

Softwaremodellierung mit dem Enterprise Architect

Resources & Tutorials

Welcome to the Sparx Systems Resources page.
Here you will find a wealth of tutorials, extensions,
white papers, e-books and much more.

[Get started](#) [Gallery](#) [Reference list](#) [Global community](#) [Social media](#)



Webinars



Demo Videos



Case Studies



Learning Center



White Papers



User Guide Library



Tutorials



Brochures Library



DBMS setup scripts



Online User Guide

- Alle Beispiele können im Nachgang des Kurses frei verwendet werden
 - Ablage in einem Git-Repository
- UML-Kenntnisse sind hier vorausgesetzt
- Screenshots und Befehle stammen aus der Enterprise Architect Version 13
 - Aktuelle Version 14.x ist in der Benutzerführung deutlich verbessert

© Javacream, 2019

Javacream

Dr. Rainer Sawitzki

Alois-Gilg-Weg 6
81373 München

Table of Contents

Grundlagen	6
Anforderungsanalyse	20
Das fachliche Modell	34
Technisches Modell	64

1

GRUNDLAGEN

1.1

ÜBERSICHT

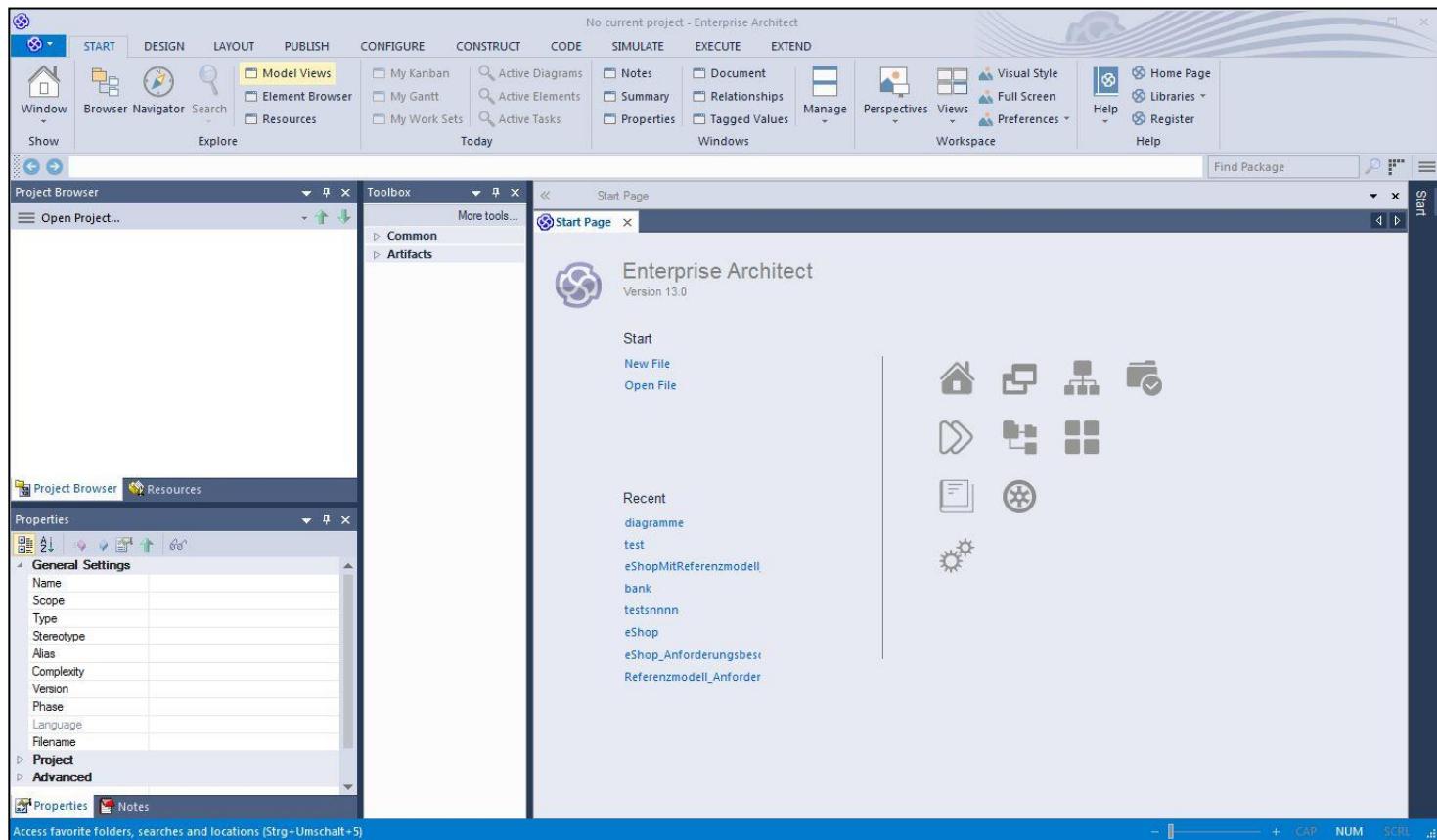
■ UML Werkzeug



- UML Werkzeug mit Erweiterungen
 - Alle UML Diagramme nutzbar
 - Toolspezifische Ergänzungen und Erweiterungen
 - BPMN
 - Requirement Modeling
 - Seit Version 11, verschiedene „Themes“ für UML
 - Seit Version 12 anderes Look and Feel
 - Seit Version 13 neues „MS“ Office Look and Feel
- Komplettes Werkzeug mit den folgenden Eigenschaften
 - UML 2.x basiert
 - Projekt-Management Ergänzungen
 - Dokumentationserstellung
 - Preis-Leistungsverhältnis
 - Große Akzeptanz und Verbreitung

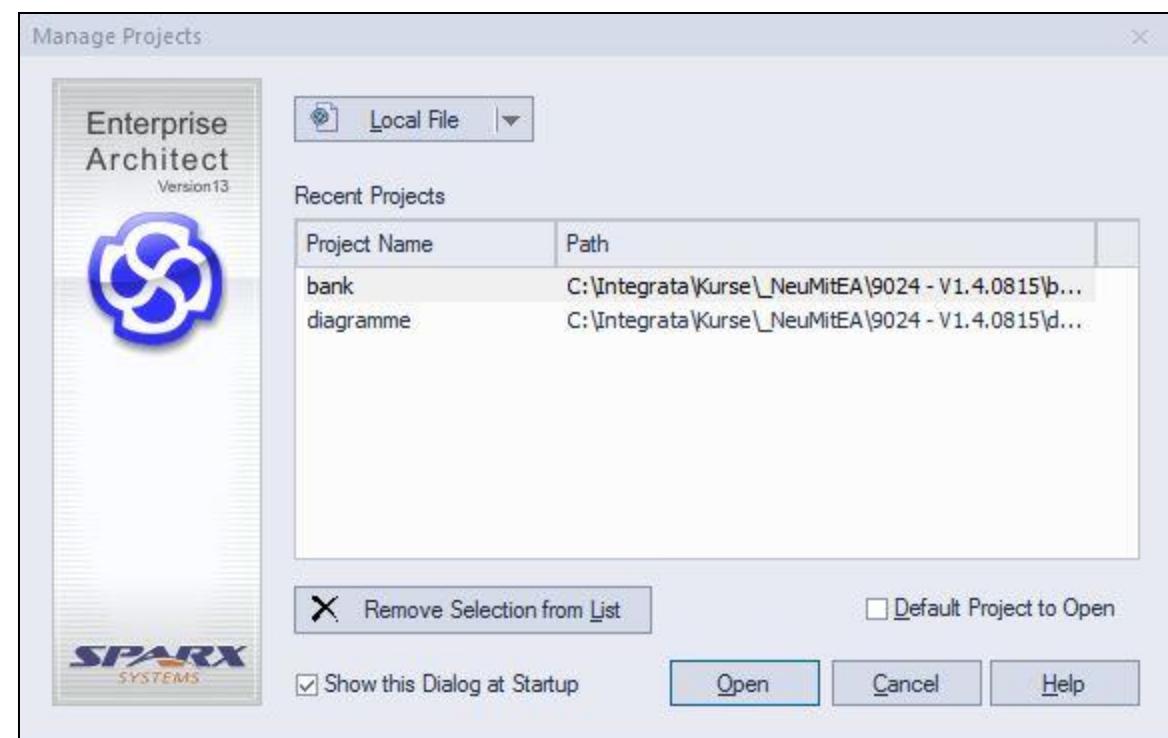
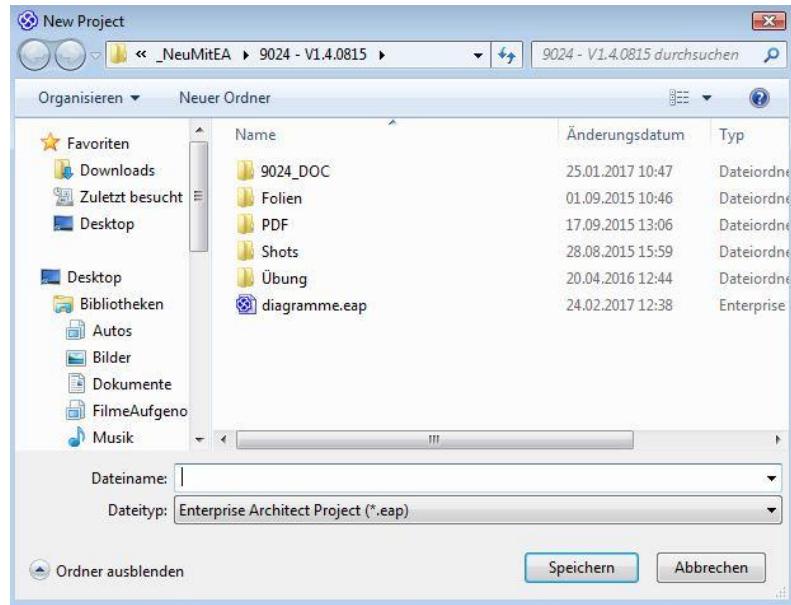
1.2

ERSTES ARBEITEN

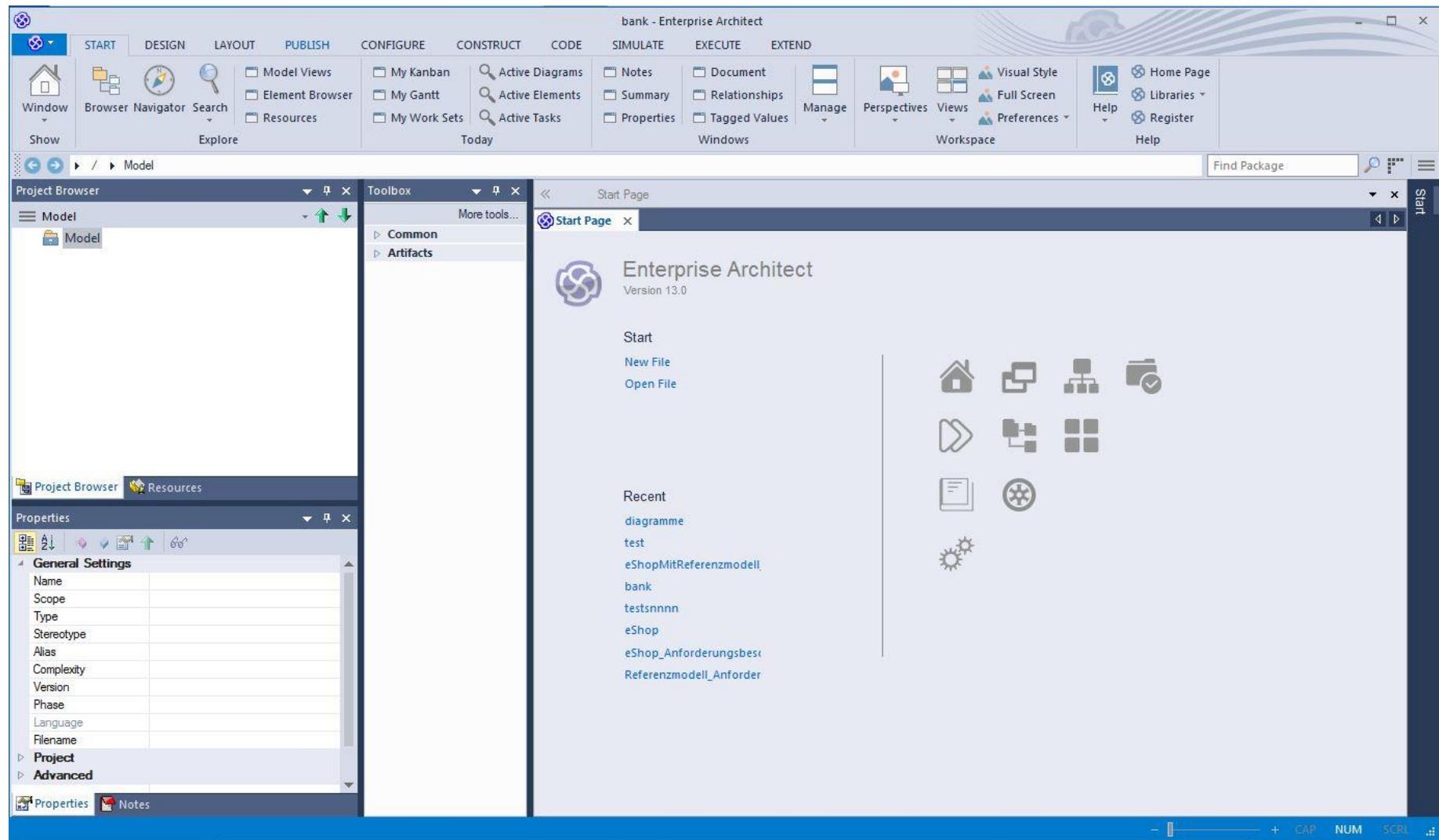


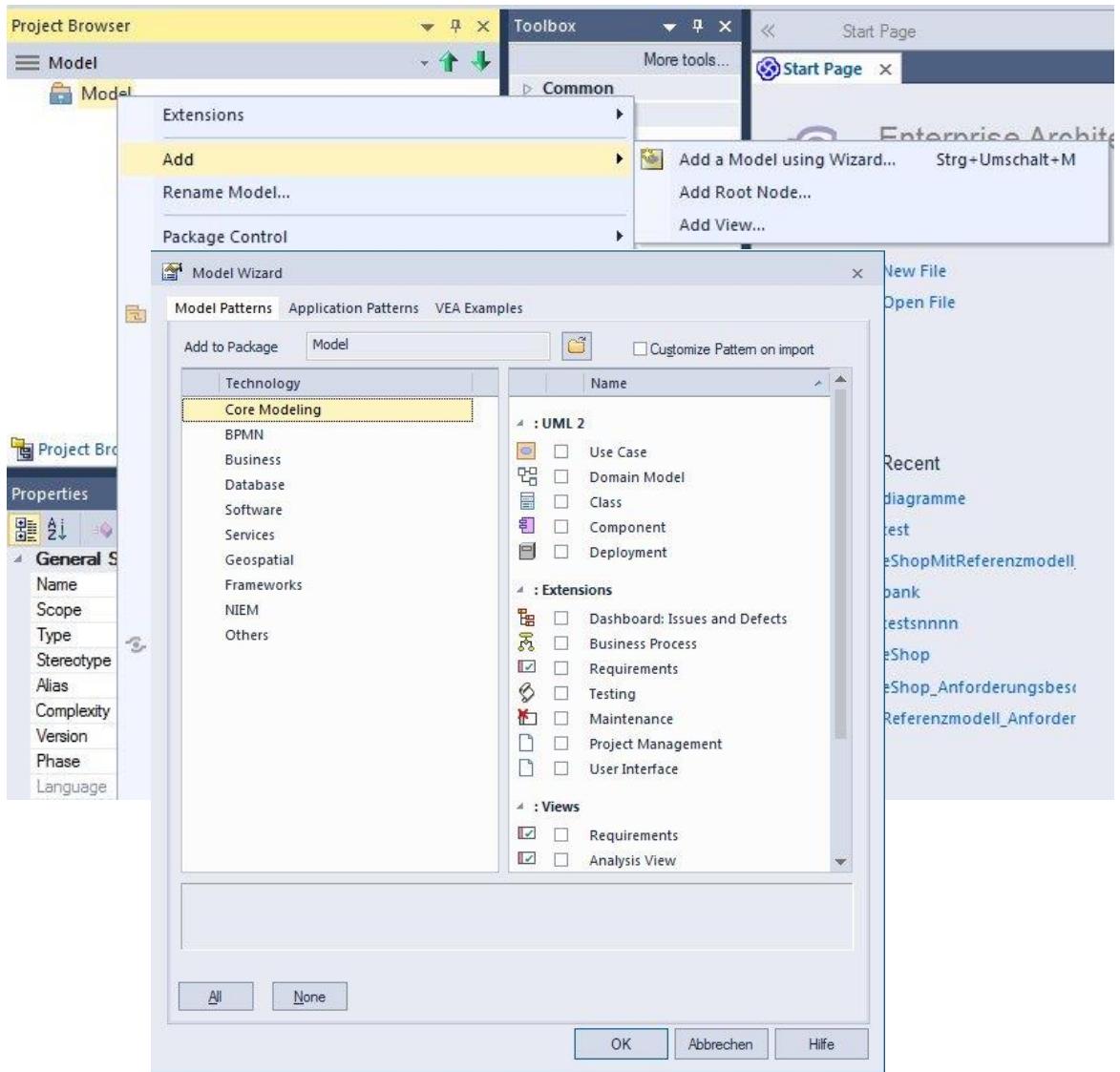
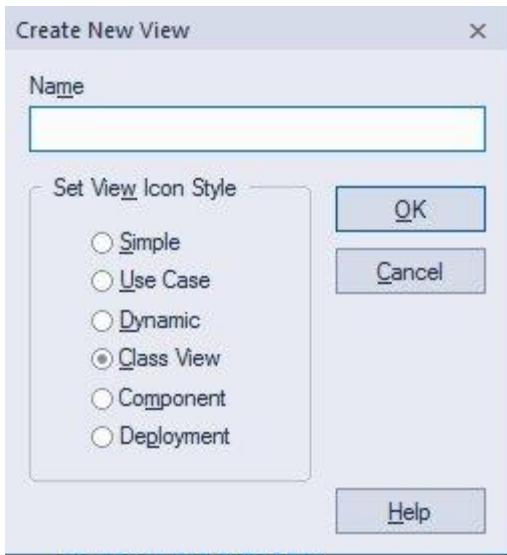
- Manage Projects
 - Open
 - Create
 - Copy
- Recent Liste kann nur über File -> Open Project geändert werden

Anlegen von Projekten



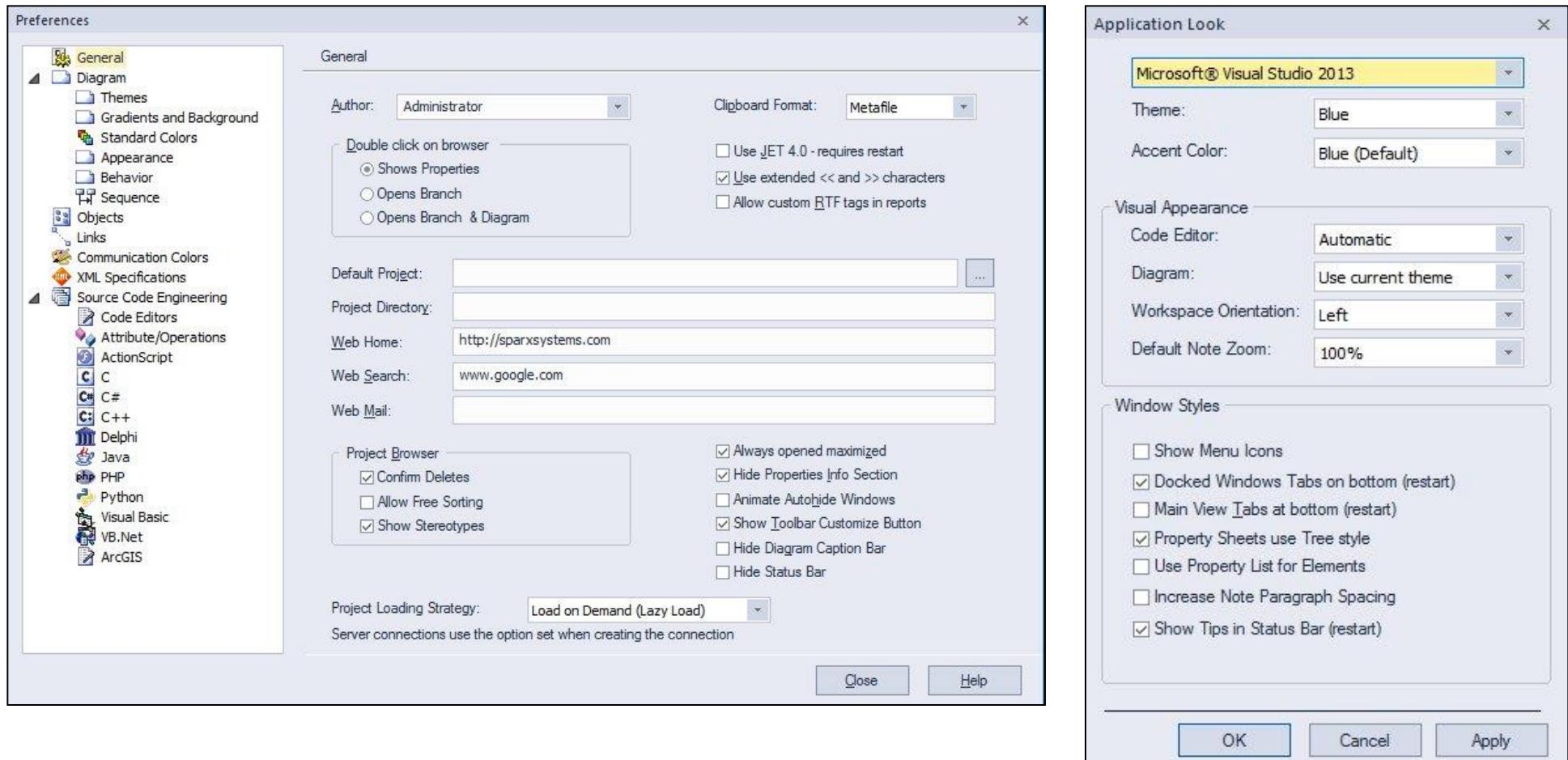
Aufbau der Oberfläche





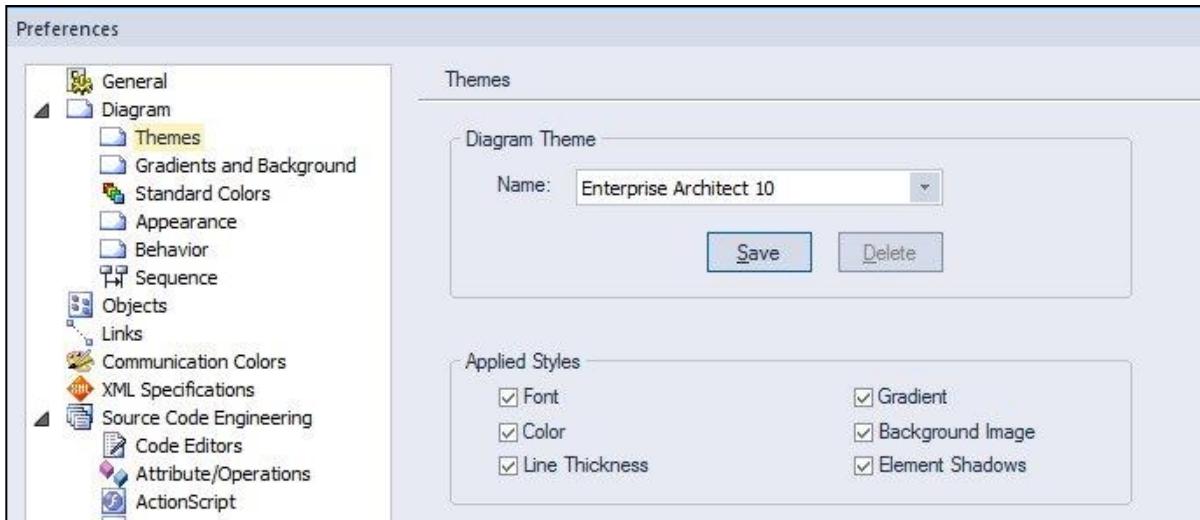
- Im Project Browser muss vor Beginn eine „ordentliche“ Struktur definiert werden
- Zwei Möglichkeiten
 - Wizard
 - Über Views manuell

Voreinstellungen



The image shows two overlapping dialog boxes from Enterprise Architect (EA). The left dialog is titled 'Preferences' and has a 'General' tab selected. It contains fields for 'Author' (set to 'Administrator'), 'Clipboard Format' (set to 'Metafile'), and 'Double click on browser' options ('Shows Properties' is selected). Below these are fields for 'Default Project', 'Project Directory', 'Web Home', 'Web Search', and 'Web Mail'. Under 'Project Browser', there are checkboxes for 'Always opened maximized', 'Confirm Deletes', 'Allow Free Sorting', and 'Show Stereotypes'. At the bottom, 'Project Loading Strategy' is set to 'Load on Demand (Lazy Load)'. The right dialog is titled 'Application Look' and shows 'Microsoft® Visual Studio 2013' selected. It includes sections for 'Theme' (set to 'Blue'), 'Accent Color' (set to 'Blue (Default)'), 'Visual Appearance' (with 'Code Editor' set to 'Automatic' and 'Diagram' set to 'Use current theme'), and 'Window Styles' (with various checkboxes like 'Show Menu Icons' and 'Docked Windows Tabs on bottom (restart)').

Voreinstellungen für EA über Start Register -> Visual Style bzw. Preferences



- Seit EA 11 möglich
 - Beispiele:
 - EA11
 - EA10
 - Ocean Breeze

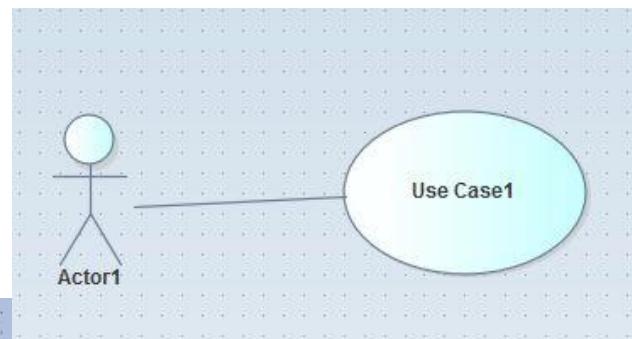
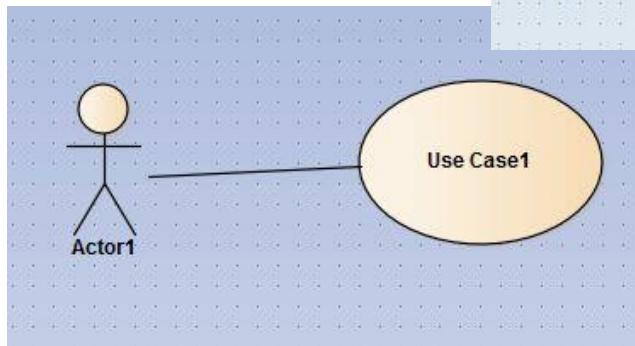
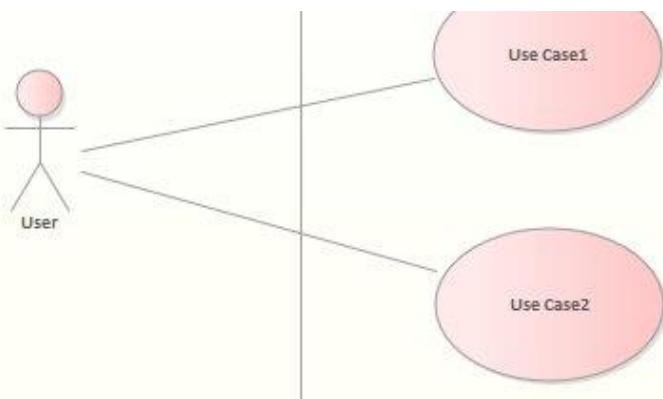
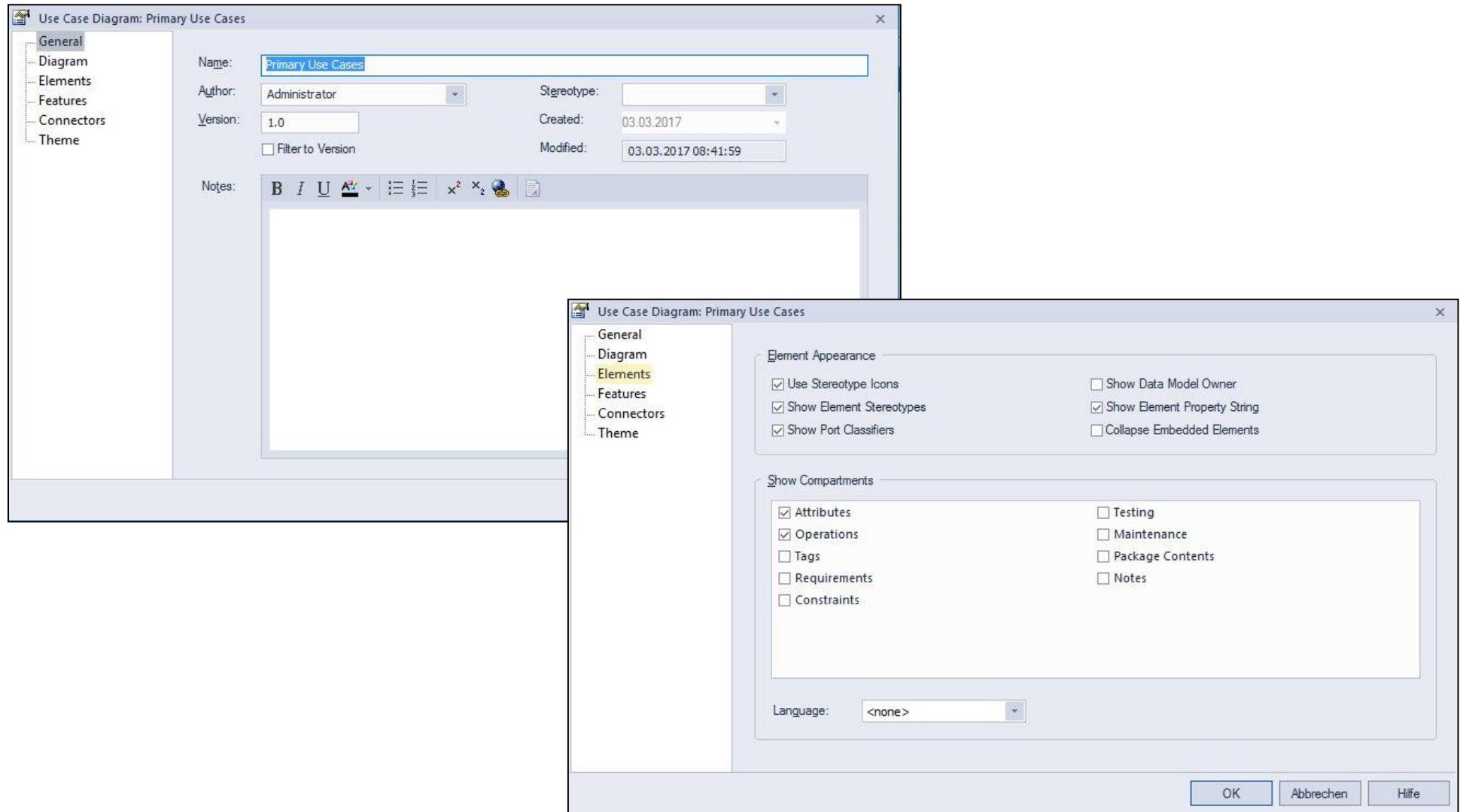
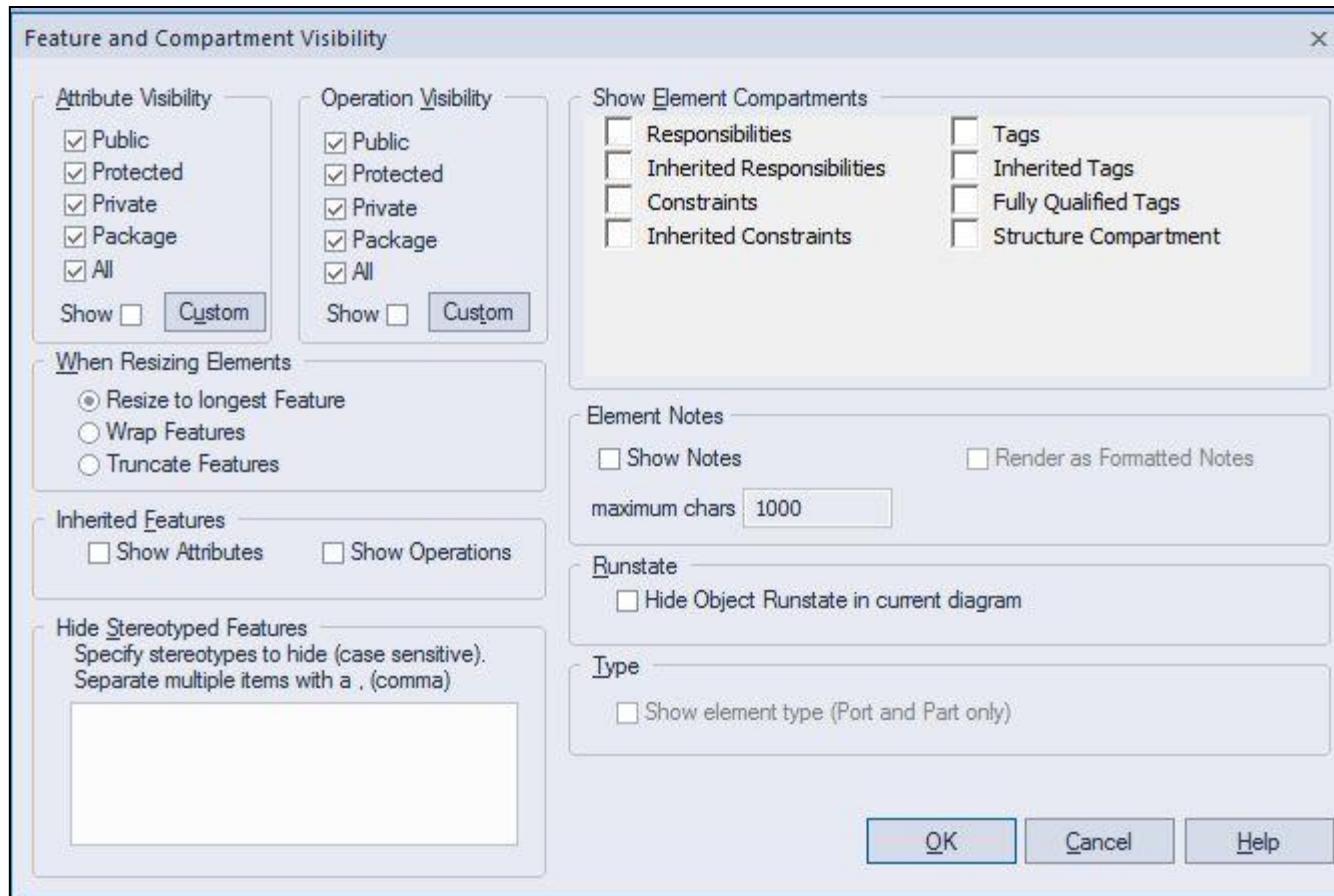


Diagramm Einstellungen



- Diagramm und Element Einstellungen können sich überlagern



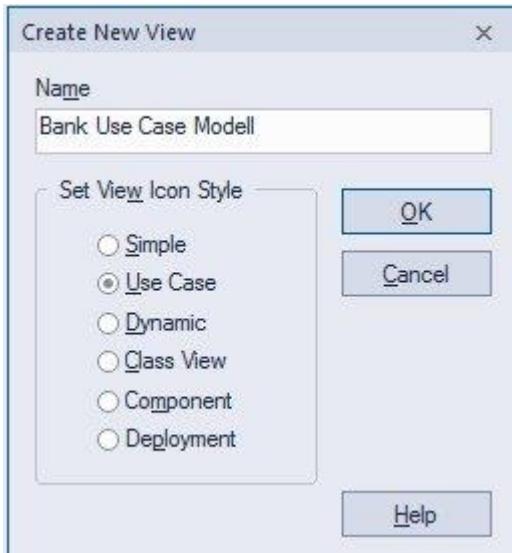
- UML Werkzeuge (hier EA) können meist mehr als der UML-Standard
 - UML ist OMG Standard (Siehe www.omg.org)
- Nicht alle Elemente in einer Toolbox sind UML-Elemente
- Teilweise sinnvolle Ergänzungen
- Teilweise nicht so sinnvolle Ergänzungen
- Ergänzungen benötigen immer eine Erklärung (Interpretation)
- Vorgehensweise:
 - Erst UML lernen
 - Dann UML Elemente mit dem Werkzeug umsetzen

2

ANFORDERUNGSANALYSE

2.1

USE CASES



- Strukturen anlegen:
 - manuell
 - Wizard

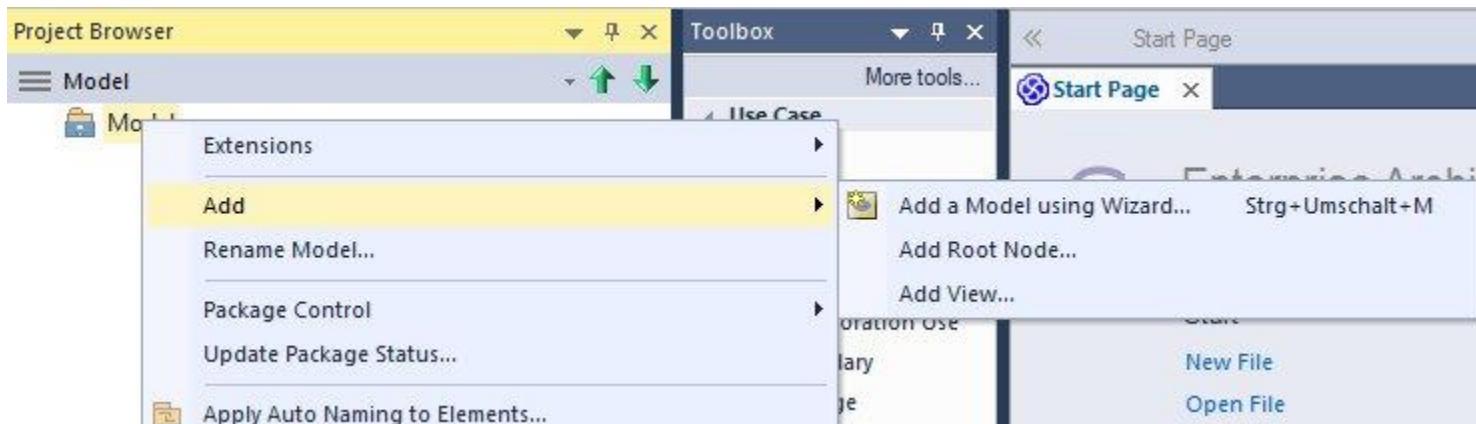
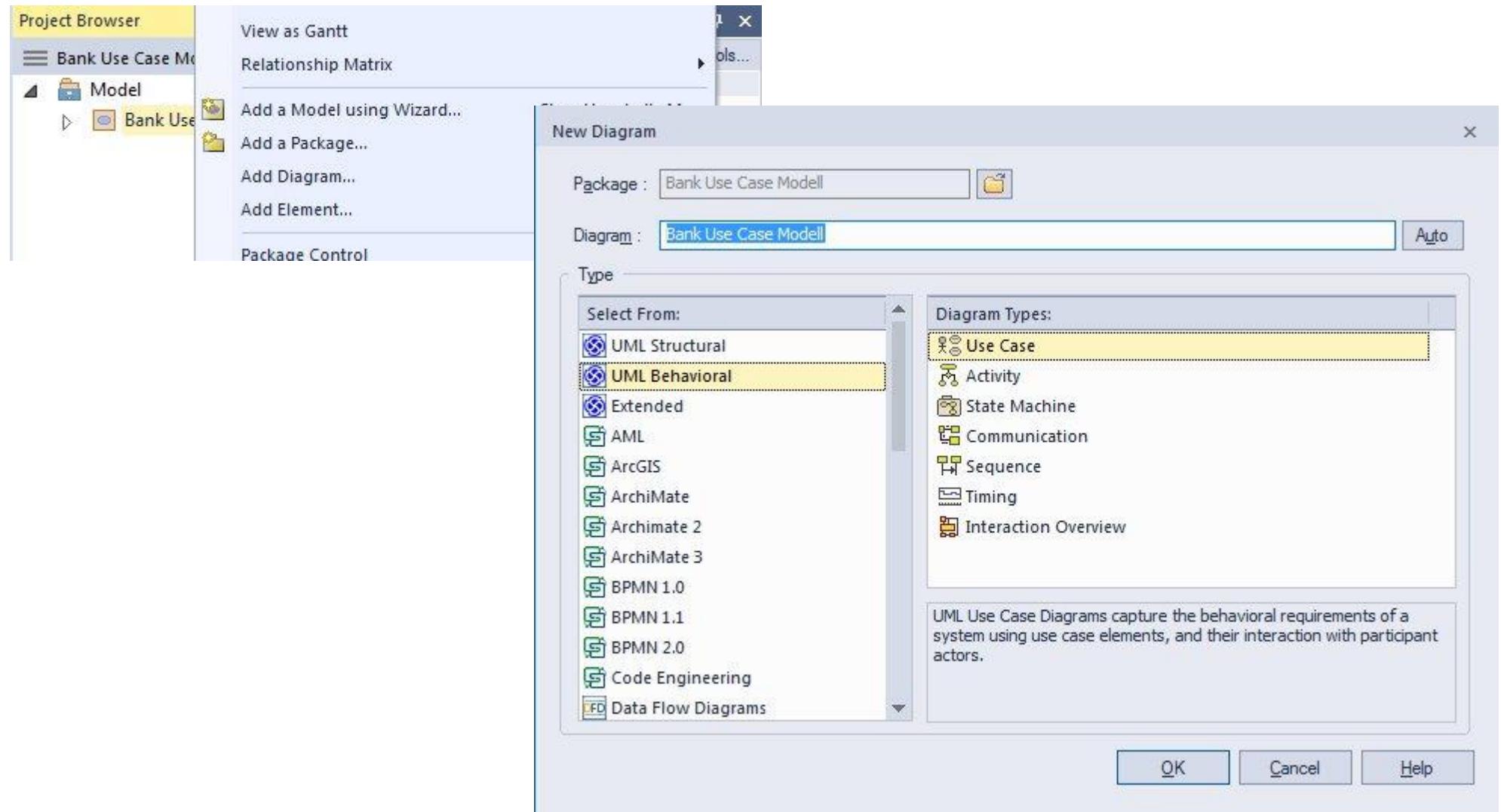
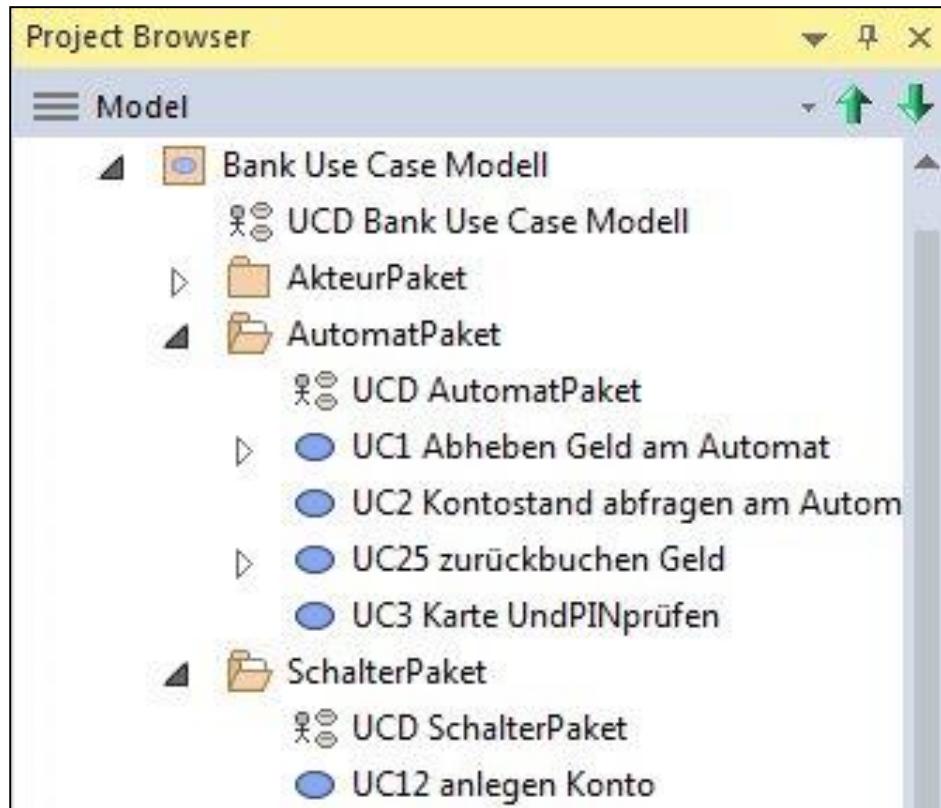


Diagramme zum Modell hinzufügen

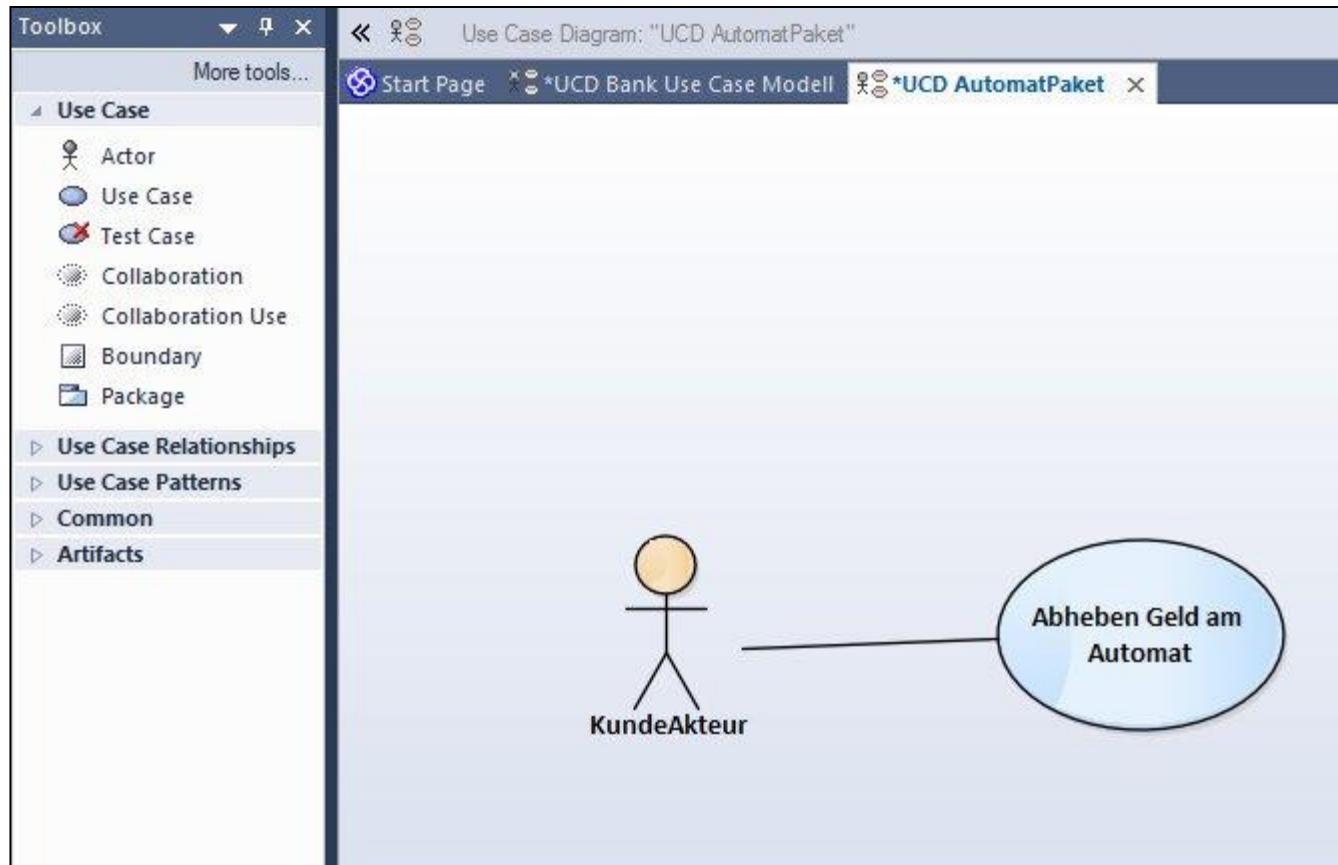


Mögliche fertige Grund-Struktur im Project Browser



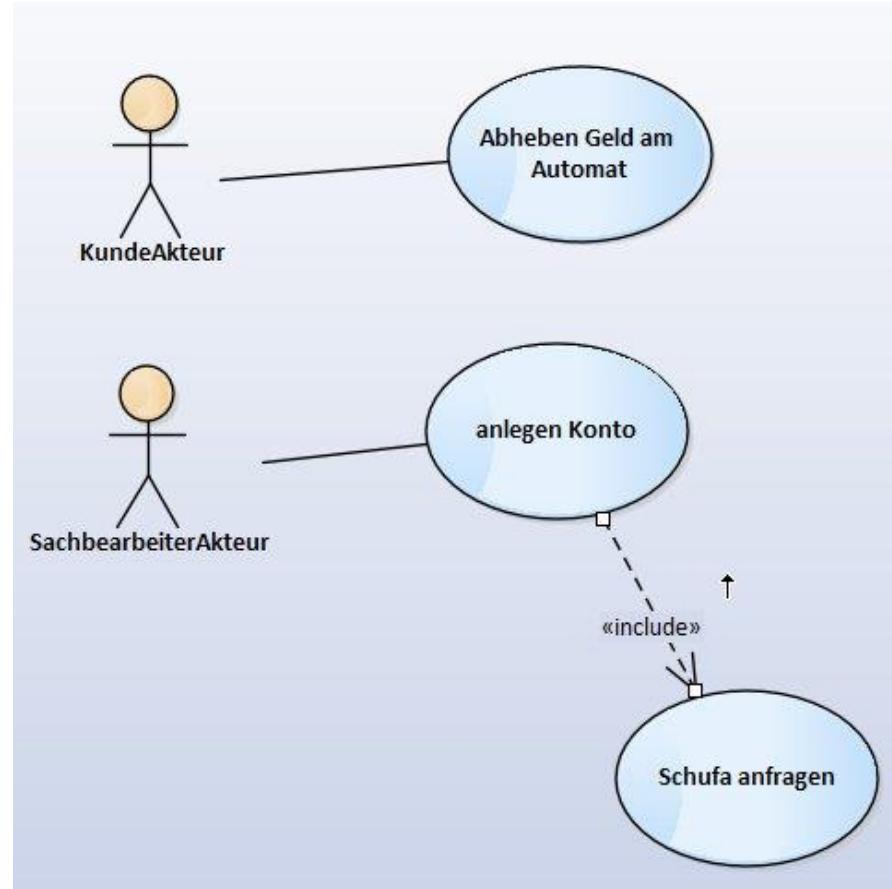
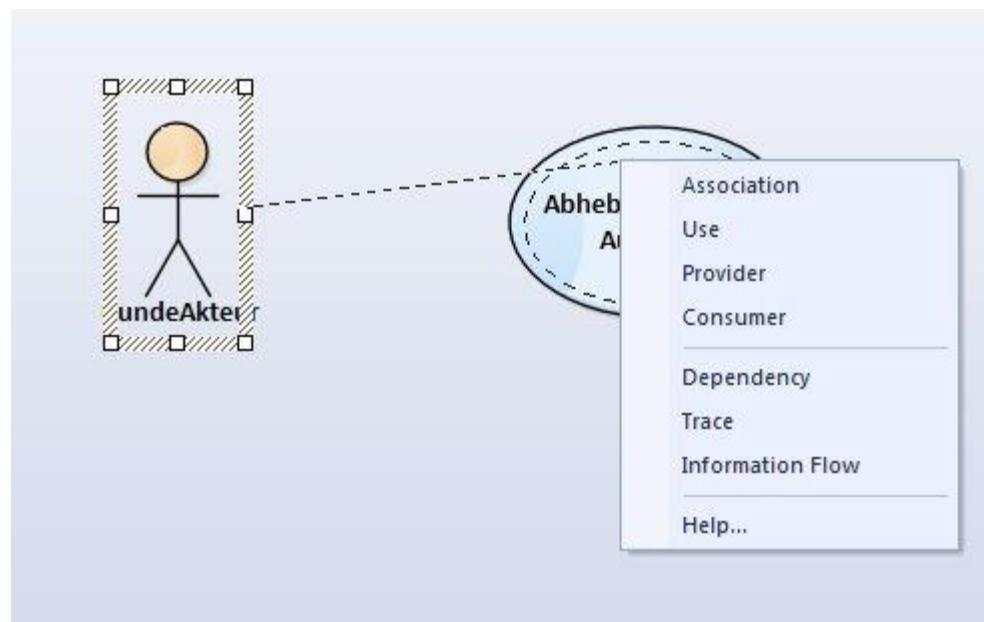
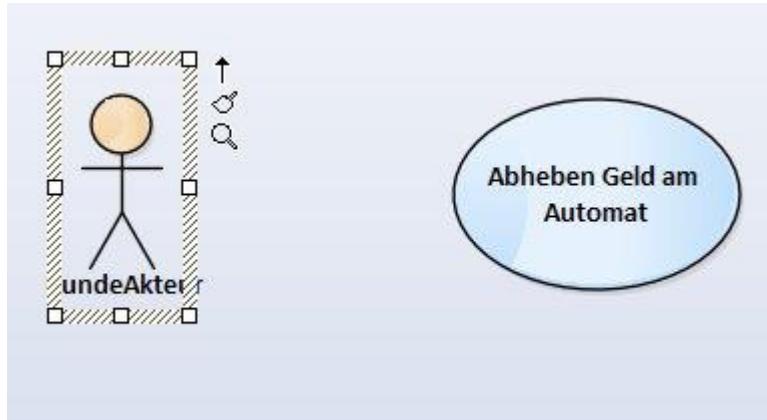
- Die Strukturen müssen vor der Modellierung erstellt werden bzw. können auch nachträglich angepasst werden
- Erste, einfache Struktur ist ohne Pakete
- Pakete sollten auf Dauer hinzugefügt werden

Erstellen des Use Case Diagramms (1)



- Use Case Diagramme lassen sich auf verschiedene Art anlegen
- Toolbox
- Werkzeugeleiste
- Aufs Element klicken

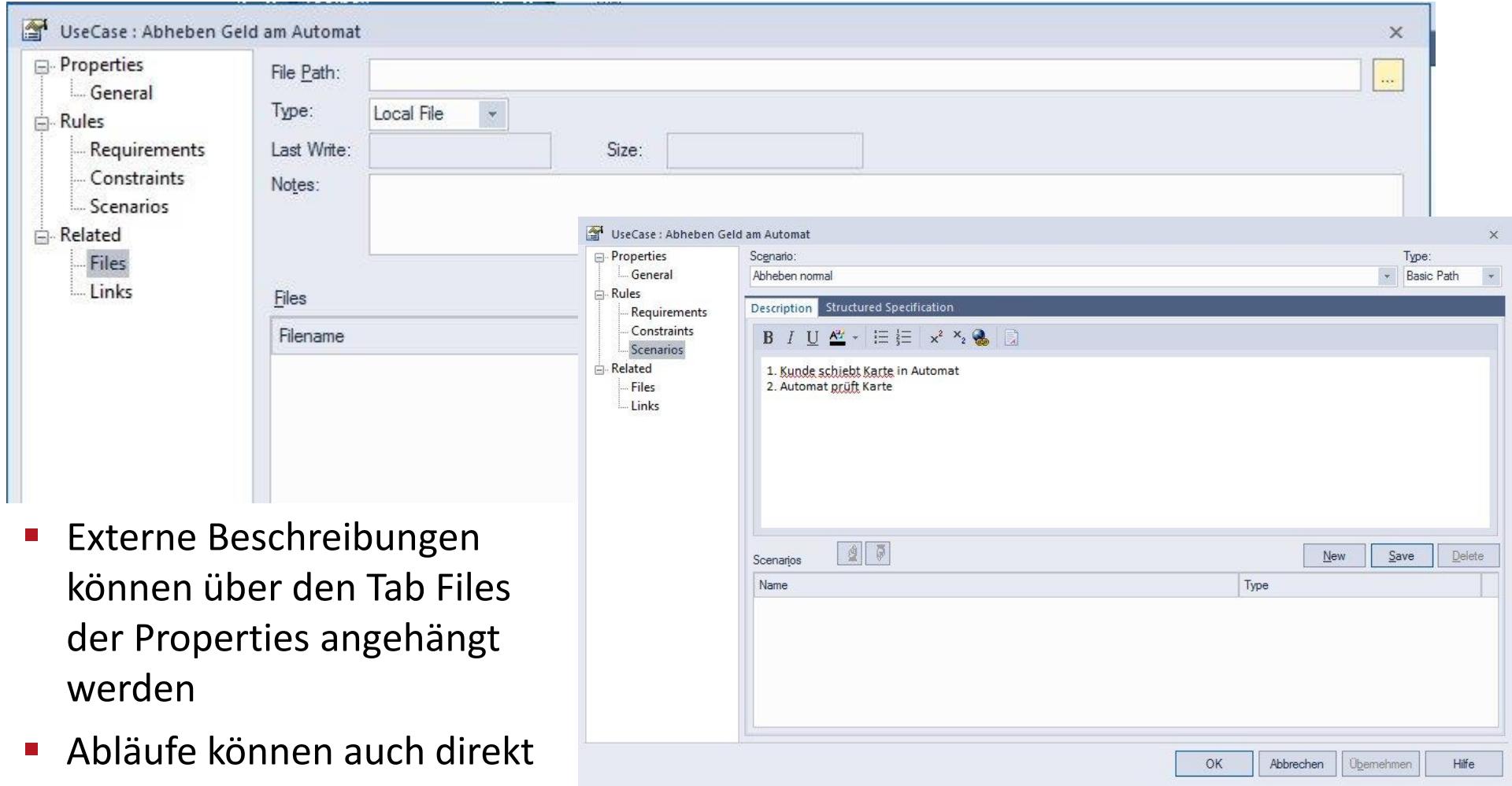
Erstellen des Use Case Diagramms (2)



- Die Beschreibung des Use Cases kann auf verschiedene Arten und Weisen erfolgen:
 - Textform
 - Prosa
 - Schablone
 - EA
 - Ergänzend können folgende Diagramme verwendet werden:
 - Aktivitätsdiagramm
 - Sequenzdiagramm

(Beide Diagramme beschreiben hier nur die Interaktion des Akteurs mit dem System)

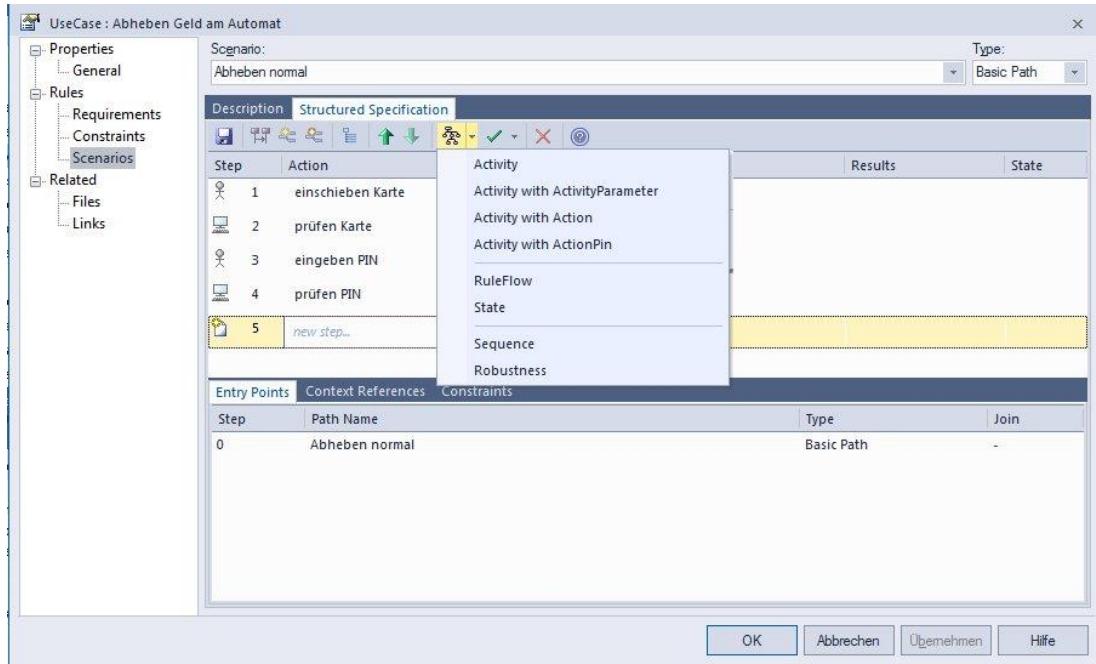
Beschreibung der Use Cases in Textform



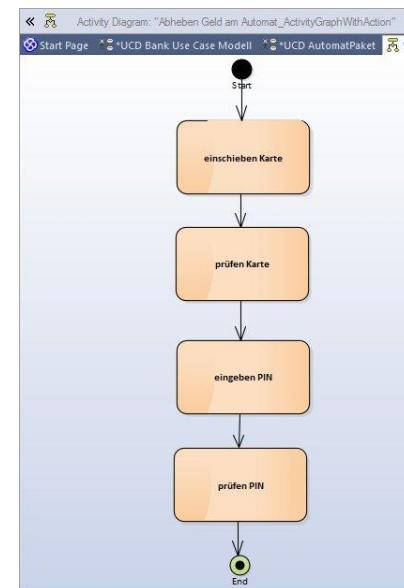
The screenshot shows two windows from the Enterprise Architect tool. The main window is titled "UseCase : Abheben Geld am Automat". It has a tree view on the left with nodes for Properties (General), Rules (Requirements, Constraints, Scenarios), and Related (Files, Links). Under the "Files" node, there is a "Files" tab containing a table with one row labeled "Filename". The "Properties" tab contains fields for File Path (empty), Type (Local File), Last Write (empty), and Notes (empty). The "Rules" tab has sections for Requirements, Constraints, and Scenarios. The "Related" tab has sections for Files and Links. A secondary window titled "Scenario: Abheben normal" is open, showing a "Description" tab with the text "1. Kunde schiebt Karte in Automat" and "2. Automat prüft Karte". The "Type" dropdown in this window is set to "Basic Path". The bottom right of the interface shows standard buttons: OK, Abbrechen, Übernehmen, and Hilfe.

- Externe Beschreibungen können über den Tab Files der Properties angehängt werden
- Abläufe können auch direkt im EA beschrieben werden

Strukturierte Beschreibung der Use Cases mit Generierung von Diagrammen

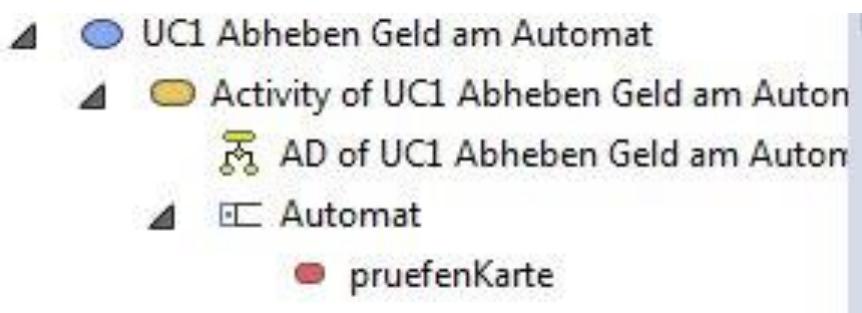


- Abläufe können strukturiert beschrieben werden
- Daraus lassen sich Diagramme generieren
 - Aktivitäts-Diagramme
 - Sequenz-Diagramme
 - Robustness-Diagramme

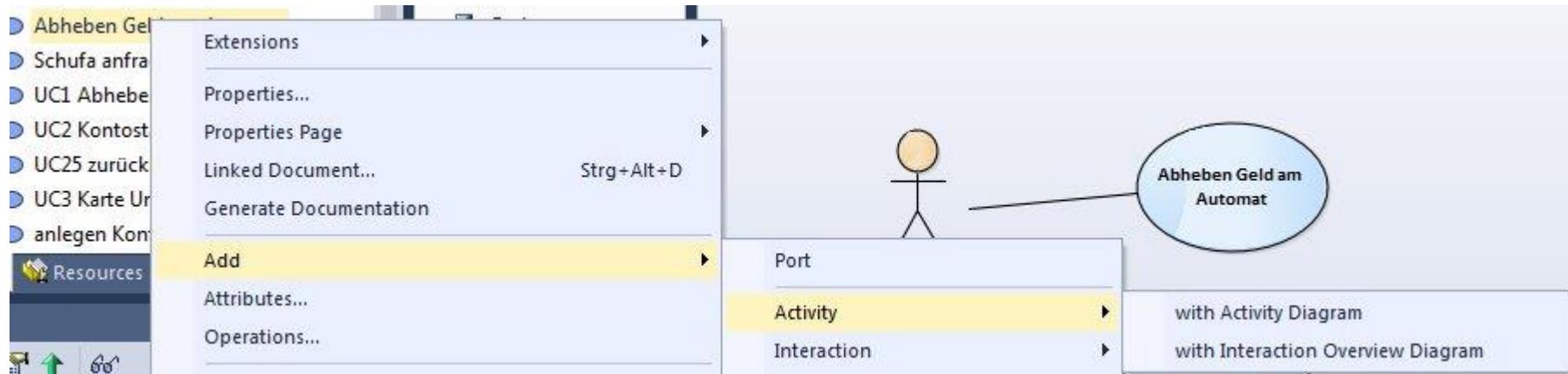


2.2

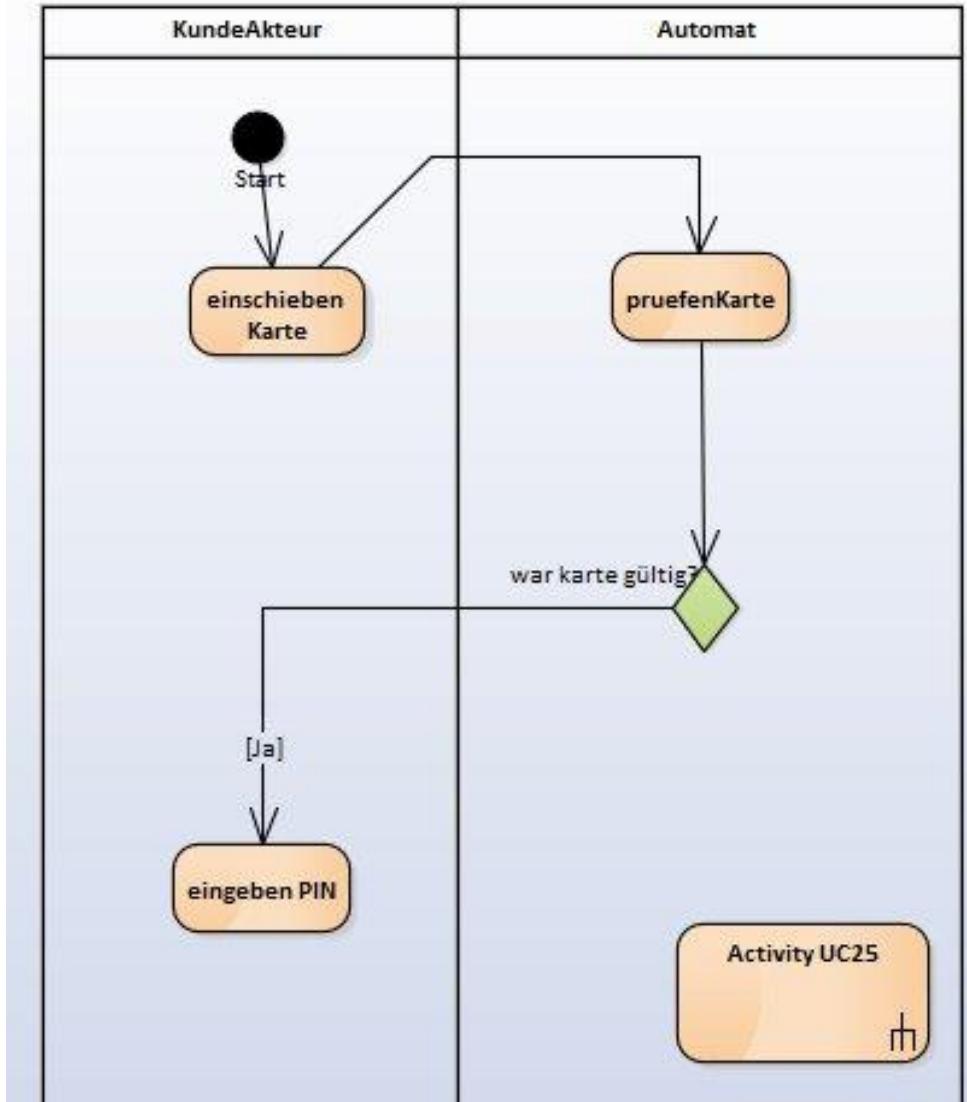
ACTIVITY



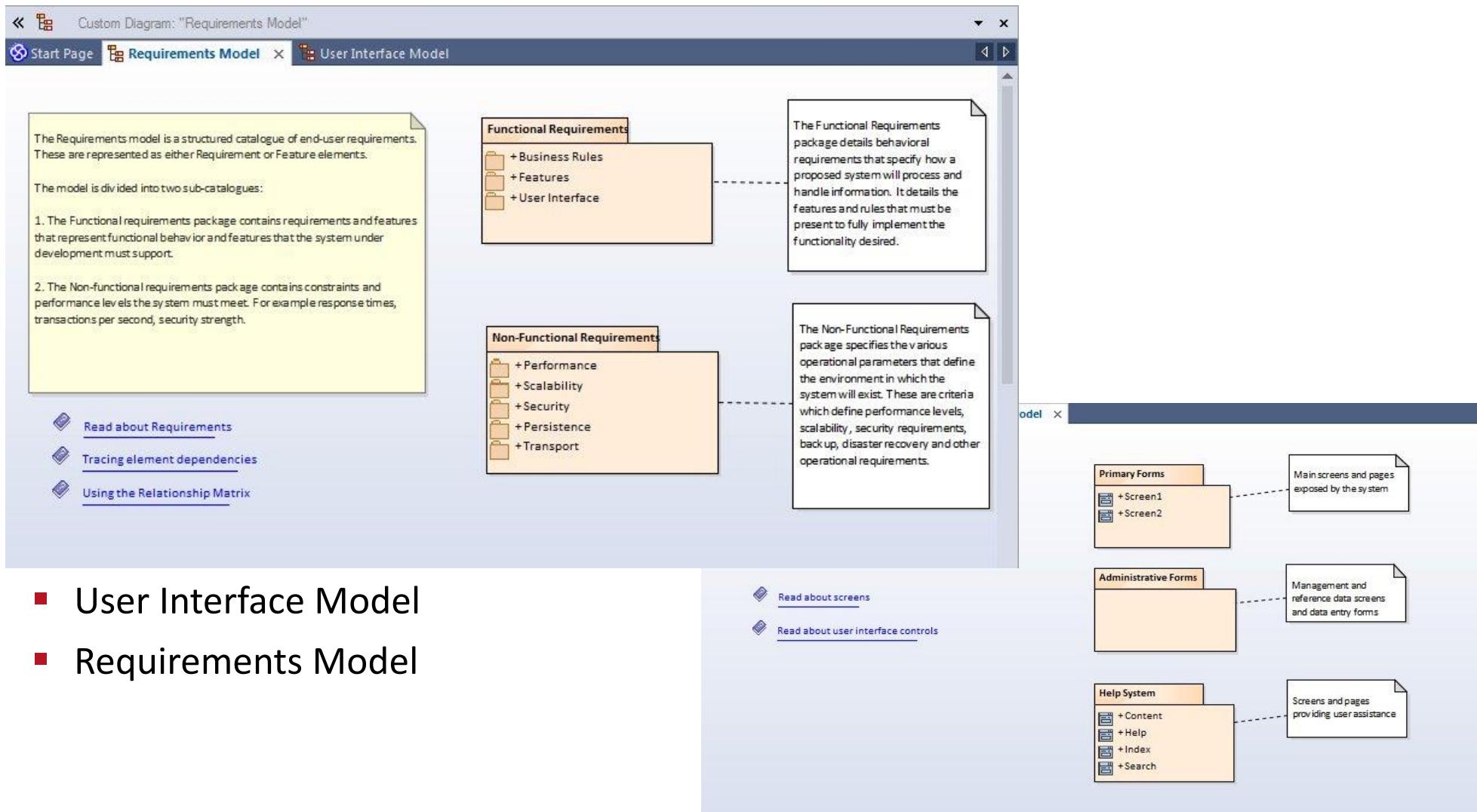
- Anlegen über rechte Maustaste -> New Child Diagram -> Activity
- Die grundlegende Struktur ist wichtig
- Partitions können verwendet werden



Aktivitäts-Diagramme (2)



Weitere Mögliche Modelle (Views) in der Anforderungsanalyse



- User Interface Model
- Requirements Model

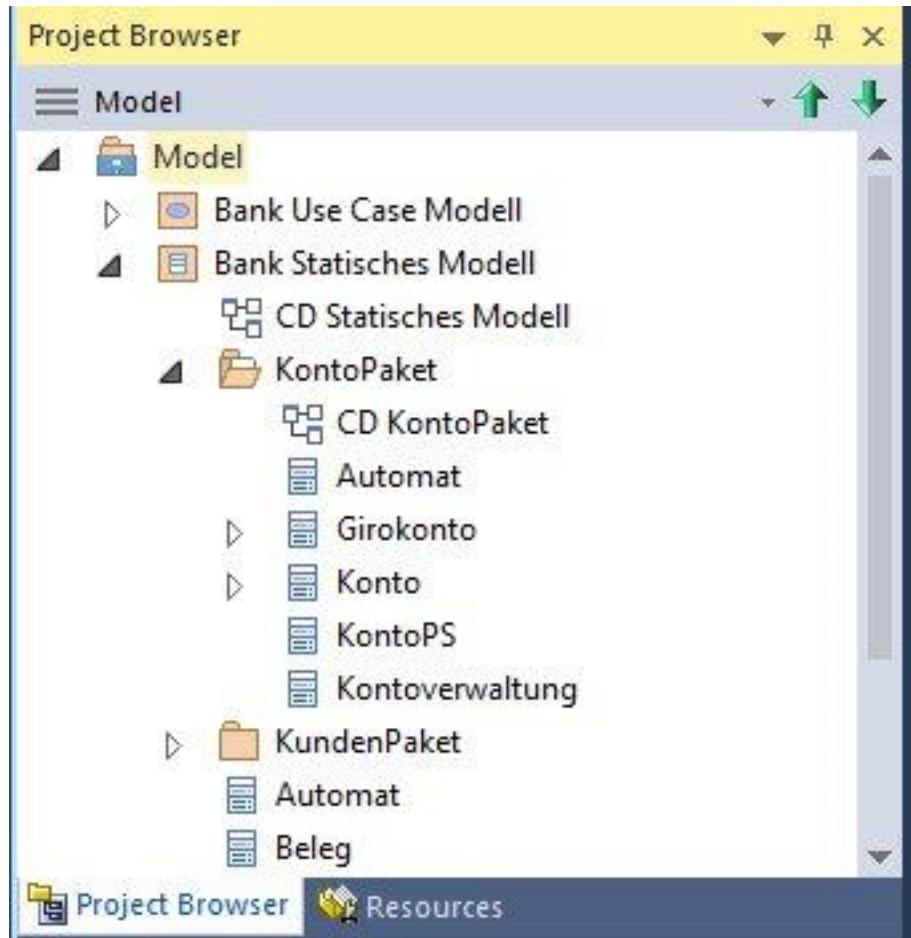
3

DAS FACHLICHE MODELL

3.1

STRUKTUR

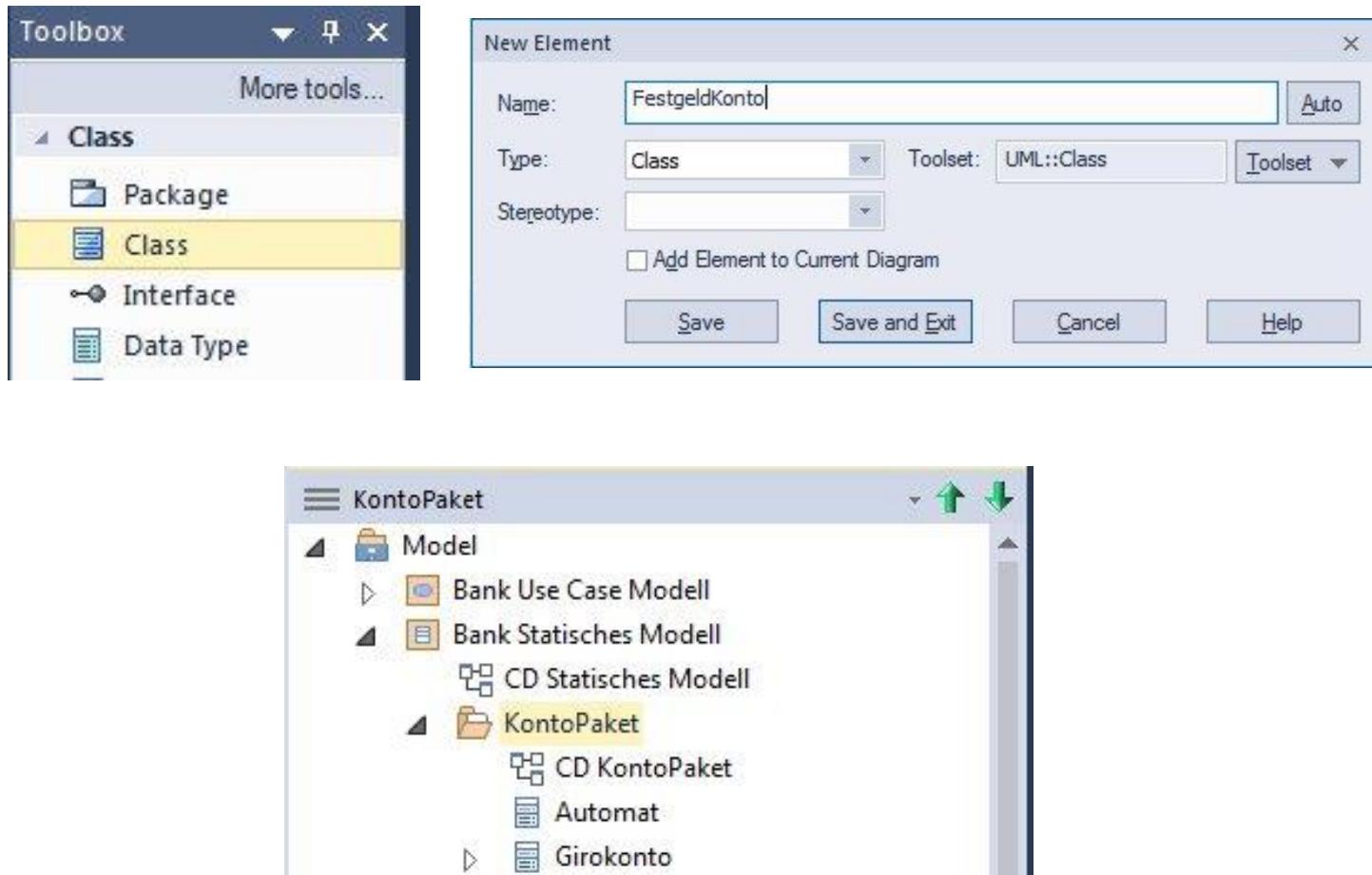
- Beschreibt die fachlichen Zusammenhänge hinter Systemgrenze
- Keine (wenig) technische Details
- Besteht aus statischem und dynamischem Modell
- Statisches Modell
 - Klassen-Diagramm bzw. Abwandlungen (Pakete / Interfaces)
- Dynamisches Modell
 - Interaktions-Diagramme
 - Sequenz-Diagramm
 - Kommunikations-Diagramm
 - (Aktivitäts-Diagramm)
 - Zustands-Diagramm
 - weitere



- Unterschiedliche Strukturen sind möglich

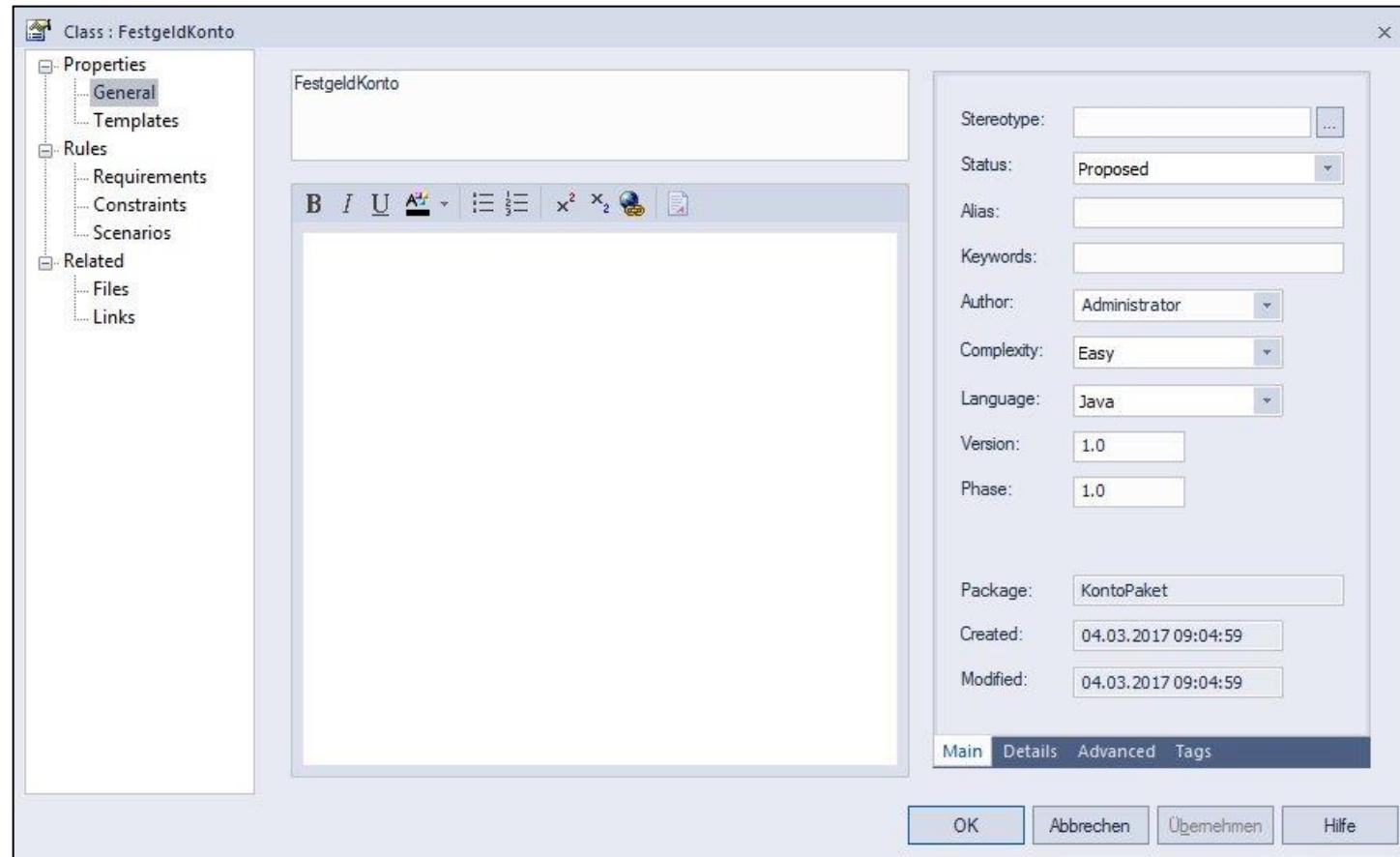
3.2

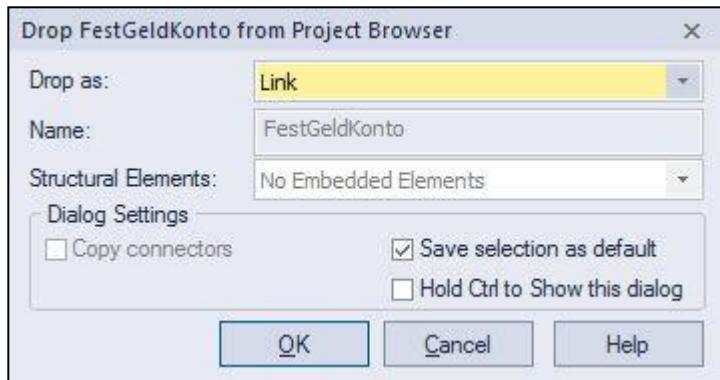
KLASSENDIAGRAMM



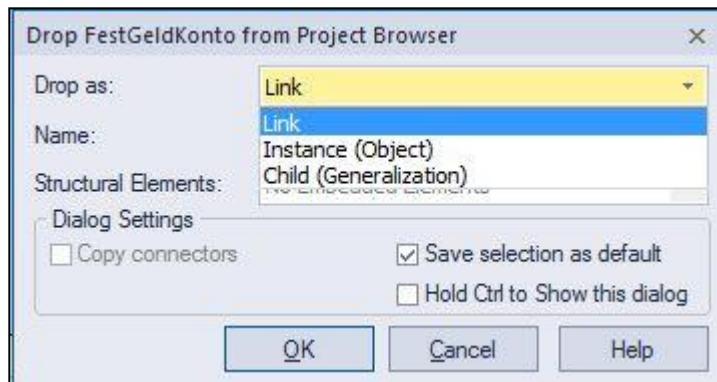
- Das Erzeugen von Klassen kann auf verschiedene Arten und Weisen erfolgen:
 - Toolbox
 - Über die Struktur
 - Toolset auswählen

- Jede Klasse kann über die Properties und den einzelnen Tabs weiter detailliert werden

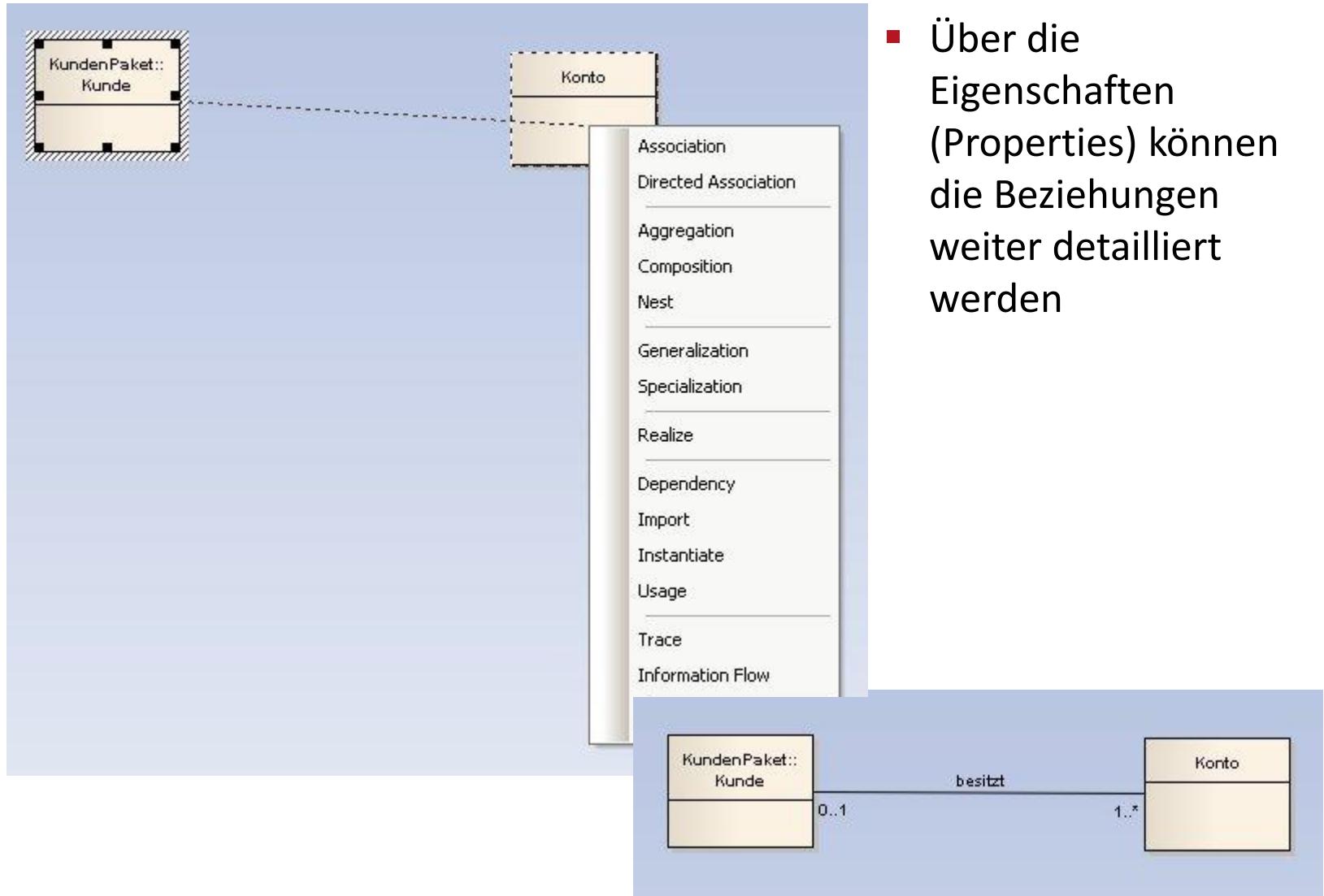




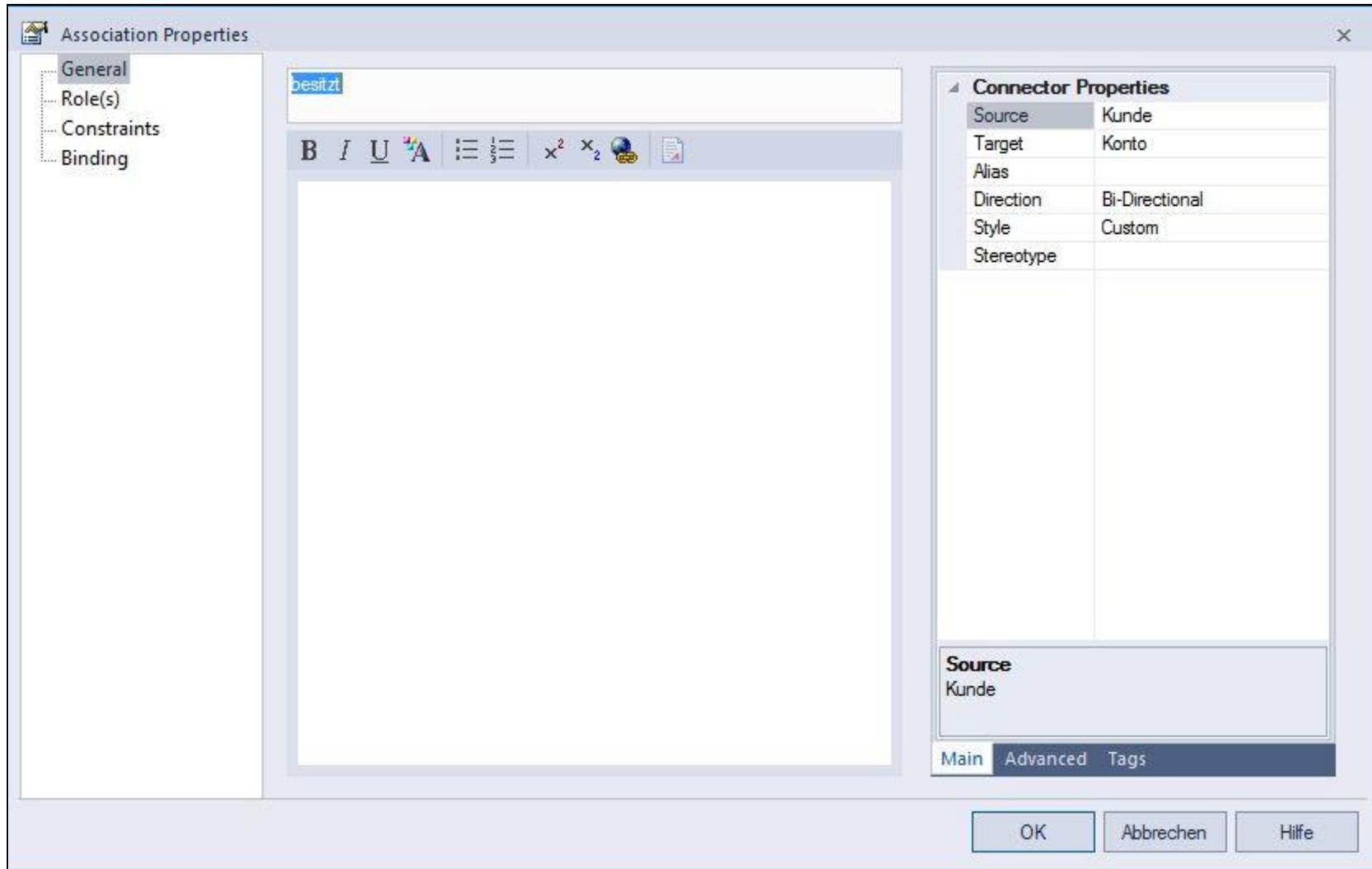
- Elemente (Klassen, Use Cases, Akteure etc. lassen sich aus dem Project Browser in das entsprechende Diagramm ziehen
- Verschiedene Optionen sind beim Ziehen möglich



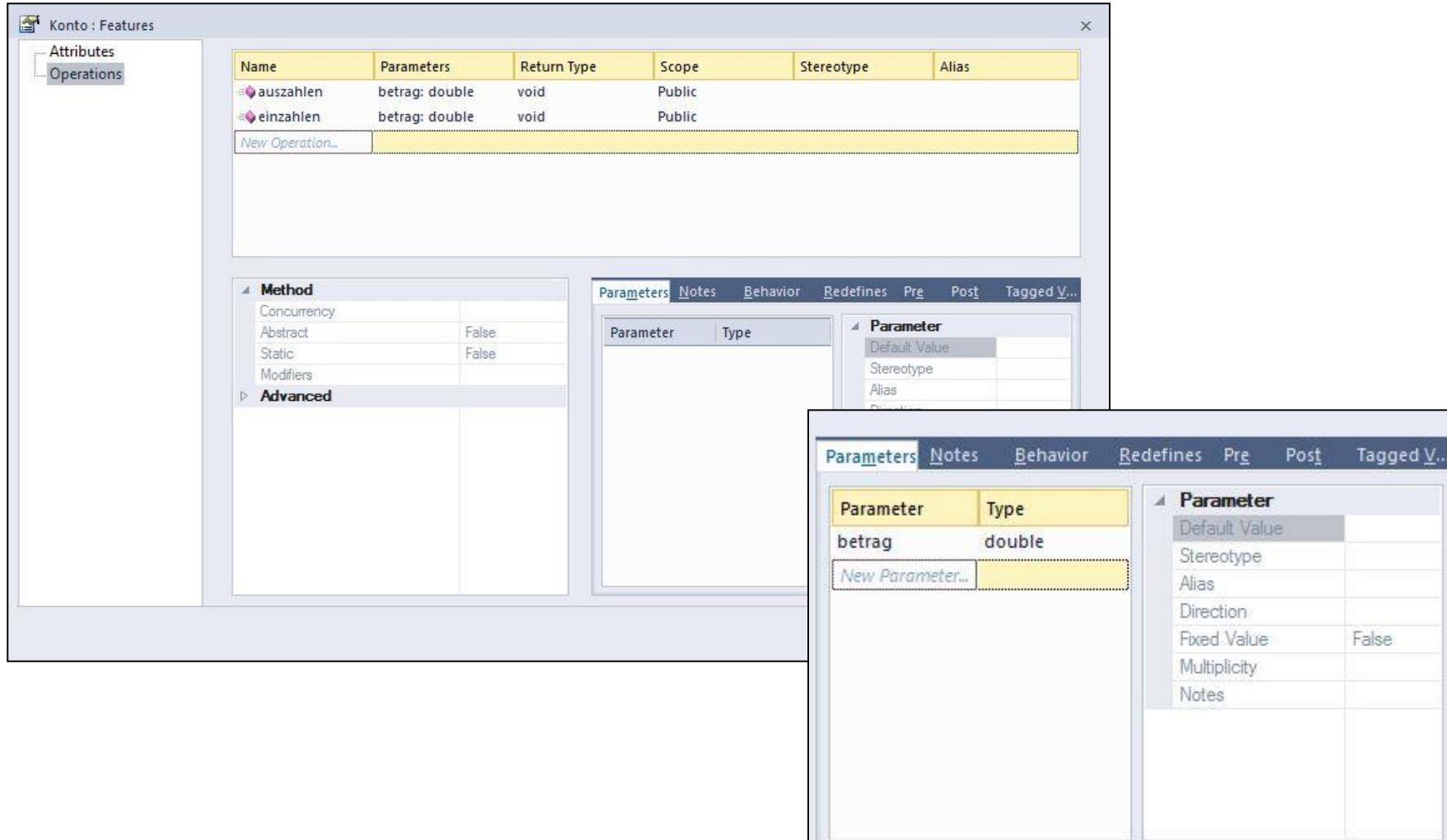
Anlegen von Beziehungen im Klassen-Diagramm



Attribute für die Instanzen festlegen



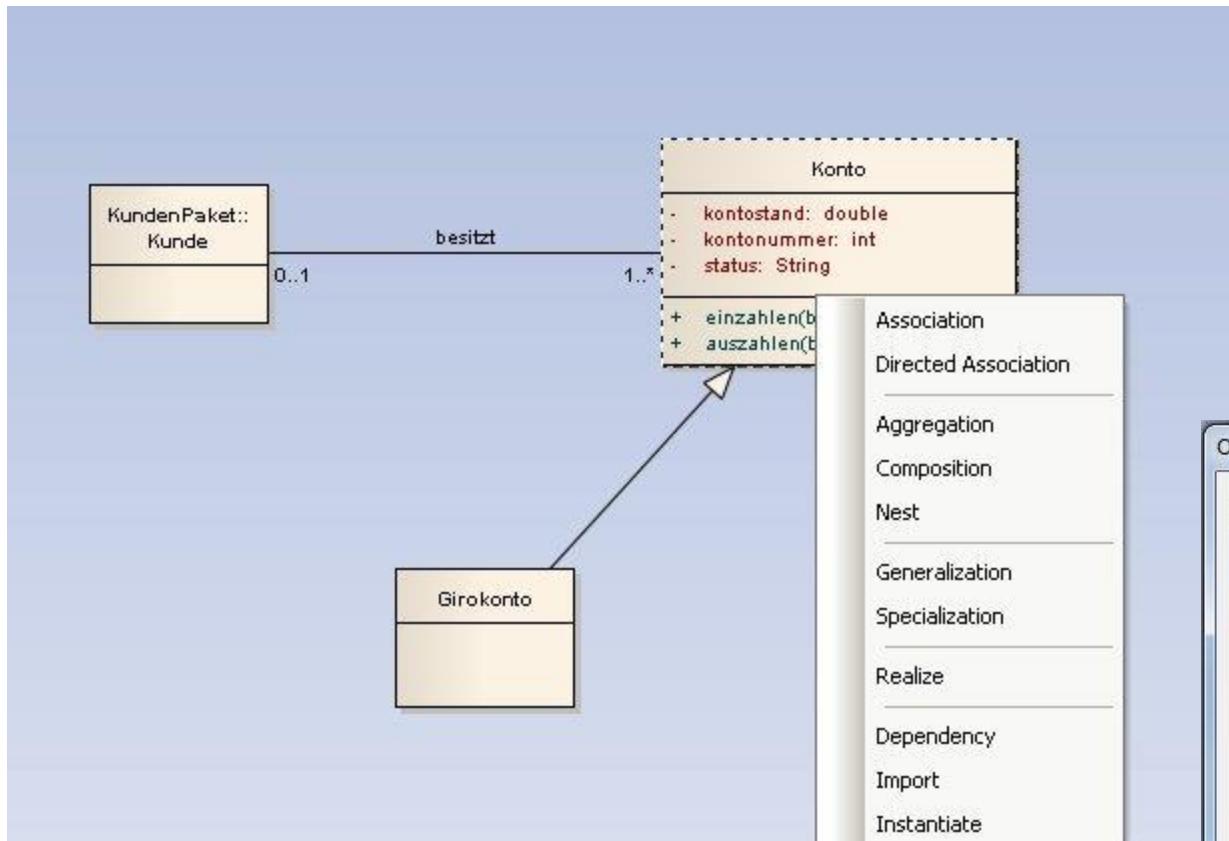
Operationen und Parameter festlegen



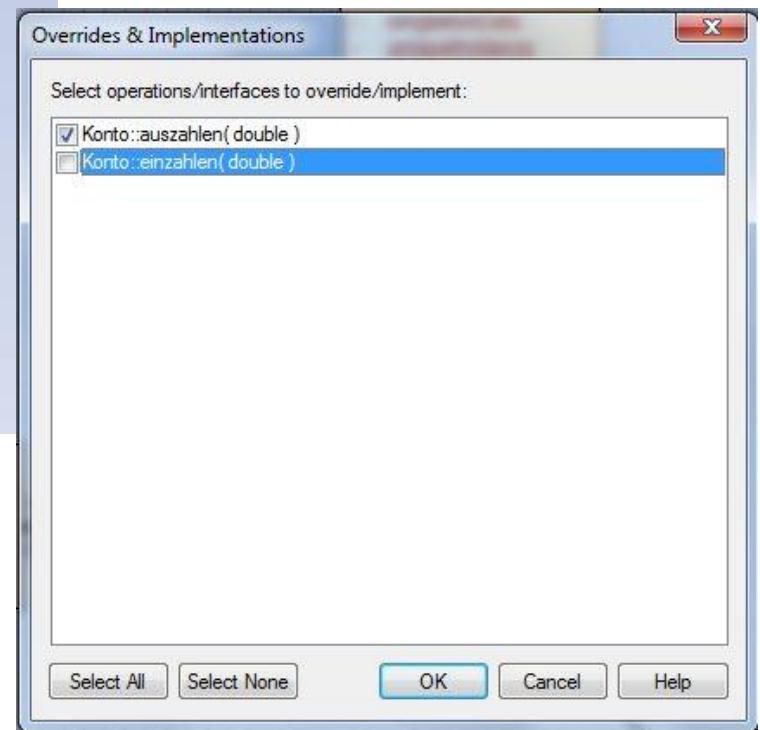
The screenshot shows the 'Operations' tab of the 'Features' dialog for a 'Konto' feature. It lists two operations: 'auszahlen' and 'einzahlen', both taking a 'betrag: double' parameter and returning 'void', with 'Public' scope.

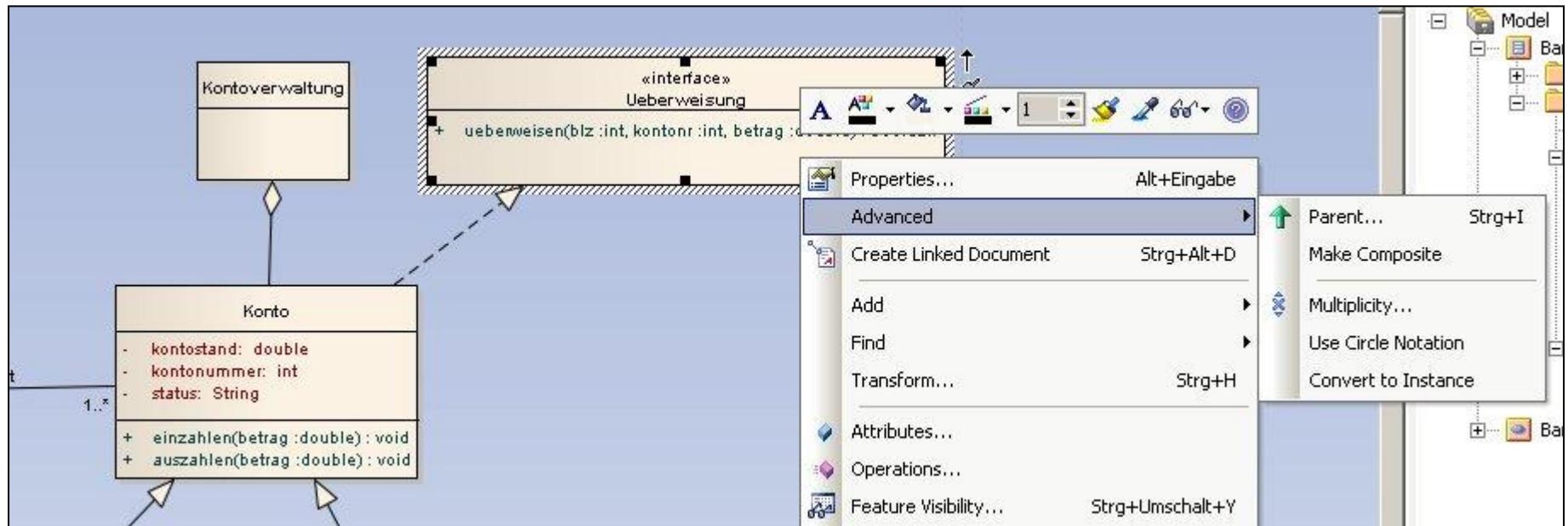
A detailed view of the 'auszahlen' operation is shown in a modal dialog. The 'Method' tab shows 'Concurrency' and 'Modifiers'. The 'Parameters' tab shows a single parameter named 'betrag' of type 'double'. The 'Parameter' details pane shows fields for 'Default Value', 'Stereotype', 'Alias', and 'Direction'.

A second modal dialog is open, showing the creation of a new parameter for another operation. This dialog has tabs for 'Parameters', 'Notes', 'Behavior', 'Redefines', 'Pre', 'Post', and 'Tagged V...'. The 'Parameters' tab lists a new parameter named 'betrag' of type 'double'. The 'Parameter' details pane shows fields for 'Default Value', 'Stereotype', 'Alias', 'Direction', 'Fixed Value' (set to 'False'), 'Multiplicity', and 'Notes'.



- Bei der Vererbung können die Operationen, die Überladen oder Überschrieben werden sollen, ausgewählt werden

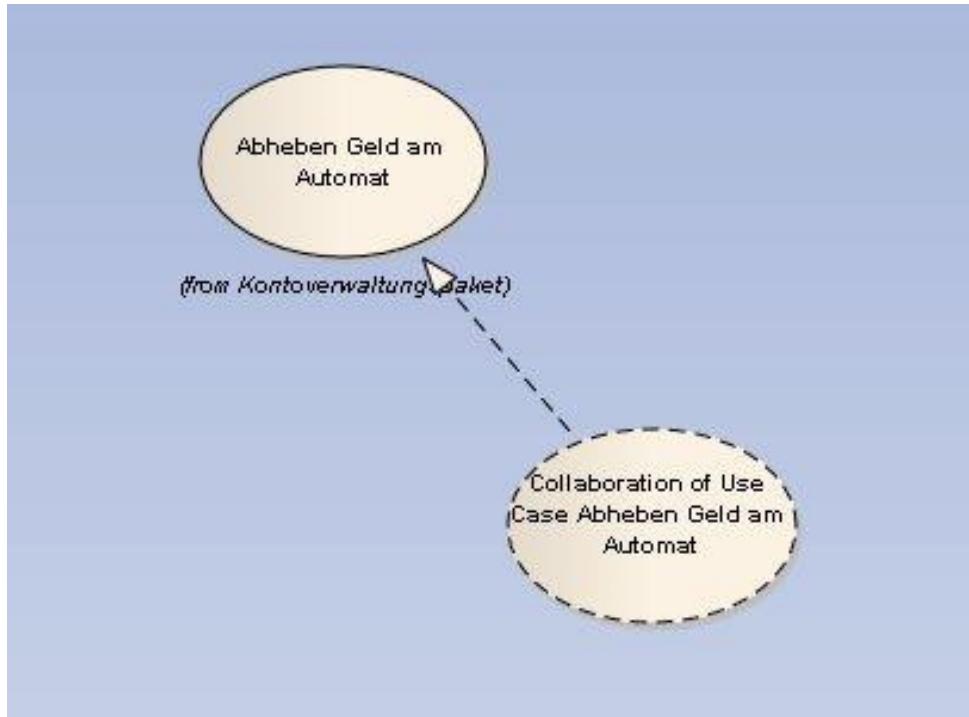




- Neben den Assoziationen lassen sich auch Interfaces mit Realize verbinden
- Aggregationen und Kompositionen können direkt oder über die Properties der Assoziation geändert werden

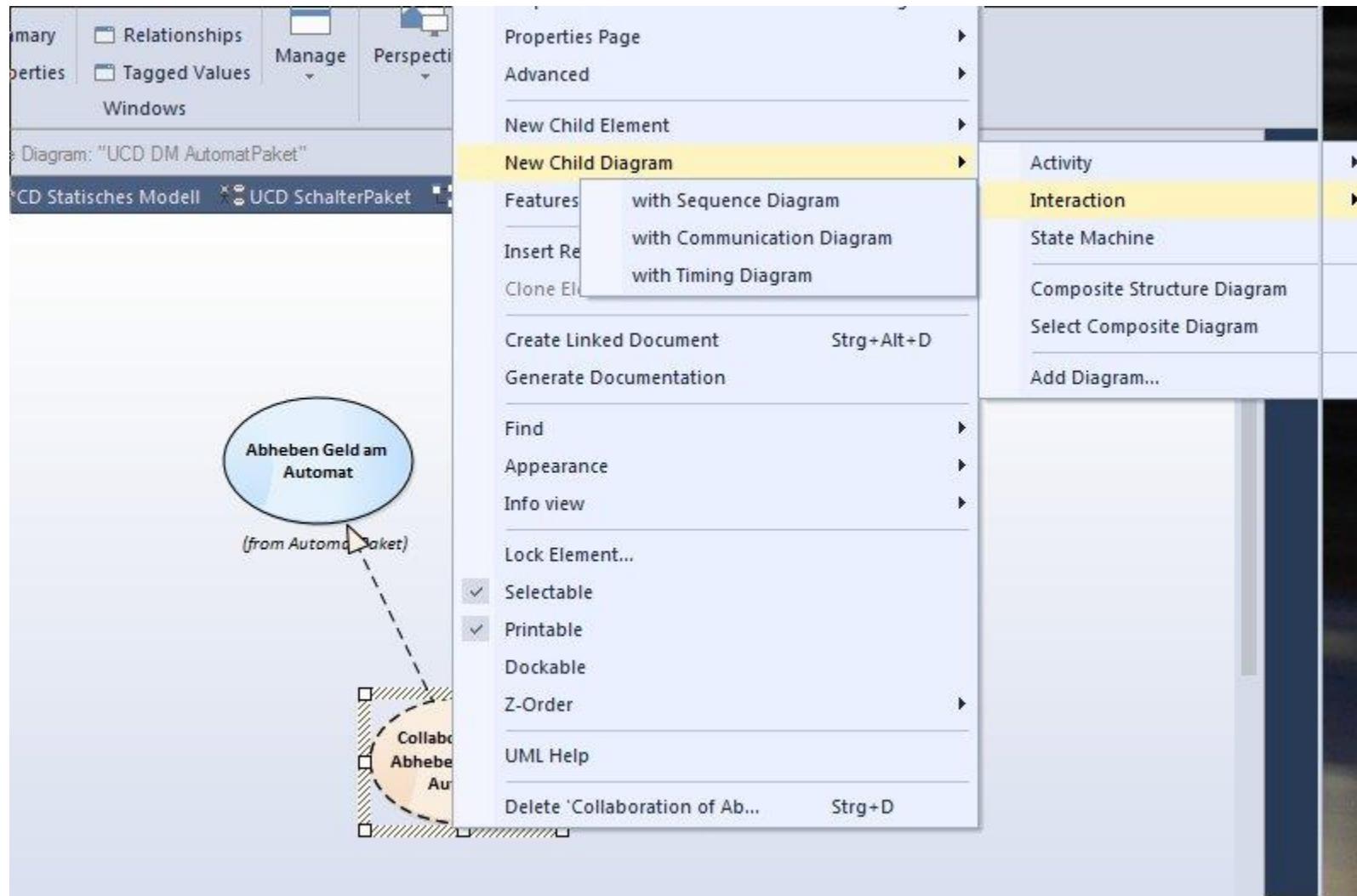
3.3

INTERAKTIONEN

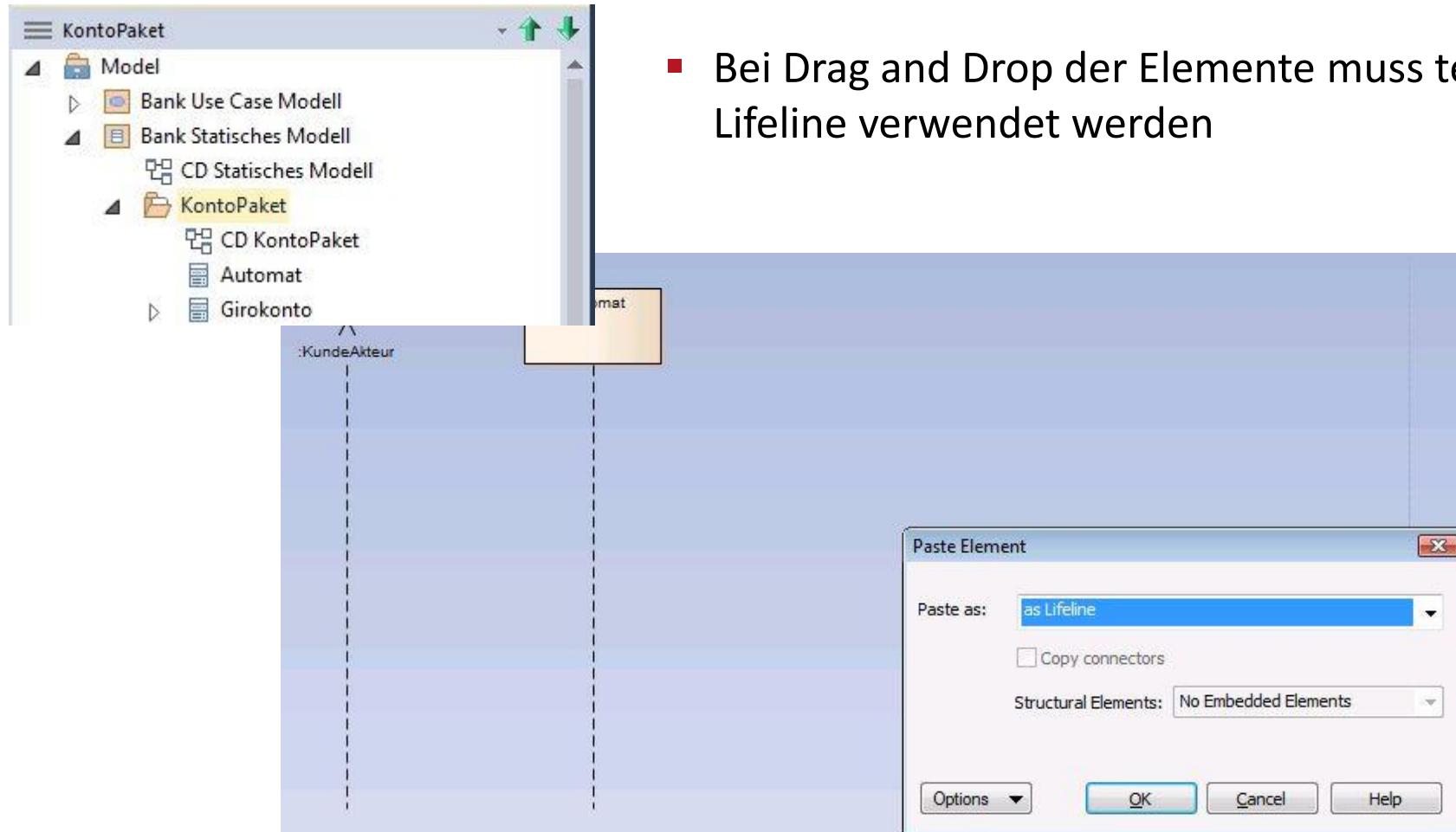


- Bei sauberer Planung können die Interaktionen über eine Collaboration realisiert werden
- Eine geeignete Struktur muss vorher angelegt werden

Die verschiedenen Diagramme zur Interaktions-Modellierung

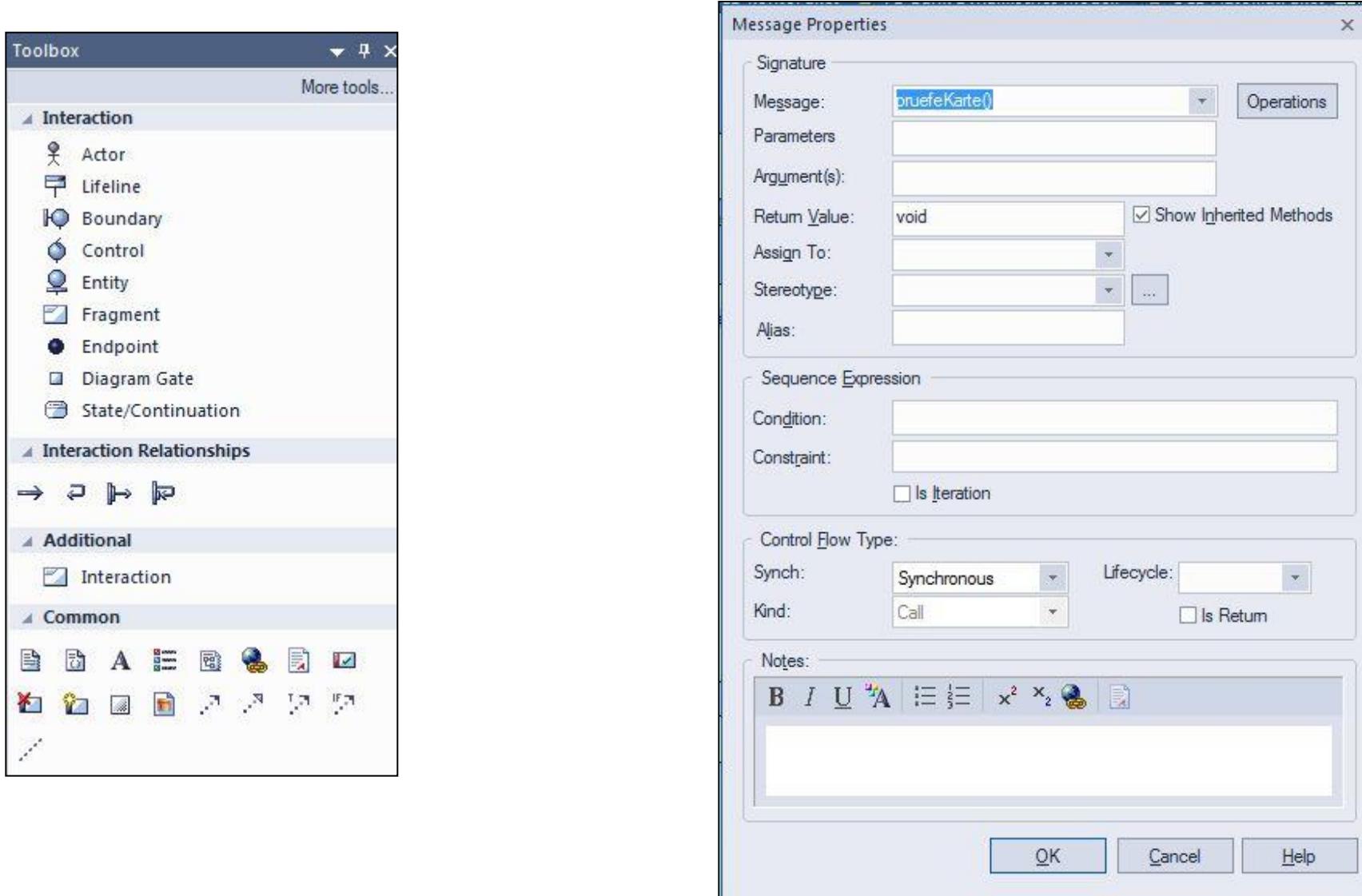


Sequenz-Diagramm: Struktur und erste Elemente



- Bei Drag and Drop der Elemente muss teilweise **as Lifeline** verwendet werden

Sequenz-Diagramm Toolbox und Nachrichten Einstellungen



Message Properties vs. Operations

Message Properties

Signature

Message: pruefeKarte()

Parameters:

Argument(s):

Return Value: void

Assign To:

Stereotype:

Alias:

Sequence Expression

Condition:

Constraint:

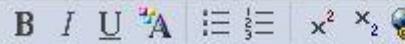
Is Iteration

Control Flow Type:

Synch: Synchronous

Kind: Call

Notes:

B I U A 

Karte : Features

Operations

Name	Parameters	Return Type	Scope	Stereotype	Alias
erzeugen		void	Public		
pruefeKarte		void	Public		

New Operation...

Method (pruefeKarte)

Concurrency: Sequential

Abstract: False

Static: False

Modifiers: Advanced

Parameters Notes Behavior Redefines Pre Post Tagged Values

Parameter	Type
New Parameter...	

Parameter

Default Value

Stereotype

Alias

Direction

Fixed Value

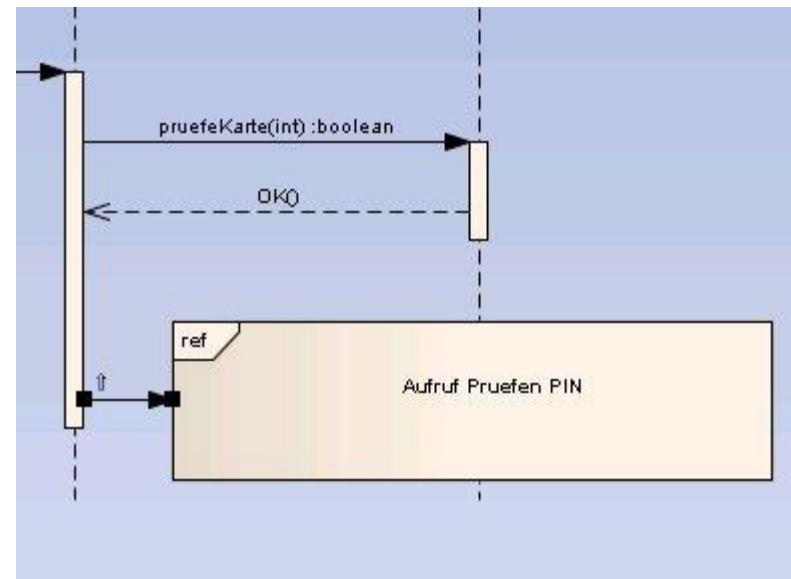
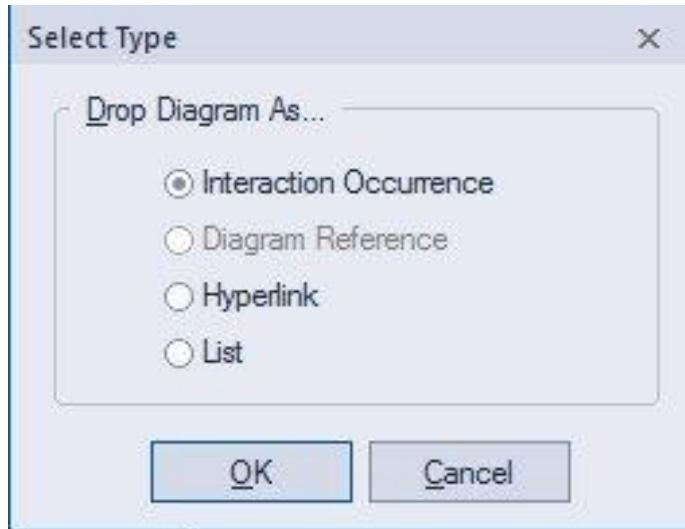
Multiplicity

Notes

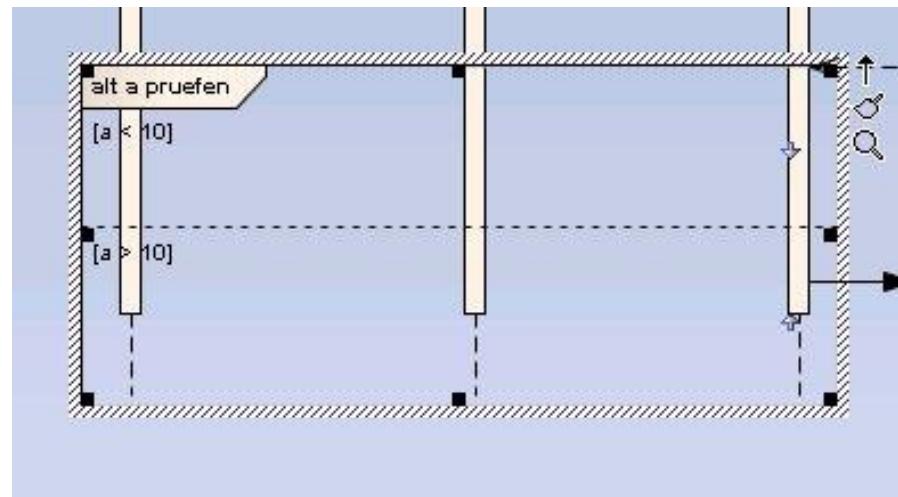
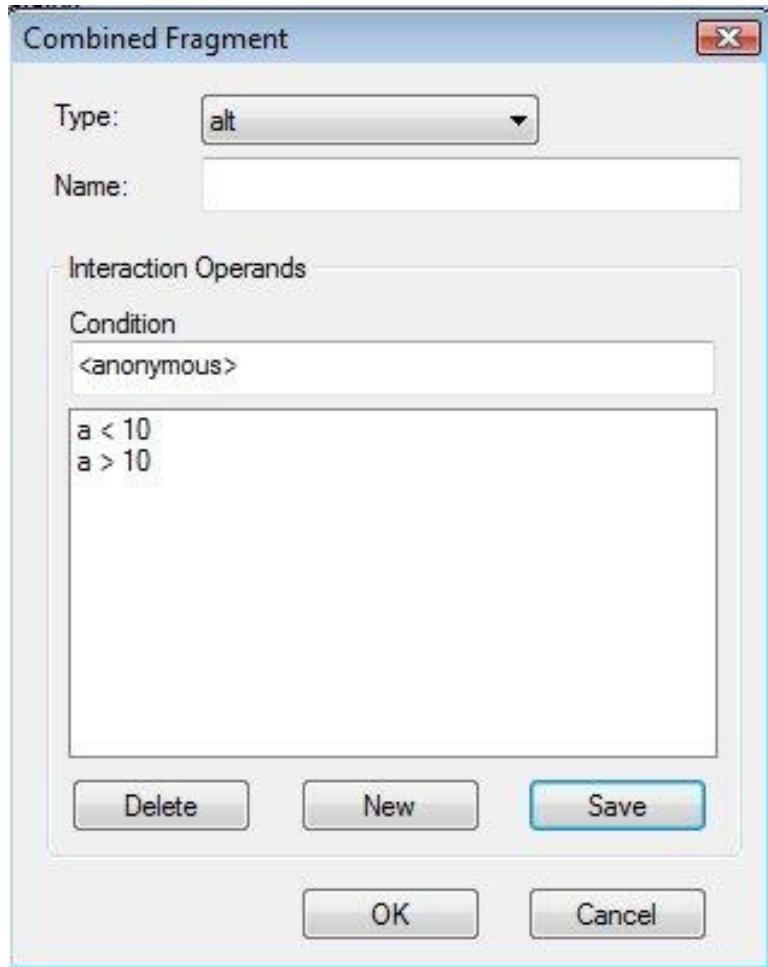
Close Hilfe

OK Cancel Help

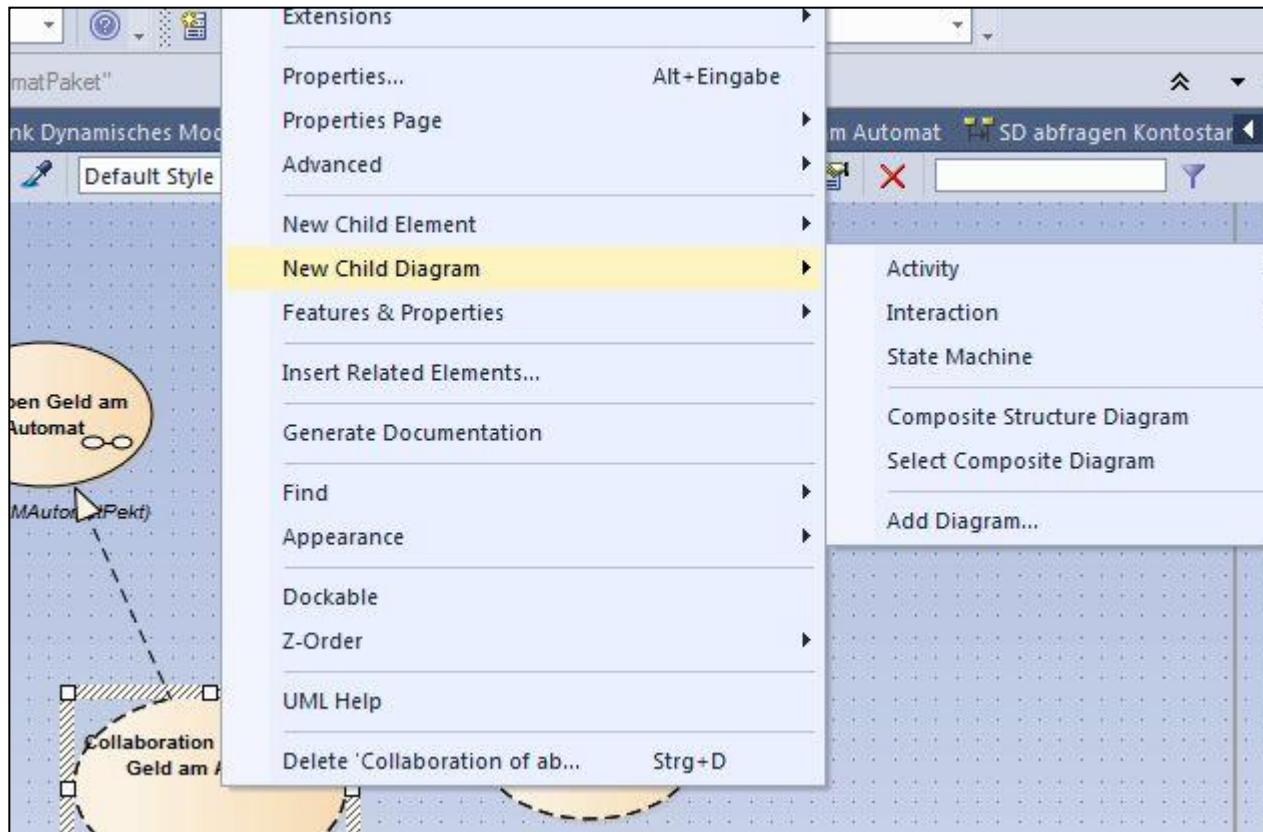
- Ein Sequenz-Diagramm kann ein anderes aufrufen. Diese ref kann durch ziehen des vorhandenen Diagramms erzeugt werden.



Kontrollstrukturen im Sequenz-Diagramm

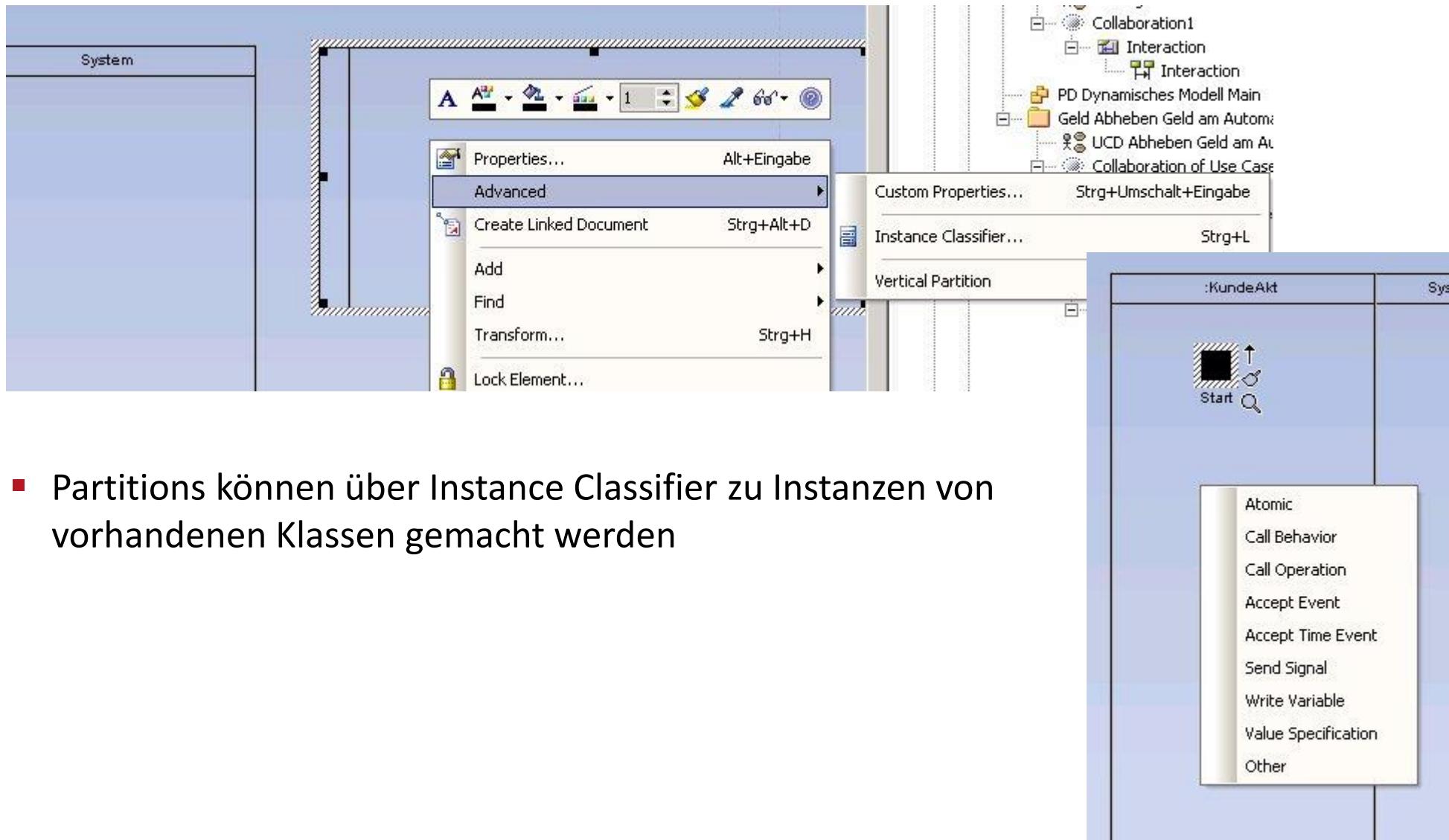


Link auf vorhandene Diagramme



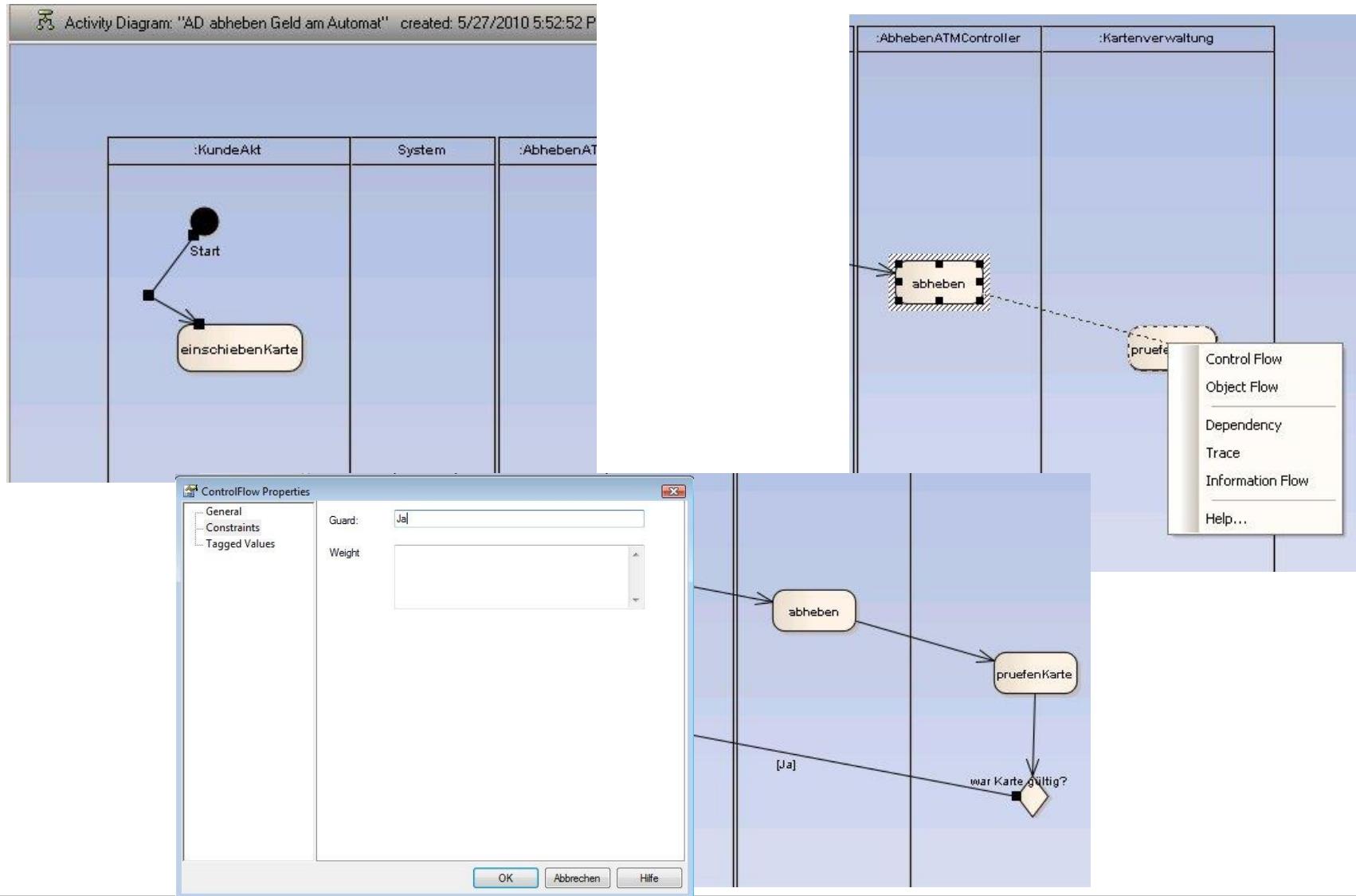
- Falls erforderlich lässt sich das System so konfigurieren, dass untergeordnete Diagramme über einen Doppelklick aufrufbar sind
- über *New Chlid Diagram* -> *Select Composite Diagram* auf dem Element (hier Collaboration)
- Auswahl des Diagramms

Aktivitäts-Diagramme zur Interaktions-Modellierung

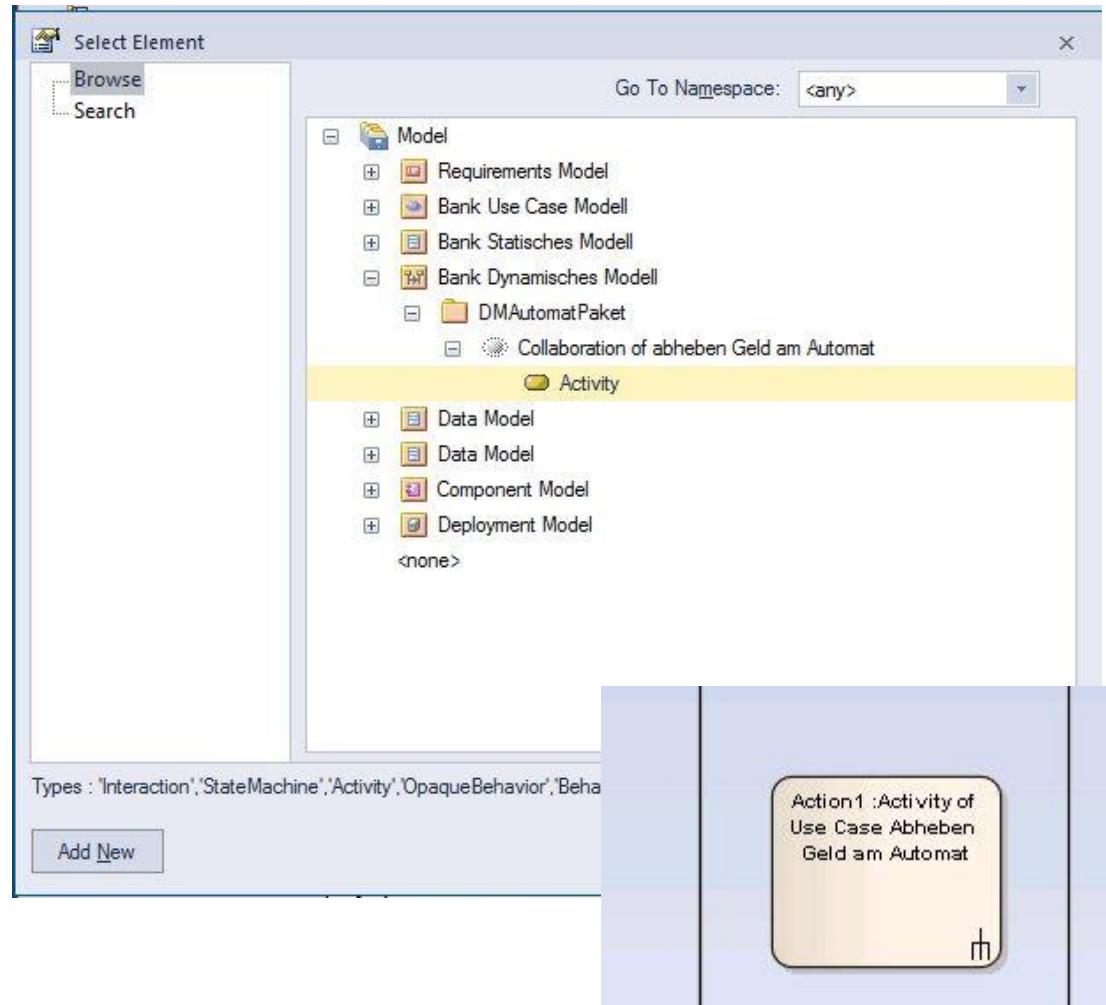


- Partitions können über Instance Classifier zu Instanzen von vorhandenen Klassen gemacht werden

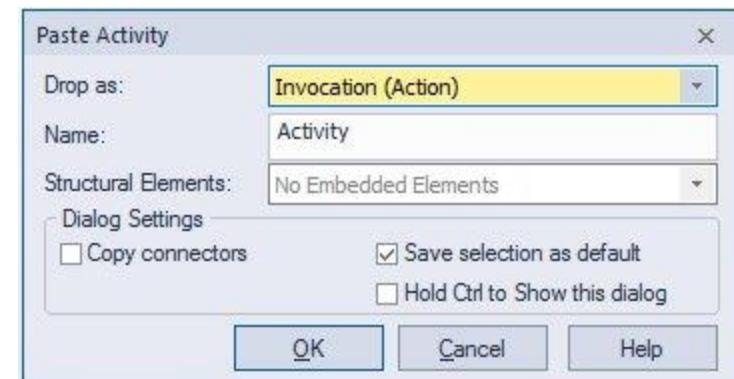
Details zum Aktivitäts-Diagramm



Call Behavior Action

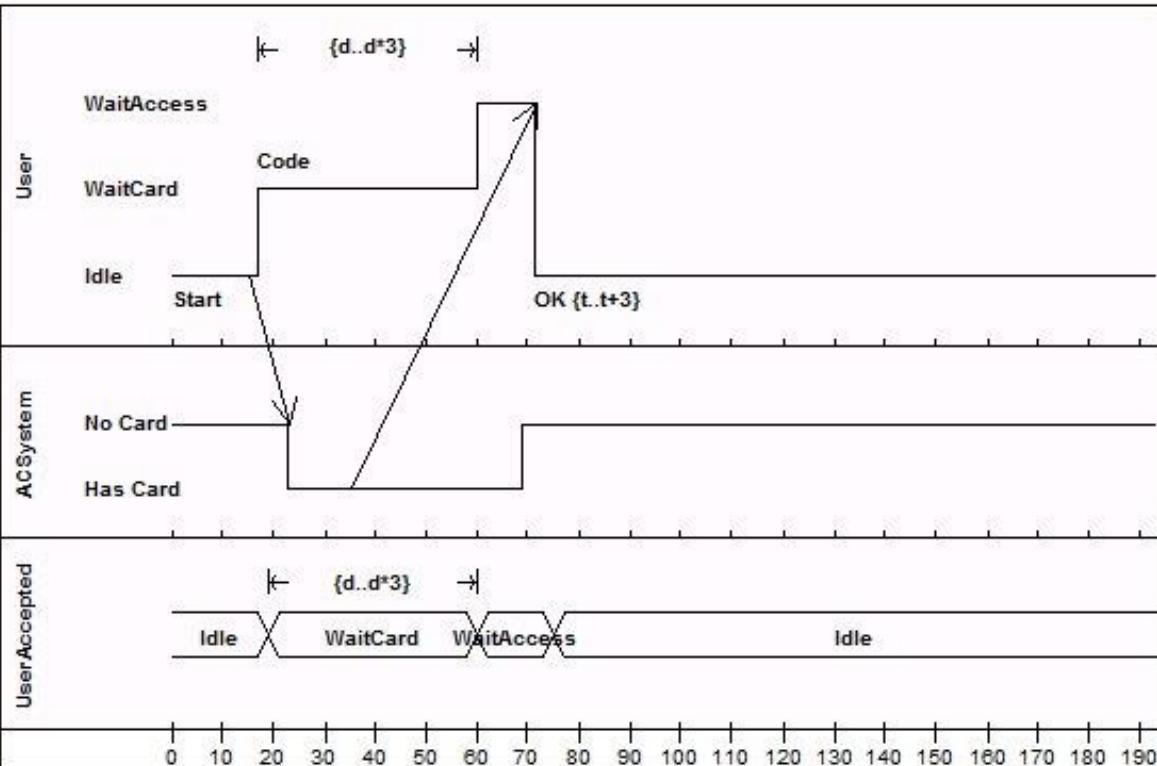


- Activity ins Diagramm ziehen
- Auswahl Drop as *Invocation (Action)*



Example Diagram

An example of a Timing diagram is shown below:



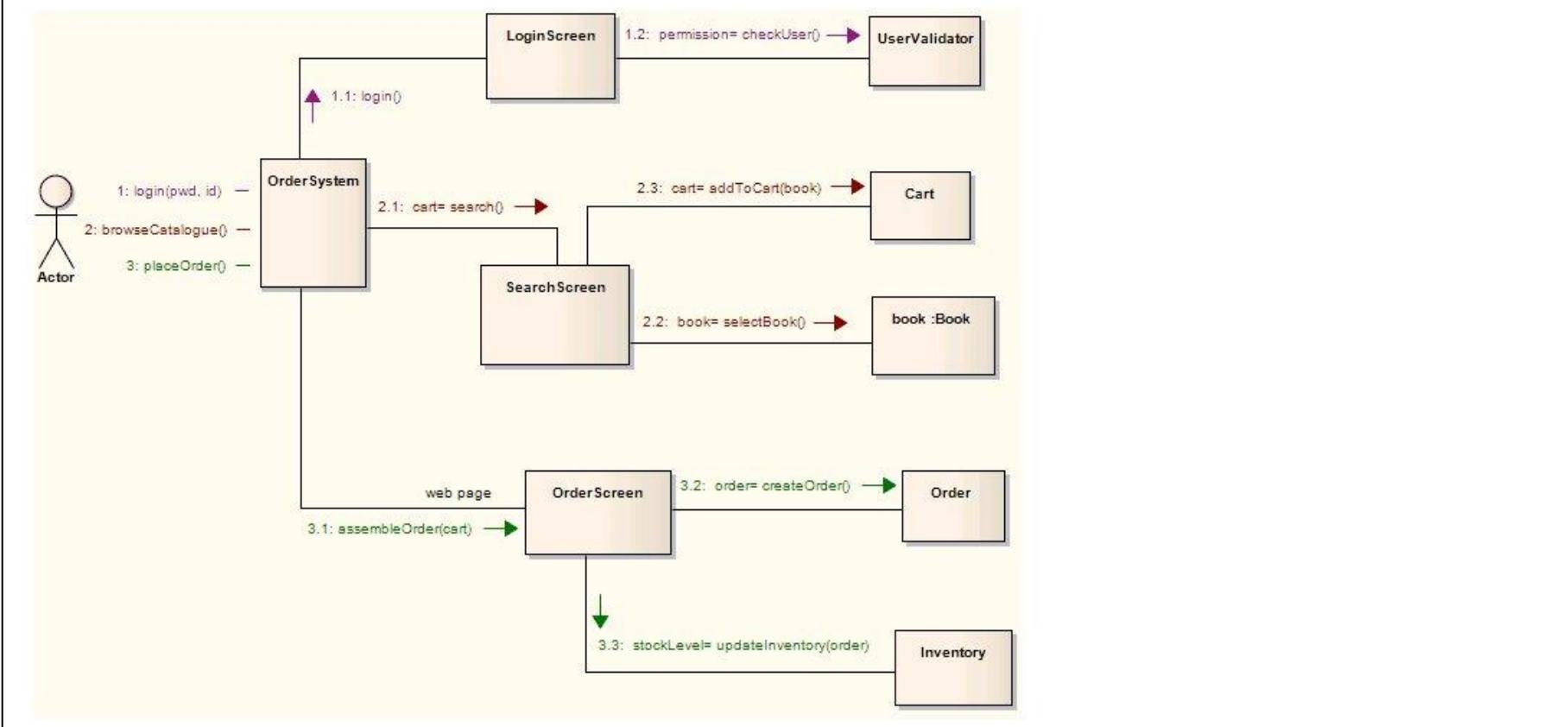
(See OMG UML Superstructure Specification, v2.1.1, p. 454, figures 14.30 and 14.31).

Weitere Diagramme: Kommunikations-Diagramm

- Kommunikations-Diagramme sind eine alternative zu den Sequenz-Diagrammen

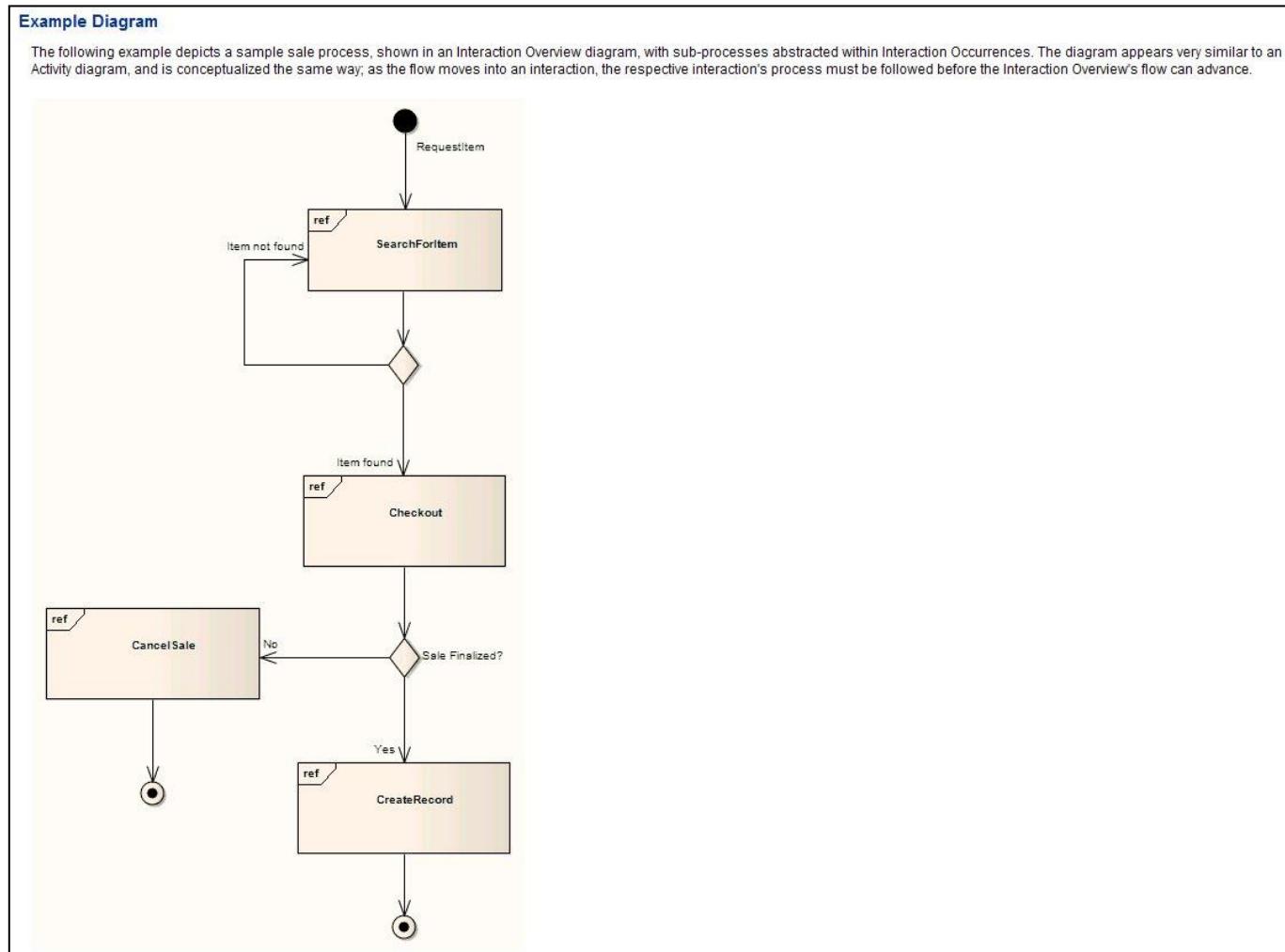
Example Diagram

The example below illustrates a Communication diagram among cooperating object instances. Note the use of message levels to capture related flows, and the different [colors](#) of the [messages](#).

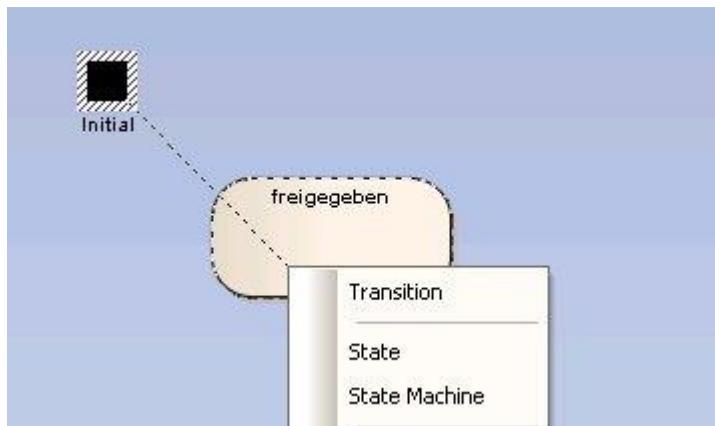
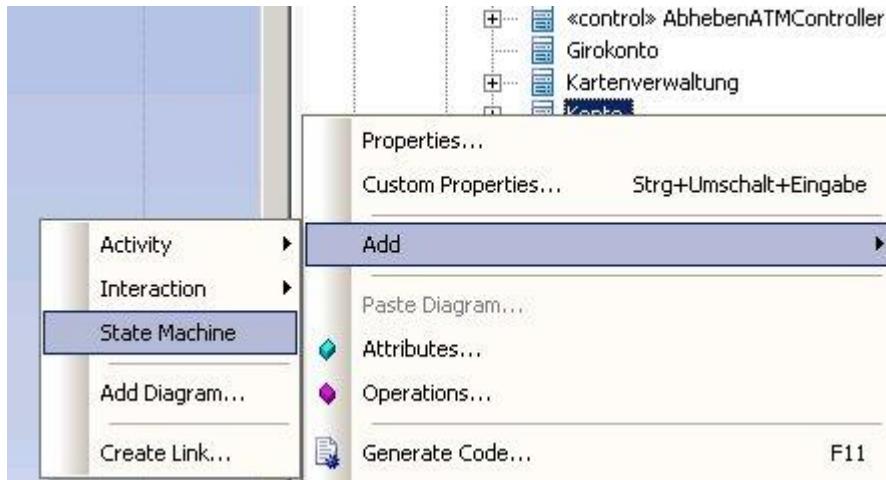


Weitere Diagramme: Kommunikations-Übersichts-Diagramm

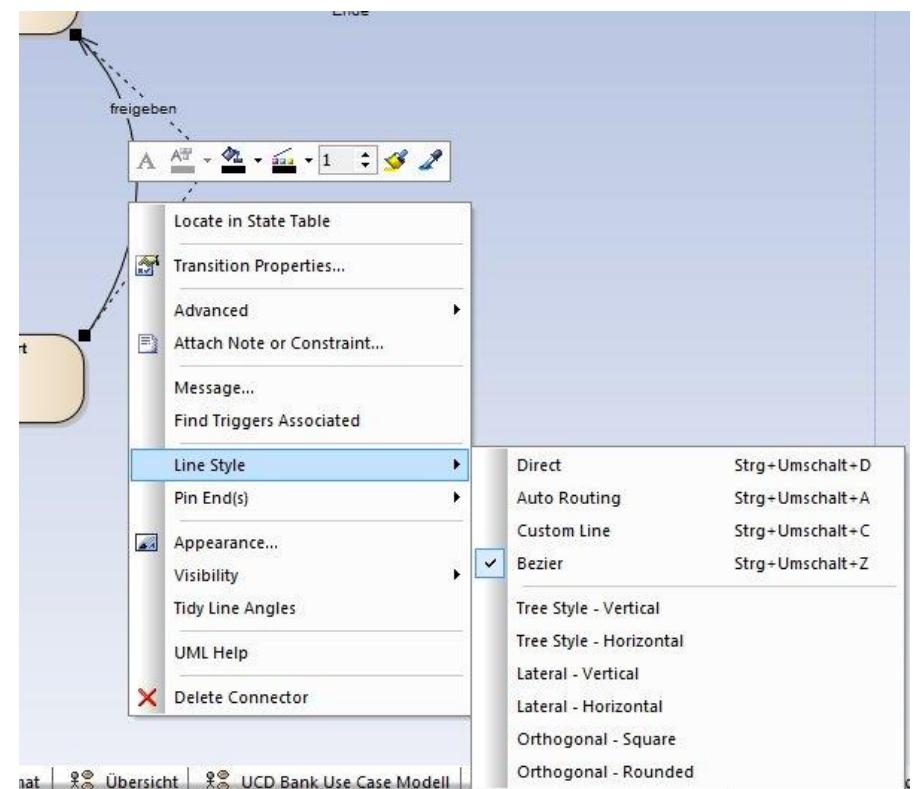
- Die Kommunikations-Übersichts-Diagramme können bei komplexen Zusammenhängen als Übersicht und Referenz auf verschiedene Unter-Diagrammtypen verwendet werden



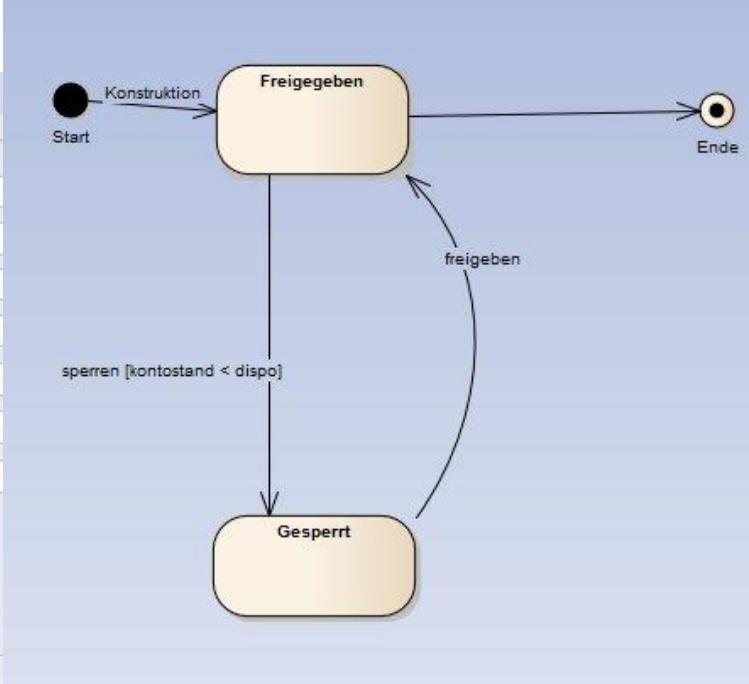
Zustands-Diagramm (1)



- Zustand-Diagramme werden pro Klasse erstellt



Zustands-Diagramm (2)



The diagram illustrates a UML Statechart. It starts with a black initial state (Start) leading to a rounded rectangle labeled "Freigegeben". From "Freigegeben", an arrow labeled "Konstruktion" leads to a final state (Ende). A curved arrow labeled "freigeben" loops back from "Freigegeben" to itself. Below the diagram is a screenshot of a software interface showing the "Freigegeben" state properties.

Properties

- General
- Rules
- Requirements
- Constraints
- Scenarios
- Related
 - Files
 - Links

Freigegeben

Properties

- Stereotype:
- Status: Proposed
- Alias:
- Keywords:
- Author: Administrator
- Complexity: Easy
- Language: Java
- Version: 1.0
- Phase: 1.0
- Package: KontoPaket
- Created: 01.02.2012 12:03:41
- Modified: 01.02.2012 12:03:48

Main Advanced Tags

OK Abbrechen Übernehmen Hilfe

4

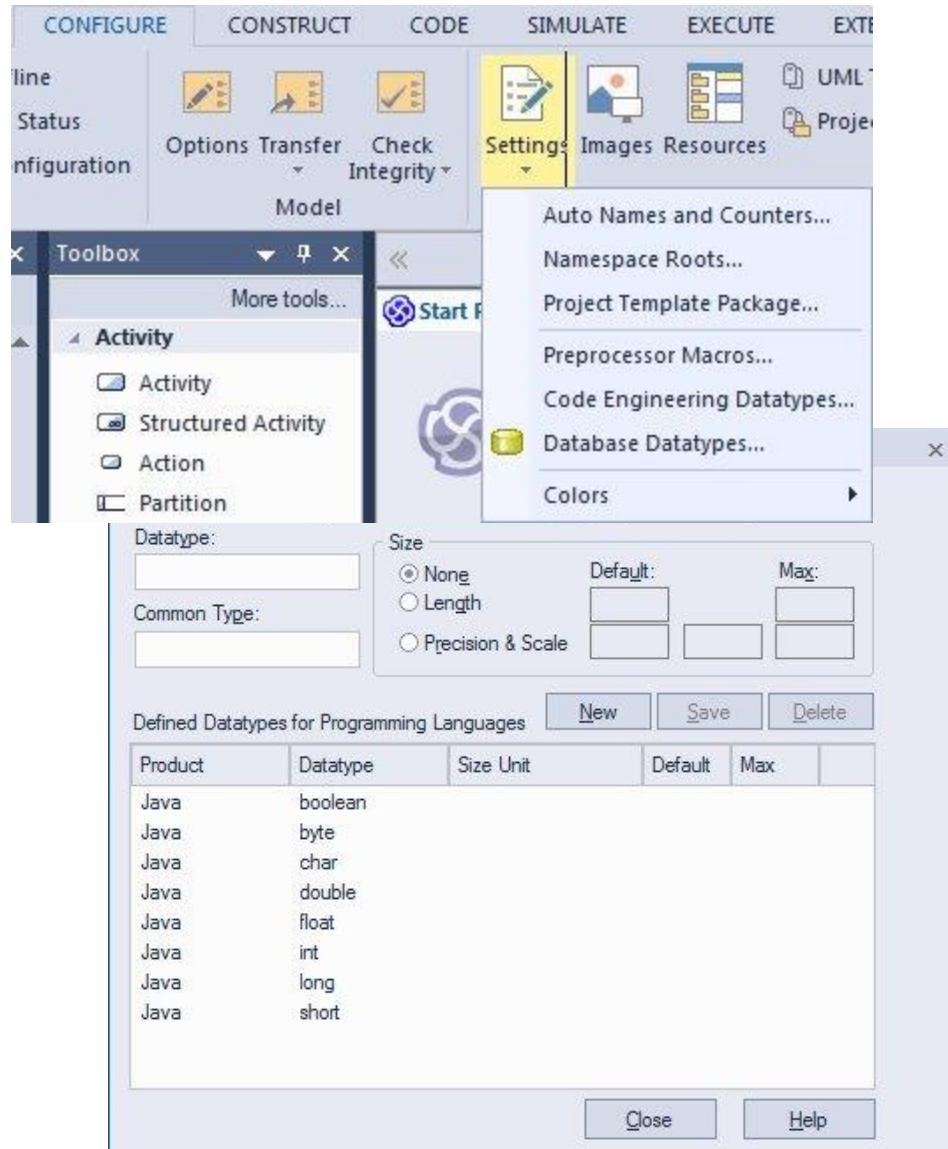
TECHNISCHES MODELL

4.1

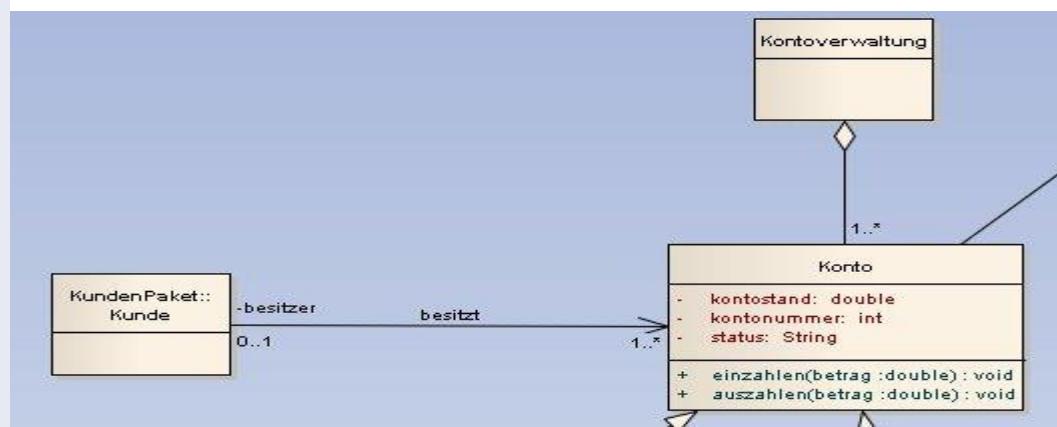
SPRACH-SPEZIFISCHE DETAILS

- Spätesten jetzt muss das Modell weiter verfeinert werden
 - Anpassung an die Zielsprache
 - Klassen müssen vollständig beschrieben sein
 - Alle Beziehungen müssen vollständig dargestellt sein
 - Das Klassen-Diagramm muss spätestens jetzt in Pakete unterteilt werden
- Die meisten Verfeinerungen auf Klassenebene im EA wurden bereits angesprochen
- Ergänzungen in diesem Kapitel

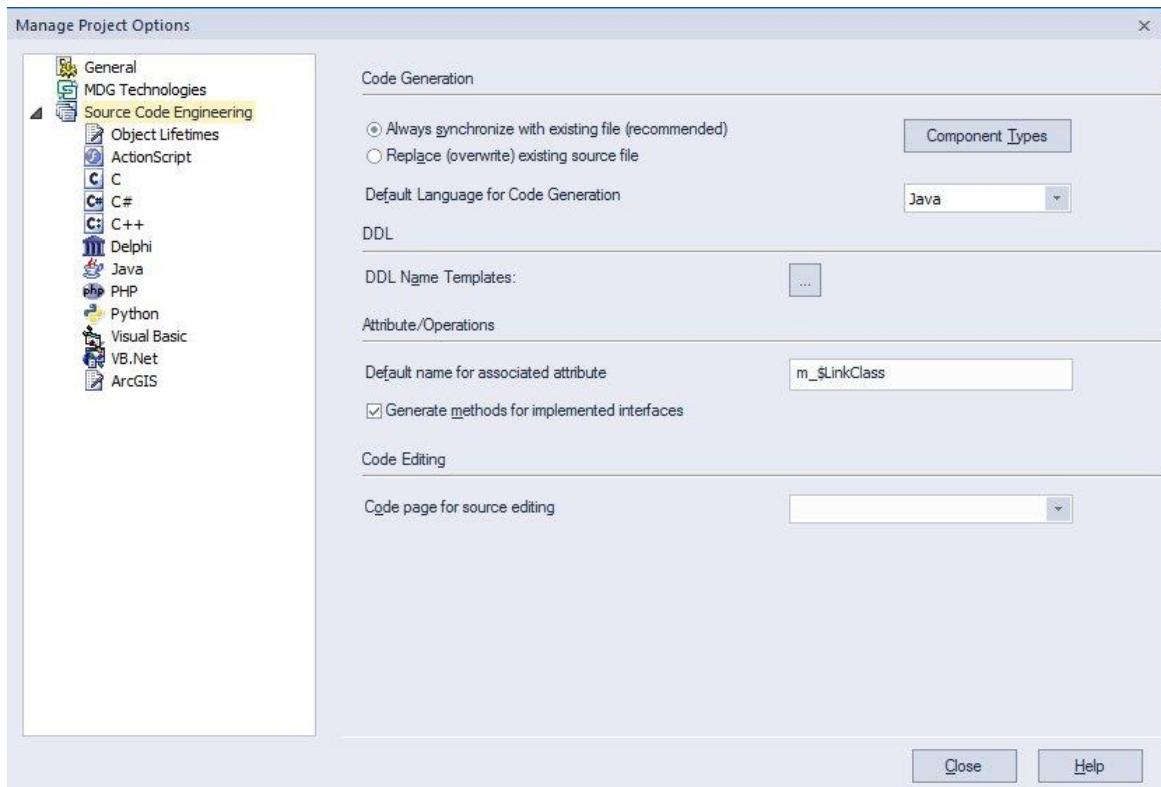
Verfeinerung des vorhandenen Modells



- Das vorhandene Modell muss hinsichtlich der Realisierung verfeinert werden:
 - Attribute und Methoden vollständig festlegen
 - Traversieren der Beziehungen
 - Rollen
 - Datentypen der Zielsprache

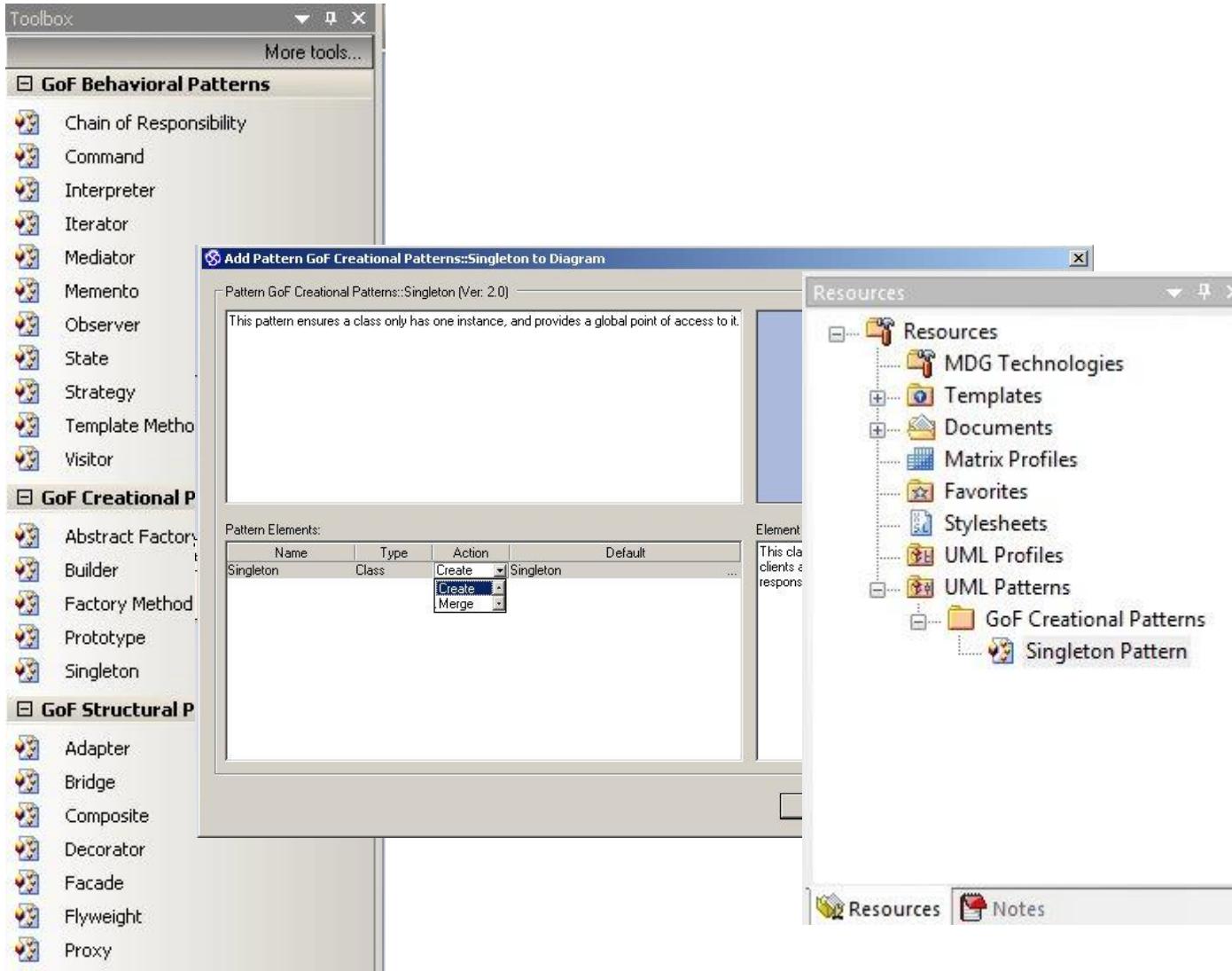


Festlegen über Register Configure -> Options



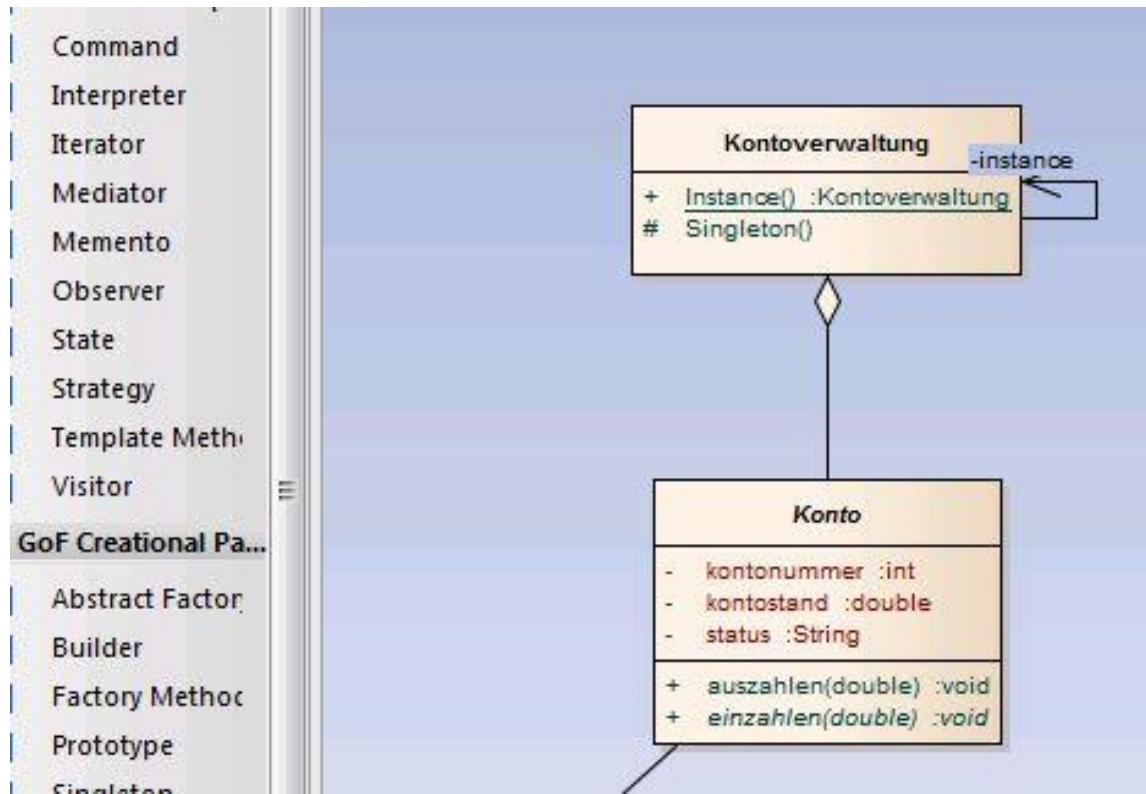
- Das vorhandene Modell muss hinsichtlich der Realisierung verfeinert werden:
 - Datentypen der Zielsprache

Defined Datatypes for Programming Languages				
Product	Datatype	Size Unit	Default	Max
Java	boolean			
Java	byte			
Java	char			
Java	double			
Java	float			
Java	int			
Java	long			
Java	short			
Java	String			

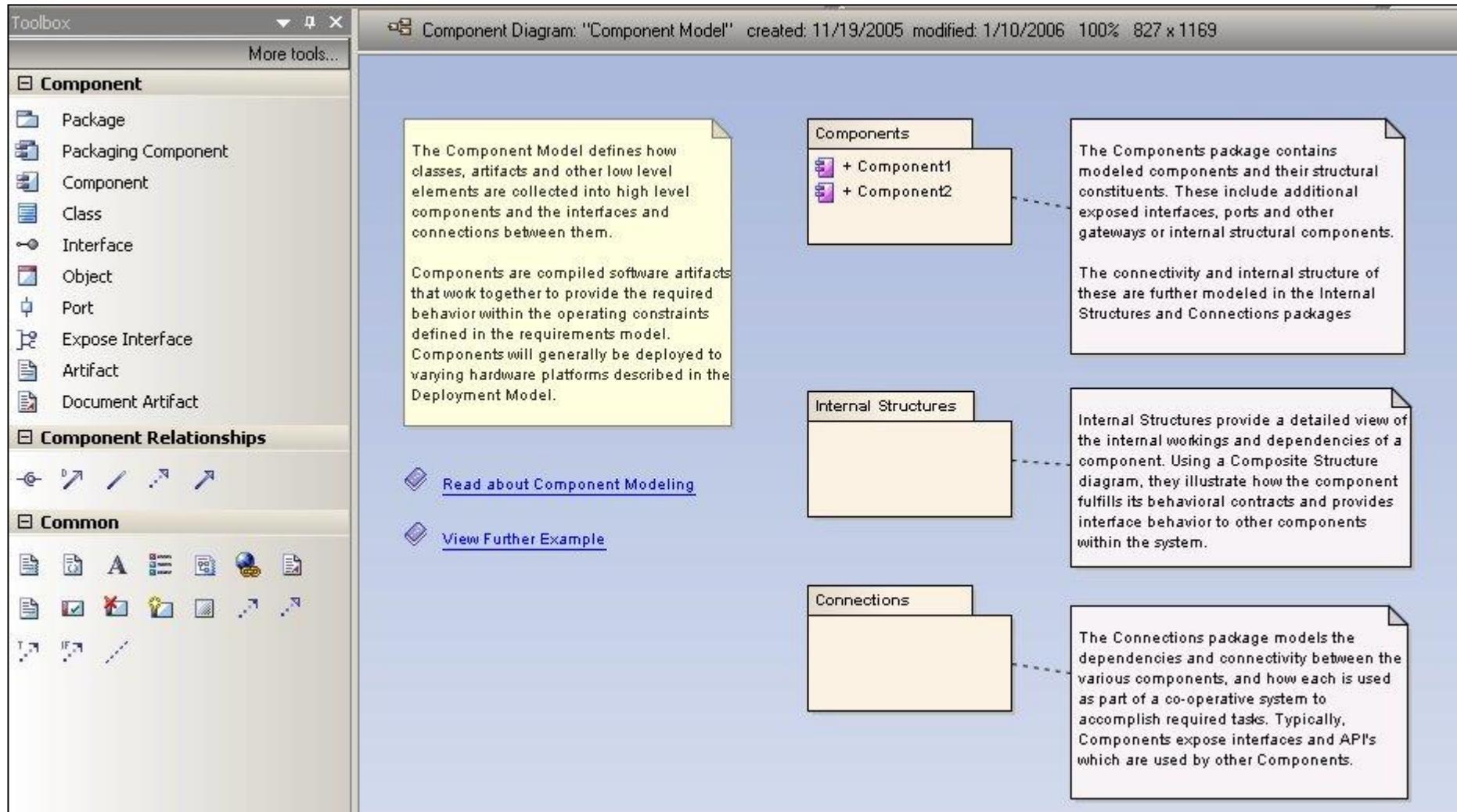


- Alle 23 GoF Pattern können im Modell verwendet werden
- Die Auswahl erfolgt über die Toolbox: GoF Behavioral Pattern
 - Create
 - Merge
- Nutzung allerdings nicht ideal

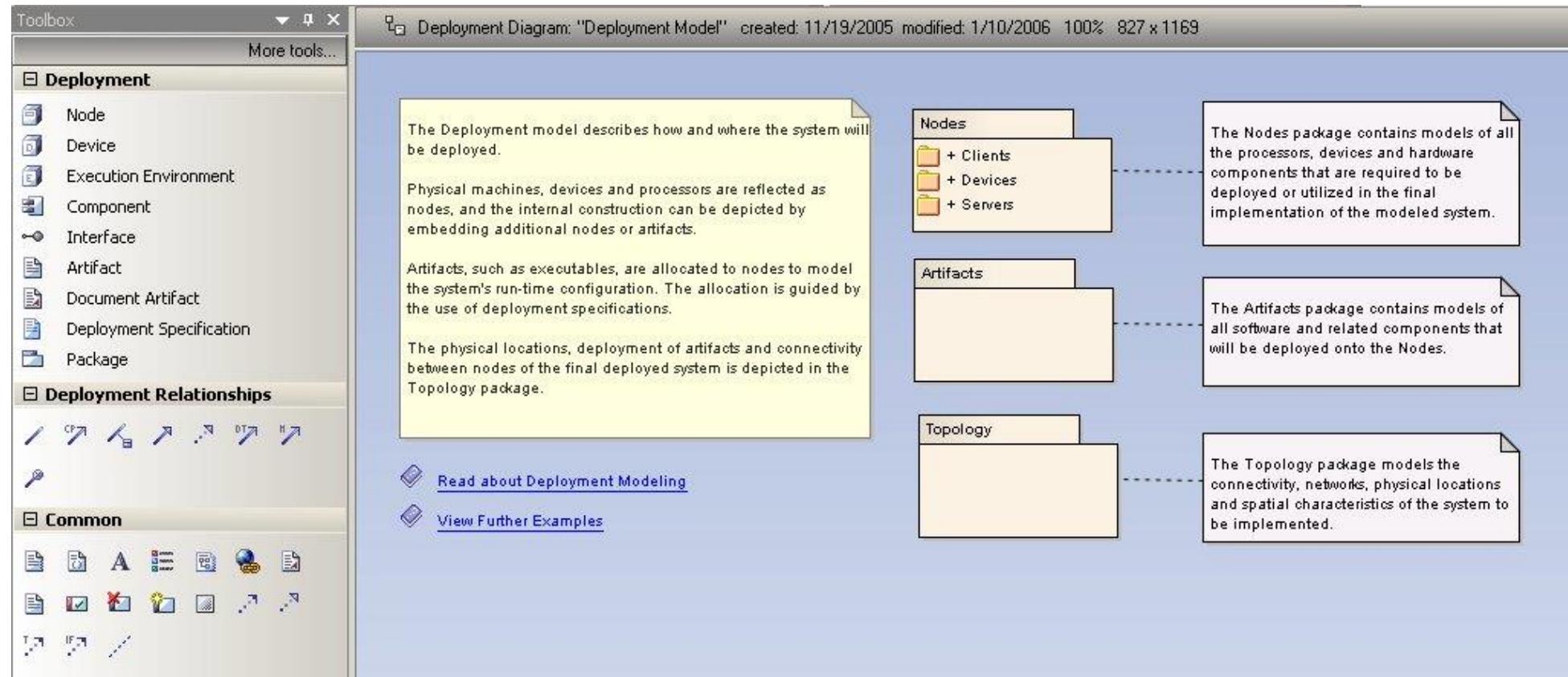
- Bei der Verwendung von Merge lassen sich die Pattern auf vorhandene Klassen anwenden



Weitere Diagramme: Komponenten-Diagramm

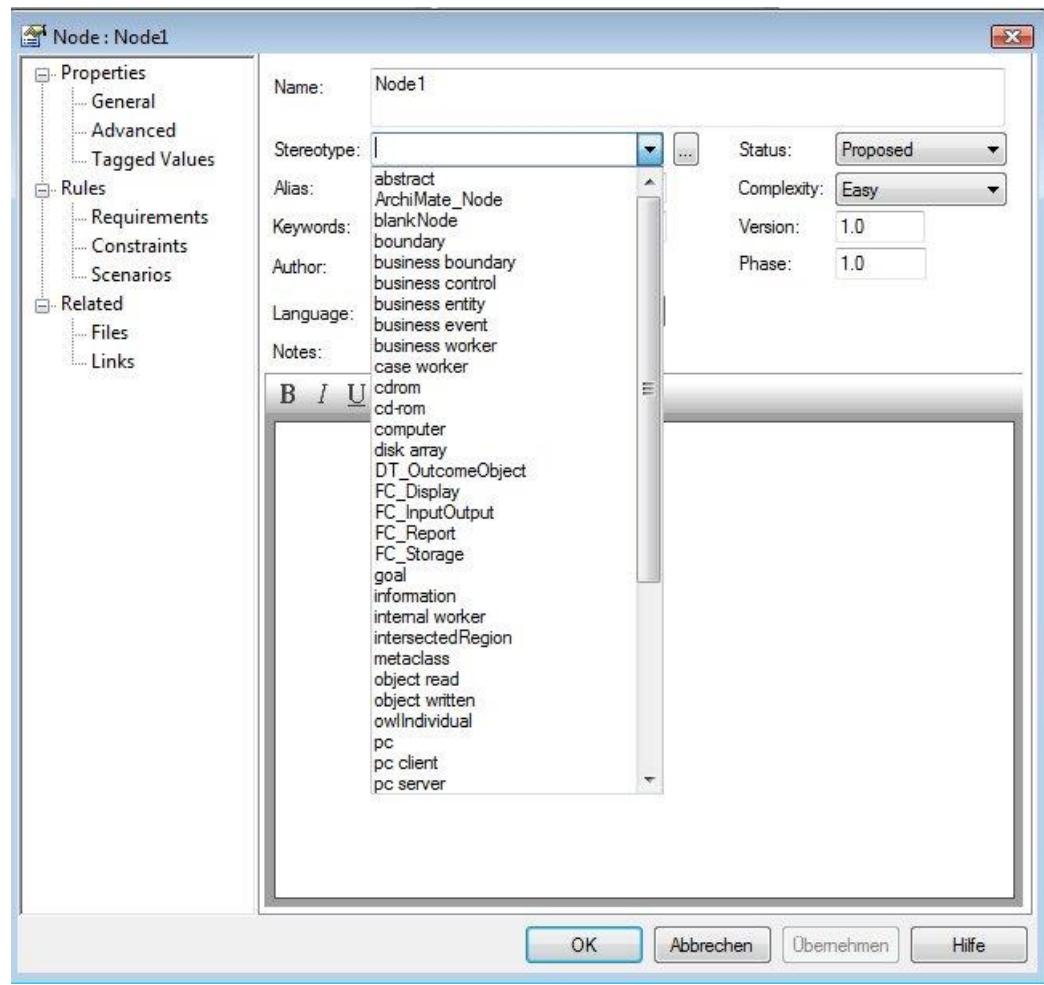


Weitere Diagramme: Deployment-Diagramm





- Seit UML 2 sind stereotyped Nodes möglich



Kombinationen mit Komponenten im Deployment-Diagramm und neue Elemente in der UML

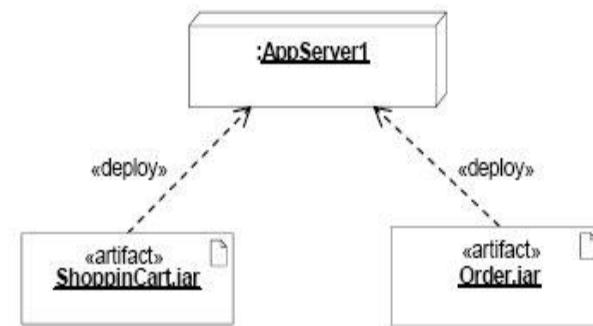
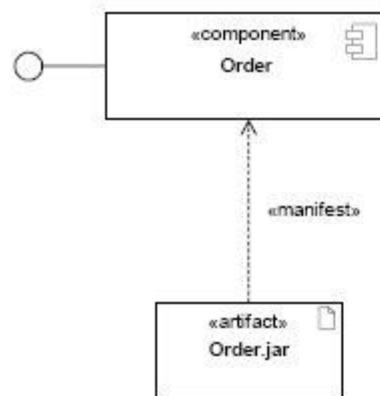
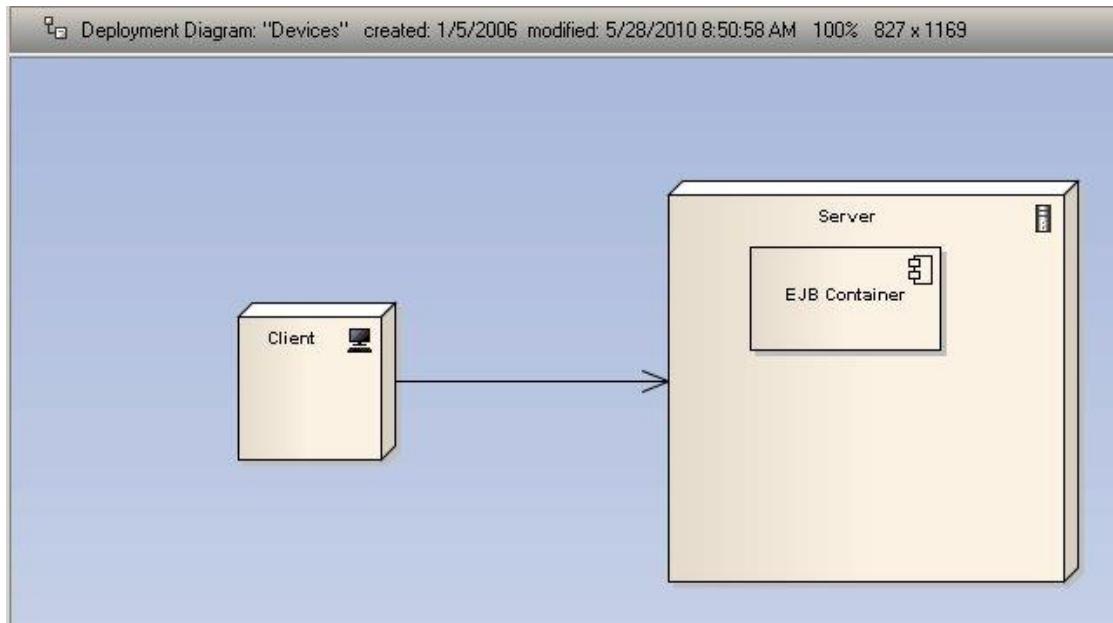
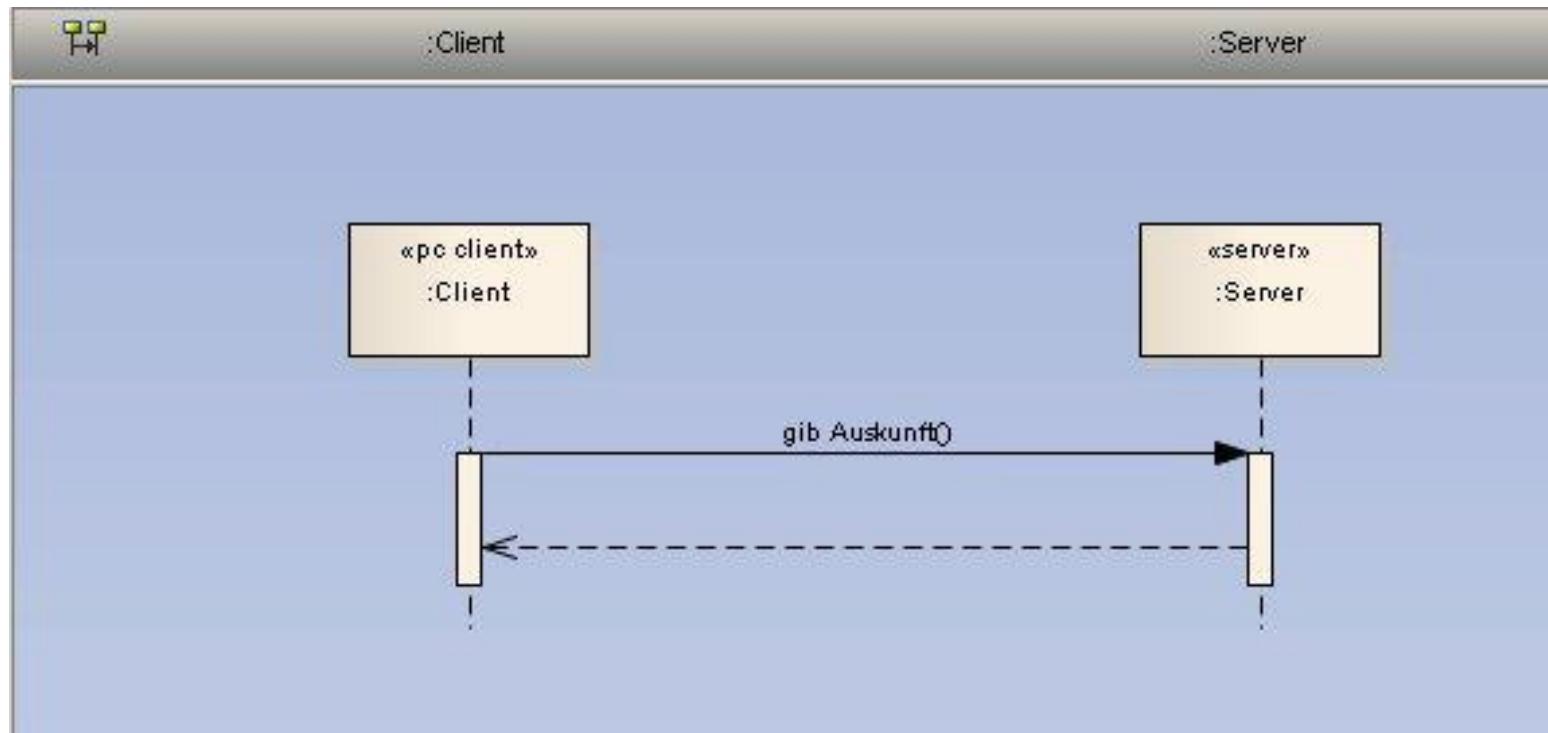


Figure 10.9 - Alternative deployment representation of using a dependency called «deploy»

Figure 10.7 - A visual representation of the manifestation relationship between artifacts and components

Sequenz-Diagramme für Deployment - Diagramme

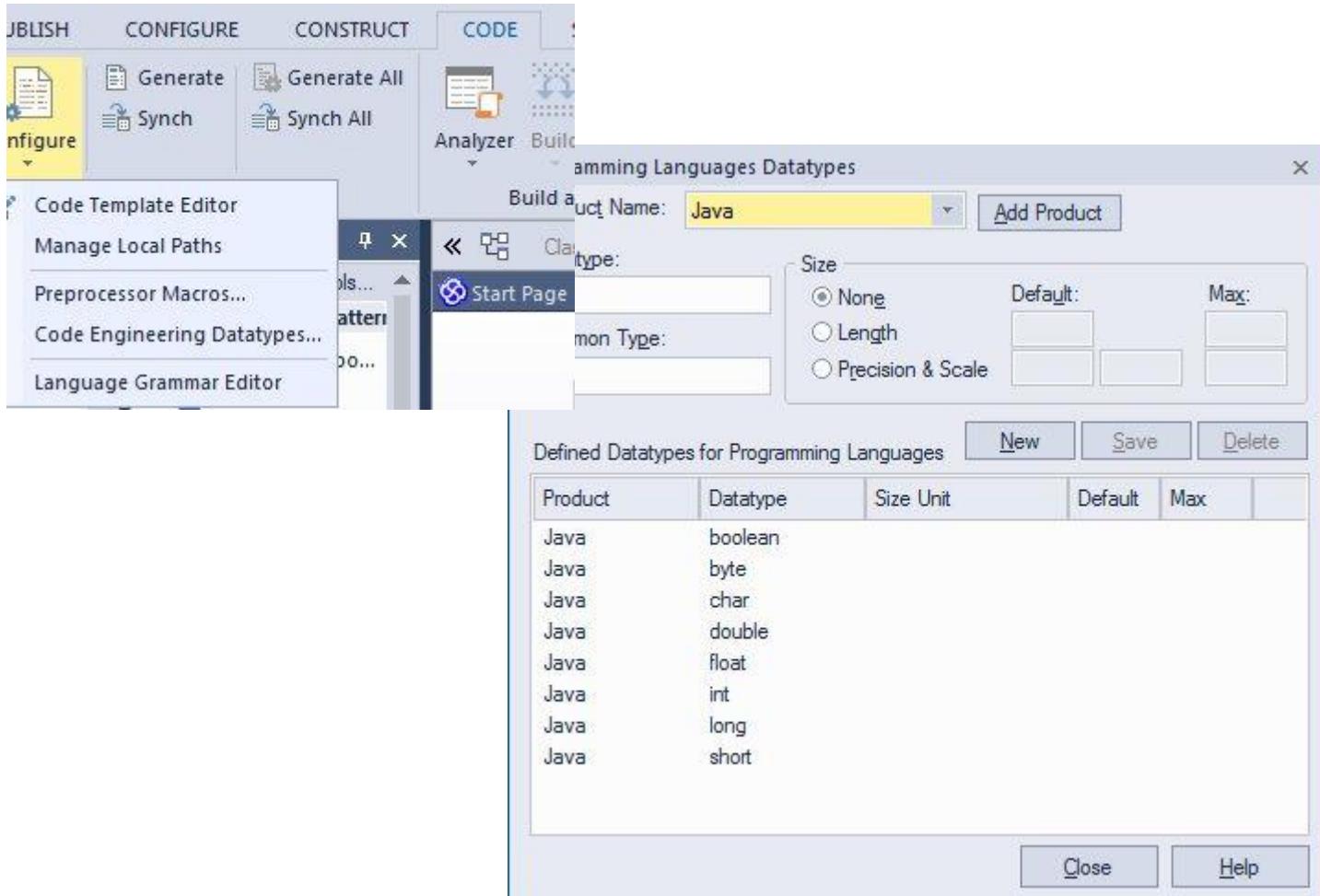
- Die Beschreibung eines Roundtrips z.B. für die Architektur ist mit den Sequenz-Diagrammen auf Basis der Deployment- und Komponenten-Diagramme möglich



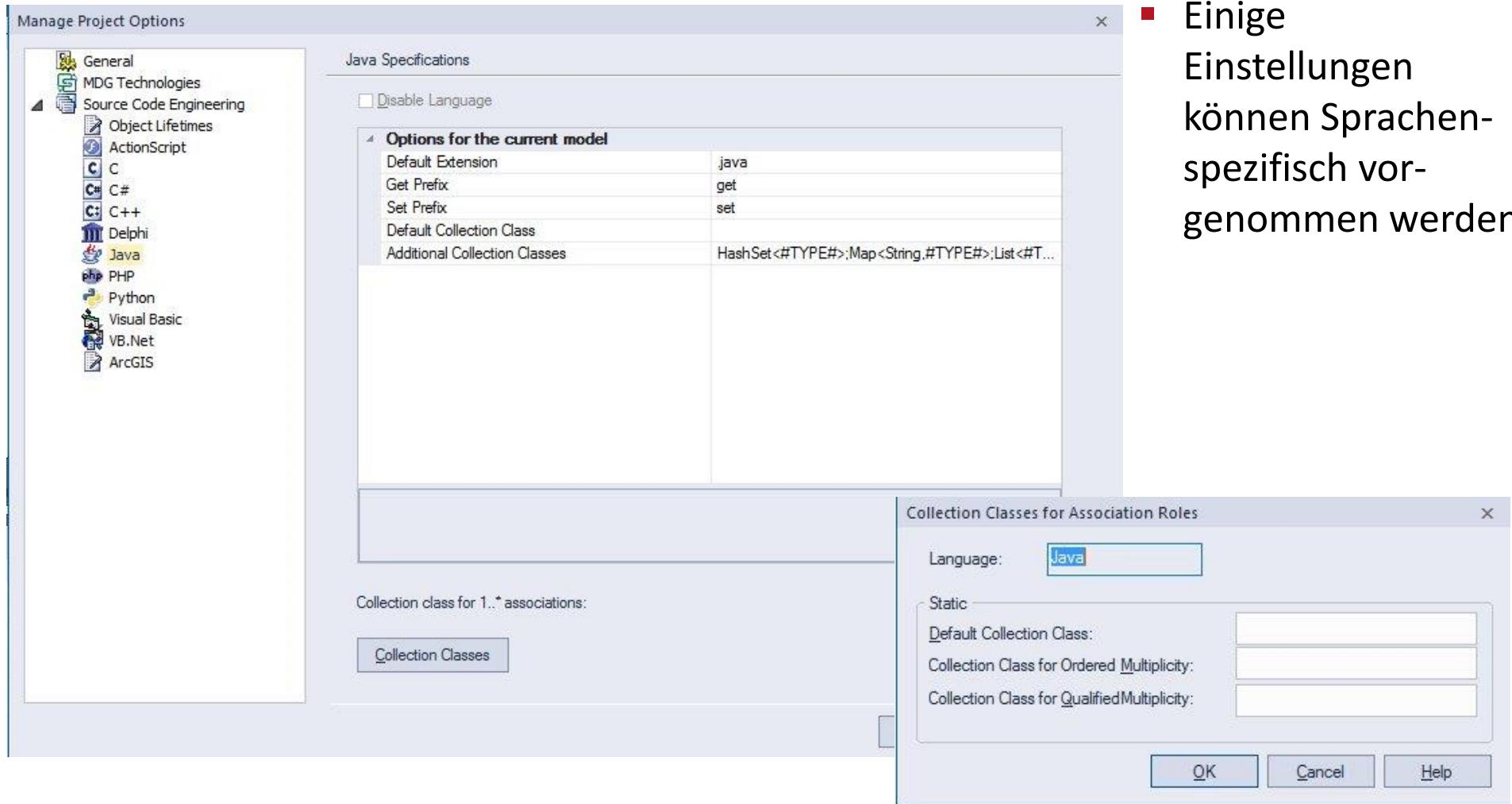
4.2

ENTERPRISE ARCHITECT - ERWEITERUNGEN

- Code-Generierungs-optionen sind für die jeweiligen Sprachen einstellbar



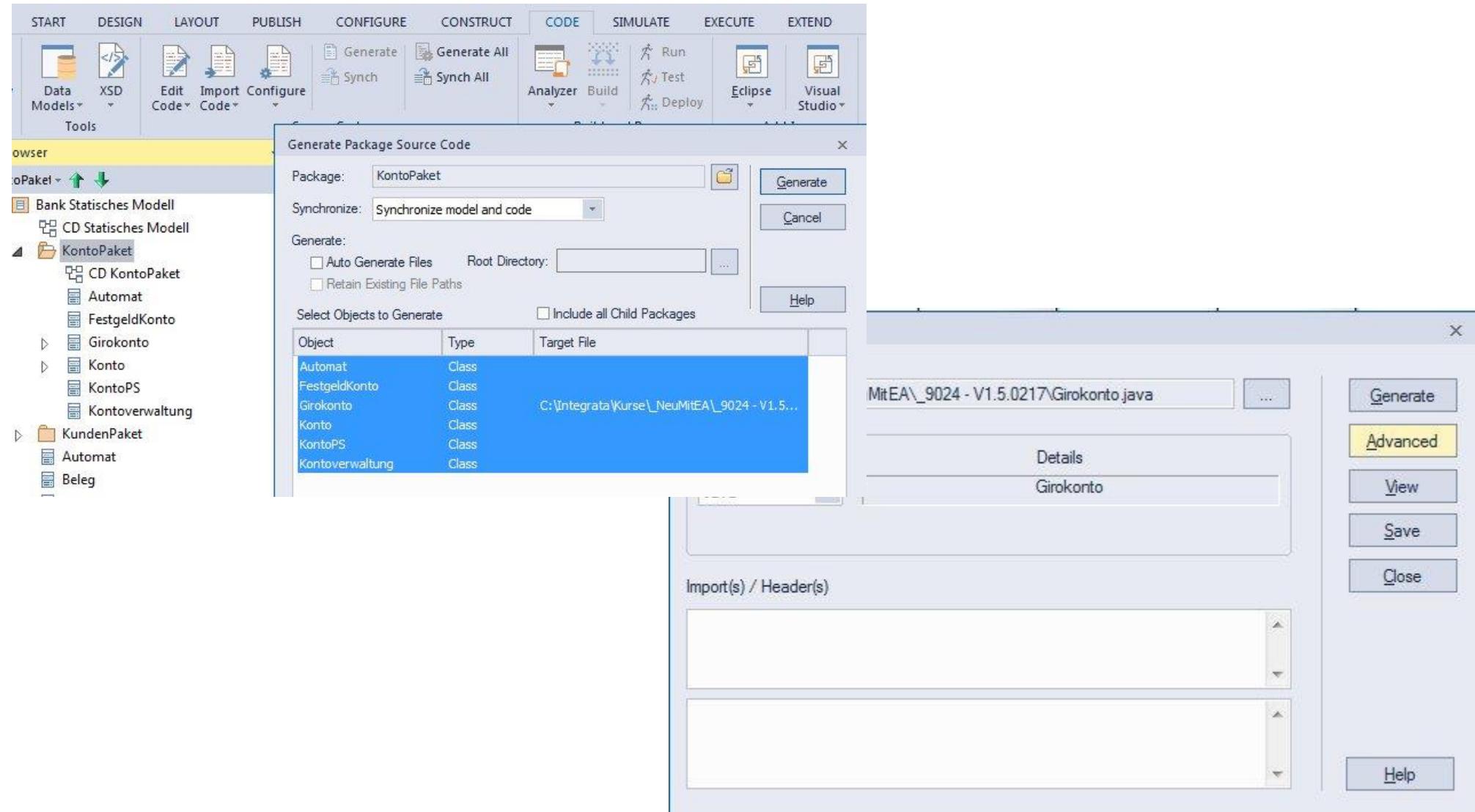
Sprachspezifische Einstellungen am Beispiel Java



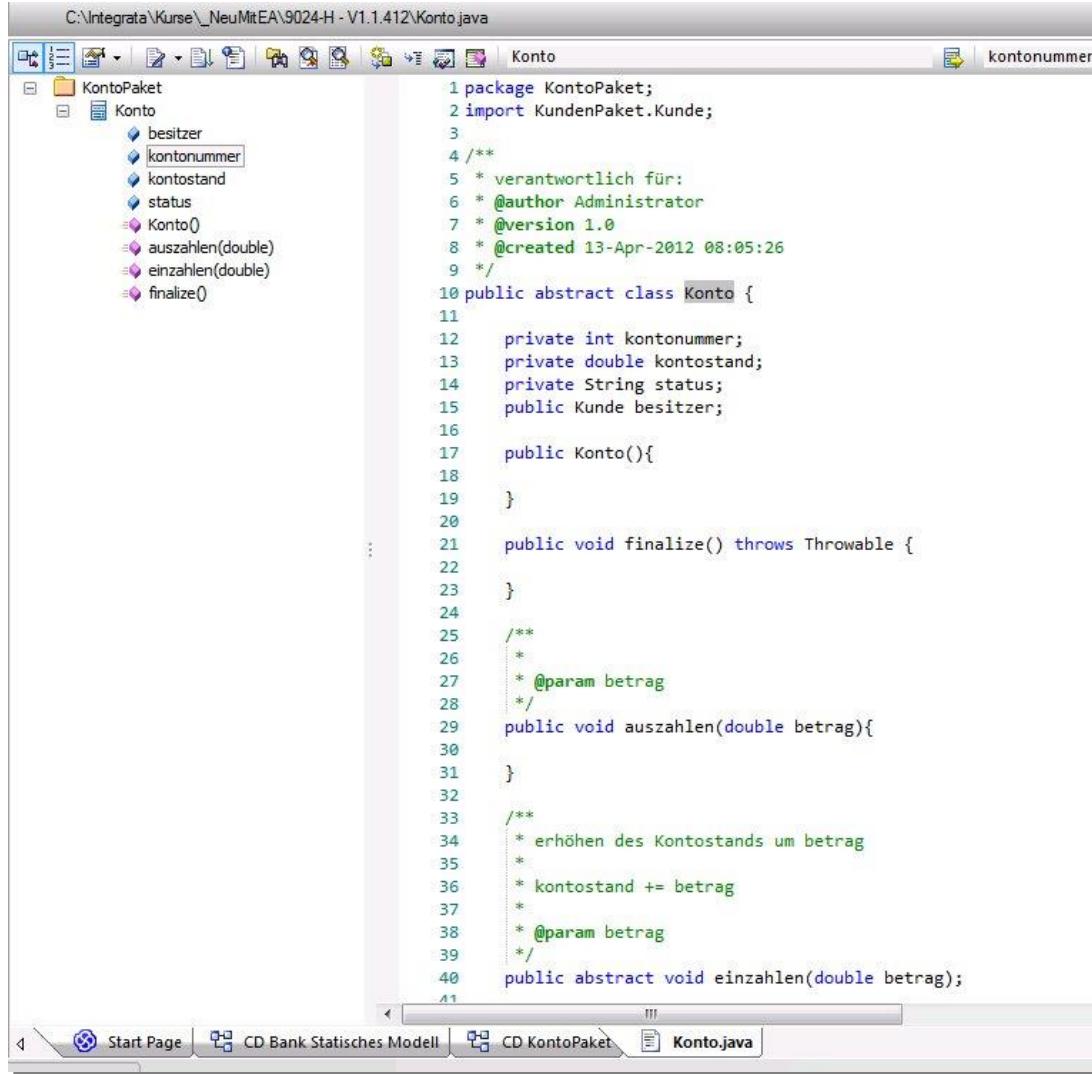
The screenshot shows the 'Manage Project Options' dialog in Enterprise Architect. The left sidebar lists various technologies, with 'Java' selected. The main area is titled 'Java Specifications' and contains a table for 'Options for the current model'. The table includes rows for 'Default Extension' (set to 'java'), 'Get Prefix' (set to 'get'), 'Set Prefix' (set to 'set'), 'Default Collection Class' (set to 'HashSet<#TYPE#>;Map<String,#TYPE#>;List<#T...>'), and 'Additional Collection Classes'. Below this is a 'Collection Classes for Association Roles' dialog, also set to 'Java'. It has sections for 'Language' (selected), 'Static', 'Default Collection Class', 'Collection Class for Ordered Multiplicity', and 'Collection Class for Qualified Multiplicity'. Buttons at the bottom include 'OK', 'Cancel', and 'Help'.

- Einige Einstellungen können sprachspezifisch vorgenommen werden

Die Kontext-Menüs für die Code-Generierung



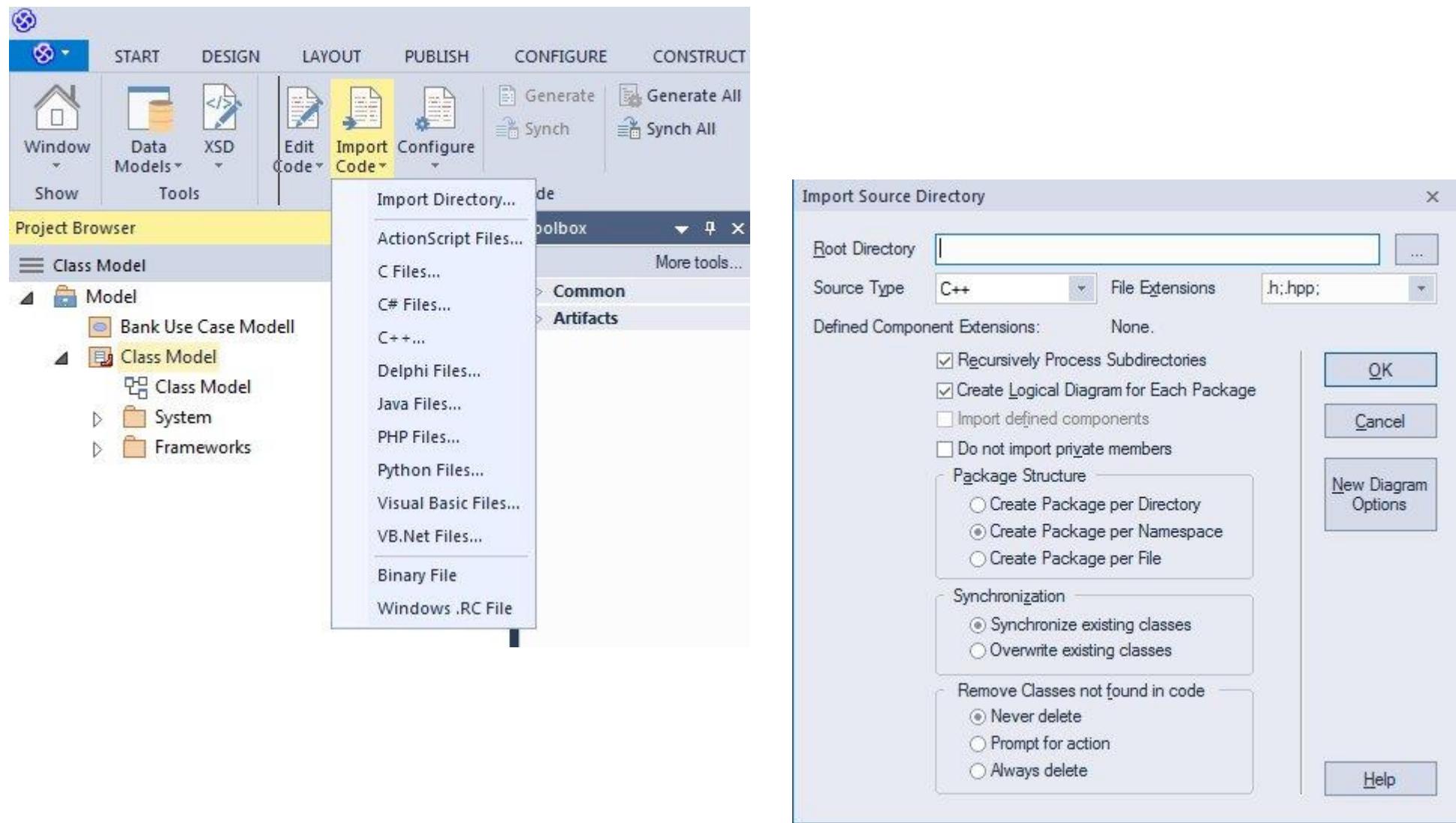
Quellcode Editor im EA

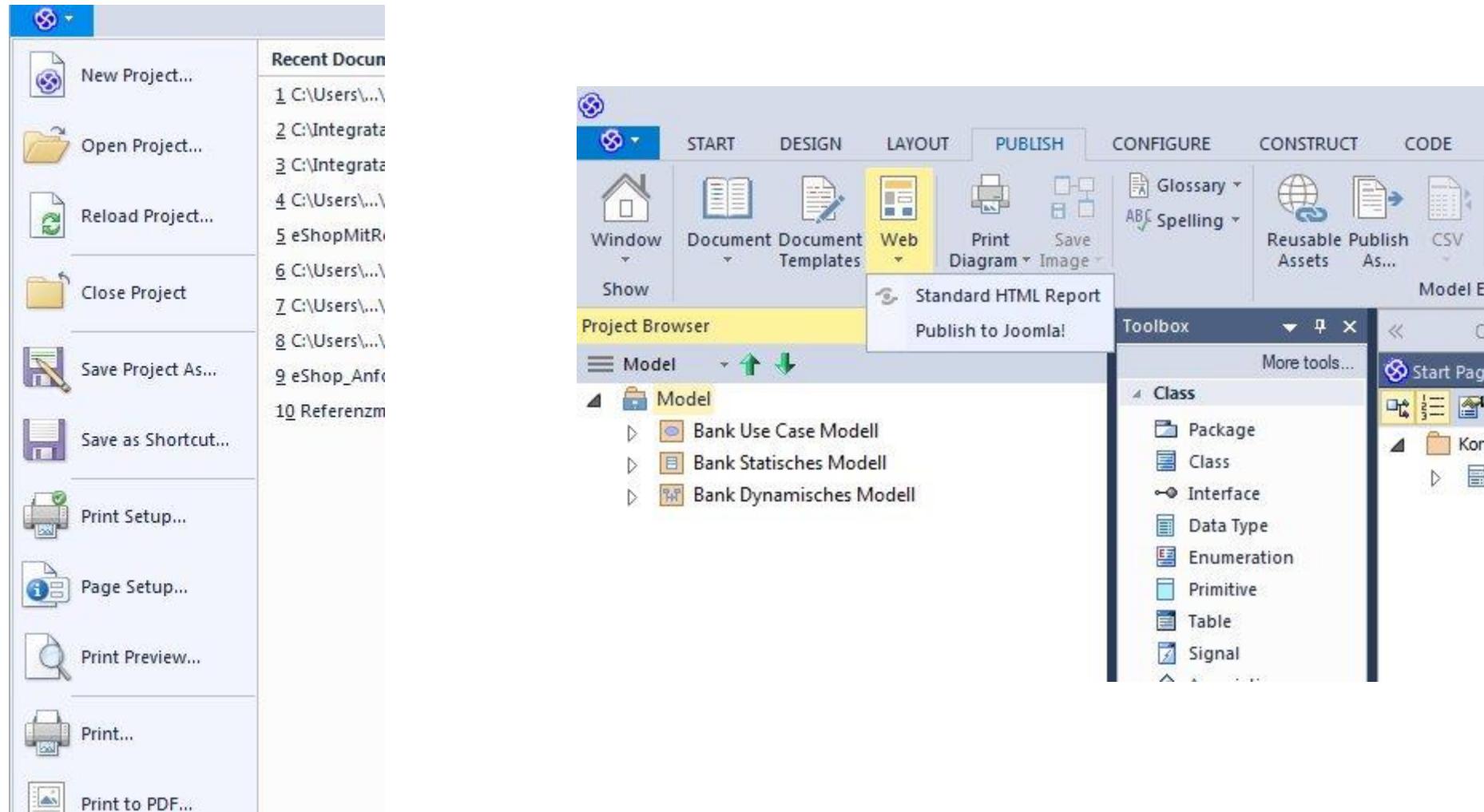


The screenshot shows the Enterprise Architect Quellcode Editor interface. The title bar displays the file path: C:\Integrata\Kurse_NeuMitEA\9024-H - V1.1.412\Konto.java. The left pane shows a class hierarchy tree under the package KontoPaket, with Konto as the active class. The right pane contains the Java code for the Konto class. The code defines an abstract class Konto with attributes kontonummer, kontostand, and status, and methods auszahlen(double) and einzahlen(double). It also includes a constructor, a finalize() method, and a comment block for the auszahlen method.

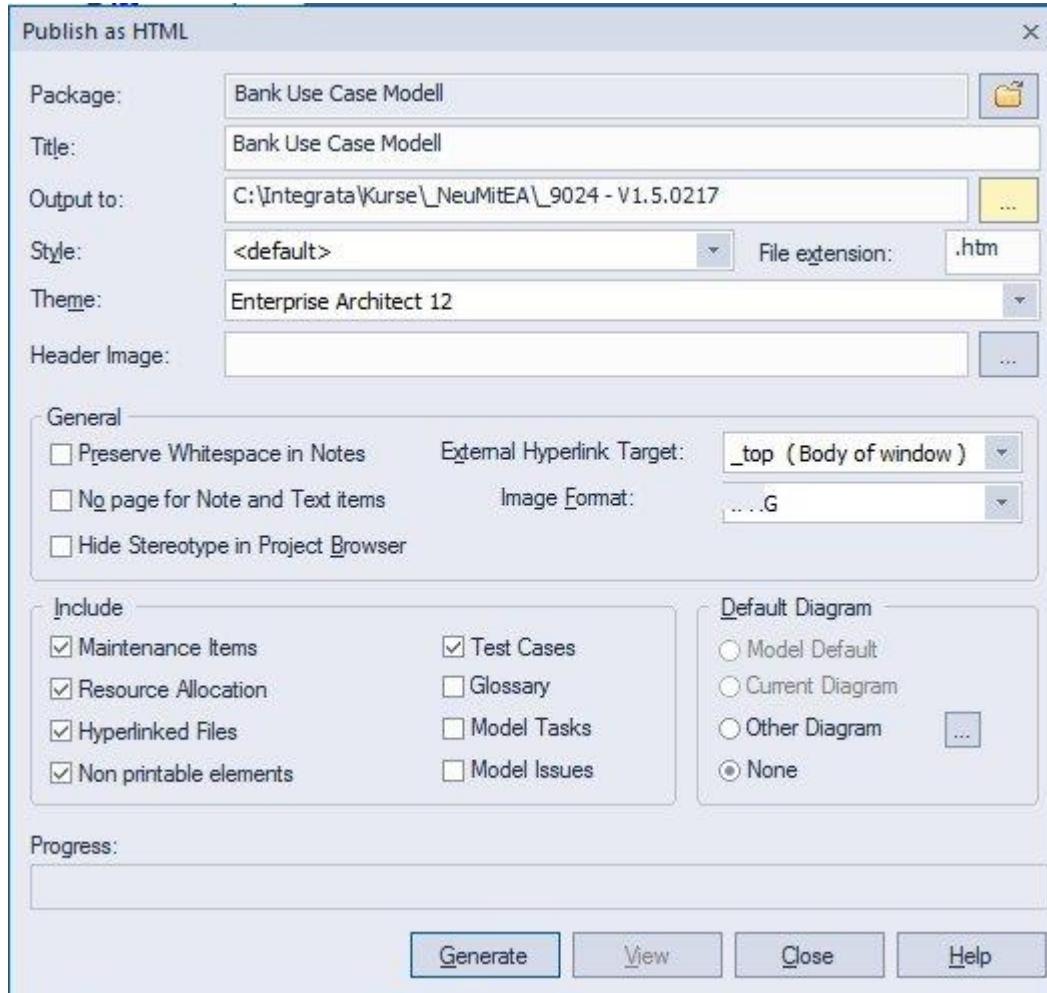
```
1 package KontoPaket;
2 import KundenPaket.Kunde;
3
4 /**
5 * verantwortlich für:
6 * @author Administrator
7 * @version 1.0
8 * @created 13-Apr-2012 08:05:26
9 */
10 public abstract class Konto {
11
12     private int kontonummer;
13     private double kontostand;
14     private String status;
15     public Kunde besitzer;
16
17     public Konto(){
18
19    }
20
21     public void finalize() throws Throwable {
22
23    }
24
25 /**
26 *
27 * @param betrag
28 */
29     public void auszahlen(double betrag){
30
31    }
32
33 /**
34 * erhöhen des Kontostands um betrag
35 *
36 * kontostand += betrag
37 *
38 * @param betrag
39 */
40     public abstract void einzahlen(double betrag);
41 }
```

Das Register Code mit den Code Import Möglichkeiten





- Achtung: Der Fokus ist wichtig



RTF und DOC Dokumentation erzeugen

Generate Documentation

Element: Girokonto

Output to File: | ... Resource Document

Template: Model Report

Output Format: Rich Text Format (RTF)

Cover Page: <none>

Table of Contents: <none>

Stylesheet: <none>

Diagram Theme: <current user theme>

Watermark:

Progress:

Generate Documentation

Generate

Templates

Options

Exclude Filters

Element Filters

Other Filters

Project Constants

Word Substitution

Codepage

Package: Bank Dynamisches Modell

Output to File: | ... Resource Document

Template: Model Report

Output Format: Rich Text Format (RTF) Microsoft Document Format (DOCX) Portable Document Format (PDF)

Cover Page: Rich Text Format (RTF)

Table of Contents: <none>

Stylesheet: <none>

Diagram Theme: <current user theme>

Watermark: | ... Wash image

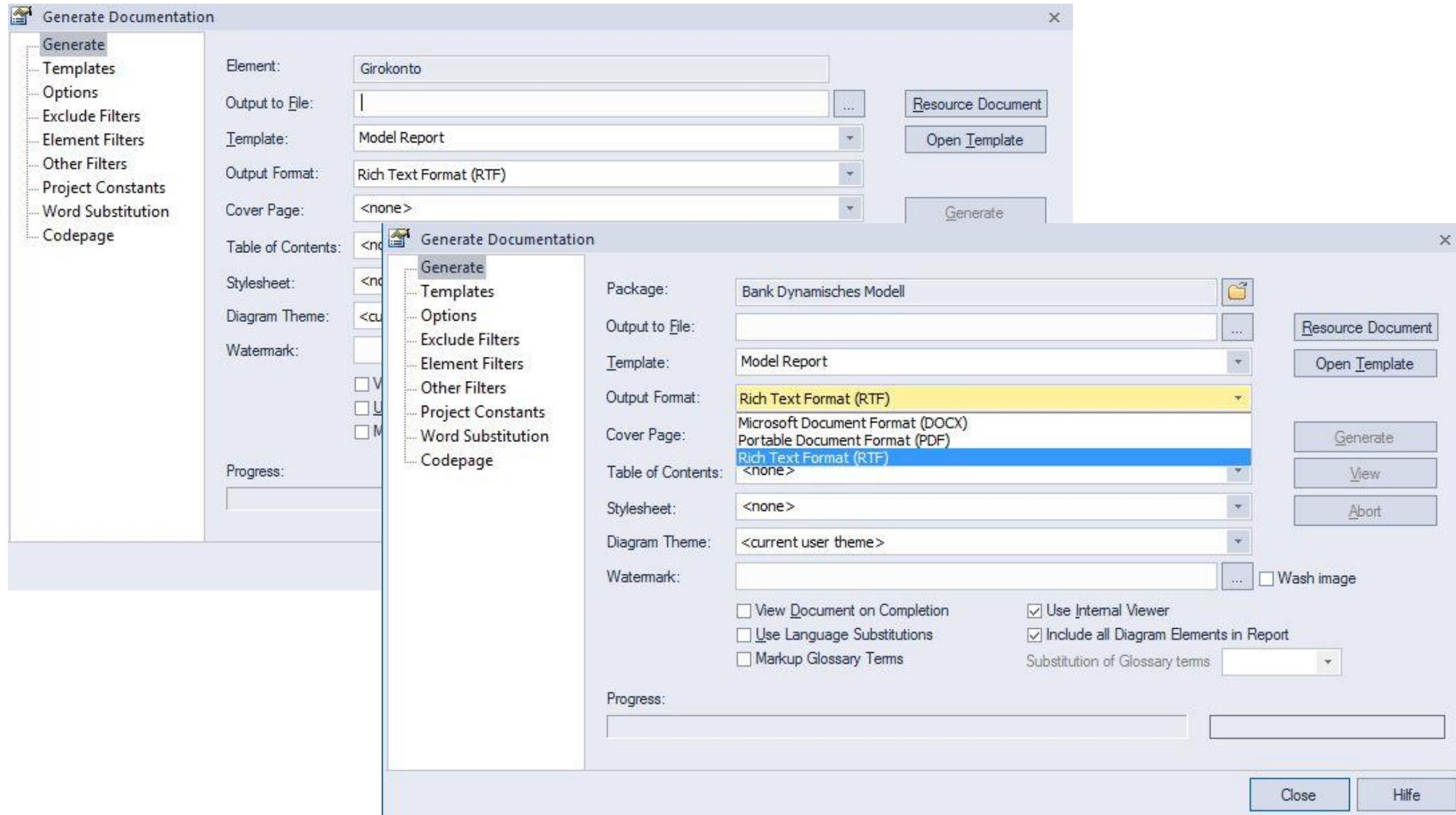
View Document on Completion Use Internal Viewer

Use Language Substitutions Include all Diagram Elements in Report

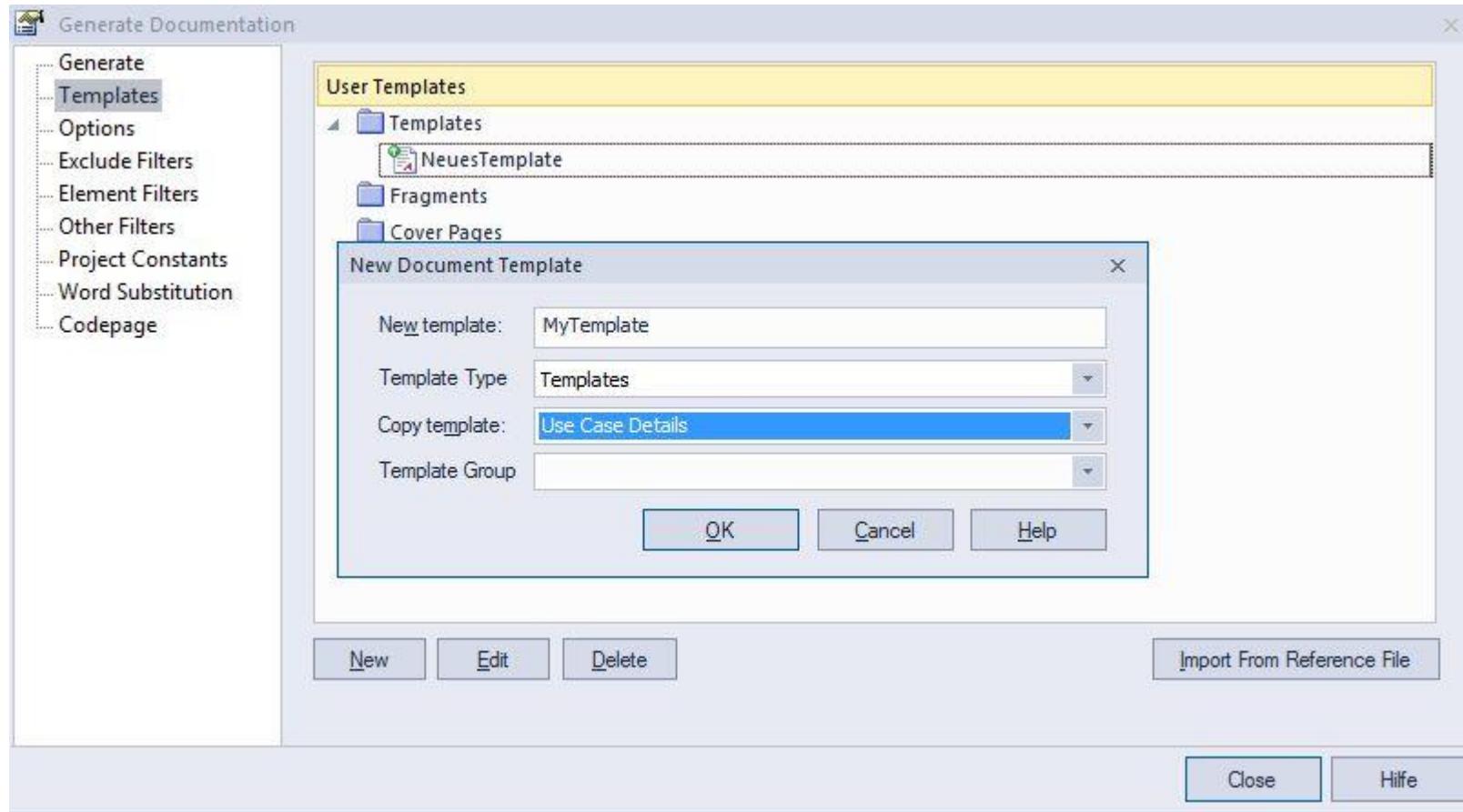
Markup Glossary Terms Substitution of Glossary terms: |

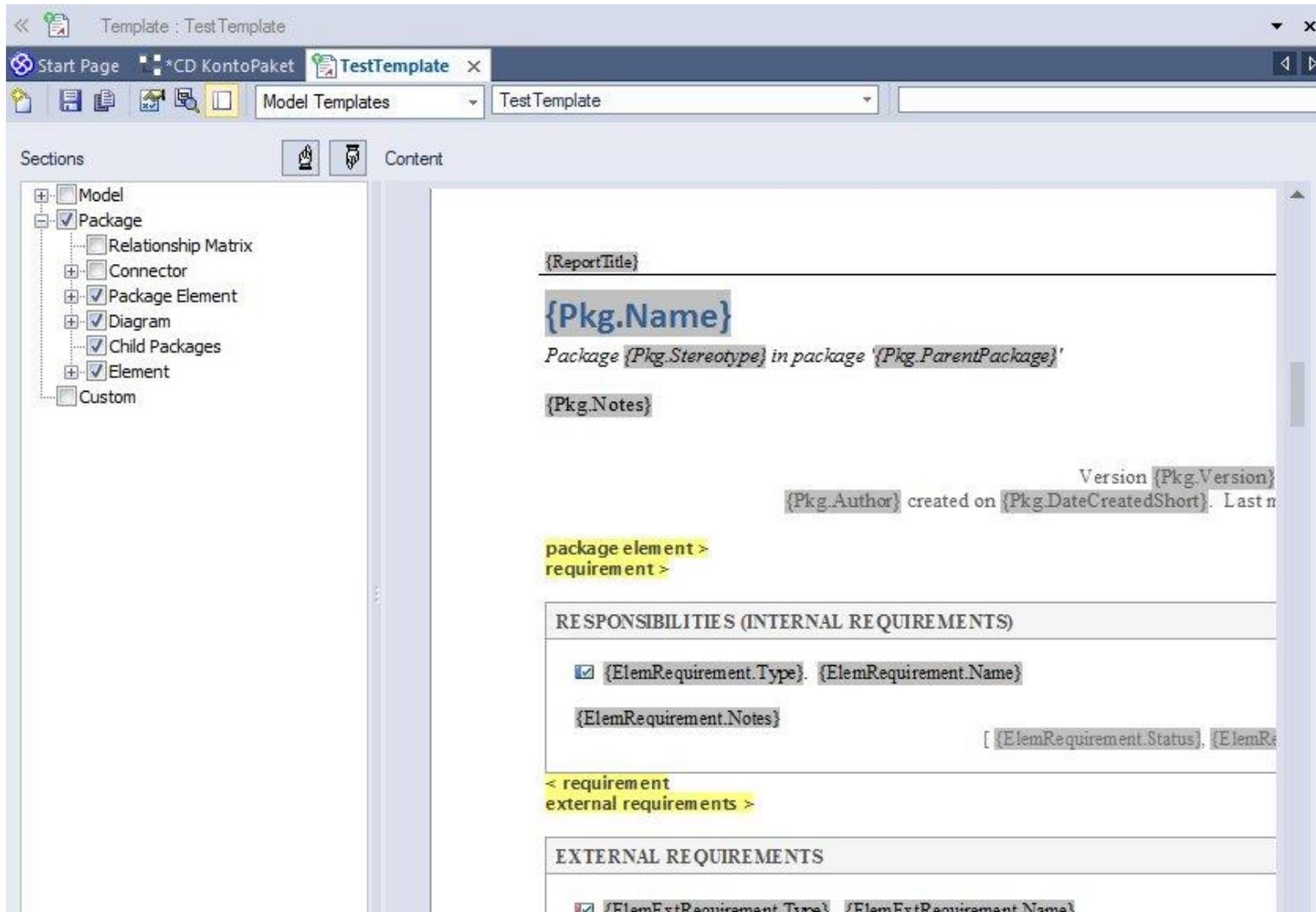
Progress:

Close Hilfe



- Vorhandene Templates können als Basis genommen werden





The screenshot shows the Enterprise Architect Template Editor interface. The title bar reads "Template : TestTemplate". The toolbar includes icons for Start Page, Model Templates, and the current template. The left sidebar lists "Sections" such as Model, Package, and Custom, with Package selected. The main content area displays a template for a package:

{ReportTitle}

{Pkg.Name}

Package {Pkg.Stereotype} in package {Pkg.ParentPackage}'

{Pkg.Notes}

Version {Pkg.Version}
(Pkg.Author) created on {Pkg.DateCreatedShort}. Last n

package element >
requirement >

RESPONSIBILITIES (INTERNAL REQUIREMENTS)

{ElemRequirement.Type}. {ElemRequirement.Name}

{ElemRequirement.Notes}

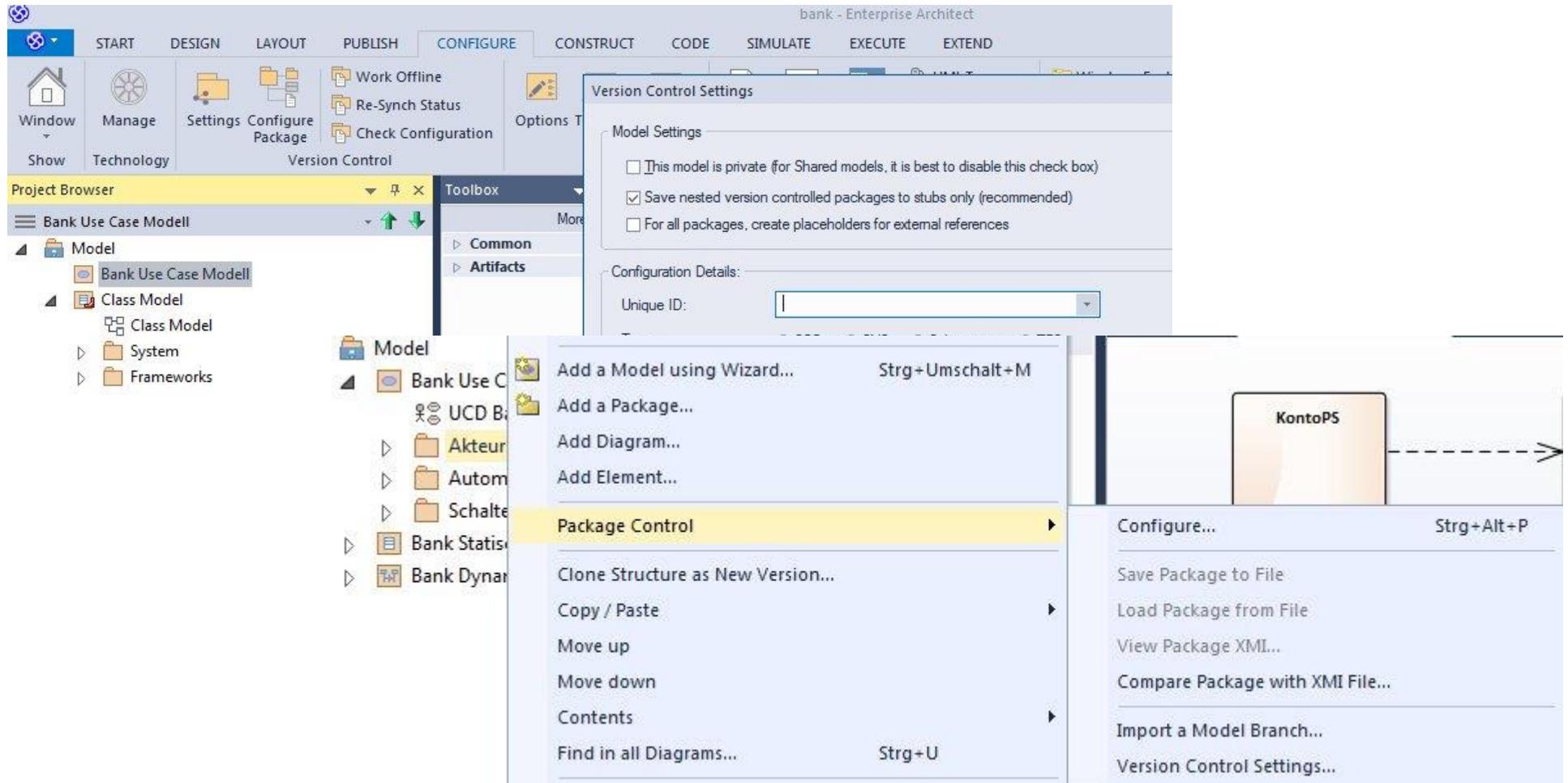
[{ElemRequirement.Status}, {ElemRequirement.Priority}]

< requirement
external requirements >

EXTERNAL REQUIREMENTS

{ElemExtRequirement.Type}. {ElemExtRequirement.Name}

- Verschiedene externe Versionskontrollsystme sind möglich
- Einbindung über Register Configure -> Settings



Versionskontrolle Einstellungen

Version Control Settings

Model Settings

This model is private (for Shared models, it is best to disable this check box)

Save nested version controlled packages to stubs only (recommended)

For all packages, create placeholders for external references

Configuration Details:

Unique ID:

Type: SCC CVS Subversion TFS

New Save Delete

Defined Configurations:

Unique ID	Type	Files	Location

In future, do not prompt for incomplete configurations

Close Help

Package Control Options

Control Package For all packages, create placeholders for external references

Version Control: (None)

XMI Filename: AkteurPaket.xml

UML/XMI Type: Enterprise Architect XMI/UML 1.3

Version ID: 1.0

Owner: Administrator

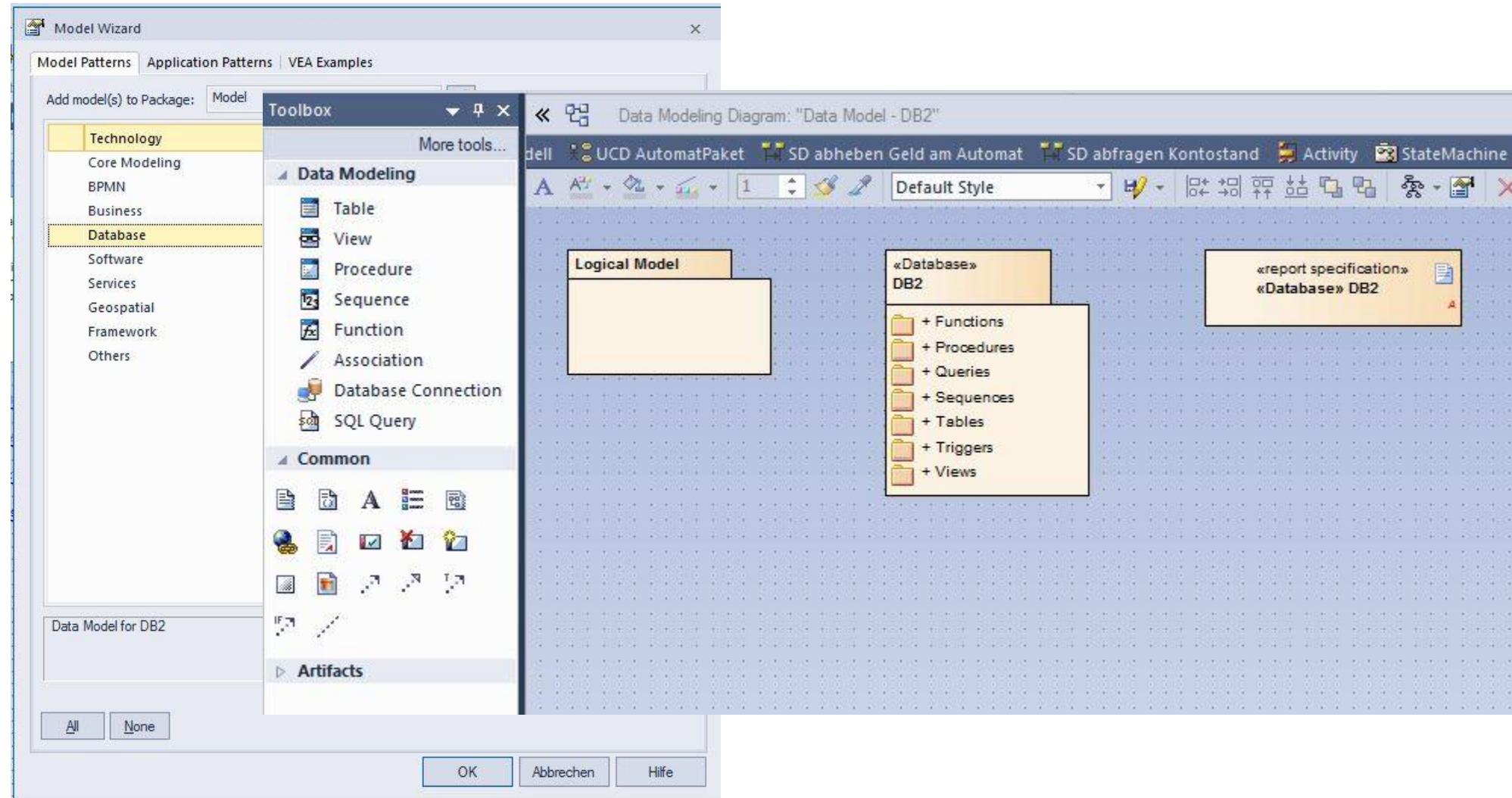
Last Load Date:

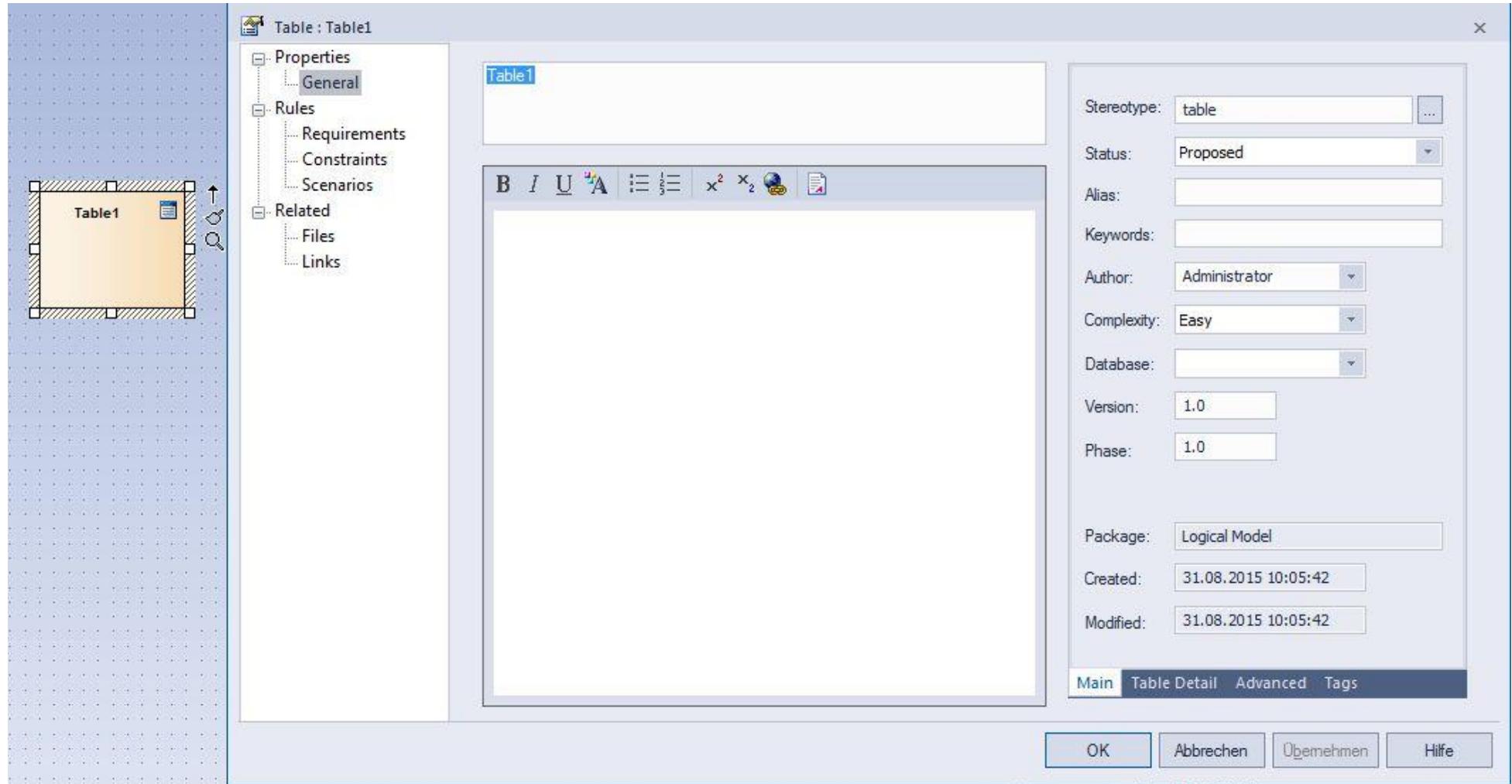
Last Save Date:

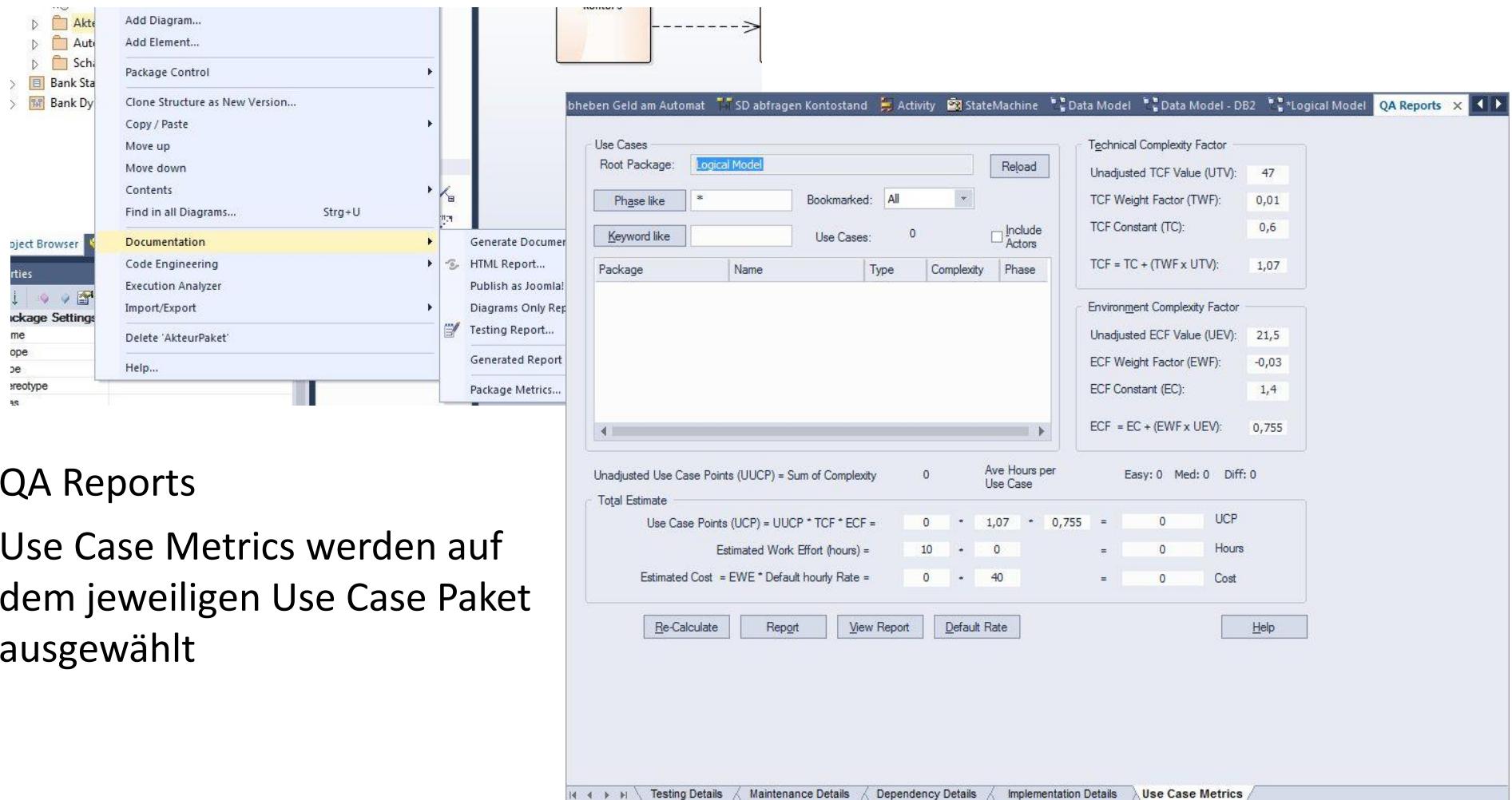
Other Options

Use DTD
 Log Import/Export
 Batch Import
 Batch Export
 Include sub-packages

OK Cancel Help







The screenshot shows the Enterprise Architect interface with the 'QA Reports' tab selected. On the left, the Object Browser displays a package named 'Bank'. A context menu is open over a UML diagram, with 'Documentation' highlighted. The main window shows the 'Use Case Metrics' dialog.

Use Cases:

- Root Package: Logical Model
- Phase like: *
- Bookmarked: All
- Keyword like:
- Use Cases: 0
- Include Actors:

Technical Complexity Factor:

- Unadjusted TCF Value (UTV): 47
- TCF Weight Factor (TWF): 0,01
- TCF Constant (TC): 0,6
- TCF = TC + (TWF x UTV): 1,07

Environment Complexity Factor:

- Unadjusted ECF Value (UEV): 21,5
- ECF Weight Factor (EWF): -0,03
- ECF Constant (EC): 1,4
- ECF = EC + (EWF x UEV): 0,755

Metrics:

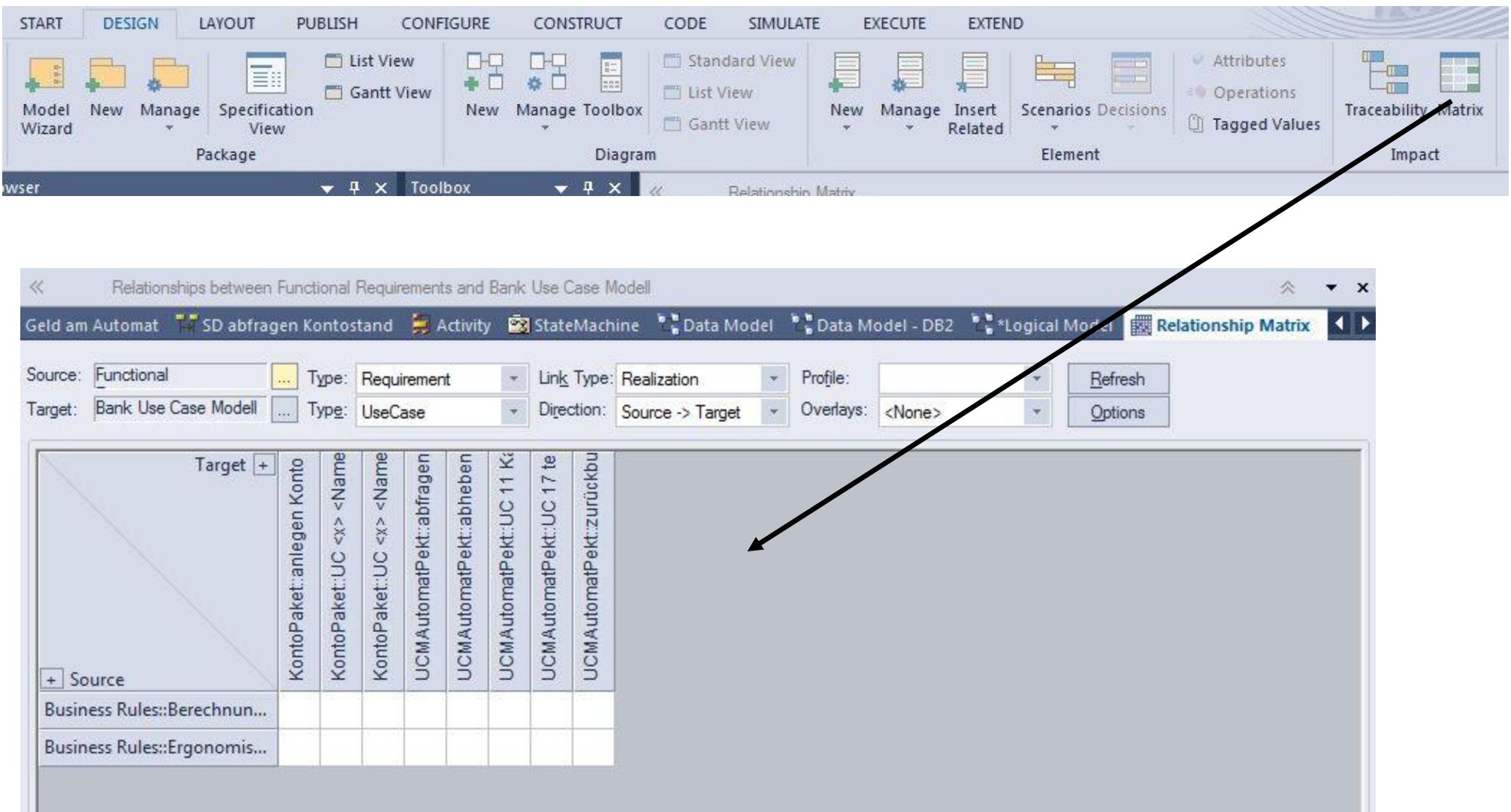
- Unadjusted Use Case Points (UUCP) = Sum of Complexity: 0
- Ave Hours per Use Case: 0
- Easy: 0 Med: 0 Diff: 0

Total Estimate:

- Use Case Points (UCP) = UUCP * TCF * ECF = $0 \cdot 1,07 \cdot 0,755 = 0$ UCP
- Estimated Work Effort (hours) = $10 \cdot 0 = 0$ Hours
- Estimated Cost = EWE * Default hourly Rate = $0 \cdot 40 = 0$ Cost

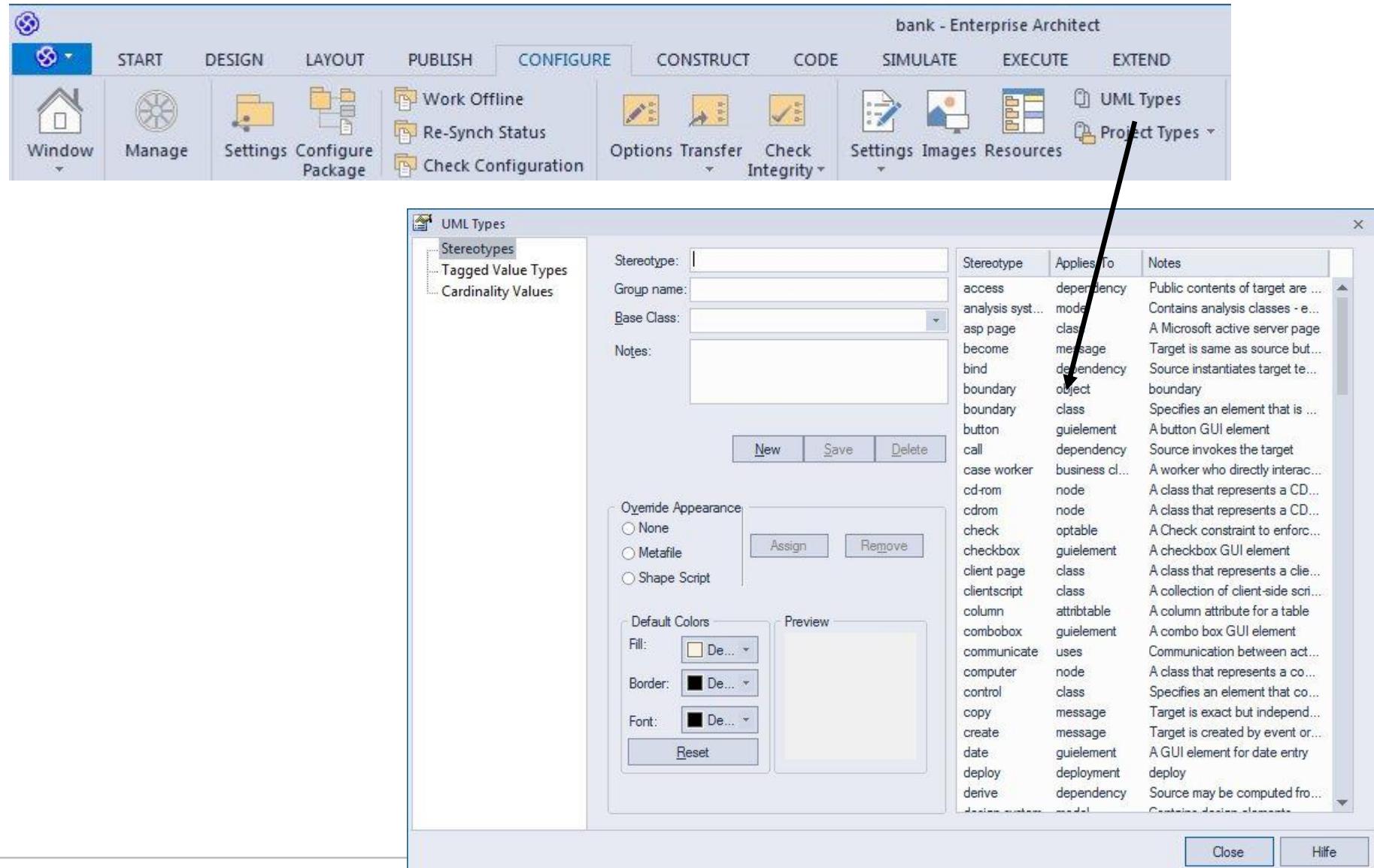
Buttons at the bottom: Re-Calculate, Report, View Report, Default Rate, Help.

Die Relationship Matrix



The screenshot shows the Enterprise Architect interface with the 'Relationship Matrix' tool selected. The toolbar at the top has tabs for START, DESIGN (selected), LAYOUT, PUBLISH, CONFIGURE, CONSTRUCT, CODE, SIMULATE, EXECUTE, and EXTEND. Under the DESIGN tab, there are buttons for Model Wizard, New, Manage, Specification View, List View, Gantt View, and a Toolbox. The Toolbox section includes buttons for New, Manage, and Insert Related, along with Scenario, Decisions, and Tagged Values. On the far right of the toolbar, there are buttons for Attributes, Operations, and Traceability Matrix. The main window title is 'Relationships between Functional Requirements and Bank Use Case Modell'. The toolbar below the title bar includes buttons for Geld am Automat, SD abfragen Kontostand, Activity, StateMachine, Data Model, Data Model - DB2, *Logical Model, and Relationship Matrix (which is highlighted). The Relationship Matrix dialog box shows settings for Source (Functional) and Target (Bank Use Case Modell), Type (Requirement and UseCase), Link Type (Realization), Profile, Direction (Source > Target), Overlays, Refresh, and Options. The matrix grid displays relationships between various elements, with a large black arrow pointing from the 'Relationship Matrix' button in the toolbar to the grid area.

Eigene Stereotypen festlegen



The screenshot shows the Enterprise Architect interface with the 'CONFIGURE' tab selected. A callout arrow points from the 'Stereotype' column header in the 'UML Types' dialog to the 'Stereotype' input field in the dialog itself.

UML Types Dialog (Open)

Stereotypes Tab

- Stereotype: [Empty]
- Group name: [Empty]
- Base Class: [Empty]
- Notes: [Empty]
- Buttons: New, Save, Delete
- Override Appearance:
 - None
 - Metafile
 - Shape Script
- Default Colors:
 - Fill: [Color Chooser]
 - Border: [Color Chooser]
 - Font: [Color Chooser]
- Preview: [Image Placeholder]

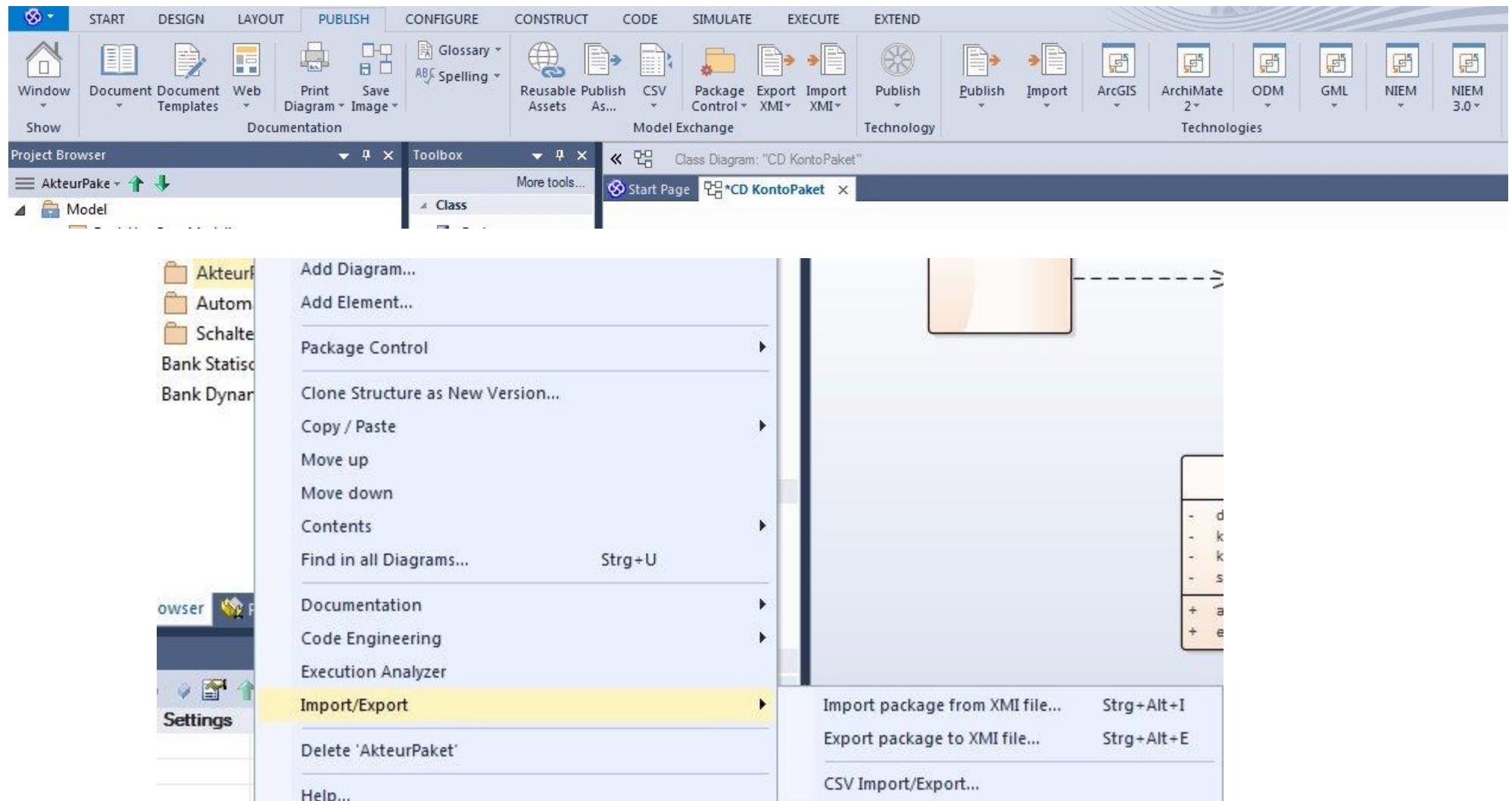
Stereotype List (Table)

Stereotype	Applies To	Notes
access	dependency	Public contents of target are ...
analysis syst...	mode	Contains analysis classes - e...
asp page	clas	A Microsoft active server page
become	message	Target is same as source but...
bind	dependency	Source instantiates target te...
boundary	object	boundary
boundary	class	Specifies an element that is ...
button	guielement	A button GUI element
call	dependency	Source invokes the target
case worker	business cl...	A worker who directly interact...
cd-rom	node	A class that represents a CD...
cdrom	node	A class that represents a CD...
check	optable	A Check constraint to enforc...
checkbox	guielement	A checkbox GUI element
client page	class	A class that represents a clie...
clientscript	class	A collection of client-side scri...
column	attributable	A column attribute for a table
combobox	guielement	A combo box GUI element
communicate	uses	Communication between act...
computer	node	A class that represents a co...
control	class	Specifies an element that co...
copy	message	Target is exact but independ...
create	message	Target is created by event or...
date	guielement	A GUI element for date entry
deploy	deployment	deploy
derive	dependency	Source may be computed fro...

Dialog Buttons

- Close
- Help

Import und Export Aufruf



Import und Export Konfiguration

