Projektentwicklung mit Together R2 for Eclipse

Im Rahmen des Labors für Softwaretechnik soll eine objektorientierte Softwareentwicklung durchgeführt werden. Als Entwicklungswerkzeug wird ’Together R2 for Eclipse’ von Borland eingesetzt, welches die grafische Spezifikationssprache UML umsetzt.

Um die einzelnen Entwicklungsphasen zu verdeutlichen und um die Handhabung des Werkzeugs zu erläutern, wird ein Beispielprojekt, die Entwicklung eines Kniffelspiels, durchgeführt.

Im vorliegenden Dokument wird die Handhabung von Together in groben Zügen an Hand von Beispielen dargestellt, teilweise sind auch Verweise auf die Dokumentation eingefügt.

1 Analyse

Das komplette Dokument zur Analyse ist unter Kniffel-Analyse.doc zu finden.

1.1 Finden der Funktionalität

Es soll die zu realisierende Funktionalität erfasst werden.
Sichtbarkeit:
• **sichtbar:** Die Funktion und ihre Auswirkungen werden von den Spielern wahrgenommen.
• **versteckt:** Die Spieler nehmen die Auswirkungen dieser Funktionen nicht oder nur indirekt wahr.

Relevanz:
• **notwendig:** Diese Funktion ist zwingend erforderlich. Ohne Sie kann das Spiel nicht gespielt werden
• **wichtig:** Das Fehlen dieser Funktion würde zu einer Beeinträchtigung des Spiels führen. Das Spiel kann aber noch gespielt werden.
• **sinnvoll:** Diese Funktionen würden das Spiel als Ganzes abrunden. Ihr Fehlen führt aber nur zu einer unwesentlichen Beeinträchtigung.

Siehe Kniffel-Analyse.doc 2.1 Funktionalität

1.2 Finden der Eigenschaften

Es werden die zu realisierenden Eigenschaften erfasst.

Siehe Kniffel-Analyse.doc 2.2 Eigenschaften

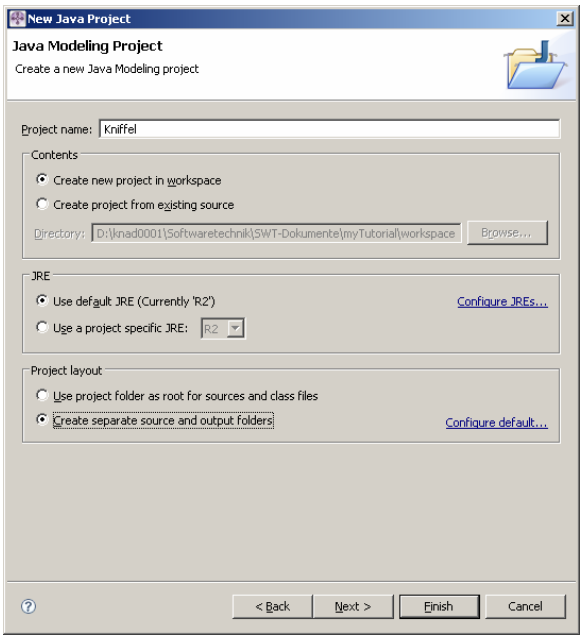
1.3 Starten von Together

Zu finden ist Together auf den Laborrechnern im LKIT unter :

1.4 Projekt anlegen

- **Together starten**
- **Neues Projekt anlegen**

  File → new → Project
  Modeling → Java Modeling Project **click next**



click Finish

- **Package Analyse in Projektknoten anlegen**
  Select Projektknoten, rechtsclick → new →Package
  Name eingeben, z.b. 'Analyse'

- **Diagramme anlegen**
  Select Packageknoten 'Analyse', rechtsclick → New Diagram

- **Diagramme als Bilder exportieren**
  Diagrammhintergrund selektieren
  Diagram → Export to Image..

1.5 Use Case Diagramm erstellen

Ein Use-Case ist im allgemeinen eine sehr grobe Beschreibung eines Ablaufs aus Benutzersicht. Diese Beschreibung enthält viele Schritte und Aktionen.

Was ist ein Use Case Diagramm?

• Es beschreibt das Verhalten eines Systems aus Benutzersicht.
• Es ist eine funktionale Beschreibung eines Systems und der wichtigsten Abläufe.
• Ist eine grafische Beschreibung für "wer benutzt das System" und "welche Interaktionen finden mit dem System statt".
• Abläufe, die innerhalb der Anwendung stattfinden werden Use Cases genannt.
• Die Einheiten außerhalb der Anwendung, die die Anwendung benutzen, werden Akteure benannt.

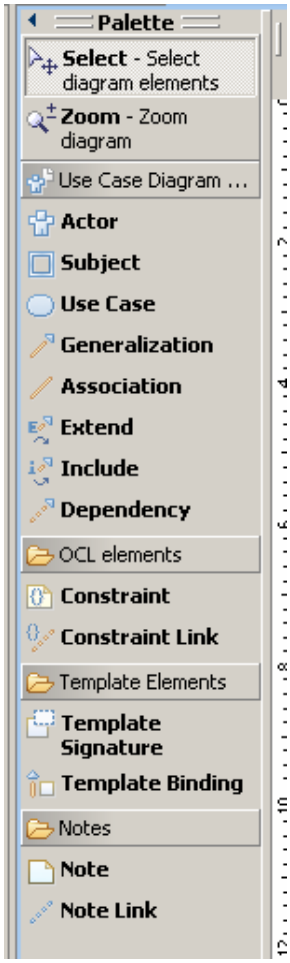
Wann wird ein Use Case Diagramm benutzt?

Es wird benutzt, um Anforderungen an das System zu beschreiben.

Siehe auch Skript Softwaretechnik ab Folie " Use Cases"(S.176 ff)

Erstellen eines Use Case Diagramms

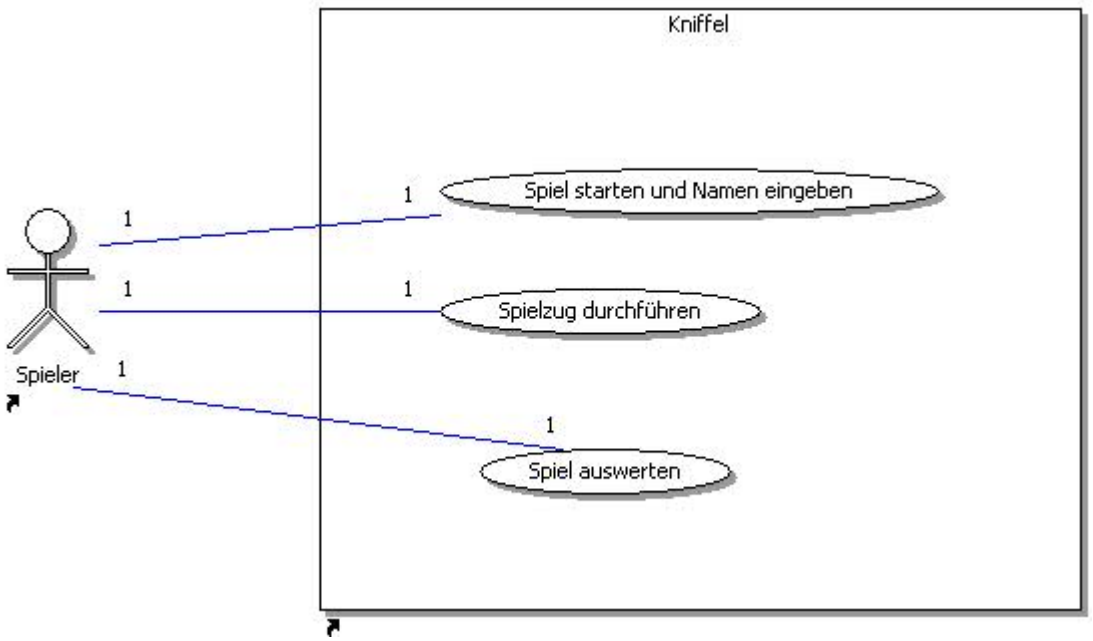
Select Packageknoten 'Analyse', rechtsclick → New Diagram → Use Case



- Zeichnen der Systemgrenze (**Subject)**
- Zeichnen der Use Case Symbole (**Use Case**)
- Bezeichnen der Symbole
- Zeichnen der Akteure (**Actor**)
- Benennen der Akteure
- Verbinden der Use Cases und der Akteure durch Beziehungen (**Generalization, Association, Extent, Include, Dependency**)

Tipp: Falls mehrere gleiche Symbole gezeichnet werden sollen, zuerst das zu erzeugende Symbol wählen, dann beim Clicken auf den Diagrammhintergrund die <Strg> Taste gedrückt halten. Zum Zurückschalten Select Icon auswählen.

Beispiel für ein Use Case Diagramm



Jede gefundene Funktion aus der Liste Funktionalität und jede Eigenschaft sollte mindestens einem Use Case zugeordnet werden können. Jeder Use Case sollte mehr als eine Funktion beinhalten.

Zu jedem Use Case soll eine textuelle Beschreibung erstellt werden.

Siehe Kniffel-Analyse.doc 2.3.2 Use Case Beschreibungen

1.6 Aktivitätsdiagramme erstellen

Was ist ein Aktivitätsdiagramm?

- stellt Abläufe und Datenflüsse eines Systems dar
- stellt die Aktivtäten innerhalb eines Use Cases dar

Wann wird ein Aktivitätsdiagramm benutzt?

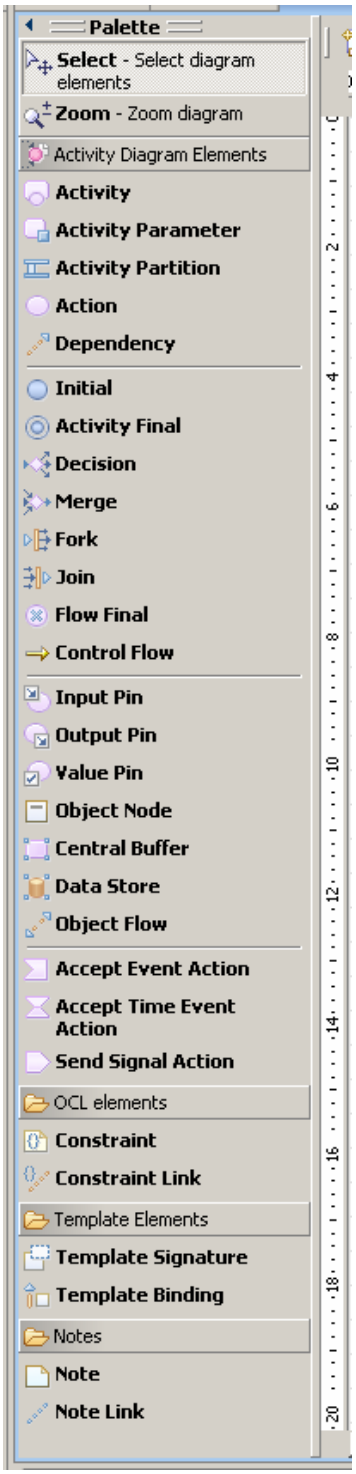
- Um den Ablauf zwischen vielen Use Cases darzustellen
- Um den Ablauf innerhalb eines Use Case darzustellen

Aktivitäten können in Swimlanes(Verantwortlichkeitsbereiche) aufgeteilt werden.

Siehe auch Skript Softwaretechnik ab Folie „Aktivitätsdiagramme(Uml)“ S. 196ff

Erstellen eines Aktivtätsdiagramms

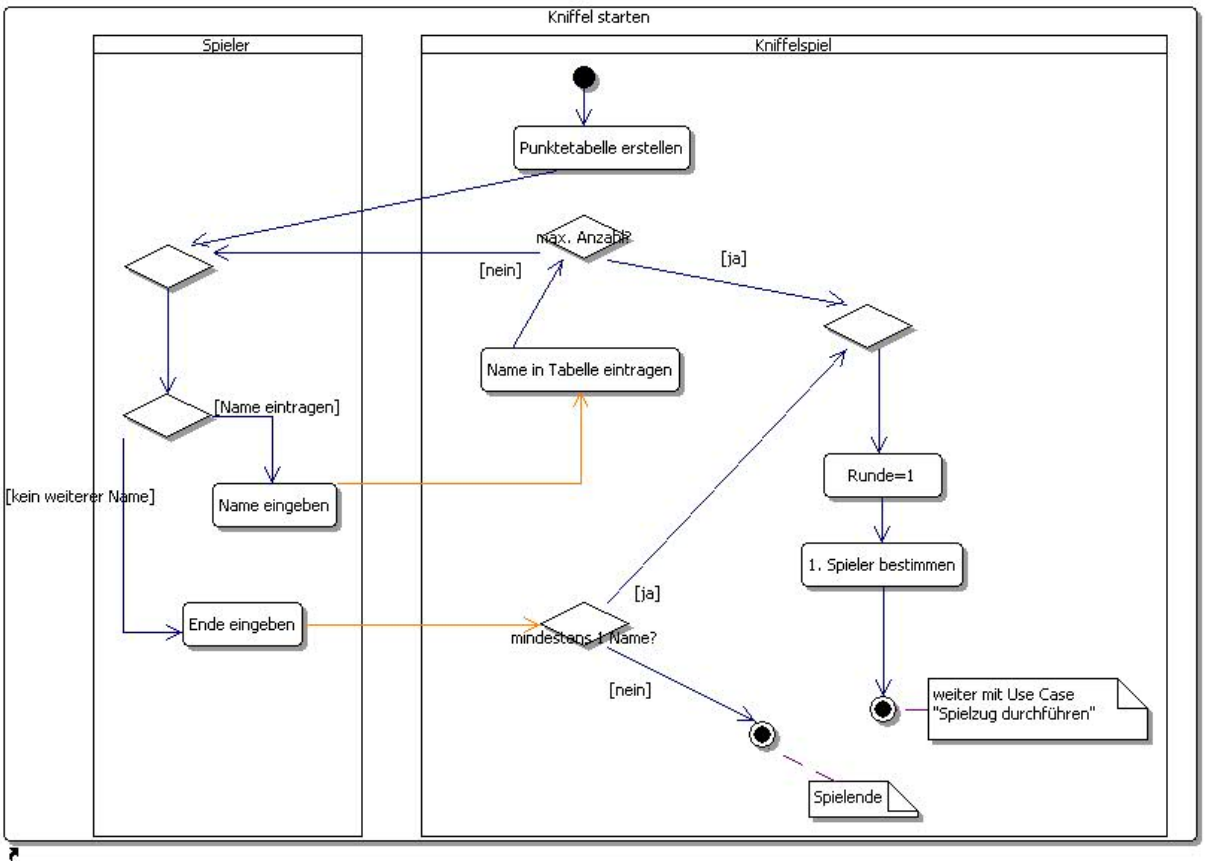
Select Packageknoten 'Analyse', rechtsclick → New Diagram → Activity



Activities (**Activity**) zeichnen
Partitions /Swimlanes (**Activity Partition**) zeichnen

Tipp: partitions drehen: rechtsclick auf activity → rotate activity partitions

Aktionen (**Action**) in den entsprechenden Swimlanes eintragen. Den Kontrollfluss (**Control Flow)** durch Verbinden der Aktionen darstellen, evtl sind Entscheidungen (**Decision)** und Zusammenführungen (**Merge**) zu zeichnen.



Siehe Kniffel-Analyse.doc 2.3.3 Detaillierte Beschreibung der Use Cases durch Aktivitätsdiagramme

Um Text in den Decision bzw. Merge Elementen anzuzeigen, muss in

Diagram → Preferences → View Management "Show Name of Decision / Merge element" ein Haken gesetzt werden.

Um die Ausgangsflüsse eines Decision-Elementes zu beschriften, die Properties anzeigen und unter "**guard"** den Text eingeben.

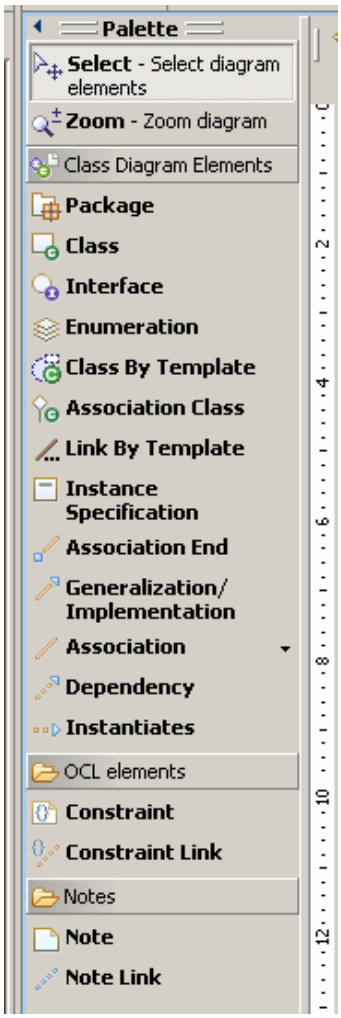
1.7 Objektmodell erstellen

- Objekte identifizieren
- Assoziationen identifizieren
- wesentliche Attribute ergänzen

Siehe auch Skript Softwaretechnik ab Folie “Erstellung des Objektmodells“ (S. 205ff)

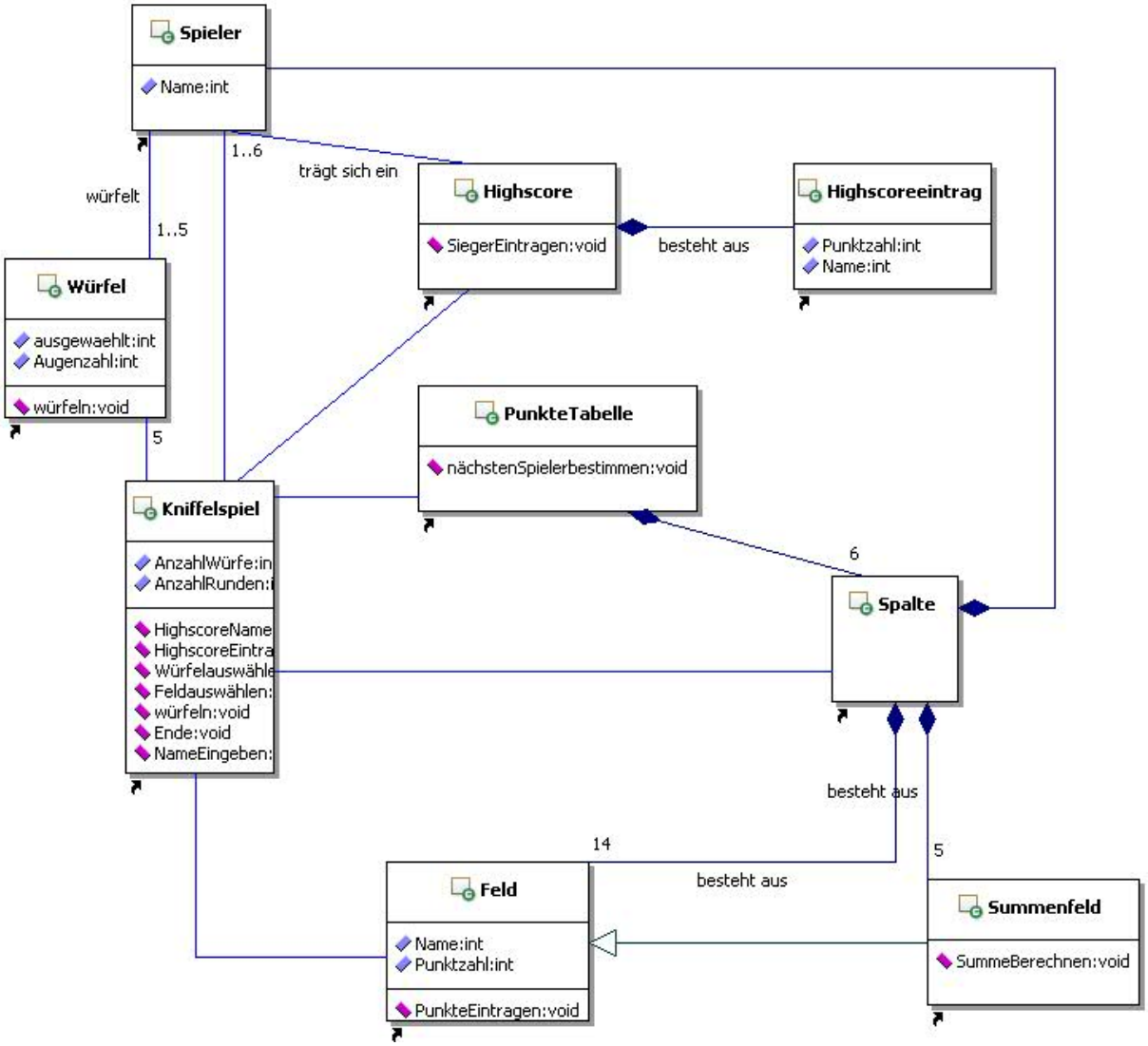
Select Packageknoten ’Analyse’, rechtsclick → New Diagram → Class

Es ist sinnvoll ein Package für die Objekte anzulegen, um eine bessere Übersicht zu haben
Es sollten Einstellungen fürs Klassendiagramm vorgenommen werden:
diagram → preferences → view management
uncheck ‘use fully qualified names in short cuts to Classes or Packages from different Packages’
Objekte werden automatisch in demselben Packages wie Diagramm angelegt, um die Übersichtlichkeit zu verbessern, sollten die Objekt in das neu angelegte Package verschoben werden.



Anlegen der Klassen und der zugehörigen Methoden und Attribute

Eintragen der Beziehungen der Klassen untereinander



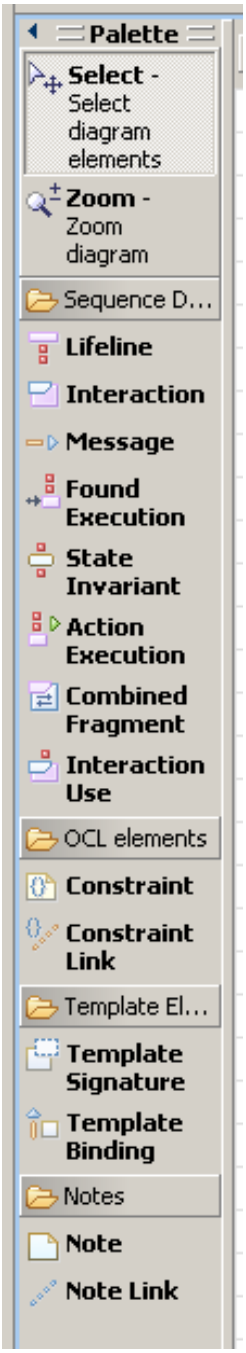
1.8 System-Sequenz-Diagramme erstellen

1.8.1 Sequenzdiagramme erstellen

Ein Sequenzdiagramm stellt die Interaktion zwischen zusammenarbeitenden Objekten dar.
Zeigt Objekte und Messages, die zwischen den Objekten verschickt werden, im zeitlichen Ablauf und somit auch den Kontrollfluss.

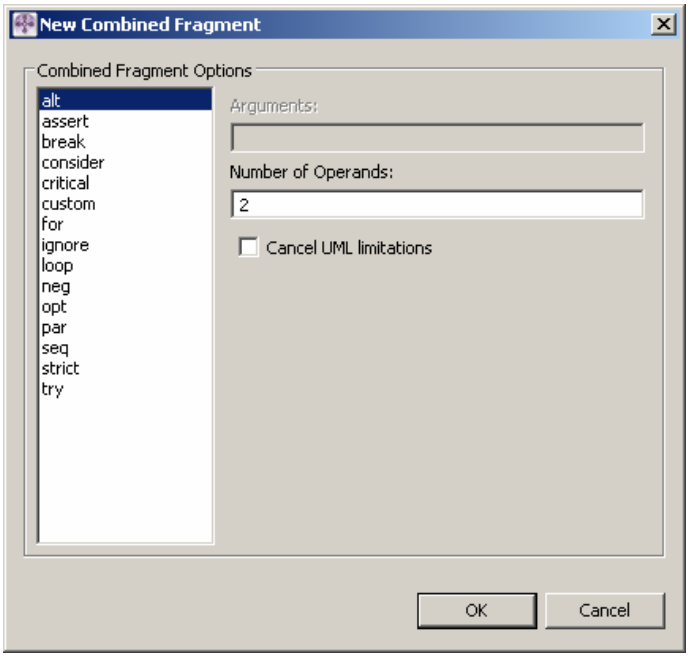
Siehe auch Skript Softwaretechnik ab Folie "System-Sequenz-Diagramme" (S. 217ff)

Select Packageknoten 'Analyse', rechtsclick → New Diagram → Sequence



Interaction erstellen
- **Lifeline**s eintragen
- **Messages** erzeugen

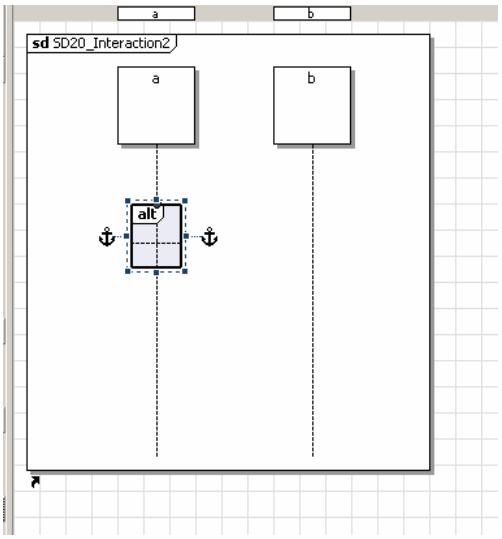
Unter **Combined Fragment** verbergen sich



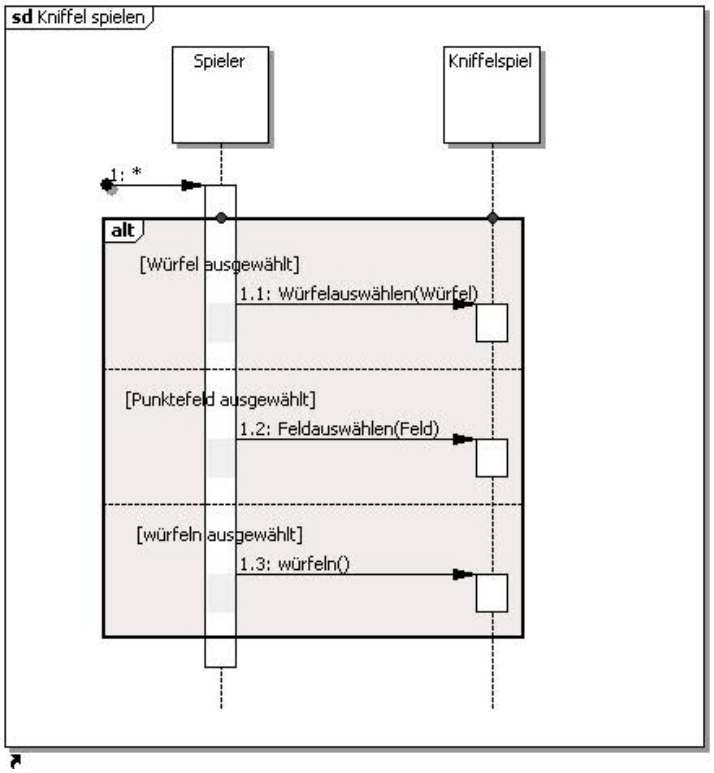
Die wichtigsten:

alt -- Alternatives Fragment
opt – Optionales Fragment
loop, for -- Schleifen

Tipp: wird der Name der message direkt im Diagramm eingegeben, werden die Parameter nicht angezeigt. Werden die Angaben in den Properties unter signature eingetragen, erscheinen diese auch im Diagramm



Durch Ziehen an den Ankern, kann das Fragment auf weitere Lifelines ausgedehnt werden.



1.8.2 Systemoperationen finden

Die Systemoperationen sind die Operationen, die Übergänge vom Benutzer zum System darstellen. Im vorliegenden Beispiel sind dies die Übergänge vom Spieler zum Kniffelspiel. Diese Übergänge wurden im Aktivitätsdiagramm orange dargestellt.

Siehe Kniffel-Analyse.doc 2.5 System-Sequenz-Diagramme

2 Design

Wird im 2. Teil veröffentlicht.