
14.1. Presentación del capítulo

Los diagramas de actividad son "diagramas de flujo orientados a objetos". Le permiten modelar un proceso como una actividad que consta de una colección de nodos conectados por extremos. UML 2 introduce una nueva semántica para diagramas de actividad que les proporciona mucho más poder y flexibilidad de lo que jamás hayan tenido. En este capítulo tratamos los diagramas básicos de actividad; esto puede ser todo lo que necesite para la mayoría de su modelado de actividad. Para completar la información, presentamos temas más avanzados en el siguiente capítulo.

14.2. ¿Qué son diagramas de actividad?

Los diagramas de actividad a menudo se denominan "diagramas de flujo orientados a objetos". Le permiten modelar un proceso como una actividad que consta de una colección de nodos conectados por extremos.

En UML 1, los diagramas de actividad eran casos especiales de máquinas de estado (véase el capítulo 21) donde cada estado tenía una acción de entrada que especificaba algún proceso o función que ocurrió cuando se incorporó el estado. En UML 2, los diagramas de actividad tienen una semántica completamente nueva basada en Petri Nets. Esto tiene dos ventajas:

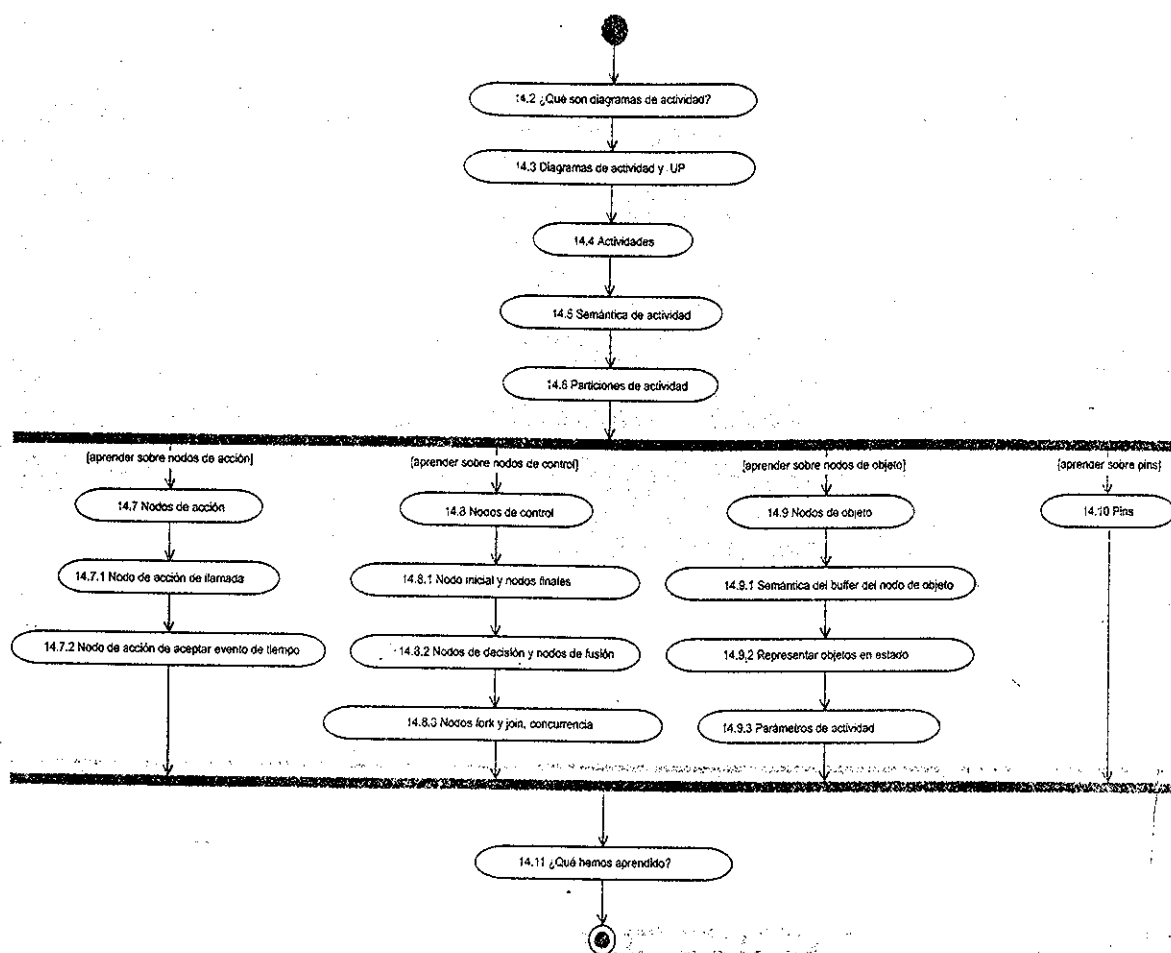


Figura 14.1.

1. El formalismo Petri Nets proporciona mayor flexibilidad en el modelado de diferentes tipos de flujo.
2. Existe una clara distinción en UML entre diagramas de actividad y máquinas de estado.

Una actividad se puede anexas a cualquier elemento de modelado con la finalidad de modelar su comportamiento. El elemento proporciona el contexto para la actividad, y la actividad puede hacer referencia a características de su contexto. Las actividades se anexas normalmente a:

- Casos de uso.
- Clases.
- Interfaces.
- Componentes.
- Colaboraciones.
- Operaciones.

También puede utilizar diagramas de actividad para modelar procesos de negocio y flujos de trabajo. Indicamos cómo puede hacer esto, pero puede ser un tema

muy complejo y está más allá del alcance de este libro. Aunque un uso común de los diagramas de actividad es dibujar diagramas de flujos de operaciones, merece la pena considerar que el código fuente para una operación, en código o pseudocódigo, podría ser su mejor y más concisa representación. Por lo tanto, juzgue cada caso según sus méritos.

La esencia de un buen diagrama de actividad es que está centrado en comunicar un aspecto específico de un comportamiento dinámico de un sistema. Como tal, debe estar en el nivel correcto de abstracción para comunicar ese mensaje a su audiencia objetivo y debería contener la cantidad mínima de información necesaria para que tenga sentido. Es muy fácil adornar diagramas de actividad con estados de objeto, flujos de objeto, etc., pero siempre debe preguntarse si estos adornos clarifican o complican el diagrama. Como siempre, es mejor mantenerlo lo más sencillo posible.

14.3. Diagramas de actividad y UP

Debido a su flexibilidad, no hay ningún lugar donde los diagramas de actividad encajen en el UP. Proporcionan un mecanismo de carácter general para modelar comportamientos y puede utilizarlos allí donde añadan valor. Nosotros los tratamos aquí en el workflow de análisis ya que ahí es donde tendemos a utilizarlos más.

La posibilidad única de los diagramas de actividad es que le permiten modelar un proceso sin tener que especificar la estructura estática de clases y objetos que realizan ese proceso. Claramente, esto es de mucha utilidad cuando se encuentra en los primeros niveles de análisis y está tratando de descubrir qué es un proceso determinado.

En nuestra experiencia, los diagramas de actividad se utilizan más comúnmente en las siguientes formas:

- En el workflow de análisis:

- Para modelar el flujo en un caso de uso de una forma gráfica que es fácil de entender para los grupos de decisión.
- Para modelar el flujo entre casos de uso. Esto utiliza una forma especial de diagrama de actividad denominado diagrama de visión de interacción (véase el capítulo 15).

- En diseño:

- Para modelar los detalles de una operación.
- Para modelar los detalles de un algoritmo.

- En el modelado de negocio:

- Para modelar un proceso de negocio.

Los diagramas de actividad son fáciles de entender para los grupos de decisión. Esto es porque la mayoría de los grupos de decisión han tenido algún tipo de relación con los diagramas de flujo de alguna forma. Los diagramas de actividad pueden ser un estupendo mecanismo de comunicación siempre y cuando los mantenga sencillos.

Como verá en este capítulo y en el siguiente, UML 2 presenta numerosa sintaxis y semántica nueva para diagramas de actividad y es importante no emocionarse con ello. Cuando construya cualquier diagrama UML, siempre recuerde su audiencia objetivo y utilice características UML en consecuencia. No tiene sentido utilizar todas las características más novedosas si nadie entiende el diagrama.

14.4. Actividades

Las actividades son redes de nodos conectados por extremos. Existen tres categorías de nodo:

1. Nodos de acción: Representan unidades de trabajo que son atómicas dentro de la actividad.
2. Nodos de control: Controlan el flujo por medio de la actividad.
3. Nodos de objeto: Representan objetos utilizados en la actividad.

* Los extremos representan flujo por la actividad. Existen dos categorías de extremo:

1. Flujos de control: Representan el flujo de control por medio de la actividad.
2. Flujos de objeto: Representan el flujo de objetos por medio de la actividad.

Examinamos cada uno de los tipos de nodo y extremo en detalle en los siguientes apartados.

Examinemos un ejemplo. La figura 14.2 muestra un sencillo diagrama de actividad para el proceso de negocio Enviar carta. Observe que las actividades pueden tener precondiciones y postcondiciones como los casos de uso. Las precondiciones deben ser ciertas antes de que la actividad pueda empezar y las postcondiciones serán ciertas después de que la actividad haya terminado. Las acciones dentro de la actividad pueden tener también sus propias precondiciones y postcondiciones locales, según se ilustra:

Las actividades empiezan a menudo con un solo nodo de control, el nodo inicial, que indica el lugar donde empezará la ejecución cuando se invoca la actividad. Uno o más nodos finales indican los lugares donde termina la actividad.

En el ejemplo de la figura 14.2, la actividad empieza en el nodo inicial y luego el control pasa al nodo de acción Escribir carta vía un extremo. Este nodo indica un comportamiento que es atómico en lo que se refiere a la actividad que lo contiene. El flujo progresa hacia Escribir dirección y Enviar carta y luego al nodo final donde termina la actividad.

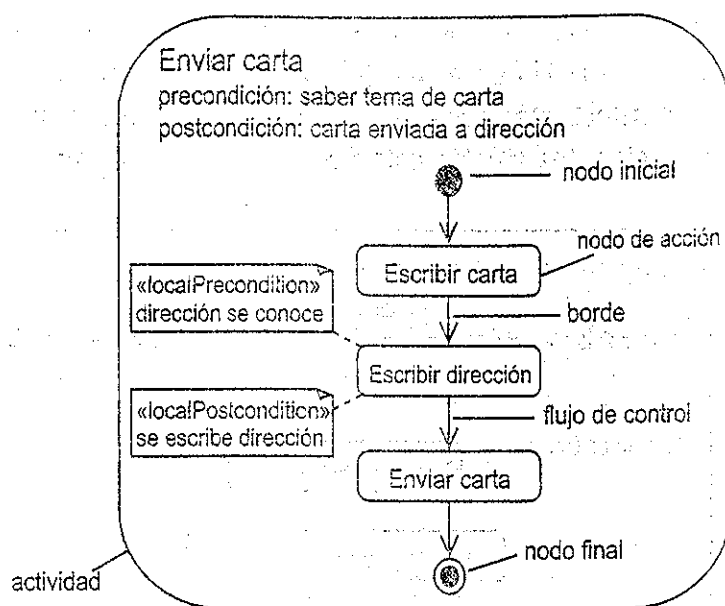


Figura 14.2.

Un uso común de los diagramas de actividad es modelar un caso de uso como una serie de acciones. La figura 14.3 muestra el caso de uso PagarImpuestoVentas del capítulo 4. Este caso de uso se puede expresar como un diagrama de actividad según se muestra en la figura 14.4.

Caso de uso: PagarImpuestoVentas	
ID: 1	
Breve descripción:	Pagar Impuesto Ventas al Organismo correspondiente al final del trimestre
Actores principales:	Tiempo
Actores secundarios:	Organismo
Precondiciones:	1. Es el final del trimestre
Flujo principal:	1. El caso de uso empieza cuando es el final del trimestre. 2. El sistema determina la cantidad de Impuesto Ventas que se debe pagar al Organismo correspondiente. 3. El sistema envía un pago electrónico al Organismo correspondiente.
Postcondiciones:	1. El Organismo recibe la cantidad correcta de Impuesto Ventas
Flujos alternativos:	Ninguno.

Figura 14.3.

Observe que el diagrama de actividad le proporciona una forma mucho más compacta y gráfica del caso de uso. El diagrama de actividad expresa el caso de uso como dos acciones, Calcular impuesto ventas y Enviar pago electrónico.

Cada una de estas acciones se podría expresar como un diagrama de actividad y esto probablemente ocurriría en el workflow de diseño cuando necesite descubrir cómo están implementadas las acciones. El actor y su interacción con el sistema son características estructurales y éstas están ausentes del diagrama.

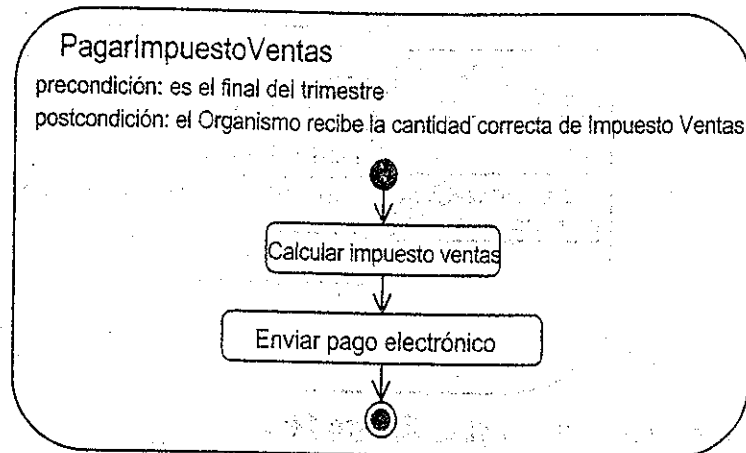


Figura 14.4.

Los casos de uso expresan comportamiento del sistema como una interacción entre un actor y el sistema, mientras que los diagramas de actividad lo expresan como una serie de acciones. Son vistas complementarias del mismo comportamiento.

14.5. Semántica de actividad

Los diagramas de actividad tienen una semántica bastante intuitiva, como podrá haber observado por los apartados anteriores. En este apartado examinamos la semántica de actividad en profundidad. Los diagramas de actividad de UML 2 están basados en Petri Nets. Petri Nets está fuera del alcance de este libro, pero puede saber más de ello en www.daimi.au.dk/PetriNets/ si está interesado.

Los diagramas de actividad modelan comportamiento al utilizar el juego *token*. Este juego describe el flujo de tokens alrededor de una red de nodos y extremos de acuerdo a reglas específicas. Los tokens en los diagramas de actividad UML pueden representar:

- El flujo de control.
- Un objeto.
- Algunos datos.

El estado del sistema en cualquier punto en el tiempo está determinado por la disposición de sus tokens.

En el ejemplo de la figura 14.2, el token es el flujo de control ya que no existen objetos o datos que se pasan entre nodos en este caso en particular.

Los tokens se pasan de un nodo origen a un nodo destino a través de un extremo. El movimiento de un token está sujeto a condiciones y solamente puede ocurrir cuando todas las condiciones se cumplen. Las condiciones varían dependiendo del tipo de nodo. Para los nodos en la figura 14.5 (nodos de acción), estas condiciones son:

- Las postcondiciones del nodo origen.
- Condiciones de protección en el extremo.
- Las precondiciones del nodo destino.

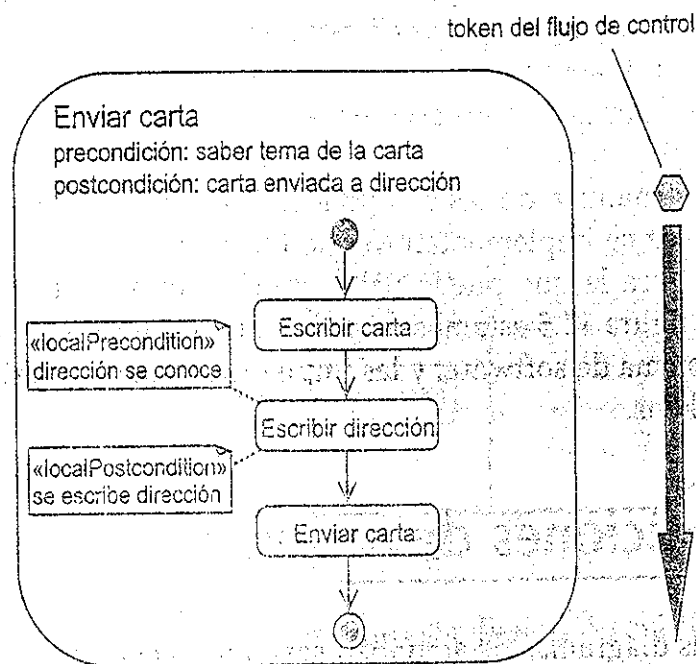


Figura 14.5.

Al igual que nodos de acción, existen nodos de control y nodos de objeto. Los nodos de control tienen semántica especial que controla cómo los tokens se pasan de sus extremos de entrada a los extremos de salida. Por ejemplo, el nodo inicial empieza una actividad, el nodo final termina una actividad y un nodo join ofrecerá un token en su extremo de salida si, y sólo si, existen tokens en todos sus extremos de entrada. Los nodos de objeto representan objetos que fluyen por el sistema. Tratamos los nodos de control y los nodos de objeto en más detalle más adelante en este capítulo.

Considere cómo este juego funciona para la actividad que se ilustra en la figura 14.5. Cuando se ejecuta la actividad, un flujo de token de control empieza en el nodo inicial. No existen condiciones en este nodo, su extremo de salida, o el nodo destino, y por lo tanto el token atraviesa automáticamente el extremo de salida hasta el nodo destino, Escribir carta. Esto hace que la acción especificada por el nodo de acción Escribir carta se ejecute. Cuando Escribir carta ha terminado, el flujo del token de control se desplaza hasta el nodo de acción Escribir dirección si, y sólo si, su precondición, dirección es conocida, se cumple. Cuando la postcondición, carta es dirigida, se cumple, el control fluye de

Escribir dirección a Enviar carta. Por último, puesto que no existen condiciones que impidan que el flujo salga de Enviar carta, el flujo de control se mueve al último estado y la actividad termina.

En este sencillo ejemplo, el flujo de control pasa por cada nodo de acción en turno haciendo que se ejecute. Ésta es la semántica principal de la actividad.

Como hemos mencionado, el estado del sistema que se ejecuta se puede representar en cualquier punto en el tiempo por la disposición de sus tokens. Por ejemplo, cuando el token está en el nodo de acción Escribir carta, puede decir que el sistema está en el estado Escribir carta. Sin embargo, no toda ejecución de acción o flujo de token constituye un cambio notable en el estado del sistema desde el punto de vista de sus máquinas de estado (véase el capítulo 21). No obstante, la disposición de los tokens proporciona un vínculo entre los diagramas de actividad y las máquinas de estado. Debe asegurarse de que los diagramas de actividad y las máquinas de estado para un elemento de modelo en particular son coherentes entre sí. Aunque la semántica de las actividades UML 2 se describen con el juego del token, casi nunca se implementan de esa forma. De hecho, una actividad es una especificación para la que puede haber muchas implementaciones posibles. Por ejemplo, en la figura 14.5 estamos describiendo un sencillo proceso de negocio en lugar de un sistema de software, y las implementaciones de este proceso no implicarán pasar tokens.

14.6. Particiones de actividad

Para que sus diagramas de actividad sean más fáciles de leer, puede dividir las actividades en particiones al utilizar líneas verticales, horizontales o curvas. Cada partición de actividad representa una agrupación de alto nivel de acciones relacionadas. Las particiones de actividad a veces se denominan carriles. Realizar particiones es una técnica potente; cuando se utiliza bien, puede hacer que los diagramas de actividad sean mucho más fáciles de entender.

En UML 2, el modelador define la semántica de las particiones de actividad; no tienen semántica heredada. Por lo tanto, puede utilizarlas para dividir diagramas de actividad de la forma que quiera. Las particiones de actividad se utilizan comúnmente para representar:

- Casos de uso.
- Clases.
- Componentes.
- Unidades organizativas (el modelado de negocio).
- Roles (en modelado de workflow).

Sin embargo, éstas no son las únicas posibilidades. Por ejemplo, en los modelos de diseño para sistemas distribuidos puede incluso utilizar particiones de activi-

dad para modelar la distribución de procesos en máquinas físicas. Cada conjunto de particiones debería tener una única dimensión que describa la semántica base del conjunto. Dentro de esta dimensión, las particiones pueden estar jerárquicamente anidadas. La figura 14.6 muestra una actividad que tiene un conjunto jerárquicamente anidado de particiones de actividad.

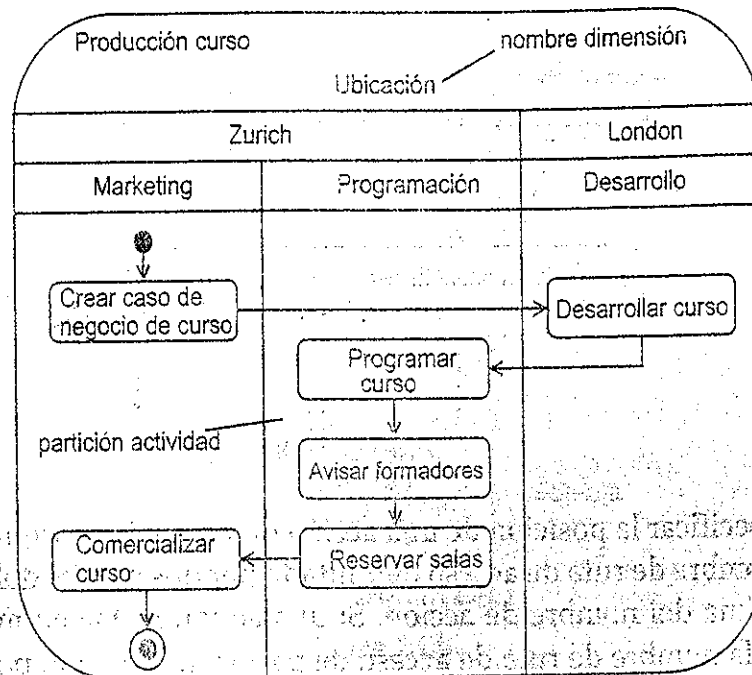


Figura 14.6.

La dimensión es Ubicación y dentro de esta dimensión existe una jerarquía de particiones según se muestra en la figura 14.7. Este diagrama modela el proceso de negocio de generación de curso en nuestra empresa asociada Zuhlke Engineering AG. Muchos de sus cursos los desarrollamos nosotros en Londres.

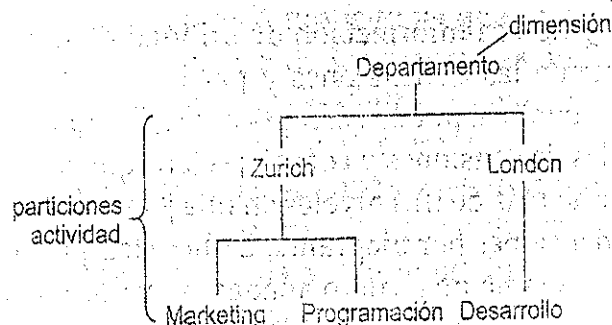


Figura 14.7.

A menudo existe una conexión entre las particiones de actividad y los flujos concurrentes de control. Tratamos la concurrencia de modelado en diagramas de actividad más adelante en este libro. Por ejemplo, es común que departamentos diferentes o unidades de negocio realicen líneas concurrentes de trabajo y luego se sincronicen en algún punto. Los diagramas de actividad con particiones de actividad son una forma excelente de modelar esto.

Algunas veces no es posible organizar los nodos en particiones verticales u horizontales sin dificultar la lectura del diagrama. En este caso podría utilizar líneas curvas para crear particiones irregulares, o podría indicar particiones al utilizar texto. UML tiene una notación textual para particiones de actividad como se ilustra en la figura 14.8. Sin embargo, normalmente solamente utilizará esta notación como último recurso porque la notación gráfica normalmente es mucho más clara.

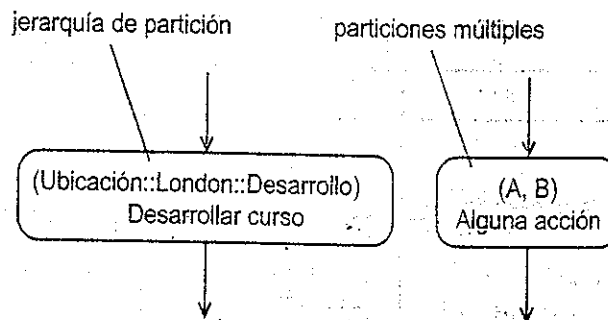


Figura 14.8.

Puede especificar la posición de una acción dentro de la jerarquía de partición al utilizar un nombre de ruta de acceso delimitado por dos puntos dobles entre paréntesis por encima del nombre de acción. Si una acción reside en más de una partición, liste cada nombre de ruta de acceso de partición separado por comas.

En raras ocasiones, puede necesitar mostrar comportamiento en un diagrama de actividad que está fuera del ámbito del sistema. Esto podría ser para mostrar cómo el sistema interactúa con algún otro sistema externo. Los diagramas de actividad dan cabida a esto; simplemente añada el estereotipo `<<external>>` directamente encima del nombre de partición. Observe que la partición externa no es una parte del sistema y por lo tanto, no se puede anidar dentro de ninguna jerarquía de partición del modelo.

Puede añadir numerosa información de utilidad a un diagrama de actividad al elegir cuidadosamente las dimensiones y particiones de actividad. Sin embargo, estas características también pueden complicar los diagramas, particularmente cuando existen múltiples dimensiones y complejas jerarquías de partición. En la práctica, trate de utilizar no más de tres niveles en una jerarquía (incluida la dimensión) y no más de dos dimensiones por diagrama. Utilice siempre su juicio y aplique particiones de actividad solamente cuando añadan un valor real al modelo.

14.7. Nodos de acción

Los nodos de acción se ejecutan cuando:

- Existe un token simultáneamente en cada uno de sus extremos de entrada AND.

- Los tokens de entrada satisfacen todas las precondiciones locales del nodo de acción.

Esto se ilustra en la figura 14.9.

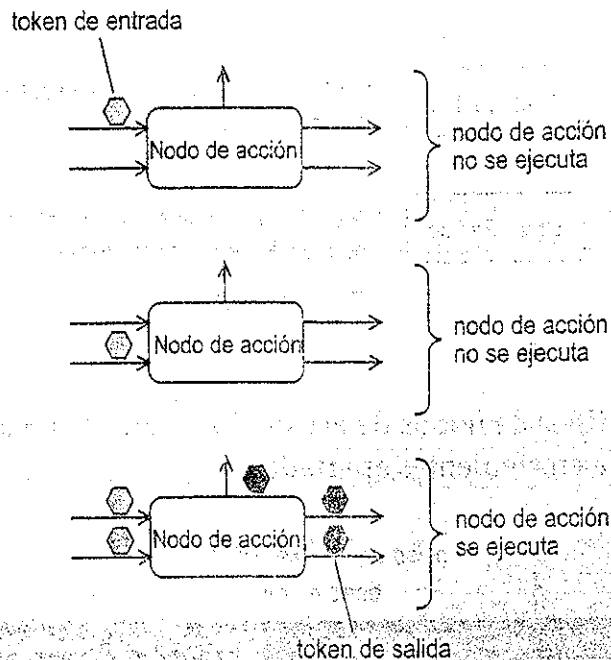


Figura 14.9.

Los nodos de acción realizan un AND lógico en sus tokens de entrada; el nodo no está listo para ejecutarse hasta que los tokens están presentes en todos sus extremos de entrada. Incluso cuando los tokens necesarios están presentes, el nodo solamente se ejecutará cuando su precondición local se cumple.

Cuando el nodo de acción ha terminado de ejecutarse, se comprueba la postcondición local. Si se cumple, el nodo ofrece simultáneamente tokens en todos sus extremos de salida. Esto es un *fork* implícito ya que un nodo de acción puede dar lugar a muchos flujos. A diferencia de los diagramas de flujo convencionales, los diagramas de actividad son concurrentes intrínsecamente.

Puesto que los nodos de acción realizan algo, normalmente se nombran con un verbo o frase verbal. La especificación UML no proporciona ninguna directriz sobre el nombrado de nodos de acción. La convención que utilizamos es nombrar el nodo empezando con una letra en mayúscula y continuando en minúscula, utilizando espacios donde sea apropiado. La única excepción a esta regla ocurre cuando un nodo de acción contiene una referencia a otro elemento de modelo. En este caso, siempre utilizamos el nombre del elemento de modelo como está sin cambiar la mayúscula o minúscula o añadir espacios. La figura 14.10 muestra dos ejemplos. El ejemplo en la parte superior hace referencia a algo denominado "pedido" mientras que el ejemplo en la parte inferior hace referencia explícitamente a una clase denominada `Pedido` que se puede encontrar en algún lugar en el modelo.

Los detalles de la acción se capturan en la especificación del nodo de acción. A menudo es simplemente una descripción de texto como "Escribir una carta", pero

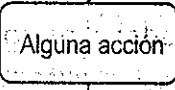
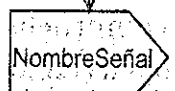
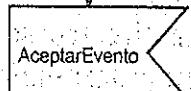
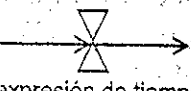
en diseño, podría ser texto estructurado, pseudocódigo o código. Si el diagrama de actividad está modelando un caso de uso, entonces podría ser uno o más pasos del flujo del caso de uso. Sin embargo, recuerde que esto puede crear un problema de mantenimiento porque tiene que mantener el caso de uso y el diagrama de actividad asociado al día si uno o el otro cambia.

Crear pedido	"pedido" hace referencia a un pedido de alguna clase
Crear Pedido	"Pedido" hace referencia a un elemento de modelo denominado Pedido

Figura 14.10.

Existen cuatro tipos de nodos de acción, los cuales se resumen en la tabla 14.1 y se tratan en detalle en siguientes apartados.

Tabla 14.1.

Símbolo	Nombre	Significado
	Nodo de acción de llamada.	Invoca una actividad, comportamiento u operación.
	Enviar señal.	Envía acción de señal: envía una señal asincrónicamente (el emisor no espera a la confirmación de la recepción de la señal). Puede aceptar parámetros de entrada para crear la señal.
	Nodo de acción de aceptar evento.	Acepta un evento: espera eventos detectados por el objeto que los posee y ofrece el evento en su extremo de salida. Se activa cuando obtiene un token en su extremo de entrada. Si no existe extremo de entrada, empieza cuando la actividad que lo contiene se inicia y siempre está activado.
 expresión de tiempo	Nodo de acción de aceptar evento de tiempo.	Acepta un evento de tiempo: responde a tiempo. Genera eventos de tiempo según su expresión de tiempo.

14.7.1. Nodo de acción de llamada

El tipo más común de nodo de acción es el nodo de acción de llamada. Este tipo de nodo puede invocar:

- Una actividad.
- Un comportamiento.
- Una operación.

Algunos ejemplos de la sintaxis de nodo de acción de llamada se ilustran en la figura 14.11. Como puede ver por la figura, la sintaxis del nodo de acción de llamada es muy flexible.

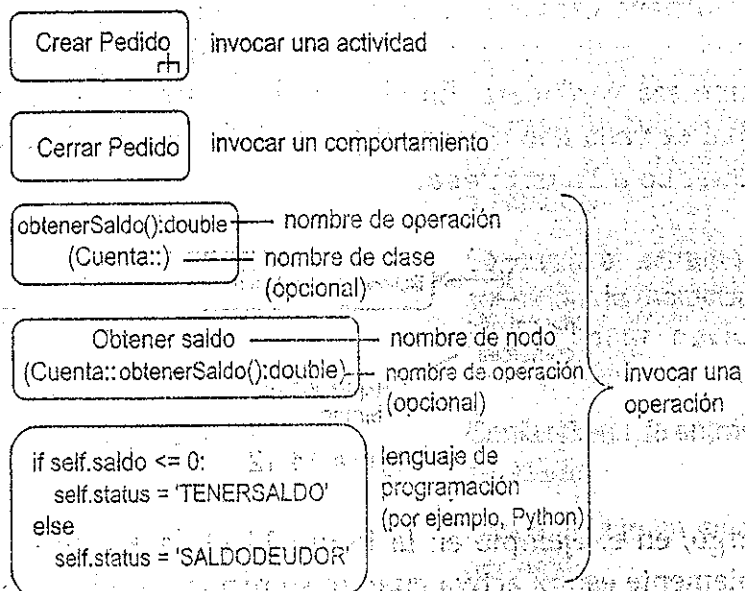


Figura 14.11. Sintaxis de los nodos de acción de llamada

- Puede indicar que la acción invoca a otra actividad al utilizar el símbolo especial de rastriillo en la esquina inferior derecha del icono de nodo. El nombre del nodo es el nombre de la actividad que invoca.
- Puede invocar un comportamiento; ésta es una invocación directa de un comportamiento del contexto de la actividad sin especificar ninguna operación en particular.
- Puede invocar una operación al utilizar la sintaxis estándar de operación descrita en el capítulo 7.
- Puede invocar una operación al especificar los detalles de la operación en un lenguaje determinado de programación. Esto puede ser particularmente de utilidad si utiliza una herramienta UML que le permite generar código de diagramas de actividad (por ejemplo iUML de Kennedy Carter, www.kc.com).
- Puede hacer referencia a características del contexto de la actividad al utilizar la palabra clave `self`.

Cuando utiliza nodos de acción de llamada en diagramas de actividad a nivel de análisis, normalmente invocará un comportamiento. Los nodos de acción de llamada que invocan operaciones específicas tienden a utilizarse para modelado detallado de actividad en diseño.

14.7.2. Nodo de acción de aceptar evento de tiempo

Un nodo de acción de aceptar evento de tiempo responde a tiempo. Este tipo de nodo tiene una expresión de tiempo y genera un evento de tiempo cuando esta expresión es verdadera. Este nodo se comporta de forma diferente dependiendo de si tiene o no un extremo de entrada.

Por ejemplo, la figura 14.12 muestra un nodo de acción de aceptar evento de tiempo que no tiene extremo de entrada. Cuando se activa la actividad que lo posee, este nodo se activará y generará un evento de tiempo siempre que su expresión de tiempo sea verdadera. En el ejemplo mostrado, se genera un evento de tiempo al final de cada año comercial y esto hace que se ejecute la actividad Devolver impuesto a la empresa.

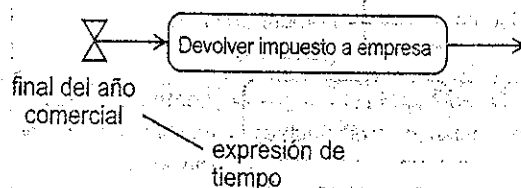


Figura 14.12.

Sin embargo, en el ejemplo en la figura 14.13, la acción tiene un extremo de entrada y solamente estará activa cuando se reciba un token en ese extremo. Este ejemplo es un fragmento de un sistema de ascensor.

La primera acción abre la puerta del ascensor y activa la acción de aceptar evento de tiempo. Esta acción espera diez segundos y luego ofrece un token a la acción Cerrar puerta.

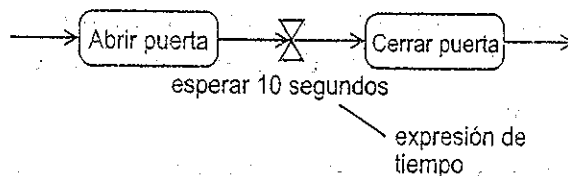


Figura 14.13.




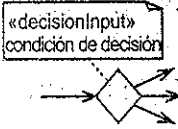


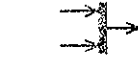
Observe que la expresión de tiempo puede hacer referencia a:

- Un evento en el tiempo (por ejemplo, final del año).
- Un punto en el tiempo (por ejemplo, el 11/03/1960).
- Una duración (por ejemplo, esperar 10 segundos).

14.8. Nodos de control

Los nodos de control gestionan el flujo de control dentro de una actividad. La tabla 14.2 resume todos los nodos de control de UML 2. Tratamos cada uno de estos nodos en detalle en los siguientes apartados.

Tabla 14.2.

Sintaxis	Nombre	Semántica
	Nodo inicial.	Indica dónde empieza el flujo cuando se invoca una actividad.
	Nodo final de actividad.	Termina una actividad.
	Nodo final de flujo.	Termina un flujo específico dentro de una actividad; los otros flujos no se ven afectados.
	Nodo de decisión.	Se cruza el extremo de salida cuya condición de protección es verdadera. Puede tener opcionalmente una <<decisionInput>>.
	Nodo de fusión.	Copia tokens de entrada en su extremo de salida.
	Nodo fork.	Divide el flujo en múltiples flujos concurrentes.
	Nodo join (sincronización).	Sincroniza múltiples flujos concurrentes. Puede tener opcionalmente una especificación de sincronización para modificar su semántica.

14.8.1. Nodo inicial y nodos finales

Como hemos mencionado anteriormente, el nodo inicial es el punto en el que empieza el flujo cuando se invoca una actividad. Una actividad puede tener más de un nodo inicial. En este caso, el flujo empieza en todos los nodos iniciales simultáneamente y se ejecuta concurrentemente.

Una actividad también se puede iniciar por una acción de aceptar evento (véase el capítulo 15), o por un nodo de parámetro de actividad (se verá en una sección posterior), por lo que los nodos iniciales no son obligatorios dado que existe otra forma de iniciar la actividad. El nodo final de actividad detiene todos los flujos dentro de una actividad. Una actividad puede tener muchos nodos finales de actividad y el primero en activarse termina el resto de flujos y la propia actividad.

El nodo final de flujo simplemente detiene uno de los flujos dentro de la actividad, los otros flujos continúan. Véase la figura 15:10 para un ejemplo.

14.8.2. Nodos de decisión y nodos de fusión

Un nodo de decisión tiene un extremo de entrada y dos o más extremos de salida. Un token que llega al extremo de entrada se ofrecerá a todos los extremos de salida pero atravesará al menos uno de ellos. El nodo de decisión actúa como un cruce de caminos en el flujo donde el token debe tomar solamente una dirección.

Cada uno de los extremos de salida está protegido por una condición de protección de modo que el extremo aceptará un token si, y sólo si, la condición de protección evalúa como verdadero. Es importante asegurarse de que las condiciones de protección se excluyen mutuamente de modo que solamente una de ellas pueda ser cierta en cualquier momento en el tiempo. Si no es así, el comportamiento del nodo de decisión está sin definir según la especificación de UML 2.

La palabra clave *else* se puede utilizar para especificar el extremo atravesado si ninguna de las condiciones de protección es verdadera.

La figura 14.14 muestra un sencillo ejemplo de un nodo de decisión. Después de la acción Comprobar correo, el flujo de control llega a un nodo de decisión. Si la condición [es basura] es verdadera, entonces el correo se envía a la basura, de lo contrario el correo se abre.

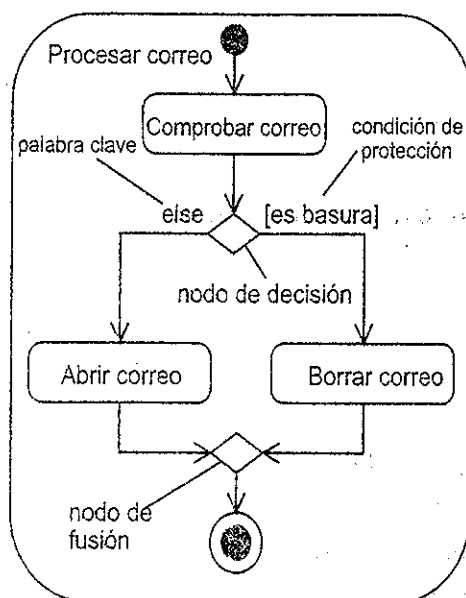


Figura 14.14.

Una nota estereotipada `<<decisionInput>>` proporciona una condición de decisión para un nodo de decisión. El resultado de esta condición se utiliza por las condiciones de protección en los extremos de salida. Un ejemplo de fragmento de actividad se muestra en la figura 14.15. En este fragmento, la condición de decisión compara la cantidad de retirada de fondos solicitada con el saldo actual. Si el saldo es mayor que o igual a la cantidad solicitada, entonces la condición evalúa como

verdadero y el flujo pasa a Retirar cantidad. De lo contrario, se registra un fallo.

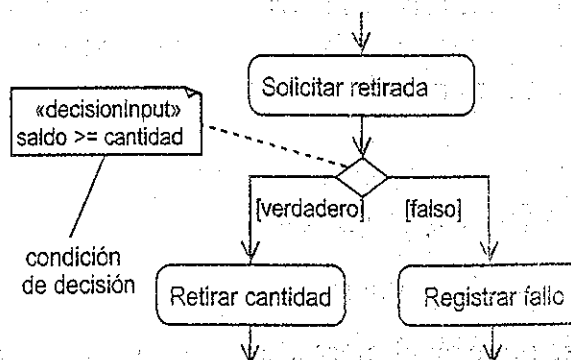


Figura 14.15.

La figura 14.14 muestra un nodo de fusión. Los nodos de fusión tienen dos o más extremos de entrada y un solo extremo de salida. Fusionan todos sus flujos entrantes en un solo flujo de salida. La semántica es muy sencilla; todos los tokens ofrecidos en los extremos entrantes se ofrecen en el extremo saliente y no existe modificación del flujo o los tokens.

Un nodo de fusión y un nodo de decisión se pueden combinar en un solo símbolo como muestra la figura 14.16. Sin embargo, no recomendamos esta notación abreviada ya que normalmente es mucho más claro mostrar nodos separados de fusión y decisión.

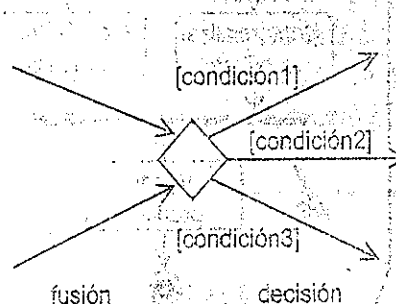


Figura 14.16.

14.8.3. Nodos fork y join, concurrencia

Puede crear flujos concurrentes dentro de una actividad al utilizar un nodo fork para dividir un simple flujo en múltiples flujos concurrentes. Mientras que la concurrencia es normalmente una decisión de diseño, a menudo necesita mostrar actividades concurrentes cuando está modelando procesos de negocio. Por lo tanto, normalmente utilizará nodos fork y join en el workflow de análisis al igual que en el de diseño.

Un nodo fork tiene un extremo entrante y dos o más extremos salientes. Los tokens que llegan al extremo entrante se duplican y se ofrecen en todos los extremos salientes simultáneamente. Esto divide el único flujo entrante en múltiples flujos salientes paralelos. Cada extremo saliente puede tener una condición de pro-

tección y, como los nodos de decisión, un token solamente puede atravesar el extremo saliente si la condición de protección es verdadera.

Los nodos join tienen múltiples extremos entrantes y un solo extremo saliente. Sincronizan flujos al ofrecer un token en su único extremo de salida cuando existe un token en todos sus extremos de entrada. Realiza un AND lógico en sus extremos de entrada.

La figura 14.17 muestra un sencillo ejemplo de un Procesar producto que utiliza nodos fork y join. En este ejemplo:

- El producto se diseña primero.
- El producto se pone en el mercado y manufactura concurrentemente.
- El producto se vende solamente después de que esos procesos están completos.

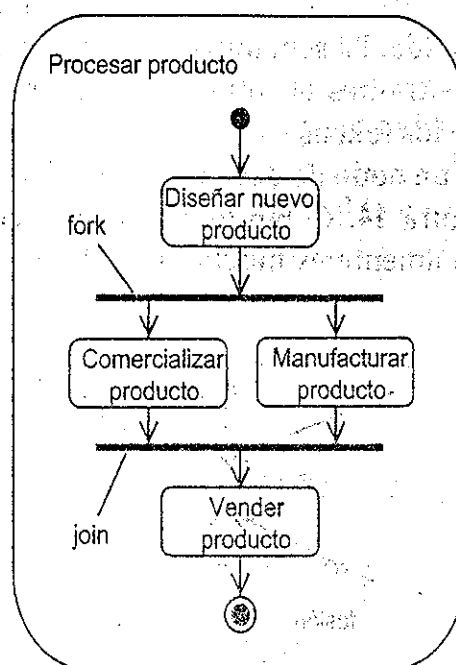


Figura 14.17.

En la figura 14.17 la actividad empieza con la acción Diseñar nuevo producto. Después de esta acción, un nodo fork divide el flujo único en dos flujos concurrentes. En uno de estos flujos el producto se pone en el mercado y en el otro se manufactura. El nodo join sincroniza los dos flujos concurrentes porque espera un token de cada una de las acciones concurrentes. Cuando tiene un token de cada acción, ofrece un token a su extremo de salida y el flujo pasa a la acción Vender producto.

Cuando modela nodos join, es importante asegurarse de que todos los extremos de entrada recibirán un token. Por ejemplo, en la figura 14.17, si existieran condiciones de protección mutuamente exclusivas en los flujos de salida del fork, el join nunca recibiría suficientes tokens a activar y esto podría hacer que la actividad se colgara.

14.9. Nodos de objeto

Los nodos de objeto son nodos especiales que indican que las instancias de un clasificador especial están disponibles en un punto específico en la actividad. Se etiquetan con el nombre del clasificador y representan instancias de ese clasificador o sus subclases. El fragmento de actividad en la figura 14.18 muestra un nodo de objeto que representa instancias del clasificador Pedido o subclases Pedido.

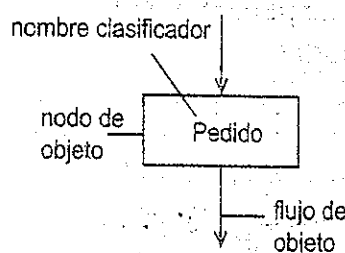


Figura 14.18.

Los extremos de entrada y salida de los nodos de objeto son flujos de objeto. Éstos son tipos especiales de flujos que representan el movimiento de objetos alrededor de la actividad. Los propios objetos se crean y consumen por nodos de acción. La figura 14.19 muestra la actividad Procesar producto incorporada en la figura 14.17, actualizada para incluir particiones y para mostrar la creación de un objeto EspecificacionProducto por la acción Diseñar nuevo producto. El objeto EspecificacionProducto se consume por la acción Manufacturar producto que lo utiliza para definir el proceso de manufacturación.

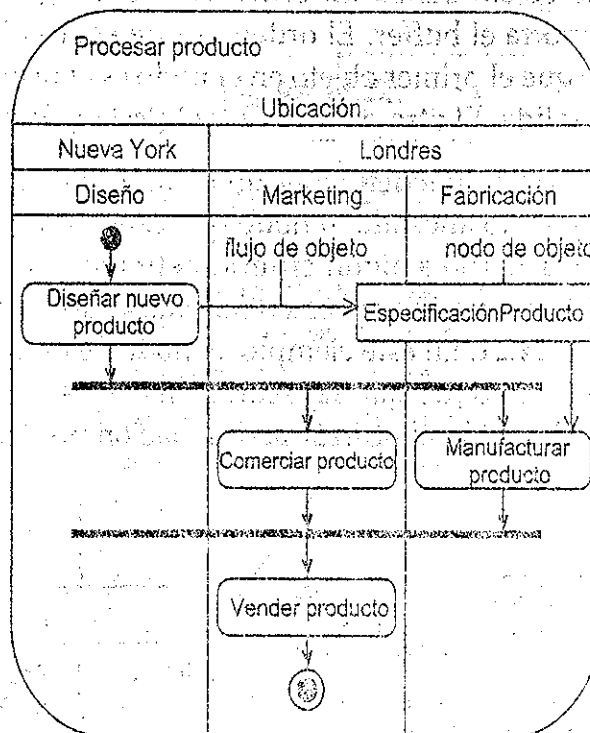


Figura 14.19.

Cuando un nodo de objeto recibe un token de objeto en uno de sus extremos de entrada, ofrece este token en todos sus extremos de salida simultáneamente y estos extremos completan el token. El punto clave es que sigue habiendo un solo token, el token no se replica en los extremos. El primer extremo que acepta el token lo coge.

14.9.1. Semántica del buffer del nodo de objeto

Los nodos de objeto pueden tener semántica interesante. Actúan como *buffers*, lugares en la actividad donde pueden residir los tokens de objeto mientras esperan a ser aceptados por otros nodos.

Por defecto, todo nodo de objeto puede albergar un número infinito de tokens de objeto. Sin embargo, algunas veces necesita indicar que el buffer tiene un tamaño finito. Puede realizar esto al asignar al nodo de objeto un límite superior que indica el número máximo de tokens que es capaz de albergar en cualquier momento. Cuando los tokens de objeto se ofrecen al nodo de objeto, los acepta hasta que está lleno. Un ejemplo de un nodo de objeto con un límite superior especificado se muestra en la figura 14.20.

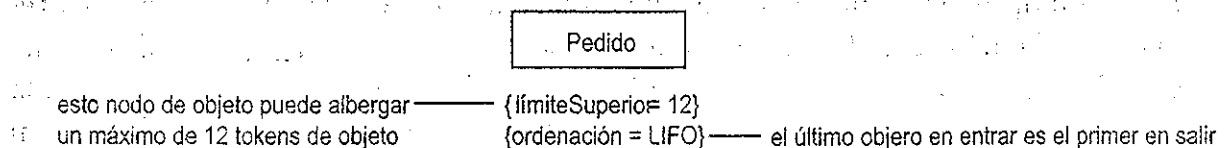


Figura 14.20.

Puede especificar dos aspectos de la semántica de buffer para nodos de objeto.

- Los nodos de objeto tienen un orden (véase la figura 14.20) que especifica cómo se comporta el buffer. El orden por defecto es FIFO (*first in- first out*), esto significa que el primer objeto en el nodo es el primero que se ofrece a sus extremos de salida. El otro orden es LIFO (*last in, first out*).
- Los nodos de objeto pueden tener un comportamiento de selección. Éste es un comportamiento anexado al nodo que selecciona objetos de los extremos de entrada de acuerdo a algún criterio definido por el modelador. La selección se especifica por una nota estereotipada <<selection>> como se muestra en la figura 14.21. En este ejemplo, el nodo de objeto selecciona solamente aquellos objetos Pedido que se crearon en el mes de diciembre. Los ofrece a sus extremos de salida al utilizar la ordenación predeterminada (FIFO).

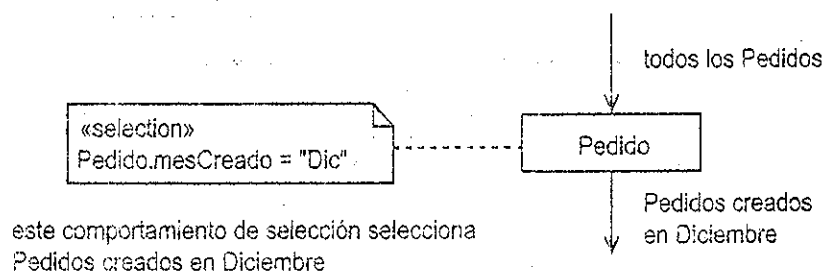


Figura 14.21.

Puede utilizar un nodo de objeto para recopilar objetos de múltiples flujos entrantes de objeto o para distribuir objetos a múltiples flujos salientes de objeto. En estos casos, está utilizando el nodo específicamente para su semántica de buffer y puede asignarle el estereotipo <<centralBuffer>> para resaltar este hecho.

Los nodos de objeto pueden poner en buffer conjuntos de objetos. Un conjunto es una colección de objetos en los que no hay duplicado, es decir, cada objeto tiene una identidad única. Para mostrar esto, simplemente prefija el nombre de clasificador con Conjunto de. Puede ver un ejemplo de esto en la figura 14.23.

Tratamos <<selection>> y <<centralBuffer>> en algo más de detalle en el capítulo 15.

14.9.2. Representar objetos en estado

Los nodos de objeto pueden representar objetos en un estado determinado. Por ejemplo, la figura 14.22 muestra un fragmento de una actividad de procesamiento de pedido que acepta objetos Pedido que están en el estado Abierto y los entrega. Los estados de objeto referenciados por nodos de objeto se pueden modelar con máquinas de estado como describimos en el capítulo 21.

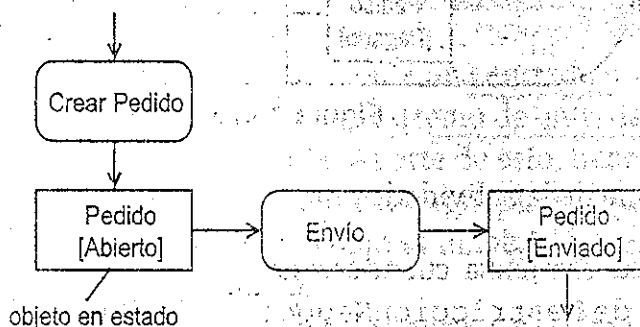


Figura 14.22.

14.9.3. Parámetros de actividad

Puede utilizar nodos de objeto para proporcionar entradas y salidas de actividades, según se ilustra en la figura 14.23. Los nodos de objeto de entrada y salida se deberían dibujar solapando el marco de actividad. Los nodos de objeto de entrada tienen uno o más extremos de salida en la actividad; los nodos de objeto de salida tienen uno o más extremos de entrada fuera de la actividad.

En la figura 14.23 la actividad Proceso de producto encargado tiene tres parámetros de entrada, SolicitudCliente, Conjunto de RestricciónNegocio y Pedido, y un parámetro de salida. El nodo Conjunto de RestricciónNegocio contiene un conjunto de objetos RestricciónNegocio.

Existen varios requisitos de negocio para este proceso:

- Los productos se diseñan basándose en una SolicitudCliente. Esto crea una EspecificacionProducto.

- El diseño del producto tiene en cuenta cualquier RestriccionNegocio.
- El pago solamente se requiere después de diseñar el producto.
- El producto no se puede manufacturar hasta que se ha recibido el pago y existe una EspecificacionProducto.
- La entrega no puede ocurrir hasta que se ha manufacturado el producto.

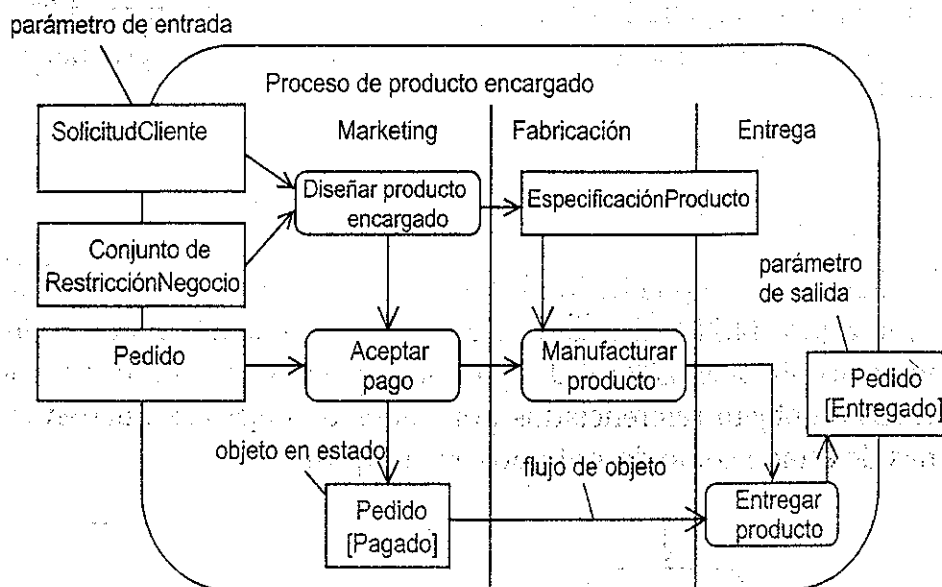


Figura 14.23.

Este es el detalle de la actividad:

1. La actividad comienza cuando existe una *SolicitudCliente* AND y un *Conjunto de RestriccionNegocio* en los flujos del objeto de entrada de la acción *Diseñar producto encargado*. Esta acción consume sus objetos de entrada y muestra como salida un objeto *EspecificaciónProducto*.
2. Cuando la acción *Aceptar pago* recibe un token de control de *Diseñar producto encargado* AND se ofrece un objeto *Pedido* en el flujo de objeto de entrada que ejecuta. Cambia el estado del objeto *Pedido* a *Pagado* y lo muestra como salida en su único flujo de objeto de salida.
3. El flujo de control pasa entonces a la acción *Manufacturar producto*. Esto consume la salida de objeto *EspecificaciónProducto* por *Diseñar producto encargado* y ofrece un token de control a *Entregar producto*.
4. *Entregar producto* se ejecuta cuando un token de control está disponible desde *Manufacturar producto* AND un objeto *Pedido* en el estado *Pagado* se encuentra disponible en el flujo de objeto de entrada. Esto tiene como salida el objeto *Pedido* en el estado *Entregado*. Este objeto *Pedido* es el parámetro de salida de la actividad.

Puede ver que podemos satisfacer los requisitos de negocio bastante fácilmente de la siguiente forma:

- No nos dedicamos a Diseñar producto encargado hasta que tenemos una SolicitudCliente AND un Conjunto de RestricciónNegocio.
- No podemos Aceptar pago hasta que tengamos un objeto Pedido AND la acción Diseñar producto encargado haya terminado.
- No podemos Manufacturar producto hasta que tengamos una EspecificaciónProducto AND la acción Aceptar pago haya terminado (es decir, no lo manufacturaremos hasta que se haya pagado).
- No podemos Entregar producto hasta que se haya manufacturado (Manufacturar producto haya terminado) AND se haya pagado (Pedido esté en el estado Pagado).

Esto ilustra parte del potencial de los diagramas de actividad. Pueden modelar procesos complejos de forma concisa y precisa.

14.10. Pins

Una actividad que tenga numerosos flujos de objeto puede hacerse muy complicada. Puede utilizar pins para aclararlo.

Un pin es un nodo de objeto que representa una entrada o salida de una acción.

Los pins de entrada tienen exactamente un extremo de entrada y los pins de salida tienen exactamente un extremo de salida. Aparte de esto, tienen la misma semántica y sintaxis que los nodos de objeto. Sin embargo, puesto que son tan pequeños, tiene que escribir toda la información, como el nombre del clasificador, fuera del pin, pero tan cerca de éste como pueda.

La figura 14.24 muestra una actividad Conectar que tiene dos flujos de objeto. La actividad comienza con la acción Obtener NombreUsuario. Esto tiene como resultado un objeto válido NombreUsuario. La siguiente acción es Obtener Contraseña que tiene como resultado un objeto válido Contraseña. La actividad Autenticar usuario se ejecuta cuando recibe un NombreUsuario válido y una Contraseña válida en su flujo de objeto de entrada. El usuario se autentica y la actividad termina.

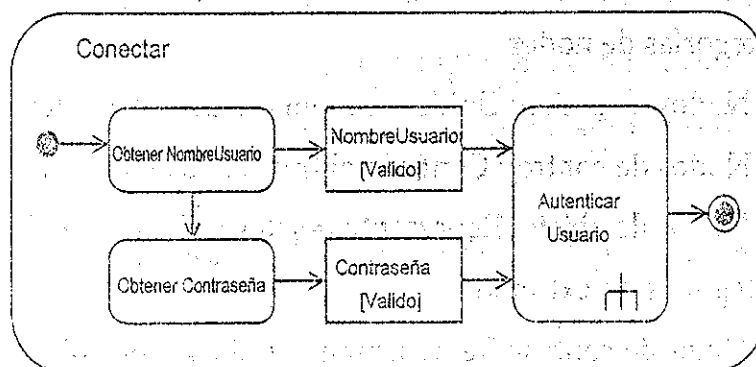


Figura 14.24.

La figura 14.25 muestra exactamente la misma actividad pero dibujada utilizando pins. Puede ver que la notación de pin es más compacta y mantiene el diagrama algo más claro.

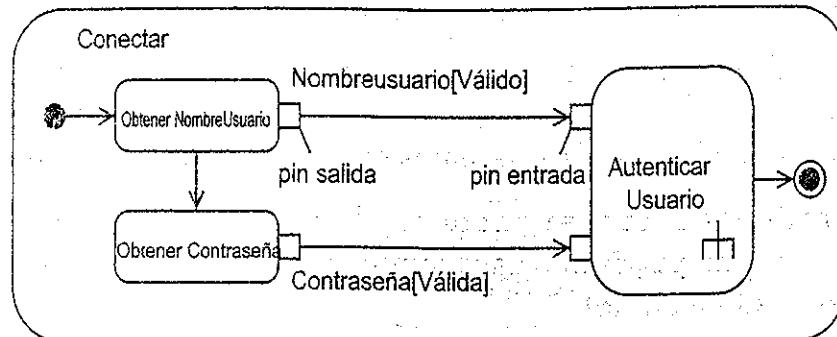


Figura 14.25.

Comparando la figura 14.24 y la figura 14.25 puede ver que el nodo de objeto NombreUsuario es exactamente equivalente a la combinación del pin de salida en Obtener NombreUsuario y el pin de entrada en Autenticar usuario. Debido a esto, el nodo de objeto algunas veces se considera como un pin independiente.

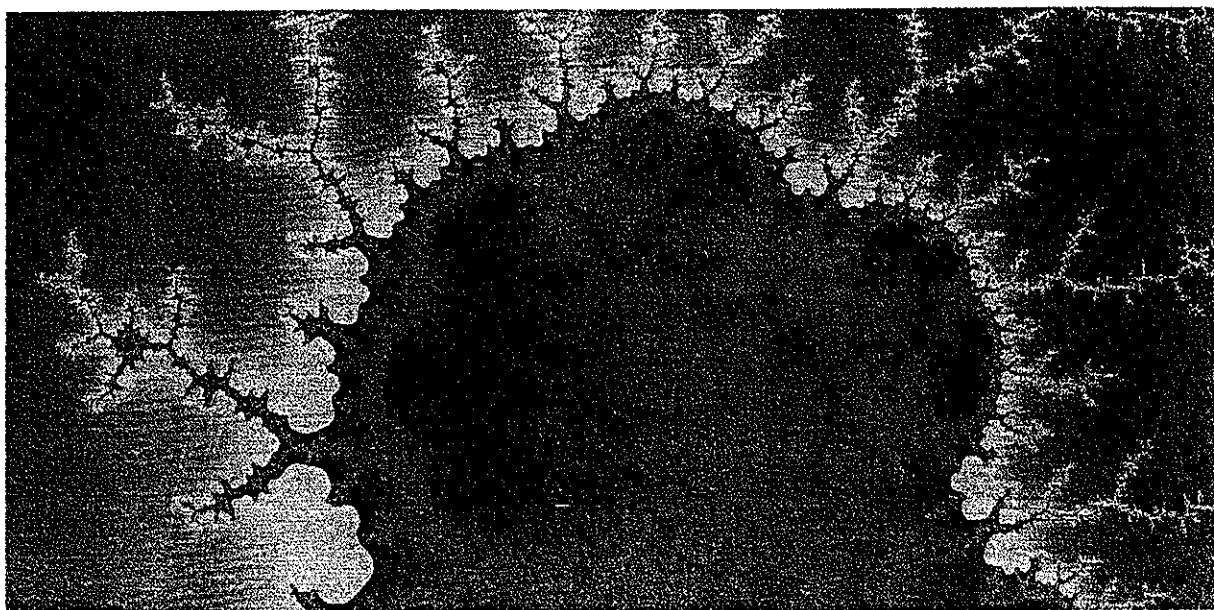
14.11. ¿Qué hemos aprendido?

En este capítulo ha visto que puede utilizar diagramas de actividad para modelar muchos tipos diferentes de procesos. Ha aprendido lo siguiente:

- Los diagramas de actividad son diagramas de flujo orientados a objeto:
 - Los utiliza para modelar todo tipo de procesos.
 - Puede anexar diagramas de actividad a cualquier elemento de modelado para capturar su comportamiento.
 - Un buen diagrama de actividad comunica un aspecto específico del comportamiento del sistema.
 - En UML 2 los diagramas de actividad tienen semántica de Petri Net.
- Las actividades son redes de nodos conectadas por extremos.
 - Categorías de nodos:
 - Nodos de acción: Unidades atómicas de trabajo dentro de la actividad.
 - Nodos de control: Controla el flujo en la actividad.
 - Nodos de objeto: Representa objetos utilizados en la actividad.
 - Categorías de extremos:
 - Flujos de control: Representan el flujo de control en la actividad.
 - Flujos de objeto: Representan el flujo de objetos en la actividad.

- Los tokens fluyen alrededor de la red y pueden representar:
 - El flujo de control.
 - Un objeto.
 - Algunos datos.
- Los tokens se mueven desde un nodo origen a un nodo destino a través de un extremo dependiendo de:
 - Postcondiciones del nodo origen.
 - Condiciones del extremo de protección.
 - Precondiciones del destino.
- Las actividades pueden tener precondiciones y postcondiciones.
- Nodos de acción:
 - Se ejecutan cuando existe un token simultáneamente en cada uno de sus extremos de entrada y se cumplen sus precondiciones.
 - Después de ejecutarse, los nodos de acción ofrecen tokens simultáneamente en todos los extremos de salida cuyas postcondiciones se cumplen.
 - Un fork implícito.
 - Nodo de acción de llamada:
 - Invocar una actividad; utiliza el símbolo rastrillo.
 - Invocar un comportamiento.
 - Invocar una operación.
 - Enviar un nodo de acción de señal.
 - Nodo de acción de aceptar evento.
 - Nodo de acción de aceptar evento de tiempo, se ejecuta cuando su expresión de tiempo es verdadera:
 - Un evento en el tiempo (por ejemplo, final del año).
 - Un punto en el tiempo (por ejemplo, el 11/03/1960).
 - Una duración (por ejemplo, esperar 10 segundos).
- Nodos de control:
 - Nodo inicial: Indica dónde empieza el flujo cuando se invoca una actividad.
 - Nodo final de actividad, termina una actividad.
 - Nodo final de flujo, termina un flujo específico dentro de una actividad.

- Nodo de decisión, se cruza el extremo de salida cuya condición de protección es verdadera: puede tener un `<<decisionInput>>`.
- Nodo de fusión, copia tokens de entrada en su único extremo de salida.
- Nodo fork, divide el flujo en múltiples flujos concurrentes.
- Nodo join, sincroniza múltiples flujos concurrentes.
- Particiones de actividad, una agrupación de alto nivel de acciones relacionadas.
 - Particiones de una jerarquía basada en una dimensión.
- Los nodos de objeto representan instancias de un clasificador.
 - Los extremos de entrada y salida son flujos de objeto, representan el movimiento de objetos.
 - Los extremos de salida de nodos de objeto compiten por cada token de salida.
 - Los nodos de objeto actúan como buffers:
 - `{límiteSuperior=n}`.
 - `{orden=FIFO} XOR {orden=LIFO}`.
 - `{orden=FIFO}` es el predeterminado.
 - pueden tener una `<<selection>>`.
 - Los nodos de objeto pueden representar objetos en un estado determinado:
 - Deben ser coherentes con la máquina de estado.
 - Los parámetros de actividad son entrada o salida de nodos de objeto de una actividad:
 - Dibujada solapando el marco de actividad.
 - Los parámetros de entrada tienen uno o más extremos de salida en la actividad.
 - Los parámetros de salida tienen uno o más extremos de entrada fuera de la actividad.
- Pins.
 - Un nodo de objeto que representa una entrada o salida de una acción o actividad.



15

**Diagramas
avanzados
de actividad**
