

T4. DIAGRAMA DE ACTIVIDAD

Entornos de desarrollo
CFGs DAW

Autor: Paco Aldarias
paco.aldarias@ceedcv.es

Curso: 2016/2017

Revisión: b69d8fe

Versión:170110.1944

Índice

1. DIAGRAMAS DE ACTIVIDAD.....	3
2. ELEMENTOS DEL DIAGRAMA.....	3
3. CUANTOS DIAGRAMAS REALIZO.....	5
4. NODO INICIAL Y FINAL.....	5
5. CARRILES.....	5
5.1. Ejemplo. Proceso de pedido.....	6
6. DESCOMPOSICIÓN DE UNA ACTIVIDAD.....	7
6.1. Ejemplo: Determinar el pago.....	7
7. OBJETOS.....	8
8. ACTIVIDAD.....	9
8.1. Ejemplo: Auditorio.....	10
8.2. Ejemplo: Bebida.....	10
9. CONCURRENCIA.....	12
9.1. Ejemplo: Fibonacci.....	14
10. HERRAMIENTAS GRÁFICAS.....	16
11. BIBLIOGRAFÍA Y ENLACES.....	16

1 DIAGRAMAS DE ACTIVIDAD

Los diagramas de actividad son un tipo de diagrama de comportamiento que representan un flujo de actividades paso a paso y permiten expresar condiciones, iteraciones y concurrencia.

Un diagrama de actividad es un tipo especial de diagrama de estados que muestra el flujo de una actividad a otra actividad dentro del sistema.

Al igual que los diagrama de estados, los diagramas de actividad son usados para capturar el comportamiento del flujo de información, sin embargo, ellos especifican las transiciones entre estados sin mostrar estímulos externos.

La transición de estados dentro de un diagrama de actividad ocurre simplemente porque se completan las acciones asociadas con estados previos.

Los diagramas de actividad capturan transiciones de estados para una entidad dada, tal como una clase, una operación, un caso de uso o un subsistema. Son especialmente importantes en el modelado de funciones de un sistema y enfatiza el flujo de control entre objetos. Representan la vista dinámica del sistema. Es un tipo de diagramas de comportamiento.

NOTA: Recuerda que los diagramas UML de comportamiento describen cómo se comporta el sistema, y teníamos:

DIAGRAMA	DESCRIPCION
Diagramas de casos de uso	Organiza el comportamiento del sistema
Diagramas de secuencia	Se centra en el orden de los mensajes
Diagramas de colaboración	Se centra en la estructura organizativa de los objetos que envían y reciben mensajes.
Diagramas de actividad.	Se centra en el flujo del control de una actividad a otra.
Diagrama de estados	Se centra en los cambios de estados del sistema conducido por eventos

Donde los de vista estática son sobre la información almacenada y los dinámicos sobre las operaciones:

VISTA ESTÁTICA	VISTA DINÁMICA
Diagramas de clases.	Diagrama de secuencia.
Diagrama de objetos.	Diagramas de colaboración.
	Diagramas de actividad.
	Diagrama de estados.


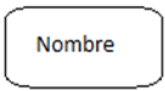


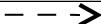
2 ELEMENTOS DEL DIAGRAMA

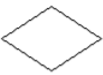
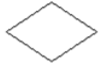


Los diagramas de actividades son parecidos a los ordinogramas, con la ventaja de que permiten expresar concurrencia mientras que los diagramas de flujo no.

Los diagramas de actividad están formados por una serie de figuras conectadas por flechas. Las figuras más importantes son:

- **Círculo negro:** representa el inicio del flujo o actividad inicial.
- **Un círculo negro rodeado de una circunferencia:** representa el final.
- **Rectángulos redondeados:** representan una actividad. Algunos autores las actividades les llaman estados, pero usaremos la notación de actividad para no confundirlo con los diagramas de estados. Las actividades se expresan mediante verbos.
- **Objetos:** Los objetos se representan mediante rectángulos. Y son datos de entrada o salida de actividades. Se expresan mediante sustantivos.
- **Rombos:** Representan puntos donde el flujo de control se ramifica en base a alguna condición (*branch*). También se usan para “desramificar” reagrupar (*merge*) flujos previamente ramificados. Los rombos también permiten unir flujos que proceden de varias actividades. Las condiciones de los rombos se ponen en las fechas de salida.
- **Condiciones de guardia:** En las ramas se ponen las condiciones de guardia. Suelen ponerse en los flujos que salen de los rombos o flujos de control.
- **Barras horizontales:** sirven para sincronizar el inicio o el fin de actividades (*threads*) concurrentes.
- **Flechas:** las flechas se trazan desde unas figuras a otras para indicar el orden en que se suceden las actividades.

Se observan en la siguiente tabla:

Símbolo	Nombre	Descripción
	Nodo inicial	Muestra el punto de partida del flujo de acciones.
	Acción	Representa una actividad o acción. El nombre generalmente comienza con un verbo.
	Flujo o transición	Muestra el orden de ejecución de las actividades.
	Nodo final	El final de todos los flujos de acciones en el diagrama.
	Flujo de datos	Línea punteada con la flecha abierta

Símbolo	Nombre	Descripción
	Decisión	Representa un punto en el flujo donde deben tomarse una decisión para saber con que actividad continuar. De un rombo de decisión salen dos o mas flujos.
	Unión (Merge)	A este punto llegan uno o mas líneas y sale una sola. El proceso continua cuando cualquiera de los flujos llega a este punto.
	(Fork), sincronización o concurrency	Es el comienzo de varias actividades que se realizan en paralelo. De la barra salen varias líneas.
	(Join), sincronización o concurrency	A esta barra llegan varias lineas y sale una sola. Indica que deben terminarse todas las actividades que llegan aquí para poder terminar.

3 CUANTOS DIAGRAMAS REALIZO.

Se realizarán tantos diagramas como sean necesarios. Lo importante es que quepa en una hoja.

Es recomendable también poner un título al diagrama para aclarar lo que se explica dentro.

4 NODO INICIAL Y FINAL.

El nodo inicial representa el inicio del diagrama de actividad. El nodo final que representa el final. Tanto el nodo inicial como el final pueden haber más de uno en un diagrama normal.

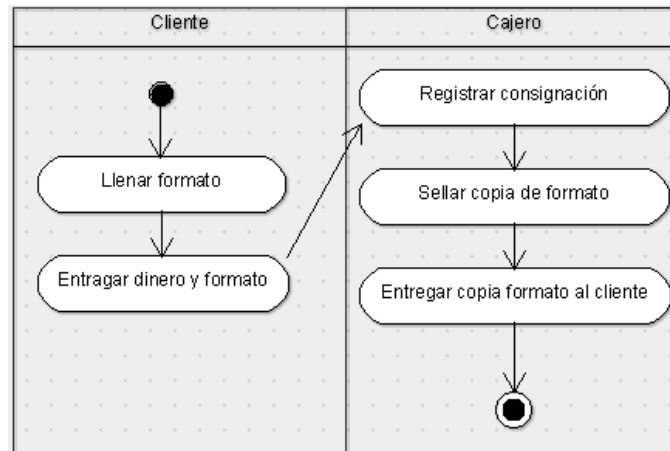
5 CARRILES

A veces puede ser útil organizar las actividades del modelo según la responsabilidad ("quién las realiza"). Para ello dividimos el diagrama en particiones mediante líneas verticales.

El concepto de quien realiza una actividad en un diagrama de casos de uso era el actor, es por ello que debe ser diagramas que deben coincidir los nombres de las calles con los actores. Es importante no confundir una calle con el sistema que estamos diseñando.

Los carriles agrupan acciones que realiza un responsable dentro de una organización. Un carril tiene un nombre único en el diagrama. Cada actividad está

en un único carril. Las transiciones y las barras concurrentes pueden cruzar líneas de los carriles.



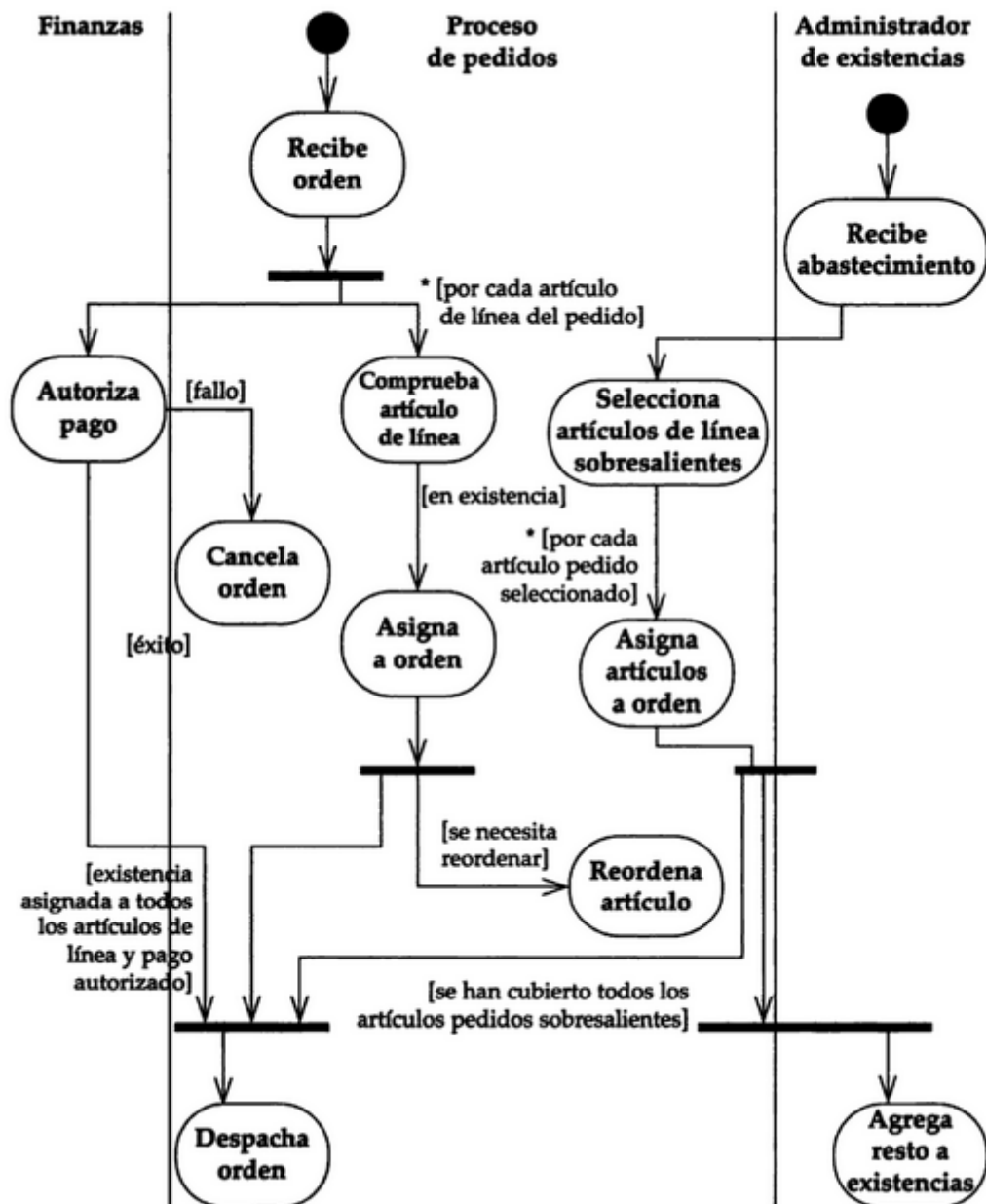
Por el aspecto del diagrama a las particiones también se les puede llamar calles o carriles.

5.1 Ejemplo. Proceso de pedido.

Fuente: UML gota a gota.

Los carriles representan la responsabilidad de cada departamento. Comprobar que el carril central es el proceso de pedidos es decir el sistema que estamos programando.

En el siguiente diagrama se puede apreciar que no hay rombos de toma de decisiones y debería de tenerlos. Por eso pone [Por cada artículo de la línea de pedido]* para indicar un bucle o conjunto de elementos procesados.



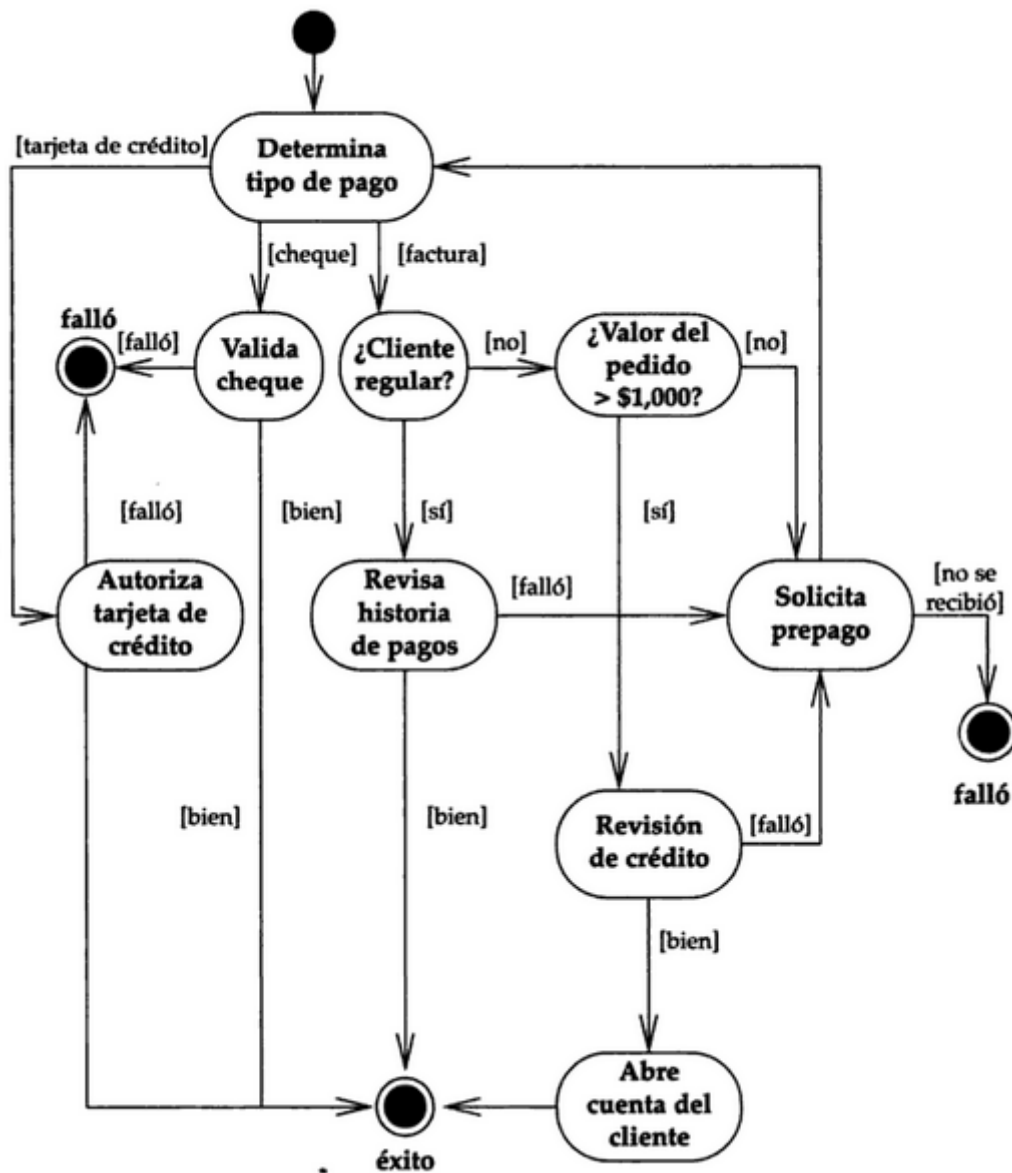
6 DESCOMPOSICIÓN DE UNA ACTIVIDAD

Una actividad puede descomponerse en un diagrama nuevo.

6.1 Ejemplo: Determinar el pago.

Fuente: UML gota a gota.

Cuando se crea un diagrama que es una descomposición de una actividad, de debe poner un sólo punto de partida (o símbolo círculo todo negro).



7 OBJETOS

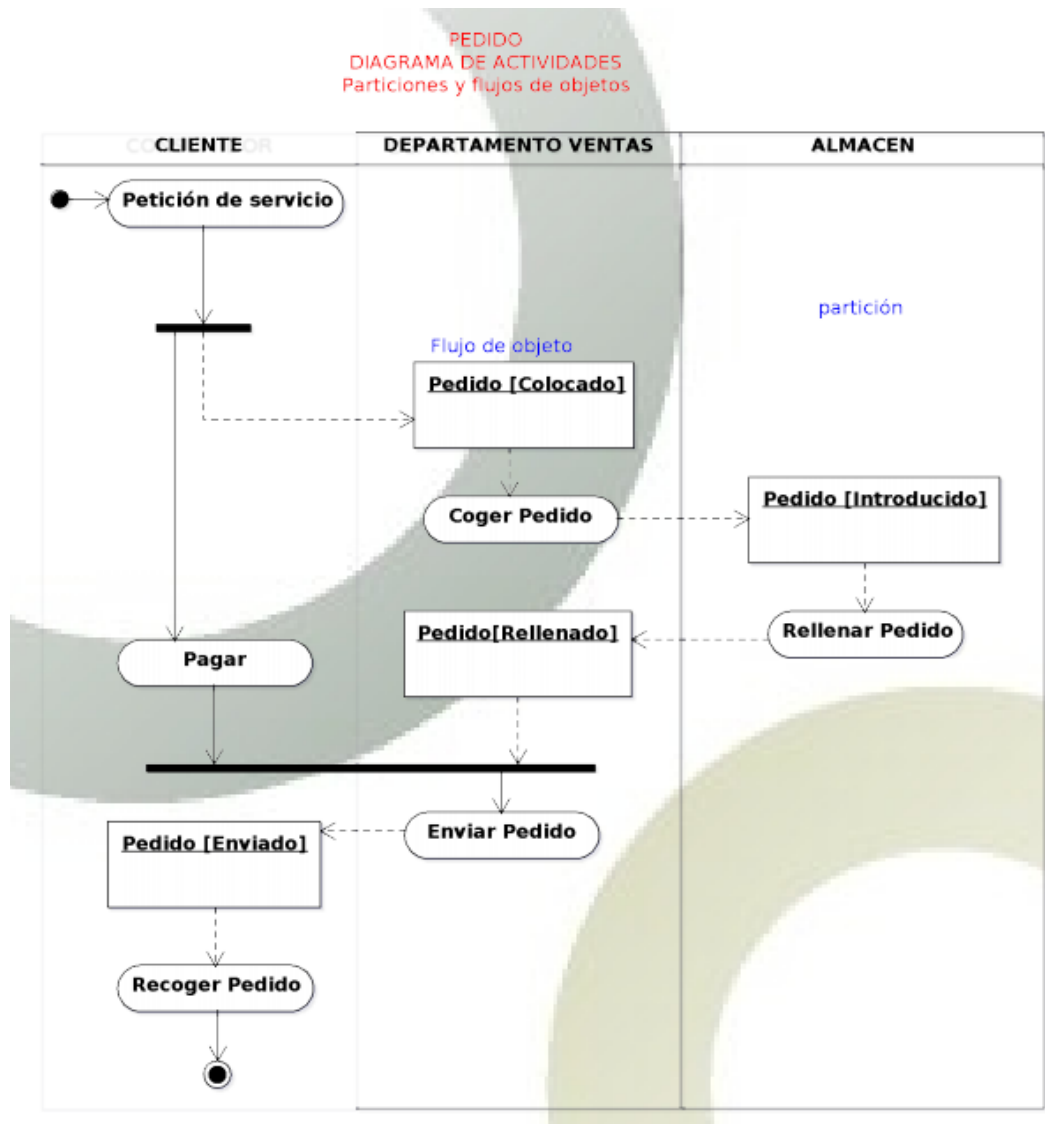
Los objetos se dibujan como rectángulos (sin redondear) y representan resultados o valores de entrada a alguna actividad. Cuando representan el resultado se dibuja una flecha desde la actividad al objeto, mientras que si representan los datos de entrada para alguna actividad, la flecha va desde el objeto a la actividad.

Los objetos se asocian a clases y se pone entre corchetes la propiedad o estado del objeto.

Veámoslo en el siguiente ejemplo.

Ejemplo: Pedido

Muestra el flujo de actividades en un almacén desde que llega un pedido hasta que se recoge. Los responsables involucrados son el cliente, el departamento de ventas y el almacén.



8 ACTIVIDAD

En los diagramas de actividad el símbolo clave es la actividad que se representa mediante **rectángulos redondeados**. La actividad tiene varias interpretaciones:

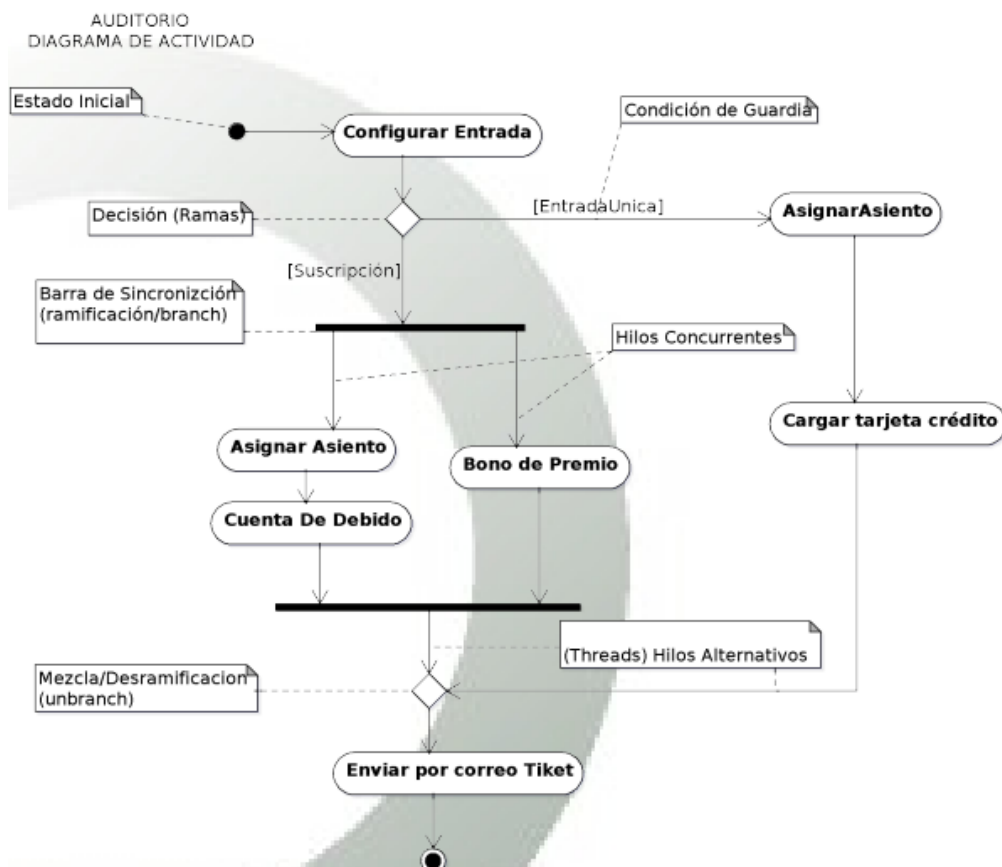
- En un diagrama conceptual una actividad es cierta tarea que debe ser llevada a cabo, ya sea por un ser humano o un ordenador.

- En un diagrama de implementación, es una instrucción de un método sobre una clase.

Veámoslo con ejemplos.

8.1 Ejemplo: Auditorio

El siguiente diagrama modela el proceso de pedidos de entradas de un auditorio. Las entradas pueden ser de dos tipos: única o abono. En caso de entrada única se asignan los asientos y se cobra por tarjeta de crédito. Si es un abono se concede una bonificación (*bonus*) al tiempo que se asignan los asientos y se carga en cuenta.

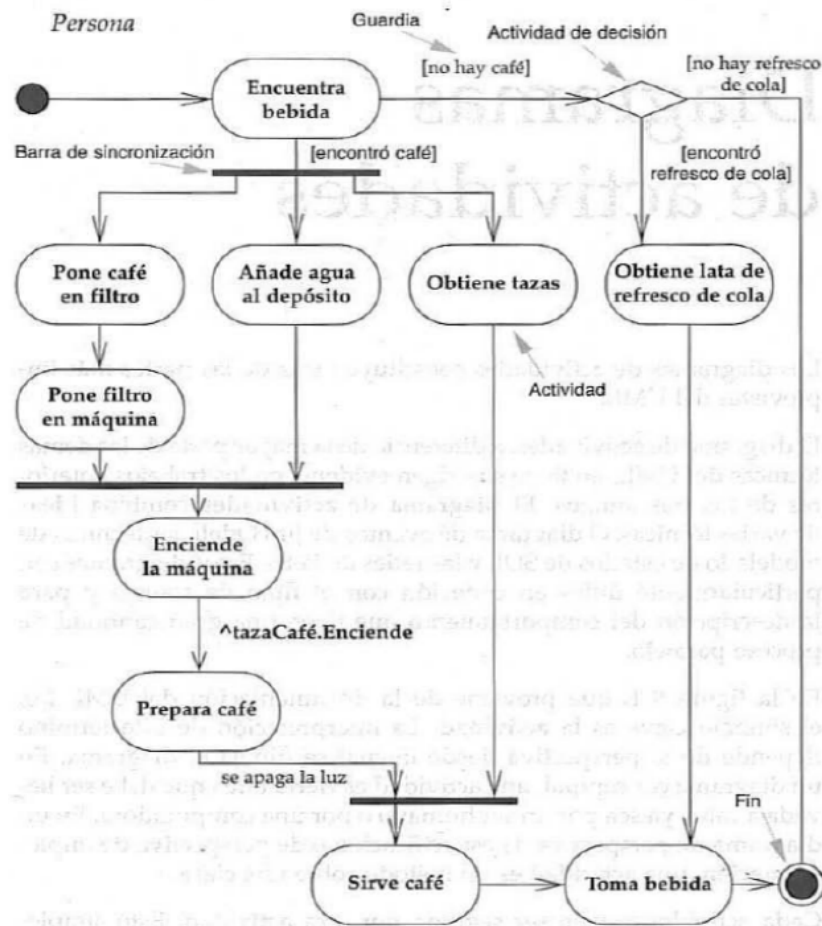


8.2 Ejemplo: Bebida

El siguiente ejemplo muestra el diagrama de actividad del proceso: “Tomar Bebida”.

En el diagrama cada actividad puede ser seguida por otra actividad. Esto simplemente es secuenciación. Por ejemplo, la actividad “Poner el café en el filtro” va seguida por la actividad “Poner el filtro en la máquina”.

Hasta ahora, el diagrama de actividades parece un *flowchart* o diagrama de flujo. Podemos investigar las diferencias observando la actividad “Encuentra bebida”. De “Encuentra bebida” salen dos disparadores. Cada disparador tiene un guardia, una expresión lógica que se evalúa como “verdadero” o “falso”. La persona seguirá la actividad “Encuentra bebida” considerando las opciones de café o refresco de cola.



Supongamos que podremos disfrutar de una bolsa de café de Colombia y seguiremos la ruta del café. Este disparador nos conduce a la barra de sincronización, a la cual están unidos otros tres disparadores que, a su vez, conducen a las actividades “Poner café en filtro”, “Añade agua al depósito” y “Obtiene tazas”.

El diagrama señala que estas actividades pueden suceder en paralelo, es decir, su orden no es significativo. Se podría poner el café en el filtro primero, después añadir agua al depósito y, por último, obtener las tazas. También podríamos conseguir las tazas, después poner el café en el filtro... ya se entiende la idea.

También puedo llevar a cabo estas actividades en forma intercalada. Podría obtener una taza, añadir después un poco de agua al depósito, obtener otra taza, agregar otro poco más de agua y así sucesivamente. O también podría llevar a cabo estas operaciones simultáneamente: agregar el agua al depósito con una mano y con la otra alcanzar la taza. De acuerdo con el diagrama, cualquiera de estas formas de operar es correcta.

El diagrama de actividades me permite seleccionar el orden en que se harán las cosas. Esto es, simplemente me dice las **reglas esenciales de secuenciación** que tengo que seguir. Esta es la **diferencia clave entre un diagrama de actividades y un diagrama de flujo**. Los diagramas de flujo se limitan normalmente a procesos secuenciales; los diagramas de actividades pueden manejar procesos paralelos.

Esta característica es importante para el modelado de negocios. Los negocios con frecuencia tienen procesos secuenciales innecesarios. Una técnica como ésta, que promueve el **comportamiento paralelo**, es valiosa en estas situaciones, porque

auspicio que las personas se aparten de las secuencias innecesarias en su comportamiento y descubran oportunidades para hacer cosas en paralelo. Esto puede mejorar la eficiencia y capacidad de respuesta de los procesos del negocio.

Los diagramas de actividades también son útiles para los **programas concurrentes**, ya que se pueden plantear gráficamente cuáles son los hilos y cuándo necesitan sincronizarse.

Cuando se tiene un comportamiento paralelo, se impone la necesidad de sincronizar. No queremos prender la cafetera hasta haberle colocado el filtro y llenado de agua el depósito. Por eso, vemos a los disparadores de estas actividades saliendo juntos de una barra sincronizadora. Una barra de sincronización simple como ésta indica que el disparo de salida ocurre sólo cuando han sucedido ambos disparos de entrada. Como veremos después, estas barras pueden ser más complicadas.

Más adelante se dará otra sincronización: el café debe ser preparado y las tazas deben estar disponibles, antes de poder servir el café.

Vayamos ahora al otro carril. En este caso, tenemos una decisión compuesta. La primera decisión es sobre el café. Esta decisión es la que determina los dos disparadores que salen de “Encuentra bebida”. Si no hay café, nos enfrentamos a una segunda decisión, basada ésta en refresco de cola.

Cuando tenemos decisiones como la anterior, señalamos la segunda decisión con un rombo de decisión. Esto nos permite describir decisiones anidadas, de las cuales podemos tener cualquier cantidad.

En la actividad “Toma bebida” convergen dos disparadores, lo que significa que se llevará a cabo en cualquiera de los dos casos. Por el momento, se puede considerar esto como un caso OR (lo hago, si sucede uno u otro disparador) y a la barra de sincronización como el caso AND (lo hago, si suceden ambos disparadores).

9 CONCURRENCIA

La concurrencia es una línea que permite indicar que dos actividades deben terminarse para empezar otra, o para que puedan hacerse varias actividades al mismo tiempo a partir de otra.

No hay que confundir la selección múltiple que expresa un diagrama de caso de uso con extend como la siguiente imagen:

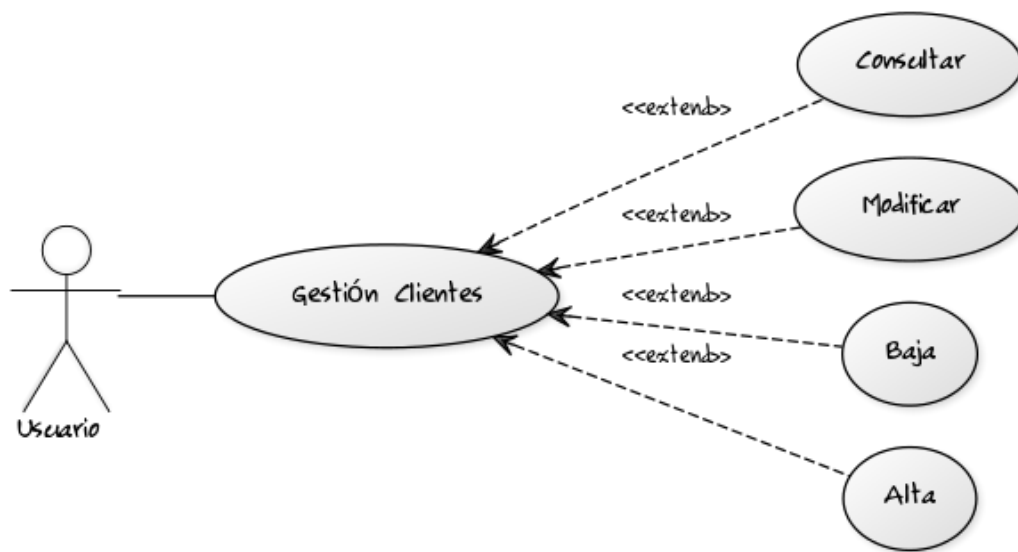


Imagen 1: Caso uso con selección múltiple o extend.

El diagrama de actividad incorrecto, que indica que se pueda hacer una actividad u otra, pero no todas a la vez, es la siguiente imagen:

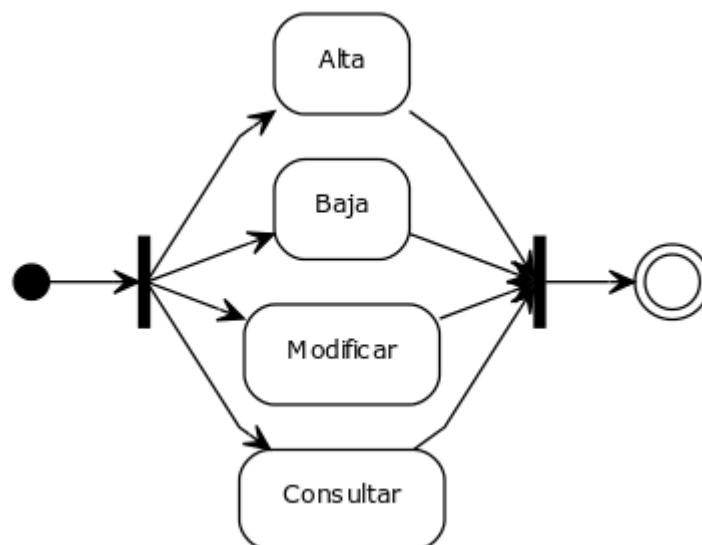


Imagen 2: Diagrama Actividad Incorrecto

El diagrama de actividad CORRECTO, que indica que se pueda hacer una actividad u otra, pero no todas a la vez, es la siguiente imagen:

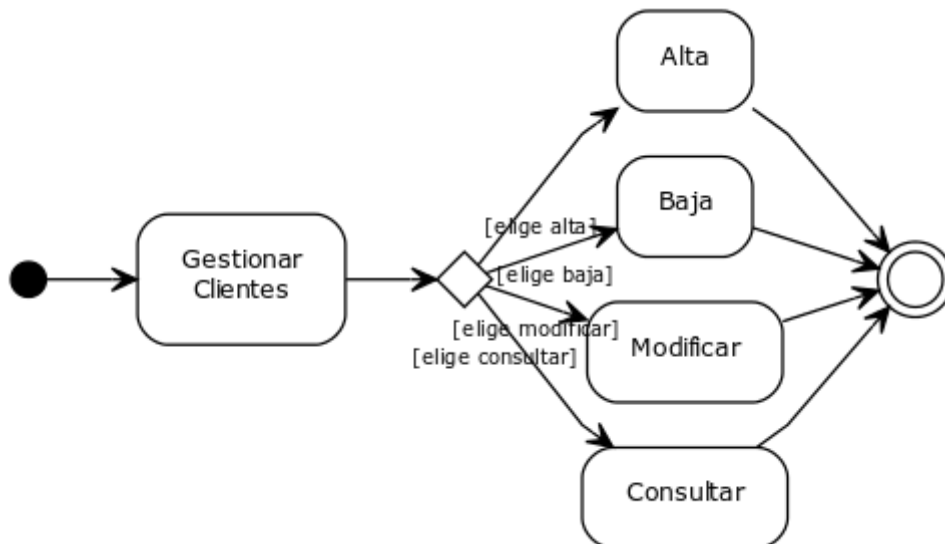


Imagen 3: Diagrama actividad correcto

9.1 Ejemplo: Fibonacci

La serie de Fibonacci la escribió un matemático italiano hace 800 años, está formada por los números: 1,1,2,3,5,8,13,

Cada número es un fib, así que el primer fib es 1, lo expresaremos por fib(1)=1, fib(2) es 1, etc. La regla de cada fib, excepto en los dos primeros, es la suma del anterior par de fibs, por ejemplo, fib(3)=fib(2)+fib(1)=1+1=2.

Supongamos que una de sus clases es una calculadora, y que una de sus operaciones fuese la de calcular el enésimo fib y mostrarlo. A esta operación podría llamarla calcularFib(n). Crearemos un diagrama de actividades que modele a esta operación.

Necesitaremos algunas variables como son: un contador para llevar un control para verificar si se ha llegado al enésimo fib, una variable para acumular los resultados, y dos más para almacenar dos fibs que tendrá que sumar entre sí.

En Java sería:

```

public class calculadora {
    public static void mostrar ( int Respuesta , int Contador ) {
        System . out . print ( " El " + Contador + " Fib es: " + Respuesta );
    }
    public static void calcularFib ( int n ) {
        int Respuesta =0;
        int Respuesta1 =1;
        int Respuesta2 =0;
        int Contador =1;
        if ( n ==1 ) {
            Respuesta = Respuesta1 ; // Fib (1)=1
            mostrar ( Respuesta , Contador );
        }
        if ( n >1 ) {
            Respuesta2 =1;
            Contador =2;
            if ( n ==2 ) { // Fib (2) = 1
                Respuesta = Respuesta2 ;
            }
        }
    }
}
  
```

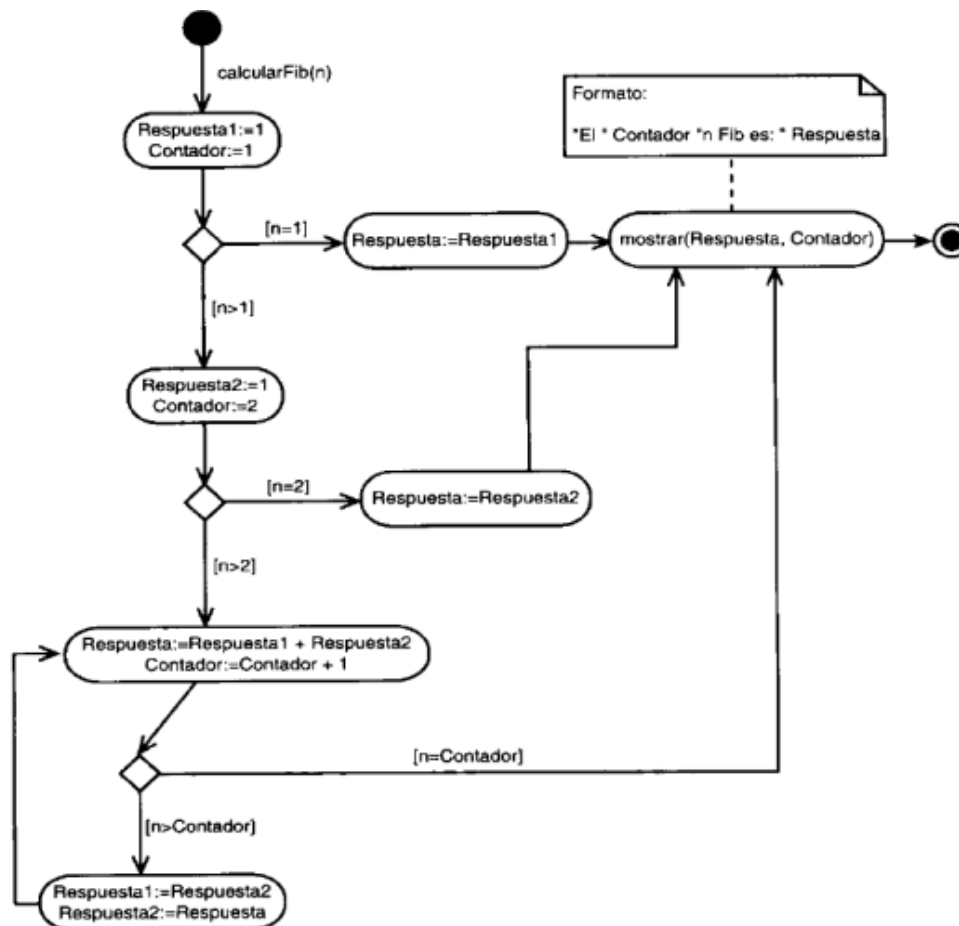
```

        mostrar ( Respuesta , Contador );
    }
}
if (n >2) { // Fib (n)= Fib (n -2)+ Fib (n -1)
    while ( n> Contador ) {
        Respuesta = Respuesta1 + Respuesta2 ;
        Contador = Contador +1;
        if ( n== Contador ) {
            mostrar ( Respuesta , Contador );
        } else { // n> Contador
            Respuesta1 = Respuesta2 ;
            Respuesta2 = Respuesta ;
        }
    }
} // while
}
System . out . println ("");
}
public static void main ( String args []) {
    calcularFib (1);
    calcularFib (2);
    calcularFib (3);
    calcularFib (4);
}
}
/* EJECUCION :
El 1 Fib es: 1
El 2 Fib es: 1
El 3 Fib es: 2
El 4 Fib es: 3
*/

```

[calculadora.java]

Y el diagrama sería:



10 HERRAMIENTAS GRÁFICAS.

Las herramientas gráficas para elaborar diagramas UML son:

- ArgoUml. Software Libre. Soporta carriles.
- Dia. No soporta carriles. Software Libre. No tiene carriles.
- Visual Paradigm. Software Propietario.
- OpenOffice Draw. Software Libre
- Yuml. Web. No tiene Carriles.

11 BIBLIOGRAFÍA Y ENLACES

1. Amescua, A.; Cuadrado, JJ; Ernica, E. (2003): *Análisis y Diseño Estructurado y Orientado a Objetos de Sistemas Informáticos*, Mcgraw-hill
2. Booch, G.; Rumbaugh, J.; Jacobson, I. (1998): *The Unified Modeling Language User Guide*, Addison Wesley
3. Booch, G.; Rumbaugh, J.; Jacobson, I. (2005): *UML. Reference Manual*, Addison Wesley
4. Casado, C. (2012): *Entornos de desarrollo*, RA-MA, Madrid

5. Fowler, M. (1999): *UML gota a gota*, Addison Wesley
6. Jesse, M.; Schardt, J.(2003): *UML 2 for dummies*, Wiley Publishing
7. Larman, C. (1999): *UML y patrones. Introducción al análisis y diseño orientado a objetos*, Prentice Hall
8. Martin, R. (2002): *UML for Java Programmers*, Prentice Hall
9. Pilone, D. (2006): *UML 2.0 Pocket Reference*, O'Really Media Edition
10. Schmuller, J. (2000): *Aprendiendo UML en 24 horas*, Pearson Educación, Madrid