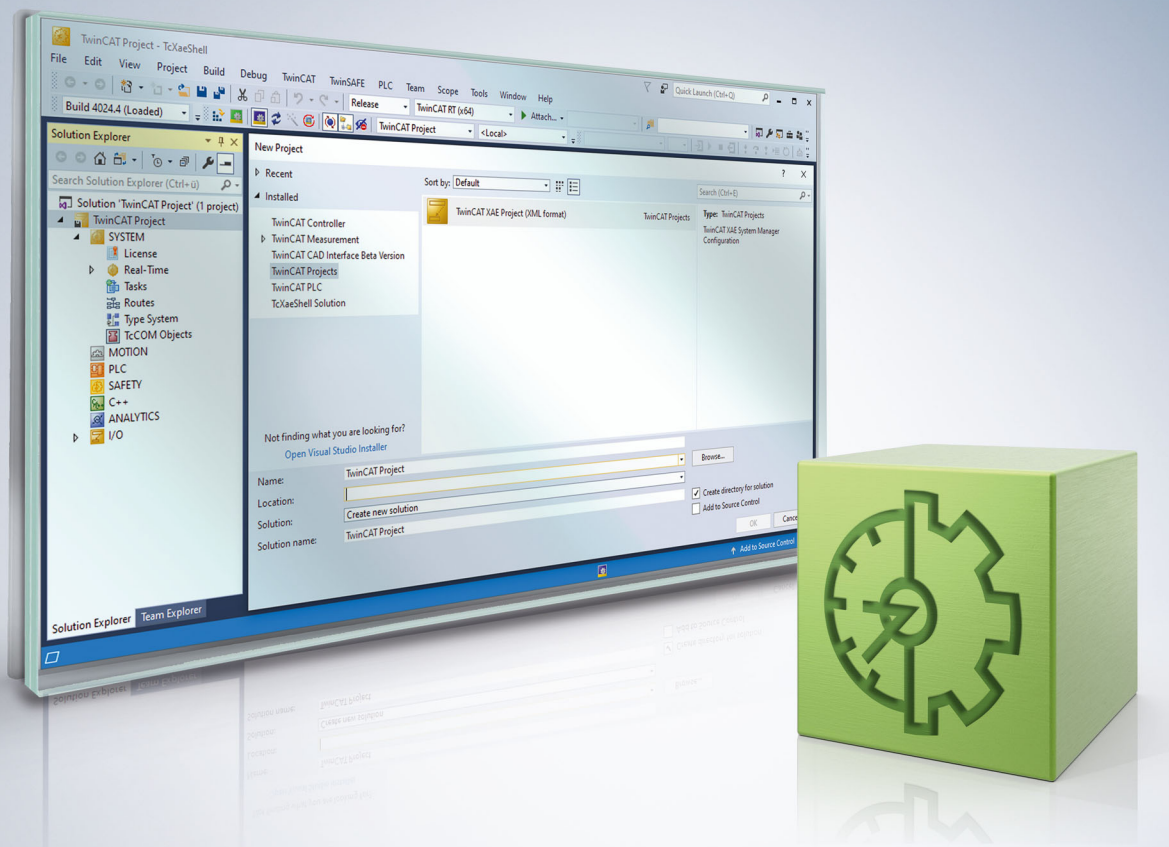
BECKHOFF New Automation Technology

Handbuch | DE

TF1910

TwinCAT 3 | UML



07.12.2023 | Version: 2.6.1

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, stets die aktuell gültige Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiterentwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

Warnungen vor Personenschäden

⚠ GEFAHR

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

⚠ WARNUNG

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

⚠ VORSICHT

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

Warnung vor Umwelt- oder Sachschäden

HINWEIS

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

Information zum Umgang mit dem Produkt

Diese Information beinhaltet z. B.:
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie hier.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed.

2 Überblick

UML allgemein

UML (Unified Modeling Language) ist eine grafische Sprache, die zur Analyse, Konstruktion und Dokumentation von Software genutzt werden kann. Besonders geeignet ist die Verwendung von UML bei objektorientierten Implementierungen. Durch die einheitliche Modellierung der SPS-Applikation entsteht eine allgemein verständliche Softwaredokumentation, die auch von anderen Fachbereichen - abgesehen von der Softwareentwicklung - analysiert und diskutiert werden kann.

Diagrammkategorien

Einige UML-Diagramme lassen sich als Strukturdiagramme und andere als Verhaltensdiagramme kategorisieren. Strukturdiagramme werden im Wesentlichen zur statischen Modellierung und Analyse eingesetzt, da sie die Softwarearchitektur schematisch darstellen. Verhaltensdiagramme dienen hingegen der dynamischen Modellierung. Bei dieser Art der UML-Diagramme handelt es sich um ausführbare Modelle, aus denen direkt Programmcode generiert werden kann.

UML in TwinCAT 3.1 PLC

Mit der Integration von UML (Unified Modeling Language) in TwinCAT 3.1 stehen zwei zusätzliche Editoren zur Modellierung von SPS-Software zur Verfügung. Die bestehenden TwinCAT-SPS-Programmiersprachen werden dabei um das UML-Klassen- und das UML-Zustandsdiagramm erweitert.

-  : die Funktionalität des Objekts **UML-Klassendiagramm**
-  : Programmierung einer POU in der Implementierungssprache **UML-Zustandsdiagramm**

UML-Klassendiagramm

Das UML-Klassendiagramm gehört zur Gruppe der UML-Strukturdiagramme und kann zur schematischen Darstellung der Softwarearchitektur verwendet werden. Dadurch können Objektklassen, die enthaltenen Elemente und die Objektbeziehungen übersichtlich abgebildet werden.

UML-Zustandsdiagramm

Das UML-Zustandsdiagramm ist hingegen Teil der UML-Verhaltensdiagramme und dient der dynamischen Softwaremodellierung. Dabei kann das Zeitverhalten bzw. der zustandsabhängige Ablauf eines Systems grafisch spezifiziert werden. Beim Kompilieren des Zustandsdiagramms wird Programmcode generiert, sodass die Zustandsmaschine direkt ausgeführt werden kann. Der Entwicklungsprozess wird durch ein mögliches Debuggen im Online-Modus unterstützt.

Vorteile

Die Vorteile, UML-Diagramme für die Analyse, Konstruktion und/oder Dokumentation von Software einzusetzen, sind vielfältig. Die wesentlichen Aspekte werden in den folgenden Punkten genannt:

- Zunächst bietet eine grafische Darstellung, bei der der Fokus nicht auf technischen Details liegt, eine gute Übersicht, um Softwareanforderungen vor der Implementierung zu überprüfen. Dies beugt einer unvollständigen oder fehlerhaften Applikationsumsetzung vor.
- Durch die grafische Abbildung des Steuerungscodes wird die Entwicklung einer durchdachten Softwarearchitektur erheblich unterstützt. Eine solche Architektur ist die Basis, um auch komplexe Systeme bzw. Anforderungen einfach und zielgerichtet umzusetzen. Des Weiteren kann eine durchdachte Softwarearchitektur dazu beitragen, autarke Module zu entwickeln, die zeit- und kostensparend wiederverwendet werden können. Generell führt eine gut geplante Software in der Regel zu weniger Programmierfehlern und somit zu einer höheren Codequalität.
- Der grafische Zugang zur Software erleichtert die Wartung und das Debugging.
- Mithilfe von UML-Diagrammen entsteht üblicherweise eine allgemein verständliche Dokumentation der Software. Diese kann zum einen als Koordinierungswerkzeug im Entwicklungsteam eingesetzt werden, um zum Beispiel Ideen und Konzepte auszutauschen oder Anforderungen festzulegen. Zum anderen kann die Steuerungsapplikation mittels der UML-Diagramme gegenüber anderen Technologiespezialisten, wie beispielsweise Maschinenbauern oder Prozesstechnikern, dargestellt werden.

3 Installation

3.1 Systemvoraussetzungen

UML Klassendiagramm:

Entwicklungsumgebung
TwinCAT v3.1.4018.16

UML Zustandsdiagramm:

Entwicklungsumgebung	Zielplattform	Einzubindender Bibliotheks-platzhalter
TwinCAT v3.1.4016.0	PC oder CX (x86, x64, ARM)	UML State Chart Types

3.2 Installation

Engineering:

Die Engineering-Komponenten der Function "TF1910 | TC3 UML" sind im TC3.1 XAE-Setup enthalten (sowohl UML-Klassen- als auch UML-Zustandsdiagramm).

Diese können Sie nach der Installation des XAE direkt nutzen. Hierfür wird keine Lizenz benötigt.

Runtime:

Falls das UML-Zustandsdiagramm verwendet wird, werden die erforderlichen Laufzeitkomponenten mit dem TC3.1 XAE- oder XAR-Setup installiert.

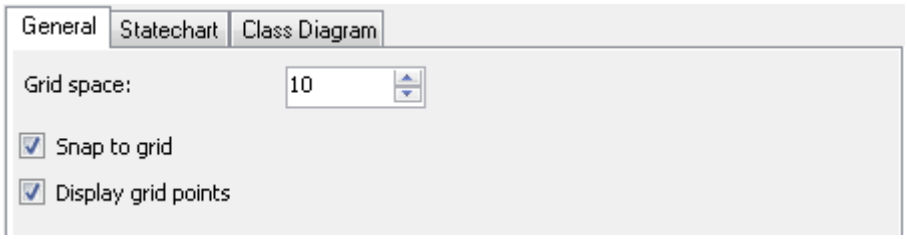
Für die zusätzliche Laufzeitkomponente UML Zustandsdiagramm benötigen Sie die TF1910-Laufzeitlizenz. Sehen Sie hierzu auch die Dokumentation zur Lizenzierung.
Ohne die Laufzeitlizenz für TF1910 können Sie das UML-Zustandsdiagramm mittels 7-Tage-Testlizenz testen. Die Testlizenz können Sie während der Testphase immer wieder neu erzeugen. Sehen Sie hierzu auch die Dokumentation zu den TwinCAT-3-Testlizenzen.

4 Einstellungen

4.1 Optionen

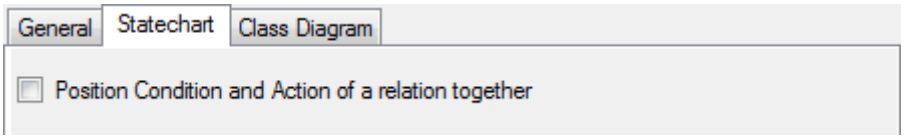
In den TwinCAT UML Optionen (**Extras > Optionen > TwinCAT > PLC Environment > UML**) konfigurieren Sie Einstellungen, die die UML-Editoren projektweit betreffen. Geänderte Optionen sind mit Schließen des Dialogs auch in bereits offenen UML-Editoren gültig.

Allgemein



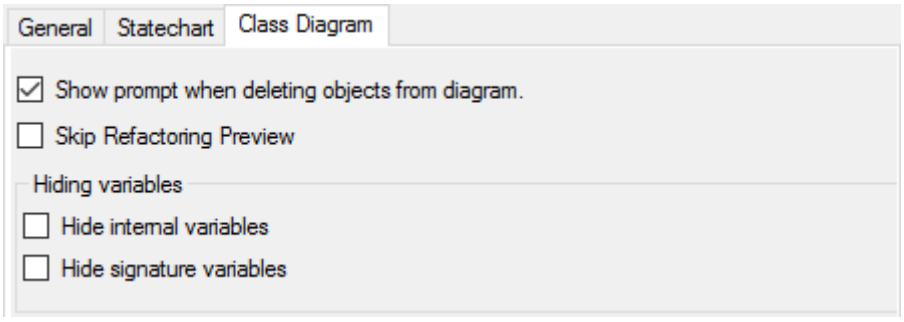
- **Rasterabstand** (Grid space): Geben Sie einen Ganzzahlwert an, der als Rasterabstand in Pixel verwendet wird. Default: 10
- **Am Raster ausrichten** (Snap to grid): Aktivieren Sie diese Option, um alle Elemente in den UML-Editoren am Raster auszurichten.
- **Rasterpunkte anzeigen** (Display grid points): Aktivieren Sie diese Option, um die Rasterpunkte in den UML-Editoren anzuzeigen.

Zustandsdiagramm



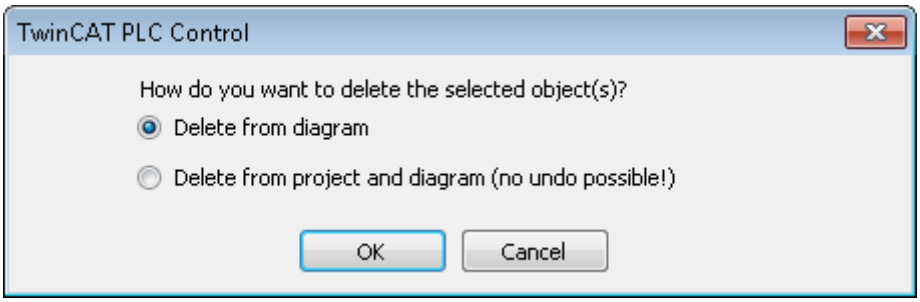
- **Bedingung und Aktion einer Beziehung gemeinsam positionieren** (Position Condition and Action of a relation together):
  Aktivieren Sie die Option, um im Zustandsdiagramm eine Wächterbedingung und eine Aktion, die zur selben Transition gehören, synchron zu verschieben.

Klassendiagramm



- **Eingabeaufforderung, wenn Objekte aus dem Diagramm gelöscht werden** (Show prompt when deleting objects from diagram):
  Objekte können entweder nur aus dem Diagramm oder aus dem Diagramm und aus dem Projekt gelöscht werden. Dabei gibt es zwei mögliche Vorgehensweisen:
  - Wird im Klassendiagramm ein Objekt markiert, erscheinen über dem Objekt zwei Befehlssymbole, um das Objekt nur aus dem Diagramm oder aus dem Diagramm und aus dem Projekt zu löschen.

◦ Alternativ kann ein markiertes Objekt über die Taste [Entf] gelöscht werden. Ist die Option, ein Auswahlfenster anzuzeigen, deaktiviert, wird das Objekt standardmäßig nur aus dem Diagramm gelöscht. Bei aktivierter Einstellung erscheint beim Löschen ein Auswahlfenster, um zu konfigurieren, ob das Objekt nur aus dem Diagramm oder auch aus dem Projekt gelöscht werden soll.



- **Refactoring-Vorschau überspringen** (Skip Refactoring Preview):
  Wenn diese Option aktiviert ist und wenn im Diagramm Refactoring angestoßen wird, wird die projektweite Änderung durchgeführt, ohne vorher den Dialog **Refactoring** mit einer Vorschau aller Änderungsstellen zu öffnen.

Variablen verstecken

Verfügbar ab TwinCAT 3.1 Build 4026

Um den Umfang an Informationen innerhalb des Klassendiagramms auf den gewünschten Fokus zu reduzieren, stehen die folgenden Optionen zur Verfügung.

- **Interne Variablen verstecken** (Hide internal variables):
  Wenn diese Option aktiviert ist, werden interne Variablen nicht im Klassendiagramm angezeigt. Dazu gehören VAR, VAR_TEMP, VAR_STAT und VAR_INST.
- **Signaturvariablen verstecken** (Hide signature variables):
  Wenn diese Option aktiviert ist, werden Signatur- bzw. Schnittstellenvariablen nicht im Klassendiagramm angezeigt. Dazu gehören VAR_INPUT, VAR_OUTPUT und VAR_IN_OUT.

4.2 UML Compiler-Version

Die UML Compiler-Version kann in den folgenden Dialogen geändert werden:

- SPS-Projekteigenschaften
- ProfileUpdate-Dialog

Des Weiteren steht bezüglich der UML Compiler-Version die folgende Option zur Verfügung:

- Autoupdate Uml Profile

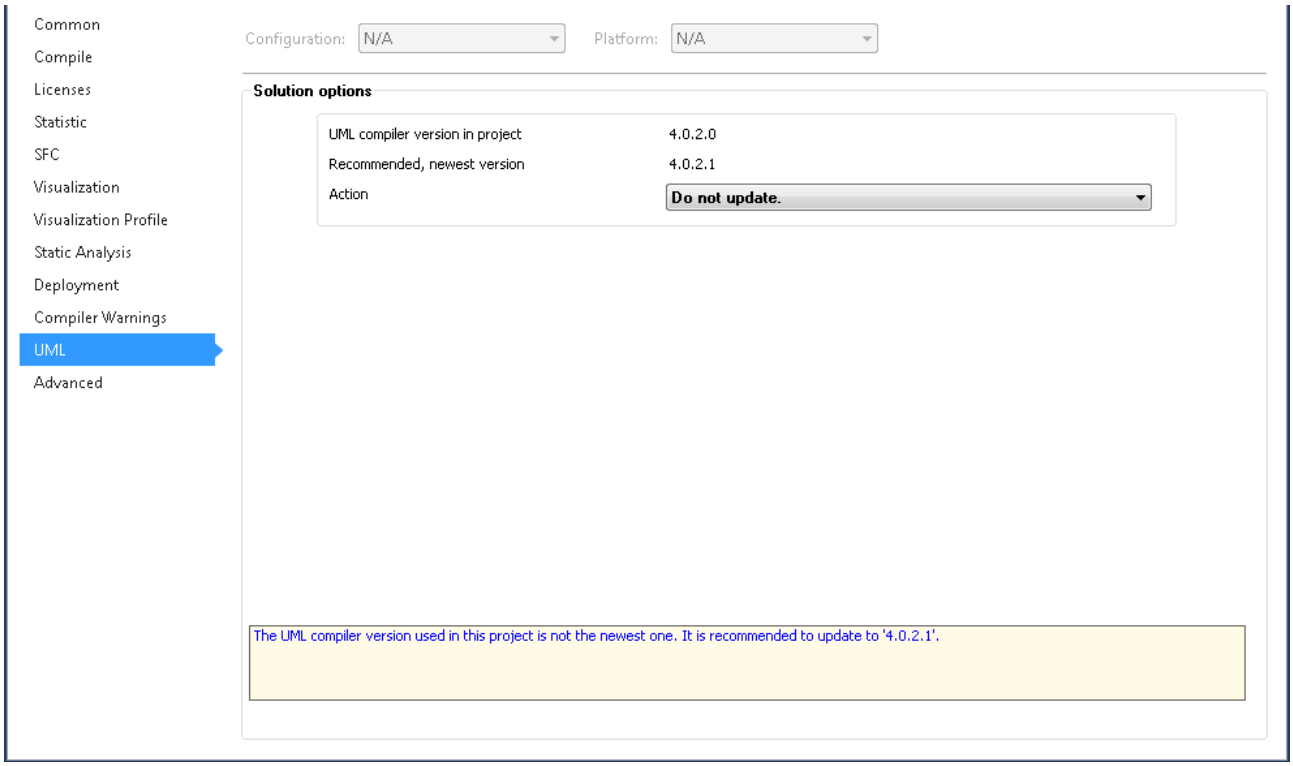
Die UML Compiler-Version ist nur bei Verwendung des UML Zustandsdiagramms relevant.

Geltungsbereich der Einstellung „UML Compiler-Version“

Die Einstellung der UML Compiler-Version ist eine „Solution option“ und wirkt sich daher nicht nur auf das SPS-Projekt aus, dessen Eigenschaften Sie aktuell konfigurieren. Die eingestellte UML Compiler-Version betrifft alle SPS-Projekte, die sich in der Entwicklungsumgebung befinden.

SPS-Projekteigenschaften

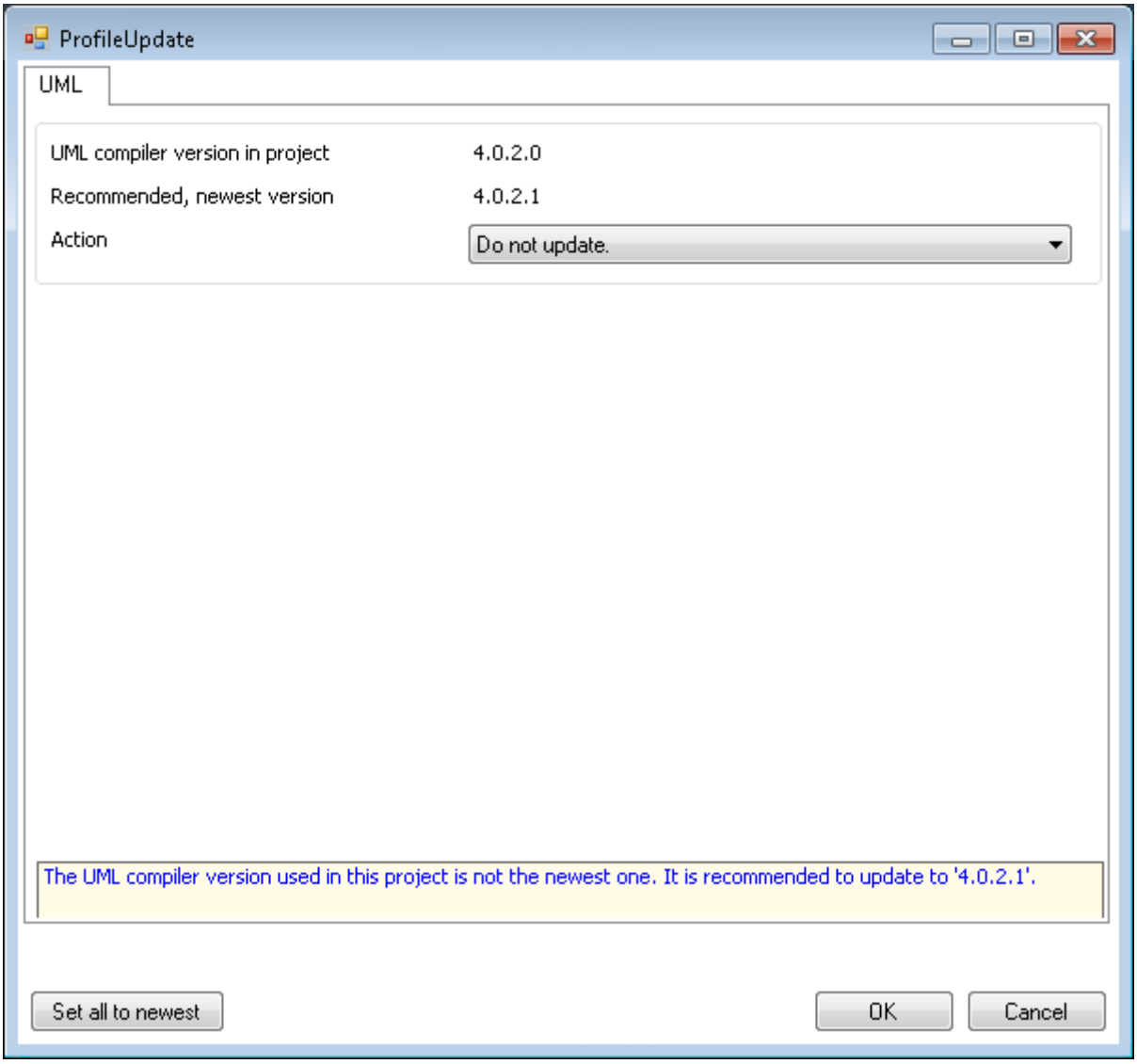
In den Eigenschaften des SPS-Projekts können Sie die UML Compiler-Version ändern.
Öffnen Sie hierzu die SPS-Projekteigenschaften und klicken Sie auf die Kategorie **UML**.



- **UML compiler version in project:** Zeigt die UML Compiler-Version an, die aktuell im Projekt verwendet wird.
- **Recommended, newest version:** Zeigt die neueste verfügbare UML Compiler-Version an, deren Verwendung empfohlen wird.
- **Action**: In diesem Dropdown-Menü können Sie die gewünschte Aktion auswählen. Die Aktion wird direkt ausgeführt, wenn Sie sie anwählen. Beispielaktionen:
  - Do not update.
  - Update to 4.0.2.1

ProfileUpdate-Dialog

Wenn Sie ein SPS-Projekt öffnen, in dem eine veraltete UML Compiler-Version verwendet wird, erscheint im Meldungsfenster eine entsprechende Warnung („neue Version für UML gefunden“). Per Doppelklick auf diese Warnung können Sie den ProfileUpdate-Dialog öffnen, in welchem Sie die UML Compiler-Version ändern können.



- **UML compiler version in project:** Zeigt die UML Compiler-Version an, die aktuell im Projekt verwendet wird.
- **Recommended, newest version:** Zeigt die neueste verfügbare UML Compiler-Version an, deren Verwendung empfohlen wird.
- **Action**: In diesem Dropdown-Menü können Sie die gewünschte Aktion auswählen. Die Aktion wird ausgeführt, wenn der Dialog über **OK** bestätigt wird. Beispielaktionen:
  - Do not update.
  - Update to 4.0.2.1
- **Set all to newest:** Klicken Sie auf die Schaltfläche, um die UML Compiler-Version auf die neueste Version zu setzen.

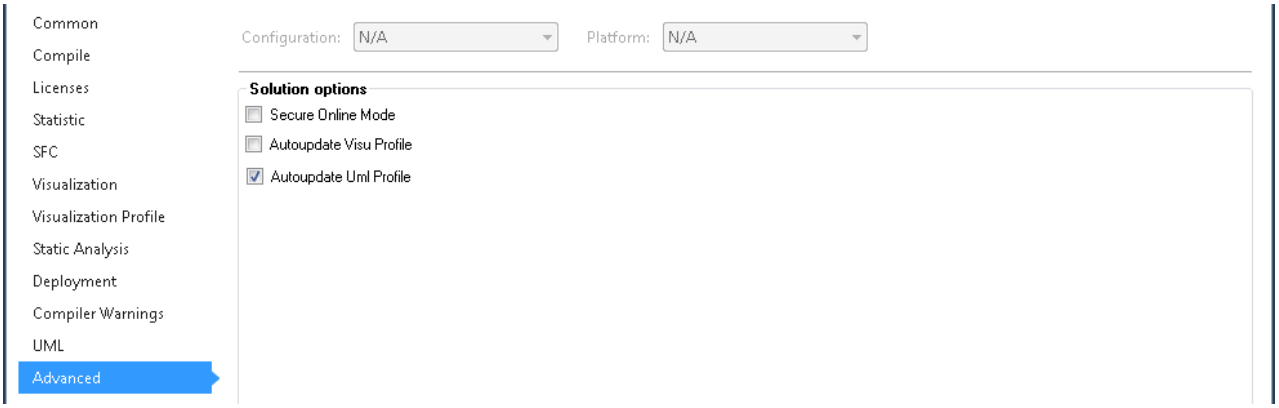
Autoupdate Uml Profile

In den Eigenschaften des SPS-Projekts steht in der Kategorie **Advanced** die Option **Autoupdate Uml Profile** zur Verfügung, mit der Sie das automatische Update-Verhalten der UML Compiler-Version konfigurieren können.

Wenn Sie ein SPS-Projekt öffnen, in dem eine veraltete UML Compiler-Version verwendet wird, erscheint im Meldungsfenster eine entsprechende Warnung („neue Version für UML gefunden“).

: In einem solchen Fall wird die UML Compiler-Version automatisch auf die neueste Version gesetzt, falls die Option **Autoupdate Uml Profile** aktiviert ist. Bei einem solchen automatischen Update der UML Compiler-Version wird im Meldungsfenster eine entsprechende Warnung angezeigt (z.B. „UML set from '4.0.2.0' to '4.0.2.1'“).

: Wenn die Option **Autoupdate Uml Profile** deaktiviert ist, wird die UML Compiler-Version nicht automatisch geändert. Per Doppelklick auf die Warnung „neue Version für UML gefunden“ können Sie den ProfileUpdate-Dialog [▶ 12] öffnen, in welchem Sie die UML Compiler-Version manuell ändern können.



5 Befehle

Gemeinsame Befehle aller UML-Diagramme

Wenn ein UML-Diagramm im Projektbaum selektiert ist, ist folgender Befehl im Kontextmenü verfügbar:

- Bitmap aus dem selektierten UML-Diagramm erzeugen [▶ 15]

Innerhalb des Editors eines UML-Diagramms kann die Ausrichtung der Elemente an einem Raster aktiviert bzw. deaktiviert werden.

- Raster aktiviert [▶ 16]
- Raster deaktiviert [▶ 16]

5.1 Bild erstellen

Der Befehl **Bild erstellen** ist verfügbar, wenn

- der Projektbaum fokussiert ist
- und dort ein Klassen- oder Zustandsdiagramm selektiert und dessen Kontextmenü geöffnet ist

Verwenden Sie den Befehl, um von einem Klassen- oder Zustandsdiagramm eine Grafik im BMP-, PNG- oder JPG-Format zu erzeugen und in ihrem Dateisystem zu speichern.

Wenn Sie den Befehl **Bild erstellen** ausführen, öffnet der dazugehörige Dialog. Innerhalb dieses Dialogs stehen Ihnen die folgenden Konfigurationsmöglichkeiten zur Verfügung.

Größe der längeren Seite (in Pixel):

- Der angezeigte Wert bezieht sich auf die Kantenlänge der Grafik. Abhängig vom Layout des UML-Diagramms wird die Grafik entweder im Quer- oder in Hochformat exportiert. Wenn das Diagramm ein Querformat hat, enthält der Wert die Länge der Grafik. Wenn das Diagramm ein Hochformat hat, enthält der Wert die Höhe der Grafik.

Bitmap auf Desktop speichern:

- ☑ : Bei aktivierter Option wird der Objektname als Dateiname verwendet, zum Beispiel "FB_UML_SC.bmp" bei einem Objekt mit dem Namen "FB_UML_SC". Als Dateityp wird das BMP-Format verwendet.
  - Wenn Sie die Einstellung mit [OK] bestätigen, wird die Grafik auf dem Desktop gespeichert. *Hinweis* **Bei dieser Option werden auf dem Desktop ggf. bestehende Dateien mit diesem Namen ohne Warnung überschrieben.**
  - Wenn Sie [Abbrechen], wird der Dialog beendet, ohne etwas zu speichern.
- ☐ : Der Name, das Verzeichnis und das Format der Grafik sind editierbar.
  - Wenn Sie die Einstellung mit [OK] bestätigen, öffnet der Standarddialog zum Speichern einer Datei. Geben Sie dort einen Verzeichnis- und einen Dateinamen ein, wählen Sie BMP, PNG oder JPG als Format und beenden Sie den Dialog mit [Speichern]. Die Grafik wird in Ihrem Dateisystem gespeichert.

Zugriff per Automation Interface:

Der Befehl **Bild erstellen** ist per ConsumeXML-Methode auf dem POU-Knoten per Automation Interface erreichbar.

```
<TreeItem>
 <PlcPouDef>
  <Commands>
   <CreateBitmapCommand>
    <Active>true</Active>
    <Parameters>
     <FileName>d:\tmp\Bitmap.png</FileName>
```

```
    <Width>1200</Width>
    <Height>-1</Height>       (* 1 adapts the height to the width to keep the ratio;
                                 alternatively, you can enter a fix height which might deform the bitmap *)
   </Parameters>
  </CreateBitmapCommand>
 </Commands>
</PlcPouDef>
</TreeItem>
```

5.2 Raster aktiviert

Symbol: 

Wenn dieser Befehl im Kontextmenü innerhalb eines Klassen- oder Zustandsdiagramm-Editors verfügbar ist, wird ein Element bei einer Positionsänderung an einem Raster ausgerichtet. Die Option **Am Raster ausrichten** (Snap To Grid) in den TwinCAT UML Optionen (**Extras > Optionen > TwinCAT > PLC Environment > UML**) ist aktiviert.

Wenn Sie den Befehl ausführen, wechselt das Diagramm nach **Raster deaktiviert**. Die Option wird deaktiviert.

Siehe dazu:

- Raster deaktiviert [▶ 16]
- Optionen [▶ 10]

5.3 Raster deaktiviert

Symbol: 

Wenn dieser Befehl im Kontextmenü innerhalb eines Klassen- oder Zustandsdiagramm-Editors verfügbar ist, wird ein Element bei einer Positionsänderung **nicht** an einem Raster ausgerichtet. Die Option **Am Raster ausrichten** (Snap To Grid) in den TwinCAT UML Optionen (**Extras > Optionen > TwinCAT > PLC Environment > UML**) ist deaktiviert.

Wenn Sie den Befehl ausführen, wechselt das Diagramm nach **Raster aktiviert**. Die Option wird aktiviert.

Siehe dazu:

- Raster aktiviert [▶ 16]
- Optionen [▶ 10]

6 UML-Klassendiagramm

Bitte beachten Sie zusätzlich zu den folgenden Informationen auch die Samples [▶ 96], die eine erste Einführung in das Tool geben.

6.1 Grundlagen

Das UML-Klassendiagramm kann verwendet werden, um die Struktur eines (komplexen) Systems zu dokumentieren, zu analysieren, zu gestalten und zu erweitern. Dabei können Klassen entworfen und Beziehungen zwischen ihnen abgebildet werden. Die übersichtliche Darstellung von PLC-Programmelementen umfasst u.a. Vererbungs- und Implementierungsbeziehungen, sodass Zusammenhänge visuell deutlich werden. Ein UML-Klassendiagramm kann daher optimal für eine grafische Systemdokumentation verwendet werden und bietet eine verständliche Basis, um technische Inhalte zu vermitteln.

Der Klassendiagrammeditor stellt Elemente zur Verfügung, die die Objektorientierung des Projekts abbilden. Da der Editor in den SPS-Bereich der TwinCAT 3 Entwicklungsumgebung eingebettet ist, ist eine automatische Generierung von Code möglich. Umfangreiche Features und Tools stehen integriert zur Verfügung.

Das Klassendiagramm kann in zwei Richtungen verwendet werden. Zum einen ist es möglich, die bestehende Projektstruktur ins Klassendiagramm zu importieren bzw. ausgewählte Elemente der bestehenden Projektstruktur zum Klassendiagramm hinzuzufügen. Dadurch kann die bereits existierende Softwarearchitektur dokumentiert und analysiert werden. Zum anderen bietet das Klassendiagramm die Möglichkeit, bestehende SPS-Elemente bzw. die bestehende Projektstruktur zu verändern und zu erweitern. Diese Modifikation kann mithilfe des Klassendiagramm-Editors [▶ 25] und der dazugehörigen Elemente [▶ 25] des Werkzeugkastens durchgeführt werden. Hierüber kann die Softwarearchitektur verändert und erweitert sowie gleichzeitig dokumentiert und analysiert werden.

Begriffe aus der Objektorientierung	Synonym in IEC 61131-3
Klasse (UML: *class*)	POU-Typen: • Programm (PRG): PROGRAM • Funktionsbaustein (FB): FUNCTION_BLOCK • Funktion (FUN): FUNCTION
Attribut (UML: *attribute*) • Interne Variable • Parameter: {input} • Eigenschaft: {property} • Ausgangsparameter: {output}	Variablentypen: • Variablen: VAR • Eingangsvariablen: VAR_INPUT • Eigenschaft: PROPERTY • Ausgangsvariablen: VAR_OUTPUT
Operation (UML: *operation*)	• Methode: METHOD • Aktion
Schnittstelle (UML: *interface*)	Schnittstelle: INTERFACE
	Globale Variablenliste (GVL): VAR_GLOBAL
	Benutzerdefinierter Datentyp (DUT): TYPE

Synchronität

Objekte im Klassendiagramm und im Projekt werden identisch gehalten, sodass Benutzereingaben auf beide Sichten wirken. Das bedeutet zum einen, dass bei der Änderung von Objekten über das Klassendiagramm die entsprechenden Objekte im Projektbaum automatisch mitgeändert werden. Zum anderen werden Änderungen im Projektbaum automatisch im Klassendiagramm sichtbar, sofern die entsprechenden Objekte im Klassendiagramm dargestellt werden.

Verwendungsmöglichkeiten

Generell gilt folgendes:

- Nicht alle Elemente eines Projekts müssen im Klassendiagramm dargestellt werden.
- Zu einem Projekt können mehrere Klassendiagramme hinzugefügt werden.

In der Regel ist es übersichtlicher, auf einem Klassendiagramm nur einige Objekte darzustellen. Auf diese Weise können Klassendiagramme beispielsweise themenbezogen oder pro Projektabschnitt angelegt werden. Die abgebildeten Objekte können zum einen in bestimmten Abhängigkeiten zueinander stehen, sodass die Objekte samt ihrer Abhängigkeiten übersichtlich dargestellt werden. Zum anderen können bei Bedarf auch Objekte ohne explizite Abhängigkeiten zueinander in einem Diagramm dargestellt werden, um sie anhand der parallelen Darstellung gegebenenfalls miteinander zu vergleichen.

Darauf aufbauend ergeben sich für das Klassendiagramm verschiedene Verwendungsmöglichkeiten:

- Als Design- und Entwicklungstool verwenden
- Als Analysewerkzeug eines bestehenden Projekts verwenden
- Als Projektnavigator verwenden

Nachfolgend finden Sie einige Verweise, welche Befehle bzw. Aktionen für die einzelnen Verwendungsmöglichkeiten genutzt werden können.

Als Design- und Entwicklungstool verwenden

- Legen Sie ein neues, leeres Klassendiagramm [▶ 19] an.
- Bearbeiten Sie das Klassendiagramm [▶ 22] mittels unterschiedlicher Aktionsmöglichkeiten.
  → Alle Eingaben wirken auch auf die Objekte im Projekt und sind im Projektbaum sofort automatisch sichtbar.

Als Analysewerkzeug eines bestehenden Projekts verwenden

- Fügen Sie bestehende Elemente zum Klassendiagramm hinzu [▶ 20].
- Analysieren Sie die bestehende Projektstruktur mithilfe des erzeugten Klassendiagramms und bearbeiten Sie das Diagramm [▶ 22] bei Bedarf.
  → Alle Eingaben wirken auch auf die Objekte im Projekt und sind im Projektbaum sofort automatisch sichtbar.

Als Projektnavigator verwenden

- Öffnen Sie ein Klassendiagramm und doppelklicken Sie auf ein Element im Klassendiagramm, um den dazugehörigen Editor zu öffnen.
- Bei Bedarf können Sie die Deklaration und Implementierung wie gewohnt bearbeiten.
  → Die Änderungen der Deklaration werden im Klassendiagramm automatisch aktualisiert.

Befehle

Für das Klassendiagramm stehen folgende Befehle bzw. Aktionsmöglichkeiten zur Verfügung:

- Neues Klassendiagramm anlegen [▶ 19]
- Bestehende Elemente zu einem Diagramm hinzufügen [▶ 20]
- Klassendiagramm bearbeiten [▶ 22]

Beachten Sie außerdem die Befehle, die für alle UML-Diagramme verfügbar sind: Gemeinsame Befehle aller UML-Diagramme [▶ 15].

Zugriffsmodifizierer

Verfügbar ab TwinCAT 3.1 Build 4026

Der Zugriffsmodifizierer einer Methode oder einer Eigenschaft wird im Klassendiagramm mithilfe eines Symbols angezeigt. Die folgende Tabelle zeigt, welches Symbol dabei für welchen Zugriffsmodifizierer steht.

Symbol	Zugriffsmodifizierer
+	PUBLIC
#	PROTECTED
-	PRIVATE
~	INTERNAL

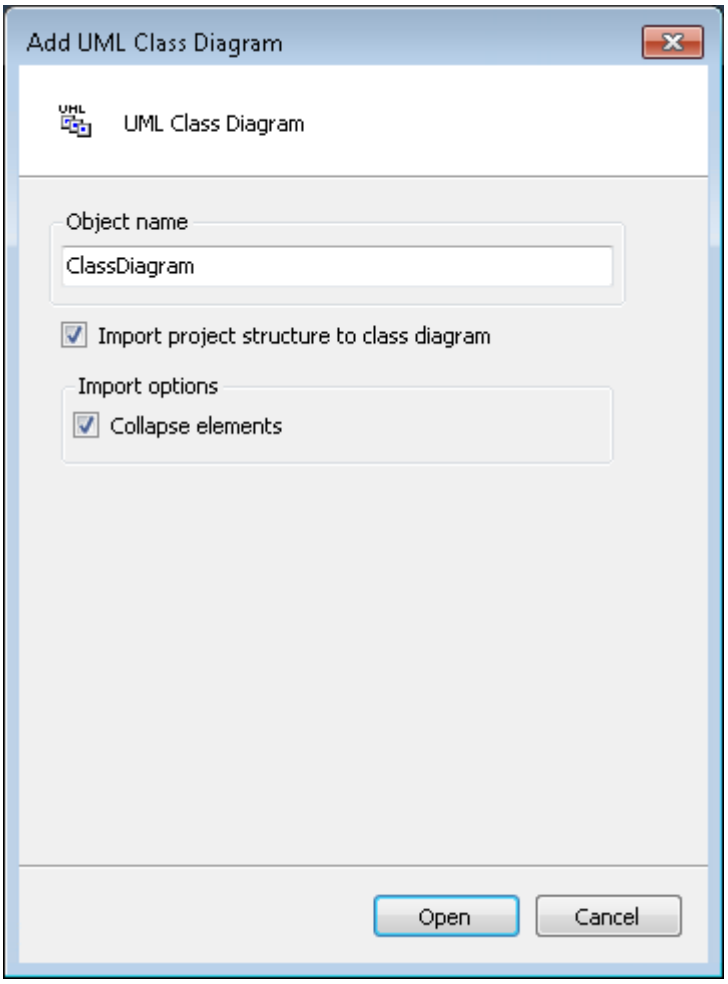
6.2 Befehle

6.2.1 Neues Klassendiagramm anlegen

Wenn Sie ein Klassendiagramm erzeugen, können Sie entweder ein leeres Klassendiagramm anlegen oder optional die bestehende Projektstruktur importieren. Bei Verwendung dieser Import-Option werden alle relevanten Objekte ins Klassendiagramm importiert. Relevante Objekte sind Programme, Funktionsbausteine, Funktionen, Schnittstellen, DUTs, GVLs und jeweils ihre Bestandteile (Methoden, Properties, Variablen etc.).

1. Wählen Sie im Kontextmenü des Projektbaums den Befehl **Objekt hinzufügen** und wählen Sie das Objekt **Klassendiagramm** aus.
2. Geben Sie im aufgehenden Dialog **UML Klassendiagramm hinzufügen** einen Namen für das Klassendiagramm an.
3. **Aktivieren** Sie die Option **Projektstruktur ins Klassendiagramm importieren** (Import project structure to class diagram), falls Sie die bestehende Projektstruktur in das neue Klassendiagramm importieren möchten.
   Deaktivieren Sie die Option, falls Sie ein leeres Klassendiagramm erzeugen möchten.
4. Importoptionen (nur relevant, falls Sie die Projektstruktur importieren):
   Aktivieren Sie die Option **Elemente reduzieren** (Collapse elements), falls die Details (Attribut- bzw. Operationsliste) der Elemente minimiert angezeigt werden sollen.

Deaktivieren Sie die Option, falls die Details der Elemente expandiert angezeigt werden sollen.



5. Bestätigen Sie die Eingaben und Konfigurationen über die **Öffnen**-Schaltfläche.

⇨ Im Projektbaum wird das neue Klassendiagramm-Objekt eingefügt und der Editor des neuen Diagramms wird geöffnet.

6.2.2 Bestehende Elemente zu einem Diagramm hinzufügen

Bestehende Projektelemente können auf verschiedene Weisen zu einem Klassendiagramm hinzugefügt werden. Zum einen können mehrere Elemente gleichzeitig hinzugefügt werden, indem über einen Befehl die gesamte Struktur des Projekts oder eines ausgewählten Ordners importiert wird. Zum anderen kann ein einzelnes Element hinzugefügt werden, indem es via Drag'n'Drop auf den Klassendiagrammeditor gezogen wird.

Je nach Elementanzahl stehen folgende Aktionsmöglichkeiten zur Verfügung.

- Mehrere bestehende Elemente hinzufügen:
  - Gesamte Struktur eines Projekts beim Anlegen eines neuen Diagramms importieren
  - Gesamte Struktur eines Projekts in ein leeres Diagramm importieren
  - Gesamte Struktur eines Ordners in ein leeres Diagramm importieren
- Einzelnes bestehendes Element hinzufügen:
  - Bestehendes Element aus dem Projektbaum auf dem Diagramm visualisieren
  - Bestehendes Element aus den Querverweisen auf dem Diagramm visualisieren

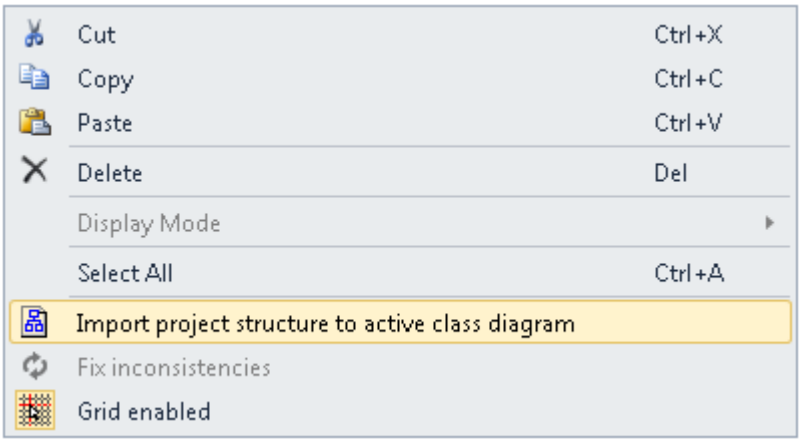
Gesamte Struktur eines Projekts beim Anlegen eines neuen Diagramms importieren

1. Legen Sie ein neues Klassendiagramm an [▶ 19] und aktivieren Sie dabei die Option **Projektstruktur ins Klassendiagramm importieren** (Import project structure to class diagram).

⇨ Im Projektbaum ist das neue Klassendiagramm-Objekt eingefügt. Der Klassendiagrammeditor wird geöffnet und zeigt das Klassendiagramm des bestehenden Projekts an.

Gesamte Struktur eines Projekts in ein leeres Diagramm importieren

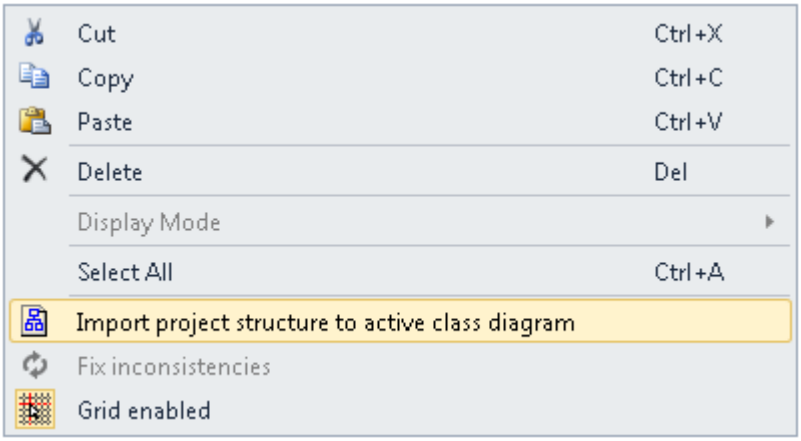
1. Öffnen Sie ein leeres Klassendiagramm, das im Projektbaum direkt auf oberster Projektebene und nicht in einem Projektordner liegt.
2. Führen Sie den Befehl **Projektstruktur in aktives Klassendiagramm importieren** (Import project structure to active class diagram) aus, der im Kontextmenü des Klassendiagrammeditors zur Verfügung steht.



⇨ Das Klassendiagramm zeigt die bestehende Struktur des gesamten Projekts an.

Gesamte Struktur eines Ordners in ein leeres Diagramm importieren

1. Öffnen Sie ein leeres Klassendiagramm, das in dem Ordner liegt, dessen Struktur Sie ins Klassendiagramm importieren möchten.
2. Führen Sie den Befehl **Projektstruktur in aktives Klassendiagramm importieren** (Import project structure to active class diagram) aus, der im Kontextmenü des Klassendiagrammeditors zur Verfügung steht.



⇨ Das Klassendiagramm zeigt die bestehende Struktur des Ordners an, in dem sich das Klassendiagramm befindet.

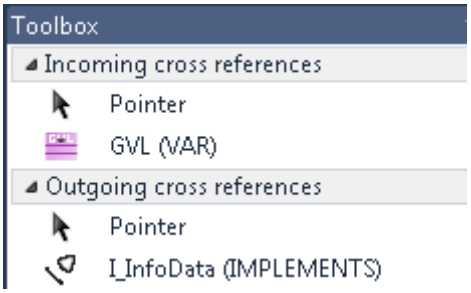
Bestehendes Element aus dem Projektbaum auf dem Diagramm visualisieren

1. Markieren Sie ein Element vom Typ POU, INTERFACE, GVL oder DUT im Projektbaum und ziehen Sie es per Drag'n'Drop auf das geöffnete Klassendiagramm. Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.

⇨ Das entsprechende Element wird auf dem Diagramm dargestellt. Falls Beziehungen zu bereits abgebildeten Elementen bestehen, werden diese automatisch angezeigt.

Bestehendes Element aus den Querverweisen auf dem Diagramm visualisieren

Unter der Überschrift **Eingehende Querverweise** oder **Ausgehende Querverweise** wird im Fenster **Werkzeuge** das Klassenelement angezeigt, das eine Beziehung zum selektierten Element hat, aber nicht im Klassendiagramm enthalten ist. Zuerst wird der Beziehungstyp, der die beiden Elemente verbindet, als Symbol angezeigt. Danach folgt der Name des Ziel- oder Quellelements. Sie können das Element via Drag-and-Drop auf das Diagramm ziehen, sodass das Element im Klassendiagramm dargestellt wird.



Querverweise anzeigen

1. Öffnen Sie das Fenster **Werkzeuge** über das Menü **Ansicht**.
2. Selektieren Sie ein Rechteckelement im geöffneten Klassendiagramm, das über Beziehungen verfügt, die nicht im Klassendiagramm dargestellt sind.

⇨ Unter **Werkzeuge** werden die Beziehungen zu den Elementen aufgelistet, die noch nicht im Klassendiagramm abgebildet sind. Unter **Eingehende Querverweise** werden fehlende eingehende Beziehungen mit Quellelementen aufgelistet. Unter **Ausgehende Querverweise** werden fehlende ausgehende Beziehungen mit Zielelementen aufgelistet.

Bestehendes Element aus den Querverweisen auf dem Diagramm visualisieren

1. Ziehen Sie das Element, das unter **Eingehende Querverweise** oder **Ausgehende Querverweise** aufgelistet ist und im Diagramm angezeigt werden soll, via Drag-and-Drop auf das Klassendiagramm. Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.

⇨ Das entsprechende Element wird auf dem Diagramm dargestellt. Die Beziehungen zu den bereits abgebildeten Elementen werden automatisch angezeigt.

6.2.3 Klassendiagramm bearbeiten

Um ein Klassendiagramm zu editieren stehen unter anderem die folgenden Aktionen zur Verfügung. Weitere Bearbeitungsmöglichkeiten finden Sie unter Editor [▶ 25].

Neues Element einfügen

1. Öffnen Sie das Fenster **Werkzeuge** über das Menü **Ansicht**.
2. Markieren Sie ein Element [▶ 25] in der Ansicht **Werkzeuge** und ziehen Sie es per Drag'n'Drop auf das geöffnete Klassendiagramm. Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.

⇨ Das neue Element wird im Projektbaum erzeugt und im Diagramm dargestellt.

Selektion der Ansicht "Werkzeuge" abwählen

✓ Im Fenster **Werkzeuge** ist ein Element selektiert. Im Editor hat der Mauscursor die Form des selektierten Elements.

1. Drücken Sie die rechte Maustaste.

⇨ Die Selektion des Elements wird abgewählt und das Standardelement **Zeiger** wird selektiert.

Bezeichner editieren

1. Öffnen Sie mit zwei Einzelklicks den Zeileneditor eines Bezeichners.

2. Geben Sie einen neuen Namen ein und bestätigen Sie die Eingabe über die [Enter]-Taste oder indem Sie in einen freien Bereich des Diagramms klicken.

⇨ Der Bezeichner trägt den neuen Namen.

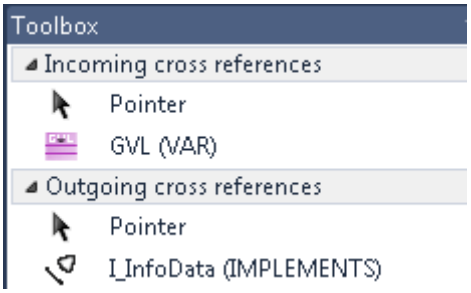
Bestehendes Element aus dem Projektbaum auf dem Diagramm visualisieren

1. Markieren Sie ein Element vom Typ POU, INTERFACE, GVL oder DUT im Projektbaum und ziehen Sie es per Drag'n'Drop auf das geöffnete Klassendiagramm. Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.

⇨ Das entsprechende Element wird auf dem Diagramm dargestellt. Falls Beziehungen zu bereits abgebildeten Elementen bestehen, werden diese automatisch angezeigt.

Querverweise

Unter der Überschrift **Eingehende Querverweise** oder **Ausgehende Querverweise** wird im Fenster **Werkzeuge** das Klassenelement angezeigt, das eine Beziehung zum selektierten Element hat, aber nicht im Klassendiagramm enthalten ist. Zuerst wird der Beziehungstyp, der die beiden Elemente verbindet, als Symbol angezeigt. Danach folgt der Name des Ziel- oder Quellelements. Sie können das Element via Drag-and-Drop auf das Diagramm ziehen, sodass das Element im Klassendiagramm dargestellt wird.



Querverweise anzeigen

1. Öffnen Sie das Fenster **Werkzeuge** über das Menü **Ansicht**.
2. Selektieren Sie ein Rechteckelement im geöffneten Klassendiagramm, das über Beziehungen verfügt, die nicht im Klassendiagramm dargestellt sind.

⇨ Unter **Werkzeuge** werden die Beziehungen zu den Elementen aufgelistet, die noch nicht im Klassendiagramm abgebildet sind. Unter **Eingehende Querverweise** werden fehlende eingehende Beziehungen mit Quellelementen aufgelistet. Unter **Ausgehende Querverweise** werden fehlende ausgehende Beziehungen mit Zielelementen aufgelistet.

Bestehendes Element aus den Querverweisen auf dem Diagramm visualisieren

1. Ziehen Sie das Element, das unter **Eingehende Querverweise** oder **Ausgehende Querverweise** aufgelistet ist und im Diagramm angezeigt werden soll, via Drag-and-Drop auf das Klassendiagramm. Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.

⇨ Das entsprechende Element wird auf dem Diagramm dargestellt. Die Beziehungen zu den bereits abgebildeten Elementen werden automatisch angezeigt.

Beziehung zwischen Diagrammelementen generieren

Möglichkeit 1 – via Icon:

1. Markieren Sie ein Element im geöffneten Klassendiagramm.
2. Klicken Sie auf ein Beziehungsicon, das über dem Element erscheint (das markierte Element agiert als Quellelement).

3. Klicken Sie auf das Zielelement der gewählten Beziehung.

⇨ Sie haben eine Beziehung zwischen dem Quell- und dem Zielelement erstellt, welche sowohl im Projekt aktiv ist als auch im Diagramm dargestellt wird.

Möglichkeit 2 – via **Werkzeuge**:

1. Klicken Sie auf ein Beziehungselement aus **Werkzeuge**.
2. Klicken Sie im geöffneten Klassendiagramm zunächst auf das Quellelement und anschließend auf das Zielelement.

⇨ Sie haben eine Beziehung zwischen dem Quell- und dem Zielelement erstellt, welche sowohl im Projekt aktiv ist als auch im Diagramm dargestellt wird.

Element aus Diagramm entfernen

Möglichkeit 1 – via Icon:

1. Markieren Sie das Element im Klassendiagramm, das Sie aus dem Diagramm entfernen möchten.
2. Verwenden Sie das Icon **Aus Diagramm entfernen**, das über dem Element erscheint.

⇨ Das ausgewählte Element wird aus dem Diagramm entfernt.

Möglichkeit 2 – via [Entf]-Taste:

1. Markieren Sie das Element im Klassendiagramm, das Sie aus dem Diagramm entfernen möchten.
2. Drücken Sie die Taste [Entf].
   Falls die Option [▶ 10] **Eingabeaufforderung, wenn Objekte aus dem Diagramm gelöscht werden** deaktiviert ist, wird das Objekt standardmäßig nur aus dem Diagramm gelöscht.
   Falls die Option aktiviert ist, wählen Sie im aufgehenden Dialog die Auswahlmöglichkeit **Aus Diagramm entfernen**.

⇨ Das ausgewählte Element wird aus dem Diagramm entfernt.

Element aus Diagramm und aus dem Projekt entfernen

Möglichkeit 1 – via Icon:

1. Markieren Sie das Element im Klassendiagramm, das Sie aus dem Diagramm und aus dem Projekt entfernen möchten.
2. Verwenden Sie das Icon **Aus Projekt und Diagramm entfernen**, das über dem Element erscheint.

⇨ Das ausgewählte Element wird aus dem Diagramm und aus dem Projekt entfernt.

Möglichkeit 2 – via [Entf]-Taste:

1. Aktivieren Sie die Option [▶ 10] **Eingabeaufforderung, wenn Objekte aus dem Diagramm gelöscht werden**.
2. Markieren Sie das Element im Klassendiagramm, das Sie aus dem Diagramm und aus dem Projekt entfernen möchten.
3. Drücken Sie die Taste [Entf] und wählen Sie im aufgehenden Dialog die Auswahlmöglichkeit **Aus Projekt und Diagramm entfernen**.

⇨ Das ausgewählte Element wird aus dem Diagramm und aus dem Projekt entfernt.

Als Projektnavigator verwenden

1. Öffnen Sie ein Klassendiagramm und doppelklicken Sie auf ein Element im Klassendiagramm.
   ⇨ Der zu dem Element gehörende Editor wird geöffnet.
2. Bei Bedarf können Sie die Deklaration und Implementierung innerhalb des geöffneten Editors wie gewohnt bearbeiten.
   ⇨ Die Änderungen der Deklaration werden im Klassendiagramm automatisch aktualisiert.

Mehrfachauswahl

- Wenn **Zeiger** in **Werkzeuge** aktiviert ist (Standard), können Sie im Klassendiagramm mit gedrückter linker Maustaste ein Rechteck über mehrere Elemente ziehen. Alle erfassten Elemente werden dann selektiert.
- Mehrfachauswahl ist auch möglich, indem Sie die gewünschten Elemente bei gedrückter [Strg]-Taste nacheinander via Mausklick auswählen.
- [Strg+A] bzw. **Alles selektieren** bewirkt, dass alle als Rechteck dargestellten Elemente, aber keine Beziehungselemente selektiert werden.

Refactoring

Folgende Änderungen, die im Klassendiagramm-Editor vorgenommen werden, können über Refactoring einfach auf das gesamte Projekt angewendet werden:

- Umbenennen von Variablen, Bausteinen oder Eigenschaften
- Hinzufügen und Entfernen von Variablen des Typs VAR_INPUT, VAR_OUTPUT oder VAR_IN_OUT

Standardmäßig ist die Refactoring-Funktionalität und die zugehörige Vorschau (Dialog **Refactoring**) im Klassendiagramm aktiviert. Dies kann jedoch über die folgenden Optionendialoge eingeschränkt werden:

- Dialog: Optionen – SPS Umgebung - Refactoring – UML Klassendiagramm
- Dialog: Optionen – SPS Umgebung - UML – Klassendiagramm [▶ 10]

Beachten Sie dabei vor allem, dass in den Optionen für das UML Klassendiagramm der Vorschaudialog übersprungen werden kann und in diesem Fall die Änderung ohne Nachfrage projektweit angewendet wird.

6.3 Editor

Ein UML-Klassendiagramm visualisiert den statischen Aufbau eines Projektes und gibt die Deklaration der Elemente grafisch wieder.

Um eine spezielle oder gefilterte Sicht auf das Projekt zu erhalten, kann entweder eine Auswahl an Objekten vom Projektbaum ins Klassendiagramm gezogen werden oder es können gezielt Elemente aus dem Diagramm entfernt werden.

Das Klassendiagramm kann mittels unterschiedlicher Aktionen editiert werden. Weiterführende Informationen zu den Bearbeitungsmöglichkeiten finden Sie unter Klassendiagramm bearbeiten [▶ 22].

Beachten Sie außerdem weitere, elementabhängige Benutzereingaben:

- Klasse editieren [▶ 27]
- Schnittstelle editieren [▶ 33]
- Generalisierung editieren [▶ 52]
- Realisierung editieren [▶ 50]
- Assoziation editieren [▶ 48]
- Komposition editieren [▶ 45]

6.4 Elemente

Das Fenster **Werkzeuge** stellt die Elemente des Klassendiagramms zur Verfügung. Sie können via Drag'n'Drop in das Klassendiagramm-Fenster eingefügt werden. Die Elemente können beliebig platziert werden.

Element einfügen

1. Führen Sie **Ansicht > Werkzeuge** aus, um über die Elemente zu verfügen.
2. Ziehen Sie ein Element in das Klassendiagramm-Fenster und lassen Sie es an geeigneter Stelle fallen.

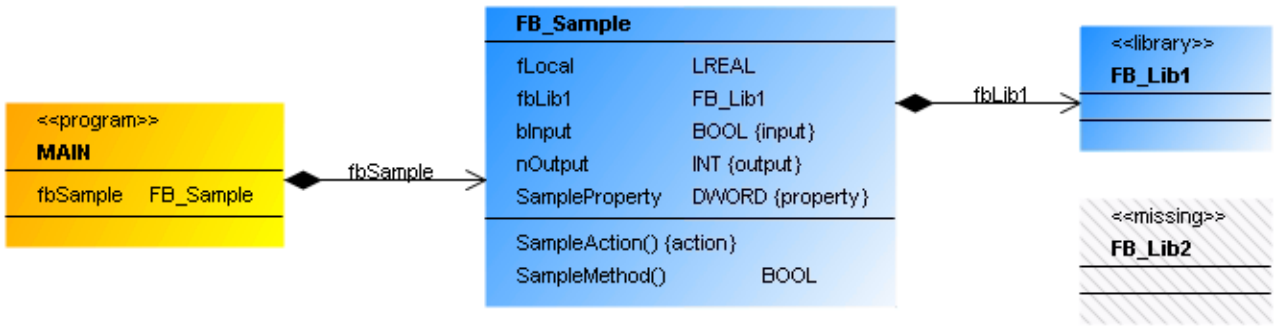
	Klasse [▶ 26] (POU)
	Schnittstelle [▶ 32]
	Globale Variablenliste [▶ 35] (GVL)
	Datentyp [▶ 38] (DUT)
	Variablendeklaration [▶ 41]
	Eigenschaft [▶ 41]
	Methode [▶ 42]
	Aktion [▶ 43]
	Komposition [▶ 43] (VAR)
	Assoziation [▶ 47] (POINTER) oder Assoziation [▶ 47] (REFERENCE)
	Realisierungsbeziehung [▶ 49] (IMPLEMENTS)
	Generalisierung [▶ 51] (EXTENDS)
	Notiz [▶ 53]

6.4.1 Klasse

Eine Klasse ist eine logische Einheit, in der Daten und Operationen gekapselt werden. Sie stellt außerdem einen Variablentyp dar, der instanziiert werden kann. Eine Klasse kann eine Methode FB_Init haben, die bei der Initialisierung einer Instanz aufgerufen wird.

Eine Klasse kann über folgende Beziehungstypen verfügen:

- Komposition: Eine Klasse kann andere Programmelemente enthalten.
- Assoziation: Eine Klasse kann andere Programmelemente kennen.
- Realisierung: Eine Klasse kann eine Schnittstelle einbinden.
- Generalisierung: Eine Klasse kann von einer anderen Klasse erben.



Eine Klasse wird durch ein dreigeteiltes Rechteck dargestellt:

- Orange mit Überschrift <<program>>: POU (PROGRAM)
- Blau ohne Überschrift: POU (FUNCTION_BLOCK oder FUNCTION)
- Blau mit Überschrift <<library>>: POU (FUNCTION_BLOCK oder FUNCTION) aus einer Bibliothek (Bibliotheksfunktionsbaustein)
- Gegraut mit Überschrift <<fehlt>>: POU (FUNCTION_BLOCK oder FUNCTION) aus einer Bibliothek, die aber aktuell nicht im Projekt eingebunden ist

Es folgt fettgedruckt der Klassenname.

Nach der ersten Trennlinie werden alle Attribute angezeigt. Jedes sichtbare Attribut hat eine Kennung. Wenn in geschweiften Klammern {input} angefügt ist, handelt es sich um eine Variable vom Typ VAR_INPUT, bei {output} handelt es sich um eine Variable vom Typ VAR_OUTPUT. Eine Eigenschaft hat die Kennung {property}. Eine interne (nicht sichtbare) Variable vom Typ VAR hat keine Kennung:

<Attributname> : <Datentyp> {'{input}' | '{output}' | '{property}'}

Nach der zweiten Trennlinie folgen alle Operationen der Klasse, d.h. ihre Methoden oder Aktionen. Zuerst kommt jeweils der Name der Operation mit abschließender runder Klammer. Anschließend kennzeichnet {action} eine Aktion:

<Aktionsname>() {action}

Handelt es sich um eine Methode, wird in den runden Klammern ggf. auf eine Variablenübergabe hingewiesen. Falls für eine Methode ein Rückgabetyp deklariert ist, folgt dieser in der rechten Spalte. Methoden besitzen im Gegensatz zu Aktionen ({action}) keine abschließende Kennung:

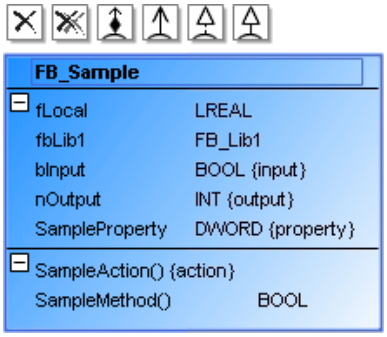
<Methodenname>(…) : <Rückgabetyp>

Eigenschaften

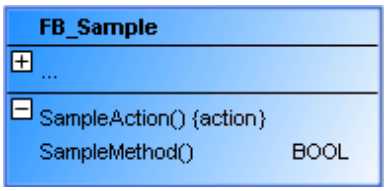
„Eigenschaft“	Beschreibung
„Bezeichner“	Name der Klasse Beispiel: FB_EventLogger

Benutzereingaben bei einer Klasse

Die folgenden Benutzereingaben sind unter der Bedingung verfügbar, dass „Zeiger“ in der Toolbox aktiviert ist (Default).

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Klasse (POU)“:  Klicken Sie in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen“ öffnet. Geben Sie für das neue Objekt einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Klasse wird erstellt.	Das Objekt existiert im Diagramm und im Projekt. Die Ansicht im Projektbaum wird automatisch aktualisiert.
Selektieren Sie eine Klasse.		Die Klasse hat expandierte Attribut- und Operationslisten, die mit ⊟ gekennzeichnet sind. Links oberhalb der Klasse sind nun Befehlssymbole sichtbar, um Beziehungselemente einzufügen. Tipp: Wenn die Klasse über Beziehungen verfügt, die zur Zeit nicht im Klassendiagramm dargestellt werden (fehlende Rechteckelemente), dann erscheint in der Ansicht „Werkzeuge“ eine Liste „Eingehende Querverweise“ und/oder „Ausgehende Querverweise“. Darunter sind die fehlenden Rechteckelemente aufgelistet, welche Sie per Drag’n’Drop ins Klassendiagramm ziehen können, um sie dort darzustellen.
Klicken Sie auf  .	Die Klasse wird im Diagramm entfernt, sodass sie nicht mehr dargestellt wird.	Verwenden Sie das flache Entfernen einer Klasse, wenn sie nur in der Klassendiagrammsicht entfernt werden soll. Das Objekt existiert nach wie vor und ist im Projektbaum sichtbar. Tipp: Die Klasse wird in der Ansicht „Werkzeuge“ unter „Eingehende Querverweise“ und/oder „Ausgehende Querverweise“ angezeigt, wenn ein Rechteckelement im Klassendiagramm selektiert ist, das eine Beziehung zu dem entfernten Element hat.
Klicken Sie auf  .	Die Klasse wird im Diagramm und im Projekt entfernt.	Das Objekt erscheint nicht mehr als Objekt im Klassendiagramm oder im Projektbaum. Es existiert nicht mehr.

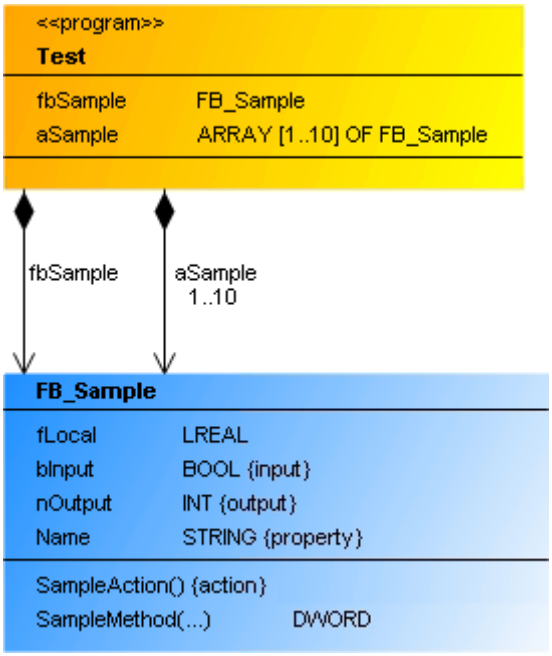
Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Klicken Sie auf  und dann in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen “ öffnet. Geben Sie für das neue Objekt einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Ein Kompositionspfeil weist ausgehend von der bestehenden Klasse auf die neue Klasse.	Die bestehende Klasse enthält eine Instanz auf die neue Klasse. Zum Beispiel: fbNew : FB_New;
Klicken Sie auf  und dann auf eine bestehende Klasse.	Ein Kompositionspfeil weist ausgehend von der ersten Klasse auf die zweite Klasse.	Die erste Klasse enthält eine Instanz auf die zweite Klasse. Zum Beispiel: fbExistent : FB_Existent;
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen“ öffnet. Geben Sie einen Namen für das neue Objekt ein, passen Sie die Einstellungen an und beenden Sie dann den Dialog mit „Hinzufügen“.	Ein Assoziationspfeil weist ausgehend von der ersten Klasse auf die neue Klasse.	Eine neue Klasse FB_New wird erzeugt. Die bestehende Klasse kennt nun die neue Klasse und enthält eine Assoziation auf die neue Klasse. Zum Beispiel: pNew : POINTER TO FB_New .
Klicken Sie auf  und anschließend auf eine bestehende Klasse.	Ein Assoziationspfeil weist ausgehend von der ersten Klasse auf die zweite Klasse.	Die erste Klasse kennt die zweite Klasse. Sie enthält nun eine neue Variable vom Typ. Zum Beispiel: pExistent : POINTER TO FB_Existent;
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen“ öffnet. Geben Sie einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Generalisierung weist von der bestehenden Klasse zur neuen Klasse. Die bestehende Klasse erbt von der neuen.	Die bestehende Klasse enthält die Deklaration. Zum Beispiel: FUNCTION_BLOCK FB_Sub EXTENDS FB_New
Klicken Sie auf  und anschließend auf eine bestehende Klasse.	Eine Generalisierung weist von der ersten Klasse zur zweiten Klasse.	Die erste Klasse enthält die Deklaration. Zum Beispiel: FUNCTION_BLOCK FB_Sub EXTENDS FB_Existent
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Der Dialog „Schnittstelle hinzufügen“ öffnet. Geben Sie einen Schnittstellennamen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Ein Realisierungspfeil weist von der Klasse zur neuen Schnittstelle.	Die Klasse implementiert die neue Schnittstelle. Im Deklarationsteil der Klasse steht. Zum Beispiel: FUNCTION_BLOCK FB_Sample IMPLEMENTS I_SampleNew

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Klicken Sie auf  und anschließend auf eine Schnittstelle.	Ein Realisierungspfeil weist von der Klasse zur Schnittstelle.	Die Klasse implementiert nun die Schnittstelle. Im Deklarationsteil der Klasse steht. Zum Beispiel: FUNCTION_BLOCK FB_Sample IMPLEMENTS I_SampleExistent
Klicken Sie auf ⊟.		Die Attribut- bzw. Operationsliste minimiert sich.
Klicken Sie auf ⊞.		Die Attribut- bzw. Operationsliste expandiert.
Klicken Sie auf den Klassennamen. Ist er selektiert, dann ein weiteres Mal auf ihn klicken.	Nach dem ersten Klick ist der Name blau umrandet. Nach dem zweiten Klick öffnet der Zeileneditor.	Die Änderung wird im Projekt synchron und automatisch übernommen. D.h., der Objektname im Projektbaum und im Deklarationsteil der POU wird sofort angepasst.
Klicken Sie zwei Mal auf einen Attribut- oder Operationsnamen. Ändern Sie dann den Namen im Zeileneditor.	Nach dem ersten Klick ist der Name blau umrandet. Nach dem zweiten Klick öffnet der Zeileneditor.	Die Änderung wird im Projekt synchron und automatisch übernommen.
Doppelklicken Sie das Klassenelement.	Der zugehörige Objekteditor öffnet sich und Deklaration und Implementation werden angezeigt.	Sie können die Deklaration bzw. die Implementierung ändern. Nach Schließen des Objekts kehren Sie zum Klassendiagramm zurück. Die Änderungen werden automatisch ins Klassendiagramm übernommen.

Beachten Sie, dass die Voreinstellungen im Dialog „POU hinzufügen“ oder „Schnittstelle hinzufügen“ von der letzten Anwendung dieses Dialogs stammen.

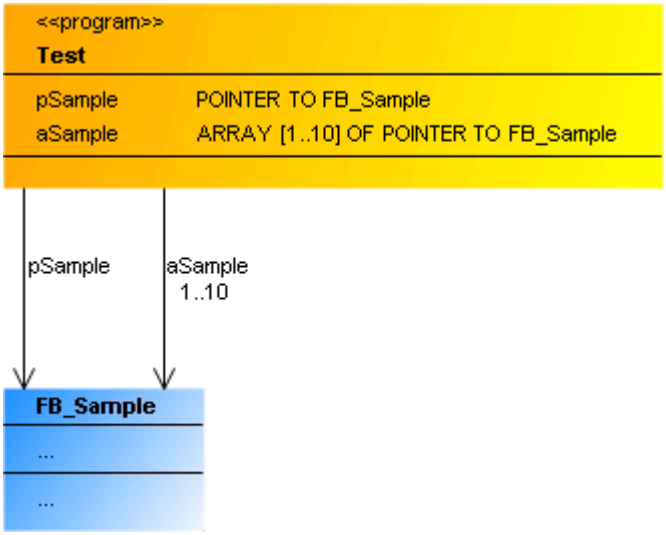
Beispiele

- Komposition



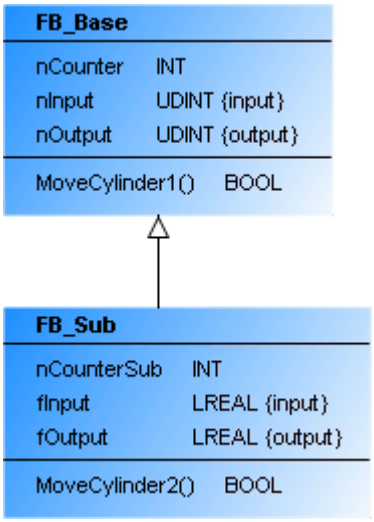

```
PROGRAM Test
VAR
    fbSample    : FB_Sample;
    aSample     : ARRAY[1..10] OF FB_Sample;
END_VAR
```

- Assoziation



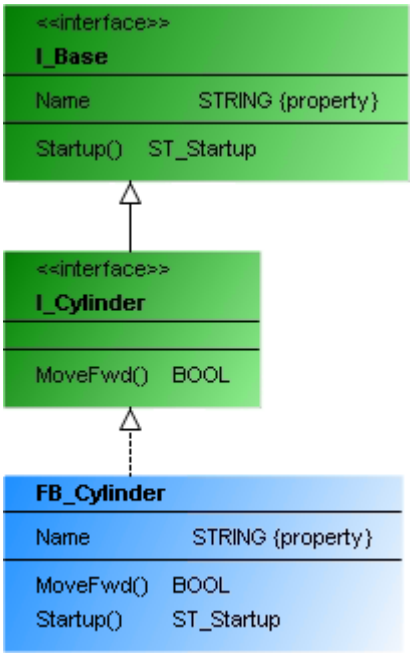

```
PROGRAM Test
VAR
    pSample    : POINTER TO FB_Sample;
    aSample    : ARRAY[1..10] OF POINTER TO FB_Sample;
END_VAR
```

- Generalisierung




```
FUNCTION_BLOCK FB_Sub EXTENDS FB_Base
```

- Realisierung



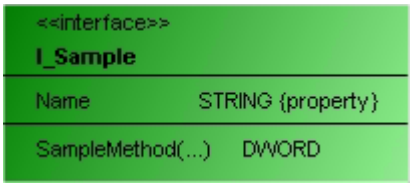

```
INTERFACE I_Cylinder EXTENDS I_Base
FUNCTION_BLOCK FB_Cylinder IMPLEMENTS I_Cylinder
```

6.4.2 Schnittstelle

Eine Schnittstelle definiert Methoden- und Eigenschaftsdeklarationen, die ein extern sichtbares Verhalten beschreiben. Sie enthält keine Variablen und keine Implementierung, sondern lediglich die Definition von Methoden und/oder Eigenschaften. Eine Schnittstelle stellt einen Variablentyp dar, der instanziiert werden kann.

Eine Schnittstelle kann über den folgenden Beziehungstyp verfügen:

- Generalisierung: Eine Schnittstelle kann von einer anderen Schnittstelle erben.



Schnittstellen werden wie Klassen dreigeteilt dargestellt. Das Rechteck ist grün und mit <<interface>> überschrieben. Darunter folgt fettgedruckt der Schnittstellenname. Nach der ersten Trennlinie werden die Eigenschaften der Schnittstelle nach folgender Syntax angezeigt:
<Name der Eigenschaft> : <Datentyp> {property}

Nach der zweiten Trennlinie folgen alle Methoden der Schnittstelle. Nach dem Methodennamen wird in runden Klammern ggf. auf eine Variablenübergabe hingewiesen. Falls für eine Methode ein Rückgabetyp deklariert ist, folgt dieser in der rechten Spalte.

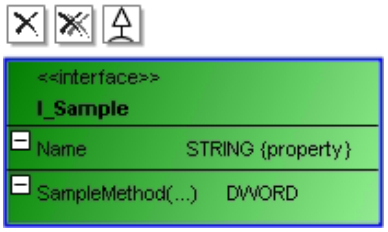
<Methodenname>(…) : <Rückgabetyp>

Eigenschaften

„Eigenschaft“	Beschreibung
„Bezeichner“	Geben Sie hier einen eindeutigen Namen für das selektierte Element ein. Sie können den Namen auch im Klassendiagramm ändern, indem Sie den Namen erst selektieren und mit einem weiteren Klick den Zeileneditor öffnen.

Schnittstelle editieren

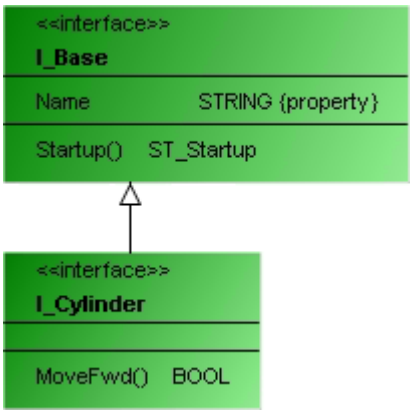
Die folgenden Benutzereingaben sind unter der Bedingung verfügbar, dass „Auswahl“ oder „Selektion“ in „Werkzeuge“ aktiviert ist (Default).

Benutzereingabe im Klassendiagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Interface“:  Klicken Sie in einen freien Bereich des Diagramms. Der Dialog „Schnittstelle hinzufügen“ öffnet. Geben Sie für das neue Objekt einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Schnittstelle wird erstellt.	Das Objekt existiert im Diagramm und im Projekt. Die Ansicht im Projektbaum wird automatisch aktualisiert.
Klicken Sie auf ein Schnittstellensymbol.		Links oberhalb der Schnittstelle sind nun Symbole sichtbar, die es ermöglichen, Beziehungselemente einzufügen. Das Beispiel links hat expandierte Property- und Methodenlisten nach ⊟.
Klicken Sie auf  .	Die Schnittstelle wird nur im Diagramm entfernt.	Verwenden sie das flache Entfernen einer Schnittstelle, wenn sie nur in der Klassendiagrammsicht entfernt werden soll. Das Objekt existiert nach wie vor und ist im Projektbaum sichtbar.
Klicken Sie auf  .	Die Schnittstelle wird im Diagramm und im Projekt entfernt.	Das Objekt wird entfernt. Danach existiert es nicht mehr.
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Der Dialog „Schnittstelle hinzufügen“ öffnet. Geben Sie einen Namen ein und beenden Sie den Dialog mit „Hinzufügen“.	Eine Generalisierung weist ausgehend von der bestehenden Schnittstelle auf die neue Schnittstelle. Die bestehende Schnittstelle erbt von der neuen.	Die bestehende Schnittstelle enthält die Deklaration. Zum Beispiel: INTERFACE I_Sample EXTENDS I_New Beachten Sie, dass die Voreinstellungen im Dialog „Schnittstelle hinzufügen“ von der letzten Anwendung dieses Dialogs stammen.
Klicken Sie auf  und anschließend auf eine bestehende Schnittstelle.	Eine Generalisierung weist von der ersten Schnittstelle zur zweiten Schnittstelle.	Die erste Schnittstelle enthält die Deklaration. Zum Beispiel: INTERFACE I_Sample EXTENDS I_Existent
Klicken Sie auf ⊟.		Die Eigenschaften- bzw. Methodenliste minimiert sich.
Klicken Sie auf ⊞.		Die Eigenschaften- bzw. Methodenliste wird expandiert.

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Klicken Sie auf den Namen. Ist er selektiert, dann ein weiteres Mal auf ihn klicken.	Nach dem ersten Klick ist der Name blau umrandet. Nach dem zweiten Klick öffnet der Zeileneditor.	Ändern Sie den Schnittstellennamen im Zeileneditor. Die Änderung wird im Projekt synchron und automatisch übernommen. Das heißt, der Objektname im Projektbaum und im Deklarationsteil der POU wird sofort angepasst.
Doppelklicken Sie auf eine Schnittstelle.	Der zugehörige Objekteditor öffnet mit dem Deklarationseditor.	Editieren Sie die Deklaration. Nach Schließen des Objekts kehren Sie zum Klassendiagramm zurück. Die Änderungen werden automatisch ins Klassendiagramm übernommen.

Beispiel

- Generalisierung



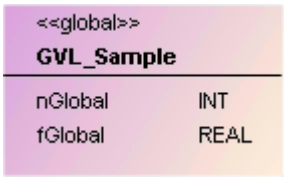

```
INTERFACE I_Cylinder EXTENDS I_Base
```

6.4.3 Globale Variablenliste

Eine Globale Variablenliste (GVL) wird verwendet, um globale Variablen zu deklarieren. Diese stehen projektweit zur Verfügung.

Eine GVL kann über folgende Beziehungstypen verfügen:

- Komposition: Eine Klasse kann andere Programmelemente enthalten.
- Assoziation: Eine Klasse kann andere Programmelemente kennen.



Eine Globale Variablenliste wird durch ein zweigeteiltes Rechteck in blassrosa dargestellt und ist mit <<global>> überschrieben. Nach der ersten Trennlinie werden alle Attribute angezeigt:

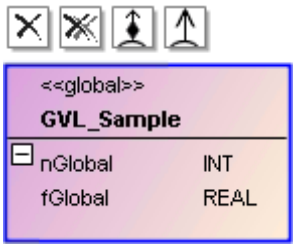
<Attributname> : <Datentyp>

Eigenschaften

„Eigenschaft“	Beschreibung
„Bezeichner“	Der Namen des Elements wird angezeigt. Er ist editierbar.

GVL editieren

Die folgenden Benutzereingaben sind unter der Bedingung verfügbar, dass „Selektion“ in der Toolbox aktiviert ist (Default).

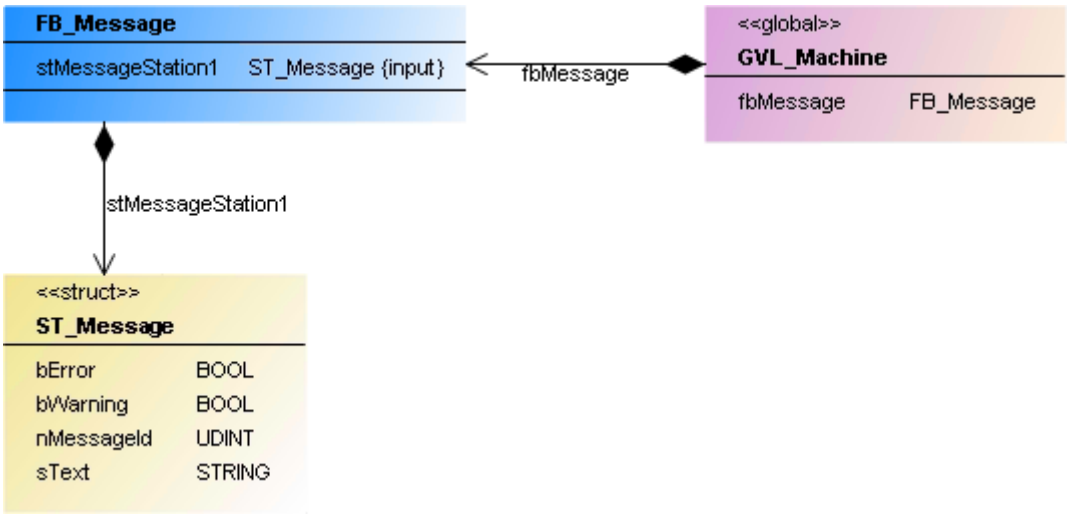
Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Globale Variablenliste (GVL)“:  Klicken Sie in einen freien Bereich des Diagramms. Der Dialog „Globale Variablenliste hinzufügen“ öffnet. Geben Sie für das neue Objekt einen Namen ein und beenden Sie den Dialog mit „Hinzufügen“.	Eine GVL wird erstellt.	Das Objekt existiert im Diagramm und im Projekt. Die Ansicht im Projektbaum wird automatisch aktualisiert.
Klicken Sie auf ein GVL-Objekt.		Links oberhalb sind Befehlssymbole sichtbar.
Klicken Sie auf  .	GVL wird nur im Diagramm entfernt.	Verwenden Sie das flache Entfernen, wenn nur in der Klassendiagrammsicht entfernt werden soll. Das Objekt existiert nach wie vor und ist im Projektbaum sichtbar.
Klicken Sie auf  .	GVL wird im Diagramm und im Projekt entfernt.	Das Objekt wird entfernt. Danach existiert es nicht mehr.
Klicken Sie auf  und dann auf eine existierende Klasse oder eine DUT.	Eine Komposition weist von der GVL auf die selektierte Klasse oder DUT.	Die Deklaration von GVL enthält die Instanziierung auf das selektierte Element. Zum Beispiel: fbExistent : FB_Existent;
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Dialog „POU hinzufügen“ öffnet. Geben Sie einen Namen ein und beenden Sie den Dialog mit „Hinzufügen“.	Eine Komposition weist von der GVL auf die neue Klasse.	Die GVL enthält die Deklaration. Zum Beispiel: fbNew : FB_New;
Klicken Sie auf  und dann auf eine existierende Klasse bzw. DUT.	Eine Assoziation weist von der GVL auf die selektierte Klasse bzw. DUT.	Die GVL enthält die Deklaration auf das selektierte Element. Zum Beispiel: pExistent : POINTER TO FB_Existent;
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen“ öffnet. Geben Sie einen Namen ein und beenden Sie den Dialog mit „Hinzufügen“.	Eine Assoziation weist vom GVL auf die neue Klasse.	Die GVL enthält die Deklaration auf die neue Klasse. Zum Beispiel: pNew : POINTER TO FB_New;.
Klicken Sie auf ⊟ .		Die Attribut- beziehungsweise Operationsliste minimiert sich.

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Klicken Sie auf ⊞.		Die Attribut- beziehungsweise Operationsliste expandiert.
Klicken Sie auf einen Bezeichner. Ist er selektiert, klicken Sie erneut.	Nach dem ersten Klick ist der Name blau umrandet. Nach dem zweiten Klick öffnet der Zeileneditor.	Ändern Sie den Klassennamen im Zeileneditor. Die Änderung wird im Projekt synchron übernommen. Das heißt, der Objektname im Projektbaum und im Deklarationsteil der POU wird sofort angepasst.
Doppelklicken Sie ein Element.	Der zugehörige Objekteditor öffnet und Deklaration und Implementation werden angezeigt.	Editieren Sie die Deklaration oder Implementierung. Nach Schließen des Objekts kehren Sie zum Klassendiagramm zurück. Die Änderungen werden automatisch ins Klassendiagramm übernommen.

Beachten Sie, dass die Voreinstellungen im Dialog „POU hinzufügen“ von der letzten Anwendung dieses Dialogs stammen.

Beispiel

- Komposition



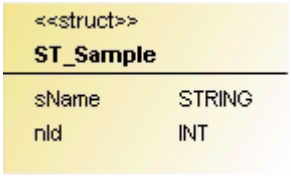

```
VAR_GLOBAL
    fbMessage    : FB_Message;
END_VAR
```

6.4.4 Benutzerdefinierter Datentyp

Über einen benutzerdefinierten Datentyp („Data Unit Type“ = DUT) kann der Anwender eigene Datentypen, z.B. in Form von Strukturen oder Aufzählungen, definieren.

Ein benutzerdefinierter Datentyp als Struktur kann über den folgenden Beziehungstyp verfügen:

- Generalisierung: Eine Struktur kann von einer anderen Struktur erben.



Ein benutzerdefinierter Datentyp wird durch ein zweigeteiltes Rechteck in blassgelb dargestellt und ist mit <<struct>> überschrieben, wenn es sich um eine Struktur oder eine Union handelt. <<enum>> weist auf einen Aufzählungstyp hin. Es folgt fettgedruckt der Bezeichner. Nach der ersten Trennlinie werden alle Attribute angezeigt:

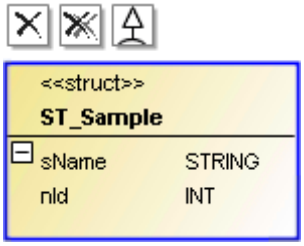
<Attributname> : <Datentyp>

Eigenschaften

„Eigenschaft“	Beschreibung
„Bezeichner“	Eindeutigen Namen eingeben oder ändern des selektierten Elements.

DUT editieren

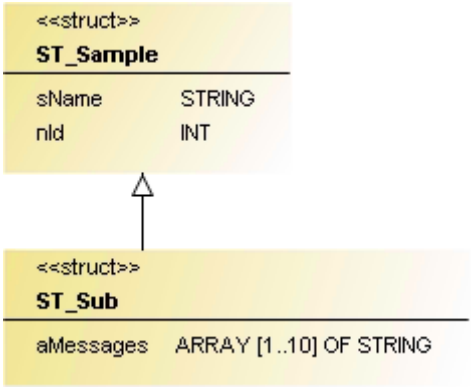
Die folgenden Benutzereingaben sind unter der Bedingung verfügbar, dass „Selektion“ in der Toolbox aktiviert ist (Default).

Benutzereingabe im Klassendiagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Data Unit Type (DUT)“:  Klicken Sie in einen freien Bereich des Diagramms. Der Dialog „DUT hinzufügen“ öffnet. Geben Sie für das neue Objekt einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Ein Datentypobjekt wird erstellt.	Das Objekt existiert im Diagramm und im Projekt. Die Ansicht im Projektbaum wird automatisch aktualisiert.
Klicken Sie auf eine DUT.		Links oberhalb sind Icons sichtbar.
Klicken Sie auf  .	DUT wird nur im Diagramm entfernt.	Verwenden Sie das flache Entfernen, wenn nur in der Klassendiagrammsicht entfernt werden soll. Das Objekt existiert nach wie vor und ist im Projektbaum sichtbar.
Klicken Sie auf  .	DUT wird im Diagramm und im Projekt entfernt.	Das Objekt wird entfernt. Danach existiert es nicht mehr.
Klicken Sie auf  und anschließend in einen freien Bereich des Diagramms. Dialog „DUT hinzufügen“ öffnet. Geben Sie einen Namen für das Vaterobjekt ein und beenden Sie den Dialog mit „Hinzufügen“.	Eine Generalisierung weist von der bestehenden DUT zur neuen DUT. Die bestehende DUT erbt von der neuen.	Die bestehende DUT enthält die Deklaration. Zum Beispiel: TYPE ST_Sample EXTENDS ST_New
Klicken Sie auf  und anschließend auf eine bestehende DUT.	Ein Generalisierungspfeil weist von der ersten DUT zur zweiten DUT.	Die zweite DUT erbt an die erste DUT. Die erste DUT enthält die Deklaration. Zum Beispiel: TYPE ST_Sample EXTENDS ST_Existent
Klicken Sie auf ⊟ .		Die Attribut- bzw. Operationsliste minimiert sich.
Klicken Sie auf ⊞ .		Die Attribut- bzw. Operationsliste expandiert.
Klicken Sie zweimal auf den Bezeichner. Ändern Sie dann den Namen im Zeileneditor.	Nach dem ersten Klick ist der Name blau umrandet. Nach dem zweiten Klick öffnet der Zeileneditor.	Die Änderung wird im Projekt synchron und automatisch übernommen.
Doppelklicken Sie das Element.	Der zugehörige Objekteditor öffnet.	Editieren Sie die Deklaration oder Implementierung. Nach Schließen des Editors kehren Sie zum Klassendiagramm zurück. Die Änderungen werden automatisch ins Klassendiagramm übernommen.

Die Voreinstellungen im Dialog „DUT hinzufügen“ stammen von der letzten Anwendung dieses Dialogs.

Beispiel

- Generalisierung




```
TYPE ST_Sub EXTENDS ST_Sample :
STRUCT
    aMessages    : ARRAY[1..10] OF STRING;
END_STRUCT
END_TYPE
```

6.4.5 Variablendeklaration

Das Element „Variablendeklaration“ wird verwendet, um im Klassendiagramm eine Klasse (Programm, Funktionsbaustein, Funktion) oder eine Globale Variablenliste (GVL) um eine Variable zu erweitern.

Variable hinzufügen

Wählen Sie das Werkzeug „Variablendeklaration“: 

Anschließend können Sie im Diagramm das Element selektieren, das um eine Variable erweitert werden soll. An nicht sinnvollen Stellen hat der Cursor die Form eines Verbotsschildes. An sinnvollen Stellen ist der Cursor ein blaues bzw. schwarzes Kreuz. Wenn Sie an eine solche Stelle klicken, wird im fokussierten Element eine weitere Variable hinzugefügt. Der Dialog „Variable deklarieren“ öffnet. Geben Sie wie gewohnt die gewünschten Einstellungen ein. Mit Schließen des Dialogs wird das Programmelement um diese Variable erweitert, was sowohl im Klassendiagramm als auch im Objekteditor sofort aktualisiert wird.

Cursorform

-  : An diesen Stellen im Diagramm ist Erweitern nicht möglich.
-  : An diesen Stellen im Diagramm können Sie das fokussierte Element um eine Variable erweitern.

6.4.6 Eigenschaft

Das Element „Eigenschaft“ wird verwendet, um eine Klasse (Programm oder Funktionsbaustein) oder eine Schnittstelle um eine Eigenschaft zu erweitern.

Eigenschaft hinzufügen

Wählen Sie das Werkzeug „Eigenschaft“: 

Anschließend können Sie im Diagramm das Element selektieren, das um eine Eigenschaft erweitert werden soll. An nicht sinnvollen Stellen hat der Cursor die Form eines Verbotsschildes. An sinnvollen Stellen ist der Cursor ein blaues bzw. schwarzes Kreuz. Wenn Sie an eine solche Stelle klicken, wird dem fokussierten Element eine weitere Eigenschaft hinzugefügt. Der Dialog „Eigenschaft hinzufügen“ öffnet. Geben Sie wie gewohnt die gewünschten Einstellungen ein. Mit Schließen des Dialogs wird das Programmelement um diese Eigenschaft erweitert, was sowohl im Klassendiagramm als auch im Projektbaum sofort aktualisiert wird.

Cursorform:

-  : An diesen Stellen im Diagramm ist Erweitern nicht möglich.
-  : An diesen Stellen im Diagramm können Sie das fokussierte Element um eine Eigenschaft erweitern.

Zugriffsmodifizierer

Verfügbar ab TwinCAT 3.1 Build 4026

Der Zugriffsmodifizierer einer Methode oder einer Eigenschaft wird im Klassendiagramm mithilfe eines Symbols angezeigt. Die folgende Tabelle zeigt, welches Symbol dabei für welchen Zugriffsmodifizierer steht.

Symbol	Zugriffsmodifizierer
+	PUBLIC
#	PROTECTED
-	PRIVATE
~	INTERNAL

6.4.7 Methode

Das Element „Methode“ wird verwendet, um eine Klasse (Programm oder Funktionsbaustein) oder eine Schnittstelle um eine Methode zu erweitern.

Methode hinzufügen

Wählen Sie das Werkzeug „Methode“: 

Anschließend können Sie im Diagramm das Element selektieren, das um eine Methode erweitert werden soll. An nicht sinnvollen Stellen hat der Cursor die Form eines Verbotsschildes. An sinnvollen Stellen ist der Cursor ein blaues bzw. schwarzes Kreuz. Wenn Sie an eine solche Stelle klicken, wird dem fokussierten Element eine weitere Methode hinzugefügt. Der Dialog „Methode hinzufügen“ öffnet. Geben Sie wie gewohnt die gewünschten Einstellungen ein. Mit Schließen des Dialogs wird das Programmelement um diese Methode erweitert, was sowohl im Klassendiagramm als auch im Projektbaum sofort aktualisiert wird.

Cursor

-  : An diesen Stellen im Diagramm ist Erweitern nicht möglich.

-  : An diesen Stellen im Diagramm können Sie das fokussierte Element um eine Methode erweitern.

Zugriffsmodifizierer

Verfügbar ab TwinCAT 3.1 Build 4026

Der Zugriffsmodifizierer einer Methode oder einer Eigenschaft wird im Klassendiagramm mithilfe eines Symbols angezeigt. Die folgende Tabelle zeigt, welches Symbol dabei für welchen Zugriffsmodifizierer steht.

Symbol	Zugriffsmodifizierer
+	PUBLIC
#	PROTECTED
-	PRIVATE
~	INTERNAL

6.4.8 Aktion

Das Element „Aktion“ wird verwendet, um ein Programm oder einen Funktionsbaustein um eine Aktion zu erweitern.

Aktion hinzufügen

Wählen Sie das Werkzeug „Aktion“: 

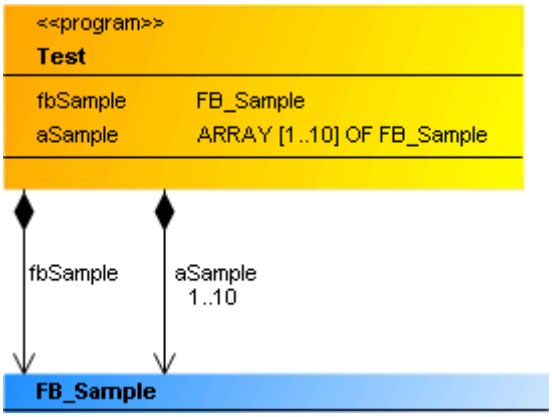
Anschließend können Sie im Diagramm das Element selektieren, das um eine Aktion erweitert werden soll. An nicht sinnvollen Stellen hat der Cursor die Form eines Verbotsschildes. An sinnvollen Stellen ist der Cursor ein blaues bzw. schwarzes Kreuz. Wenn Sie an eine solche Stelle klicken, wird dem fokussierten Element eine weitere Aktion hinzugefügt. Der Dialog „Aktion hinzufügen“ öffnet. Geben Sie wie gewohnt die gewünschten Einstellungen ein. Mit Schließen des Dialogs wird das Programmelement um diese Aktion erweitert, was sowohl im Klassendiagramm als auch im Projektbaum sofort aktualisiert wird.

Cursor

-  : An diesen Stellen im Diagramm ist Erweitern nicht möglich.
-  : An diesen Stellen im Diagramm können Sie das fokussierte Element um eine Aktion erweitern.

6.4.9 Komposition

Bei einer Komposition handelt es sich um eine UML-Beziehung, die ein „Enthalten“ ausdrückt: Ein Element enthält ein anderes Element. In IEC-Code entspricht das der Instanziierung eines Elements: fbSample : FB_Sample. Über die Kardinalität wird angegeben, wie oft die Beziehung besteht. In IEC-Code entspricht das einem ARRAY[...]. Wird eine Kardinalität größer als 1 angegeben, wird folgendermaßen deklariert: aSample : ARRAY[1..10] OF FB_Sample.



Eine Komposition wird als Pfeil mit einer gefüllten schwarzen Raute dargestellt, der von einer Klasse oder einer Globalen Variablenliste auf eine Klasse vom Typ FUNCTION_BLOCK oder eine DUT zeigt.

Eigenschaft

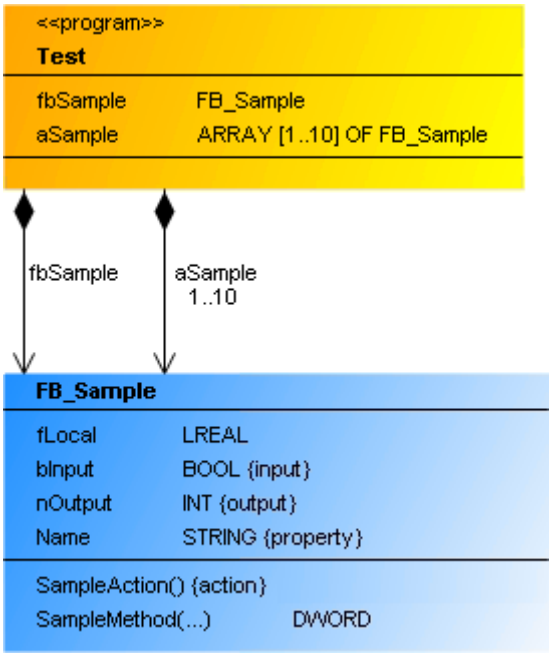
„Eigenschaft“	Beschreibung
„Beziehung“	Komposition (nicht editierbar)
„Route optimieren“	Wenn die Option aktiviert ist, wird die Route des Beziehungspfeils automatisch optimiert. Fixpunkte sind dabei der Startpunkt am Startelement und der Endpunkt am Zielelement. Wird zum Beispiel das Zielelement verschoben, bleibt der Punkt, an dem der Pfeil auf das Zielelement zeigt, bestehen. Wenn die Option deaktiviert ist, bleibt der Streckenverlauf erhalten. Sobald im Klassendiagramm ein Beziehungselement manuell positioniert wird, ist diese Option abgewählt. Aktivieren Sie die Option, wenn ein automatisches Optimieren gewünscht ist.
„Startelement“	Der Name des Elements, bei dem das Beziehungselement startet, wird angezeigt.
„Zielelement“	Der Name des Elements, auf den das Beziehungselement zeigt, wird angezeigt.
„Bezeichner“	Der Name des Beziehungselements wird angezeigt.

Komposition editieren

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Komposition“:  Selektieren Sie eine Klasse oder eine GVL und klicken Sie dann auf das Element, das Bestandteil werden soll.	Eine Komposition zwischen den Elementen wird eingezeichnet.	Der IEC-Code wird automatisch angepasst, indem der Deklarationsteil des bestehenden Elements erweitert wird. Zum Beispiel: fbExistent : FB_Existent;.
Wählen Sie das Werkzeug „Komposition“:  Selektieren Sie eine Klasse oder eine GVL und klicken Sie dann in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen“ öffnet. Fügen Sie einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Komposition, die von der Klasse oder GVL auf die neue Klasse weist, ist erzeugt.	Der IEC-Code wird automatisch angepasst, indem der Deklarationsteil des bestehenden Elements erweitert wird. Zum Beispiel: fbNew : FB_New;
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Komposition und verschieben Sie die Linie mit der Maus.		Die selektierte und deshalb blaue Komposition verläuft auf der neuen Position. Die Eigenschaft „Routing optimieren“ wird automatisch deaktiviert.
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Komposition und verwenden Sie die [Entf]-Taste oder klicken Sie im Kontextmenü „Löschen“.		Die Assoziation ist im Diagramm und im IEC-Code entfernt. Im Deklarationsteil des Elements ist die Instanziierung der Klasse bzw. des Datentyps entfernt.

Beispiele

- Komposition einer Klasse



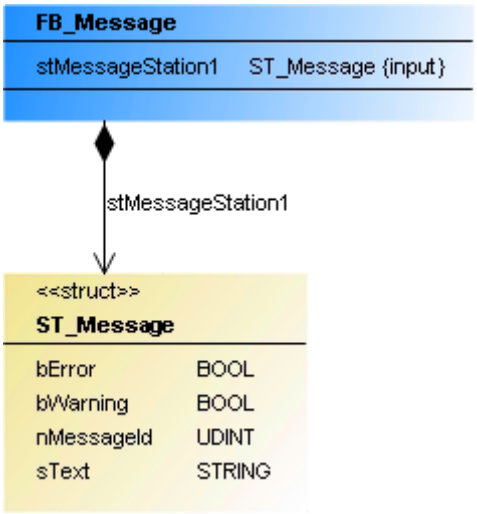
- Einfachkomposition

```
PROGRAM Test
VAR
    fbSample    : FB_Sample;
END_VAR
```

- Vielfachkomposition

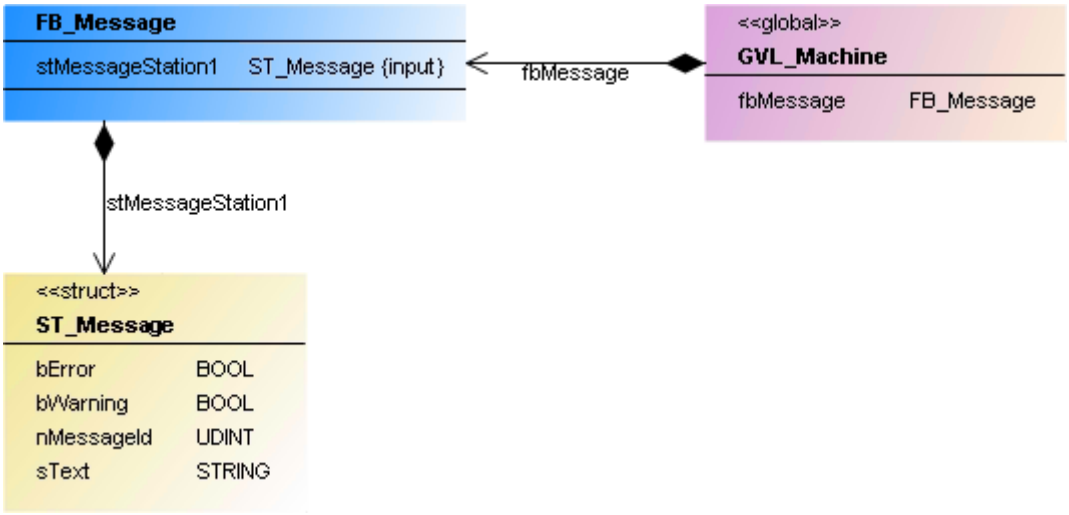
```
PROGRAM Test
VAR
    aSample    : ARRAY[1..10] OF FB_Sample;
END_VAR
```

- Komposition eines Datenobjekts




```
FUNCTION_BLOCK FB_Message
VAR_INPUT
    stMessageStation1    : ST_Message;
END_VAR
```

- Komposition in GVL

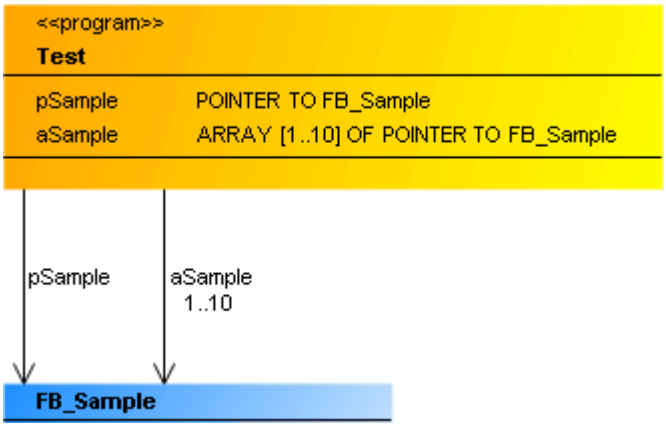



```
VAR_GLOBAL
    fbMessage    : FB_Message;
END_VAR
```

6.4.10 Assoziation

Bei einer Assoziation handelt es sich um eine UML-Beziehung, die ein „Kennen“ ausdrückt. Das kennende Element weist als Zeiger oder Referenz auf ein anderes Element. In IEC-Code entspricht das einer POINTER TO-Anweisung (pSample : POINTER TO FB_Sample) oder einer REFERENCE TO-Anweisung (refSample : REFERENCE TO FB_Sample).

Bei Zeigern wird über die Kardinalität angegeben, wie oft die Beziehung besteht. In IEC-Code entspricht das einem ARRAY[…]. Wird eine Kardinalität größer als 1 angegeben, wird folgendermaßen deklariert: aSample : ARRAY[1..10] OF POINTER TO FB_Sample.



Eine Assoziation wird als Pfeil dargestellt, der von einer Klasse oder einer Globalen Variablenliste auf eine Klasse vom Typ FUNCTION_BLOCK oder DUT zeigt.

Eigenschaft

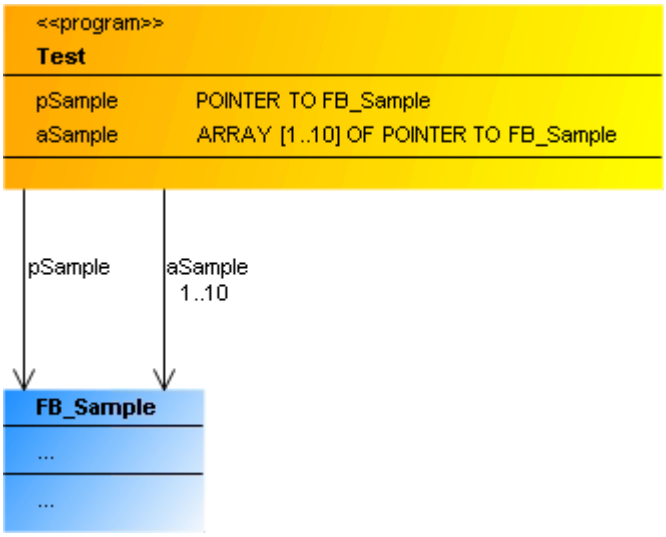
„Eigenschaft“	Beschreibung
„Beziehung“	Assoziation (nicht editierbar)
„Route optimieren“	Wenn die Option aktiviert ist, wird die Route des Beziehungspfeils automatisch optimiert. Fixpunkte sind dabei der Startpunkt am Startelement und der Endpunkt am Zielelement. Wird zum Beispiel das Zielelement verschoben, bleibt der Punkt, an dem der Pfeil auf das Zielelement zeigt, bestehen. Wenn die Option deaktiviert ist, bleibt der Streckenverlauf erhalten. Sobald im Klassendiagramm ein Beziehungselement manuell positioniert wird, ist diese Option abgewählt. Aktivieren Sie die Option, wenn ein automatisches Optimieren gewünscht ist.
„Startelement“	Der Name des Elements, bei dem das Beziehungselement startet, wird angezeigt.
„Zielelement“	Der Name des Elements, auf den das Beziehungselement zeigt, wird angezeigt.
„Bezeichner“	Der Name des Beziehungselements wird angezeigt.

Assoziation editieren

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Assoziation“:  Selektieren Sie eine Klasse oder eine GVL und klicken Sie dann auf das Objekt, auf das assoziiert werden soll.	Eine Assoziation zwischen den Elementen wird eingezeichnet.	Der IEC-Code wird automatisch angepasst, indem der Deklarationsteil des bestehenden Elements erweitert wird. Zum Beispiel: pExistent : POINTER TO FB_Existent;
Wählen Sie das Werkzeug „Assoziation“:  Selektieren Sie eine Klasse oder eine GVL und dann klicken Sie dann in einen freien Bereich des Diagramms. Der Dialog „POU hinzufügen“ öffnet. Fügen Sie einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Assoziation, die von der Klasse oder GVL auf die neue Klasse weist, ist erzeugt.	Der IEC-Code wird automatisch angepasst, indem der Deklarationsteil des bestehenden Elements erweitert wird. Zum Beispiel: pNew : POINTER TO FB_New;
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Assoziation und verschieben Sie die Linie mit der Maus.		Die selektierte und deshalb blaue Assoziation verläuft auf der neuen Position. Die Eigenschaft „Routing optimieren“ wird automatisch deaktiviert.
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Assoziation und verwenden Sie die [Entf]-Taste oder klicken Sie im Kontextmenü „Löschen“.		Die Assoziation ist im Diagramm und im IEC-Code entfernt. Im Deklarationsteil des Elements ist die POINTER TO- bzw. die REFERENCE TO-Anweisung entfernt.

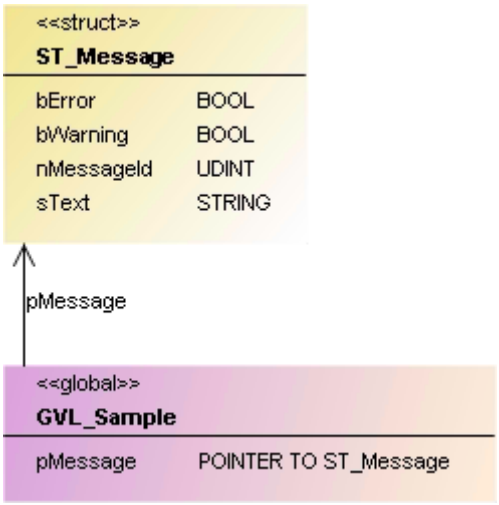
Beispiele

- Assoziation auf eine Klasse (von einem Programm) – einfach und mehrfach




```
PROGRAM Test
VAR
    pSample    : POINTER TO FB_Sample;
    aSample    : ARRAY[1..10] OF POINTER TO FB_Sample;
END_VAR
```

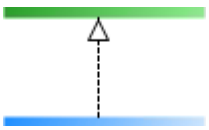
- Assoziation auf eine Datenstruktur (von einer GVL) – einfach




```
VAR_GLOBAL
    pMessage    : POINTER TO ST_Message;
END_VAR
```

6.4.11 Realisierung

Bei einer Realisierung handelt es sich um eine UML-Beziehung, die eine Interface-Einbindung ausdrückt. Das realisierende bzw. einbindende Klassenobjekt (Funktionsblock) implementiert die Eigenschaften und Methoden der Schnittstelle. In IEC-Codierung entspricht diese Beziehung dem Schlüsselwort IMPLEMENTS.



Eine Realisierung wird durch einen gestrichelten Pfeil symbolisiert, der von einer Klasse des Typs FUNCTION_BLOCK zu einer Schnittstelle zeigt.

Eigenschaft

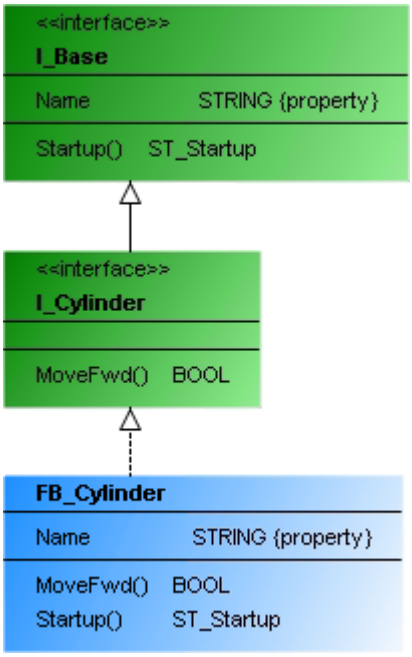
„Eigenschaft“	Beschreibung
„Beziehung“	Realisierung (nicht editierbar)

Realisierung editieren

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Realisierung“:  Selektieren Sie eine Klasse und dann eine Schnittstelle.	Eine Realisierung, die von der Klasse auf die Schnittstelle weist, ist erzeugt.	Der IEC-Code wird automatisch angepasst, indem im Deklarationsteil der Klasse die Schnittstelle angegeben ist. Zum Beispiel: FUNCTION_BLOCK FB_Sample IMPLEMENTS I_Existent
Wählen Sie das Werkzeug „Realisierung“:  Selektieren Sie eine Klasse und klicken Sie dann in einen freien Bereich des Diagramms. Der Dialog „Schnittstelle hinzufügen“ öffnet. Fügen Sie einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Realisierung, die von der Klasse auf die neue Schnittstelle weist, ist erzeugt.	Der IEC-Code wird automatisch angepasst, indem eine neue Schnittstelle angelegt wird und im Deklarationsteil der Klasse diese Schnittstelle angegeben wird. Zum Beispiel: FUNCTION_BLOCK FB_Sample IMPLEMENTS I_New
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Realisierung und verschieben Sie die Linie mit der Maus.		Die selektierte und deshalb blaue Realisierung verläuft auf der neuen Position.
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Realisierung und verwenden Sie die [Entf]-Taste oder klicken Sie im Kontextmenü „Löschen“.		Die Realisierung ist im Diagramm und im IEC-Code entfernt. Im Deklarationsteil der Klasse ist die Anweisung IMPLEMENTS entfernt.

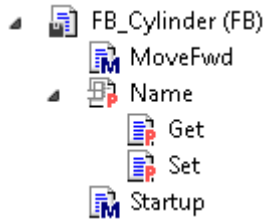
Beispiel

- Realisierung




```
INTERFACE I_Cylinder EXTENDS I_Base
```

```
FUNCTION_BLOCK FB_Cylinder IMPLEMENTS I_Cylinder
```



6.4.12 Generalisierung

Bei einer Generalisierung handelt es sich um eine UML-Beziehung, die eine Vererbung oder Spezialisierung ausdrückt. Das erbende Element verfügt über die Attribute und Operationen des vererbenden Elements. In IEC-Codierung entspricht diese Beziehung dem Schlüsselwort EXTENDS.

Eine Generalisierung wird durch einen Pfeil symbolisiert, der über eine geschlossene Spitze verfügt und von der erbenden Klasse zur vererbenden Basisklasse zeigt. Die Richtung des Pfeils gibt also an, wer von wem erbt.

Vererbung ist zwischen Klassen, Schnittstellen und benutzerdefinierten Datentypen möglich:

- Ein Funktionsbaustein kann von einem anderen Funktionsbaustein erben.
- Eine Schnittstelle kann von einer anderen Schnittstelle erben.
- Ein DUT kann von einem anderen DUT erben.
- Programme und Funktionen können nicht erben oder vererben.

Eigenschaft

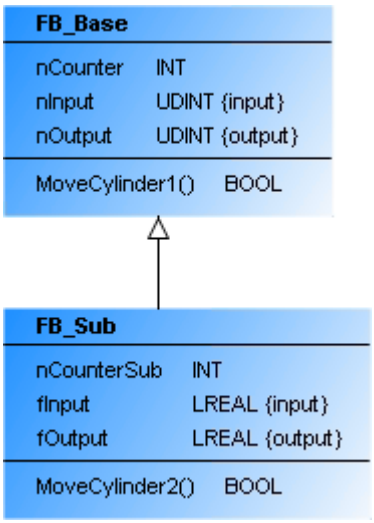
„Eigenschaft“	Beschreibung
„Beziehung“	Generalisierung (nicht editierbar)

Generalisierung editieren

Benutzereingabe im Klassen-diagramm	Reaktion im Klassendiagramm	Beschreibung
Wählen Sie das Werkzeug „Generalisierung“:  Selektieren Sie das Objekt, das erben soll, und klicken Sie dann auf das Vaterobjekt.	Eine Generalisierung, die vom Erbenden zum Vaterelement weist, ist erzeugt.	Der IEC-Code wird automatisch angepasst, indem im Deklarationsteil der erbenden Klasse das Vaterelement angegeben ist. Zum Beispiel: FUNCTION_BLOCK FB_Sample EXTENDS FB_BaseExistent
Wählen Sie das Werkzeug „Generalisierung“:  Selektieren Sie das Objekt, das erben soll, und klicken Sie dann in einen freien Bereich des Diagramms. Ein Dialog zum Erzeugen eines neuen Objekts öffnet. Fügen Sie einen Namen ein, passen Sie die Einstellungen an und beenden Sie den Dialog mit „Hinzufügen“.	Eine Generalisierung, die vom Erbenden zum neuen Vaterelement weist, ist erzeugt.	Der IEC-Code wird automatisch angepasst, indem das neue Objekt angelegt wird und im Deklarationsteil des erbenden Objekts das Vaterobjekt angegeben wird. Zum Beispiel: FUNCTION_BLOCK FB_Sample EXTENDS FB_BaseNew
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Generalisierung und verschieben Sie die Linie mit der Maus.		Die selektierte und deshalb blaue Generalisierung verläuft auf der neuen Position.
Wählen Sie das Werkzeug „Zeiger“. Klicken Sie auf eine Realisierung und verwenden Sie die [Entf]-Taste oder klicken Sie im Kontextmenü „Löschen“.		Die Generalisierung ist im Diagramm und im IEC-Code entfernt. Im Deklarationsteil des erbenden Objekts ist die Anweisung EXTENDS entfernt.

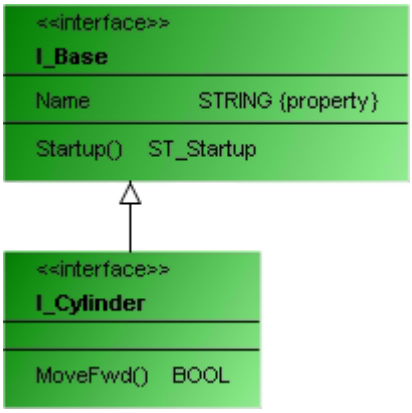
Beispiele

- Funktionsbaustein



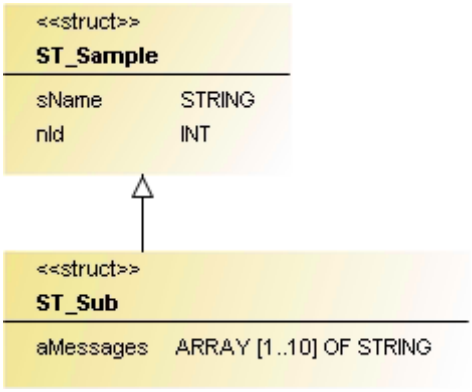

```
FUNCTION_BLOCK FB_Sub EXTENDS FB_Base
```

- Schnittstelle




```
INTERFACE I_Cylinder EXTENDS I_Base
```

- DUT

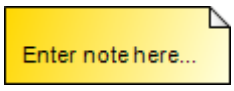



```
TYPE ST_Sub EXTENDS ST_Sample :
STRUCT
    aMessages    : ARRAY[1..10] OF STRING;
END_STRUCT
END_TYPE
```

6.4.13 Notiz

Mit dem Element „Notiz“ können Sie im Editor eines Klassendiagramms oder Zustandsdiagramms einen kommentierenden Text einfügen.

Aktuell sind nur einzeilige Notizen möglich.



Notiz hinzufügen

Wählen Sie das Werkzeug „Notiz“: 

Klicken Sie im Editor auf die gewünschte Einfügeposition. Doppelklicken Sie anschließend auf den Text im Element und ersetzen Sie ihn durch den gewünschten Text.

Cursor

-  : An nicht erlaubten Einfügestellen hat der Cursor die Form eines Verbotsschilds.

-  : An erlaubten Stellen ist der Cursor ein blaues Kreuz.

7 UML-Zustandsdiagramm

Bitte beachten Sie zusätzlich zu den folgenden Informationen auch die Samples [▶ 97], die eine erste Einführung in das Tool und in das Aufrufverhalten vom UML Zustandsdiagramm geben.

7.1 Grundlagen

Das UML Zustandsdiagramm ist ein grafischer Formalismus, um den Ablauf eines ereignisdiskreten Systems zu spezifizieren und zu designen bzw. um ein Zeitverhalten zu programmieren. Integriert in den SPS-Bereich der TwinCAT 3 Entwicklungsumgebung verfügt ein Zustandsdiagramm über einen Editor mit Zustands- und Transitionselementen. Beim Kompilieren der Applikation wird ausführbarer Code erzeugt. Das Schaltverhalten eines Zustandsdiagramms ist bei der Ausführung standardmäßig abhängig vom Taskzyklus getaktet, aber je nach Konfiguration der Zustände kann es auch unabhängig schalten. In diesem Fall handelt es sich um zyklusinterne Zustände. Leistungsstarke Funktionalitäten wie Syntaxüberwachung oder Debuggen im Online-Modus unterstützen den Entwicklungsprozess.

Anwendungsfall

Ein Zustandsdiagramm wird verwendet, um zeitdiskrete Systeme zu programmieren. Programme, Funktionsbausteine, Funktionen, Methoden oder Aktionen können mit der Implementierungssprache **Zustandsdiagramm** als Graph aus Zuständen und Transitionen erstellt werden.

Funktionen/Methoden und ihre Daten sind üblicherweise temporär. Wenn Sie eine Funktion/Methode mit der Implementierungssprache UML Zustandsdiagramm implementieren, beinhaltet das Diagramm zyklusinterne Zustände und initialisiert am Anfang des Taskzyklus die Daten neu.

Bei Methoden gibt es hingegen die Möglichkeit, die Methode als VAR_INST zu deklarieren. Dann werden die Daten wie bei einem Funktionsbaustein gehalten und sind nicht mehr temporär, sodass das Zustandsdiagramm wie gewohnt in mehreren Taskzyklen durchlaufen wird. Sehen Sie hierzu auch die Option VarInstNutzen (UseVarInst) [▶ 90]. Bei Funktionen steht diese Option nicht zur Verfügung.

Syntax

Die Syntax wird vom Compiler überprüft:

- Eine Transition zwischen Zuständen, die in unterschiedlichen Regionen liegen, ist nicht erlaubt
- Orthogonale Zustände können nicht in anderen Zuständen verschachtelt werden
- Genau eine Abschlusstransition und/oder eine Ausnahmetransition geht von einem zusammengesetzten Zustand ab
- Eine bedingte Transition in einen orthogonalen Zustand hinein ist nicht erlaubt. Dann ist zusätzlich eine Gabelung oder Verbindung erforderlich

Befehle

Für das Zustandsdiagramm stehen folgende Befehle bzw. Aktionsmöglichkeiten zur Verfügung:

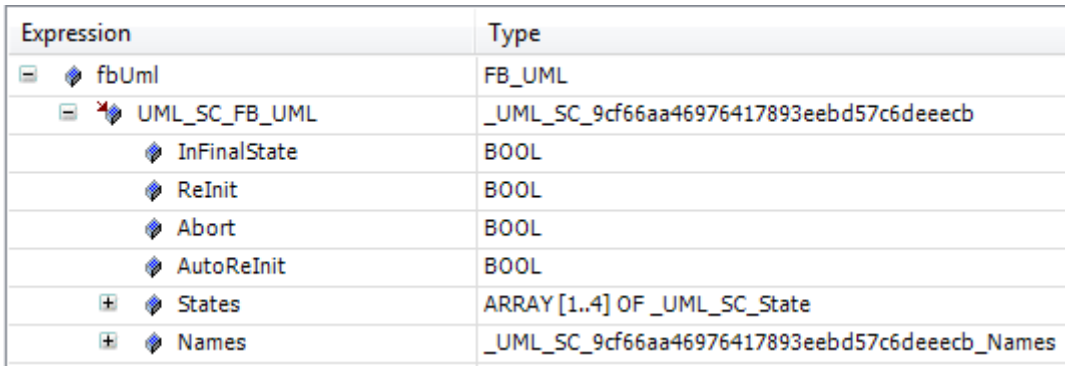
- Neues Zustandsdiagramm anlegen [▶ 58]
- Zustandsdiagramm bearbeiten [▶ 59]
- Gehe zur Definition [▶ 62]

Beachten Sie außerdem die Befehle, die für alle UML-Diagramme verfügbar sind: Gemeinsame Befehle aller UML-Diagramme [▶ 15].

Implizite Variablen eines Zustandsdiagramms

Die impliziten Variablen befinden sich in der Programmeinheit, die das Zustandsdiagramm abbildet (z.B. in der Instanz des Funktionsbaustein oder in einem Programm). Die internen Variablen befinden sich innerhalb der Programmeinheit unter dem Elementnamen „UML_SC_<POU-Name>“ mit dem Datentyp „UML_SC_<id>“.

Für den Funktionsbaustein „FB_UML“ lautet der Elementname der impliziten Variablen beispielsweise „UML_SC_FB_UML“. Sie befinden sich in der Funktionsbausteininstanz „fbUml“.



„UML_SC_<POU-Name>“ vom Datentyp _UML_SC_<id>:

Einige implizite Variablen geben den Status des Objekts während der Laufzeit wieder (*InFinalState*, *States*), andere dienen zur Verhaltenssteuerung zur Laufzeit (*ReInit*, *Abort*, *AutoReInit*).

Name	Datentyp	Bedeutung
InFinalState	BOOL	Diese Variable besitzt den Wert TRUE, wenn sich das Zustandsdiagramm im Endzustand befindet.
ReInit	BOOL	Wenn Sie diese Variable auf TRUE setzen, wird das Zustandsdiagramm reinitialisiert, d.h. der Startzustand des Diagramms wird aktiviert.
Abort	BOOL	Wenn Sie diese Variable auf TRUE setzen, wird die aktuelle Operation des Diagramms abgebrochen und der Endzustand des Diagramms wird aktiviert.
AutoReInit	BOOL	Wenn diese Variable den Wert TRUE besitzt, wird automatisch wieder in den Startzustand geschaltet, sobald der Endzustand des Diagramms erreicht ist. Default-Wert: TRUE
States	ARRAY[<Anzahl der Zustände>] OF _UML_SC_State	siehe unten
Names	_UML_SC_<id>_Names	siehe unten

„UML_SC_<POU-Name>.States“ vom Datentyp ARRAY[<Anzahl der Zustände>] OF _UML_SC_State:

Die Variablen der Struktur _UML_SC_State dienen der Beschreibung eines Zustands. Dieser Datentyp ist der Basistyp eines Arrays mit dem Namen „States“, das in den impliziten Variablen deklariert ist und die einzelnen Zustände des Zustandsdiagramms beschreibt.

In der nachfolgenden Tabelle sind die einzelnen Variablen der Struktur _UML_SC_State beschrieben.

Name	Datentyp	Bedeutung
Active	BOOL	Flag, um zu ermitteln, ob der Zustand aktuell aktiv ist.
FastExecutionFault	BOOL	Relevant für IntraCycle-States: Wenn ein IntraCycle-State länger als einen Zyklus aktiv ist, wird dieses Flag vom System gesetzt. Das Flag wird bei Verlassen des IntraCycle-States zurückgesetzt.
ID	INT	ID des Zustands Die ID des Zustands entspricht dem Index, an dem der Zustand im „States“-Array beschrieben ist.
ActivationTime	TIME	Zeitstempel der letzten Aktivierung des Zustands
Name	STRING	Name des Zustands

„UML_SC_<POU-Name>.Names“ vom Datentyp _UML_SC_<id>_Names:

Der Datentyp _UML_SC_<id>_Names enthält für jeden Zustand des Zustandsdiagramms eine INT-Variable, die mit dem Namen des Zustands deklariert ist und deren Wert die ID des Zustands repräsentiert. Die ID des Zustands entspricht dem Index, an dem der Zustand im „States“-Array beschrieben ist.

Name	Datentyp	Bedeutung
<Name des Zustands> **Beispiel:** StartState1	INT	ID des Zustands (= Index, an dem der Zustand im „States“-Array beschrieben ist) **Beispiel:** ID des Zustands „StartState1“, z.B. 4
<Name des Zustands> **Beispiel:** State1	INT	ID des Zustands (= Index, an dem der Zustand im „States“-Array beschrieben ist) **Beispiel:** ID des Zustands „State1“, z.B. 5
<Name des Zustands> **Beispiel:** State2	INT	ID des Zustands (= Index, an dem der Zustand im „States“-Array beschrieben ist) **Beispiel:** ID des Zustands „State2“, z.B. 6
…	…	…

Beispiel:

Es wird ein Funktionsbaustein mit der Implementierungssprache UML SC und dem Namen „FB_UML“ angelegt. Das Statechart-Diagramm verfügt über einen Zustand mit dem Namen „MyStateName“. Der FB beinhaltet zusätzlich die Methode „MyMethod“ in der Sprache ST. In dem folgenden Beispiel wird von innerhalb sowie von außerhalb des FBs abgefragt, ob sich das Statechart-Diagramm aktuell in dem Zustand „MyStateName“ befindet.

Zugriff von innerhalb des FBs / in FB_UML.MyMethod:

```
FUNCTION_BLOCK FB_UML
VAR
    bInSpecificState : BOOL;
    …
END_VAR
```

```
METHOD MyMethod : BOOL
bInSpecificState := UML_SC_FB_UML.States[UML_SC_FB_UML.Names.MyStateName].Active;
```

Zugriff von außerhalb des FBs / in MAIN:

```
PROGRAM MAIN
VAR
    fbInstance       : FB_UML;
    bInSpecificState : BOOL;
END_VAR
```

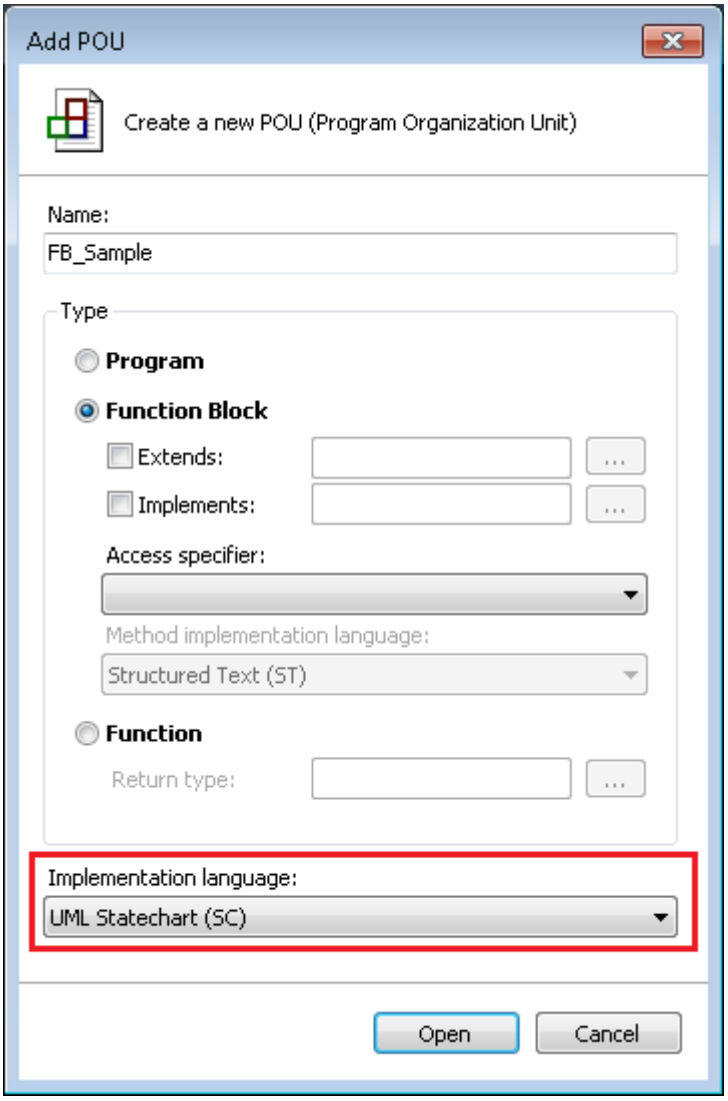
```
fbInstance();
fbInstance.MyMethod();

bInSpecificState := fbInstance.UML_SC_FB_UML.States[fbInstance.UML_SC_FB_UML.Names.MyStateName].Acti
ve;
```

7.2 Befehle

7.2.1 Neues Zustandsdiagramm anlegen

Programme, Funktionsbausteine, Funktionen, Methoden oder Aktionen können mit der Implementierungssprache „Zustandsdiagramm“ als Graph aus Zuständen und Transitionen erstellt werden.

1. Wählen Sie im Kontextmenü des Projektbaums den Befehl, um das gewünschte Programmelement (z.B. Funktionsbaustein, Methode) hinzuzufügen.
2. Konfigurieren Sie den aufgehenden Dialog wie gewohnt (für ein POU-Element z.B. Name und Typ) und wählen Sie als Implementierungssprache **UML Zustandsdiagramm (Statechart SC)** (UML Statechart (SC)).



3. Bestätigen Sie die Eingaben und Konfigurationen über die **Öffnen**-Schaltfläche.

⇨ Im Projektbaum wird das neue Zustandsdiagramm-Objekt eingefügt und der Editor des neuen Diagramms wird geöffnet.

7.2.2 Zustandsdiagramm bearbeiten

Um ein Zustandsdiagramm zu editieren stehen unter anderem die folgenden Aktionen zur Verfügung. Weitere Bearbeitungsmöglichkeiten finden Sie unter Editor [▶ 63].

Neues Element einfügen

1. Öffnen Sie das Fenster **Werkzeuge** über das Menü **Ansicht**.
2. Markieren Sie ein Element [▶ 63] in der Ansicht **Werkzeuge** und ziehen Sie es per Drag'n'Drop auf das geöffnete Zustandsdiagramm. Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.

⇨ Das neue Element wird im Diagramm dargestellt.

Selektion der Ansicht "Werkzeuge" abwählen

✓ Im Fenster **Werkzeuge** ist ein Element selektiert. Im Editor hat der Mauscursor die Form des selektierten Elements.

1. Drücken Sie die rechte Maustaste.

⇨ Die Selektion des Elements wird abgewählt und das Standardelement **Zeiger** wird selektiert.

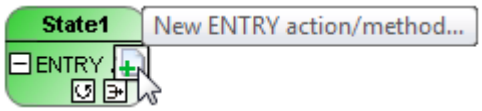
Bezeichner editieren

1. Öffnen Sie mit zwei Einzelklicks den Zeileneditor eines Bezeichners.
2. Geben Sie einen neuen Namen ein und bestätigen Sie die Eingabe über die [Enter]-Taste oder indem Sie in einen freien Bereich des Diagramms klicken.

⇨ Der Bezeichner trägt den neuen Namen.

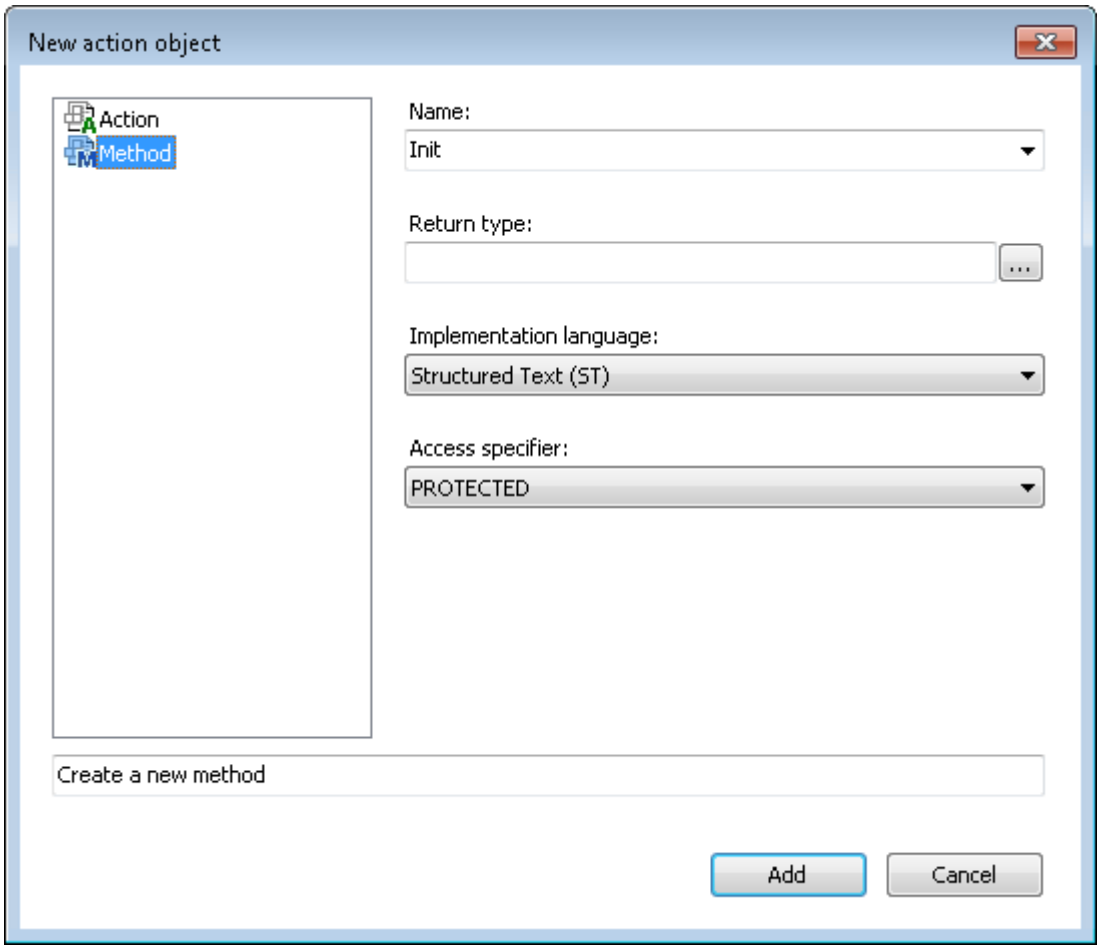
Neues Aktionsobjekt hinzufügen

1. Klicken Sie auf das Symbol **Neue <ENTRY/DO/EXIT> Aktion/Methode**, welches bei einem Zustand am Ende einer Aktionszeile und bei einem zusammengesetzten Zustand am Anfang einer Aktionszeile erscheint.
   Voraussetzung: Die Aktionszeile verfügt bislang über keine Deklaration oder Implementation.



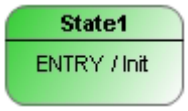
   ⇨ Der Dialog **Neues Aktionsobjekt** wird geöffnet.
2. Legen Sie im linken Teil des Dialogs den Objekttyp fest, indem Sie **Aktion** oder **Methode** auswählen.

3. Falls Sie den Objekttyp **Aktion** ausgewählt haben: Konfigurieren Sie den Namen und die Implementierungssprache der neuen Aktion.
   Falls Sie den Objekttyp **Methode** ausgewählt haben: Konfigurieren Sie den Namen, den Rückgabetyp, die Implementierungssprache und den Zugriffsmodifizierer der neuen Methode.



4. Bestätigen Sie die Eingaben und Konfigurationen über die **Hinzufügen**-Schaltfläche.

⇨ Das neue Aktions- bzw. Methodenobjekt wird im Projektbaum eingefügt und in dem bearbeiteten Zustand als verknüpftes Objekt anzeigt. Außerdem wird der Editor der neuen Aktion bzw. der neuen Methode zur Bearbeitung geöffnet.



Neue Transition erzeugen

Transitionen können ausgehend von verschiedenen Quellelementen erzeugt werden: Startzustand, Zustand, zusammengesetzter Zustand, Auswahl, Gabelung/Verbindung.
Die möglichen Zielelemente, für die eine eingehende Transition erzeugt wird, sind abhängig von der Art des Quellelements und der Art der Transition (Transition, Abschlusstransition, Ausnahmetransition).

Im Folgenden wird die generelle Vorgehensweise zur Erzeugung einer Transition erläutert.

Möglichkeit 1 – via Icon:

1. Markieren Sie ein Element im geöffneten Zustandsdiagramm.
2. Klicken Sie auf ein Transitionsicon, das über dem Element erscheint (das markierte Element agiert als Quellelement).

3. Klicken Sie auf ein bestehendes Element, um dieses als Zielelement der gewählten Transition zu konfigurieren. Klicken Sie alternativ auf einen freien Bereich des Editors, um einen neuen Zustand zu erzeugen.
4. Konfigurieren Sie je nach Bedarf die Aktion und die Wächterbedingung der Transition, indem Sie die jeweiligen Zeileneditoren mittels zweier Einzelklicks öffnen. Falls die Wächterbedingung von mehr als einer Variablen abhängt, können Sie bei der Konfiguration der Transitionsbedingung über [Strg+Enter] eine neue Zeile einfügen, um den gesamten Transitionsausdruck übersichtlich darzustellen.

⇨ Sie haben eine Transition vom Quell- zum Zielelement erstellt, welche das Schaltverhalten des Zustandsdiagramms bestimmt.

Möglichkeit 2 – via **Werkzeuge**:

1. Klicken Sie auf ein Transitionselement aus **Werkzeuge**.
2. Klicken Sie im geöffneten Zustandsdiagramm zunächst auf das Quellelement und anschließend auf ein bestehendes Element, um dieses als Zielelement der gewählten Transition zu konfigurieren. Klicken Sie alternativ auf einen freien Bereich des Editors, um einen neuen Zustand zu erzeugen.
3. Konfigurieren Sie je nach Bedarf die Aktion und die Wächterbedingung der Transition, indem Sie die jeweiligen Zeileneditoren mittels zweier Einzelklicks öffnen. Falls die Wächterbedingung von mehr als einer Variablen abhängt, können Sie bei der Konfiguration der Transitionsbedingung über [Strg+Enter] eine neue Zeile einfügen, um den gesamten Transitionsausdruck übersichtlich darzustellen.

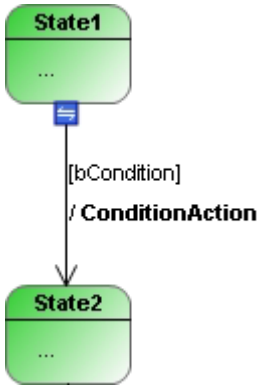
⇨ Sie haben eine Transition vom Quell- zum Zielelement erstellt, welche das Schaltverhalten des Zustandsdiagramms bestimmt.

Endpunkte einer bestehenden Transition ändern

Die Endpunkte einer bestehenden Transition können mithilfe der Reconnect-Funktionalität mit anderen Quell- oder Zielelementen verbunden werden. Dadurch bleiben die bereits vorgenommen Konfigurationen der Transition (Aktion, Wächterbedingung) erhalten, während sich die Anschlüsse der Transition ändern.

1. Klicken Sie, je nachdem ob Sie das Quell- oder das Zielelement ändern möchten, auf den Anfang oder das Ende einer Transition.

   ⇨ Das Reconnect-Symbol erscheint an dem Transitionsanschluss.



2. Ziehen Sie das Symbol per Drag-and-Drop auf ein anderes Quell- bzw. ein anderes Zielelement, um die Transition dort anzuschließen.

⇨ Die Transition wurde mit einem anderen Quell-/Zielelement verbunden, ohne die anderen Transitionskonfigurationen zu verändern.

Als Projektnavigator verwenden

1. Öffnen Sie ein Zustandsdiagramm und doppelklicken Sie auf ein Aktionsobjekt eines (zusammengesetzten) Zustands.

   ⇨ Der Editor des Aktionsobjekts wird geöffnet.
2. Bei Bedarf können Sie die Deklaration und Implementierung des Aktionsobjekts in dem geöffneten Editor bearbeiten.

⇨ Mithilfe der Projektnavigator-Funktionalität kann das Verhalten eines ausgewählten Aktionsobjekts leicht eingesehen bzw. bei Bedarf angepasst werden.

Mehrfachauswahl

- Wenn **Zeiger** in **Werkzeuge** aktiviert ist (Standard), können Sie im Zustandsdiagramm mit gedrückter linker Maustaste ein Rechteck über mehrere Elemente ziehen. Alle erfassten Elemente werden dann selektiert.
- Mehrfachauswahl ist auch möglich, indem Sie die gewünschten Elemente bei gedrückter [Strg]-Taste nacheinander via Mausklick auswählen.
- [Strg+A] bzw. **Alles selektieren** bewirkt, dass alle Elemente des Diagramms selektiert werden.

7.2.3 Gehe zu Definition

Für Transitionen steht der Befehl **Gehe zu Definition** (Go To Definition) im dazugehörigen Kontextmenü zur Verfügung.

Der Befehl steht im Offline- und Online-Modus zur Verfügung.

1. Öffnen Sie über einen Rechtsklick auf eine Transition das dazugehörige Kontextmenü.
2. Wählen Sie den Befehl **Gehe zu Definition**.

⇨ Der Editor, in dem die erste Variable der Transition bzw. des Transitionsausdrucks deklariert ist, wird geöffnet und die Variable wird im Deklarationseditor markiert.

7.2.4 Alle Verweise suchen

Für Transitionen steht der Befehl **Alle Verweise suchen** (Find All References) im dazugehörigen Kontextmenü zur Verfügung.

Der Befehl steht im Offline- und Online-Modus zur Verfügung.

1. Öffnen Sie über einen Rechtsklick auf eine Transition das dazugehörige Kontextmenü.
2. Wählen Sie den Befehl **Alle Verweise suchen**.

⇨ Die Ansicht **Querverweisliste** öffnet sich und zeigt die Verwendungsstellen der ersten Variablen der Transition bzw. des Transitionsausdrucks.

7.2.5 Zur Überwachungsliste hinzufügen

Für Transitionen steht der Befehl **Zur Überwachungsliste hinzufügen** (Add Watch) im dazugehörigen Kontextmenü zur Verfügung.

Der Befehl steht nur im Online-Modus zur Verfügung und stellt eine – zusätzlich zu der integrierten Online-Ansicht neben der Transition – eine weitere Debug-Möglichkeit dar, indem die Variablen einer Transition per Befehl in die Überwachungsliste übernommen werden können. Der Befehl ist besonders dann hilfreich, wenn sich ein Transitionsausdruck aus der Kombination mehrerer Variablen zusammensetzt (z.B. „bCondition1 AND bCondition2“). Wenn Sie für diese Transition den Befehl **Zur Überwachungsliste hinzufügen** ausführen, werden beide Variablen des Transitionsausdrucks zur Überwachungsliste hinzugefügt, also *bCondition1* und *bCondition2*. Dadurch können Sie sich einen einfachen Überblick verschaffen, welches Teilsignal des Gesamtausdrucks für die Weiterschaltung in den nächsten Zustand noch nicht den benötigten Wert hat.

1. Öffnen Sie über einen Rechtsklick auf eine Transition das dazugehörige Kontextmenü. Der Transitionsausdruck kann sich dabei aus einer oder mehrerer Variablen zusammensetzen.
2. Wählen Sie den Befehl **Zur Überwachungsliste hinzufügen**.

⇨ Die Variable(n) des Transitionsausdrucks wird/werden in die gerade geöffnete Überwachungsliste eingefügt. Wenn gerade keine Überwachungsliste geöffnet ist, fügt der Befehl die Variable(n) in Überwachungsliste 1 ein und öffnet deren Ansicht.

7.3 Editor

Der Zustandsdiagramm-Editor dient dem Entwerfen und Programmieren von Abläufen mittels eines Zustandsdiagramms. Der **Zustand** ist das Hauptelement des Zustandsdiagramms. Normalerweise wird ein Zustandsdiagramm durch den Taskzyklus getaktet, aber auch zyklusinterne Zustandsmaschinen sind möglich (siehe Konfiguration eines Zustands [▶ 65]).

Das Zustandsdiagramm kann mittels unterschiedlicher Aktionen editiert werden. Weiterführende Informationen zu den Bearbeitungsmöglichkeiten finden Sie unter Zustandsdiagramm bearbeiten [▶ 59].

Beachten Sie außerdem weitere, elementabhängige Benutzereingaben:

- Zustand editieren [▶ 67]
- Transition editieren [▶ 84]
- Abschlusstransition editieren [▶ 85]
- Ausnahmetransition editieren [▶ 89]
- Startzustand editieren [▶ 65]
- Auswahl editieren [▶ 82]
- Zusammengesetzten Zustand editieren [▶ 72]
- Gabelung/Verbindung editieren [▶ 80]

7.4 Elemente

Das Fenster **Werkzeuge** stellt die Elemente des Zustandsdiagramms zur Verfügung. Sie können via Drag'n'Drop in das Zustandsdiagramm-Fenster eingefügt werden. Zustände und Pseudozustände können beliebig platziert werden.

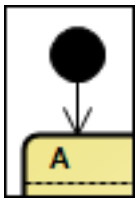
Element einfügen

1. Führen Sie **Ansicht > Werkzeuge** aus, um über die Elemente zu verfügen.
2. Ziehen Sie ein Element in das Zustandsdiagramm-Fenster und lassen Sie es an geeigneter Stelle fallen.

	Startzustand [▶ 64] (Start state)
	Endzustand [▶ 65] (End state)
	Zustand [▶ 65] (State)
	Zusammengesetzter Zustand [▶ 69] (Composite)
	Gabelung/Verbindung [▶ 79] (Fork/Join)
	Auswahl [▶ 81] (Choice)
	Transition [▶ 82] (Transition)
	Abschlusstransition [▶ 85] (Completion transition)
	Ausnahmetransition [▶ 86] (Exception transition)
	Notiz [▶ 90] (Note)

7.4.1 Startzustand

Ein Startzustand ist ein Pseudozustand, der den initialen Zustand markiert. Befindet er sich auf oberster Ebene, startet damit der Ablauf des Zustandsdiagramms. Befindet er sich in einem zusammengesetzten Zustand oder im Bereich eines orthogonalen Zustands, dann startet damit der Ablauf des verschachtelten Zustandsdiagramms.



Der Startzustand ist als schwarzer gefüllter Kreis dargestellt.

Eigenschaften

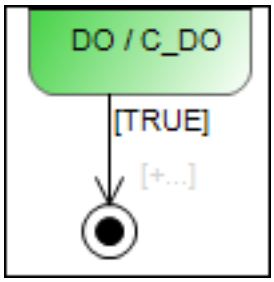
„Eigenschaft“	Beschreibung
„Bezeichner“	Hier können Sie einen Namen eingeben. Er wird im Zustandsdiagramm nicht angezeigt.

Startzustand editieren

Benutzereingabe im Zustands-diagramm	Reaktion im Zustands-diagramm	Beschreibung
Fokussieren Sie einen Startzustand.		Der Startzustand ist editierbar. • Sie können über das Befehlsicon, das oberhalb des Startzustands angezeigt wird, eine Abschlusstransitionen hinzufügen.
Klicken Sie auf das Symbol 		Eine Abschlusstransition wird hinzugefügt. Wenn Sie dabei auf einen bestehenden Zustand klicken, wird dieser zum Zielzustand der Transition. Wenn Sie auf einen freien Bereich klicken, wird ein neuer Zustand erzeugt.

7.4.2 Endzustand

Ein Endzustand ist ein Pseudozustand, der anzeigt, dass die Ausführung des Bereichs, der diesen Endzustand enthält, beendet ist. Endzustände können in einem Zustandsdiagramm an oberster Position, in zusammengesetzten Zuständen oder in Bereichen orthogonaler Zustände positioniert sein.



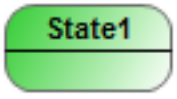
Ein Endzustand wird als kleiner schwarzer gefüllter Kreis innerhalb eines zweiten, leeren Kreises dargestellt. Ein Endzustand verfügt über keine Befehlsicons, um das Zustandsdiagramm zu editieren.

Eigenschaften

„Eigenschaft“	Beschreibung
„Bezeichner“	Hier können Sie einen Namen eingeben. Er wird im Zustandsdiagramm nicht angezeigt.

7.4.3 Zustand

Ein **Zustand** ist das Hauptelement eines Zustandsdiagramms. Ein Zustandsautomat (bzw. Zustandsdiagramm) durchläuft während seiner Laufzeit verschiedene Zustände und führt deren Aktionen aus. Ein Zustand kann über ENTRY-, DO-, und EXIT-Aktionen verfügen, die zu festgelegten Zeitpunkten in der Laufzeit des Zustands ausgeführt werden.



Ein Zustand wird als Rechteck mit abgerundeten Ecken dargestellt.

Normaler Zustand

Ein normaler Zustand ist entsprechend der Task, in der er aufgerufen wird, getaktet. Das heißt, dass der **Übergang** in den nächsten Zustand erst mit dem nächsten Taskzyklus geschaltet wird.

Zyklusinterner Zustand

Besteht ein Zustandsautomat aus zyklusinternen Zuständen, ist das Schaltverhalten unabhängig vom Taskzyklus. Das bedeutet, dass, wenn die Aktionen eines internen Zustands abgeschlossen sind, unmittelbar in die Transition geschaltet wird. Es wird damit sofort die Transitionsbedingung geprüft und bei erfüllter Bedingung die Transitionsaktion ausgeführt. Ebenso wird bei erfüllter Transitionsbedingung sofort anschließend in den Zielzustand geschaltet.

Ob ein Zustand zyklusintern schaltet, wird in der Eigenschaft „Interner Zustand“ eingestellt.

ENTRY-, DO-, EXIT-Aktionen/-Methoden

Ein Zustand kann eine ENTRY-, eine DO- und/oder eine EXIT-Aktion bzw. -Methode haben. Für diese Aktionen bzw. Methoden können Sie eine beliebige Implementierungssprache wählen.

- Die ENTRY-Aktion kann den Zustand initialisieren. Sie wird einmal ausgeführt, wenn alle eingehenden Transitionen schalten, so dass der Zustand aktiv wird.
- Die DO-Aktion wird solange ausgeführt, wie der Zustand aktiv ist.
- Die EXIT-Aktion soll sicherstellen, dass der Zustand in einem gültigen Zustand verlassen wird. Die EXIT-Aktion wird einmal ausgeführt, wenn alle ausgehenden Transitionen schalten, so dass der Zustand verlassen wird.

Aufrufverhalten

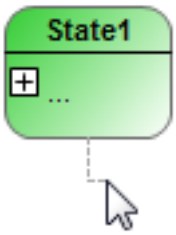
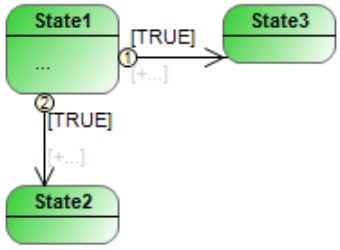
Bitte beachten Sie insbesondere die Samples [▶ 97], die u.a. das Aufrufverhalten vom UML-Zustandsdiagramm beschreiben und dieses anhand eines Beispiels darstellen.

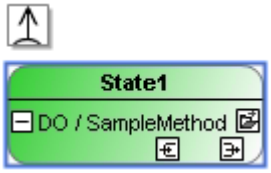
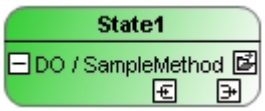
Eigenschaften

„Eigenschaft“	Beschreibung
„Bezeichner“	Name des Zustands Beispiel: Enable, Production, ErrorHandling
„Farbe“	Farbe des Zustands Klicken Sie auf die eingestellte Farbe, um über das sich öffnende Dropdown-Menü die Farbe des Zustands zu ändern. Voreinstellung: 50, 205, 50
„Impliziter Startzustand“	• Nein: [Default] Setzen Sie die Eigenschaft auf Nein, wenn Sie einen normalen Zustand benötigen. • Diagramm: Setzen Sie die Eigenschaft auf Diagramm, wenn Sie wollen, dass der Zustand zusätzlich ein Startzustand ist. Ein Zustandsautomat benötigt auf oberster Ebene einen Startzustand und auch jede Region in einem zusammengesetzten Zustand.
„Zyklusintern“	• : Aktivieren Sie die Checkbox, um den Zustandszyklus auf intern zu setzen. In „Zyklusintern“ wird sofort die Transition in den nächsten Zustand geschaltet. • : [Default] Deaktivieren Sie die Checkbox, um den Zustandszyklus auf normal zu setzen. Dann wechselt der Zustand mit dem Taskzyklus. Eine Transition in den nächsten Zustand wird bei einem Zykluswechsel der Task ausgeführt.
„Max. DO-Zyklus-Aufrufe “	Voraussetzung: Diese Eigenschaft ist verfügbar, wenn die Eigenschaft „Zyklusintern“ aktiviert ist. Definieren Sie die maximale Anzahl an Aufrufen der DO-Aktion pro Zyklus. Geben Sie eine Zahl zwischen 1 und 32767 an. Diese Einstellung ist für zyklusinterne Zustände relevant, wenn die ausgehende Transition des zyklusinternen Zustands während der Abarbeitung des zyklusinternen Zustands nicht erfüllt ist und sich während der DO-Aufrufe auch nicht erfüllen wird. Durch die eingestellte maximale Aufrufanzahl ist sichergestellt, dass es bei dem Aufruf des zyklusinternen Zustands nicht zu einer Endlosschleife kommt.
„ENTRY-Aktion“	Weisen Sie dem selektierten Zustand eine Aktion zu, indem Sie deren Aktionsnamen angeben.
„DO-Aktion“	
„EXIT-Aktion“	

Zustand editieren

Die folgenden Benutzereingaben stehen unter der Bedingung zur Verfügung, dass „Zeiger“ bzw. „Auswahl“ in „Werkzeuge“ aktiviert ist (Standardmodus).

Benutzereingabe im Zustandsdiagramm	Reaktion im Zustandsdiagramm	Beschreibung
Fokussieren Sie einen Zustand.		Der Zustand ist editierbar. • Sie können den Namen durch zwei Einzelklicks auf den Namen bearbeiten. • Sie können über das Befehlsicon, das oberhalb des Zustands angezeigt wird, ausgehende Transitionen hinzufügen. • Über die drei Icons in der Zustandsmitte können Sie den Zustand um eine ENTRY-, DO- und/oder EXIT-Aktion erweitern. • Sie können den Zustand mit der Taste [Entf] entfernen.
Klicken Sie auf das Symbol 		Eine abgehende Transition wird erzeugt. Wenn Sie dabei auf einen bestehenden Zustand klicken, wird dieser zum Zielzustand der Transition. Wenn Sie auf einen freien Bereich klicken, wird ein neuer Zustand erzeugt.
Erzeugen Sie mehrere Transitionen auf demselben Zustand.		Wenn ein Zustand mehr als eine abgehende/eingehende Transition hat, definieren deren Prioritäten die Reihenfolge der Ausführung. Die Priorität der Transitionen ist in einem kleinen Kreis dargestellt.
Klicken Sie auf  :		Der Zustand wird um eine ENTRY-Aktion/-Methode erweitert.
Klicken Sie auf  :		Der Zustand wird um eine DO-Aktion/-Methode erweitert.
Klicken Sie auf  :		Der Zustand wird um eine EXIT-Aktion/-Methode erweitert.
Fokussieren Sie einen Zustand, der um eine Aktion erweitert wurde, und klicken Sie auf das Symbol  .		Das Symbol ist bei Aktionszeilen am Ende der Zeile sichtbar, wenn bislang keine Aktion zugewiesen wurde. Der Dialog „Neues Aktionsobjekt“ öffnet, um eine neue Aktion zu erzeugen. Danach wird der Name der Aktion hinter einem Schrägstrich angezeigt. Siehe auch: „Neues Aktionsobjekt hinzufügen“ [▶ 59]

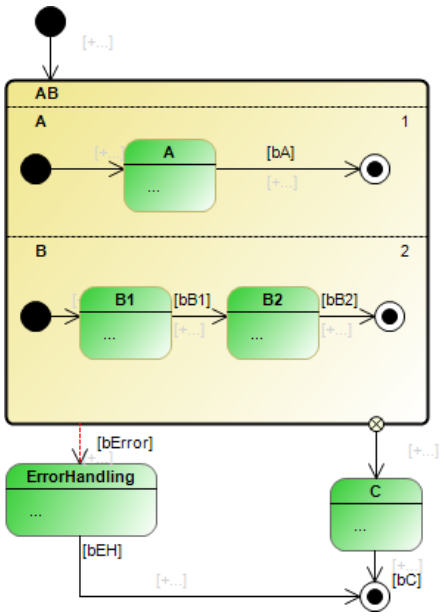
Benutzereingabe im Zustandsdiagramm	Reaktion im Zustandsdiagramm	Beschreibung
Klicken Sie mit zwei Einzelklicks auf den ENTRY-, DO- oder EXIT-Ausdruck (unabhängig davon, ob bereits eine Aktion zugewiesen wurde oder nicht).		Der Zeileneditor öffnet mit IntelliSense-Unterstützung, so dass eine neue Aktion zugewiesen werden kann. Ist die gewünschte Aktion im IntelliSense ausgewählt, kann die Aktion mittels Doppelklick oder mittels Selektieren plus [Enter] ausgewählt werden.
Fokussieren Sie einen Zustand, dem eine Aktion zugewiesen wurde. Klicken Sie auf das Symbol  oder doppelklicken Sie auf die zugewiesene Aktion.		Die zugewiesene Aktion, im Beispiel „SampleMethod“, wird im Editor geöffnet. Im aufgehenden POU-Editor können Sie das Aktionsobjekt editieren.
Klicken Sie auf  :		Die Aktionsliste des Zustands klappt zu.
Klicken Sie auf  :		Die Aktionsliste des Zustands klappt auf.

7.4.4 Zusammengesetzter Zustand

Ein **Zusammengesetzter Zustand** dient der Gruppierung der Zustände, die er umfasst, und kann für zwei verschiedene Anwendungsfälle eingesetzt werden:

- Gruppierung/Verschachtelung:
  - Wenn ein zusammengesetzter Zustand aus genau einer Region besteht, werden die inneren Zustände zur Laufzeit sequentiell durchlaufen. Dabei kann ein innerer Zustand wieder ein zusammengesetzter Zustand sein, sodass eine Verschachtelung von zusammengesetzten Zuständen entsteht. Zusätzlich können solche zusammengesetzten Zustände jeweils ENTRY/DO/EXIT-Aktionen aufrufen.
  - Die Verwendung von zusammengesetzten Zuständen mit jeweils einer Region dient zur Gruppierung von Zuständen, sodass beispielsweise mithilfe von (Pseudo-) Ausnahmetransitionen eine gemeinsame Weiterschaltung oder eine gemeinsame Fehlerbehandlung implementiert werden kann. Zusätzlich oder alternativ kann der zusammengesetzte Zustand mit eigenen ENTRY/DO/EXIT-Aktionen versehen werden, die den inneren Zuständen thematisch zugeordnet werden können. Dies ist beispielsweise interessant, wenn zu den DO-Aktionen der inneren Zustände zusätzlich eine übergeordnete DO-Aktion aufgerufen werden soll. Die übergeordnete DO-Aktion könnte in diesem Fall in dem zusammengesetzten Zustand platziert werden.
  - Falls mehrere zusammengesetzte Zustände ineinander verschachtelt werden, darf lediglich der innerste zusammengesetzte Zustände mehr als eine Region haben.
- Parallele Sub-Zustandsmaschinen:
  - Wenn ein zusammengesetzter Zustand aus mehreren Regionen besteht, werden deren innere Zustände orthogonal gruppiert. Die Regionen des zusammengesetzten Zustands werden mit Prioritäten versehen, die die Abarbeitungsreihenfolge zur Laufzeit regeln. Die Zustände der Regionen werden pseudo-parallel entsprechend ihrer internen Sequentialisierung durchlaufen.
  - Zusammengesetzte Zustände mit jeweils mehreren Regionen bzw. die sogenannten orthogonalen Zustände dienen zur Programmierung von parallelen Zuständen.

Beispielhafte Umsetzungen verschiedener Anwendungsfälle finden Sie unten auf dieser Seite.



Ein zusammengesetzter Zustand wird mit einem gelben, gefüllten Rechteck mit abgerundeten Ecken dargestellt. Sein Name wird in der oberen linken Ecke des Rechtecks angezeigt. Regionen sind durch eine gestrichelte, schwarze Linie abgeteilt.

Bei einem zusammengesetzten Zustand mit mehreren Regionen werden die Priorität jeder Region in der oberen rechten Ecke und der Name jeder Region in der oberen linken Ecke der Region angezeigt. Die Linien, Namen und Prioritäten sind editierbar.

Syntaxregeln

Allgemeine Syntaxregeln für zusammengesetzte Zustände

- Ein zusammengesetzter Zustand kann eine oder mehrere abgehende Ausnahmetransitionen besitzen. Mithilfe einer Ausnahmetransition können Sie beispielsweise eine Fehlerbehandlung implementieren.
- Abschlusstransition
  - Ein zusammengesetzter Zustand hat höchstens eine abgehende Abschlusstransition.
  - Wenn ein zusammengesetzter Zustand über eine abgehende Abschlusstransition verfügt, muss jede Region einen Start- und einen Endzustand enthalten.
  - Wenn der zusammengesetzte Zustand keine abgehende Abschlusstransition hat, können die Regionen ohne Endzustand sein.
  - Wenn allerdings alle Regionen des zusammengesetzten Zustands einen Endzustand enthalten, muss der zusammengesetzte Zustand über eine abgehende Abschlusstransition verfügen.
  - Der zusammengesetzte Zustand erreicht seinen Endzustand, wenn alle Regionen ihren Endzustand erreicht haben.

Syntaxregeln für einen zusammengesetzten Zustand mit genau einer Region

- Startzustand/Zusammengesetzten Zustand aktivieren
  - Wenn der zusammengesetzte Zustand über eine abgehende Abschlusstransition verfügt, muss die Region einen Start- und einen Endzustand enthalten (bzgl. Endzustand sehen Sie bitte auch die allgemeinen Syntaxregeln) und die eingehende Transition, um den zusammengesetzten Zustand zu aktivieren, muss mit dem zusammengesetzten Zustand verbunden werden.
  - Wenn der zusammengesetzte Zustand keine abgehende Abschlusstransition besitzt, kann die Region optional einen Startzustand enthalten.
    Falls ein Startzustand vorhanden ist, muss die eingehende Transition, um den zusammengesetzten Zustand zu aktivieren, mit dem zusammengesetzten Zustand verbunden werden. Eine direkte Verbindung über eine bedingte Transition zwischen einem Zustand innerhalb des zusammengesetzten Zustands und einem Zustand außerhalb des zusammengesetzten Zustands ist nicht erlaubt – weder von außen nach innen noch von innen nach außen.

Falls kein Startzustand vorhanden ist, werden die eingehenden Transitionen hingegen direkt mit den inneren Zuständen verbunden. Dies können eine oder mehrere eingehende Transitionen sein. Des Weiteren ist in diesem Fall auch die andere Richtung möglich: ausgehende bedingte Transitionen können ausgehend von den Zuständen innerhalb des zusammengesetzten Zustands direkt mit Zuständen außerhalb des zusammengesetzten Zustands verbunden werden. Dies können eine oder mehrere ausgehende Transitionen sein. Bitte beachten Sie, dass Zustände innerhalb und außerhalb eines zusammengesetzten Zustands nur dann direkt miteinander verbunden werden können (egal in welche Richtung), wenn den inneren und den äußeren Zustand nur eine Ebene eines zusammengesetzten Zustands trennt. D.h. ein Zustand, der sich innerhalb eines zusammengesetzten Zustands befindet, welcher sich wiederum in einem anderen zusammengesetzten Zustand befindet (Verschachtelung), kann nicht direkt mit einem Zustand verbunden werden, der sich außerhalb dieser Verschachtelung von zusammengesetzten Zuständen befindet. Diese Verbindung würde über zwei Grenzen von zusammengesetzten Zuständen hinweggehen, es ist aber nur eine Grenzüberschreitung möglich. Der genannte Zustand kann nur direkt mit einem außenliegenden Zustand verbunden werden, wenn sich dieser Zielzustand in dem äußeren zusammengesetzten Zustand befindet (nur eine Ebene überschritten).

- Verschachtelung
  - Ein zusammengesetzter Zustand, der über genau eine Region verfügt, kann einen anderen zusammengesetzten Zustand enthalten. Sie sind dann verschachtelt.
  - Die Verschachtelung von zusammengesetzten Zuständen kann beliebig tief sein, wobei lediglich der innerste zusammengesetzte Zustände mehr als eine Region haben darf.

Syntaxregeln für einen zusammengesetzten Zustand mit mehreren Regionen

- Startzustand/Zusammengesetzten Zustand aktivieren
  - Die eingehende Transition, um den zusammengesetzten Zustand zu aktivieren, muss mit dem zusammengesetzten Zustand verbunden werden.
  - Jede Region muss einen Startzustand enthalten.
- Eine Transition zwischen Zuständen, die in unterschiedlichen Regionen liegen, ist nicht erlaubt.
- Eine direkte Verbindung über eine bedingte Transition zwischen einem Zustand innerhalb des zusammengesetzten Zustands und einem Zustand außerhalb des zusammengesetzten Zustands ist nicht erlaubt – weder von außen nach innen noch von innen nach außen.
  Ausnahme: Mithilfe einer Gabelung, die sich außerhalb des zusammengesetzten Zustands befindet, können Sie Transitionen anlegen, die zu Zuständen innerhalb des zusammengesetzten Zustands gehen.

ENTRY-, DO-, EXIT-Aktionen/-Methoden

Ein zusammengesetzter Zustand, der über genau eine Region verfügt, kann ENTRY/DO/EXIT-Aktionen/-Methoden zugewiesen bekommen.

Um diese Funktionalität freizuschalten, muss die Option „ENTRY/DO/EXIT Aktionen erlauben“, die in den Eigenschaften des zusammengesetzten Zustands zu finden ist, aktiviert sein. Für die Aktionen bzw. Methoden können Sie eine beliebige Implementierungssprache wählen.

- Die ENTRY-Aktion kann den zusammengesetzten Zustand initialisieren. Sie wird einmal ausgeführt, wenn alle eingehenden Transitionen schalten bzw. wenn ein innenliegender Zustand aktiv wird, sodass der zusammengesetzte Zustand aktiviert wird.
- Beachten Sie die Beschreibung der Eigenschaft „DO-Aktionen auch ausführen, wenn die inneren zusammengesetzten Zustände aktiv sind“, die das Verhalten der DO-Aktion festlegt.
- Die EXIT-Aktion soll sicherstellen, dass der zusammengesetzte Zustand in einem gültigen Zustand verlassen wird. Die EXIT-Aktion wird einmal ausgeführt, wenn alle ausgehenden Transitionen schalten bzw. wenn ein außenliegender Zustand aktiv wird, sodass der zusammengesetzte Zustand verlassen wird.

Aufrufverhalten

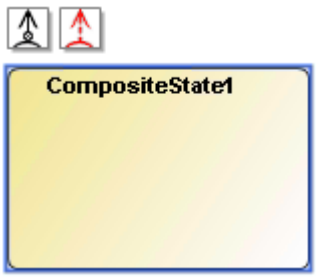
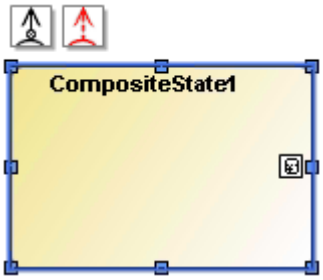
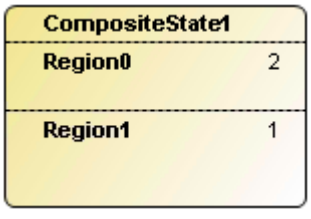
Bitte beachten Sie insbesondere die Samples [▶ 97], die u.a. das Aufrufverhalten vom UML-Zustandsdiagramm beschreiben und dieses anhand eines Beispiels darstellen.

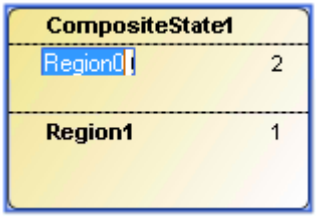
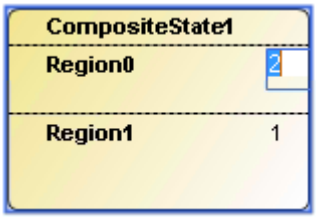
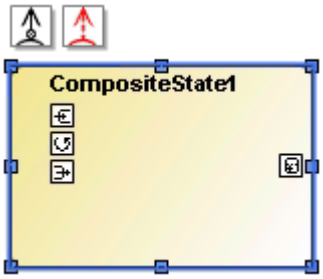
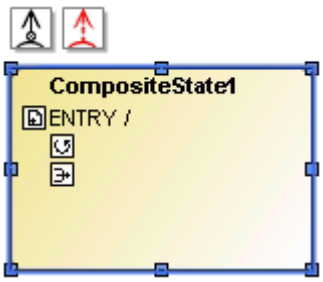
Eigenschaften

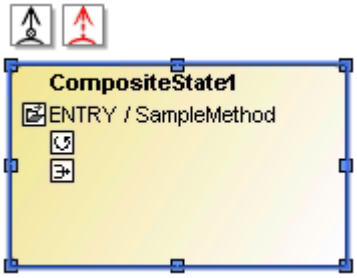
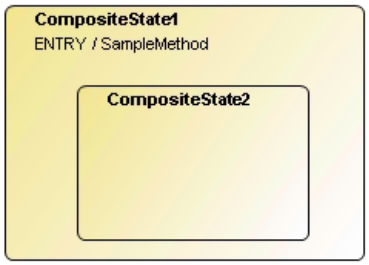
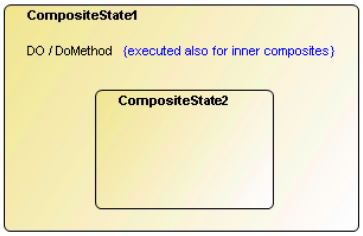
„Eigenschaft“	Beschreibung
„Bezeichner“	Name des zusammengesetzten Zustands Beispiel: DoorAutomation
„Farbe“	Farbe des zusammengesetzten Zustands Klicken Sie auf die eingestellte Farbe, um über das sich öffnende Dropdown-Menü die Farbe des zusammengesetzten Zustands zu ändern. Voreinstellung: 240, 230, 140
„ENTRY/DO/EXIT Aktionen erlauben“	Voraussetzung: Diese Eigenschaft ist verfügbar, wenn der selektierte zusammengesetzte Zustand genau eine Region hat. In diesem Fall können Sie einem zusammengesetzten Zustand auch Aktionen zuweisen, wenn er Teil einer Verschachtelung von zusammengesetzten Zuständen ist. ☑ : Sie können dem selektierten zusammengesetzten Zustand eine ENTRY-, DO- oder EXIT-Aktion zuweisen. ☐ : Sie können dem selektierten zusammengesetzten Zustand keine eigenen ENTRY- DO- oder EXIT-Aktionen zuweisen.
„DO-Aktionen auch ausführen, wenn die inneren zusammengesetzten Zustände aktiv sind“	Voraussetzung: Mehrere zusammengesetzte Zustände sind grafisch ineinander verschachtelt. Die Option ist nur beim äußersten zusammengesetzten Zustand verfügbar und wird an die inneren vererbt. ☑ : Zur Laufzeit wird die DO-Aktion des äußeren zusammengesetzten Zustands ständig ausgeführt, auch wenn ein innerer zusammengesetzter Zustand aktiv ist. Im Editor erscheint neben der DO-Aktion der Hinweis „{wird auch für innere zusammengesetzte Zustände ausgeführt}“, um auf dieses Verhalten der Aktion hinzuweisen. ☐ : Sobald einer der inneren zusammengesetzten Zustände aktiv ist, pausiert die DO-Aktion des äußeren zusammengesetzten Zustands.
„ENTRY-Aktion“	Voraussetzung: Der selektierte zusammengesetzte Zustand hat genau eine Region und die Eigenschaft „ENTRY/DO/EXIT Aktionen erlauben“ ist aktiviert. Weisen Sie dem selektierten Zustand eine Aktion zu, indem Sie deren Aktionsnamen angeben.
„DO-Aktion“	
„EXIT-Aktion“	

Zusammengesetzten Zustand editieren

Die Benutzereingaben im Zustandsdiagrammeditor können wie folgt zusammengefasst werden:

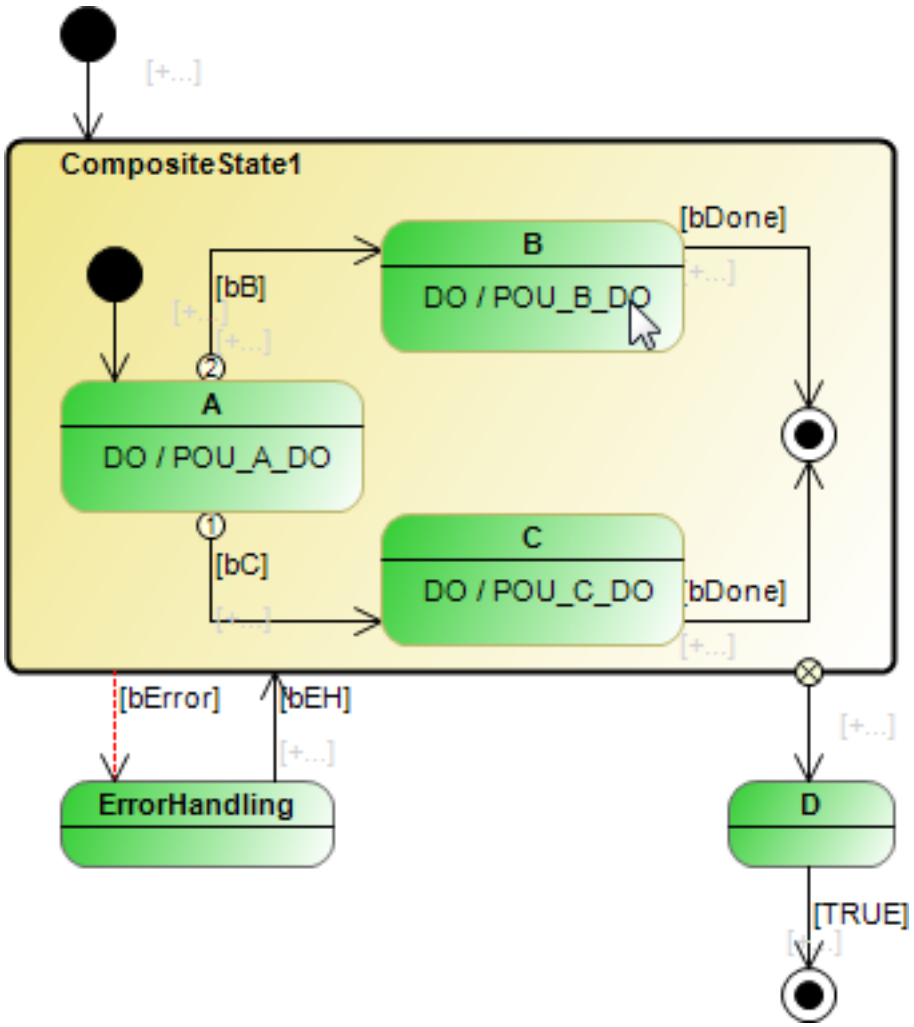
Benutzereingabe im Zustandsdiagrammeditor	Reaktion im Zustandsdiagramm	Beschreibung
Fokussieren Sie einen zusammengesetzten Zustand.		Der zusammengesetzte Zustand ist editierbar. • Sie können den Namen durch zwei Einzelklicks auf den Namen bearbeiten. • Sie können über die Befehlsicons, die oberhalb des Zustands angezeigt werden, ausgehende Transitionen hinzufügen. • Sie können die Größe des zusammengesetzten Zustands anpassen. • Sie können den Zustand mit der Taste [Entf] entfernen. Bedingte Editierungsmöglichkeiten: • Sie können den zusammengesetzten Zustand um eine ENTRY-, DO- und/ oder EXIT-Aktion erweitern, falls der zusammengesetzte Zustand eine Region hat und falls die Eigenschaft „ENTRY / DO / EXIT Aktionen erlauben“ aktiviert ist.
Klicken Sie auf das Symbol 		Eine Abschlusstransition wird hinzugefügt. Wenn Sie dabei auf einen bestehenden Zustand klicken, wird dieser zum Zielzustand der Transition. Wenn Sie auf einen freien Bereich klicken, wird ein neuer Zustand erzeugt.
Klicken Sie auf das Symbol 		Eine Ausnahmetransition wird hinzugefügt. Wenn Sie dabei auf einen bestehenden Zustand klicken, wird dieser zum Zielzustand der Transition. Wenn Sie auf einen freien Bereich klicken, wird ein neuer Zustand erzeugt.
Fokussieren Sie einen zusammengesetzten Zustand und halten Sie den Mauszeiger über den Zustand.		
Ziehen Sie eines der blauen Quadrate  auf eine andere Position.		Die Größe des zusammengesetzten Zustands wurde angepasst.
Klicken Sie auf das Symbol 		Der Zustand wird unterteilt und eine weitere Region wird hinzugefügt. Der Name und die Priorität einer Region werden pro Region angezeigt.

Benutzereingabe im Zustandsdiagrammeditor	Reaktion im Zustandsdiagramm	Beschreibung
Klicken Sie mit zwei Einzelklicks auf den Namen einer Region.		Der Name der Region ist editierbar.
Klicken Sie mit zwei Einzelklicks auf die Priorität einer Region.		Die Priorität der Region ist editierbar. Geben Sie eine Zahl ein, um die Priorität der Region zu definieren. Die Prioritäten der anderen Regionen werden automatisch angepasst.
Klicken Sie auf die Trennlinie und verschieben Sie diese.		Die Trennlinie wird verschoben, damit wird die Größe der Regionen angepasst.
Klicken Sie auf die Trennlinie und drücken Sie die Taste [Entf].		Die Trennlinie wird entfernt, dadurch werden die zwei von der Trennlinie getrennten Regionen zu einer Region.
Fokussieren Sie einen zusammengesetzten Zustand und halten Sie den Mauszeiger über den Zustand.		Voraussetzungen: • Der zusammengesetzte Zustand hat eine Region. • Die Eigenschaft „ENTRY / DO / EXIT Aktionen erlauben“ ist aktiviert
Klicken Sie auf eines der drei Symbole 		Diese Befehlsicons erweitern einen zusammengesetzten Zustand um eine ENTRY-, DO- oder EXIT-Aktion. Wenn auf eines der Rechtecke geklickt wird, erscheint: • ENTRY / : Erweiterung um ENTRY-Aktion • DO / : Erweiterung um DO-Aktion • EXIT / : Erweiterung um EXIT-Aktion
Fokussieren Sie einen Zustand, der um eine Aktion erweitert wurde, und klicken Sie auf das Symbol  .		Das Symbol ist bei Aktionszeilen am Ende der Zeile sichtbar, wenn bislang keine Aktion zugewiesen wurde. Der Dialog „Neues Aktionsobjekt“ öffnet, um eine neue Aktion zu erzeugen. Danach wird der Name der Aktion hinter einem Schrägstrich angezeigt. Siehe auch: „Neues Aktionsobjekt hinzufügen“ [▶ 59]

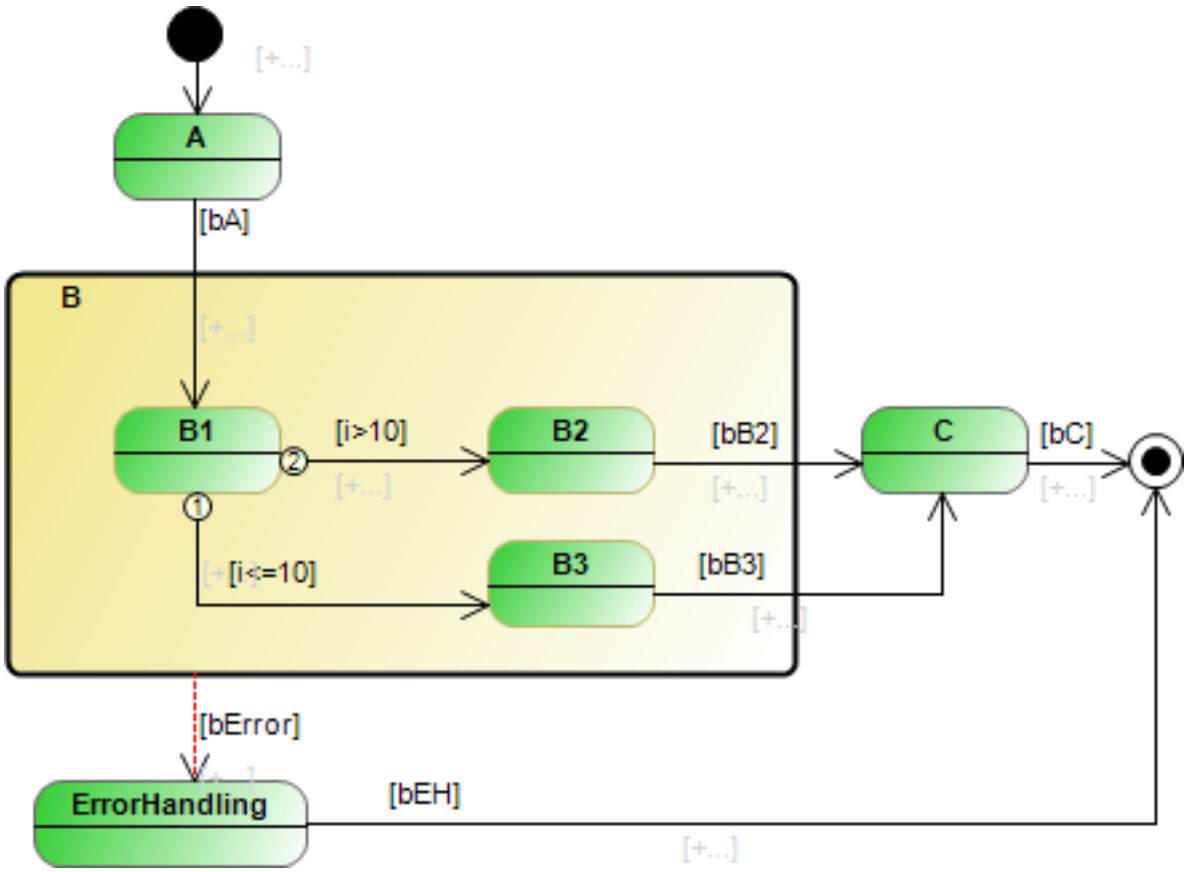
Benutzereingabe im Zustandsdiagrammeditor	Reaktion im Zustandsdiagramm	Beschreibung
Klicken Sie mit zwei Einzelklicks auf den ENTRY-, DO- oder EXIT-Ausdruck (unabhängig davon, ob bereits eine Aktion zugewiesen wurde oder nicht).		Der Zeileneditor öffnet mit IntelliSense-Unterstützung, sodass eine neue Aktion zugewiesen werden kann. Ist die gewünschte Aktion im IntelliSense ausgewählt, kann die Aktion mittels Doppelklick oder mittels Selektieren plus [Enter] ausgewählt werden.
Fokussieren Sie einen Zustand, dem eine Aktion zugewiesen wurde. Klicken Sie auf das Symbol  oder doppelklicken Sie auf die zugewiesene Aktion.		Die zugewiesene Aktion, im Beispiel „SampleMethod“, wird im Editor geöffnet. Im aufgehenden POU-Editor können Sie das Aktionsobjekt editieren.
Ziehen Sie einen zusammengesetzten Zustand von der Ansicht Werkzeuge auf einen zusammengesetzten Zustand, der über genau eine Region verfügt.		Die zusammengesetzten Zustände sind verschachtelt. Wenn der äußerste Zustand die Eigenschaft „ENTRY / DO / EXIT Aktionen erlauben“ aktiviert hat, können Sie jedem zusammengesetzten Zustand eigene Aktionen (ENTRY/ DO/EXIT) zuweisen.
Fügen Sie dem äußersten zusammengesetzten Zustand eine DO-Aktion hinzu und aktivieren Sie in der Ansicht Eigenschaften die Option „DO-Aktionen auch ausführen, wenn die inneren zusammengesetzten Zustände aktiv sind“.		Zur Laufzeit wird die DO-Aktion, im Beispiel „DoMethod“, ständig ausgeführt, auch wenn einer der inneren zusammengesetzten Zustände aktiv ist. Bei einer höheren Verschachtelungstiefe wird diese Option mit ihrem Wert an innere Zustände vererbt. Im Editor erscheint der Hinweis „{wird auch für innere zusammengesetzte Zustände ausgeführt}“.
Ziehen Sie einen Zustand vom Fenster „Werkzeuge“ in eine Region des zusammengesetzten Zustands.		Der Zustand wird dieser Region zugeordnet. Falls das Symbol  erscheint, ist die Einfügeposition, die Sie mit dem Mauszeiger fokussieren, nicht erlaubt.

Beispiele zusammengesetzter Zustand

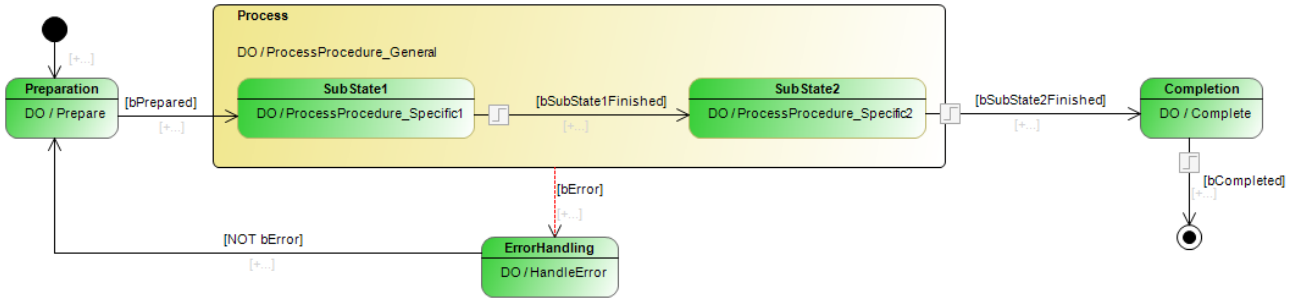
Zusammengesetzter Zustand mit Ausnahme- und Abschlusstransition und mit einer Region mit Start- und Endzustand:



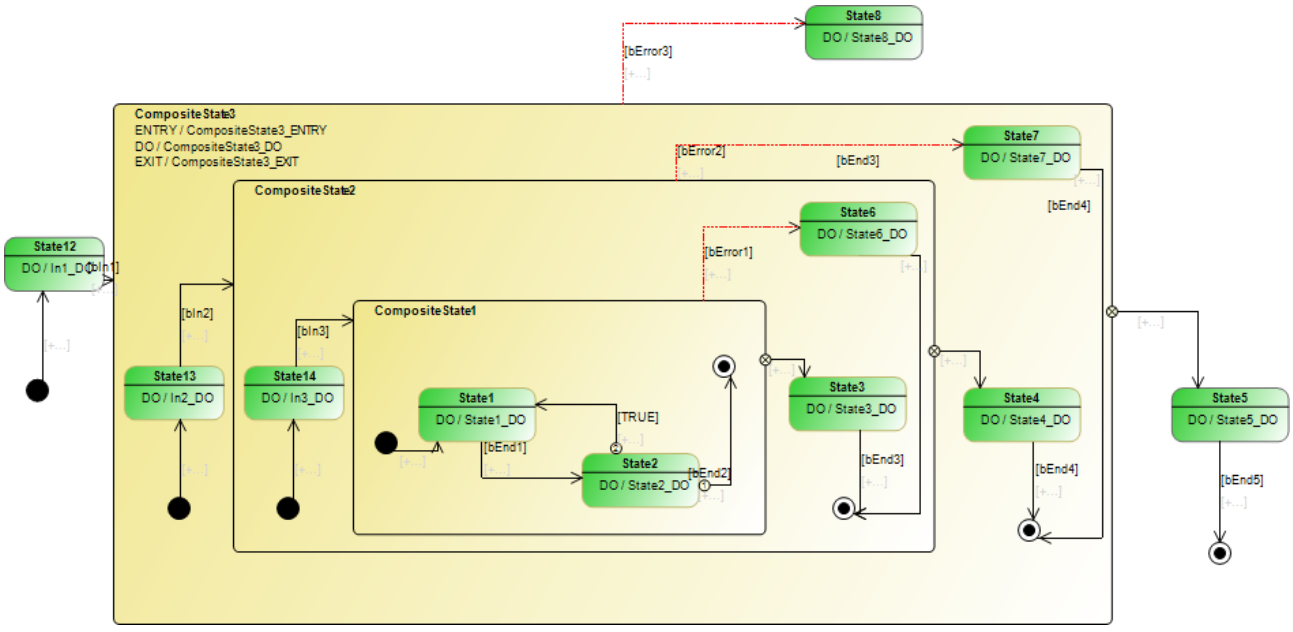
Zusammengesetzter Zustand ohne Abschlusstransition und mit einer Region ohne Start- und Endzustand:



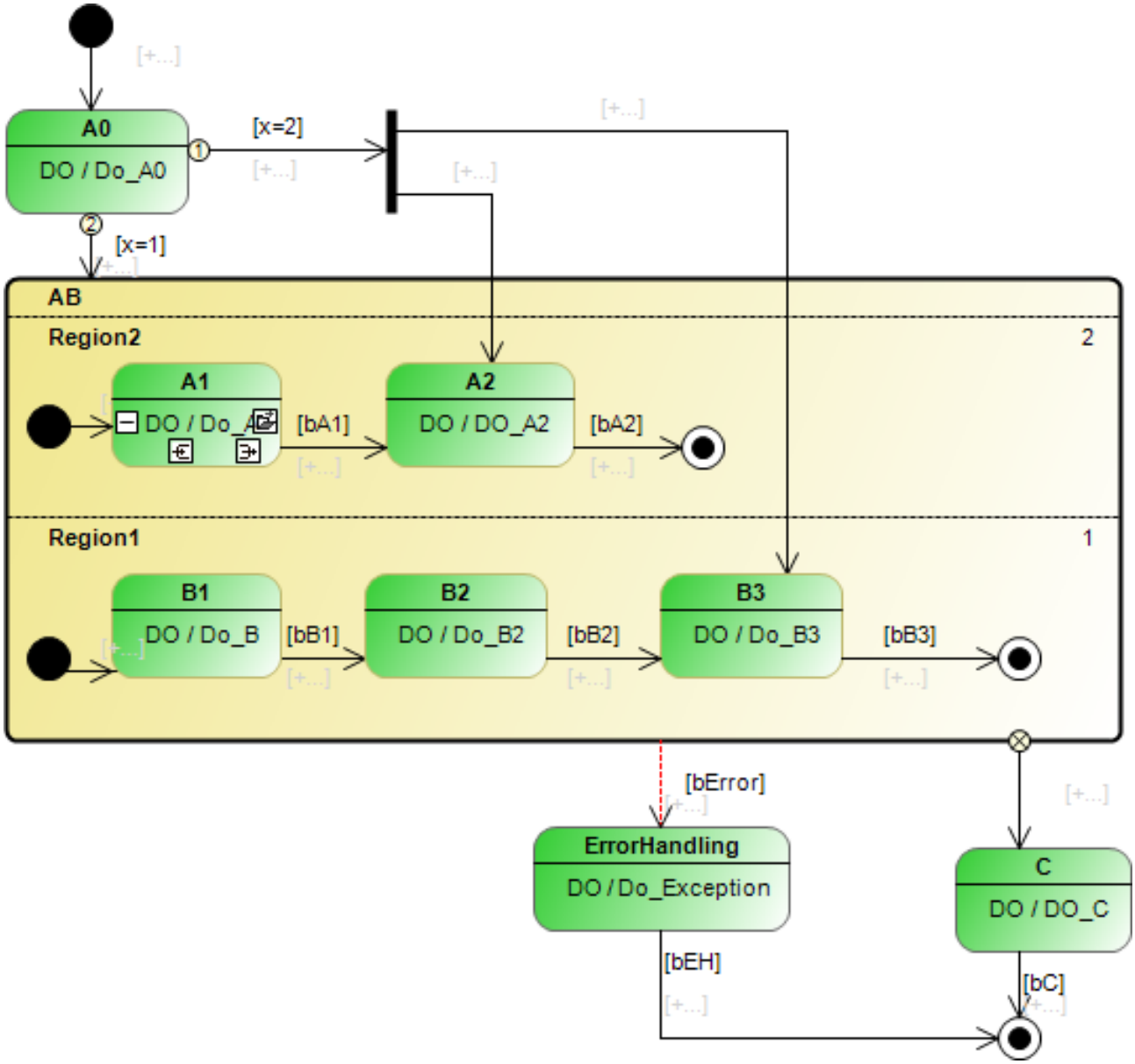
Zusammengesetzter Zustand mit einer Region und eigener DO-Aktion:



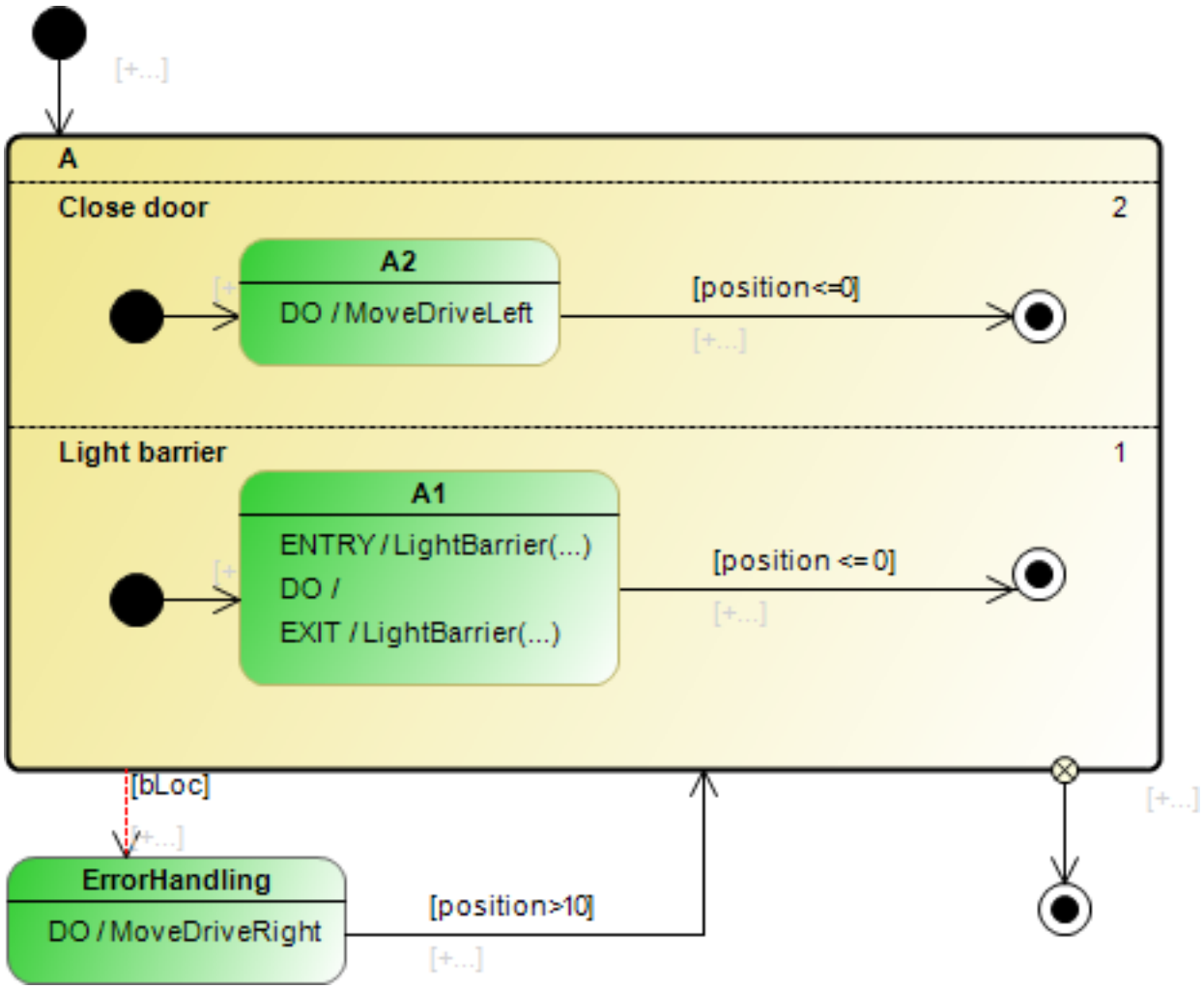
Verschachtelte zusammengesetzte Zustände, teilweise mit eigener ENTRY/DO/EXIT-Aktion:



Zusammengesetzter Zustand mit mehreren Regionen/Orthogonaler Zustand mit Gabelung:



Beispiel „Lift“: Orthogonaler Zustand mit Abschluss- und Ausnahmetransition:

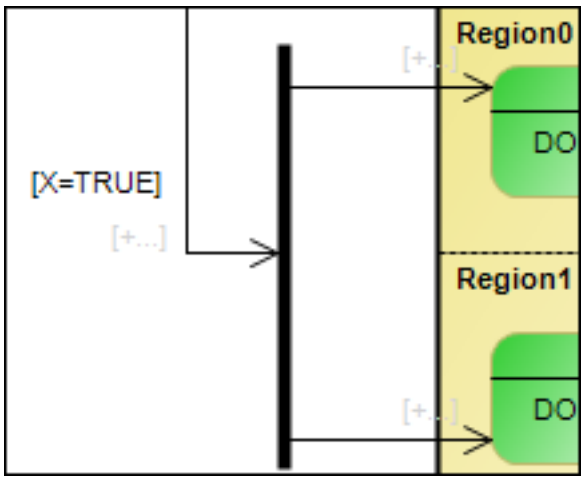


7.4.5 Gabelung

Gabelung ist ein Pseudostatus, um Transitionen zu gabeln.

Eine Gabelung kann eine oder mehrere eingehende Transitionen besitzen. Sie verfügt über mehrere ausgehende Transitionen, die in verschiedenen Regionen eines zusammengesetzten/orthogonalen Zustands enden müssen. Die Gabelung muss die gleiche Anzahl an ausgehenden Transitionen haben, wie es Regionen in dem zusammengesetzten/orthogonalen Zustand gibt.

Alle Transitionen, die von einer Gabelung ausgehen, sind Abschlusstransitionen. Sie sehen wie normale Transitionen aus, da sie im Gegensatz zu Abschlusstransitionen nicht mit einem kleinen, mit Kreuz versehenen Kreis dargestellt werden. Sie sind jedoch bedingungslos und sie geben nach einem abschließenden Ereignis ein Signal.

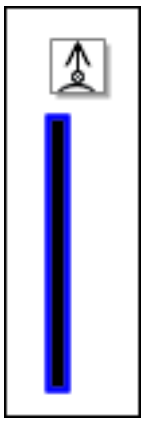


Eine Gabelung wird als schwarzer Balken dargestellt, horizontal oder vertikal ausgerichtet.

Eigenschaften

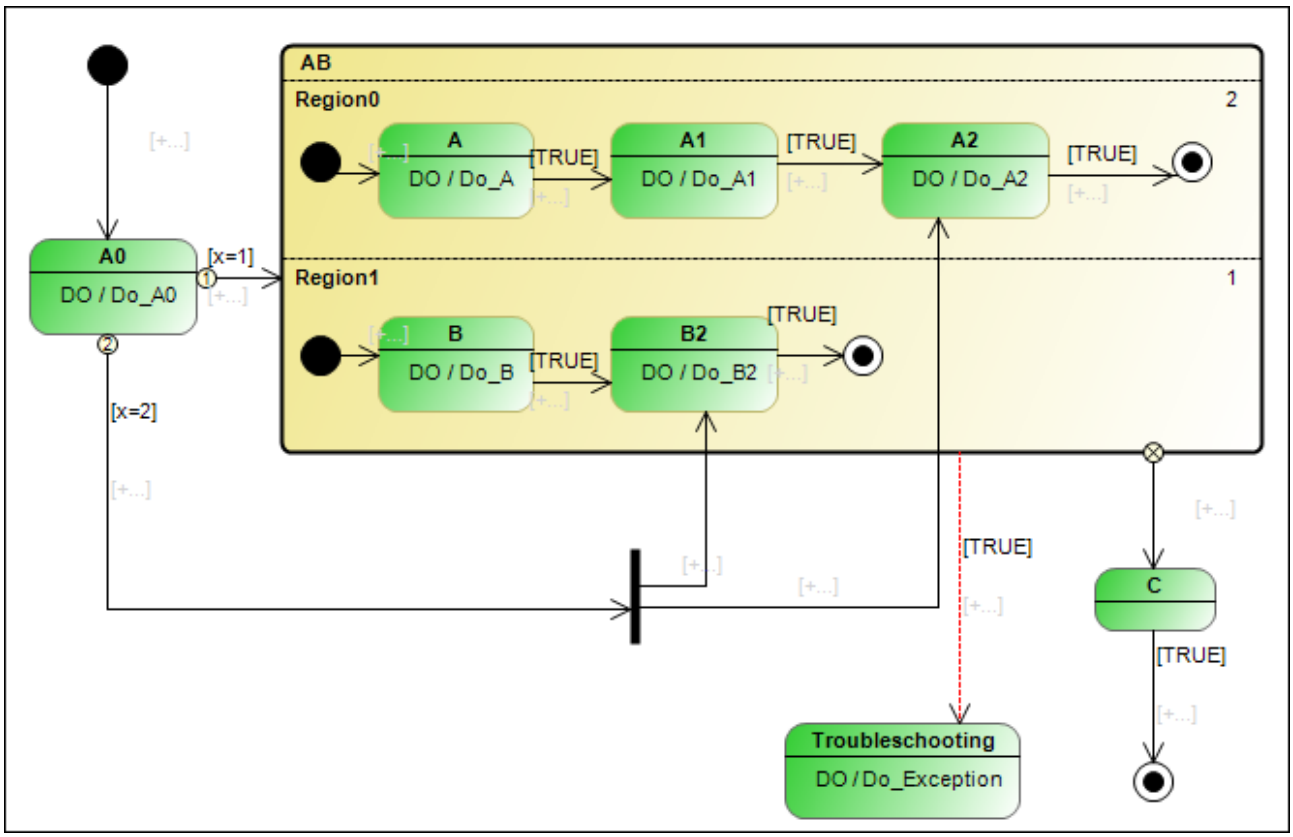
„Eigenschaft“	Beschreibung
„Bezeichner“	Hier können Sie einen Namen eingeben. Er wird im Zustandsdiagramm nicht angezeigt.
„Richtung vertikal“	• : [Default] Aktivieren Sie die Checkbox, um den Balken vertikal auszurichten. • : Deaktivieren Sie die Checkbox, um den Balken horizontal auszurichten.

Gabelung editieren

Benutzereingabe im Zustands-diagramm	Reaktion im Zustandsdiagramm	Beschreibung
Fokussieren Sie eine Gabelung.		Die Gabelung ist editierbar. • Sie können die Größe der Gabelung anpassen. • Sie können über das Befehlsicon, das oberhalb der Gabelung angezeigt wird, eine Abschlusstransition hinzufügen.
Klicken Sie auf das Symbol 		Eine Abschlusstransition wird hinzugefügt. Wenn Sie dabei auf einen bestehenden Zustand klicken, wird dieser zum Zielzustand der Transition. Wenn Sie auf einen freien Bereich klicken, wird ein neuer Zustand erzeugt.
Ziehen Sie eines der blauen Quadrate auf eine andere Position.		Die Größe der Gabelung wurde angepasst.

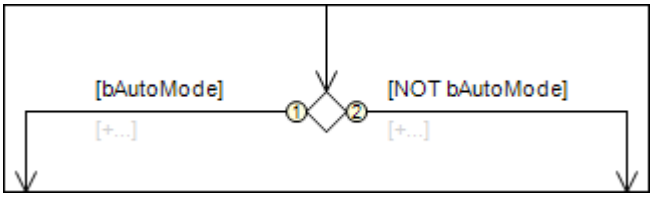
Beispiel

Gabelung mit orthogonalen Zuständen



7.4.6 Auswahl

Eine **Auswahl** ist ein Knoten in einer Zustandsmaschine. An dem Knoten wird innerhalb eines Zyklus eine Evaluierung der nachfolgenden Überwachungsbedingungen durchgeführt. Sie hat mindestens eine eingehende und eine ausgehende Transition. Auswahlknoten können in einem Zustandsdiagramm an oberster Position, in zusammengesetzten Zuständen oder in Bereichen orthogonaler Zustände positioniert sein. Wenn mehrere Auswahlknoten durch Transitionen verbunden sind, dürfen keine zirkulären Verknüpfungen entstehen, wie beispielsweise eine Rückführung auf eine Auswahl, die bereits Quellelement für eine Transition innerhalb dieser Kette ist.

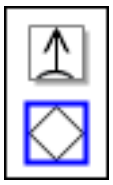


Eine Auswahl wird als Raute dargestellt. Der Kreis mit der Ablaufkontrollnummer gibt Auskunft über die Reihenfolge der Abarbeitung.

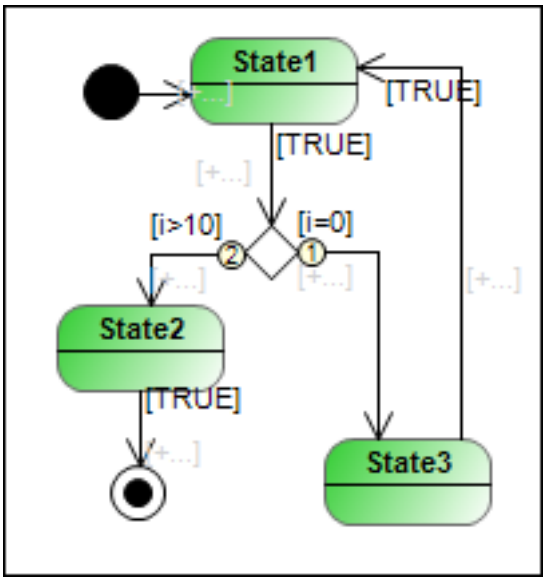
Eigenschaften

„Eigenschaft“	Beschreibung
„Bezeichner“	Hier können Sie einen Namen eingeben. Er wird im Zustandsdiagramm nicht angezeigt.

Auswahl editieren

Benutzereingabe im Zustands-diagramm	Reaktion im Zustandsdiagramm	Beschreibung
Fokussieren Sie eine Auswahl.		Die Auswahl ist editierbar. • Sie können über das Befehlsicon, das oberhalb der Auswahl angezeigt wird, ausgehende Transitionen hinzufügen.
Klicken Sie auf das Symbol 		Eine abgehende Transition wird erzeugt. Wenn Sie dabei auf einen bestehenden Zustand klicken, wird dieser zum Zielzustand der Transition. Wenn Sie auf einen freien Bereich klicken, wird ein neuer Zustand erzeugt.

Beispiel



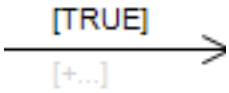
7.4.7 Transition

Eine **Transition** regelt das Übergangsverhalten zwischen Zuständen. Eine Transition ist möglich, wenn eines der folgenden Ereignisse eintritt:

- bedingtes Ereignis oder Änderungsereignis
- Beendigungsereignis (wenn die Aktionen des Quellzustands abgeschlossen sind)
- Zeitereignis

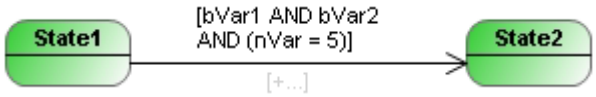
Eine Transition hat eine Wächterbedingung und optional eine Aktion. Üblicherweise wird der Zustandsübergang ausgeführt, wenn die Auswertung der Wächterbedingung eine steigende Flanke erkennt. Alternativ können Sie die Transition so konfigurieren, dass erkannt wird, ob die Wächterbedingung TRUE ist. In diesem Fall ist die Weiterschaltung unabhängig vom Flankenverhalten der Wächterbedingung. Bei der Ausführung des Zustandsübergangs mit dem nächsten Taktzyklus wird zunächst die Transitionsaktion ausgeführt und anschließend in den Zielzustand geschaltet.

Hat ein Quellzustand mehrere abgehende Transitionen, dann ist jede Transition mit einer Priorität versehen. Diese können Sie in den Elementeigenschaften ändern (Ansicht → „Eigenschaften“ → „Priorität“). Die Priorität bestimmt, in welcher Reihenfolge die Wächterbedingungen geprüft werden und damit in welcher Reihenfolge die Transitionen geschaltet werden.



Eine Transition wird als dünner Pfeil dargestellt, dessen Spitze zum nächsten Zustand zeigt.

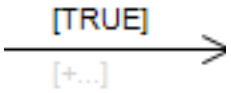
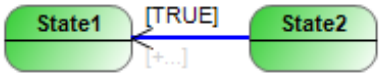
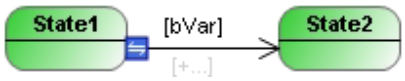
Die Wächterbedingung einer Transition kann aus mehreren booleschen Variablen bzw. Ausdrücken bestehen. Des Weiteren können Transitionselemente, die als separate Objekte zu einem POU-Objekt hinzugefügt werden können, als Transitionsbedingung verwendet werden. Um die einzelnen Bedingungen grafisch übersichtlicher darzustellen, kann mittels [Strg+Enter] eine neue Zeile in dem Zeileneditor der Transitionsbedingung eingefügt werden.



Eigenschaften

„Eigenschaft“	Beschreibung
„Beziehungstyp“	Transition (nicht editierbar)
„Priorität“	Priorität, die die Abarbeitungsreihenfolge festlegt Beispiel: 1 Hinweis: Wenn der Zustand weitere Transitionen hat und Sie die Priorität ändern, sind alle Transitionen von der Änderung betroffen und werden automatisch angepasst.
„Steigende Flanke“	•  : Wenn die Wächterbedingung eine steigende Flanke (von 0 auf 1) liefert, wird die Transition durchlaufen und der Zustandsübergang ausgeführt. Tipp: Im Editor ist eine Transition, die sich so verhält, mit dem Symbol  markiert. •  : Wenn die Wächterbedingung TRUE ist, wird die Transition durchlaufen. Tipp: Wenn die Wächterbedingung immer TRUE ist, wird die Transition einmal ausgeführt.

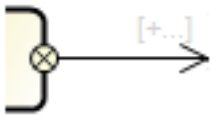
Transition editieren

Benutzereingabe im Zustands-diagramm	Reaktion im Zustandsdiagramm	Beschreibung
Selektieren Sie „Transition“ in „Werkzeuge“. Klicken Sie auf einen Zustand (Quellzustand) im Zustandsdiagramm und klicken Sie dann auf einen anderen Zustand (Zielzustand).		Eine Transition mit TRUE-Bedingung und einem Aktionssymbol wird erzeugt. Verwenden Sie dieses Verfahren, um eine Transition zwischen existierenden Zuständen zu erzeugen.
Klicken Sie auf einen Zustand (Quellzustand) im Zustandsdiagramm.		Das Icon, über das eine ausgehende Transition zu dem ausgewählten Zustand hinzugefügt werden kann, ist verfügbar. Wenn Sie es verwenden, wird der Zustand um eine ausgehende Transition erweitert. Der Zielzustand wird mit einem weiteren Klick bestimmt.
Doppelklick Sie auf das Symbol [TRUE]		Ein Eingabefeld mit IntelliSense-Funktionalität öffnet. Wählen Sie eine boolesche Variable oder einen booleschen Ausdruck aus. Ist die gewünschte Auswahl im IntelliSense selektiert, kann die Auswahl mittels Doppelklick oder mittels Fokussieren plus [Enter] ausgewählt werden.
Zwei Einzelklicks auf [+...] :		Ein Eingabefeld mit IntelliSense-Funktionalität öffnet. Wählen Sie eine Methode oder Aktion aus. Ist die gewünschte Auswahl im IntelliSense selektiert, kann die Auswahl mittels Doppelklick oder mittels Fokussieren plus [Enter] ausgewählt werden.
Klicken Sie einmal auf eine Transition, auf das Aktionssymbol oder auf das Wächtersymbol:		Die Transition ist selektiert (dargestellt durch einen blauen Pfeil). Eine selektierte Transition kann via Drag’n‘Drop verschoben werden. Die Position der verbundenen Zustände bleibt unverändert.
Klicken auf den Anfang oder das Ende einer Transition.		Das Symbol  ermöglicht ein erneutes Anschließen der Transition, ohne dass die bisherige Transitionskonfiguration verloren geht. Sie können das Symbol auf einen anderen Quell- bzw. Zielzustand ziehen, um die Transition dort anzuschließen. Die zugehörige Bedingung und Aktion bleiben erhalten.

7.4.8 Abschlusstransition

Eine **Abschlusstransition** ist bedingungslos. Das heißt, sie hat **keine** Wächterbedingung, die einen Schaltvorgang auslöst, sondern sie schaltet ohne zusätzliche Bedingung weiter, wenn der Quellzustand vollständig abgearbeitet ist. Während des nächsten Taskzyklus wird eine möglicherweise zugewiesene Aktion ausgeführt.

Alle ausgehenden Transitionen eines Startzustands und einer Gabelung/Verbindung sind Abschlusstransitionen. Ein zusammengesetzter Zustand kann ebenfalls über eine Abschlusstransition verfügen (je nach Anwendungsfall des zusammengesetzten Zustands).



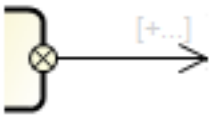
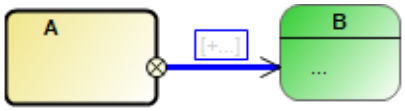
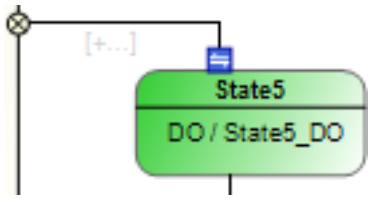
Eine Abschlusstransition ist als dünner Pfeil dargestellt, dessen Spitze zu seinem Zielzustand zeigt. Der kleine mit einem Kreuz versehene Kreis am Pfeilanfang zeigt an, dass es sich um eine Abschlusstransition handelt.

Eigenschaften

„Eigenschaft“	Beschreibung
„Beziehungstyp“	Abschlusstransition (nicht editierbar)
„Priorität“	1 (nicht editierbar)

Abschlusstransition editieren

Die Benutzereingaben im Zustandsdiagrammeditor können wie folgt zusammengefasst werden:

Benutzereingabe im Zustands-diagramm	Reaktion im Zustandsdiagramm	Beschreibung
Selektieren Sie „Abschlusstransition“ in der Werkzeugbox. Klicken Sie auf einen Zustand und danach auf einen anderen Zustand (Zielzustand).		Eine bedingungslose Transition mit einem Aktionssymbol wird angelegt. Diese Art, eine Transition zu erzeugen, wird verwendet, wenn die Transition bestehende Zustände verknüpfen soll.
Klicken Sie auf  .		Das Icon ist verfügbar, wenn der zugehörige Zustand selektiert ist. Wenn Sie es verwenden, wird der Zustand um eine ausgehende Abschlusstransition erweitert. Der Zielzustand wird mit einem weiteren Klick bestimmt.
Klicken Sie auf die Transition selbst oder auf das Aktionssymbol.		Die Transition ist selektiert (dargestellt durch einen blauen Pfeil). Eine selektierte Transition kann via Drag’n‘Drop verschoben werden. Die Position der verbundenen Zustände bleibt unverändert.
Klicken Sie zweimal auf [+...].		Ein Inline-Editor öffnet sich. Wenn Sie mit Tippen beginnen, erscheinen die passenden Komponenten in der IntelliSense-Auswahl. Sie können mit einem Doppelklick ein(e) Programm/ Methode/Aktion auswählen.
Klicken auf den Anfang oder das Ende einer Abschlusstransition.		Das Symbol  ermöglicht ein erneutes Anschließen der Transition, ohne dass die bisherige Transitionskonfiguration verloren geht. Sie können das Symbol auf einen anderen Quell- bzw. Zielzustand ziehen, um die Transition dort anzuschließen. Die zugehörige Aktion bleibt erhalten.

7.4.9 Ausnahmetransition

Eine **Ausnahmetransition** kontrolliert das Schalten in den nächsten Zustand oder Pseudozustand, wenn ein Fehler oder eine Ausnahme (Exception) auftritt. Sie hat eine Wächterbedingung und optional eine Aktion.

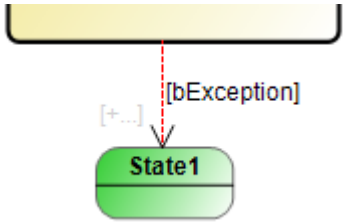
Der Quellzustand ist üblicherweise ein zusammengesetzter oder ein orthogonaler Zustand. Ausnahmetransitionen können nicht für „normale“ Zustände verwendet werden. Innerhalb des zusammengesetzten oder orthogonalen Zustands, der über eine Ausnahmetransition verfügt, ist jedoch i.d.R. ein „normaler“ Zustand aktiv.

Eine Ausnahmetransition beendet die gerade laufende Abarbeitung und schaltet in den Zustand, der für die Reaktion auf dieses Ereignis vorgesehen ist. In diesem Zustand sind Fehlerbehandlung und Ausnahmeverhalten definiert. Durch eine Ausnahme getriggert beendet die Ausnahmetransition den Prozess des gerade aktiven Zustands bzw. Unterzustands. Die Ausführung der aktuell ausgeführten DO-Aktion(en) wird abgeschlossen.

Die Ausnahmetransition wird verwendet, um einen zusammengesetzten Zustand von jedem Unterzustand aus zu verlassen. Ihr Zustand wird bewertet, **nachdem** die DO-Aktion für den aktiven Zustand ausgeführt wurde. Selbst wenn also die Bedingung für die Ausnahmetransition bereits beim ersten Eintritt eines zusammengesetzten Zustands TRUE ist, wird die DO-Aktion des ersten Zustands ausgeführt. Da ENTRY- und EXIT-Aktionen nicht an Bedingungen geknüpft sind, werden sie immer unabhängig von der Ausnahmetransition ausgeführt.

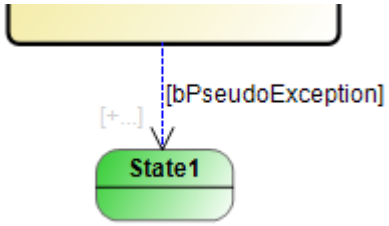
Sie können eine Ausnahmetransition dahingehend interpretieren, dass sie nicht vom zusammengesetzten Zustand ausgeht, sondern direkt von jedem Unterzustand. Die Bedingungen für die Ausnahmetransitionen werden **nach** ihren DO-Aktionen bewertet. Da ENTRY- und EXIT-Aktionen nicht an Bedingungen geknüpft sind, werden sie immer unabhängig von der Ausnahmetransition ausgeführt.

Eine Ausnahmetransition wird durch einen rot gestrichelten Pfeil dargestellt und seine Spitze zeigt auf den nächsten Zustand.



Pseudo-Ausnahmetransition

Eine Pseudoausnahmetransition ersetzt den Endzustand in einem zusammengesetzten Zustand und wird durch einen blau gestrichelten Pfeil dargestellt.

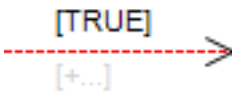
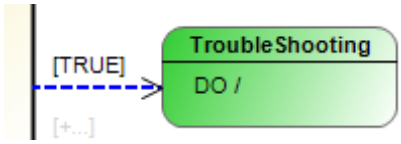
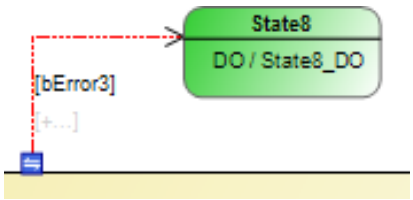


Manchmal ist es übersichtlicher, anstelle eines Endzustandes in einem zusammengesetzten Zustand, der mit sehr vielen Transitionen verbunden ist, eine Transition am Rande des zusammengesetzten Zustands zu zeichnen, unter deren Bedingung der gesamte zusammengesetzte Zustand verlassen wird. Dies geschieht analog zur Ausnahmetransition, aber ohne deren inhaltliche Bedeutung, dass dies ein Fehlverhalten darstellt. Es wird daher als Pseudo-Ausnahmetransition bezeichnet. Eine Pseudo-Ausnahmetransition hat keinen Einfluss auf das zyklische Ausführungsverhalten.

Eigenschaften

„Eigenschaft“	Beschreibung
„Beziehungstyp“	Ausnahmetransition (nicht editierbar)
„Priorität“	Priorität, die die Abarbeitungsreihenfolge festlegt Beispiel: 3 Hinweis: Wenn der Zustand weitere Transitionen hat und Sie die Priorität ändern, sind alle Transitionen von der Änderung betroffen und werden automatisch angepasst.
„Pseudoausnahme“	• ☐ : Ausnahmetransition. Im Editor wird der Pfeil mit gestrichelter roter Linie dargestellt. [Voreinstellung] • ☑ : Pseudo-Ausnahmetransition. Im Editor wird der Pfeil mit gestrichelter blauer Linie dargestellt. Wenn die Bedingung erfüllt ist, wird der Zustand verlassen. Allerdings handelt es sich nicht um ein Fehlersignal, das das Weiterschalten verursacht.
„Steigende Flanke“	• ☑ : Wenn die Wächterbedingung eine steigende Flanke (von 0 auf 1) liefert, wird die Transition durchlaufen und der Zustandsübergang ausgeführt. Tipp: Im Editor ist eine Transition, die sich so verhält, mit dem Symbol  markiert. • ☐ : Wenn die Wächterbedingung TRUE ist, wird die Transition durchlaufen.

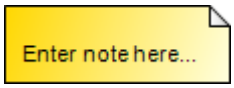
Ausnahmetransition editieren

Benutzereingabe im Zustands-diagramm	Reaktion im Zustandsdiagramm	Beschreibung
Selektieren Sie „Ausnahmetransition“ in der Werkzeugbox. Klicken Sie auf einen zusammengesetzten Zustand (Quellzustand) und danach auf einen anderen Zustand (Zielzustand).		Eine Ausnahmetransition mit TRUE-Bedingung und einem Aktionssymbol wird angelegt.
Klicken Sie auf einen zusammengesetzten Zustand (Quellzustand) im Zustandsdiagramm.		Das Icon, über das eine ausgehende Ausnahmetransition zu dem ausgewählten Zustand hinzugefügt werden kann, ist verfügbar. Wenn Sie es verwenden, wird der Zustand um eine ausgehende Ausnahmetransition erweitert. Der Zielzustand wird mit einem weiteren Klick bestimmt.
Doppelklicken Sie auf das Symbol [TRUE]		Ein Eingabefeld mit IntelliSense-Funktionalität öffnet. Wählen Sie eine boolesche Variable oder einen booleschen Ausdruck aus. Ist die gewünschte Auswahl im IntelliSense selektiert, kann die Auswahl mittels Doppelklick oder mittels Fokussieren plus [Enter] ausgewählt werden.
Klicken Sie zwei Mal auf das Symbol [+...]		Ein Eingabefeld mit IntelliSense-Funktionalität öffnet. Wählen Sie eine Methode oder Aktion aus. Ist die gewünschte Auswahl im IntelliSense selektiert, kann die Auswahl mittels Doppelklick oder mittels Fokussieren plus [Enter] ausgewählt werden.
Klicken Sie auf die Transition, auf das Aktionssymbol oder auf das Überwachungssymbol.		Die Transition ist selektiert (dargestellt durch einen blauen Pfeil). Eine selektierte Transition kann via Drag-and-Drop verschoben werden. Die Position der verbundenen Zustände bleibt unverändert.
Klicken auf den Anfang oder das Ende einer Ausnahmetransition.		Das Symbol  ermöglicht ein erneutes Anschließen der Transition, ohne dass die bisherige Transitionskonfiguration verloren geht. Sie können das Symbol auf einen anderen Quell- bzw. Zielzustand ziehen, um die Transition dort anzuschließen. Die zugehörige Bedingung und Aktion bleiben erhalten.

7.4.10 Notiz

Mit dem Element „Notiz“ können Sie im Editor eines Klassendiagramms oder Zustandsdiagramms einen kommentierenden Text einfügen.

Aktuell sind nur einzeilige Notizen möglich.



Notiz hinzufügen

Wählen Sie das Werkzeug „Notiz“: 

Klicken Sie im Editor auf die gewünschte Einfügeposition. Doppelklicken Sie anschließend auf den Text im Element und ersetzen Sie ihn durch den gewünschten Text.

Cursor

-  : An nicht erlaubten Einfügestellen hat der Cursor die Form eines Verbotsschilds.
-  : An erlaubten Stellen ist der Cursor ein blaues Kreuz.

7.5 Objekteigenschaften

1. Selektieren Sie ein UML Zustandsdiagramm im Projektbaum.
2. Öffnen Sie das Kontextmenü und führen Sie **Eigenschaften** aus oder klicken Sie auf **Ansicht > Eigenschaften**.

Im Folgenden werden die UML-relevanten Registerblätter beschrieben.

UML

- **Abbrechbar** (Abortable):
  - Option verfügbar für: alle Zustandsdiagramme
  - Option aktiviert (Default): Die Abarbeitung des Zustandsdiagramm kann abgebrochen werden. Ein solches Zustandsdiagramm hat eine zusätzliche interne Variable: _UML_SC_<name>.Abort. Wenn diese Variable mit IEC-Code gesetzt wird, kann die Abarbeitung des Zustandsdiagramms unmittelbar gestoppt werden, unabhängig vom internen Zustand.
  - Option deaktiviert: Die Abarbeitung des Zustandsdiagramm kann nicht abgebrochen werden.
- **VarInstNutzen** (UseVarInst):
  - Option verfügbar für: Zustandsdiagramme, die als Programmelement „Methode“ angelegt wurden und zu einem Funktionsbaustein gehören
  - Option aktiviert (Default): Die Daten der Zustandsdiagramm-Methoden werden nicht als temporär, sondern als Instanzvariablen des dazugehörigen Funktionsbausteins gehalten. Wenn Sie die Variablen der Methode als VAR_INST deklarieren, werden die Daten wie bei einem Funktionsbaustein gehalten und sind nicht mehr temporär, so dass das Zustandsdiagramm wie gewohnt in mehreren Taskzyklen durchlaufen wird.
  - Option deaktiviert: Die Daten der Zustandsdiagramm-Methoden werden als temporär gehalten. Methoden und ihre Daten sind üblicherweise temporär. Wenn Sie eine Methode mit der Implementierungssprache UML Zustandsdiagramm implementieren und die Option „UseVarInst“ deaktivieren, verhält sich die Methode wie ein zyklusinterner Zustand und initialisiert am Anfang des Taskzyklus die Daten neu. In diesem Fall können nur zyklusinterne Zustände verwendet werden.

- Beachten Sie, dass diese Option nur für Zustandsdiagramm-Methoden zur Verfügung steht, die zu einem Funktionsbaustein gehören, und nicht für solche, die zu einem Programm gehören, da Programme keine VAR_INST-Deklarationen enthalten können.

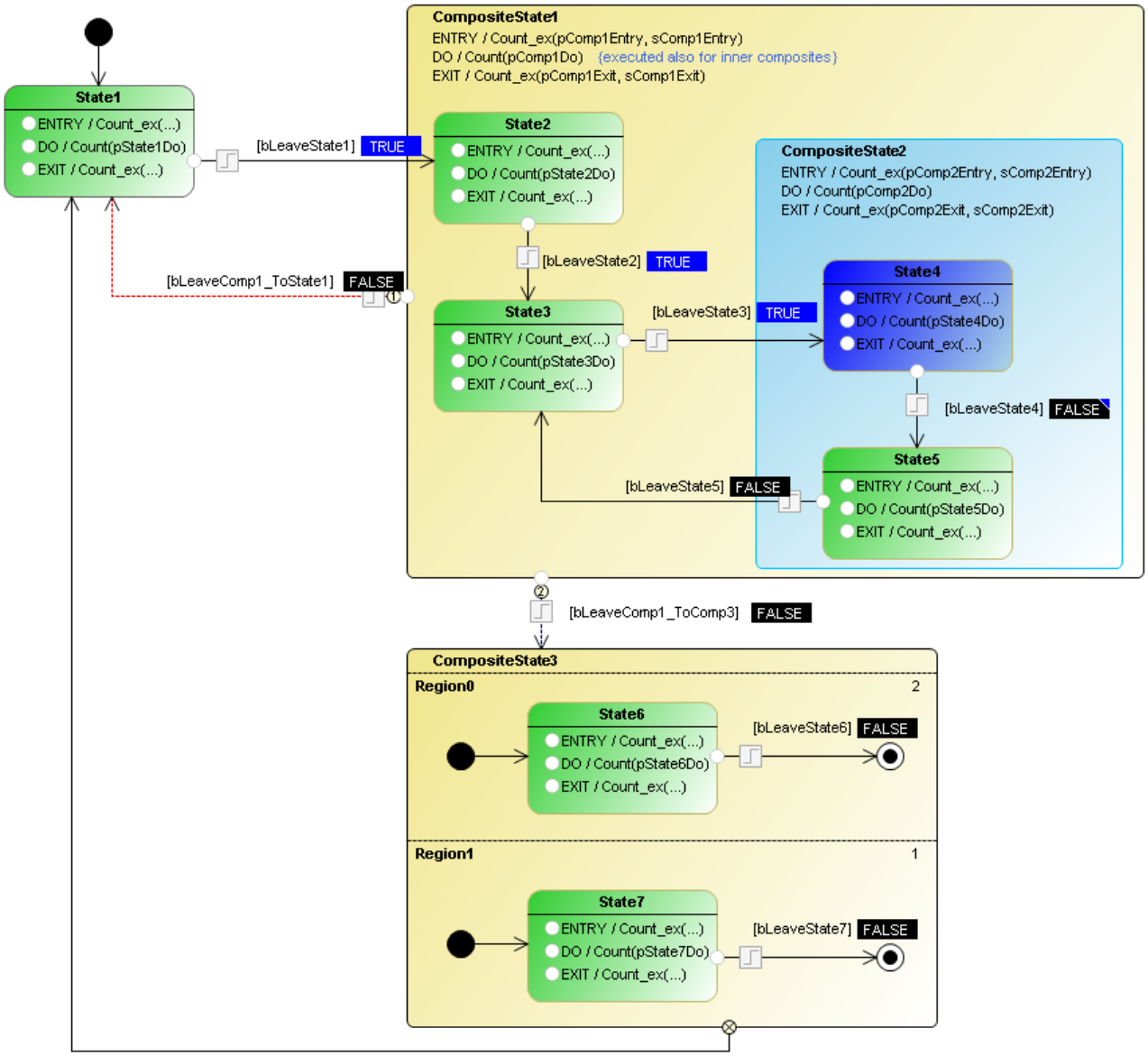
7.6 Online-Modus

Das Verhalten von Zustandsdiagramm-Objekten im Online-Modus entspricht dem normalen Online-Verhalten in TwinCAT 3. Wie üblich ist es möglich, eine geladene Applikation zu monitoren, Variablen zu schreiben oder zu forcen, Breakpoints zu setzen oder ein Programm in Einzelzyklen abarbeiten zu lassen.

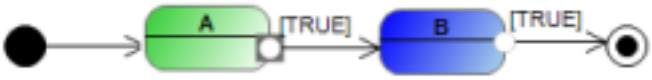
Im Online-Modus ist ein aktiver Zustand blau und ein aktiver zusammengesetzter Zustand hellblau dargestellt. Der Wert eines Transitionsausdrucks wird im Online-Modus neben der Transition angezeigt. Falls die Transition nur aus einer Transitionsvariablen besteht, kann der Wert der Variablen per Doppelklick auf den Monitoring-Wert fürs Schreiben oder Forcen verändert werden. Ist ein Wert fürs Schreiben oder Forcen vorbereitet worden, ist dies an der oberen rechten Ecke des Monitoring-Bereichs sichtbar (siehe das Monitoring-Feld der Transitionsvariablen *bLeaveState4* am rechten, mittleren Rand des folgenden Bilds: die blaue Ecke zeigt an, dass für die Variable der Wert TRUE fürs Schreiben oder Forcen vorbereitet ist).

Eine zusätzliche Debug-Möglichkeit wird dadurch geboten, dass die Variablen einer Transition per Befehl in die Überwachungsliste übernommen werden können. Wählen Sie dazu den Befehl Zur Überwachungsliste hinzufügen [▶ 62] aus dem Kontextmenü der Transition. Dies ist besonders dann hilfreich, wenn sich ein Transitionsausdruck aus der Kombination mehrerer Variablen zusammensetzt (z.B. „bCondition1 AND bCondition2“). Wenn Sie für diese Transition den Befehl **Zur Überwachungsliste hinzufügen** ausführen, werden beide Variablen des Transitionsausdrucks zur Überwachungsliste hinzugefügt, also *bCondition1* und *bCondition2*. Dadurch können Sie sich einen einfachen Überblick verschaffen, welches Teilsignal des Gesamtausdrucks für die Weiterschaltung in den nächsten Zustand noch nicht den benötigten Wert hat.

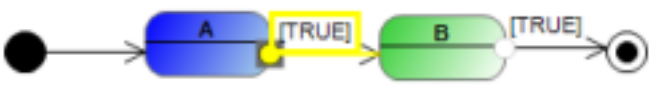
Des Weiteren werden die Befehle Gehe zu Definition [▶ 62] und Alle Verweise suchen [▶ 62] für Transitionen unterstützt.



Die möglichen Positionen für Breakpoints (Haltepunkte) werden als weißer Kreis mit grauem Rand im Zustandsdiagramm angezeigt. Wenn Sie einen Haltepunkt setzen möchten, selektieren Sie einen Kreis. Der Kreis wird grau hinterlegt: .



Wählen Sie **Debug > Haltepunkt umschalten** oder [F9]. Das Programm hält am Haltepunkt und die Stelle im Zustandsdiagramm ist gelb markiert.



Online-Change

Offline-Bearbeitungen an einem bestehenden Zustandsdiagramm können in der Regel ohne Reinitialisierung des Diagramms mit einem Online-Change auf die Steuerung geladen werden, wenn Sie Transitionen, Bedingungen, Prioritäten, Aktionen oder Namen ändern. Der Zustand, der vor dem Online-Change aktiv war, ist somit auch nach dem Online-Change noch aktiv.

Das Zustandsdiagramm wird jedoch bei einem Online-Change immer dann neu gestartet, d. h. es wird der Startzustand aktiviert, wenn ein Zustand oder eine Region hinzugefügt, entfernt, oder ersetzt wurde. Das trifft auch für implizite Zustände wie zum Beispiel das Element „Auswahl“ zu, oder für die Eigenschaft, ein Zustandsdiagramm „abbrechbar“ zu machen.

8 FAQ

OOP und UML

Müssen die objektorientierte Programmierung (OOP) und UML stets zusammen verwendet werden?

- Die kombinierte Nutzung von OOP und UML bietet viele Vorteile (s. nächste Frage), ist allerdings nicht zwangsläufig nötig. Es ist auch ohne die Verwendung von UML möglich, eine Applikation objektorientiert zu programmieren. Genauso kann UML auch in SPS-Projekten verwendet werden, die nicht objektorientiert programmiert werden bzw. wurden.

Welche Vorteile bietet es, OOP und UML zusammen zu verwenden?

- Um die Vorteile der OOP optimal zu nutzen, sollte die Struktur einer objektorientierten Software vor Implementierungsbeginn konzipiert bzw. erstellt werden (z. B.: Welche Klassen sind vorhanden, in welcher Beziehung stehen sie zueinander, welche Funktionalitäten stellen sie bereit, etc.). Vor, während und nach der Programmierung helfen Dokumentationen dabei, die Software zu verstehen, zu analysieren und zu warten.
- Als Analyse-, Design- und Dokumentationswerkzeug von Software bietet UML entsprechende Möglichkeiten, um die Applikation zu planen, zu erzeugen und zu dokumentieren. Dabei eignet sich die Verwendung von UML besonders bei objektorientierten Implementierungen, da vor allem modular aufgebaute Software mithilfe einer grafischen Sprache dargestellt werden kann.
- So wird beispielsweise das Klassendiagramm zum Analysieren und Erzeugen der Programmstruktur verwendet – und je modularer die Software aufgebaut ist, desto einfacher und effizienter kann das Klassendiagramm genutzt werden (z. B.: Grafische Darstellung von separaten Funktionsblöcken mit einzelnen Methoden zur Bereitstellung der Funktionalitäten, etc.).
- Das Zustandsdiagramm kann genutzt werden, um den Ablauf eines ereignisdiskreten Systems zu spezifizieren – und je konsequenter die Software objekt- bzw. ereignisorientiert aufgebaut ist, desto übersichtlicher und effektiver können Zustandsmaschinen entworfen werden (z. B.: Das Verhalten von Modulen/Systemen basiert auf einem Zustandsmodell mit Zuständen (wie Aufstarten, Produktion, Pausezustand) und innerhalb der Zustände werden entsprechende Funktionalitäten aufgerufen, die in Methoden (wie Aufstarten, Ausführen, Pausieren) gekapselt sind, etc.).

UML Zustandsdiagramm-Methoden mit deaktivierter "UseVarInst"-Option

Die folgenden drei Fragen beziehen sich auf Methoden, deren Implementierungssprache **UML Zustandsdiagramm** ist und die zusätzlich eine der nachfolgenden Bedingung erfüllen:

- Die Zustandsdiagramm-Methode gehört zu einem Programm.
- Die Zustandsdiagramm-Methode gehört zu einem Funktionsbaustein und die Option **UseVarInst** der Methode ist deaktiviert.

Zustandsdiagramm-Methoden, die diese Bedingungen erfüllen, werden in den folgenden Fragen als „die genannten Methoden“ bezeichnet.

Weiterführende Informationen zu der Option **UseVarInst** finden Sie unter UML Zustandsdiagramm > Objekteigenschaften. [▶ 90]

Warum können die genannten Methoden nur zyklusinterne Zustände beinhalten und warum gibt es kein Online-View?

- Generell gilt, dass standardmäßig alle Daten einer Methode temporär und nur während der Ausführung einer Methode gültig sind (Stack-Variablen). Wenn Sie eine Variable einer Methode hingegen mittels VAR_INST als Instanzvariable deklarieren, wird sie nicht auf dem Methoden-Stack, sondern auf dem Stack der Funktionsbaustein-Instanz abgelegt. Sie verhält sich also wie andere Variablen der Funktionsbaustein-Instanz und wird nicht bei jedem Aufruf der Methode neu initialisiert.
- Da die Daten der genannten Zustandsdiagramm-Methode nicht als Instanzvariablen deklariert werden, handelt es sich hierbei um zyklusinterne Zustandsmaschinen. Somit werden die in den Diagrammen enthaltenen Zustände zyklusintern und nicht vom Taskzyklus geschaltet, da die Methodenausführung aufgrund der temporären Datenhaltung mit jedem Taskzyklus „neu beginnt“. Folglich ist für diese Zustandsdiagramm-Methoden auch kein Online-View möglich, da die Variablen der Zustandsmaschine temporär sind und sich der aktive Zustand innerhalb eines Zyklus mehrfach ändern kann.

- Dies steht im Gegensatz zu Zustandsdiagrammen ohne temporäre Datenhaltung (z. B. Programme oder Funktionsbausteine als Zustandsdiagramm, Funktionsbausteinmethode als Zustandsdiagramm mit aktivierter **UseVarInst**-Option), bei denen die Zustände standardmäßig vom Taskzyklus geschaltet werden und optional als „zyklusintern“ konfiguriert werden können.

Warum enthält die Werkzeugliste der genannten Methoden keine Composite States (zusammengesetzte Zustände), Forks (Gabelungen) und Joins (Verbindungen)?

- Da die Daten dieser Methoden temporär und nur während der Methodenausführung gültig sind, können die genannten UML Zustandsdiagramm-Methoden nur zyklusinterne Zustände besitzen (s. vorherige Frage).
- Ein Composite State müsste demzufolge auch in einem Zyklus beendet werden. Damit würde das Element stets sequenziell abgearbeitet werden, selbst wenn die Ausführung einer Region nicht abgeschlossen wird.
- Dieses Verhalten wäre damit fundamental anders als bei „normalen“ Composite States, die sich ihren internen Zustand merken können.
- Um für ein Composite State nicht verschiedene Verhaltensweisen anzubieten, je nachdem bei welcher POU-Art das Element verwendet wird, sind die Composite States in den genannten Methoden nicht erlaubt.
- Da Forks und Joins nur in Zusammenhang mit Composite States erlaubt sind, stehen auch diese Elemente in den genannten Zustandsdiagramm-Methoden nicht zur Verfügung.

Was ist bei der Programmierung der genannten Methoden zu beachten? Was könnte die Ursache für eine hohe Systemauslastung bei der Ausführung dieser Methoden sein?

- Wie in den vorherigen Fragen erläutert, handelt es sich bei den genannten UML Zustandsdiagramm-Methoden um zyklusinterne Zustandsmaschinen. Sie werden also zyklusintern und nicht vom Taskzyklus geschaltet.
- Das bedeutet, dass, wenn die Aktionen eines internen Zustands abgeschlossen sind, unmittelbar in die Transition geschaltet wird. Es wird damit sofort die Transitionsbedingung geprüft und bei erfüllter Bedingung die Transitionsaktion ausgeführt. Ebenso wird bei erfüllter Transitionsbedingung sofort anschließend in den Zielzustand geschaltet.
- Sollte die Transitionsbedingung hingegen nicht erfüllt sein, wird folglich auch nicht in den Zielzustand geschaltet und der aktuelle Zustand bleibt aktiv. Damit wird (wegen der zyklusinternen Zustände) noch im selben Zyklus erneut die DO-Aktion des aktuellen Zustands aufgerufen, falls die maximalen DO-Zyklus Aufrufe noch nicht erreicht wurden.
  - **Max. DO-Zyklus Aufrufe** (Max. DO-cycle calls): Für einen Zustand können die maximalen DO-Zyklus Aufrufe konfiguriert werden, wobei ein Wert zwischen 1 und 32767 eingestellt werden kann. Diese Zahl gibt die maximale Anzahl an Aufrufen der DO-Aktion an.
- Folglich ist zu beachten: Sollte die Transitionsbedingung während des gesamten Taskzyklus nicht erfüllt sein, wird die DO-Aktion des aktuell aktiven Zustands in diesem Zyklus so lange ausgeführt, bis die maximalen DO-Zyklus Aufrufe dieses Zustands erreicht sind.

Beispiel: Falls für "max. DO-Zyklus Aufrufe" ein Wert von 32767 eingestellt ist und die Transitionsbedingung während des Taskzyklus unerfüllt bleibt, wird die DO-Aktion des entsprechenden Zustands 32767-mal in einem Zyklus ausgeführt! Da dies je nach Umfang der DO-Aktion zu einer hohen Systemauslastung führen kann, sollte genau geprüft werden, ob die Zustandsmaschine und speziell die Transitionsbedingungen sowie die Werte der maximalen DO-Zyklus Aufrufe korrekt konfiguriert sind und dem gewünschten Applikationsverhalten entsprechen.

9 Samples

9.1 UML-Klassendiagramm

9.1.1 1 Basis

Dieses „UML Class Diagram“-Beispiel zeigt die grundlegende Funktionalität vom UML-Klassendiagramm [▶ 17].

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716013451.zip

ClassDiagram_1

Auf diesem Klassendiagramm wird die Struktur des SPS-Projekts dargestellt. Es beinhaltet verschiedene SPS-Elemente (wie Funktionsbaustein, Schnittstelle, globale Variablenliste, usw.).

Außerdem zeigt das Klassendiagramm die Beziehung zwischen den Elementen (Vererbung zwischen Funktionsbausteinen, Implementierung einer Schnittstelle, Instanziierung eines Funktionsbausteins).

ClassDiagram_2

Mithilfe dieses Klassendiagramms können Sie die folgenden grundlegenden Funktionalitäten ausprobieren:

- Bestehendes Element aus den Querverweisen auf dem Diagramm visualisieren
  - Fenster „Werkzeuge“ anzeigen
  - FB_InfoData auf dem Klassendiagramm selektieren
  - Werkzeugfenster: Unter der Überschrift „Eingehende Querverweise“ und „Ausgehende Querverweise“ werden Elemente angezeigt, die eine Beziehung zum selektierten Element haben, aber nicht im Klassendiagramm enthalten sind.
  - Sie können das Element via Drag-and-Drop auf das Diagramm ziehen, sodass das Element im Klassendiagramm dargestellt wird.
  - Weiterführende Informationen hierzu finden Sie unter Bestehende Elemente zu einem Diagramm hinzufügen [▶ 20].
- Bestehendes Element aus dem Projektbaum auf dem Diagramm visualisieren
  - Markieren Sie ein Element vom Typ POU, INTERFACE, GVL oder DUT im Projektbaum und ziehen Sie es per Drag-and-Drop auf das geöffnete Klassendiagramm.
  - Lassen Sie es an einer geeigneten Stelle fallen, um es dort zu visualisieren.
  - Daraufhin wird das entsprechende Element auf dem Diagramm dargestellt. Falls Beziehungen zu bereits abgebildeten Elementen bestehen, werden diese automatisch angezeigt.
  - Weiterführende Informationen hierzu finden Sie unter Bestehende Elemente zu einem Diagramm hinzufügen [▶ 20].
- Neue Elemente hinzufügen oder neue Beziehungen zwischen Elementen erstellen
  - Verwenden Sie die Elemente des Fensters Werkzeuge.
  - Weiterführende Informationen hierzu finden Sie Klassendiagramm bearbeiten [▶ 22] und unter Elemente [▶ 25].

9.1.2 2 Einfache Maschine

Dieses „UML Class Diagram“-Beispiel zeigt die grundlegende Funktionalität vom UML-Klassendiagramm [▶ 17] am Beispiel einer exemplarischen, objektorientiert programmierten Maschine.

Beachten Sie, dass die Module nicht funktional implementiert wurden. Das Beispiel soll dazu dienen, die Funktionalitäten vom UML Klassendiagramm kennenzulernen und an einer beispielhaften Programmstruktur nachzuvollziehen.

Mithilfe des Klassendiagramms kann die Struktur eines SPS-Programms erzeugt, erweitert und verändert werden.
Zusätzlich, wie in diesem Beispiel, kann die Struktur eines SPS-Programms Mithilfe des Klassendiagramms dokumentiert und auf Basis dieser Grafik einfach nachvollzogen werden.

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716015115.zip

Maschinenmodule

Die Applikation verfügt über die folgenden Ebenen und Module.

Maschinenebene: Maschine (FB_Machine)

Subsystem-Ebene: Ausschleusung (FB_Ejector)

Submodul-Ebene: Zylinder ohne bzw. mit Hardware-Feedback-Signal (FB_Cylinder, FB_CylinderFeedback)

Vererbung

Da alle diese genannten Funktionsbausteine Gemeinsamkeiten besitzen (gemeinsame Daten und Funktionalitäten), gibt es eine Basisklasse, in der diese Gemeinsamkeiten einmal implementiert werden. Über den Vererbungsmechanismus der objektorientierten Programmierung werden diese Implementierungen an die Unterklassen vererbt.

- Die vier Maschinenmodule erweitern FB_ModuleRoot.

Des Weiteren stellt der Zylinder mit Feedback-Funktionalität eine Erweiterung des Zylinders ohne diese Funktionalität dar. Daher wird auch an dieser Stelle die Vererbung verwendet.

- FB_CylinderFeedback erweitert FB_Cylinder.

Schnittstelle

Um die Anforderungen an unterschiedliche Zylinderarten zu definieren, werden die grundlegenden Methoden und Eigenschaften, die ein Zylinder bereitstellen muss, in einer Schnittstelle festgelegt.

- I_Cylinder definiert die Zylinderanforderungen.
- FB_Cylinder implementiert I_Cylinder.

Instanziierung

- MAIN instanziiert FB_Machine.
- FB_Machine instanziiert FB_Ejector zweimal, da die Maschine über zwei Ausschleusungsmodule verfügt.
- FB_Ejector instanziiert I_Cylinder, FB_Cylinder und FB_CylinderFeedback. Als Beispiel ist immer nur ein Zylinder gleichzeitig aktiv, entweder der Zylinder mit Feedback-Funktionalität oder der Zylinder ohne Feedback-Funktionalität. Der Interfacevariablen iCylinder wird der aktuell aktive Zylinderinstanz zugewiesen. Dadurch kann der aktive Zylinder generalisiert über die Interfacevariable angesteuert werden.
- FB_ModuleRoot instanziiert ST_Error.

9.2 UML-Zustandsdigramm

Bei den nachfolgenden UML SC Beispielen erhöhen sich der Schwierigkeitsgrad und der Umfang der Beispiele mit aufsteigender Nummerierung.

9.2.1 1 Lampe

Dieses „UML Statechart“-Beispiel zeigt die grundlegende Funktionalität vom UML-Zustandsdiagramm [▶ 55] und beinhaltet die folgenden UML SC Elemente:

- Startzustand [▶ 64]
- Zustand [▶ 65]

- Transition [▶ 82]

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716016779.zip

Überblick

Mithilfe von UML SC ist das Verhalten einer Lampe programmiert, die über einen Schalter auf der Visualisierung ein- und ausgeschaltet werden kann. Dafür verfügt die Lampe über die beiden Zustände „On“ und „Off“.

Beide Zustände beinhalten jeweils eine ENTRY- und eine DO-Aktion.

- In den ENTRY-Aktionen wird die Lampe ein- bzw. ausgeschaltet. Die ENTRY-Aktion wird jeweils einmal aufgerufen, wenn der zugehörige Zustand aktiviert wird.
- In den DO-Aktionen wird ein ON- bzw. OFF-Zähler inkrementiert. Dadurch ist ersichtlich, dass eine DO-Aktion permanent aufgerufen wird, solange der zugehörige Zustand aktiv ist.
- Als Transition zwischen den Zuständen wird der Wert des Schalters abgefragt.

Visualisierung

Ergänzend zu dem komfortablen Online-Modus [▶ 91] des UML SC Diagramms können das Verhalten der Lampe und die Werte der Zähler über die Visualisierung verfolgt werden.

9.2.2 2 Fußgängerampel

Dieses „UML Statechart“-Beispiel zeigt die grundlegende Funktionalität vom UML-Zustandsdiagramm [▶ 55] und beinhaltet die folgenden UML SC Elemente:

- Startzustand [▶ 64]
- Zustand [▶ 65]
- Transition [▶ 82]

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716018443.zip

Überblick

Mithilfe von UML SC ist das Verhalten einer Fußgängerampel programmiert. Über einen Taster auf der Visualisierung kann eine Grünphasen-Anforderung abgesetzt werden. Dafür verfügt die Fußgängerampel über die beiden Zustände „Red“ und „Green“.

Wenn eine Grünphase angefordert wird, schaltet die Ampel nach Ablauf der Zeit „cTimeWaitForGreen“ auf grün um. Sobald die Zeit „cTimeGreenPhase“ abgelaufen ist, schaltet die Ampel wieder auf rot.

Die beiden Zustände „Red“ und „Green“ beinhalten jeweils eine ENTRY- und eine DO-Aktion.

- In den ENTRY-Aktionen wird die Ampel auf rot bzw. grün geschaltet. Außerdem wird der jeweilige Timer-Baustein (vom Typ TON) zurückgesetzt. Die ENTRY-Aktion wird jeweils einmal aufgerufen, wenn der zugehörige Zustand aktiviert wird.
- In den DO-Aktionen wird ein Red- bzw. Green-Zähler inkrementiert. Dadurch ist ersichtlich, dass eine DO-Aktion permanent aufgerufen wird, solange der zugehörige Zustand aktiv ist. Zudem wird der jeweilige Timer-Baustein aufgerufen.
- Als Transition zwischen den Zuständen wird der Q-Ausgang des jeweiligen Timer-Bausteins verwendet, sodass der Zustand nach Ablauf der entsprechenden Zeit gewechselt wird.

Visualisierung

Ergänzend zu dem komfortablen Online-Modus [▶ 91] des UML SC Diagramms können das Verhalten der Fußgängerampel, die Werte der Zähler sowie die bereits abgelaufene Wartezeit über die Visualisierung verfolgt werden.

9.2.3 3 SaveText-Simulation

Dieses „UML Statechart“-Beispiel zeigt die grundlegende Funktionalität vom UML-Zustandsdiagramm [▶ 55] und beinhaltet die folgenden UML SC Elemente:

- Startzustand [▶ 64]
- Zustand [▶ 65]
- Transition [▶ 82]
- Auswahl [▶ 81]
- Zusammengesetzter Zustand [▶ 69]
- Abschlusstransition [▶ 85]

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716020107.zip

Überblick

Mithilfe von UML SC wird das Verhalten einer Anwendung simuliert, bei der ein Text als XML- oder Textformat gespeichert werden soll. Anschließend sollen die Informationen an einen Client und an einen Master gesendet werden.

Dieses Verhalten ist in dem Beispielprojekt nicht tatsächlich implementiert worden. Stattdessen wird die äußere Hülle bzw. die Basisstruktur der Zustandsmaschine exemplarisch mittels UML SC umgesetzt.

Zustände

Das UML SC Diagramm verfügt über die folgenden Zustände:

- InitState
- XmlFormat
- TextFormat
- PublishToMaster
- PublishToClient

Zähler/Aufrufhäufigkeit

Die Zustände beinhalten jeweils eine DO-Aktion sowie zusätzlich eine ENTRY- und/oder EXIT-Aktion. In den Aktionen wird jeweils ein entsprechender Zähler inkrementiert. Dadurch ist ersichtlich:

- dass die ENTRY-Aktion jeweils einmal aufgerufen wird, wenn der zugehörige Zustand aktiviert wird
- dass die EXIT-Aktion jeweils einmal aufgerufen wird, wenn der zugehörige Zustand verlassen wird.
- dass die DO-Aktion permanent aufgerufen wird, solange der zugehörige Zustand aktiv ist.

Auswahlelement

Ob der Zustand „XmlFormat” oder „TextFormat“ aktiviert wird, wird mithilfe des Auswahlelements entschieden. Das Element verfügt über zwei ausgehende Transitionen. Die Transition mit der höheren Priorität, d.h. mit der niedrigeren Prioritätszahl, wird zuerst überprüft. Wenn die zugehörige Transitionsbedingung erfüllt ist, wird der Zielzustand aktiviert.

Zusammengesetzter Zustand

Die Zustände „PublishToMaster“ und „PublishToClient“ befinden sich in unterschiedlichen Regionen eines zusammengesetzten Zustands, sodass sie pseudo-parallel abgearbeitet werden. Das bedeutet, dass beide Zustände gleichzeitig aktiv sind, jedoch wird die Region mit der höheren Priorität, d.h. mit der niedrigeren Prioritätszahl, zuerst ausgeführt. Welche Region zuerst abgearbeitet wurde, wird auf der Visualisierung anhand einer farblichen Hervorhebung dargestellt. Außerdem ist bei der Ausführung des Beispielprojektes zu erkennen, dass der zusammengesetzte Zustand erst dann verlassen wird, wenn beide Unterzustandsmaschinen ihren Endzustand erreicht haben.

Projektänderung möglich: Die Prioritäten der Regionen können Sie verändern, indem Sie die Zahl in der oberen rechten Ecke der Region anpassen. Wenn Sie Region1 mit der Priorität 1 konfigurieren und diese Änderung per Online Change herunterspielen, können Sie bei der nächsten Aktivierung des zusammengesetzten Zustands erkennen, dass nun „Client first“ anstelle von „Master first“ hervorgehoben wird. Folglich wurde nun die Region1 zuerst ausgeführt.

Visualisierung

Ergänzend zu dem komfortablen Online-Modus [▶ 91] des UML SC Diagramms können das Verhalten der Zustandsmaschine sowie die Werte der Zähler über die Visualisierung verfolgt werden. Außerdem befinden sich auf der Visualisierung Schalter, um zwischen den Zuständen hin- und herzuschalten.

Die Werte der Transitionsbedingungen können Sie nicht nur über die Visualisierung oder über den Deklarationseditor des FBs, sondern auch direkt über das UML SC Diagramm verändern.

Falls eine Transition aus nur einer Transitionsvariablen besteht, kann der Wert der Variablen per Doppelklick auf den Monitoring-Wert fürs Schreiben oder Forcen verändert werden. Ist ein Wert fürs Schreiben oder Forcen vorbereitet worden, ist dies an der oberen rechten Ecke des Monitoring-Bereichs sichtbar (siehe auch: Online-Modus [▶ 91]).

9.2.4 4 Aufrufverhalten - Basis

Dieses „UML Statechart“-Beispiel zeigt das grundlegende Aufrufverhalten vom UML-Zustandsdiagramm [▶ 55] und beinhaltet die folgenden UML SC Elemente:

- Startzustand [▶ 64]
- Zustand [▶ 65]
- Transition [▶ 82]
- Zusammengesetzter Zustand [▶ 69]
- Abschlusstransition [▶ 85]
- Ausnahmetransition [▶ 86]

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716021771.zip

Überblick

Mithilfe der folgenden Teilaspekte wird das Aufrufverhalten von Zuständen und zusammengesetzten Zuständen veranschaulicht. Mithilfe von verschiedenen Programmiermitteln werden die unterschiedlichen Aufrufaspekte veranschaulicht (Programmiermittel => Aufrufaspekt).

- Zähler => Aufrufhäufigkeit
- Eintragung der aufgerufenen Zustandsaktion in ein Array => Aufrufreihenfolge
- Taskzyklus-IDs => Zuordnung des Aufrufs zur Zyklus-ID

Die jeweilige Zuordnung des Programmiermittels zum Aufrufaspekt wird im Folgenden erläutert.

Bzgl. des zusammengesetzten Zustands werden beide Anwendungsfälle verwendet:

- Gruppierung/Verschachtelung, wobei der zusammengesetzte Zustand über eigene ENTRY-/DO-/EXIT-Aktionen verfügt
- Parallele Sub-Zustandsmaschinen

Zähler => Aufrufhäufigkeit

Jede ENTRY-/DO-/EXIT-Aktion, die in einem Zustand oder in einem zusammengesetzten Zustand aufgerufen wird, inkrementiert einen zugehörigen Zähler. Dadurch wird die Aufrufhäufigkeit einer Aktion veranschaulicht, sodass ersichtlich ist:

- dass die ENTRY-Aktion jeweils einmal aufgerufen wird, wenn der zugehörige (zusammengesetzte) Zustand aktiviert wird.
- dass die EXIT-Aktion jeweils einmal aufgerufen wird, wenn der zugehörige (zusammengesetzte) Zustand verlassen wird.
- dass die DO-Aktion permanent aufgerufen wird, solange der zugehörige (zusammengesetzte) Zustand aktiv ist.
- dass ein zusammengesetzter Zustand mit einer Region (Anwendungsfall Gruppierung/ Verschachtelung) aktiviert wird, wenn zuvor keiner der inneren Zustände aktiv war und sobald einer der inneren Zustände aktiviert wird.
- dass ein zusammengesetzter Zustand mit einer Region (Anwendungsfall Gruppierung/ Verschachtelung) aktiv bleibt, solange einer der inneren Zustände aktiv ist.
- dass ein zusammengesetzter Zustand mit einer Region (Anwendungsfall Gruppierung/ Verschachtelung) verlassen wird, wenn zuvor einer der inneren Zustände aktiv war und wenn dieser Zustand verlassen wird, sodass nun keiner der inneren Zustände mehr aktiv ist.
- dass es von der Option „DO-Aktionen auch ausführen, wenn die inneren zusammengesetzten Zustände aktiv sind“ abhängt, ob die DO-Aktion eines äußeren zusammengesetzten Zustands (im Beispiel „CompositeState1“) weiterhin aufgerufen wird, wenn ein innerer zusammengesetzter Zustand aktiv ist (im Beispielprojekt: „CompositeState2“).

Eintragung der aufgerufenen Zustandsaktion in ein Array => Aufrufreihenfolge

Jede ENTRY-/DO-/EXIT-Aktion, die in einem Zustand oder in einem zusammengesetzten Zustand aufgerufen wird, trägt ihren Namen (z.B. „State1__Entry“) in ein Array ein. Dadurch wird die Aufrufreihenfolge der verschiedenen Aktionen veranschaulicht, sodass ersichtlich ist:

- dass die prinzipielle Aufrufreihenfolge bei einem Zustand lautet: erst ENTRY, dann DO, dann EXIT. Im Beispielprojekt:
  - State1__Entry
  - State1__Do
  - State1__Exit
- dass, wenn ein innerer Zustand eines zusammengesetzten Zustands mit einer Region (Anwendungsfall Gruppierung/Verschachtelung) aktiviert wird und wenn damit auch der zusammengesetzte Zustand aktiviert wird, die Aufrufreihenfolge „von außen nach innen“ lautet: äußerer Zustand EXIT, zusammengesetzter Zustand ENTRY, innerer Zustand ENTRY, zusammengesetzter Zustand DO, innerer Zustand DO. Im Beispielprojekt:
  - State1__Exit
  - Comp1_Entry
  - State2__Entry
  - Comp1_Do
  - State2__Do
- dass ein innerer zusammengesetzter Zustand über eine (Pseudo-) Ausnahmetransition des äußeren zusammengesetzten Zustands verlassen werden kann (im Beispielprojekt über die Transitionen: eLeaveComp1_ToState1, eLeaveComp1_ToComp3).
- dass, wenn ein zusammengesetzter Zustand z.B. über eine (Pseudo-) Ausnahmetransition verlassen wird, die Aufrufreihenfolge „von innen nach außen“ lautet: innerer Zustand EXIT, ggf. innerer zusammengesetzter Zustand EXIT, äußerer zusammengesetzter Zustand EXIT, äußerer Zustand ENTRY. Im Beispielprojekt:
  - State4__Exit
  - Comp2_Exit
  - Comp1_Exit
  - State1__Entry / State7__Entry
- dass bei einem zusammengesetzten Zustand mit mehreren Regionen zuerst die Region mit der höheren Priorität, d.h. mit der niedrigeren Prioritätszahl, aufgerufen wird. Im Beispielprojekt:
  - State7__Entry
  - State7__Do

- State6__Entry
- State6__Do

Taskzyklus-IDs => Zuordnung des Aufrufs zur Zyklus-ID

Jede ENTRY-/DO-/EXIT-Aktion, die in einem Zustand oder in einem zusammengesetzten Zustand aufgerufen wird, speichert die Zyklus-ID des jeweils ersten und letzten Aufrufs. Dadurch wird veranschaulicht, in welchem Zyklus welcher Aufruf stattfindet, sodass ersichtlich ist:

- dass, wenn ein Zustand aktiviert wird, die ENTRY- und die DO-Aktion innerhalb desselben Zyklus aufgerufen werden.
  Im Screenshot „Visu_B_CallingOrder_CycleNr“ des ZIPs:
  - State1 / ENTRY-Call: 516 (ID des Zyklus, in dem der Aufruf stattfand)
  - State1 / DO-First Call: 516
- dass, wenn ein Zustand verlassen wird, die EXIT-Aktion im darauffolgenden Zyklus des letzten DO-Aufrufs aufgerufen wird.
  Im Screenshot „Visu_B_CallingOrder_CycleNr“ des ZIPs:
  - State1 / DO-Last Call: 1415
  - State1 / EXIT-Call: 1416
- dass, wenn ein Zustand verlassen wird, die EXIT-Aktion dieses Zustands im selben Zyklus aufgerufen wird wie die ENTRY- und DO-Aktion des aktivierten Zustands.
  Im Screenshot „Visu_B_CallingOrder_CycleNr“ des ZIPs:
  - State2 / DO-Last Call: 1762
  - State2 / EXIT-Call: 1763
  - State3 / ENTRY-Call: 1763
  - State3 / DO-First Call: 1763
- dass, wenn ein Zustand verlassen und ein zusammengesetzter Zustand mit einer Region und eigenen Aktionen aktiviert wird, die EXIT-Aktion des verlassenen Zustands im selben Zyklus aufgerufen wird wie die ENTRY- und DO-Aktion des aktivierten (zusammengesetzten) Zustands.
  Im Screenshot „Visu_B_CallingOrder_CycleNr“ des ZIPs:
  - State1 / DO-Last Call: 1415
  - State1 / EXIT-Call: 1416
  - CompositeState1 / ENTRY-Call: 1416
  - State2 / ENTRY-Call: 1416
  - CompositeState1 / DO-First Call: 1416
  - State2 / DO-First Call: 1416
- dass, wenn ein zusammengesetzter Zustand mit einer Region und eigenen Aktionen über eine **Pseudo-Ausnahmetransition** verlassen wird, die EXIT-Aktion des verlassenen (zusammengesetzten) Zustands im selben Zyklus aufgerufen wird wie die ENTRY- und DO-Aktion des aktivierten Zustands.
  Im Screenshot „Visu_B_CallingOrder_CycleNr“ des ZIPs:
  - CompositeState1 / DO-Last Call: 2367
  - CompositeState2 / DO-Last Call: 2367
  - State4 / DO-Last Call: 2367
  - State4 / EXIT-Call: 2368
  - CompositeState2 / EXIT-Call: 2368
  - CompositeState1 / EXIT-Call: 2368
  - State7 / ENTRY-Call: 2368
  - State7 / DO-First Call: 2368
  - State6 / ENTRY-Call: 2368
  - State6 / DO-First Call: 2368

- dass, wenn ein zusammengesetzter Zustand mit einer Region und eigenen Aktionen über eine **Ausnahmetransition** verlassen wird, die EXIT-Aktion des verlassenen (zusammengesetzten) Zustands im selben Zyklus aufgerufen wird wie der letzte Aufruf der DO-Aktion des verlassenen (zusammengesetzten) Zustands und wie die ENTRY- und DO-Aktion des aktivierten Zustands. Zum Beispiel (**nicht** im Screenshot „Visu_B_CallingOrder_CycleNr“ des ZIPs dargestellt):
  - CompositeState1 / DO-Last Call: 1050
  - CompositeState2 / DO-Last Call: 1050
  - State4 / DO-Last Call: 1050
  - State4 / EXIT-Call: 1050
  - CompositeState2 / EXIT-Call: 1050
  - CompositeState1 / EXIT-Call: 1050
  - State1 / ENTRY-Call: 1050
  - State1 / DO-First Call: 1050

Visualisierung

Ergänzend zu dem komfortablen Online-Modus [▶ 91] des UML SC Diagramms können das Verhalten der Zustandsmaschine sowie die Werte der Zähler, der Tabelle und der Taskzyklus-IDs über die Visualisierungen verfolgt werden. Außerdem befinden sich auf der Visualisierung Schalter, um zwischen den Zuständen hin- und herzuschalten sowie um die Werte zurückzusetzen.

Um den Einstieg in das Beispielprojekt zu erleichtern, sind zwei Visualisierungen mit einem unterschiedlichen Informationsgehalt enthalten:

- Visu_A_CallingOrder enthält:
  - Zähler => Aufrufhäufigkeit
  - Eintragung der aufgerufenen Zustandsaktion in ein Array => Aufrufreihenfolge
- Visu_B_CallingOrder_CycleNr erweitert Visu_A um:
  - Taskzyklus-IDs => Zuordnung des Aufrufs zur Zyklus-ID

Die Werte der Transitionsbedingungen können Sie nicht nur über die Visualisierung oder über den Deklarationseditor des FBs, sondern auch direkt über das UML SC Diagramm verändern.

Falls eine Transition aus nur einer Transitionsvariablen besteht, kann der Wert der Variablen per Doppelklick auf den Monitoring-Wert fürs Schreiben oder Forcen verändert werden. Ist ein Wert fürs Schreiben oder Forcen vorbereitet worden, ist dies an der oberen rechten Ecke des Monitoring-Bereichs sichtbar (siehe auch: Online-Modus [▶ 91]).

9.2.5 5 Aufrufverhalten - Transitionsaktion

Aufbauendes Beispiel

Dieses „UML Statechart“-Beispiel basiert auf dem Beispiel 4 Aufrufverhalten - Basis [▶ 100]. Alle Elemente und Aufrufaspekte des Beispiels 4 sind auch in diesem Beispiel enthalten, sodass auch die dortigen Erläuterungen für dieses Beispiel gelten und für das Verständnis dieses Beispiels erforderlich sind. Im Folgenden sind lediglich die Ergänzungen zu Beispiel 4 beschrieben. Bitte lesen Sie daher zuerst die Beschreibung von Beispiel 4.

Ergänzend zu Beispiel 4 Aufrufverhalten - Basis [▶ 100] verdeutlicht dieses „UML Statechart“-Beispiel einen weiteren Aspekt des Aufrufverhaltens vom UML-Zustandsdiagramm [▶ 55]. Dazu sind die folgenden UML SC-Elemente zusätzlich enthalten:

- Transitionen mit Transitionsaktionen (Transition [▶ 82], Abschlusstransition [▶ 85], Ausnahmetransition [▶ 86])

Beispielprojekt: https://infosys.beckhoff.com/content/1031/TF1910_Tc3_UML/Resources/11716023435.zip

Überblick:

Mithilfe der folgenden Teilaspekte wird das Aufrufverhalten von Zuständen, zusammengesetzten Zuständen und Transitionen veranschaulicht. Mithilfe von verschiedenen Programmiermitteln werden die unterschiedlichen Aufrufaspekte veranschaulicht (Programmiermittel => Aufrufaspekt).

- Zähler => Aufrufhäufigkeit
- Eintragung der aufgerufenen Zustandsaktion in ein Array => Aufrufreihenfolge
- Taskzyklus-IDs => Zuordnung des Aufrufs zur Zyklus-ID

Die jeweilige Zuordnung des Programmiermittels zum Aufrufaspekt wird im Folgenden erläutert.

Zähler => Aufrufhäufigkeit:

Eine Transitionsaktion wird einmal aufgerufen, wenn der Zustandsübergang über die Transition ausgeführt wird.

Eintragung der aufgerufenen Zustandsaktion in ein Array => Aufrufreihenfolge:

Jede Aktion, die in einem Zustand, einem zusammengesetzten Zustand oder einer Transition aufgerufen wird, trägt ihren Namen (z. B. „State1__Entry“ oder „Transition_LeaveState1“) in ein Array ein. Dadurch wird die Aufrufreihenfolge der verschiedenen Aktionen veranschaulicht, sodass ersichtlich ist:

- dass die prinzipielle Aufrufreihenfolge bei einem Zustand lautet: erst ENTRY, DO, EXIT und dann die Transitionsaktion. Anschließend wird die ENTRY-Aktion des neu aktivierten Zustands aufgerufen. Im Beispielprojekt:
  - State2__Exit
  - Transition_LeaveState2
  - State3__Entry
- dass, wenn ein innerer Zustand eines zusammengesetzten Zustands mit einer Region (Anwendungsfall Gruppierung/Verschachtelung) aktiviert wird und wenn damit auch der zusammengesetzte Zustand aktiviert wird, die Aufrufreihenfolge „von außen nach innen“ lautet: äußerer Zustand EXIT, Transitionsaktion, zusammengesetzter Zustand ENTRY, innerer Zustand ENTRY, zusammengesetzter Zustand DO, innerer Zustand DO. Im Beispielprojekt:
  - State1__Exit
  - Transition_LeaveState1
  - Comp1_Entry
  - State2__Entry
  - Comp1_Do
  - State2__Do
- dass, wenn ein zusammengesetzter Zustand z. B. über eine (Pseudo-) Ausnahmetransition verlassen wird, die Aufrufreihenfolge „von innen nach außen“ lautet: innerer Zustand EXIT, ggf. innerer zusammengesetzter Zustand EXIT, äußerer zusammengesetzter Zustand EXIT, Transitionsaktion, äußerer Zustand ENTRY. Im Beispielprojekt:
  - State4__Exit
  - Comp2_Exit
  - Comp1_Exit
  - Transition_LeaveComp1_ToComp3
  - State7__Entry

Taskzyklus-IDs => Zuordnung des Aufrufs zur Zyklus-ID:

Jede Aktion, die in einem Zustand, einem zusammengesetzten Zustand oder einer Transition aufgerufen wird, speichert die Zyklus-ID des jeweils ersten und letzten Aufrufs. Dadurch wird veranschaulicht, in welchem Zyklus welcher Aufruf stattfindet, sodass ersichtlich ist:

- dass der Zustandsübergang im darauffolgenden Zyklus ausgeführt wird, nachdem die Transitionsbedingung erfüllt ist,
  und dass, wenn ein Zustand verlassen wird, die Transitionsaktion im selben Zyklus aufgerufen wird wie die EXIT-Aktion des verlassenen Zustands und wie die ENTRY- und DO-Aktion des aktivierten Zustands.
  Im Screenshot „Visu_CallingOrder_CycleNr“ des ZIPs:
  - State2 / DO-Last Call: 1233 (ID des Zyklus, in dem der Aufruf stattfand)
  - Transition / Rising Edge: 1233
  - State2 / EXIT-Call: 1234
  - Transition-Call: 1234
  - State3 / ENTRY-Call: 1234
  - State3 / DO-First Call: 1234
- dass, wenn ein Zustand verlassen und ein zusammengesetzter Zustand mit einer Region und eigenen Aktionen aktiviert wird, die Transitionsaktion im selben Zyklus aufgerufen wird wie die EXIT-Aktion des verlassenen Zustands und wie die ENTRY- und DO-Aktion des aktivierten (zusammengesetzten) Zustands.
  Im Screenshot „Visu_CallingOrder_CycleNr“ des ZIPs:
  - State1 / DO-Last Call: 1055
  - Transition / Rising Edge: 1055
  - State1 / EXIT-Call: 1056
  - Transition / Call: 1056
  - CompositeState1 / ENTRY-Call: 1056
  - State2 / ENTRY-Call: 1056
  - CompositeState1 / DO-First Call: 1056
  - State2 / DO-First Call: 1056
- dass, wenn ein zusammengesetzter Zustand mit einer Region und eigenen Aktionen deaktiviert wird, die Transitionsaktion im selben Zyklus aufgerufen wird wie die EXIT-Aktion des verlassenen (zusammengesetzten) Zustands und wie die ENTRY- und DO-Aktion des aktivierten Zustands.
  Im Screenshot „Visu_CallingOrder_CycleNr“ des ZIPs:
  - CompositeState1 / DO-Last Call: 1743
  - CompositeState2 / DO-Last Call: 1743
  - State4 / DO-Last Call: 1743
  - Transition / Rising Edge: 1743
  - State4 / EXIT-Call: 1744
  - CompositeState2 / EXIT-Call: 1744
  - CompositeState1 / EXIT-Call: 1744
  - Transition / Call: 1744
  - State7 / ENTRY-Call: 1744
  - State7 / DO-First Call: 1744
  - State6 / ENTRY-Call: 1744
  - State6 / DO-First Call: 1744

Visualisierung:

Ergänzend zu dem komfortablen Online-Modus [▶ 91] des UML SC-Diagramms können das Verhalten der Zustandsmaschine sowie die Werte der Zähler, der Tabelle und der Taskzyklus-IDs über die Visualisierung verfolgt werden. Außerdem befinden sich auf der Visualisierung Schalter, um zwischen den Zuständen hin- und herzuschalten sowie um die Werte zurückzusetzen.

Die Werte der Transitionsbedingungen können Sie nicht nur über die Visualisierung oder über den Deklarationseditor des FBs, sondern auch direkt über das UML SC Diagramm verändern.

Falls eine Transition aus nur einer Transitionsvariablen besteht, kann der Wert der Variablen per Doppelklick auf den Monitoring-Wert für das Schreiben oder Forcen verändert werden. Ist ein Wert für das Schreiben oder Forcen vorbereitet worden, ist dies an der oberen rechten Ecke des Monitoring-Bereichs sichtbar (siehe auch: Online-Modus [▶ 91]).

Mehr Informationen:
www.beckhoff.com/tf1910

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.com
www.beckhoff.com