Representa información que ingresa o sale del sistema
Decisión		Indica un punto de toma de decisiones
Línea de Flujo		Indica el orden y sentido del flujo del proceso
Documento		Indica los documentos utilizados en el proceso
Base de datos		Representa la grabación de datos
Conector Interno/ Conector Externo		Enlace dentro de la misma página/ Enlace en diferente página
Retraso		Retraso para iniciar el siguiente proceso

LOS + UTILIZADOS

... y cómo programador ¿qué uso le doy...?



... y cómo programador ¿qué uso le doy...?

Si existe un diagrama de flujo, existe un proceso o sistema que pretende ser graficado a través de símbolos visuales que, en vez de términos verbales, simplifican el funcionamiento de dicho proceso.

Utilizado en **programación**, la economía, los procesos técnicos y/o tecnológicos, la psicología, la educación y casi cualquier temática de análisis.

Los **diagramas de flujo son múltiples y diversos**, el aspecto en común entre ellos es la presencia de un vínculo entre los conceptos enunciados y una interrelación entre las ideas.

Software gratuito para hacer diagramas

<https://app.diagrams.net/>

Apretar hipervínculo



No es necesario crear cuenta.

Permite trabajar sobre local o GOOGLE DRIVE.

Permite elaborar distintos tipos de diagramas.

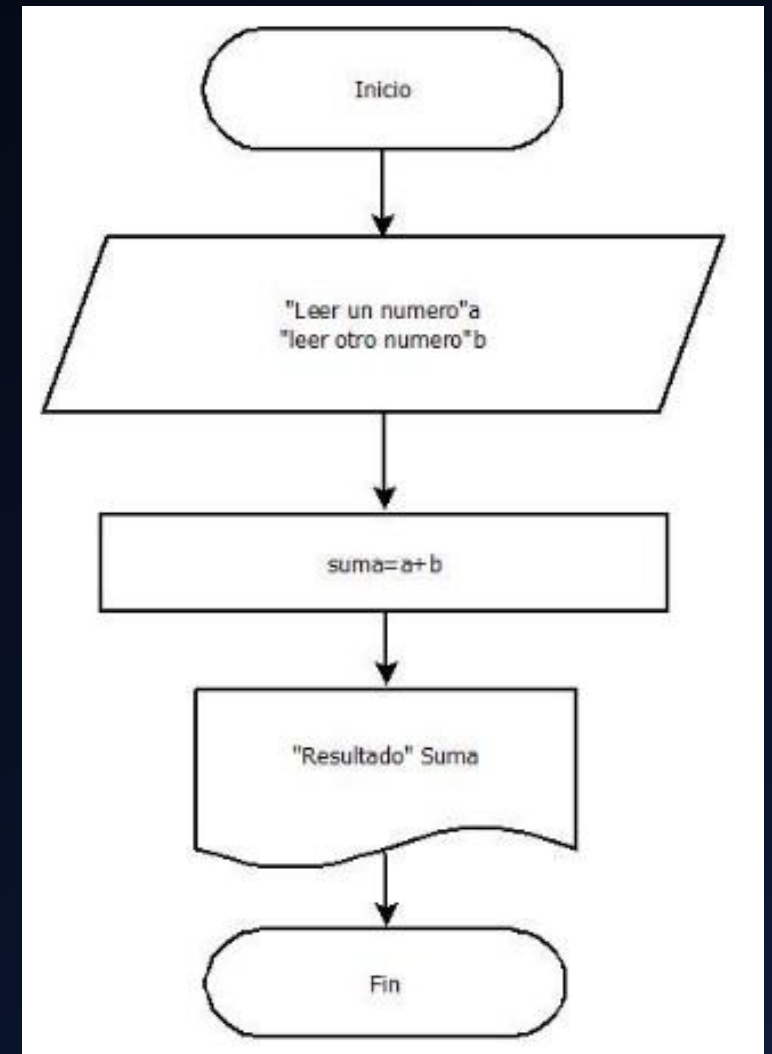
Muy intuitivo y fácil de usar.

Ejemplo 1.

Cocinar un huevo para otra persona



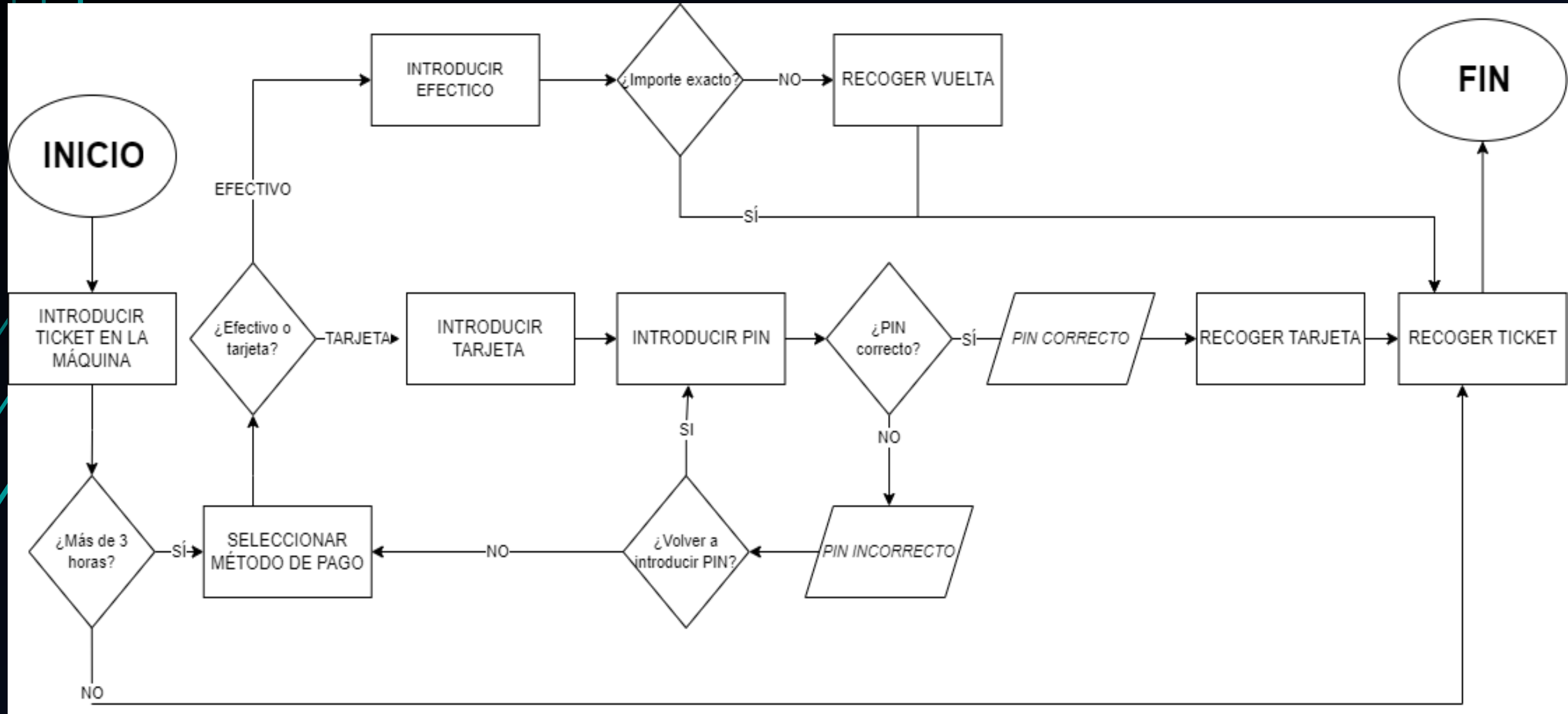
Programa informático que nos sume dos número y nos de el resultado en pantalla.



Ejemplo 2.

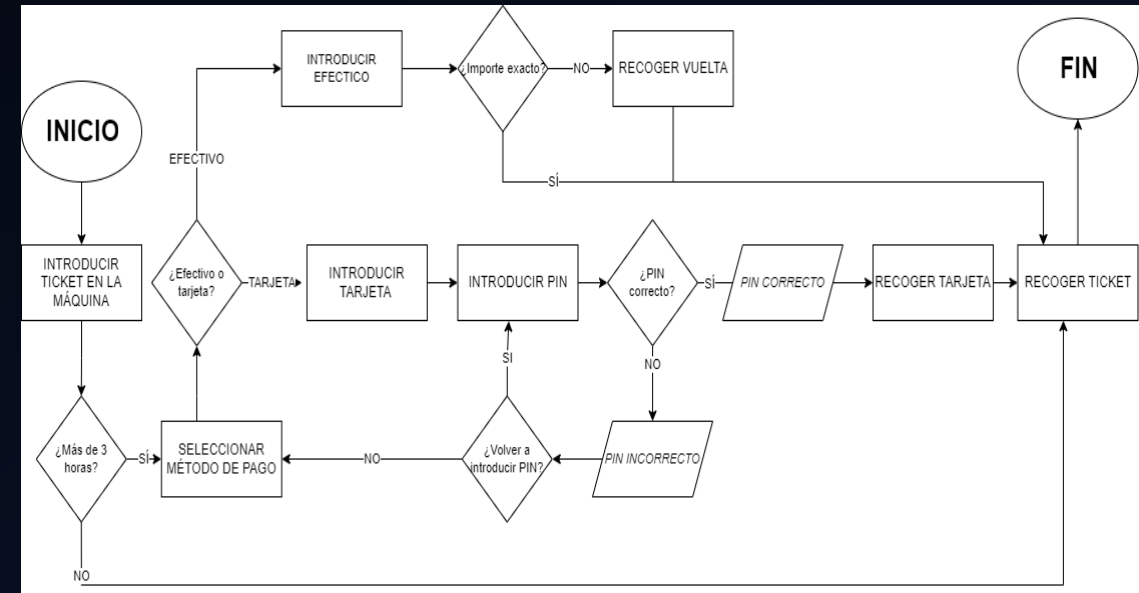
Tengo un parking de alrededor de 1000 plazas, por lo que es inviable que un operario pueda llevar el control de pagos, entradas y salidas de todos los vehículos que estacionen. Por tanto, voy a poner una valla de control de acceso, además de repartir parquímetros que permitan el pago de los usuarios por el parking. Por tanto necesito un programa para esas máquina que funcione tal y como explico. Estos parquímetros deben funcionar de la siguiente manera, una vez el usuario introduce el ticket, si han pasado menos de 3 horas se lo devuelva directamente, si ha pasado más tiempo, debe pagar. El usuario del parking podrá pagar tanto en efectivo como con tarjeta. En caso de que decida pagar con tarjeta, obviamente la máquina solo puede cobrarse si introduce el PIN correcto, en cualquier caso, el parquímetro debe mostrar un mensaje indicando si el PIN introducido es correcto o no. Después de validar el PIN, el parquímetro deberá devolver la tarjeta y luego el ticket. En caso de que el PIN sea incorrecto, además de indicarlo, la máquina debe dar la opción de volver a introducir el PIN o de cambiar el método de pago. Si por el contrario, el cliente decide pagar en efectivo, la máquina debe cobrarse el importe al tiempo que haya estacionado su vehículo. Si el usuario introduce un importe superior al correspondiente la máquina devuelve el cambio y después, devuelve el ticket, en caso de que introduzca el importe exacto, el parquímetro devolverá el ticket directamente.

Ejemplo 2.



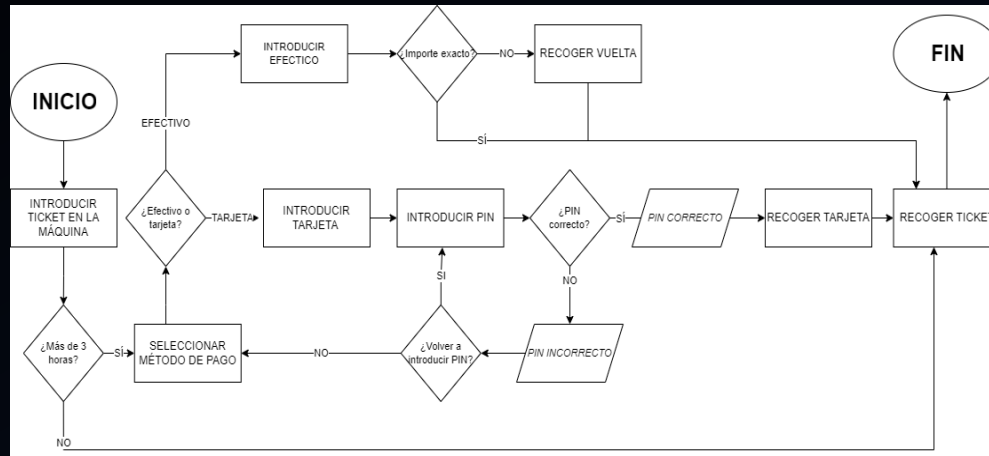
Ejemplo 2.

Tengo un parking de alrededor de 1000 plazas, por lo que es inviable que un operario pueda llevar el control de pagos, entradas y salidas de todos los vehículos que estacionen. Por tanto, voy a poner una vaya de control de acceso, además de repartir parquímetros que permitan el pago de los usuarios por el parking. Por tanto necesito un programa para esas máquina que funcione tal y como explico. Estos parquímetros deben funcionar de la siguiente manera, una vez el usuario introduce el ticket, si han pasado menos de 3 horas se lo devuelve directamente, si ha pasado más tiempo, debe pagar. El usuario del parking podrá pagar tanto en efectivo como con tarjeta. En caso de que decida pagar con tarjeta, obviamente la máquina solo puede cobrarse si introduce el PIN correcto, en cualquier caso, el parquímetro debe mostrar un mensaje indicando si el PIN introducido es correcto o no. Después de validar el PIN, el parquímetro deberá devolver la tarjeta y luego el ticket. En caso de que el PIN sea incorrecto, además de indicarlo, la máquina debe dar la opción de volver a introducir el PIN o de cambiar el método de pago. Si por el contrario, el cliente decide pagar en efectivo, la máquina debe cobrarse el importe al tiempo que haya estacionado su vehículo. Si el usuario introduce un importe superior al correspondiente la máquina devuelve el cambio y después, devuelve el ticket, en caso de que introduzca el importe exacto, el parquímetro devolverá el ticket directamente.



Ejemplo 2.

A partir de este diagrama de flujo



Entendemos claramente el funcionamiento de la máquina y los pasos que debe seguir el usuario, lo que nos permitirá ir a tiro hecho con su programación.

```
31
32 self.file = None
33 self.fingerprints = set()
34 self.logdups = True
35 self.debug = debug
36 self.logger = logging.getLogger(__name__)
37 if path:
38     self.file = open(os.path.join(path, "requests.log"),
39                     "a")
40     self.file.seek(0)
41     self.fingerprints.update([self._get_fingerprint(request) for request in requests])
42
43 @classmethod
44 def from_settings(cls, settings):
45     debug = settings.getbool("debug")
46     return cls(job_dir(settings), debug)
47
48 def request_seen(self, request):
49     fp = self.request_fingerprint(request)
50     if fp in self.fingerprints:
51         return True
52     self.fingerprints.add(fp)
53     if self.file:
54         self.file.write(fp + os.linesep)
55
56 def request_fingerprint(self, request):
57     return request_fingerprint(request)
```

Ejercicio 1.



¿A quién no le gusta el café? A mí me encanta.

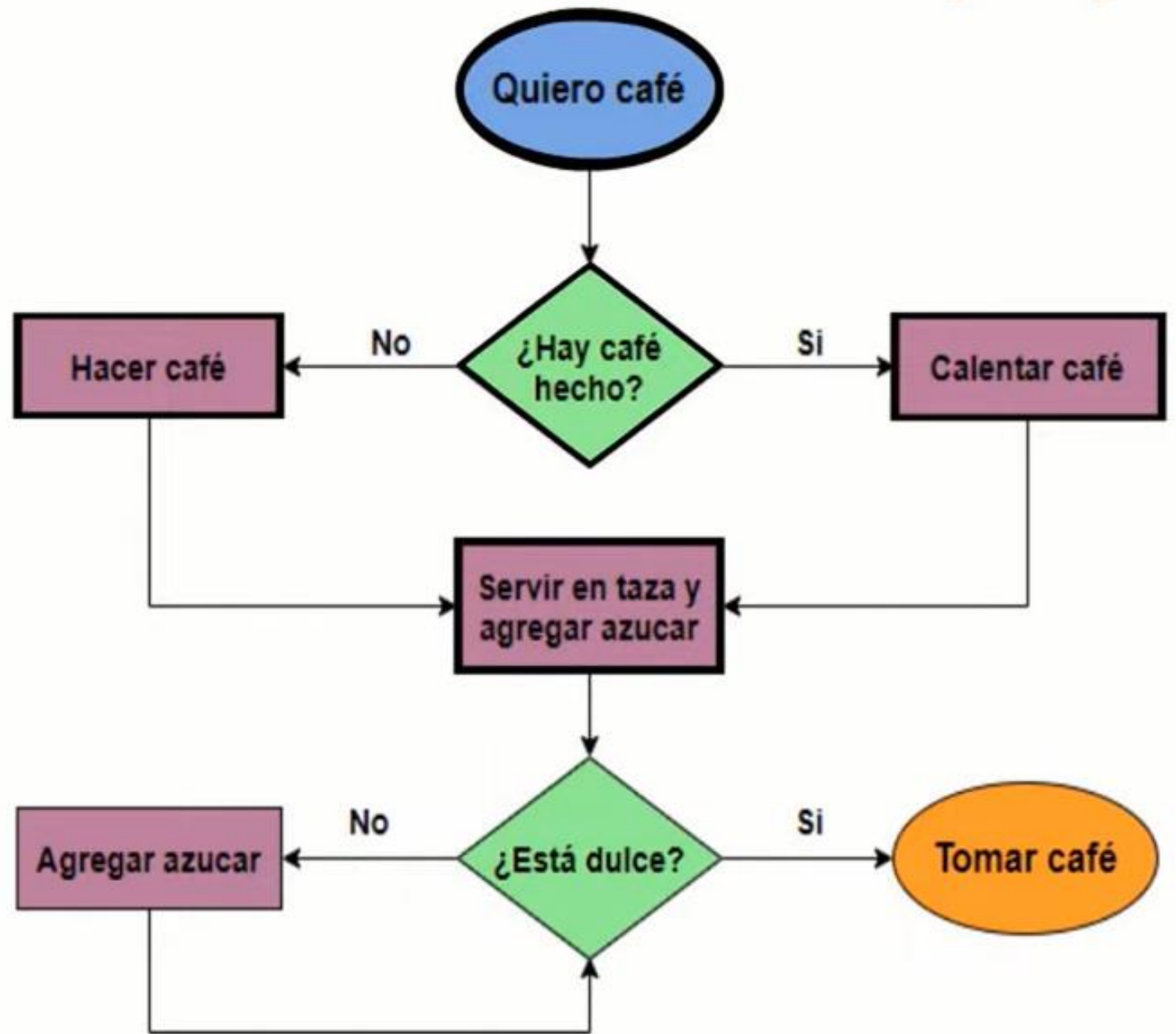


Trabajo en una oficina en la que todos somos muy cafeteros. Cuando quiero un café voy al office y... ¡Tachín! A veces me encuentro café hecho, aun así siempre lo caliente, me gusta ardiendo.

Otras veces debo hacerlo yo, siempre al servirlo en la taza le hecho azúcar, pero soy muy goloso, a veces debo echarle una segunda vez.



Ejercicio 1.



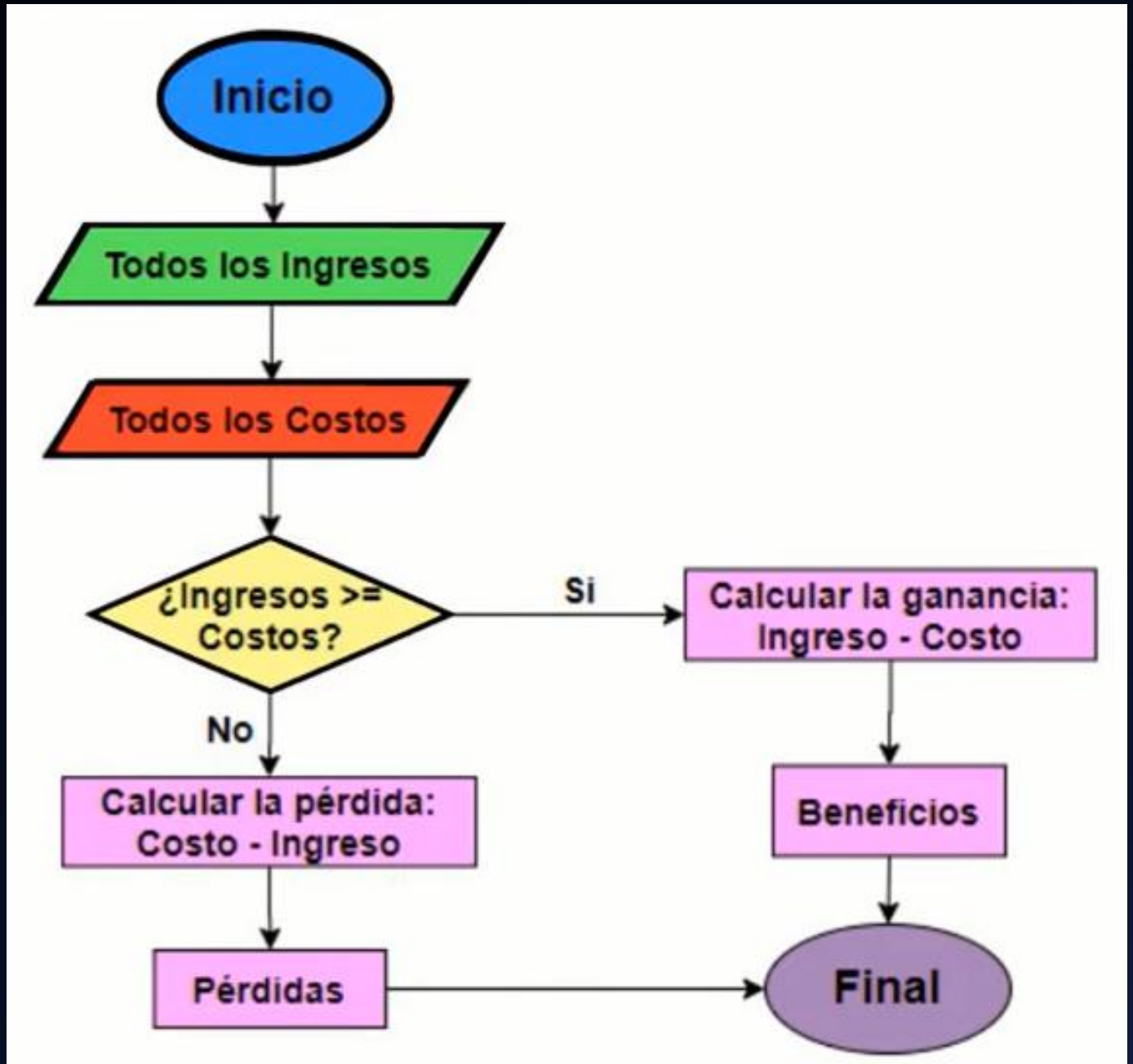
Ejercicio 2.

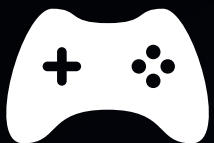
Una empresa necesita nuestra ayuda. Debemos realizar un programa que automatice su balance al final de cada jornada, para saber si ha obtenido o no beneficio. Si la empresa le es rentable o no.

Para ello, debemos introducir en nuestro programa todos los ingresos y costes realizados durante el día, datos almacenados que nos proporcionará el cliente. Si la suma de los ingresos es mayor o igual que la suma de los costes, habrá obtenido un beneficio, el cual quiere conocer numéricamente. En caso contrario, desea conocer sus pérdidas, también de manera numérica.

El cliente no entiende de programación, solo quiere saber si su negocio puede seguir abriendo y que de forma sencilla y visual, le expliques que vas a hacer.

Ejercicio 2.





¡ ¡ HORA DE JUGAR !!



El gran concurso de los diagramas de flujo:

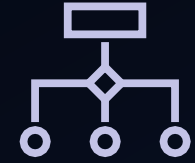
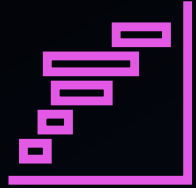
<https://wordwall.net/es/resource/23387963/diagrama-de-flujo>



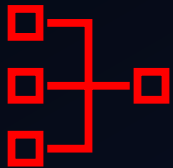
El ARCADE de los diagramas de flujo:

<https://wordwall.net/es/resource/29688334/diagrama-de-flujo>





Diagramas de casos de uso



Diagramas de casos de uso

Los casos de uso especifican un comportamiento deseado del sistema, representan requisitos funcionales del mismo.

Es importante resaltar que describen qué hace el sistema, no cómo lo hace.

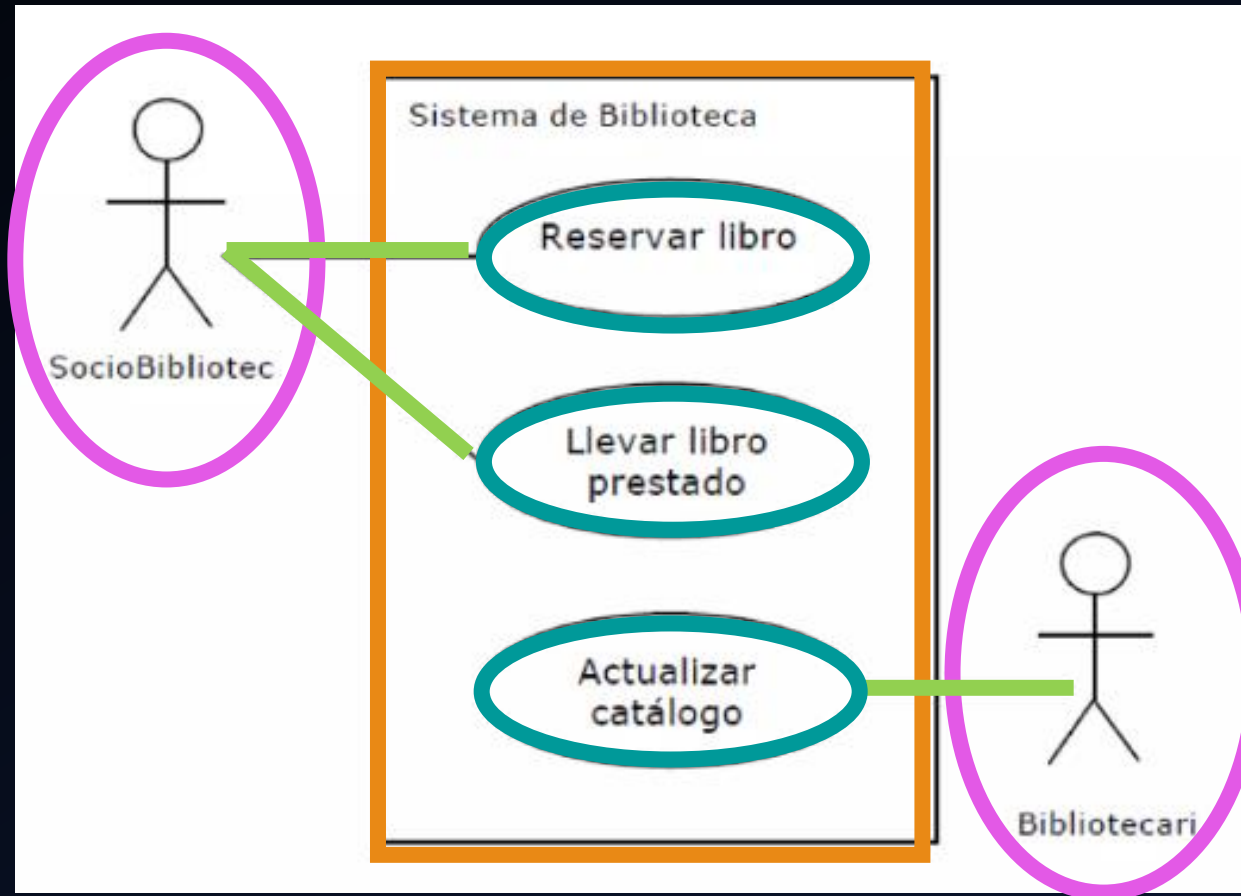
UML nos dice que: “Un caso de uso especifica un conjunto de secuencias de acciones, incluyendo variantes, que el sistema puede ejecutar y que produce un resultado observable de valor para un particular actor”.

Diagramas de casos de uso

Actores

Escenario

Relación de comunicación



Diagramas de casos de uso. RELACIÓN DE COMUNICACIÓN



La relación de comunicación indica la asociación existente entre actores y casos de uso, existen las siguientes:

- **Comunicación:** Relación (asociación) simple entre un actor y caso de uso. Su código es: <<communicate>> aunque generalmente no se nombra, solo se dibuja.
- **Inclusión:** Se usa cuando un caso de uso base (o principal, ejecutado directamente por el actor) incorpora necesariamente el comportamiento de otro caso de uso. Usado para enriquecer y compartir una funcionalidad o describir subfunciones. No responde directamente a un objetivo de un actor, ni tiene sentido por si solo. Su código es: <<include>> .
- **Exclusión:** Se usa cuando un caso de uso base incorpora, PERO NO necesariamente, el comportamiento otro caso de uso. Identifica un subflujo que sólo se ejecuta bajo ciertas condiciones. Este tipo de relación produce confusión y no debería utilizarse en exceso. Su código es: <<extend>>.
- **Especialización y generalización:** Un caso de uso “hijo” (subcaso) hereda el comportamiento y significado del caso de uso “padre” (supercaso). Los casos de uso “hijo” son una especialización del caso de uso “padre”. Suele ser una opción o decisión del actor para ejecutar correctamente una caso de uso base.

Diagramas de casos de uso. RELACIÓN DE COMUNICACIÓN



SIMBOLOGÍA:

- Comunicación
- Inclusión
- Exclusión
- Especialización y generalización

Actor

Caso de uso
base

Caso de uso
base

<<include>>

<<extend>>

Subcaso

Diagramas de casos de uso. ACTORES

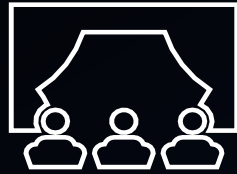


El actor en un diagrama de caso de uso es cualquier entidad que desempeñe un papel en un sistema determinado. Puede ser una persona, una organización o un sistema externo.

Por norma el cliente/usuario (quien inicia la acción) se coloca a la izquierda y la empresa/empleador (quien reacciona) a la derecha.

Siempre fuera del **escenario**.

Diagramas de casos de uso. **ESCENARIO**

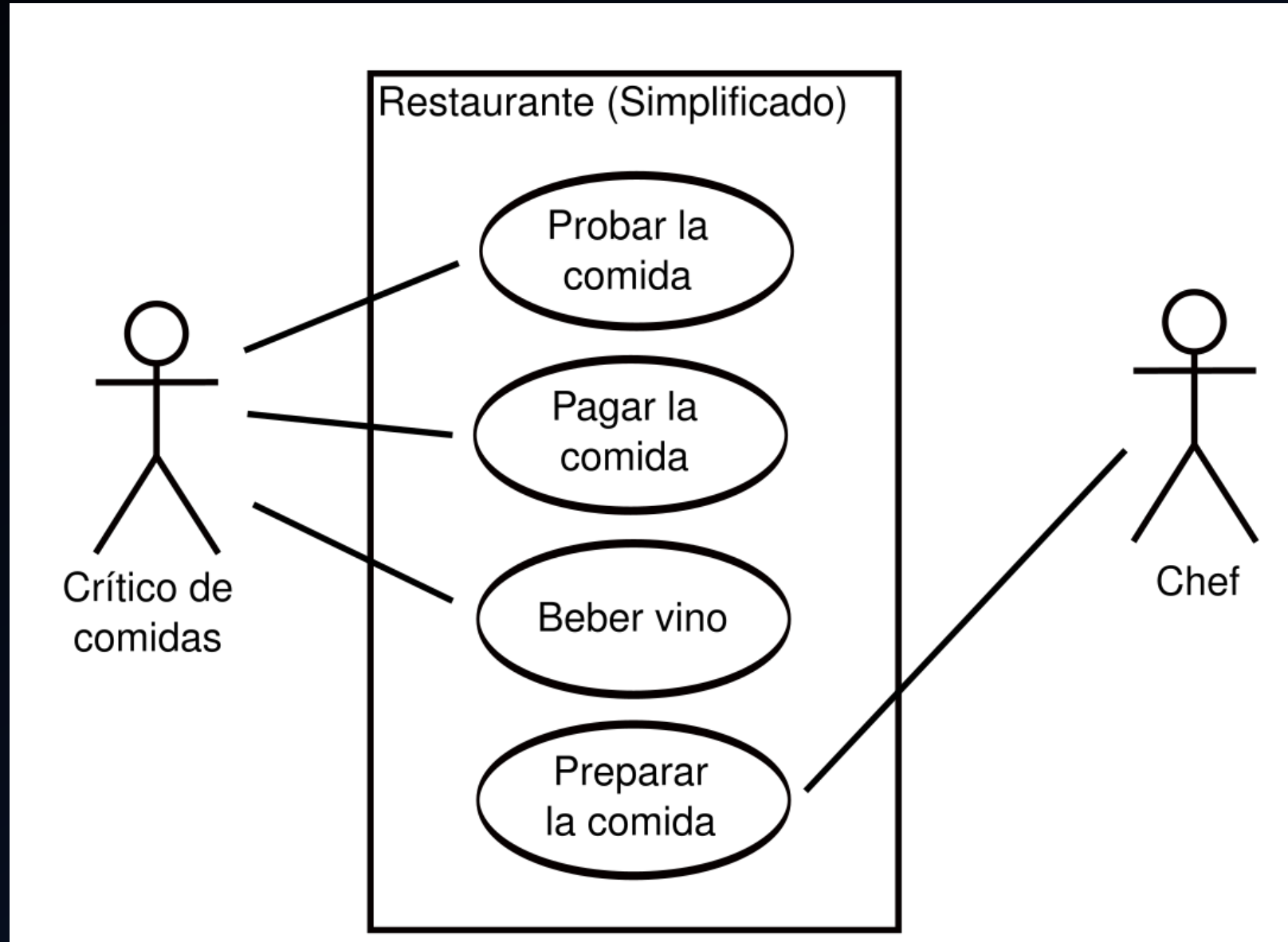


El escenario (o sistema) se utiliza para definir el alcance del caso de uso y se dibuja como un rectángulo.

Siempre debe tener un título, el cual nos permita identificar donde estamos: una APP, página web, software, etc.

EJEMPLO

Ejemplo 1.

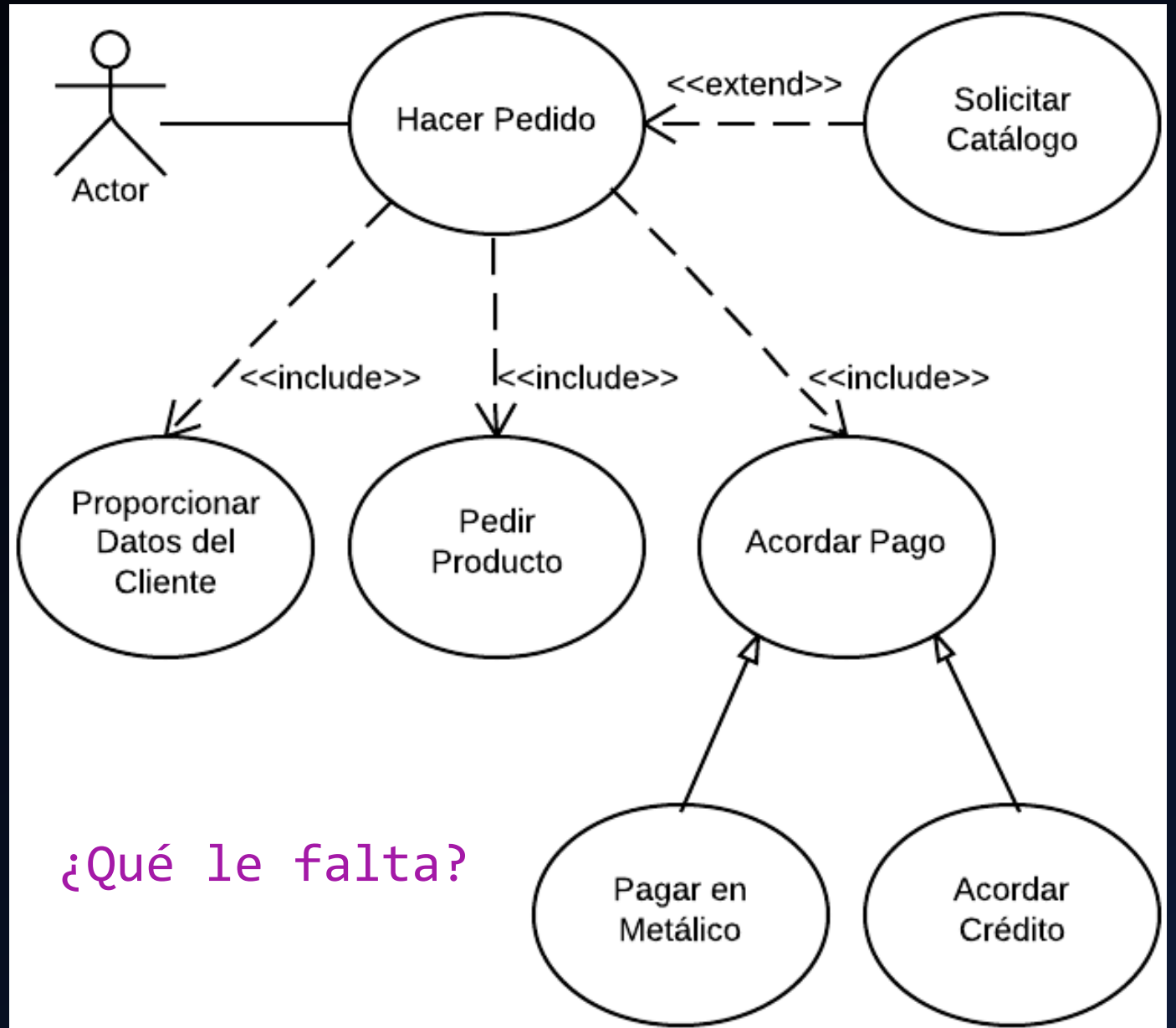


Ejemplo 2.

<<include>>: Caso de uso NECESITA comportamiento de otro caso de uso.

<<extend>>: Caso de uso base incorpora, PERO NO NECESITA, el comportamiento otro caso de uso.

Especialización y generalización: Caso de uso (subcaso) hereda el comportamiento y significado de otro. Suele ser una opción o decisión del actor.



¿Qué le falta?

Ejercicio 1.

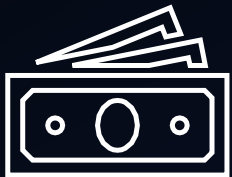


Realizar la APP móvil de un banco que permita:

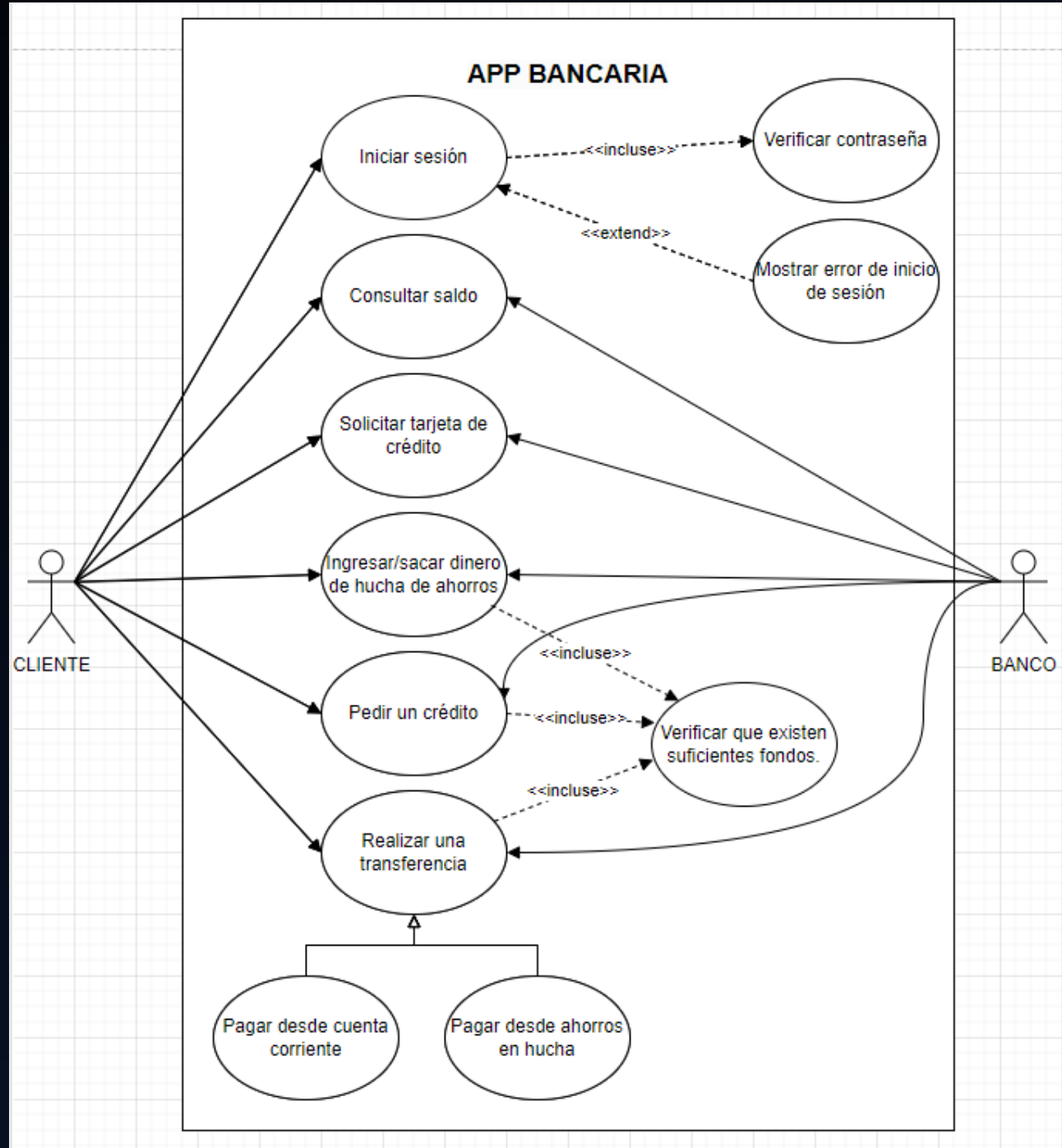
- Al usuario:
 - Iniciar sesión, la cual debe verificarse e indicar si es incorrecta.
 - Consultar saldo disponible.
 - Solicitar una tarjeta de crédito.
 - Ingresar/sacar dinero en hucha de ahorros.
 - Pedir un crédito.
 - Realizar una transferencia, desde su cuenta o su hucha de ahorros.



Habrà un trabajador del banco, el cual intervendrã en determinados procesos.



Ejercicio 1.



Bibliografía

- Diagramas de flujo
 - <https://concepto.de/diagrama-de-flujo/#ixzz6XAjPW7vn>
 - <https://www.youtube.com/watch?v=Kucgc6NpGwc>
 - <https://www.definicionabc.com/comunicacion/diagrama-de-flujo.php>
 - <https://www.ceac.es/blog/elaborar-diagramas-de-comportamiento-en-entornos-de-desarrollo#:~:text=desarrollo%20de%20software-,%C2%BFQu%C3%A9%20es%20un%20diagrama%20de%20comportamiento%3F,diferentes%20estados%20de%20un%20proceso.>
- Diagrama de casos de uso
 - <https://www.youtube.com/watch?v=zid-MVo7M-E>
 - <https://es.slideshare.net/MiguelSanchez14/diagramas-de-casos-de-uso-24202773>
 - <https://creately.com/blog/es/diagramas/tutorial-diagrama-caso-de-uso/>
 - <https://sites.google.com/site/informaticafrancisco/home/11-grado-undecimo/actividad-2?tmpl=%2Fsystem%2Fapp%2Ftemplates%2Fprint%2F&showPrintDialog=1>
- Apps para Diseño de diagramas
 - <https://app.diagrams.net/> ONLINE
 - <https://www.diagrameditor.com/> ONLINE

¡ ¡ MUCHAS GRACIAS ! !

