Plant Simulation
Finite State Machine library
Reference manual

TECNOMATIX

SIEMENS

Hinweise zu Eigentumsrechten

TECNOMATIX
Plant Simulation

SIEMENS

Plant Simulation

Finite State Machine Bibliothek

Version 9.0

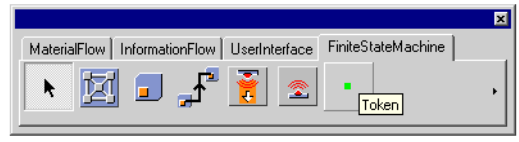
Dezember 2008

Inhaltsverzeichnis

1

Endliche Automaten

Die Objekte aus der Bibliothek *FSM* (Finite State Machines: Endliche Zustandsautomaten) ermöglichen die Modellierung von komplexen Materialflusssystemen, die beispielsweise durch Industrieroboter realisiert werden.



Die Objekte der Bibliothek *FSM*

Grundbegriffe

In diesem Kapitel wird eine kurze Einführung zu "Endlichen Automaten" (*FSM*) gegeben. Die in der Fachliteratur beschriebenen Konzepte für Endliche Automaten wurden durch zusätzliche Funktionalitäten erweitert, die Ihnen die Modellierung von dynamischen Materialflusssystemen erleichtern.

Zustände und Signale

Eine *FSM* empfängt Signale von anderen *FSM*en oder von Materialflussobjekten, die durch einen zugeordneten Sensor beobachtet werden (Eingabe der *FSM*). Die *FSM* verarbeitet diese Signale und beeinflusst den Materialfluss durch Signale an ein Objekt, das eine entsprechende Aktionen auslöst (Ausgabe der *FSM*).

Eine *FSM* kann verschiedene Zustände einnehmen. Zu einem Zeitpunkt hat die *FSM* einen eindeutig identifizierbaren Zustand, der durch das bewegliche Element *Token* auf einem Objekt *State* (Zustand) dargestellt wird. Zu Beginn der Simulation befindet sich die *FSM* im Zustand *Start*. Ein Endzustand ist in der Bibliothek *FSM* nicht vorgesehen. Die *FSM* kann von einem Zustand in einen anderen wechseln, wenn eine oder mehrere Bedingungen erfüllt sind (Überführungsfunktion der *FSM*). Eine Bedingung wird durch das Auftreten von Signalen bestimmt. Immer wenn eine *FSM* ein Signal empfängt, prüft sie, ob sich dadurch ihr Zustand ändern kann.

Entsprechend der Beständigkeit von Signalen unterscheiden wir die *Signaltypen*:

1. *permanent*: Wenn die *FSM* in einen neuen Zustand übergeht, können einige oder alle permanenten Signale gelöscht werden, die als Bedingung für diesen Zustandswechsel benötigt wurden.
2. *persistent*: Ein persistentes Signal bleibt immer bestehen. Es kann *aktiv* (d.h. gültig) oder *inaktiv* (d.h. ungültig) sein.
3. *impulse*: Ein Impulssignal wird nach der Überprüfung, ob dieses Signal Bestandteil einer Bedingung ist, sofort gelöscht. Diese Signale werden auch dann gelöscht, wenn sie nicht Bestandteil der Bedingung für den nächsten Zustandswechsel sind.

Senden von Signalen

Wechselt eine *FSM* den Zustand, so wechselt das *Token* das Objekt *State* und es können Signale an andere *FSM* gesendet und/oder Aktionen ausgelöst werden.

Tritt ein bestimmtes Materialflussereignis auf einem Grundobjekt (z.B. eine *Einzelstation*) ein, so werden von dem Sensorobjekt *SensorCtrl* Signale an die Objekte *FSM* und/oder *ActorCtrl* gesendet. Das Objekt *Sensor* reagiert auf folgende Ereignisse:

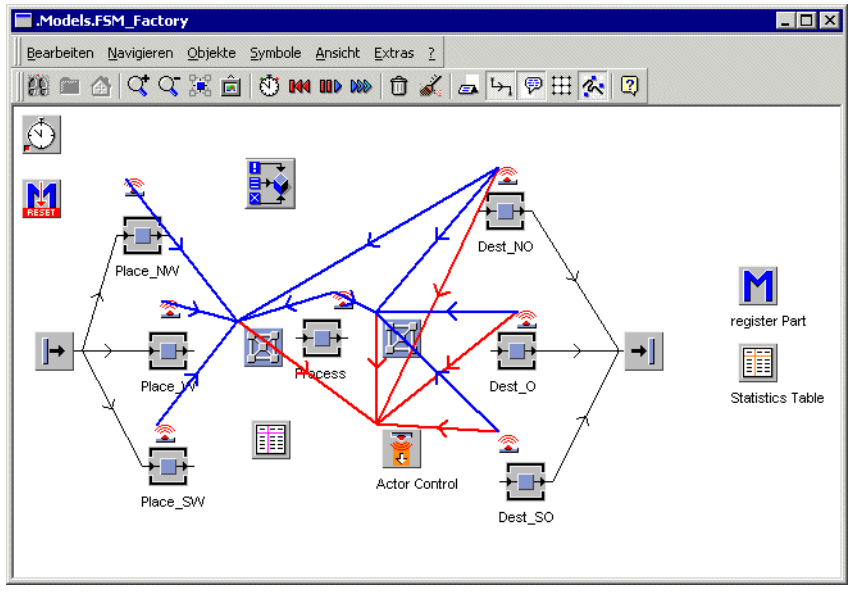
- *empty*: Das Materialfluss-Objekt wurde leer.
- *exit*: Ein bewegliches Element hat eben das Objekt verlassen.

- *entrance*: Ein bewegliches Elemente ist eben eingetreten.
- *ready*: Die Bearbeitung eines beweglichen Elementes wurde abgeschlossen.

Empfangen von Signalen

Eine *FSM* registriert alle empfangene Signale. Nach der Überprüfung, ob ein Zustandswechsel möglich ist, werden Signale entsprechend ihrer Beständigkeit gelöscht.

Soll durch ein Signal eine Aktion ausgelöst werden, so wird das Signal an ein Objekt *ActorCtrl* gesendet. Das Objekt *ActorCtrl* führt entsprechend dem Signal eine Aktion aus, die zu einem Materialflussereignis, einer Animation (Iconumschaltung) oder zu einem Methodenaufruf führen kann.



Animation der Signale mit besonderer Kennzeichnung der Signale an dem Objekt *ActorCtrl*

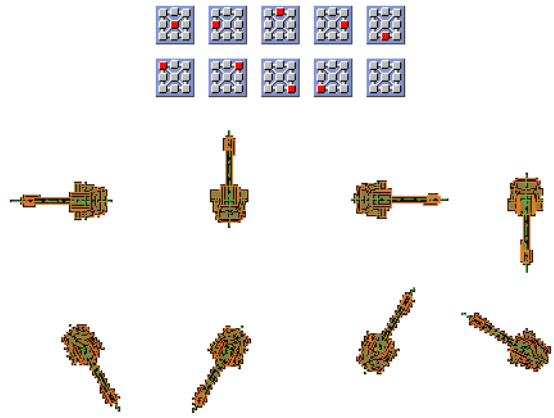
Bestandteile der FSM

Nach der Beschreibung der Konzepte und Grundbegriffe der *FSM,* werden nun die Anwendungsobjekte beschrieben. Da für Signale kein Anwenderobjekt vorgesehen ist und die Kommunikation zwischen den Komponenten der *FSM* durch Signale realisiert ist, haben die Anwenderobjekt der *FSM* Menü- und Kontextmenüeinträge Animiere Signale, Zeige Signale und Lösche Animation. Durch Einblenden von Kommentaren und Verbindungen wird der Austausch von Signalen visualisiert. Über den Menüeintrag und den Kontextmenüeintrag Hilfe können Sie für die Anwenderobjekt eine kurze Hilfe öffnen.

FSM (Finite State Machine)

Symbole: 

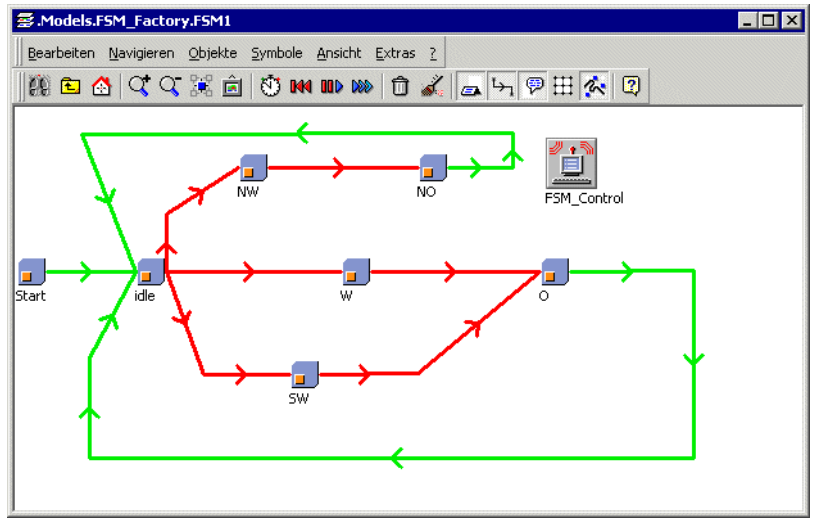
Das Objekt *FSM* stellt einen Automaten dar. In dem Objekt können Sie beliebig viele Zustände (Anwenderobjekt *State*) einsetzen, die die *FSM* einnehmen kann. Zur Animation der verschiedenen Zustände hat die *FSM* die nachfolgend dargestellten Icons. Für spezielle Anwendungen können weitere Icon hinzugefügt werden.



Vorbereitete Icons der *FSM* zur Zustandsanimation

Zu Beginn der Simulation befindet sich die *FSM* im Zustand *Start*. Das entsprechende Anwenderobjekt ist in der Klasse bereits eingesetzt.

Eine *FSM* kann gestört sein. Während dieser Zeit zeigt die *FSM* das Icon mit dem Namen *failed:* 



Zustände, Übergänge einer *FSM*

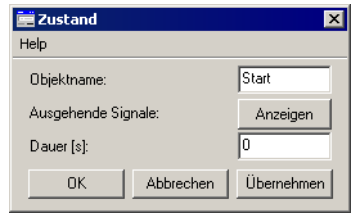
Mit dem Eintreten eines Zustandes lassen sich Aktionen verknüpfen, die durch Signale an den Anwenderobjekt *ActorCtrl* ausgelöst werden.

State

Symbole: 

Dieses Objekt repräsentiert einen Zustand der *FSM*. Sobald das Token in einen neuen Zustand eintritt, können Signale an andere *FSM* gesendet werden. Ebenso können Signale an Objekte *ActorCtrl* gesendet werden, um dadurch Aktionen, wie z. B. das Umlagern von Teilen im Modell, durchzuführen.

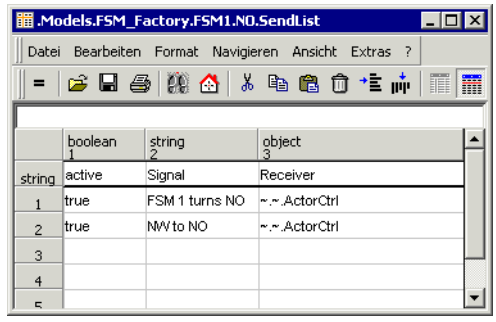
Zur Visualisierung kann ein neues Bild für die *FSM* gesetzt werden.



Dialog des Zustandsobjektes *State*

Objektname: Dieses Element enthält den Namen des Objektes. Dieser Name muss ein gültiger Bezeichner im Netzwerk der *FSM* sein. Lesen Sie dazu in der Plant Simulation Hilfe zu 'Konventionen für Namen'.

Gesendete Signale: Wenn die *FSM* einen neuen Zustand annimmt, können dadurch Signale an andere *FSM* oder an *ActorCtrls* gesendet werden.



Signale, die beim Eintreten eines Zustandes gesendet werden.

In dieser Tabelle haben die Spalten folgende Bedeutung:

1. *active*: In dieser Spalte wird eingestellt, ob das Signal gesetzt (*true*) oder zurückgesetzt (*false*) wird. Diese Spalte hat nur für persistente Signale Bedeutung.
2. *Signal*: In diese Spalte wird der Signalname eingetragen.
3. *Receiver*: In diese Spalte wird das Objekt eingetragen, an den das Signal gesendet werden soll. Tragen Sie hier nur Anwenderobjekte *FSM* oder *ActorCtrl* ein.

Dauer [s]: Nimmt die *FSM* einen neuen Zustand an, so werden zuerst die unter *Gesendete Signale* definierten Sig-nale gesendet. Danach erfolgt die Überprüfung, ob die *FSM* zu einem neuen Zustand wechseln kann. Die Dauer zwischen dem Senden der Signale und der Überprüfung ist standardmäßig 0. Das Objekt zeigt dann das erste Icon.

Zur zeitdynamischen Simulation ist es aber unter Umständen wichtig, dass eine gewisse Dauer zwischen diesen Ereignissen liegt. Diese Dauer wird durch die Verweildauer des Tokens auf dem Zustand abgebildet, die Sie hier einstellen können. Ein Zustand mit Zeitverbrauch wird durch das zweite Icon dargestellt.

signal_connector

Symbol: 

Damit die *FSM* von einem Zustand in einen anderen Zustand übergehen kann, müssen Sie diese Zustände mit dieser Kante verbinden. Der Übergang von einem Zustand zu einem anderen Zustand wird durch eine Bedingung beschrieben, die durch eine ODER-Verknüpfung von UND-Verknüpfungen von Signalen beschrieben wird. Diese Bedingungen ist der Kante zuordnen. Nach dem Verbinden können Sie mit einem Doppelklick eine Tabelle öffnen, in der Sie die Bedingungen eintragen können. Bei einer UND-Verknüpfung werden die Signalnamen mit ‚&' verknüpft, bei einer ODER-Verknüpfung werden die Signalnamen bzw. UND-Verknüpfungen in unterschiedliche Zeilen eingetragen. Für die Bedingung

(*Signal_A* UND *Signal_B*) ODER (*Signal_1* UND *Signal_2*)

müssen Sie diese Tabelle folgendermaßen füllen:

	string 1	integer 2
string	signal	priority
1	Signal_A&Signal_B	2
2	Signal_1&Signal_2	3
3		

Bedingungstabelle zu einer Verbindung zwischen zwei Zuständen

In dieser Tabelle haben die Spalten folgende Bedeutung:

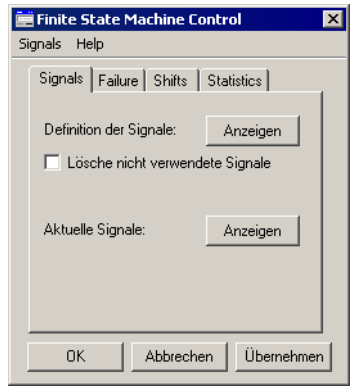
1. *signal*: In dieser Spalte wird eine UND-Verknüpfung von Signalen eingetragen.
2. *priority*: Durch die Priorität legen Sie die Reihenfolge der Auswertung der UND-Verknüpfungen fest.

Die Kante ist rot, wenn eine Bedingung definiert wurde, ohne Bedingungen sind Kanten grün.

FSM_control

Symbol: 

Dieses Objekt steuert die *FSM* und bietet verschiedene Einstellungen an. Das Objekt befindet sich standardmäßig in der *FSM*.



Die Registerkarte Signals der *FSM*-Steuerung

Definition der Signale: In dieser Tabelle stellen Sie die Beständigkeit der Signale ein. Voreinstellt ist der Wert *permanent*.

	string 1	string 2
string	signal	type
1	Process ready	permanent
2	O empty	persistent
3	SO empty	impulse
4	NO empty	permanent
5		
6		permanent
7		persistent
		impulse

Definition der Beständigkeit der Signale

In dieser Tabelle haben die Spalten folgende Bedeutung:

1. *signal*: Die Einträge dieser Spalte werden bei einer ersten Simulation automatisch eingestellt.
2. *type*: Wählen Sie hier die Beständigkeit der Signale aus.

Lösche nicht verwendete Signale: Der Übergang von einem Zustand zu einen anderen Zustand ist von einer Bedingung abhängig, die durch eine ODER-Verknüpfung von UND-Verknüpfungen von Signalen beschrieben wird. Nach einer Zustandsänderung werden permanente Signale gelöscht, die in ODER-Verknüpfungen enthalten sein können. Mit diesem Kontrollkästchen können Sie einstellen, ob alle permanenten Signale einer Bedingung oder nur die Signale der erfüllten UND-Verknüpfung gelöscht werden sollen. Werden alle Signale gelöscht, so werden auch die nicht verwendeten Signale gelöscht, die in eventuell nicht erfüllten oder nicht untersuchten UND-Verknüpfungen enthalten sind.

Betrachten wir das folgende Beispiel einer Bedingung von permanenten Signalen:

(*Signal_A* UND *Signal_B*) ODER (*Signal_1* UND *Signal_2*)

Ist dieses Kontrollkästchen ausgewählt, so werden nach einem Übergang alle permanenten Signale gelöscht, die für diesen Übergang in der Bedingung enthalten sind. Sind in unserem Beispiel die Signale *Signal_A*, *Signal_B* und *Signal_1* vorhanden, so werden alle drei Signale nach einer Zustandsänderung gelöscht.

Ist dieses Kontrollkästchen nicht ausgewählt, so werden nur die Signale nach dem Übergang gelöscht, die in der erfüllten UND-Verknüpfung enthalten sind. Sind in unserem Beispiel *Signal_A*, *Signal_B* und *Signal_1* vorhanden, so werden nur die Signale *Signal_A* und *Signal_B* gelöscht. Das nicht verwendete Signal *Signal_1* wird nun nicht gelöscht.

Aktuelle Signale: In dieser Tabelle werden die zur aktuellen Simulationszeit vorhandenen Signale verwaltet.

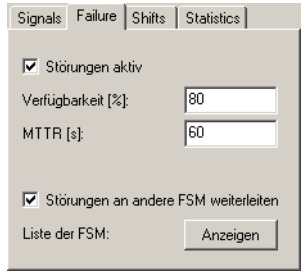
	string 1	object 2	boolean 3	string 4
string	signal	sender	active	type
1	Process empty	~.~.ProcSensor	true	permanent
2	W occupied	~.~.Sensor_W	true	permanent
3	SW occupied	~.~.SensorSW	true	permanent
4	W ready	~.~.Sensor_W	true	permanent
5				

Zum aktuellen Zeitpunkt vorhandenen Signale.

In dieser Tabelle haben die Spalten folgende Bedeutung:

1. *signal*: In dieser Spalte ist der Name des Signals enthalten.
2. *sender*: Dieses Objekt hat dieses Signal an die *FSM* geschickt.
3. *active*: Ist ein persistentes Signal aktiv (d.h. gültig), so ist hier *true* eingetragen.
4. *type*: Die Beständigkeit des Signals ist hier eingetragen.

Zur Simulation von Störungen verwenden Sie die Einstellungen auf der Registerkarte Failure.



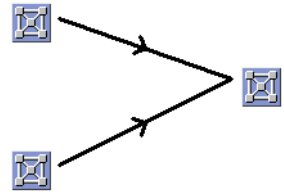
Verfügbarkeit der *FSM*

Störungen aktiv: Mit diesem Kontrollkästchen stellen Sie ein, ob die *FSM* gestört werden soll. Während einer Störung verbleibt die *FSM* in dem aktuellen Zustand. Es gibt keinen gesonderten Störungszustand. Eine gestörte *FSM* empfängt weiterhin alle Signale, ändert aber nicht ihren Zustand.

Verfügbarkeit [%], MTTR: Geben Sie hier die gewünschte Verfügbarkeit und die mittleren Stördauer (MTTR) an.

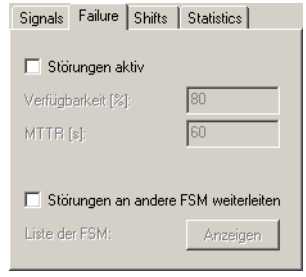
Störungen an andere FSM weiterleiten: Sobald die *FSM* gestört wird, kann diese Störung an andere *FSM* weitergeleitet werden. Die auf der Registerkarte Failure eingestellte Verfügbarkeit der in dieser Liste eingetragenen *FSM* wird sich dadurch in der Simulation verringern.

Liste der FSM: In dieser Liste geben Sie die *FSM*s an (auch mit Drag & Drop), an welche die Weiterleitung von Störungen erfolgen soll. Diese *FSM*s werden zu sogenannten Schutzkreisen zusammengefasst. Diese Liste kann auch über den Kontextmenüeintrag Störungen weiterleiten geöffnet werden.



Weiterleitung von Störungen

Die *FSM*, die auf diese Weise durch andere *FSM* gestört wird, ist gestört, wenn eine eigene Störung (d.h. durch den internen Störgenerator) oder eine Störung einer *FSM* aus dieser Liste auftritt. Die Weiterleitung der Störungen kann durch den Kontextmenüeintrag Animiere Schutzkreise visualisiert werden.

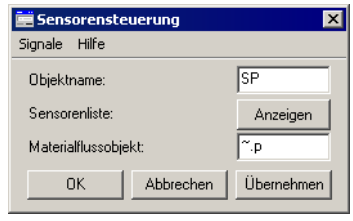


FSM ohne Erzeugung eigener Störungen

Auf der Registerkarte Shifts können Sie einen Kalenderobjekt für Schichtsysteme eintragen. Alternativ können Sie ein Kalenderobjekt auf ein Objekt SensorCtrl

Symbol: 

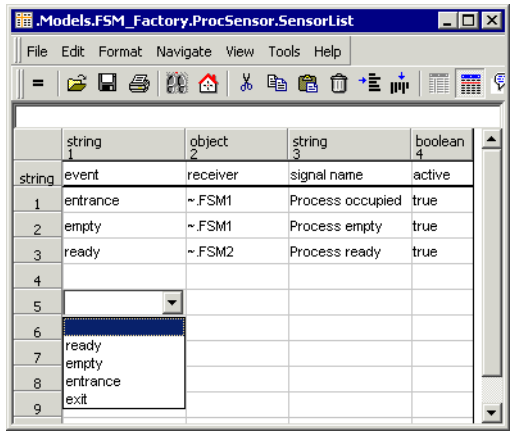
Dieses Objekt wird an ein Grundobjekt geheftet und kann Signale an *FSM*en und/oder *ActorCtrl* senden, sobald sich der Zustand des Grundobjektes ändert. Die Sensorsteuerung ist keiner *FSM* fest zugeordnet, sondern kann an mehrere *FSM* Signale senden.



Sensor an einer Einzelstation *P*

Objektname: Dieses Element enthält den Namen des Objektes. Dieser Name muss ein gültiger Bezeichner im Netzwerk sein. Lesen Sie dazu in der Plant Simulation Hilfe zu 'Konventionen für Namen'.

Sensorenliste: In dieser Tabelle wird festgelegt, bei welchen Materialflussereignissen Signale gesendet werden.



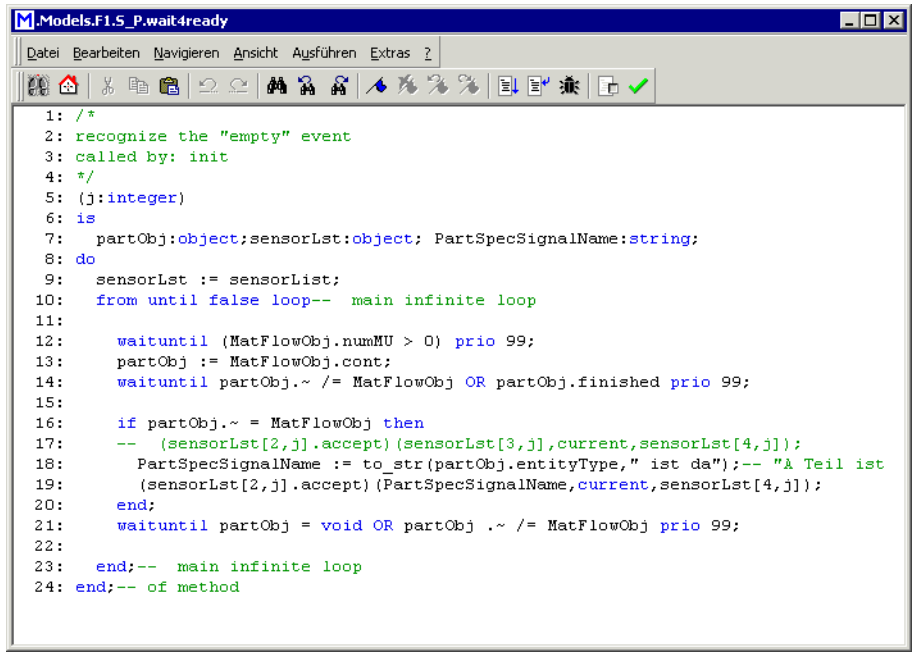
Sensorenliste des Objektes *SensorCtrl*

Die Spalten haben folgende Bedeutung:

1. event: In dieser Spalte wird das Ereignis eines beweglichen Elements auf dem angehefteten Grundobjekt definiert. Die Einträge haben folgende Bedeutung:
   - *empty*: Das Objekt wurde leer.
   - *exit*: Ein bewegliches Element hat eben das Objekt verlassen.
   - *entrance*: Ein bewegliches Elemente ist eben eingetreten.
   - *ready*: Die Bearbeitung eines beweglichen Elementes wurde abgeschlossen.
2. receiver: In dieser Spalte wird die *FSM* oder das Objekt *ActorCtrl* eingetragen, an die das Signal gesendet wird.
3. signal name: In diese Spalte wird der Signalname eingetragen, welcher gesendet wird.
4. active: In dieser Spalte wird eingestellt, ob das Signal gesetzt (Wert *true*) oder zurückgesetzt (Wert *false*) wird. Diese Einstellung wird nur für persistente Signale verwendet.

Materialflussobjekt: In diesem Dialogfeld wird das Materialflussobjekt eingetragen, das von dem Objekt *SensorCtrl* überwacht wird. Der Eintrag kann auch via Drag & Drop des Objekts auf das Objekt *SensorCtrl* erfolgen. Das Objekt *SensorCtrl* positioniert sich nach dem Fallenlasssen über das Materialflussobjekt.

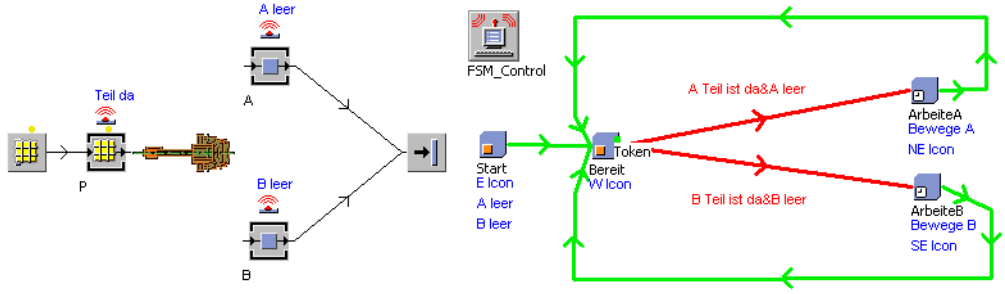
In diesem Abschnitt wird beschrieben, wie das Objekt *SensorCtrl* Signale versenden kann, die von Attributen des beweglichen Elementes abhängen. Diese Erweiterung sollten nur erfahrene Simulationsanwender vornehmen. Das Objekt *SensorCtrl* kann nur Signale an andere *FSM*en und/oder *ActorCtrl* senden. Diese beiden Objekte besitzen eine Methode *accept*, die drei Parameter erwartet. Der erste Parameter ist der Name des Signals, das gesendet wird, der normalerweise aus der dritten Spalte der Sensorenliste entnommen wird.




```
 1: /*
 2: recognize the "empty" event
 3: called by: init
 4: */
 5: (j:integer)
 6: is
 7:   partObj:object;sensorLst:object; PartSpecSignalName:string;
 8: do
 9:   sensorLst := sensorList;
10:   from until false loop--  main infinite loop
11:
12:     waituntil (MatFlowObj.numMU > 0) prio 99;
13:     partObj := MatFlowObj.cont;
14:     waituntil partObj.~ /= MatFlowObj OR partObj.finished prio 99;
15:
16:     if partObj.~ = MatFlowObj then
17:     --  (sensorLst[2,j].accept)(sensorLst[3,j],current,sensorLst[4,j]);
18:       PartSpecSignalName := to_str(partObj.entityType," ist da");-- "A Teil ist
19:       (sensorLst[2,j].accept)(PartSpecSignalName,current,sensorLst[4,j]);
20:     end;
21:     waituntil partObj = void OR partObj .~ /= MatFlowObj prio 99;
22:
23:   end;--  main infinite loop
24: end;-- of method
```

Erweiterung der Methode *wait4Ready* der Sensorsteuerung *S_P*

In der abgebildeten Methode wurde die Vererbung abgeschaltet, die neue Stringvariable *PartSpecSignalName* deklariert, ein Signalname konstruiert, der von dem Attribut *entityType* des beweglichen Elementes abhängt, und der Aufruf der Methode *accept* in Zeile 19 modifiziert. In Zeile 17 sehen Sie den ursprünglichen Auruf Methode *accept*. Der Sensor an der Station *P* in dem abgebildeten Modell sendet anstatt des Signals *Teil da* die Signale *A Teil ist da* oder *B Teil ist da*. In Abhängigkeit von dem Signal lagert die *FSM* das Teil auf die Station *A* oder *B* um.

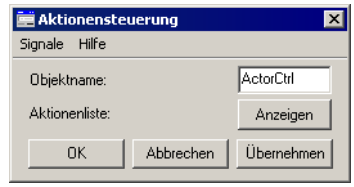


Beispiel einer *FSM*, die attributabhängige Aktionen ausführen kann.

ActorCtrl

Symbol: 

Dieses Objekt empfängt Signale von *FSM* und setzt sie in Aktionen um. Wenn die *FSM* eine Aktion durchführen möchte, sendet die *FSM* ein Signal an eine *ActorCtrl*. In der *ActorCtrl* muss für jedes empfangene Signal eine Aktion hinterlegt sein.



Aktionensteuerung *ActorCtrl*

Objektname: Dieses Element enthält den Namen des Objektes. Dieser Name muss ein gültiger Bezeichner im Netzwerk der *FSM* sein. Lesen Sie dazu in der Plant Simulation Hilfe zu 'Konventionen für Namen'.

Aktionenliste: In dieser Tabelle wird definiert, was bei welchen Signalen passieren soll.

	string 1	string 2	string 3	string 4	string 5	string 6
string	signal	action type	parameter 1	parameter 2	parameter 3	parameter 4
8	FSM 1 turns NO	SetIconName	~.FSM1	NO		
9	NW to NO	Move	~.Place_NW	~.Dest_NO		
10	register NO	Method	~.regPart	.Models.FSM_Factor		
11	register SO	Method	~.regPart	.Models.FSM_Factor		
12	register O	Method	~.regPart	.Models.FSM_Factor		
13	W ready	SetIconName	~.FSM1	W		
14	W to Process	Move	~.Place_W	~.Process		
15	FSM 2 turns O	SetIconName	~.FSM2	O		

Aktionenliste des Objektes *ActorCtrl*

In dieser Tabelle haben die Spalten folgende Bedeutung:

1. signal: In diese Spalte wird der Signalname eingetragen, welcher die Aktion anstößt.
2. action type: In dieser Spalte wird der Aktionstyp eingestellt. Die bereitgestellten Aktionstypen werden unten aufgelistet.
3. parameter 1– 4: Abhängig vom eingestellten Aktionstyp geben Sie in diesen Spalten die zugehörigen Parameter an, die bei der Beschreibung der Aktionstypen mit *P1*, ..., *P4* abgekürzt werden.

Folgende Aktionstypen stehen zur Verfügung:

- *Move:* Ein bewegliches Element wird von einem platzorientierten Materialflussobjekt (Parameter *P1*) zu einem zweiten Objekt (Parameter *P2*) umgelagert.
- *LineMove*: Ein bewegliches Element wird von einem Materialflussobjekt (Parameter *P1*) zu der Endposition eines längenorientierten Materialflussobjektes (Parameter *P2*) umgelagert.
- *LineBack*: Die Fahrtrichtung eines Fahrzeugs wird umgekehrt. Als Parameter *P1* geben sie den Standort des Fahrzeuges an.
- *Remove*: Ein bewegliches Element (Parameter *P1*) wird gelöscht.
- *Method*: Eine Methode, deren Pfad der Parameter *P1* ist, wird ausgeführt. Diese Methode muss drei *string*-Parameter haben, die als Parameter *P2* , ..., *P4* übergeben werden können.
- *SetIconName*: Das Icon eines Objektes, das der Parameter *P1* ist, wird umgeschaltet. Der Parameter *P2* ist der Iconname. Optional können Sie die Position der *FSM* ändern. Tragen Sie hierzu die x-Position oder y-Position als Parameter *P3* und *P4* ein.
- *SetIcon*: Die Position eines Objektes, das der Parameter *P1* ist, wird gesetzt. Die Parameter *P2* und *P3* sind die Positionen. Der Parameter *P4* ist der Drehwinkel des Objektes.

Erfahrene Simulationsanwender können weitere Aktionen dem Objekt *ActorCtrl* hinzufügen. Hierzu müssen Sie folgende Schritte durchführen:

Die neue Aktion muss in der Spalte action type der Tabelle *ActionList* in der Klasse des Objektes hinzugefügt werden. Dazu müssen Sie das Format der Spalte action type ändern und hierbei die neue Aktion dem bestehenden Formatstring hinzufügen. In der Methode *getMessage* muss in der *inspect*-Anweisung die neue Aktion hinzugefügt werden. Dabei

kann man sich an der bereits vorhanden Struktur orientieren. Um die Struktur des Objektes *ActorCtrl* beizubehalten, wird für die neue Aktion eine neue Methode eingefügt, die von der Methode *getMessage* mit Parameterübergabe aufgerufen wird.

Unterstützung bei der Modellierung

Die Ideen der Endlichen Zustandsautomaten (*FSM*) sind einfach zu verstehen, aber trotzdem ist die Modellierung von komplexen dynamischen Materialflusssystemen, wie sie z.B. bei Industrierobotern vorliegen, nicht einfach. Die Bibliothek unterstützt Sie bei der Modellierung durch verschieden Werkzeuge zur Modellanalyse. Plausibilitätsüberprüfungen der Tabellen sichern, dass die richtigen Objekte eingetragen sind. Absolute Pfade werden in Tabellen durch relative ersetzt, so dass der Inhalt einfacher zu lesen ist. Bitte beachten Sie, dass diese Tabelleneinträge durch das Umbenennen von Objekten ungültig werden können.

Das Protokollobjekt

Mit Hilfe des Protokollobjektes können Sie die Arbeitsweise der *FSM* beobachten. Setzen Sie zu diesem Zweck das Objekt in das *Root*-Netzwerk ein.

Symbol der Protokollobjektes: 

Lassen Sie sich alle Aktionen mit Simulationszeit registrieren. Öffnen Sie dazu die Tabelle der Einstellungen und tragen Sie in der Zeile regist dreimal *true* ein.

Einstellungen des Protokollobjektes

In der Tabelle Nachrichten, die Sie auf der Registerkarte Lists öffnen können, sehen sie sowohl alle erfolgten Zustandsänderungen als auch auch alle Versuche der *FSM*, den Zustand zu ändern. Alle ausgeführten Aktionen der Objekte *ActorCtrl* werden erfasst.

	string 1
1	0.0000: FSM2: Zustandsänderung 'Start'.
2	0.0000: ActorCtrl: Führe Aktionen zum Signal 'FSM 2 is reset' vom Typ 'SetIconName' aus.
3	0.0000: FSM1: Zustandsänderung 'Start'.
4	0.0000: ActorCtrl: Führe Aktionen zum Signal 'FSM 1 is reset' vom Typ 'SetIconName' aus.
5	0.0000: FSM1: Erhalte Signal 'Process empty' von 'ProcSensor' im Zustand 'Start'.
6	0.0000: FSM1: Zustandsänderung 'idle'.
7	0.0000: ActorCtrl: Führe Aktionen zum Signal 'FSM 1 is idle' vom Typ 'SetIconName' aus.
8	0.0000: FSM2: Der Übergang von 'Start' nach 'PartReady' ist unmöglich.
9	0.0000: FSM2: Erhalte Signal 'O empty' von 'Sensor_O' im Zustand 'Start'.
10	0.0000: FSM2: Der Übergang von 'Start' nach 'PartReady' ist unmöglich.
11	0.0000: FSM2: Erhalte Signal 'SO empty' von 'Sensor_SO' im Zustand 'Start'.
12	0.0000: FSM2: Der Übergang von 'Start' nach 'PartReady' ist unmöglich.
13	0.0000: FSM1: Erhalte Signal 'NO empty' von 'Sensor_NO' im Zustand 'idle'.
14	0.0000: FSM1: Der Übergang von 'idle' nach 'NW' ist unmöglich.

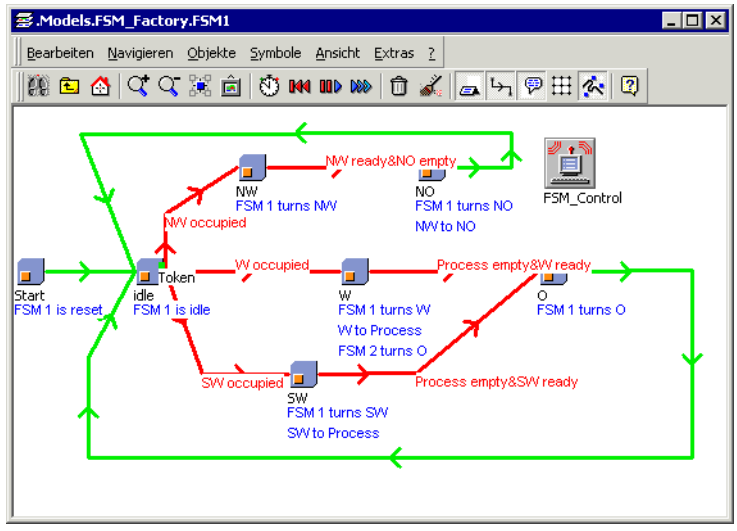
Nachrichtentabelle des Protokollobjektes.

Die Tabellen des Protokollobjektes können Sie auch über das Kontextmenü öffnen.

Animation der Signale

Eine weitere Modellierungshilfe ist die Animation der Signale im Modellnetzwerk, wie es die Abbildung Animation der Signale mit besonderer Kennzeichnung der Signale an dem Objekt ActorCtrl zeigt. Ein Doppelklick auf die farbigen Kanten öffnet eine Tabelle mit den entsprechenden Signalen, die zwischen den Objekten ausgetauscht wurden.

Darüber hinaus können die Signale zu den Zuständen und deren Übergänge im Netzwerk der *FSM* animiert werden. Das Objekt *FSM_Control* hat zu diesem Zwecke entsprechende Menü- und Kontextmenüeinträge Animiere Signale und Lösche Animation. Rot werden die Signale dargestellt, die als Bedingung in den Verbindungen der Zustände eingetragen sind. Blau sehen Sie die Signale, die bei Zustandsänderung versendet werden.



Signale der *FSM* aus der Abbildung Zustände, Übergänge einer FSM

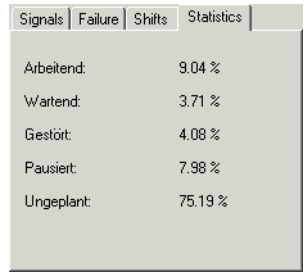
Auswertungen

Eine *FSM* kann sich in folgenden Situationen befinden:

- *Ungeplant*: Außerhalb der Schichtzeiten, die durch das Kalenderobjekt festlegt werden können, ist die *FSM* ungeplant.
- *Pausiert*: Während der Schichtzeiten treten Pausen auf, die durch den Kalenderobjekt festlegt werden können.
- *Gestört*: Durch eigene Störungen oder durch Störungen einer anderen *FSM* in einem Schutzkreis kann eine *FSM* nicht verfügbar sein.
- *Arbeitend*: Enthält die *FSM* Zustände mit Zeitverbrauch und befindet sich das *Token* in Bearbeitung auf einen solchen Zustand, so ist die *FSM* während dieser Zeit arbeitend. Das *Token* wird grün dargestellt.
- *Wartend*: Wenn die Bedingungen für einen Zustandswechsel nicht erfüllt sind, so muss die *FSM* auf die entsprechenden Signale warten. Das *Token* wird gelb dargestellt.

Diese Zeiten werden auf der Regsiterkarte Statistics dargestellt und können auch mit einem Diagrammobjekt veranschaulicht werden. Zu diesem Zweck ziehen Sie einen Diagrammobjekt auf die *FSM*, zu denen Sie diese statistischen Werte veranschaulichen möchten. Es öffnet sich ein Diagramm. Durch Betätigen der Schaltfläche Übernehmen des Dialoges der *FSM*-Steuerung wird die Anzeige der Statistikwerte im Diagramm und auf der Regsiterkarte Statistics aktualisiert. Diesen Dialog können Sie mit dem Kontextmenüeintrag Öffne Steuerung der *FSM* anzeigen.

Beachten Sie, dass sich durch die Weiterleitungen der Störungen die tatsächlichen Verfügbarkeiten und die auf der Registerkarte Failure eingestellten Verfügbarkeiten unterscheiden können.



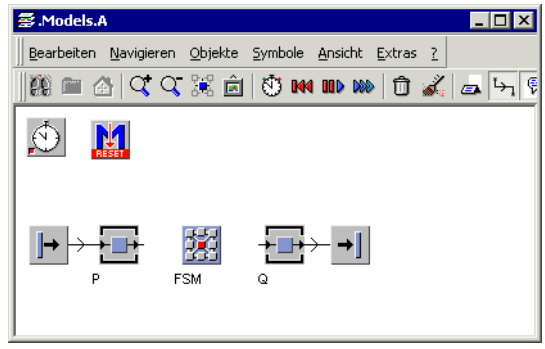
Statistik der *FSM*

Ein einfaches Beispiel

In diesem Kapitel wird eine einfache *FSM* beschrieben, die zwischen zwei Zuständen hin und her pendelt. Dieses Beispiel zeigt die prinzipielle Funktionsweise einer *FSM*.

Das Szenario

Auf einer Einzelstation *P* kommen zu bearbeitende Teile an. Ein Roboter wendet sich *P* zu, übernimmt das Teil und legt es auf einer Einzelstation *Q* wieder ab. Das Teil wird nur von dem Roboter übernommen, wenn *Q* leer. Dadurch ist sichergestellt, dass das Teil auf *Q* abgelegt werden kann. Die Bewegung des Roboters von *Q* zu *P* benötigt eine vernachlässigbare Zeit, da der Roboter nichts zu transportieren hat. Die Bewegung des Teils von *P* zu *Q* benötigt eine Minute.



Ein Roboter soll Teile von *P* nach *Q* transpotieren.

Zustände

Bei der Modellierung eines Roboters erfassen Sie zuerst alle Zustände, die der Roboter einnehmen kann. Der Roboter bleibt aus folgenden zwei Gründen eine bestimmte Zeit in einem Zustand:

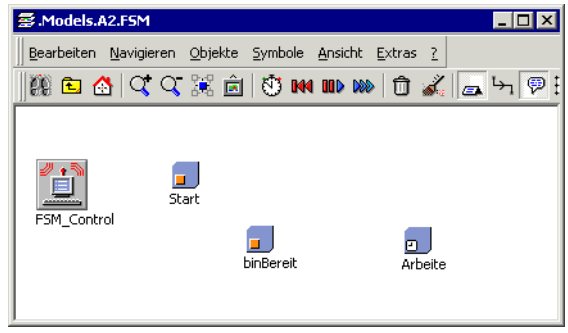
1. Der Roboter führt eine Arbeit aus. In diesem Falle nehmen wir an, dass sich sein Zustand nicht ändert, auch wenn der Roboter während dieser Zeit Bewegungen ausführt.

Für die Modellierung der Arbeitszeit verwenden Sie ein Zustandsobjekt mit Zeitverbrauch. In unserem Beispiel wird der Transport des Teils von *P* zu *Q* durch einen Zustand *Arbeite* beschrieben.

2. Der Roboter wartet auf Signale, um in den nächsten Zustand zu wechseln. Diese Signale zeigen das Auftreten einer bestimmten Situation an.

In unserem Beispiel benötigen wir einen weiteren Zustand *binBereit*, der anzeigt, dass der Roboter weiterarbeiten kann, wenn das nächste Teil auf *P* ankommt. Ist *Q* leer und ist die Arbeit des Roboters beendet, so geht er in den Zustand *binBereit*.

Öffnen Sie das Objekt *FSM* und setzen Sie die Zustände *binBereit* und *Arbeite* ein. In *Arbeite* tragen Sie die Dauer von 60 Sekunden ein.

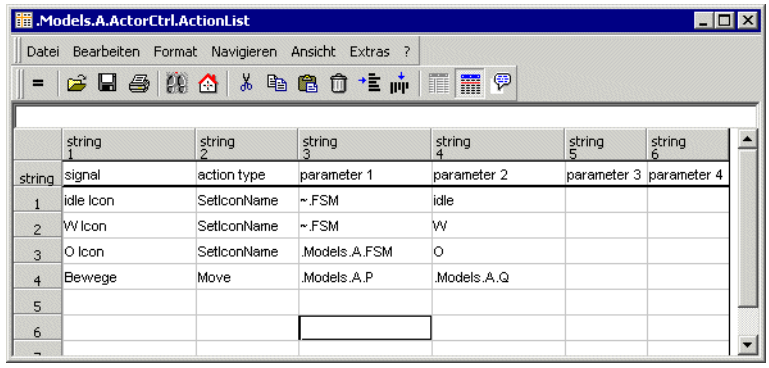


Die Zustände der *FSM*

Signale von FSM lösen Aktionen aus

Beim Eintreten eines Zustandes können bestimmte Aktionen ausgeführt werden. Typischerweise wird das Icon der *FSM* umgeschaltet. Zu diesem Zweck sendet die *FSM S*ignale an das Objekt *ActorCtrl*. Wenn in unserem Beispiel der Zustand *Arbeite* eintritt, so wird das Teil von *P* zu *Q* umgelagert.

Setzen Sie das Objekt *ActorCtrl* in das Root-Netzwerk ein und füllen die Liste der Aktionen wie folgt aus.

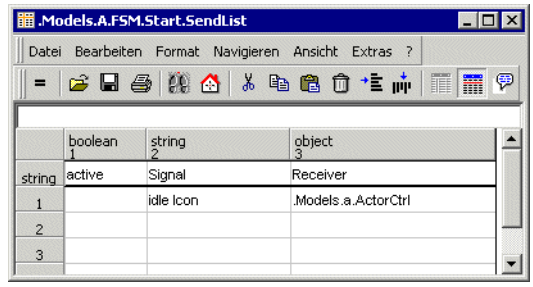


	string 1	string 2	string 3	string 4	string 5	string 6
string	signal	action type	parameter 1	parameter 2	parameter 3	parameter 4
1	idle Icon	SetIconName	~.FSM	idle		
2	W Icon	SetIconName	~.FSM	W		
3	O Icon	SetIconName	.Models.A.FSM	O		
4	Bewege	Move	.Models.A.P	.Models.A.Q		
5						
6						

Liste der Aktionen im Objekt *ActorCtrl*

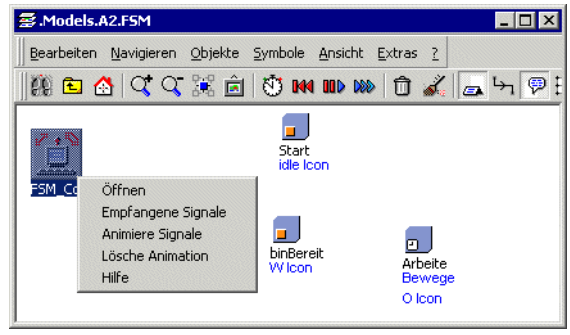
Objekteinträge können durch Drag&Drop vorgenommen werden. Nach dem Schliessen und Öffnen dieser Tabelle werden automatisch relative Pfade eingetragen.

Die in der ersten Spalte eingetragenen Signale werden durch Zustandsobjekte versendet. Öffnen Sie die Tabelle der ausgehenden Signale der Zustandsobjekte. Dazu können Sie den Kontextmenüeintrag Ausgehende Signale des Objektes *State* verwenden.



Ausgehendes Signal, das von dem Zustand *Start* versendet wird.

Zur Kontrolle der eingetragenen Signale lassen Sie sich die versendeten Signale der Zustände über den Kontextmenüeintrag Empfangene Signale des Objektes *FSM_Control* anzeigen.

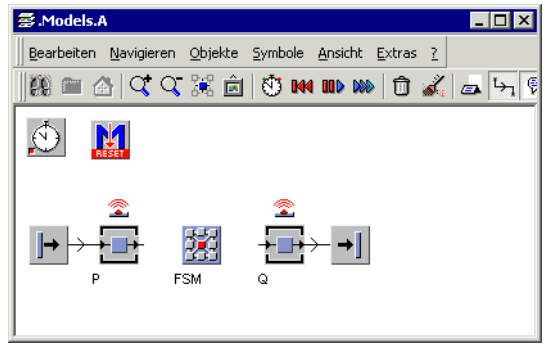


Anzeige der Signale, die Aktionen ausgelösen.

Materialflussobjekte senden Signale an die FSM

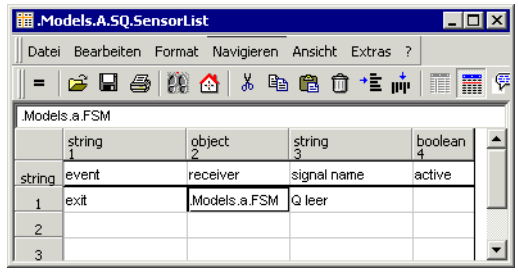
Sensoren senden an *FSM* Signale, die bestimmte Materialflussereignisse beschreiben.

Wenn ein Teil auf *P* fertig bearbeitet ist, so muss die *FSM* aktiv werden. Sie setzen ein Objekt *SensorCtrl* ein und ziehen *P* auf dieses Objekt. Es öffnet sich der Dialog des Objektes. In der Sensorenliste wählen Sie in der ersten Spalte *ready* aus. In die zweite Spalte ziehen Sie die *FSM*. Nach dem Schliessen und Öffnen dieser Tabelle wird hier ein relativer Pfad *~.FSM* eingetragen. Schliesslich tragen Sie den Signalnamen *P belegt* in die dritte Spalte ein.



Sensorsteuerungen an den Objekten *P* und *Q*

Darüberhinaus muss das Ereignis registriert werden, wenn *Q* leer wird. Ordnen Sie *Q* eine Sensorsteuerung zu, die das Signal *Q leer* an die *FSM* schickt, wenn *Q* leer wird.



Sensorenliste für die Sensorsteuerung an der Einzelstation *Q*

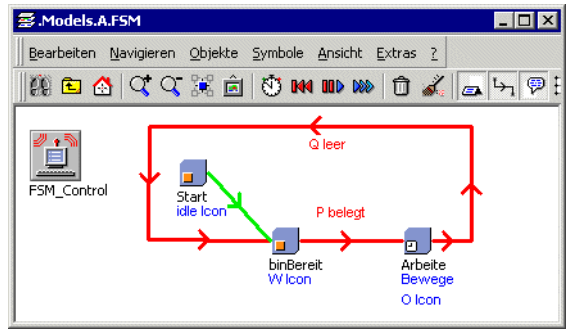
Übergänge zwischen den Zuständen

Erst nachdem Sie die Zustände der *FSM* definiert haben, können Sie über die Bedingungen für Übergänge zwischen den Zuständen nachdenken. Zu Beginn der Simulation befindet sich die *FSM* in dem Zustand *Start*. In unserem Beispiel kann die *FSM* am Anfang sofort in den Zustand *binBereit* wechseln.

Ist die *FSM* im Zustand *binBereit,* so kann sie in den Zustand *Arbeite* wechseln, wenn auf *P* ein Teil liegt und fertig bearbeitet ist. Diese Situation wird durch das Signal *P beleg* angezeigt.

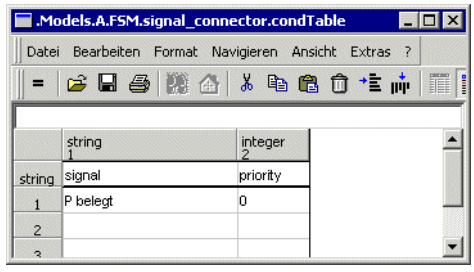
Ist die *FSM* im Zustand *Arbeite,* so kann sie in den Zustand *binBereit* wechseln, wenn *Q* leer ist. Diese Situation wird durch das Signal *Q leer* angezeigt.

Damit die Bedingungen für die Übergänge zwischen den Zuständen geprüft werden, fügen Sie mit dem Objekt *signal_connector* Verbindungen zwischen den Zustandsobjekten ein. Es können nur Übergänge zwischen Zuständen auftreten, die verbunden sind.



Übergänge zwischen Zuständen mit Bedingungen werden rot dargestellt.

Um eine Bedingung für einen Übergang festzulegen, doppelklicken Sie eine Verbindung. Es öffnet sich eine Tabelle, in der Sie das Signal eingeben, das für den Übergang notwendig ist.

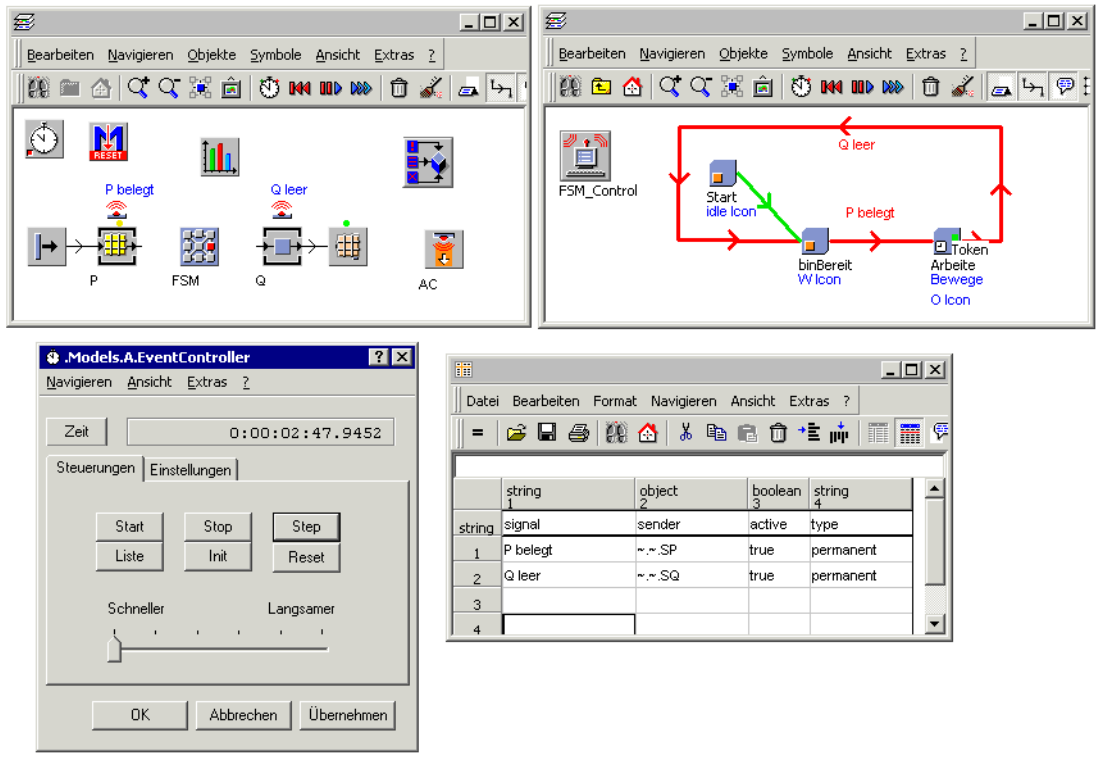


Bedingung für den Übergang vom Zustand *binBereit* zum Zustand *Arbeite*

Validierung der Arbeitsweise der FSM

Beobachten Sie nun die Arbeitsweise der *FSM*. Öffnen Sie die *FSM*, um den aktuellen Zustand zu sehen. Doppelklicken Sie das Objekt *FSM_Control*. Öffnen Sie die Tabelle der aktuellen Signale durch die Schaltfläche **Aktuelle Signale** des Dialoges **Automatensteuerung** und beobachten Sie das Eintreffen von Signalen bei Materialflussereignissen und das Verschwinden von Signalen bei Zustandsänderungen.

Beachten Sie, dass am Ende der eingetragenen Dauer des Zustandes *Arbeite* die Einzelstation *Q* leer oder belegt sein kann.



Überprüfung der richtigen Funktionsweise der *FSM*

In der Darstellung sehen Sie die *FSM* im Zustand *Arbeite*. Die Dauer dieses Zustandes *Arbeite* ist noch nicht abgelaufen, aber *Q* ist schon leer. Die Übergang zum Zustand *binBereit* kann wegen der eingetragenen Dauer des Zustandes *Arbeite* noch nicht erfolgen.

Es ist auch die Situation möglich, dass *Q* nicht leer ist, aber die Dauer des Zustandes *Arbeite* abgelaufen ist. In diesem Falle meldet die *FSM*: 'Der Übergang von *Arbeite* nach *binBereit* ist unmöglich.' Diese Meldung können Sie in der Nachrichtenliste des Protokollobjektes sehen.

About Siemens PLM Software

Siemens PLM Software, a division of Siemens Automation and Drives (A&D), is a leading global provider of product lifecycle management (PLM) software and services with 4.6 million licensed seats and 51,000 customers worldwide. Headquartered in Plano, Texas, Siemens PLM Software's open enterprise solutions enable a world where organizations and their partners collaborate through Global Innovation Networks to deliver world-class products and services. For more information on Siemens PLM Software products and services, visit www.siemens.com/plm.

SIEMENS

Division headquarters

United States
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
972 987 3000
Fax 972 987 3398

Regions

Americas
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
800 498 5351
Fax 972 987 3398

Europe
Norwich House Knoll Road
Camberley, Surrey
GU15 3SY
United Kingdom
44 1276 702000
Fax 44 1276 705150

Asia-Pacific
Suites 6804-8, 68/F., Central Plaza
18 Harbour Road, WanChai
Hong Kong
852 2230 3333
Fax 852 2230 3210

10/07