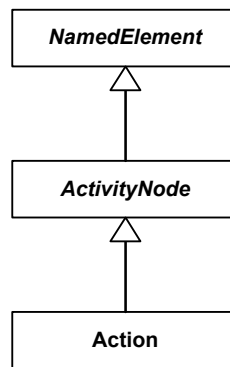


Die folgenden Abschnitte behandeln die Metamodelle zu Notationselementen aus Kapitel 13 (Aktivitätsdiagramm). Diese Abschnitte sind für die Zertifizierung zum „OMG Certified UML Professional (Fundamental)“ wichtig. Zum schnelleren Auffinden haben wir die Seitenzahlen des dazugehörigen Abschnitts in der 3. Auflage von „UML 2 glasklar“ hinzugefügt (in der 2. Auflage sind diese Abschnitte im Buch abgedruckt)

Aktion (S.269)

Metamodell



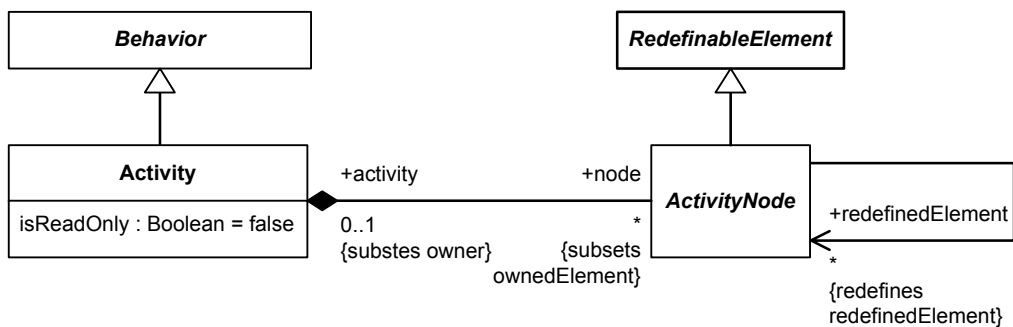
Metamodell Action

Eine Aktion wird durch die Metaklass `Action` repräsentiert. `Action` spezialisiert die abstrakte Klasse `ActivityNode`. Diese stellt eine Oberklasse für sämtliche Knoten dar, die in einer Aktivität vorkommen können.

Der Name einer Aktion ergibt sich aus der Eigenschaftsvererbung eines `NamedElements`

Aktivität (S.274)

Metamodell

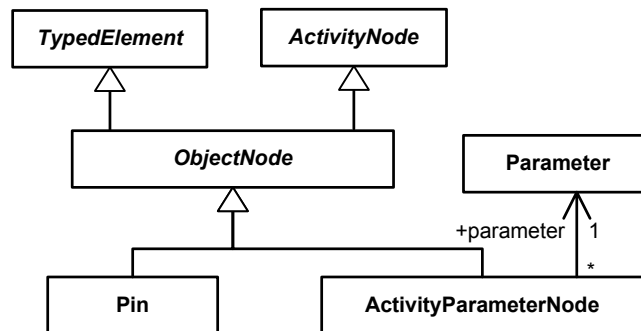


Metamodell Aktivität

Eine Aktivität wird durch die Metaklass `Activity` repräsentiert. Diese ist als Spezialisierung von `Behavior` integriert, was Auswirkungen hinsichtlich ihrer Verwendbarkeit nach sich zieht. Eine Aktivität besteht aus diversen Knoten (+node). Diese `ActivityNodes` werden durch Aktionen (Actions, 13.4.1), Objektknoten (`ObjectNode`, 13.4.3) und Kontrollelemente (`ControlNode`, 13.4.5) spezialisiert. Da eine Aktivität spezialisierbar (Behavior-Eigenschaft) bzw. überschreibbar ist, muss dies auch für ihre Knoten gelten (`RedefinableElement`, `+redefinedElement`). Die spezielle Aktivität ersetzt somit die Knoten der allgemeinen Aktivität (ähnlich wie Operationen von Unterklassen, die Operationen einer Oberklasse ersetzen). Das Attribut `isReadOnly` zeigt mit einem `true`-Wert an, ob die Aktivität Werte außerhalb der Aktivität bzw. `inout`-Werte verändert.

Objektknoten (S.276)

Metamodell

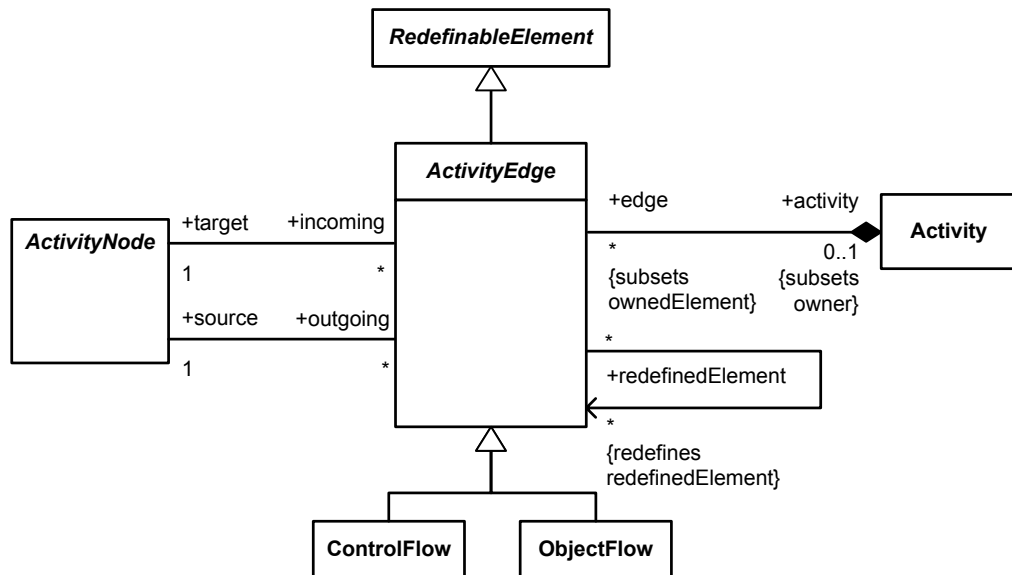


Metamodell Objektknoten

Ein Objektknoten wird durch die abstrakte Klasse `ObjectNode` im Metamodell verankert. Er vereinigt dabei zwei Eigenschaften, die eines Knotens (`ActivityNode`) und die eines typisierten Elements (ähnlich wie eine Variable oder ein Attribut) als `TypedElement`. Er bzw. seine Daten-Tokens besitzen also einen Typ. Die beiden Arten des Objektknotens: die Pins und die Eingangs- bzw. Ausgangsparameter einer Aktivität, ausgedrückt durch die Klassen `Pin` und `ActivityParameterNode`. Letzterer ist zudem ein `Parameter` zugeordnet (`+parameter`).

Kanten (S.283)

Metamodell

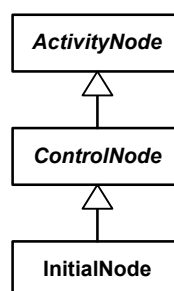


Metamodell Aktivitätskanten

Die Kanten werden allgemein durch die Klasse `ActivityEdge` bzw. deren konkrete Verfeinerungen als Kontroll- bzw. Objektfluss (`ControlFlow` und `ObjectFlow`). Da eine Kante genau zwei beliebige Knoten verbindet folgt der Bezug (`+target/+source`) zur abstrakten Klasse `ActivityNode`, die alle Knoten repräsentiert. Vollkommen analog zu den Knoten erfolgt die Eingliederung der Kanten in die Aktivität (`+edge`) und die Überschreibbarkeit (`RedefinableElement`, `+redefinedElement`).

Startknoten (S.288)

Metamodell

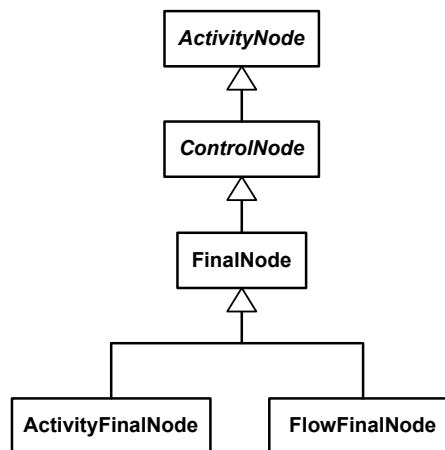


Metamodell Startknoten

Die Einbindung des Startknotens (*InitialNode*) in das UML-2-Metamodell ist leicht verständlich. Als eine Spezialisierung von *ControlNode* – einer abstrakten Klasse, die alle Kontrollelemente abbildet – bekommt der Startknoten die Eigenschaften von *ActivityNode*. Über hier nicht sichtbare Einschränkungen wird zudem geregelt, dass ein Startknoten keine eingehenden Kanten (*ActivityEdges*) haben darf und dass die ausgehenden nur Kontrollflüsse (*ControlFlow*) darstellen dürfen.

Endknoten (S.289)

Metamodell



Metamodell Endknoten

Die Einbindung des Endknotens (*FinalNode*) erfolgt analog zum Startknoten. Die Unterscheidung in beiden Knotentypen erfolgt durch Unterklassen (*ActivityFinalNode/FlowFinalNode*). Beachten Sie: Im Rahmen der Fundamental-Zertifizierung ist nur der *ActivityFinalNode* relevant.