7. Visuelle Spezifikation agentenbasierter Simulationsmodelle

In diesem Kapitel wird eine grafische Spezifikationssprache für Agentenbasierte Simulationssysteme in Form eines UML-Profils und in Form eines AORML-Profils vorgestellt. Beide Modellierungssprachen erlauben eine vollständige visuelle Spezifikation Agentenbasierter Simulationsmodelle. Abschnitt 7.1 stellt zunächst die Prinzipien der visuellen Spezifikation mit Hilfe von UML vor, während in Abschnitt 7.2 das gleiche unter Verwendung von AORML vorgenommen wird. In beiden Abschnitten werden die grafischen Modellkomponenten durch ein Beispiel aus dem FTS-Bereich und ggf. zusätzlich durch ein abstraktes Beispiel vorgestellt. Abschnitt 7.3 enthält schließlich eine vergleichende Bewertung beider Modellierungsmethoden.

7.1 Spezifikation mittels UML

Dieser Abschnitt stellt eine visuelle Spezifikationssprache für Agentenbasierte Simulationssysteme in Form eines UML-Profils vor.

In Abschnitt 1.6 wurde bereits als Anforderung für die Agentenbasierte Simulation genannt, dass es eine visuelle, UML-basierte Simulationsspezifikationssprache gibt. Der Vorteil bei einer visuellen Spezifikationssprache ist, dass sie leichter erfassbar ist und damit die Simulationsmodelle leichter kommunizierbar sind. UML-Basierung ist sinnvoll, da sich UML als Standard in Forschung und Industrie etabliert und die Konzepte der Softwaremodellierung der letzten Jahrzehnte in sich aufgenommen hat.

Deshalb wird in diesem Kapitel eine UML-basierte Sprache vorgestellt, mit deren Hilfe man ein Agentenbasiertes Simulationssystem vollständig visuell spezifizieren kann. Ein so erstelltes Simulationsmodell ist semantisch äquivalent zu einem Modell mit tabellarischen Darstellungen, wie es in Abschnitt 4.3.4 und in Kapitel 6 verwendet wurde, profitiert aber von den genannten Vorteilen einer visuellen, UML-basierten Sprache. Außerdem lässt sich mit einem entsprechenden Simulationssystem ein solches Modell automatisch ausführen. Ein zu diesem Zwecke konzipiertes Simulationssystem wird in Kapitel 8 vorgestellt.

In der Modellierung wird zwischen dem *externen Modell*, das sind alle für den Umgebungssimulator relevanten Informationen und den *internen Modellen*, das sind alle für die Agentensimulatoren relevanten Informationen, unterschieden.

Sowohl im externen als auch im internen Modell unterscheidet man drei verschiedene Submodelle. Das *Strukturmodell* beinhaltet die verschiedenen Agenten-, Objekt-, Ereignis-, Wahrnehmungs-, Aktions- und Nachrichtentypen mit ihren Zuständen/Attributen. Im *Verhaltensmodell* wird das Verhalten der Agenten und der Umgebung spezifiziert, sprich die Reaktionsregeln. Als dritter Bestandteil werden die *konkreten Instanzen* des Simulationslaufs, also die vorhandenen Objekte und Agenten mit ihren externen bzw. internen Startzuständen sowie die konkreten Stränge exogener bzw. periodischer Ereignisse aufgeführt. Für den Ablauf eines konkreten Simulationslaufes müssen alle drei Submodelle vorhanden sein. Sie unterscheiden sich durch die Häufigkeit ihrer Änderung. Das Strukturmodell wird fast nie verändert, sondern bleibt in der Regel über alle Simulationsläufe hinweg gleich. Das Verhaltensmodell wird ab und zu geändert, aber nicht sehr häufig. Die konkreten Instanzen können sich von Simulationslauf zu Simulationslauf unterscheiden, sie werden häufig geändert.

In Abschnitt 7.1.1 wird das Prinzip der Modellierung des *externen Modells* und in Abschnitt 7.1.2 das Prinzip der Modellierung der *internen Modelle* vorgestellt. Abschnitt 7.1.3 zeigt auf, wie Verbindungen zwischen dem externen und dem internen Modell spezifiziert werden.

7.1.1 Das externe Modell

Das *externe Modell* beinhaltet alle für den Umgebungssimulator relevanten Informationen.

7.1.1.1 Strukturmodell

Die am Simulationssystem beteiligten Agententypen und Objekttypen werden als UML-Klassendiagramm dargestellt, wobei die Unterscheidung in Agenten und Objekte durch die Stereotypes <<Agent>> bzw. <<Object>> kenntlich gemacht wird. Die Attribute der Objekte sowie der externen Zustände der Agenten werden dabei aufgeführt. Die ID eines Agenten wird durch das implizite Attribut[73] *id* gekennzeichnet, ebenso die ID eines Objekts. Ist der Initialwert eines Attributs für alle Instanzen einer Agenten-/ Objektklasse festgelegt, so kann er mit Hilfe des Initialwertes im UML-Klassendiagramm spezifiziert werden.

Abbildung 32: Agenten- und Objekttypen im UML-Klassendiagramm (abstraktes Beispiel)

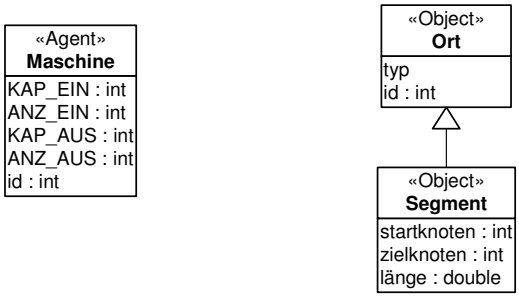

Abbildung 33: Agenten- und Objekttypen im UML-Klassendiagramm (FTS-Beispiel)

Nachrichten werden als Klassen im UML-Klassendiagramm dargestellt, die mit dem Stereotype <<Message>> versehen sind. Der Agententyp des Senders und des Empfängers wird jeweils durch eine Assoziation zur entsprechenden Agentenklasse spezifiziert, wobei die Assoziationsenden mit den Stereotypes <<Sender>> bzw. <<Receiver>> gekennzeichnet sind. Die Übertragungsdauer und die Zuverlässigkeit des Nachrichtentyps werden durch die Initialwerte der impliziten Attribute *duration* und *reliability* ausgedrückt. Parameter des Nachrichtentyps werden als Attribute spezifiziert.

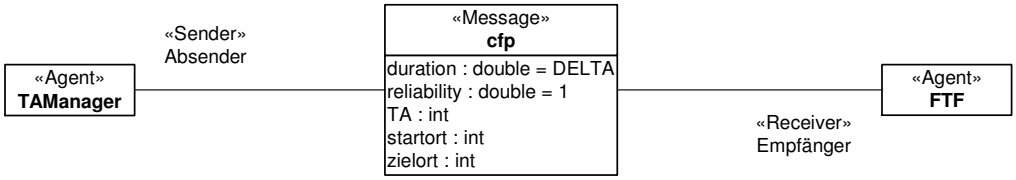

Abbildung 34: Nachrichtentypen im UML-Klassendiagramm (FTS-Beispiel)

Ereignisse in der Umgebung, auf die der Umgebungssimulator reagiert, werden als Klassen im UML-Klassendiagramm modelliert, wobei als Stereotype <<EnvironmentalEvent>> bzw. für exogene Ereignisse <<ExogenousEvent>>[74] verwendet wird. Beide Typen von Ereignissen haben ein implizites Attribut *occurrence_time*, das den Zeitpunkt bestimmt, zu dem das Ereignis stattfindet. Exogene Ereignisse haben ferner ein implizites Attribut *periodicity*, mit dessen Hilfe die Periodizität ausgedrückt wird. Ist die Periodizität für alle Stränge von exogenen Ereignissen eines Typs gleich, so kann sie im UML-Klassendiagramm angegeben werden. Gleiches gilt für die Stopbedingung, die als

[73] Implizite Attribute sind auch dann vorhanden, wenn sie nicht explizit spezifiziert werden. Man kann ihnen wie anderen Attributen in Objektdiagrammen Werte zuweisen und sie in OCL-Ausdrücken verwenden.

[74] Eigentlich wäre der Stereotype <<ExogenousEnvironmentalEvent>> exakter, aber da exogene Ereignisse immer Umgebungsereignisse sind, wurde aus Gründen der Kürze auf den Wortteil *Environmental* verzichtet.

tagged value mit dem Namen *stopcondition* als OCL-Ausdruck spezifiziert wird.[75] Aktionen werden als Klassen mit Stereotype <<ActionEvent>> dargestellt, wobei eine Assoziation mit der Klasse des Agententyps vorliegt, dessen Agenten diese Aktion durchführen. Das zugehörige Ende der Assoziation ist mit dem Stereotype <<Actor>> gekennzeichnet. Parameter der Ereignisse bzw. Aktionen werden als Attribute spezifiziert. Zusicherungen für Ereignisse, exogene Ereignisse und Aktionen werden als *tagged value* mit dem Namen *assertion* als OCL-Ausdruck spezifiziert.

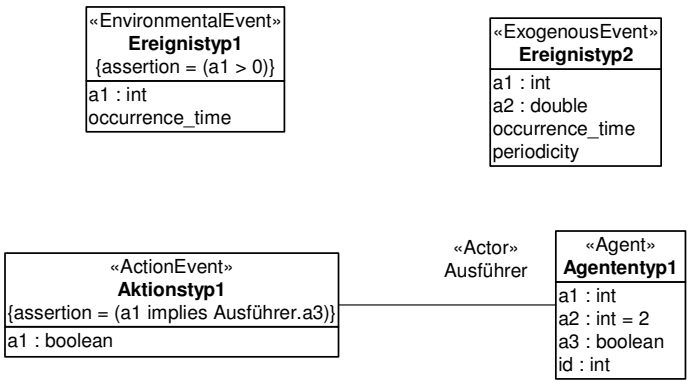

Abbildung 35: Ereignis- und Aktionstypen im UML-Klassendiagramm (abstraktes Beispiel)

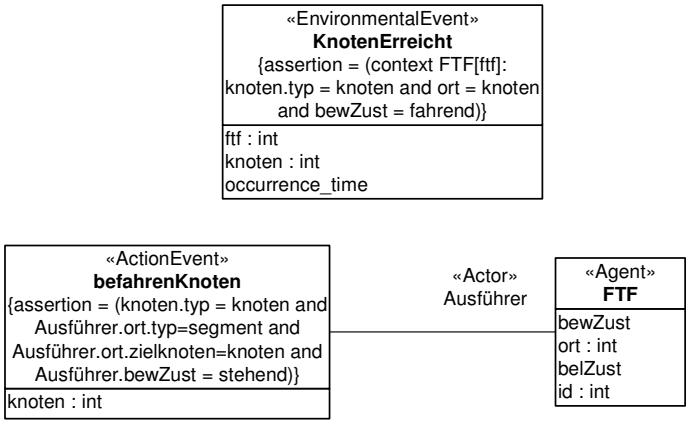

Abbildung 36: Ereignis- und Aktionstypen im UML-Klassendiagramm (FTS-Beispiel)

Wahrnehmungsereignisse werden als Klassen mit Stereotype <<PerceptualEvent>> modelliert, wobei eine Assoziation mit der Klasse des Agententyps vorliegt, dessen Agenten diese Wahrnehmung haben. Das zugehörige Ende der Assoziation ist mit dem Stereotype <<Perceptor>> gekennzeichnet.

Abbildung 37: Wahrnehmungstypen im UML-Klassendiagramm (FTS-Beispiel)

[75] Eigentlich müsste man hier zwei Klassentypen unterscheiden, den Klassentyp der exogenen Ereignisse selbst und die Stränge von exogenen Ereignissen dieses Typs. Das Attribut *occurence_time* macht nur im ersten Klassentyp einen Sinn, die Attribute *periodicity* und *stopcondition* nur im zweiten. Um die Modellierung nicht zu verkomplizieren, wird jedoch keine Trennung dieser Klassentypen vorgenommen.

7.1.1.2 Verhaltensmodell

Reaktionsregeln für den Umgebungssimulator werden als n-äre Assoziation dargestellt. Dabei befindet sich an einem Ende der Assoziation das die Regel anstoßende Ereignis, wobei das Assoziationsende mit dem Stereotype <<Trigger>> versehen ist. In Frage kommen hierbei mit dem Stereotype <<EnvironmentalEvent>>, <<ExogenousEvent>> oder <<ActionEvent>> versehene Klassen. Aktionsereignisse (Stereotype <<ActionEvent>>) haben dabei im Kontext des externen Modells ein implizites Attribut *actorID*, das den Agenten enthält, der die Aktion durchgeführt hat. Für jeden Agenten und jedes Objekt, dessen Zustand von dieser Regel betroffen ist, gibt es ein weiteres Assoziationsende, das den Stereotype <<InvolvedEntity>> enthält, ebenso für jedes Folgeereignis (Stereotype <<ResultingEvent>>) und jede Wahrnehmung (Stereotype <<ResultingPerception>>). Die Ereignisbedingung wird als in OCL formulierte Bedingung dem mit dem Ereignis verbundenen Assoziationsende unter dem Namen *cond* zugeordnet. Die Zustandsbedingung wird auf die betroffenen Agenten und Objekte aufgeteilt, so dass die tatsächliche Zustandsbedingung eine Konjunktion aller für einzelne Agenten und Objekte formulierten Zustandsbedingungen ist.[76] Gleiches gilt für den Zustandseffekt. Dabei wird die Zustandsbedingung mit *pre* und der Zustandseffekt mit *post* gekennzeichnet. Bei den Folgeereignissen und Wahrnehmungen werden die Parameter durch in OCL formulierte Bedingungen unter dem Namen *params* spezifiziert, wobei anders als bei den Zustandseffekten der konkrete Wert der Parameter alleine durch diese Bedingung exakt bestimmt sein muss. Der Stereotype <<ResultingEvent>> bzw. <<ResultingPerception>> impliziert dabei, dass kein bestehendes (Ereignis- oder Wahrnehmungs-)Objekt diese Parameter annehmen soll, sondern ein neues (Ereignis- oder Wahrnehmungs-)Objekt mit diesen Werten erzeugt werden soll. Der Zeitpunkt, zu dem Folgeereignisse stattfinden, wird durch Belegung des impliziten Attributes *occurrence_time* in der in OCL formulierten Bedingung spezifiziert. Im Kontext des externen Modells haben Wahrnehmungen ein implizites Attribut *perceptorID*, mit deren Hilfe spezifiziert wird, welcher Agent die Wahrnehmung erhält.

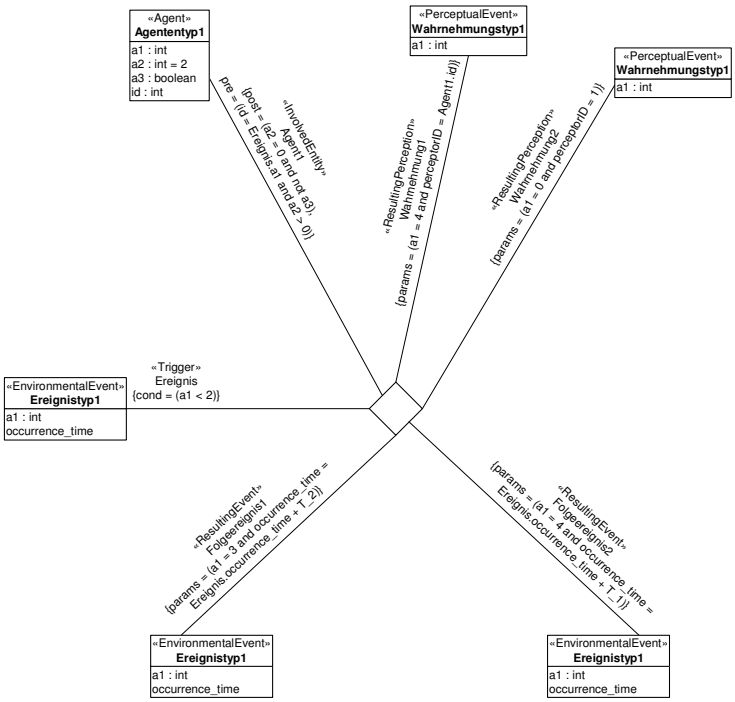

Abbildung 38: Eine Regel für den Umgebungssimulator im UML-Klassendiagramm (abstraktes Beispiel)

[76] Allgemeiner wäre es, statt nur Konjunktionen beliebige logische Verknüpfungen von Bedingungen der einzelnen betroffenen Entitäten zuzulassen. Es lässt sich aber jede logische Formel mit Konjunktionen und Disjunktionen in eine äquivalente Formel transformieren, die die Form einer Disjunktion von Konjunktionen hat (Disjunktive Form). Für jede Konjunktion wird dann die Regel (Ereignis, Ereignisbedingung, Wahrnehmungen, Folgeereignisse) dupliziert. Negationen lassen sich einfach darstellen, in dem man die Zustandsbedingung der betroffenen Entität negiert.

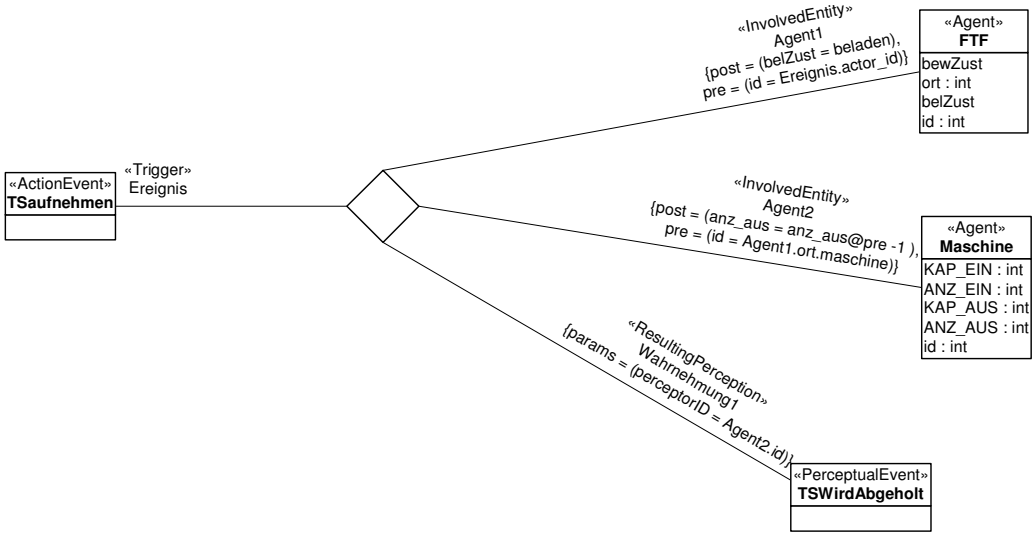

Abbildung 39: Eine Regel für den Umgebungssimulator im UML-Klassendiagramm (FTS-Beispiel)[77]

7.1.1.3 Konkrete Instanzen

Der Umgebungszustand, also die Menge der vorhandenen Agenten und Objekte, wird in einem *Objektdiagramm* dargestellt. Zugleich werden in diesem Diagramm die initialen externen Agentenzustände sowie die initialen Objektzustände spezifiziert. Dabei muss zumindest für die Attribute ein Initialwert angegeben werden, für die kein Initialwert im UML-Klassendiagramm spezifiziert worden ist. Ferner müssen die IDs angegeben werden.

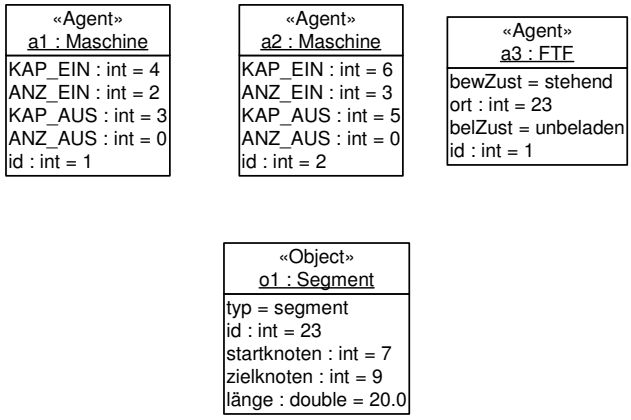

Abbildung 40: Agenten und Objekte im UML-Objektdiagramm (FTS-Beispiel)

Stränge von exogenen Ereignissen werden ebenfalls in einem Objektdiagramm dargestellt. Dabei muss der Wert für die Parameter angegeben werden. Die Periodizität und die Stopbedingung müssen angegeben werden, sofern sie nicht bereits im Klassendiagramm spezifiziert worden sind. Der Wert des Attributs *occurrence_time* gibt den Zeitpunkt des Auftretens des ersten Ereignisses des Strangs an.

[77] Regel Aktion_FTF_aufnehmenTS.1 (Regel 75)

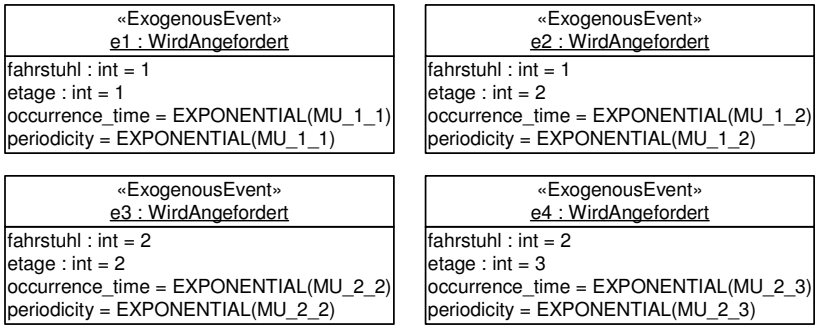

Abbildung 41: Stränge von exogenen Ereignissen im UML-Objektdiagramm (Fahrstuhl-Beispiel)[78]

7.1.2 Das interne Modell

Das *interne Modell* eines Agenten beinhaltet alle für seinen Agentensimulator relevanten Informationen.

7.1.2.1 Strukturmodell

Der interne Zustand eines am Simulationssystem beteiligten Agententypen wird im UML-Klassendiagramm dargestellt, wobei der Stereotype <<IMAgent>> (*InternalModelAgent*) verwendet wird. Die Attribute der internen Zustände der Agenten werden dabei aufgeführt. Die ID eines Agenten wird durch das Attribut *id* gekennzeichnet. Ist der Initialwert eines Attributs festgelegt, so kann er mit Hilfe des Initialwertes im UML-Klassendiagramm spezifiziert werden.

Abbildung 42: Agententypen des internen Modells im UML-Klassendiagramm (FTS-Beispiel)

Interne Zeitereignisse, auf die ein Agentensimulator reagiert, werden als Klassen im UML-Klassendiagramm modelliert, wobei als Stereotype <<TimeEvent>> bzw. für periodische interne Zeitereignisse <<PeriodicalTimeEvent>> verwendet wird. Es liegt jeweils eine Assoziation mit der Klasse des Agententyps vor, zu dem dieser Typ von Zeitereignissen gehört. Beide Typen von internen Zeitereignissen haben ein implizites Attribut *occurrence_time*, das den Zeitpunkt bestimmt, zu dem das interne Zeitereignis stattfindet. Periodische interne Zeitereignisse haben ein implizites Attribut *periodicity*, mit dessen Hilfe die Periodizität ausgedrückt werden kann. Ist die Periodizität für alle Stränge von periodischen internen Zeitereignissen eines Typs gleich, so kann sie im UML-Klassendiagramm angegeben werden. Gleiches gilt für die Stopbedingung, die als *tagged value* mit dem Namen *stopcondition* als OCL-Ausdruck spezifiziert wird. Zusicherungen für interne Zeitereignisse und periodische interne Zeitereignisse werden als *tagged value* mit dem Namen *assertion* als OCL-Ausdruck spezifiziert.

[78] Da im FTS-Beispiel keine exogenen Ereignisse modelliert wurden, wird hier ein Ereignis aus dem Fahrstuhlbeispiel verwendet.

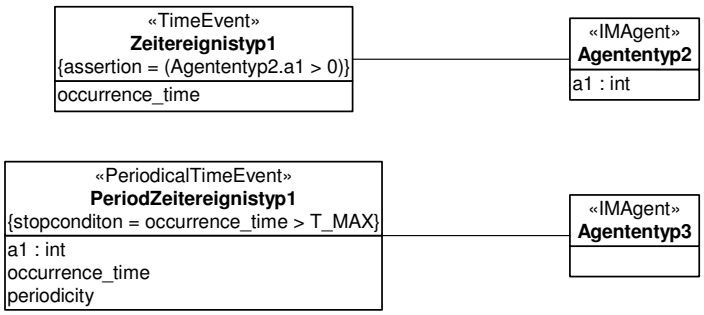

Abbildung 43: interne Zeitereignistypen im UML-Klassendiagramm (abstraktes Beispiel)

Abbildung 44: interne Zeitereignistypen im UML-Klassendiagramm (FTS-Beispiel)

Will man zu einem Wahrnehmungsereignis eine Zusicherung spezifizieren, so erneuert man die Assoziation dieses Wahrnehmungsereignisses aus dem externen Modell, diesmal aber mit dem Agententypen aus dem internen Modell. Die Zusicherung wird als in OCL formulierte Bedingung dem mit dem Wahrnehmungsereignis verbundenen Assoziationsende unter dem Namen *assertion* zugeordnet.

Will man zu einer eingehenden Nachricht eine Zusicherung spezifizieren, so erneuert man die Assoziation dieser Nachricht zur Empfängerklasse aus dem externen Modell, diesmal aber mit dem Agententypen aus dem internen Modell. Die Zusicherung wird als in OCL formulierte Bedingung dem mit der Nachricht verbundenen Assoziationsende unter dem Namen *assertion* zugeordnet. Eingehende Nachrichten haben im Kontext des internen Modells ein implizites Attribut *senderID*, das den Absender der Nachricht enthält.

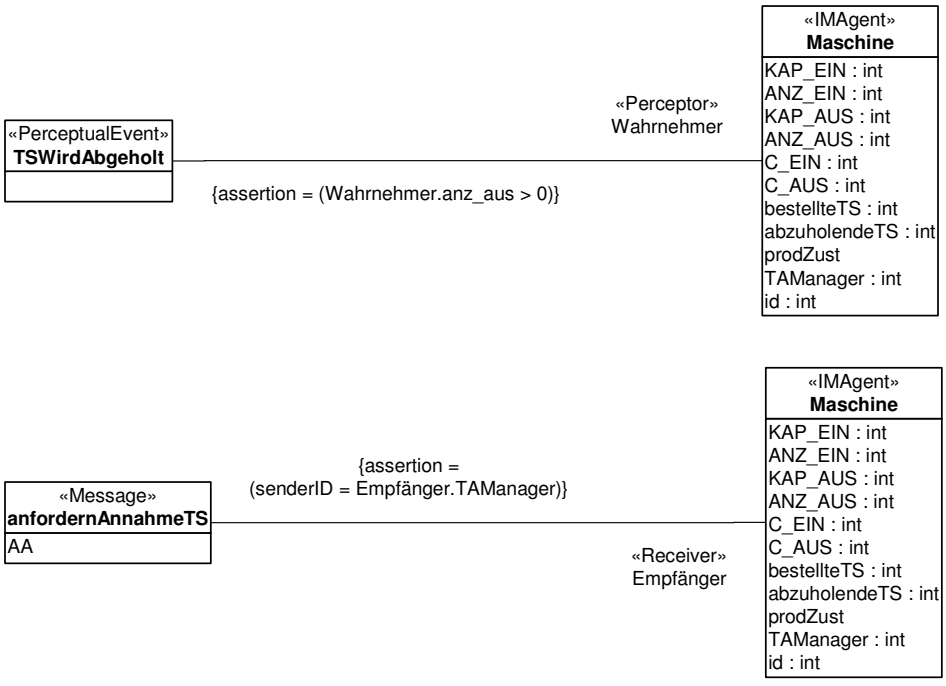

Abbildung 45: Zusicherungen zu Wahrnehmungs- und Nachrichtentypen im UML-Klassendiagramm (FTS-Beispiel)

7.1.2.2 Verhaltensmodell

Reaktionsregeln für einen Agentensimulator werden wie Reaktionsregeln für den Umgebungssimulator als n-äre Assoziation dargestellt. Dabei befindet sich an einem Ende der Assoziation das die Regel anstoßende interne Ereignis, wobei das Assoziationsende mit dem Stereotype <<Trigger>> versehen ist. In Frage kommen hierbei mit dem Stereotype <<PerceptualEvent>>, <<TimeEvent>>, <<PeriodicalTimeEvent>> oder <<Message>> versehene Klassen. Eingehende Nachrichten (Stereotype <<Message>>) haben dabei ein implizites Attribut *senderID*, das den Absender der Nachricht enthält. Für den Agententyp, dem die Regel zugeordnet ist, gibt es ein weiteres Assoziationsende, das den Stereotype <<Reactor>> enthält, ebenso für jedes Folge-Zeitereignis (Stereotype <<ResultingTimeEvent>>), für jede gesendete Nachricht (Stereotype <<OutgoingMessage>>) und jede durchgeführte Aktion (Stereotype <<ResultingAction>>). Die Ereignisbedingung wird als in OCL formulierte Bedingung dem mit dem internen Ereignis verbundenen Assoziationsende unter dem Namen *cond* zugeordnet. Die Zustandsbedingung und der Zustandseffekt werden als in OCL formulierte Bedingung dem mit dem Agenten verbundenen Assoziationsende zugeordnet. Dabei wird die Zustandsbedingung mit *pre* und der Zustandseffekt mit *post* gekennzeichnet. Bei den Folge-Zeitereignissen, Nachrichten und Aktionen werden die Parameter durch in OCL formulierte Bedingungen unter dem Namen *params* spezifiziert, wobei anders als bei den Zustandseffekten der konkrete Wert der Parameter alleine durch diese Bedingung exakt bestimmt sein muss. Der Stereotype <<ResultingTimeEvent>>, <<OutgoingMessage>> bzw. <<ResultingAction>> impliziert dabei, dass kein bestehendes (Ereignis-, Nachrichten- oder Aktions-)Objekt diese Parameter annehmen soll, sondern ein neues (Ereignis-, Nachrichten- oder Aktions-)Objekt mit diesen Werten erzeugt werden soll. Der Zeitpunkt, zu dem Folge-Zeitereignisse stattfinden, wird durch Belegung des impliziten Attributes *occurence_time* in der in OCL formulierten Bedingung spezifiziert. Dabei kann die aktuelle Zeit durch das implizite globale Attribut *clock* aufgerufen werden. Im Kontext des internen Modells haben ausgehende Nachrichten ein implizites Attribut *receiverID*, mit deren Hilfe spezifiziert wird, welcher Agent der Empfänger der Nachricht ist.

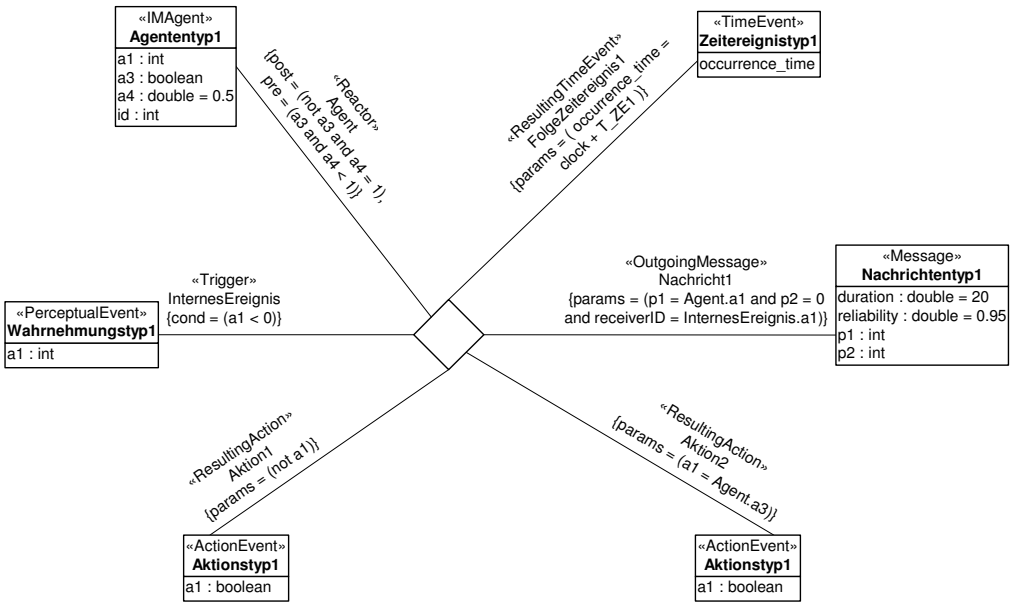

Abbildung 46: Eine Regel für den Agentensimulator im UML-Klassendiagramm (abstraktes Beispiel)

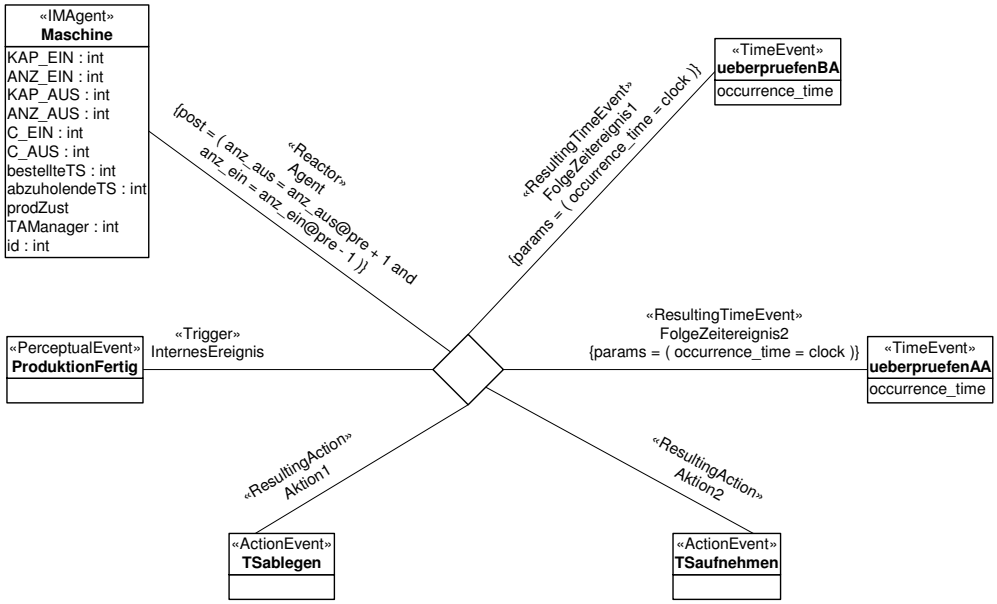

Abbildung 47: Eine Regel für den Agentensimulator im UML-Klassendiagramm (FTS-Beispiel)[79]

7.1.2.3 Konkrete Instanzen

Die initialen Agentenzustände der vorhandenen Agenten werden in einem *Objektdiagramm* dargestellt. Dabei muss zumindest für die Attribute ein Initialwert angegeben werden, für die kein Initialwert im UML-Klassendiagramm spezifiziert worden ist. Ferner müssen die IDs angegeben werden.

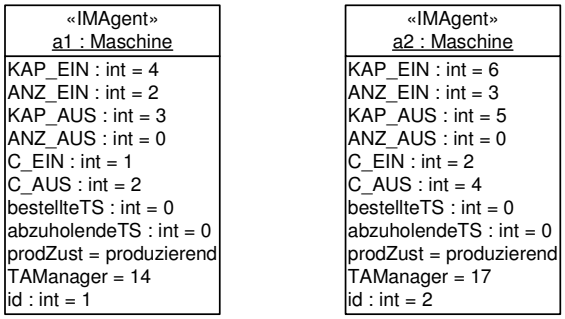

Abbildung 48: Agenten des internen Modells im UML-Objektdiagramm (FTS-Beispiel)

Stränge von periodischen Zeitereignissen werden ebenfalls in einem Objektdiagramm dargestellt. Dabei muss der Wert für die Parameter angegeben werden. Die Periodizität und die Stopbedingung müssen angegeben werden, sofern sie nicht bereits im Klassendiagramm spezifiziert worden sind. Der Wert des Attributs *occurrence_time* gibt den Zeitpunkt des Auftretens des ersten Ereignisses des Strangs an.

[79] Regel Wahrnehmung_ProduktionFertig.3 (Regel 92)

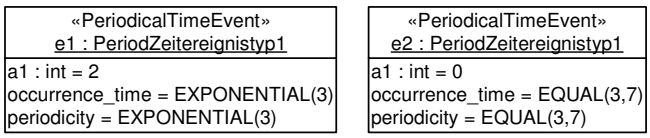

Abbildung 49: Stränge von periodischen Zeitereignissen im UML-Objektdiagramm (abstraktes Beispiel)[80]

7.1.3 Verbindungen zwischen externem und internem Modell

Um zu spezifizieren, welcher interne Agententyp (mit Stereotype <<IMAgent>>) zu welchem (externen) Agententyp (mit Stereotype <<Agent>>) gehört, werden die zusammengehörigen Agententypen gleich benannt.

Zu den Verbindungen zwischen dem externen und dem internen Modell gehören auch die in beiden Modellen verwendeten Nachrichten-, Aktions- und Wahrnehmungstypen. Da diese die Schnittstelle zwischen dem Umgebungssimulator und den Agentensimulatoren darstellen und so in beiden Modellen die gleichen Namen und die gleichen Parameternamen und -typen haben müssen, handelt es sich in beiden Modellen um die selben Klassen. Zu beachten ist lediglich, dass es implizite Attribute gibt, die jeweils nur in einem der beiden Modelle vorhanden sind. So haben Wahrnehmungen nur im Kontext des externen Modells das implizite Attribut *perceptorID*, Aktionen nur im Kontext des externen Modells das implizite Attribut *actorID*. Nachrichten haben nur im Kontext des internen Modells implizite Attribute. Eingehende Nachrichten haben das implizite Attribut *senderID*, ausgehende Nachrichten das implizite Attribut *receiverID*.

7.2 Spezifikation mittels AORML

Nachdem in Abschnitt 7.1 eine visuelle Spezifikationssprache für Agentenbasierte Simulationssysteme in Form eines UML-Profils vorgestellt wurde, zeigt dieser Abschnitt, wie mit Hilfe von AORML [Wag03] ebenfalls ein Agentenbasiertes Simulationssystem vollständig visuell spezifiziert werden kann. Ein mit AORML erstelltes Simulationsmodell ist semantisch äquivalent zu einem Modell mit tabellarischen Darstellungen, wie es in Abschnitt 4.3.4 und Kapitel 6 verwendet wurde und zu einem mit Hilfe von UML spezifizierten Modell wie in Abschnitt 7.1.

Wie bei der Spezifikation mittels UML gibt es auch hier eine Dreiteilung in das Strukturmodell, das Verhaltensmodell und die konkreten Instanzen.

In Abschnitt 7.2.1 wird zunächst das Prinzip der Modellierung des *externen Modells* vorgestellt. In Abschnitt 7.2.2 wird dann das gleiche für die *internen Modelle* durchgeführt. Abschnitt 7.2.3 zeigt auf, wie Verbindungen zwischen dem externen und dem internen Modell spezifiziert werden.

7.2.1 Das externe Modell

Das *externe Modell* wird durch ein *externes AOR-Modell* beschrieben.

7.2.1.1 Strukturmodell

Die am Simulationssystem beteiligten Agententypen und Objekttypen werden in einem *AOR-Agentendiagramm* spezifiziert. In diesem werden Agenten grafisch als Rechtecke mit abgerundeten Ecken, Objekte als Rechtecke dargestellt. Die Attribute der Objekte sowie der externen Zustände der Agenten werden dabei aufgeführt.

[80] Da im FTS-Beispiel und im Fahrstuhlbeispiel keine internen periodischen Zeitereignisse modelliert wurden, wird hier nur ein abstraktes Beispiel verwendet.

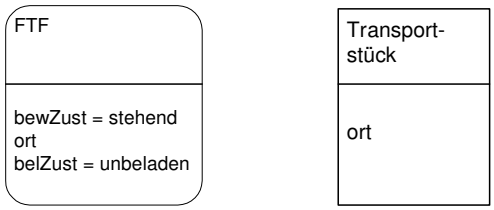

Abbildung 50: Agenten- und Objekttypen im AOR-Agentendiagramm (FTS-Beispiel)

Nachrichten werden als *kommunikative Aktionsereignisse* im *AOR-Agentendiagramm* spezifiziert. Diese werden grafisch durch ein Rechteck mit gepunktet-gestrichelter Umrandung dargestellt. Der Agententyp des Senders und des Empfängers werden jeweils durch gepunktet-gestrichelte Linien mit dem Rechteck verbunden. Die Pfeilspitze des Rechtecks zeigt zum Empfänger hin.

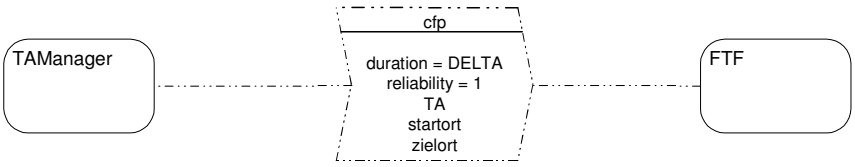

Abbildung 51: Nachrichtentypen im AOR-Agentendiagramm (FTS-Beispiel)

Ereignisse in der Umgebung, auf die der Umgebungssimulator reagiert, werden als *Nicht-Aktionsereignisse* im *AOR-Agentendiagramm* modelliert. Diese werden als Rechteck ohne ausgehenden Pfeil dargestellt. Es wird der Stereotype <<EnvironmentalEvent>> bzw. <<ExogenousEvent>> verwendet, um Umgebungsereignisse und exogene Ereignisse voneinander unterscheiden zu können.

Aktionen werden als *nichtkommunikative Aktionsereignisse* im *AOR-Agentendiagramm* modelliert. Diese haben einen eingehenden Pfeil, der mit dem Agententypen verbunden ist, der die Aktion durchführt.

Zusicherungen für Ereignisse, exogene Ereignisse und Aktionen werden als OCL-Ausdruck innerhalb einer UML-Annotation spezifiziert.

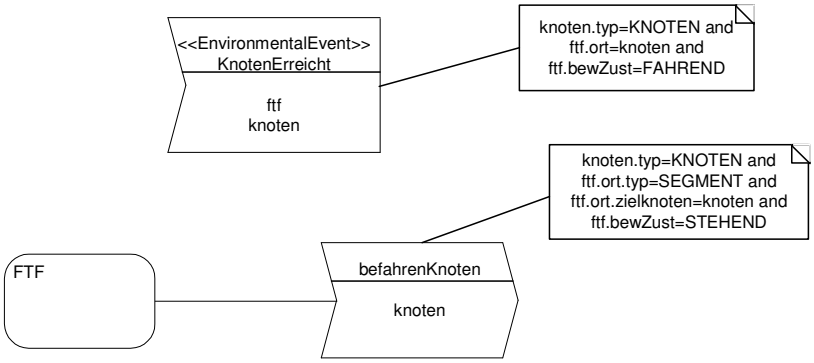

Abbildung 52: Ereignis- und Aktionstypen im AOR-Agentendiagramm (FTS-Beispiel)

Wahrnehmungsereignisse werden als *Nicht-Aktionsereignisse* im *AOR-Agentendiagramm* modelliert. Diese haben einen ausgehenden Pfeil, der mit dem Agententypen verbunden ist, zu dem die Wahrnehmung gehört.

Abbildung 53: Wahrnehmungstypen im AOR-Agentendiagramm (FTS-Beispiel)

7.2.1.2 Verhaltensmodell

Reaktionsregeln für den Umgebungssimulator werden im *AOR-Reaktionsregeldiagramm* spezifiziert. Dabei wird die Reaktionsregel selbst als Kreis mit ein- und ausgehenden Pfeilen dargestellt, wobei das die Regel anstoßende Ereignis der einzige eingehende Pfeil mit gefüllter Pfeilspitze ist. Für jede Entität (Agent oder Objekt), deren Zustand von dieser Regel betroffen ist, gibt es eine weitere ausgehende Verbindung mit gestrichelter Linie und doppelter Pfeilspitze. Für jedes Folgeereignis und jede Wahrnehmung gibt es einen ausgehenden Pfeil. Ereignisbedingungen, Vor- und Nachbedingungen betroffener Entitäten sowie die Parameter von Folgeereignissen und Wahrnehmungen werden durch als UML-Annotationen in OCL formulierte Ausdrücke spezifiziert.

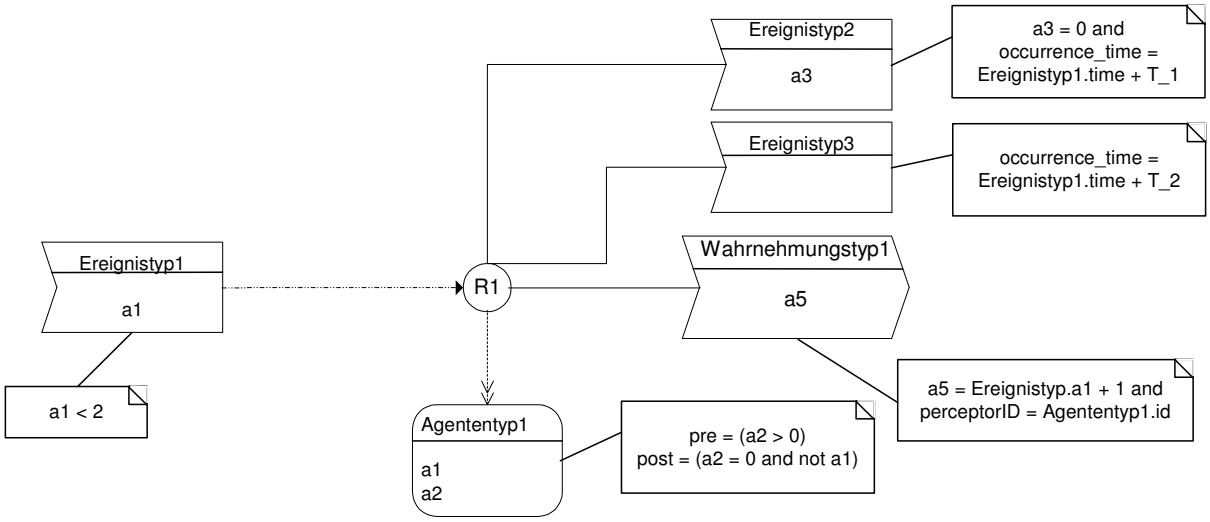

Abbildung 54: Eine Regel für den Umgebungssimulator im AOR-Reaktionsregeldiagramm (abstraktes Beispiel)

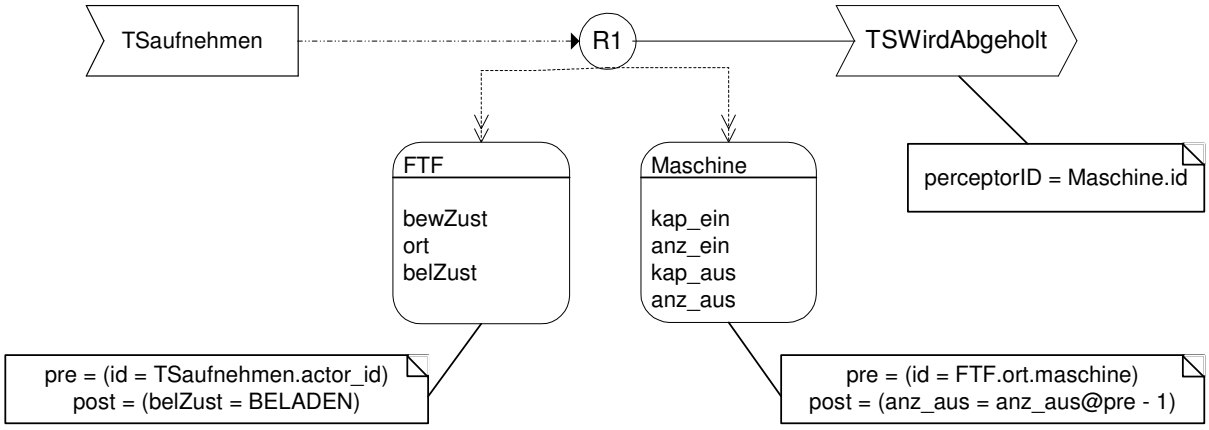

Abbildung 55: Eine Regel für den Umgebungssimulator im AOR-Reaktionsregeldiagramm (FTS-Beispiel)[81]

[81] Regel Aktion_FTF_aufnehmenTS.1 (Regel 75)

7.2.1.3 Konkrete Instanzen

Der Umgebungszustand, also die Menge der vorhandenen Agenten und Objekte werden als Instanzen im *AOR-Agentendiagramm* modelliert. Dabei muss zumindest für die Attribute ein Initialwert angegeben werden, für die kein Initialwert im AOR-Agentendiagramm spezifiziert worden ist. Ferner müssen die IDs angegeben werden.

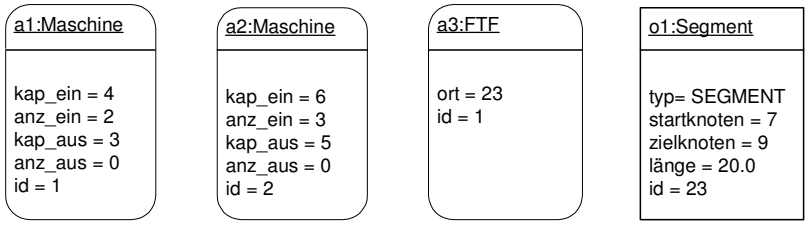

Abbildung 56: Agenten und Objekte im AOR-Agentendiagramm (FTS-Beispiel)

Stränge von exogenen Ereignissen werden als Instanzen von *Nicht-Aktionsereignissen* im *AOR-Agentendiagramm* modelliert.

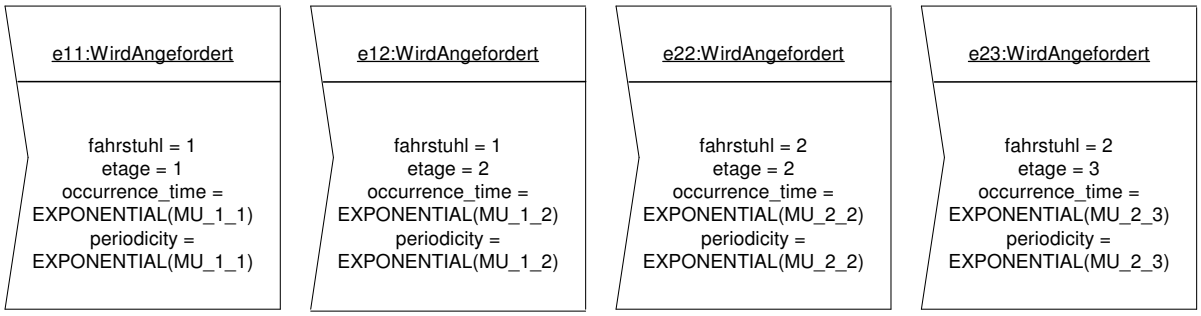

Abbildung 57: Stränge von exogenen Ereignissen im AOR-Agentendiagramm (Fahrstuhl-Beispiel)[82]

7.2.2 Das interne Modell

Das *interne Modell* eines Agenten wird durch ein *internes AOR-Modell* beschrieben. Dabei wird zunächst das externe Modell internalisiert. Dies geschieht, in dem der Focusagent, dessen interne Sicht modelliert wird, aus den Diagrammen weggelassen wird. Stattdessen werden die Diagramme mit einem umschließenden Rahmen mit abgerundeten Ecken versehen, bei denen oben links der Name des modellierten Agenten steht.

7.2.2.1 Strukturmodell

Der interne Zustand eines am Simulationssystem beteiligten Agententypen wird in einem *internen AOR-Diagramm* dargestellt. Die Attribute der internen Zustände der Agenten werden dabei aufgeführt. Ist der Initialwert eines Attributs festgelegt, so kann er mit Hilfe des Initialwertes spezifiziert werden.

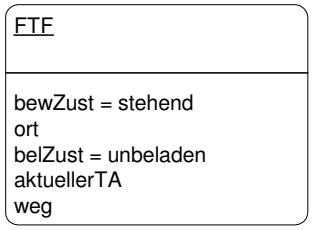

Abbildung 58: Agententypen des internen Modells im internen AOR-Diagramm (FTS-Beispiel)

[82] Da im FTS-Beispiel keine exogenen Ereignisse modelliert wurden, wird hier ein Ereignis aus dem Fahrstuhlbeispiel verwendet.

Interne Zeitereignisse und interne periodische Zeitereignisse, auf die der Agentensimulator reagiert, werden als *Nicht-Aktionsereignisse* im *internen AOR-Diagramm* modelliert. Es wird der Stereotype <<TimeEvent>> bzw. <<PeriodicalTimeEvent>> verwendet, um interne Folge-Zeitereignisse und interne periodische Zeitereignisse voneinander unterscheiden zu können.

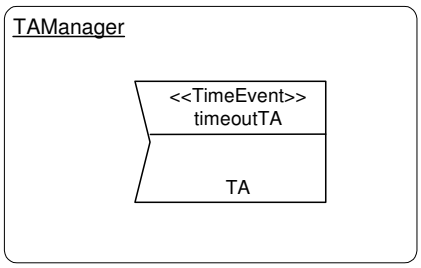

Abbildung 59: interne Zeitereignistypen im internen AOR-Diagramm (FTS-Beispiel)

Wahrnehmungsereignisse eines Agententypen aus dem externen Modell werden als *nichtkommunikative Aktionsereignisse* im internen Modell wiederholt. Will man zu einem Wahrnehmungsereignis eine Zusicherung spezifizieren, so wird diese als in OCL formulierte Bedingung innerhalb einer UML-Annotation spezifiziert.

Eingehende Nachrichten eines Agententyps aus dem externen Modell werden als *kommunikative Aktionsereignisse* im internen Modell wiederholt. Will man zu einer eingehenden Nachricht eine Zusicherung spezifizieren, so wird diese als in OCL formulierte Bedingung innerhalb einer UML-Annotation spezifiziert.

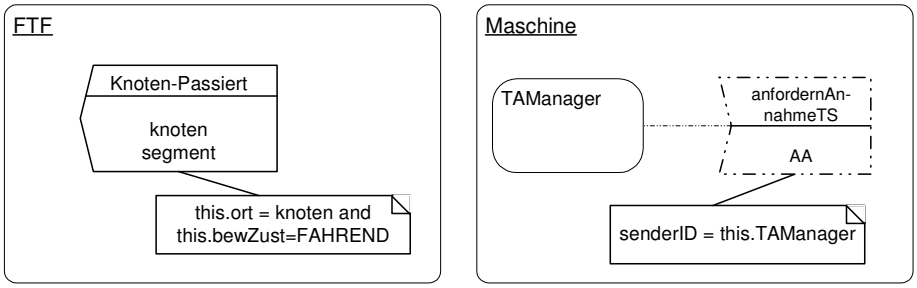

Abbildung 60: Zusicherungen zu Wahrnehmungs- und Nachrichtentypen im internen AOR-Diagramm (FTS-Beispiel)

7.2.2.2 Verhaltensmodell

Reaktionsregeln für einen Agentensimulator werden im *internen AOR-Reaktionsmusterdiagramm* spezifiziert. Dabei wird die Reaktionsregel selbst als Kreis mit ein- und ausgehenden Pfeilen dargestellt, wobei das die Regel anstoßende Ereignis der einzige eingehende Pfeil mit gefüllter Pfeilspitze ist.

Für jedes interne Folge-Zeitereignis, jede gesendete Nachricht und jede durchgeführte Aktion gibt es einen ausgehenden Pfeil. Ereignisbedingungen, sowie die Parameter von internen Folge-Zeitereignissen, ausgeführten Aktionen und versendeten Nachrichten werden durch als UML-Annotationen in OCL formulierte Ausdrücke spezifiziert. Vor- und Nachbedingungen des internen Zustands des Agenten werden ebenfalls durch als UML-Annotationen in OCL formulierte Ausdrücke spezifiziert, wobei die Annotationen am die Reaktionsregel repräsentierenden Kreis vorgenommen werden.

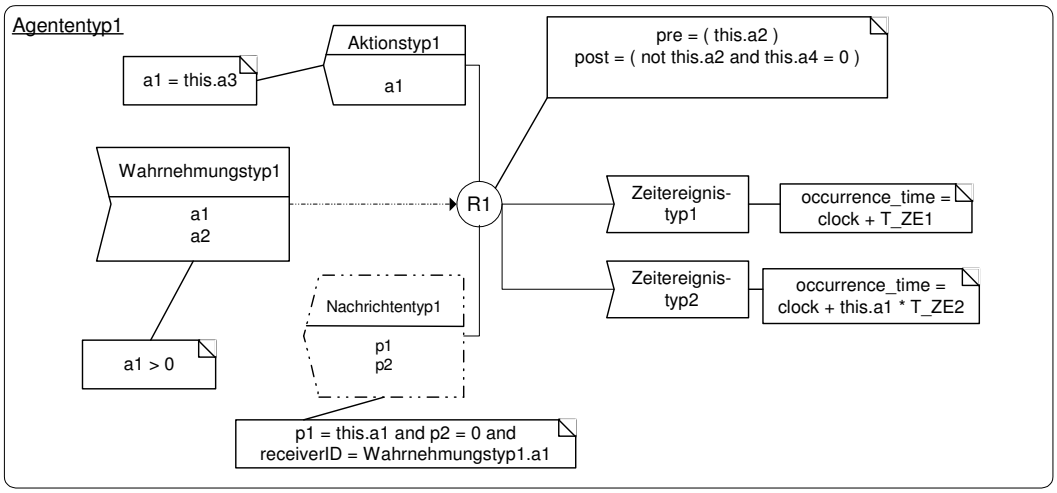

Abbildung 61: Eine Regel für den Agentensimulator im internen AOR-Reaktionsmusterdiagramm (abstraktes Beispiel)

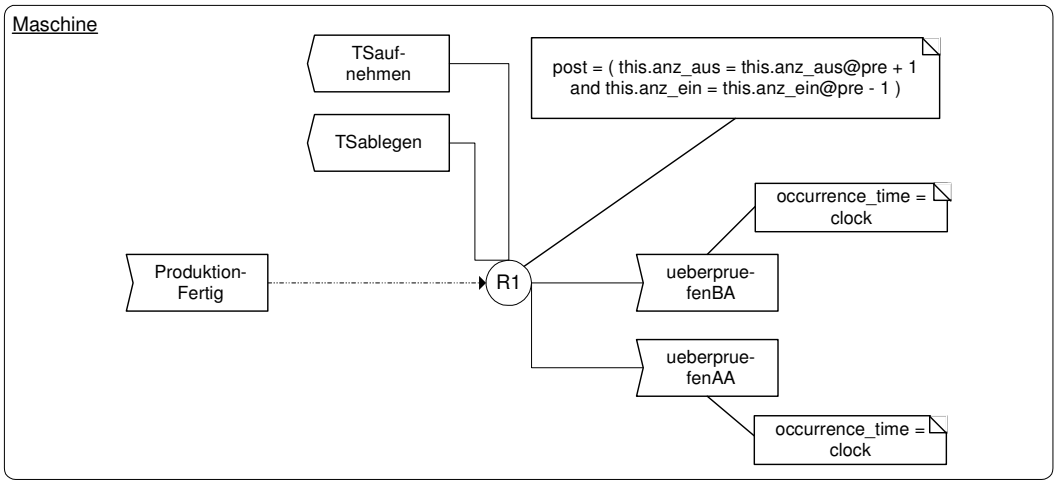

Abbildung 62: Eine Regel für den Agentensimulator im internen AOR-Reaktionsmusterdiagramm (FTS-Beispiel)[83]

7.2.2.3 Konkrete Instanzen

Die initialen Agentenzustände der vorhandenen Agenten werden durch Instanzen im *internen AOR-Diagramm* modelliert. Dabei muss zumindest für die Attribute ein Initialwert angegeben werden, für die nicht bereits ein Initialwert im internen AOR-Diagramm spezifiziert worden ist. Ferner müssen die IDs angegeben werden.

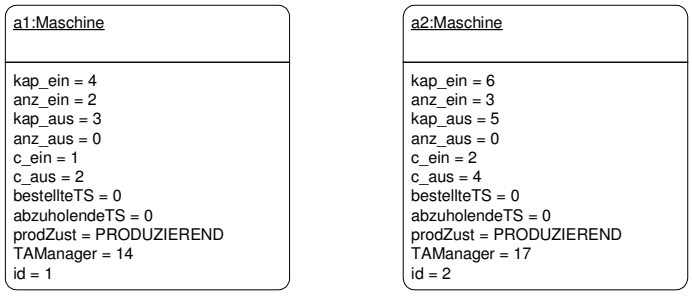

Abbildung 63: Agenten des internen Modells im internen AOR-Diagramm (FTS-Beispiel)

[83] Regel Wahrnehmung_ProduktionFertig.3 (Regel 92)

Stränge von periodischen Zeitereignissen werden als Instanzen von *Nicht-Aktionsereignissen* im *internen AOR-Diagramm* modelliert.

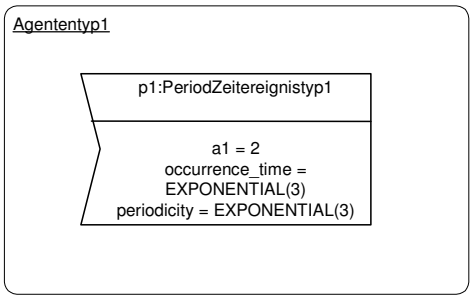

Abbildung 64: Stränge von periodischen Zeitereignissen im internen AOR-Diagramm (abstraktes Beispiel)[84]

7.2.3 Verbindungen zwischen externem und internem Modell

Zu den Verbindungen zwischen dem externen und dem internen Modell gehören die in beiden Modellen verwendeten Agenten-, Nachrichten-, Aktions- und Wahrnehmungstypen. Durch die bei AOR-Modellierung vorzunehmende Internalisierung haben sie in beiden Modellen die gleichen Namen und die Nachrichten-, Aktions- und Wahrnehmungstypen auch die gleichen Parameternamen und -typen.

7.3 Vergleich zwischen UML und AORML

In diesem Abschnitt werden die beiden vorgestellten Modellierungsmethoden UML und AORML vergleichend bewertet.

UML hat den Vorteil, dass es sich UML als Standard in Forschung und Industrie etabliert und die Konzepte der Softwaremodellierung der letzten Jahrzehnte in sich aufgenommen hat. Es gibt zahlreiche Tools für UML, die unter anderem die Möglichkeit bieten, eine XML-basierte Repräsentation des UML-Modells zu erzeugen. Dadurch ist es möglich, mit einem entsprechenden Simulationssystem ein solches Modell automatisch auszuführen. Ein zu diesem Zwecke konzipiertes Simulationssystem wird in Kapitel 8 vorgestellt. Nachteilig an UML ist, dass es als objektorientierte Sprache nicht grafisch zwischen Agenten und Objekten unterscheidet, man nicht zwischen externen und internen Modellen unterscheiden kann und Reaktionsregeln nur über Umwege (als n-äre Assoziationen) spezifiziert werden können.

AORML hat hingegen den Vorteil. dass es zwischen Agenten und Objekten unterscheidet, es eine explizite Unterscheidung zwischen externen und internen Modellen gibt und Reaktionsregeln eine eigene grafische Darstellung haben. Auf die ontologischen Prinzipien von AOR-Modellierung lassen sich die der Agentenbasierten Simulation fast 1:1 abbilden. Nachteil ist jedoch, dass AORML nicht sehr weit verbreitet ist und es demzufolge bis auf ein Template für *Microsoft Visio* noch keine Toolunterstützung gibt.

Aufgrund der vorhandenen Toolunterstützung wurde UML als grafische Modellierungssprache in der Implementierung verwendet, die in Kapitel 8 vorgestellt wird.

[84] Da im FTS-Beispiel und im Fahrstuhlbeispiel keine internen periodischen Zeitereignisse modelliert wurden, wird hier nur ein abstraktes Beispiel verwendet.