

UML

Aktivitätsdiagramme

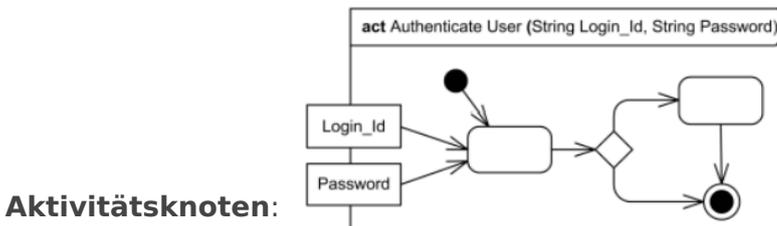
- Dynamisches Verhalten
- Modellierbeispiele

Dynamisches Verhalten

Aktivitätsdiagramme beschreiben **Ausschnitt** aus Ablaufmöglichkeiten eines Systems mit Hilfe von *Aktivitäten*

Einsatz: Detaillierte Sicht auf Anwendungsfälle | den Ablauf innerhalb von Klassen | Geschäftsprozessmodellierung mit der UML

Aktivität – Parametrisiertes Verhalten, dargestellt als koordinierter Fluss von Aktionen



Aktionen: abgerundete Rechtecke

Objekte: Rechtecke

Kontrollflusselemente: verschiedene Symbole für Entscheidungen, parallele Ausführung

Kontrollfluss: Verbinden von Knoten mit gerichteten Kanten

Aktion – Benanntes Element, das einen einzelnen atomaren Schritt innerhalb der Aktivität darstellt

Kann eingehende und ausgehende *Aktivitätskanten* enthalten, die *Kontroll- und Datenfluss* von und zu anderen Knoten definieren

Kann andere *Aktivitäten* aufrufen (call activity action)

Beginnt mit der *Ausführung*, wenn alle *Eingabebedingungen* erfüllt sind



Start- und Endzustand

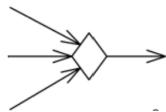
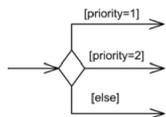
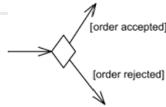


Startknoten – Start des Kontrollflusses in einer Aktivität

Aktivitäts-Endknoten – Ende jeglicher Kontrollflüsse in einer Aktivität

Fluss-Endknoten – Ende eines einzelnen Kontrollflusses

Endzustände können mehrfach vorkommen, Startzustände ideal nur einmal



Entscheidung

Steuerknoten, der eingehenden Kontrollfluss akzeptiert und Kontrollfluss aus mehreren ausgehenden Kontrollflüssen auswählt

Annotation von Bedingungen in Form von **booleschen Ausdrücken** an Transitionen

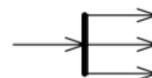
Unterstützt auch einen vordefinierten **else -Kontrollfluss**

Optionaler **Merge-Knoten** fügt mehrere eingehende alternative Flüsse zusammen

Parallele Ausführungspfade

Fork – Steuerknoten, der einen *eingehenden Kontrollfluss* in mehrere nebeneinander laufende Kontrollflüsse aufteilt

Die Ausführung des Diagramms durchläuft alle parallelen Pfade

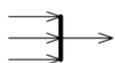


Die Ausführungsreihenfolge nebenläufiger Pfade ist *irrelevant*: nacheinander, gleichzeitig, abwechselnd

Join – Steuerknoten, der mehrere eingehende nebenläufige Kontrollflüsse synchronisiert

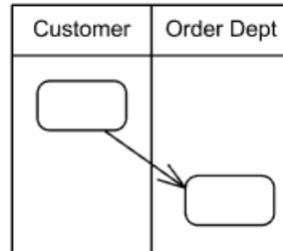
UND-Synchronisation: Die Kontrollflüsse aller Transitionen müssen vorliegen, bevor die

Transition starten kann



Probleme mit Fork und Join

- Es dürfen nur *Zweige* in einem *Join* *zusammengeführt* werden, die zuvor aus *demselben Fork* hervorgegangen sind.
- *Zweige*, die aus *verschiedenen* (unabhängigen) Forks stammen, können nicht *zugleich aktiv* sein.



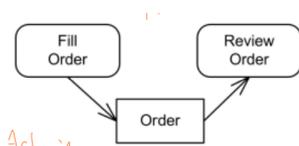
Verantwortlichkeitsbereiche

Gruppierung von Aktionen anhand gemeinsamer Merkmale

Definieren **organisatorische Verantwortlichkeiten**

Können hierarchisch aufgebaut werden: Klassenzugehörigkeit, Organisationseinheiten, Akteuren

Objekte und Objektfluss



Objektflusskanten - Kanten, mit denen der Datenfluss zwischen Aktionsknoten angezeigt wird

Objektknoten - Knoten, mit dem Objekte definiert werden. Vorliegen eines Objektes zu Zeitpunkt innerhalb einer Aktivität

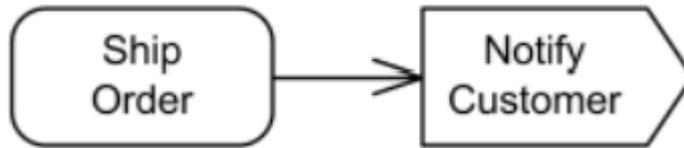


Pin - Objektknoten für Ein- und Ausgänge von Aktionen



Data Stores – Zentraler Speicherknoten für nicht-transiente Informationen

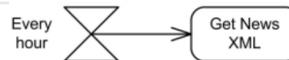
Signale senden und empfangen



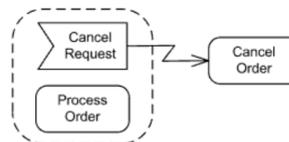
Sendsignalaktion – Senden eines Signals

Annahmeeeignis – Aktion, die auf das Eintreten eines bestimmten Ereignisses (z. B. Signal) wartet

Sender- und Empfängerobjekte von Signalen müssen nicht spezifiziert werden



Wartezeitereignisse und Unterbrechungen



Wartezeitaktion – Warten auf das Eintreten eines Zeitereignisses

Unterbrechbare Aktivitätsregionen – Aktionen, deren Ausführung durch Ereignisse unterbrochen werden können

Modellierbeispiele

Wie erstelle ich Aktivitätsdiagramme?

1. Wer sind die wichtigen **Akteure** des zu beschreibenden Prozesses?
2. Welche relevanten **Aktionen** führen diese Akteure aus?
3. Welche temporalen bzw. kausalen **Zusammenhänge** bestehen **zwischen** den **Aktionen** (Sequenzen, Entscheidungen,...)?
4. Welche **Objekte** werden für die Ausführung der Aktionen benötigt bzw. durch eine Aktion erzeugt oder verändert?
5. Welche asynchronen **Signale** werden ausgetauscht und wie hängen die Ausführungspfade zwischen den Aktionen davon ab?

Modellierungsbeispiel

Es soll die **Abwicklung einer Warenbestellung** in einem Shopssystem modelliert werden. Die **Bestellung** ist der Eingabeparameter der **Aktivität**. Falls eine **Bestellung** angenommen wurde, müssen zunächst alle **erforderlichen Informationen** gesammelt und eingegeben werden. Danach wird der **Versand der Ware** veranlasst und eine **Rechnung versendet**. Nach **Versand** und **Zahlungseingang** wird der **Vorgang abgeschlossen**.

