der Bundeswehr
Universität München

Bachelorarbeit

Erweiterung und Implementierung eines Modells zur Analyse von Fragestellungen zur Koordination verteilter Organisationsstrukturen

Eingereicht von:	Christian Gerstner
Matrikelnummer:	1070111
Aufgabensteller:	Prof. Dr. Axel Lehmann
Betreuer (UniBwM):	Robert Siegfried
Betreuer (IABGmbh):	Dr. Nane Kratzke
Abgabedatum:	31. Dezember 2009

Universität der Bundeswehr München
Fakultät für Informatik
Institut für Technische Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

A	Aufklärer
CM	Konzeptuelles Modell (Conceptual Model)
CSV-File	Comma-Separated Values-File
FM	Formales Modell (Formal Model)
GRAMS	General Referencemodel for agent-based Modelling and Simulation
JFS	Joint Fire Support
LE	Leistungserbringer
MPA	Maritime Patrol Aircraft
NATO	North Atlantic Treaty Organization
RAS	Replenishment at Sea
STF	Streitkräftegemeinsame Feuerunterstützung
TFU	Taktische Feuerunterstützung
UML	Unified Modelling Language
Z	Ziel
ZE	Zeiteinheit

1 Einleitung und Problemstellung

1.1 Einleitung

Die Bundeswehr und auch die NATO allgemein wird in der heutigen Zeit immer öfter in internationalen Krisengebieten eingesetzt. In diesen existiert ein grobes Einsatzgebiet in dem Gegner überraschend und ohne Vorwarnung auftauchen und eigene Truppen bedrohen können. Es gibt dabei keine klaren Frontlinien und keine Unterscheidung in Gefechtsfeld und rückwärtigen Raum.

Ein typisches Szenario wird in Abbildung 1.1 demonstriert. In diesem Beispiel ist der Einsatzraum in fünf verschiedene Verantwortungsbereiche unterteilt, welche verschiedenen Koordinatoren zugewiesen sind. Im gesamten Einsatzgebiet patrouillieren Aufklärer (A) auf fest definierten Pfaden. Sollte nun ein Aufklärer oder eine andere Einheit das Ziel (Z) entdecken, so wird dieses dem zuständigen Koordinator des Verantwortungsbereiches gemeldet. Dieser wird dann einen der verfügbaren Leistungserbringer (LE), wie zum Beispiel eine Gruppe Infanterie oder Kampfflugzeuge, auswählen, um so das Ziel schnellstmöglich zu bekämpfen.

Wichtig dabei ist es, solche Ziele schnell zu identifizieren und punktgenau gegen sie zu wirken. Um dies zu gewährleisten stehen dem Befehlshaber zunächst seine eigenen unterstellten Einheiten zur Verfügung. Diese werden im Rahmen der streitkräftegemeinsamen taktischen Feuerunterstützung durch alle insgesamt zeitlich und räumlich verfügbaren, auch multinationalen Streitkräfte ergänzt.

Es ist deshalb notwendig, dass nicht nur geeignete Kräfte und Mittel zur Verfügung gestellt werden, sondern auch Strukturen und Verfahren um diese effizient zu nutzen.

1.2 Vorausgegangene Arbeiten

An der Fakultät für Informatik der Universität der Bundeswehr München wurde 2009 ein Referenzmodell zur agentenbasierenden Modellierung und Simulation (GRAMS) [7] geschaffen. Anhand von diesem Referenzmodell ist es nun möglich, beliebige agentenbasierende Modelle zu erstellen, die mithilfe des GRAMS-Simulationsrahmenwerkes ausgeführt werden können.

Bisher fehlen jedoch komplexere Beispielimplementierungen, daher soll mit dieser Bachelorarbeit ein Modell erstellt werden, welches eine solch hohe Komplexität aufweist. Als zu simulierendes Szenario wurde hier das der taktischen Feuerunterstützung gewählt.

Im Sommer dieses Jahres absolvierten Studenten der Wirtschaftsinformatik der Universität der Bundeswehr München ein Industriepraktikum in der IABGmbh in Ottobrunn. Die Zielstellung des Praktikums war die gleiche wie die dieser Bachelorarbeit. Es wurden

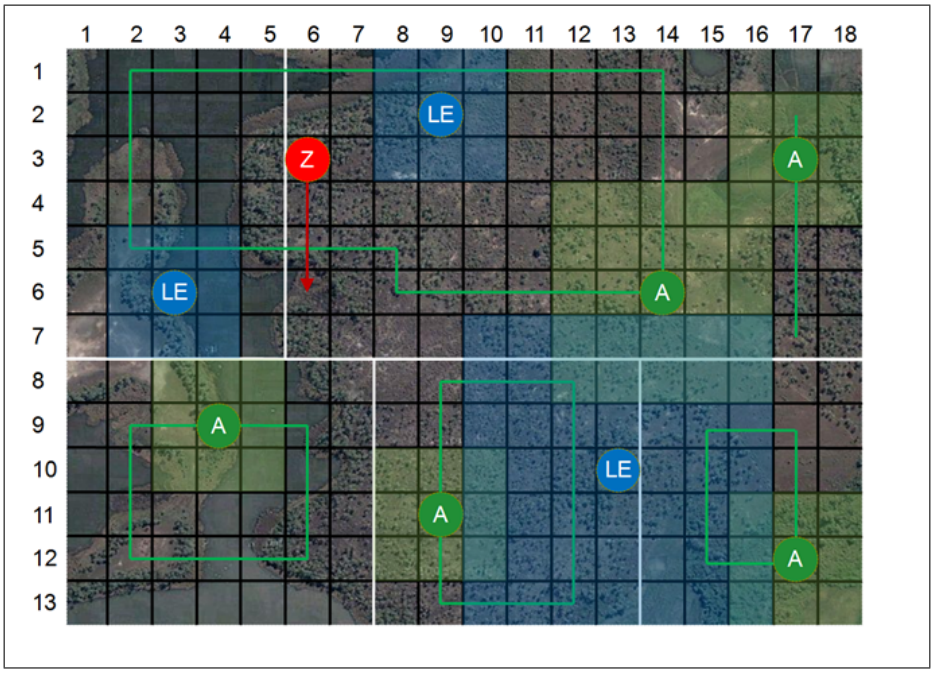

Abbildung 1.1: Beispielszenario - Streitkräftegemeinsame Feuerunterstützung

dabei drei Szenarien bearbeitet: Feuerwehreinsatz bei einem Massenunfall, Mass Behavior und die taktische Feuerunterstützung. Bei allen drei Szenarien wurde zunächst eine genaue Anwenderanforderung (Sponsor Needs) geschrieben. Darauf folgte eine strukturelle Problembeschreibung (Structured Problem Definition) und das konzeptuelle Modell. Es fand noch keine Implementierung statt. Das erstellte Phasendokument zum Szenario taktische Feuerunterstützung, das konzeptuelle Modell, ist Ausgangsbasis dieser Bachelorarbeit und befindet sich im Anhang.

1.3 Vorgehen bei der Erstellung des Modells

Zunächst wurde das formale Modell erstellt. Diesem lag das konzeptuelle Modell aus vorangegangener Arbeit [5] zu Grunde. Genau wie bei dessen Modellierung wurde auch beim formalen Modell wieder der Leitfaden zur Modelldokumentation [1] angewandt. Bei der Erstellung dieses formalen Modells wurde jedoch auch das Referenzmodell zur agentenbasierenden Modellierung und Simulation [7] genutzt. Es erfolgte hier also die Transformation eines Modells, welches auf Submodellen basiert, hin zu einem Modell, das auf selbstständig agierenden Agenten basiert.

Anschließend erfolgte die Implementierung und dazu die Erstellung des ausführbaren Modells (Executable Model). Bei der Implementierung wurden, soweit es sich anbot, Items des Buches „Effective Java“ genutzt [2]. Mit dem fertig implementierten Modell erfolgten nun mehrere Simulationsläufe zu ausgewählten Szenarien (Simulation Results) und anschließend deren Auswertung (Interpretation Results).

2 Formales Modell

In diesem Kapitel wird auf Grundlage der Ausgangsbasis „Konzeptuelles Modell“[5] das formale Modell zum Szenario *Taktische Feuerunterstützung* erstellt. Im Gegensatz zum konzeptuellen Modell ist das formale Modell „eine vom Rechner verarbeitungsfähige Form des konzeptuellen Modells, die eine plattformunabhängige, syntaktisch und (bedingt) semantisch eindeutige Spezifikation darstellt“[1]. Dieses Phasenmodell ist also die letzte Stufe vor der Implementierung. Es ist durchaus möglich, dass sich Unterschiede zum konzeptuellen Modell ergeben, da die gleiche Problemlösung formal anders beschrieben wird. Ebenfalls ist dieses Dokument nicht gleichzusetzen mit dem ausführbaren Modell, also der Implementierung selbst. Die konkrete Implementierung wird daher erst im nächsten Kapitel beschrieben.

2.1 Identifikation des Formalismus

Der Arbeit liegt das GRAMS-Referenzmodell zugrunde. Bei diesem Modell der taktischen Feuerunterstützung handelt es sich um eine Realisierung des allgemeinen Referenzmodells, dessen Wirksamkeit durch dieses Szenario überprüft werden soll. Der verwendete Formalismus für das komplette FM sind UML-Diagramme.

2.2 Struktur und Verhaltensbeschreibung

2.2.1 Identifikation des entsprechenden CM des Modells

Im konzeptuellen Modell wurde das gesamte Modell in Submodelle gegliedert. Mit der Anpassung an das GRAMS-Referenzmodell werden diese Submodelle nun in verschiedene andere Pakete transformiert.

Kapitel des konzeptuellen Modells	**Realisation im formalen Modell**
Gesamtmodellebene Einsatzraum (2.1)	Gesamtebene Einsatzraum 2.2.2
Submodell Kartengeometrie (2.2)	Bestandteil der Umwelt 2.2.2
Submodell Aufklärer (2.3)	Agent Aufklärer 2.2.3
Submodell Koordinator (2.4)	Agent Koordinator 2.2.4
Submodell Ziel (2.5)	Objekt Ziel 2.2.6
Submodell Leistungserbringer (2.6.)	Agent Leistungserbringer 2.2.5
Ausgetauschte Informationen (2.7)	Bestandtteil des Kapitels der Ereignisse 2.2.2

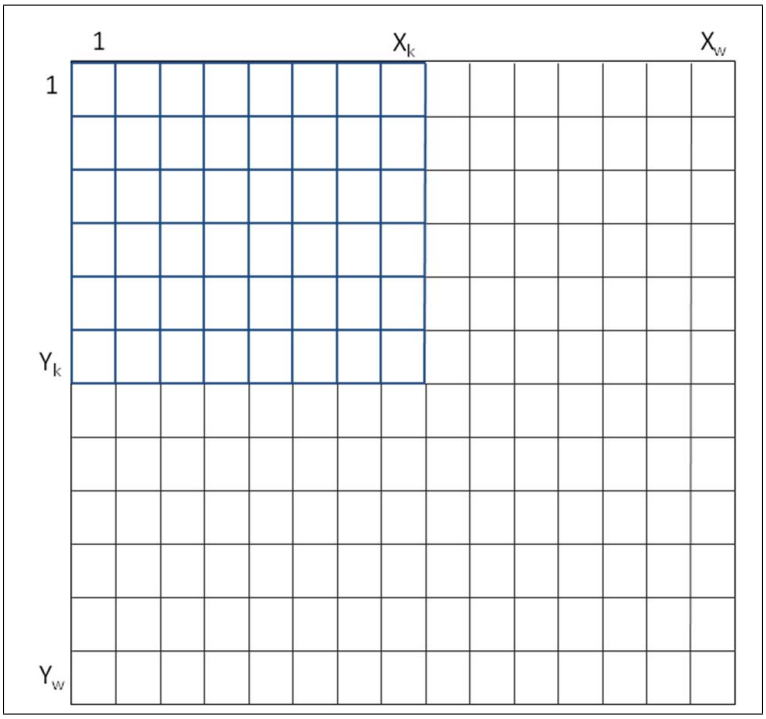

Abbildung 2.1: Die detaillierte Karte (K) ist der innere Bereich der Karte, auf dem sich während der Simulation die Agenten hauptsächlich bewegen werden. Die Simulationswelt (W) beeinhaltet außerdem eine erweiterte Karte, auf der sich Agenten aufhalten können, während sie sich zu einem Ziel bewegen.

2.2.2 Makro-Level: Die Gesamtebene Einsatzraum

Beschreibung von Struktur und internem Verhalten

In diesem Abschnitt wird die Gesamtmodellebene beschrieben. Dazu gehört die Zeit, die Umwelt, die einzelnen Einheiten, die auftretenden Ereignisse und die Rahmenbedingungen, die die gesamte Simulation über eingehalten werden müssen. Diese werden in den folgenden Kapiteln beschrieben.

Zeit

In diesem Modell wird die Zeit durch die Recheneinheit Zeiteinheit(ZE) abgebildet. Sie ist die kleinste, nicht weiter zerlegbare Einheit. Sie hat keinen Bezug zu realen Größen, sondern dient lediglich zu internen Berechnungen und der Festlegung des Simulationsendes. Die Simulationszeit beginnt dabei immer bei ZE 0 und wird dann inkrementell um 1 erhöht, bis t_{ENDE} (Simulationsende) erreicht ist. (Wertebereich: $t_{\text{ENDE}} \in \mathbb{N}$)

Umwelt

Die Umwelt modelliert die physische Grundlage des Systems und hat eine statische Struktur, die während der Simulation nicht geändert werden kann. Die Eigenschaften der Umwelt beeinflussen die handelnden Agenten, zum Beispiel ihre Bewegungsgeschwindigkeit. Die einzelnen Landschaftsmuster sind dabei zufällig verteilt, werden jedoch in einer Konfigurationsdatei gespeichert, um es dem Benutzer zu ermöglichen, den Typ einer einzelnen konkreten Koordinate zu ändern. Weiterhin werden Seegebiete immer zusammenhängend verteilt.

Der Koordinatenursprung der gesamten zweidimensionalen Karte liegt in der linken oberen Ecke, sodass die Koordinate links durch (1/1) bezeichnet wird. Die Karte wird durch einen Rand abgeschlossen. Der östliche Rand der detaillierten Karte ist $\mathrm{X_k}$, der südliche $\mathrm{Y_k}$. Es ist jedoch möglich, dass Leistungserbringer einen Startpunkt außerhalb der Karte haben, um längere Wege zum Wirkungsort zu simulieren. Diese abstrakte Karte hat den östlichen Rand $\mathrm{X_w}$ und den südlichen $\mathrm{Y_w}$, wobei $X_\mathrm{w} > X_\mathrm{k}$ und $Y_\mathrm{w} > Y_\mathrm{k}$. Leistungserbringer können diesen durch $\mathrm{X_k}$ und $\mathrm{Y_k}$ beschriebenen Rand nur überschreiten, wenn ihr Startpunkt außerhalb dieses Randes liegt und sie sich auf dem Weg zum Einsatzort oder auf der Rückkehr zum Startpunkt befinden. Für alle Einheiten, die sich außerhalb der Karte ($X_\mathrm{k} \times Y_\mathrm{k}$) bewegen, gelten die Bewegungseinschränkungen des Landschaftsmusters Neutral. Dieses Landschaftsmuster bildet keinen realen Landschaftstyp ab, sondern dient lediglich der Berechnung der Geschwindigkeit außerhalb der detaillierten Karte.

Aufgrund der verschiedenen Landschaftsmuster ergeben sich für jede Plattform individuelle Einschränkungen.

Abbildung 2.2 zeigt eine Beispielbelegung der Bewegungseinschränkungen. Diese Werte zeigen die Relation der Bewegungsgeschwindigkeiten untereinander und geben nicht die wahre Geschwindigkeit wieder. Zur Berechnung dient folgende Formel:

$$\text{Aktuelle Geschwindigkeit}[Q/ZE] = \frac{\text{Basisgeschwindigkeit}_{[Q/ZE]}}{\text{Bewegungseinschränkungen}}$$

Die Bewegungseinschränkungen werden vom Benutzer vor Simulationsstart durch eine Konfigurationstabelle vorgegeben. Sie dienen der Berechnung der Geschwindigkeit von einzelnen Objekten. Bei entstandenen Kommazahlen wird immer auf die nächste ganze Zahl abgerundet. Ein Objekt kann sich also in einer Zeiteinheit nur über eine natürliche Anzahl Koordinaten bewegen. Objekte können sich nur auf direkt oben, unten, links und rechts angrenzende Koordinaten bewegen. Eine diagonale Bewegung ist also nicht möglich. Die gesamte Karte kann durch einen Kartengenerator aus stochastischen Eingabedaten erzeugt werden. Die einzelnen Koordinaten können natürlich auch noch manuell bearbeitet werden. Die so erstellte Karte wird zum Simulationsstart geladen und stellt für dieses Modell eine deterministische Eingangsgröße dar, die durch den Simulator nicht weiter geändert wird.

Ebenfalls Bestandteil der Umwelt ist ein *Broadcast-Channel*. Dieser dient der Kommunikation aller Agenten untereinander, die keinen vorgesetzten Koordinator besitzen. Dies wird bei den jeweiligen Agenten näher erläutert.

	Luftstreitkräfte	Seestreitkräfte	Landfahrzeuge	Infanterie
Stadt	1	X	50	80
Ebene	1	X	40	80
Wald	1	X	90	75
Gebirge	2	X	X	90
Binnen-gewässer	1	20	X	90
Seegebiet	1	20	X	X
Neutral	1	20	40	80

Abbildung 2.2: Jeder Agent unterliegt abhängig von der Plattform, der er angehört, verschiedenen Bewegungseinschränkungen.

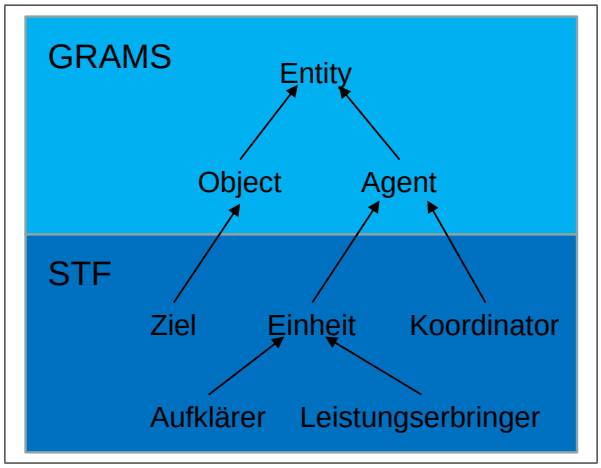

Abbildung 2.3: Alle fünf Objekte übernehmen gemäß GRAMS-Referenzmodell Interfaces oder abstrakte Klassen.

Objekte/Überblick

Alle in der Simulation auftretenden Objekte lassen wie folgt unterteilen:

i. Einheit

ii. Aufklärer

iii. Leistungserbringer

iv. Koordinator

v. Ziel

Die ersten vier Objekte bilden wie in Abb. 2.3 Agenten ab, während das letzte, das Ziel, ein normales Objekt ist. Dabei ist Einheit lediglich als abstrakter Agent zu verstehen, der nicht instanziiert werden kann, sondern im Sinne der Vererbung gemeinsame Attribute an Aufklärer und Leistungserbringer weitergibt. Agenten und Objekte, die im Modell beliebig viele Instanzen haben können, sind folgende vier:

- Aufklärer (Agent) - überwachen ein bestimmtes Gebiet anhand von Wegpunkten und geben Anforderungen auf, wenn sie Ziele identifiziert haben. Jeder Aufklärer kann Anforderer und Wirkungsanalyst sein, aber nicht jeder kann Markierer sein.
- Koordinatoren (Agent) - haben in dem Modell die Aufgabe, die unterstellten Aufklärer und Leistungserbringer zu koordinieren, damit die Anforderung ressourcengerecht ausgeführt werden können. Koordinatoren können auch hierarchisch organisiert sein.

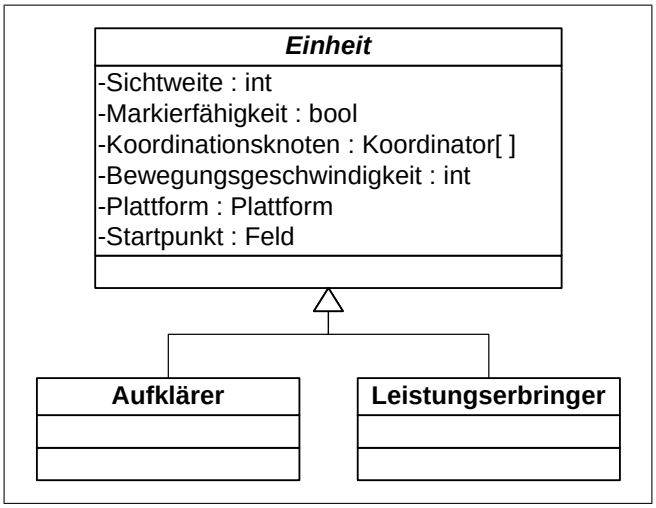

Abbildung 2.4: Das Klassendiagramm des abstrakten Agenten Einheit.

- Leistungserbringer (Agent) - werden in dem Modell entweder keinem Koordinator, einem oder mehreren unterstellt. Abhängig davon können sie einem Einsatzgebiet zugeordnet sein. Ihre primäre Aufgabe ist die Bekämpfung von Zielen. Je nach ihren Eigenschaften können Leistungserbringer bis zu drei weitere Rollen ausüben: die des Anforderers, des Markierers und Wirkungsanalyst.
- Ziele (Objekt) - spiegeln in dem Modell den sogenannten Handlungsauslöser wider. Auf Ziele müssen die anderen Agenten reagieren und sie bekämpfen.

Der abstrakte Agent Einheit und seine Attribute werden hier im Folgenden erläutert. Die restlichen drei Agenten und das Objekt werden in ihren späteren Kapiteln dieser Arbeit (2.2. bis 2.4.) noch weiter spezifiziert.

- Bewegungsgeschwindigkeit - in [Koordinaten/ZE]. Dies ist die Geschwindigkeit, mit der sich das Objekt bewegen kann. Es kann nicht wählen, ob es schneller oder langsamer fahren will. Die endgültige Geschwindigkeit ist jedoch auch von den Bewegungseinschränkungen abhängig, die sich durch den Untergrund ergeben (siehe Umwelt).
- Startpunkt - ist die Koordinate, auf der das Objekt bei Simulationsstart platziert wird (Koordinatentupel [X, Y]).
- Plattform - das Medium, dem das Objekt angehört. [Schiff/Flugzeug/Fahrzeug/ Infanterie]
- Sichtweite - in Anzahl Koordinaten (Wertebereich: Anzahl $\in \mathbb{N}$). Dies beinhaltet die Koordinate, auf der die Einheit steht. Bewegt sich die Einheit, so ist die Sichtweite von seinem neuen Standort abhängig. Dieses Attribut ist sehr wichtig für den Sensor

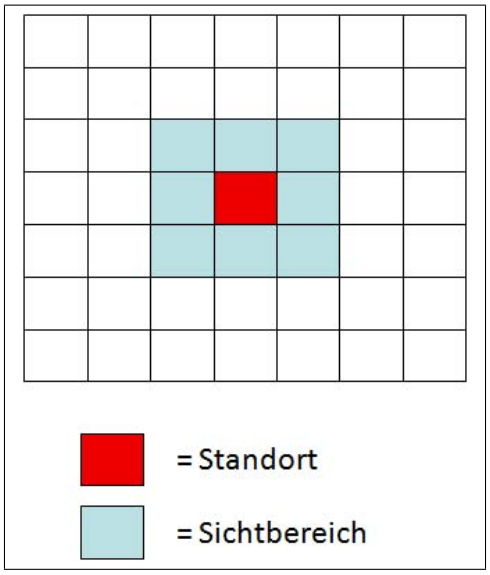

Abbildung 2.5: Die Sichtweite einer Einheit. Es ist dabei wichtig, dass der Standort der Einheit immer mitgezählt wird. Im oben gezeigten Beispiel hat die Einheit also eine Sichtweite von 2.

Wahrnehmung, den jede Einheit besitzt. In Abbildung 2.5 ist eine Einheit mit einer Sichtweite von zwei Koordinaten gezeigt. Der Standort wird also mitgezählt, was bedeutet, dass Einheiten mit einer Sichtweite von null blind sind (z.B. hochfliegender Bomber).

- Markierungsfähigkeit - Darstellung der Fähigkeit, ob die Rolle des Markierers wahrgenommen werden kann als boolescher Wert [Markierungsfähig: Ja/Nein]
- Koordinationsknoten - Ordnet der Einheit 0 bis n Koordinatoren aus der Menge der vorhandenen Koordinatoren zu. Koordinationsknoten = $(k_1,\ldots,k_i)$; $i \in \mathbb{N}$
- Geografie - die eigene Einheit besitzt das Wissen über die komplette Kartengeometrie und die jeweils eingeteilten Verantwortungsbereiche. Dies bedeutet, sie kennt zu jeder Koordinate (x,y) das Landschaftsmuster und alle verantwortlichen Koordinatoren.

Ereignisse und Informationsaustauschbeziehungen

Es werden vier verschiedene Arten von Ereignissen unterschieden[7], die hier aufgelistet werden. Auffällig ist hier, dass es kein Ereignis gibt, dass von einem Objekt ausgelöst wird und direkt ein anderes beeinflusst. Das liegt daran, dass immer der Umweg über die Umwelt gegangen werden muss. Dabei ist es oft der Fall, dass solche Ereignisse doppelt aufgelistet sind. Der Grund dafür ist, dass das Ereignis einmal von dem Agenten, der es auslöst, an die Umwelt abgegeben wird und einmal von dem empfangenden Objekt aus der Umwelt aufgenommen wird. Im Folgenden werden alle Ereignisse, die auftreten

können erläutert. Wenn dabei Informationen ausgetauscht werden, so werden diese im Anschluss an die Auflistung erläutert.

i. Ereignisse, die im Gesamtmodell Einsatzraum entstehen und auch nur dieses direkt beeinflussen:

- *Erzeugen eines Zieles* Dieses Ereignis erzeugt auf einer zufälligen Koordinate der Karte ein Ziel. Dabei wird dessen favorisiertes Gelände beachtet. Auf diesem favorisierten Gelände ist die Wahrscheinlichkeit des Auftretens fünf mal so hoch.

ii. Ereignisse, die aus dem Gesamtmodell Einsatzraum kommen und direkt auf Objekte einwirken:

- *Anforderung von Wirkung* Eine Wirkungsanforderung, die von jedem Agenten erstellt werden kann, geht bei einem Koordinator oder Leistungserbringer ein.
- *Anforderung von Markierung* Eine Markierungsanforderung, die von jedem Koordinator oder Leistungserbringer erstellt werden kann, geht bei irgendeinem der Agenten ein.
- *Anforderung von Wirkungsanalyse* Eine Markierungsanforderung, die von jedem Koordinator oder Leistungserbringer erstellt werden kann, geht bei irgendeinem der Agenten ein.
- *Priorisierungswerte des Broadcast-Channels lesen* Sobald alle Agenten auf einen Broadcast geantwortet haben, überprüft jeder ob er der Agent mit dem höchsten Priorisierungswert ist.
- *Anweisung zum Wirken* Es wird ein Wirkungsauftrag einem Leistungserbringer zugewiesen.
- *Anweisung zum Markieren* Es wird ein Markierauftrag einem Leistungserbringer oder Aufklärer zugewiesen.
- *Anweisung zum Analysieren* Es wird ein Wirkungsanalyseauftrag einem Leistungserbringer oder Aufklärer zugewiesen.
- *Markieren eines Zieles* Mit diesem Ereignis wird ein Ziel markiert, sodass es bekämpft werden kann.
- *Bekämpfung eines Zieles* Dieses Ereignis entfernt ein Ziel von der Karte.

iii. Ereignisse, die von Objekten ausgelöst werden und nur das gleiche Objekt beeinflussen:

- *Aufklären eines Ziels* Ein Leistungserbringer oder Aufklärer entdeckt ein Ziel und erweitert damit die Menge seiner bekannten Ziele.
- *Analysieren eines Zieles* Ein Leistungserbringer oder Aufklärer analysiert ein Ziel und stellt dadurch fest, ob es noch vorhanden ist.

- *Eignungsanalyse* Jeder Koordinator oder Leistungserbringer stellt durch dieses Ereignis seine Eignung für einen Auftrag fest.
- *Einsatzzeit ausreichend* Durch dieses Ereignis stellt ein Leistungserbringer fest, ob er in der Lage ist, einen Auftrag zu erfüllen und dann zum Startpunkt zurückzukehren.
- *Verfügbarkeitsmatrix aktualisieren* Ein Koordinator aktualisiert seine Verfügbarkeitsmatrix für eine konkrete Anforderung.
- *Anforderung in Anforderungsliste eintragen* Wenn ein Koordinator bereits eine Anforderung bearbeitet, so wird eine neue eingehende Anforderung in eine Warteliste gelegt.

iv. Ereignisse, die von Objekten generiert werden und die Gesamtmodellebene Einsatzraum beeinflussen:

- *Bewegung* Dieses Ereignis verändert die Position eines Objektes innerhalb der Karte.
- *Bekämpfung eines Zieles* Dieses Ereignis löst ein Leistungserbringer aus, wenn die Wirkung, abhängig von der Treffergenauigkeit, erfolgreich war.
- *Markieren eines Zieles* Mit diesem Ereignis markiert ein Leistungserbringer ein Ziel.
- *Anweisung zum Wirken* Dieses Ereignis wird erzeugt, wenn ein Leistungserbringer oder ein Koordinator einen Wirkungsauftrag vergibt.
- *Anweisung zum Markieren* Dieses Ereignis wird erzeugt, wenn ein Leistungserbringer oder ein Koordinator einen Markierauftrag vergibt.
- *Anweisung zum Analysieren* Dieses Ereignis wird erzeugt, wenn ein Leistungserbringer oder ein Koordinator einen Wirkungsanalyseauftrag vergibt.
- *Anforderung von Wirkung* Durch dieses Ereignis wird eine Wirkungsanforderung durch einen Agenten erzeugt.
- *Anforderung von Markierung* Durch dieses Ereignis wird eine Markieranforderung durch einen Koordinator oder Leistungserbringer erzeugt.
- *Anforderung von Wirkungsanalyse* Durch dieses Ereignis wird eine Wirkungsanalyseanforderung durch einen Koordinator oder Leistungserbringer erzeugt.
- *Priorisierungswert in Broadcast-Channel schreiben* Dieses Ereignis sendet den Wert, der durch die Priorisierungsmethode eines Leistungserbringers oder Koordinators ermittelt wurde, in den Broadcast-Channel.

Die Agenten können untereinander eine Vielzahl von Informationen und Anforderungen austauschen:

1. Wirkungsanforderung:

Zeitpunkt der Identifizierung t_0	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$
Position des Ziels zu t_0	Koordinate (X, Y)
Bewegungsgeschwindigkeit des Ziels v_Z	$v \in \mathbb{N}$; $v \geq 0$
Bewegungsrichtung des Ziels	[West/Nord/Ost/Süd]
Gefahrenpotential des Ziels	[gering/mittel/hoch]

2. Markierungsanforderung:

Zeitpunkt der Identifizierung t_0	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$
Position des Ziels zu t_0	Koordinate (X, Y)
Bewegungsgeschwindigkeit des Ziels v_Z	$v \in \mathbb{N}$; $v \geq 0$
Bewegungsrichtung des Ziels	[West/Nord/Ost/Süd]

3. Eigener Status: Enthält den Wert, den der Leistungserbringer abhängig von der gängigen Priorisierungsmethode ermittelt hat.

Status	$S \in \mathbb{N}$; $S \geq 0$

4. Markierauftrag:

Zeitpunkt der Identifizierung t_0	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$
Position des Ziels zu t_0	Koordinate (X, Y)
Bewegungsgeschwindigkeit des Ziels v_Z	$v \in \mathbb{N}$; $v \geq 0$
Bewegungsrichtung des Ziels	[West/Nord/Ost/Süd]

5. Wirkungsauftrag:

Zeitpunkt der Identifizierung t_0	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$
Position des Ziels zu t_0	Koordinate (X, Y)
Bewegungsgeschwindigkeit des Ziels v_Z	$v \in \mathbb{N}$; $v \geq 0$
Bewegungsrichtung des Ziels	[West/Nord/Ost/Süd]
Zeitpunkt der Markierung t_1	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$

6. Wirkungsanalyseauftrag:

Zeitpunkt der Identifizierung t_0	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$
Position des Ziels zu t_0	Koordinate (X, Y)
Bewegungsgeschwindigkeit des Ziels v_Z	$v \in \mathbb{N}$; $v \geq 0$
Bewegungsrichtung des Ziels	[West/Nord/Ost/Süd]
Zeitpunkt der Wirkung t_2	Zeiteinheit; $t \in \mathbb{N}$; $t \geq 0$

7. Wirkungsanalyse: Enthält die Information, ob ein Ziel vernichtet wurde oder nicht.

Ziel vernichtet	boolean

8. Auftrag abgeschlossen: Enthält die Information, dass das Ziel bekämpft wurde und alle Einheiten wieder ihrer normalen Aufgabe nachgehen können.

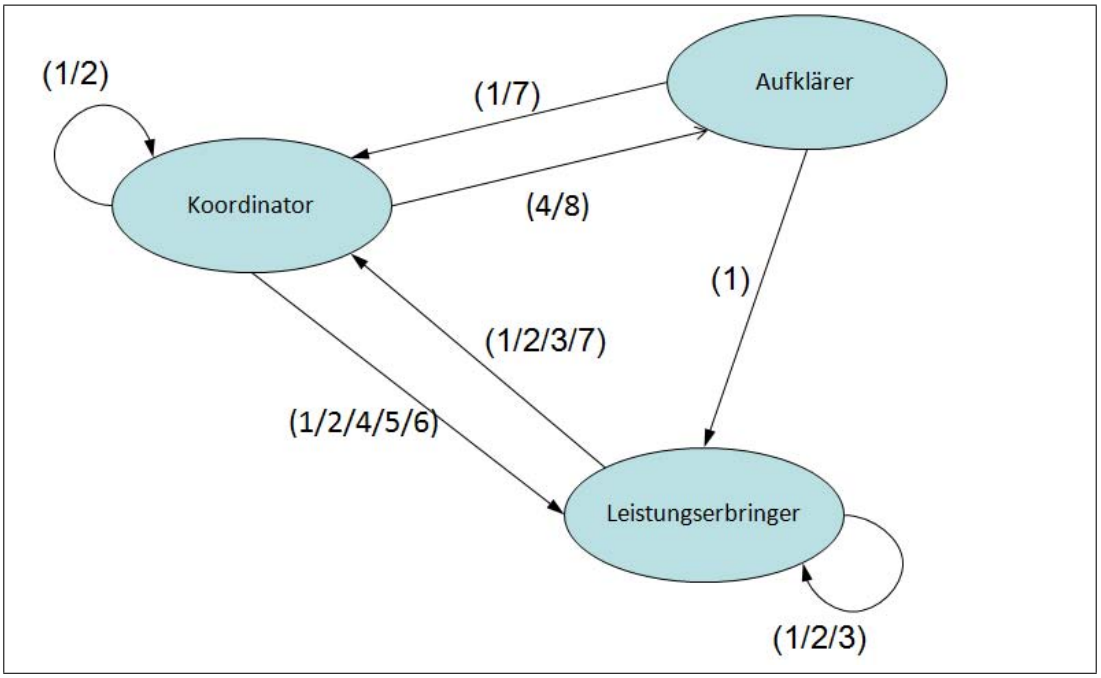

Abbildung 2.6: Diese Informationsaustauschbeziehungen können innerhalb des Modells zwischen den Agenten auftreten.

In Abbildung 2.6 ist ersichtlich, welcher Agent welche Art von Information an andere Agenten weitergibt.

- Aufklärer - Koordinator

  (1) Wenn der Aufklärer dem Koordinator unterstellt ist, so geht die Wirkungsanforderung an den Koordinator.

  (7) Wenn der Aufklärer nach Anforderung am Ziel bleibt, so übernimmt er auch die Wirkungsanalyse.

- Aufklärer - Leistungserbringer

  (1) Wenn der Aufklärer keinem Koordinator unterstellt ist, so geht die Wirkungsanforderung an alle Leistungserbringer ohne Koordinationsknoten.

- Koordinator - Aufklärer

  (4) Wenn die Art des Leistungserbringers Markierung erfordert, so überträgt der Koordinator dem anfordernden Aufklärer den Auftrag zur Markierung, falls dieser die Fähigkeit dazu besitzt.

(8) Wenn das Ziel erfolgreich bekämpft wurde, so muss der Aufklärer nicht weiter in der Nähe des Ziels warten, um es unter Umständen noch einmal zu markieren.

- Koordinator - Koordinator

(1) Ist der Koordinator nicht in der Lage, eine Wirkungsanforderung zu erfüllen, so gibt er die Anforderung an alle übergeordneten, oder (falls er keine weiteren Vorgesetzten hat) an alle gleichberechtigten Koordinatoren weiter.

(2) Ist der Koordinator nicht in der Lage, eine Markierungsanforderung zu erfüllen, so gibt er die Anforderung an alle übergeordneten, oder (falls er keine weiteren Vorgesetzten hat) an alle gleichberechtigten Koordinatoren weiter.

- Koordinator - Leistungserbringer

(1) Ist der Koordinator nicht in der Lage, eine Wirkungsanforderung zu erfüllen und hat er keinen Vorgesetzten, so gibt er die Anforderung an alle gleichberechtigten Leistungserbringer weiter.

(2) Ist der Koordinator nicht in der Lage, eine Markierungsanforderung zu erfüllen und hat er keinen Vorgesetzten, so gibt er die Anforderung an alle gleichberechtigten Leistungserbringer weiter.

(4) Erfordert das Ziel Markierung, so teilt der Koordinator einem untergeordneten Leistungserbringer den Auftrag zur Markierung zu.

(5) Der Koordinator überträgt einem untergeordneten Leistungserbringer den Auftrag zur Bekämpfung eines Zieles.

(6) Ist der Anforderer nicht mehr in der Nähe des Zieles, erfordert die Wirkung keine Markierung und kann die Wirkungsanalyse nicht vom Wirkmittel selber vorgenommen werden, so teilt der Koordinator einen weiteren Leistungserbringer zur Wirkungsanalyse ein.

- Leistungserbringer - Koordinator

(1) Wenn der Leistungserbringer einem Koordinator unterstellt ist, so geht die Wirkungsanforderung an den Koordinator.

(2) Wenn der Leistungserbringer keinem Koordinator unterstellt ist, jedoch Markierung benötigt, so geht die Markierungsanforderung an gleichgestellte Koordinatoren.

(3) Ist der Leitungserbringer keinem Koordinator unterstellt und gibt ein Koordinator in einem Broadcast-Channel eine Anforderung an gleichgestellte Leistungserbringer weiter, dann gibt der Leistungserbringer seinen Status an.

(7) Soll der Leistungserbringer eine Wirkungsanalyse vornehmen, so gibt er diese an den Koordinator zurück.

- Leistungserbringer - Leistungserbringer
  (1) Ist der Leistungserbringer keinem Koordinator unterstellt, so geht die Wirkungsanforderung an gleichgestellte Leistungserbringer.
  (2) Ist der Leistungserbringer keinem Koordinator unterstellt, so geht die Markierungsanforderung an gleichgestellte Leistungserbringer.
  (3) Ist der Leitungserbringer keinem Koordinator unterstellt und gibt ein Leistungserbringer in einem Broadcast-Channel eine Anforderung an gleichgestellte Leistungserbringer weiter, dann gibt jeder Leistungserbringer seinen Status an.

Rahmenbedingungen

Eingrenzende Bedingungen: Dies sind Rahmenbedingungen, die den möglichen Rahmen von Aktionen einschränken. Dabei müssen diese Bedingungen zu jedem Zeitpunkt der Simulation eingehalten werden und können dabei auch verschiedene Aktionen auf einmal einbeziehen. (Quelle GRAMS, 5.2.5.1.) [7]

1. Es können keine Ziele auf Koordinaten erzeugt werden, die bereits von einem Aufklärer oder Leistungserbringer besetzt sind.
2. Nur die Agenten können sich in der Luft bewegen, deren Attribut für Plattform Flugzeug beträgt. Alle anderen Agenten sind an den Boden gebunden.
3. Nur die Agenten können sich auf See bewegen, deren Attribut für Plattform Flugzeug oder Schiff beträgt. Alle anderen Agenten sind an die restlichen Landschaftsmuster gebunden.
4. Sobald die Einsatzzeit eines Leistungserbringers überschritten ist, muss er zum Startpunkt zurückkehren und mindestens für die Ruhezeit an diesem Ort bleiben.
5. Sobald ein Einsatz begonnen wurde, kann er nicht mehr abgebrochen werden. Ein Einsatz fängt dann an, wenn ein Leistungserbringer damit beginnt, seine Planung auf die Bekämpfung, Markierung oder Wirkunganalyse eines Zieles auszurichten. Als Teil dieser Planung leitet er in der Regel die Bewegung zum Ziel ein.

Bedingungen als Voraussetzungen: Diese Rahmenbedingungen sind immer an eine bestimmte Aktion gebunden. Bevor ein Agent eine Aktion ausführen kann, müssen bestimmte Voraussetzungen dieser Aktion erfüllt sein.

1. Jedes Ziel, das sich bewegt, muss markiert werden, bevor es bekämpft werden kann.
2. Jeder Leistungserbringer, dessen Art der Wirkleistung eine Zielmarkierung erfordert, benötigt diese immer bevor er ein Ziel bekämpfen kann.
3. Die Leistungserbringer, deren Wirkreichweite die Sichtweite übersteigt, benötigt immer Markierung.

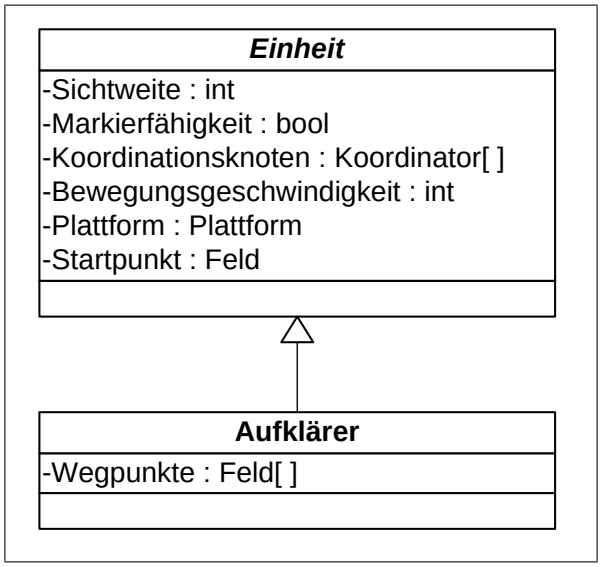

Abbildung 2.7: Das Klassendiagramm des Agenten Aufklärer zusammen mit seiner Superklasse.

4. Ein Leistungserbringer, der die Rolle des Wirkmittels innehat, kann nie gleichzeitig das zu bekämpfende Ziel markieren.
5. Damit ein Ziel zerstört wird, muss es zunächst bekämpft werden oder die Karte verlassen.
6. Damit ein Ziel bekämpft werden kann, muss sich dieses erst innerhalb der Wirkreichweite eines Ziels befinden.
7. Damit ein Ziel von einem Leistungserbringer markiert werden kann, muss sich dieses erst innerhalb der Sichtweite des Leistungserbringers befinden.

2.2.3 Agent Aufklärer

Beschreibung von Struktur und internem Verhalten

Attribute

Neben denen unter Kapitel 2.2.2 im Sinne der Vererbung von Einheit übernommenen Attributen hat dieser Agent noch das Attribut Wegpunkte:

- Wegpunkte - Angabe der Koordinaten für die Überwachungsroute. In der Liste können k Wegpunkte enthalten sein, wobei $k \in N^+$.
  Wertebereich:
  $(k_1, \ldots, k_n)$ mit $k_i = (x_i, y_i)$
  und $1 < x_i \leq X_k$ bzw. $1 < y_i \leq Y_k$
  wobei k = Ende der detaillierten Karte

Sensoren

- Visuelle Wahrnehmung - Durch diesen Sensor kann der Aufklärer Ziele in seiner Umgebung wahrnehmen und identifizieren. Die Reichweite dieses Sensors ist von seinem Attribut Sichtweite abhängig. Wenn durch diesen Sensor ein Ziel entdeckt wird, so wird dieses Ziel im Attribut Geografie als Inhalt einer Koordinate (x,y) gespeichert. Wenn dieser Aufklärer dieses Ziel vorher noch nicht identifiziert hat, so resultiert daraus das Erstellen einer Anforderung. Die Dauer dieses Sensors beträgt 1 ZE.

- Auftragswahrnehmung - Mit diesem Sensor kann der Aufklärer einen Auftrag zur Zielmarkierung, zur Wirkungsanalyse oder zum Fortsetzen der Patrouillentätigkeit erhalten. Dabei handelt es sich um einen globalen Sensor, d.h. er ist nicht an den Standort oder Status des Aufklärers gebunden, bzw. wird durch diese nicht beeinflusst. Die Dauer dieses Sensors beträgt 1 ZE.

Effektoren

- Bewegung - Der Aufklärer hat die Fähigkeit, sich von einem Wegpunkt zum nächsten zu bewegen. Dabei nimmt er immer den kürzesten Weg (minimale Anzahl Koordinaten). Mit diesem Effektor verändert der Aufklärer seine Position innerhalb der Karte um eine Koordinate entlang seiner durch die Wegpunkte definierten Patrouillenroute. Die Dauer dieses Effektors ist von der Zeit abhängig, welche benötigt wird, um die neue Koordinate zu erreichen.

- Anforderung aufgeben - Durch diesen Effektor gibt der Aufklärer ein gesichtetes Ziel an den zuständigen Koordinator oder alle Wirkmittel weiter, es wird also ein Ereignis Anforderung von Wirkung erstellt. Dies dauert 1 ZE.

- Ziel markieren - Markiert ein Ziel durch das Ereignis Markieren eines Zieles, sodass es durch das Wirkmittel bekämpft werden kann. Dafür benötigt der Agent 1 ZE.

- Ziel analysieren - Analysiert durch das Ereignis Analysieren eines Zieles, ob ein Ziel erfolgreich bekämpft wurde. Dadurch aktualisiert der Aufklärer die Koordinate (x,y) innerhalb seines Attributes Geografie, welche das Ziel enthält. Das Resultat, also ob das Ziel noch vorhanden ist oder ob die Koordinate wieder leer ist, ist das Ergebnis dieses Effektors und wird an den vorgesetzten Koordinator weitergegeben. Dieser Effektor hat die Dauer 1 ZE.

- Broadcast-Channel senden - Befähigt den Aufklärer in den Broadcast-Channel Meldungen zu veröffentlichen, indem er ein Ereignis Anforderung von Wirkung erstellt. Dies dauert 1 ZE.

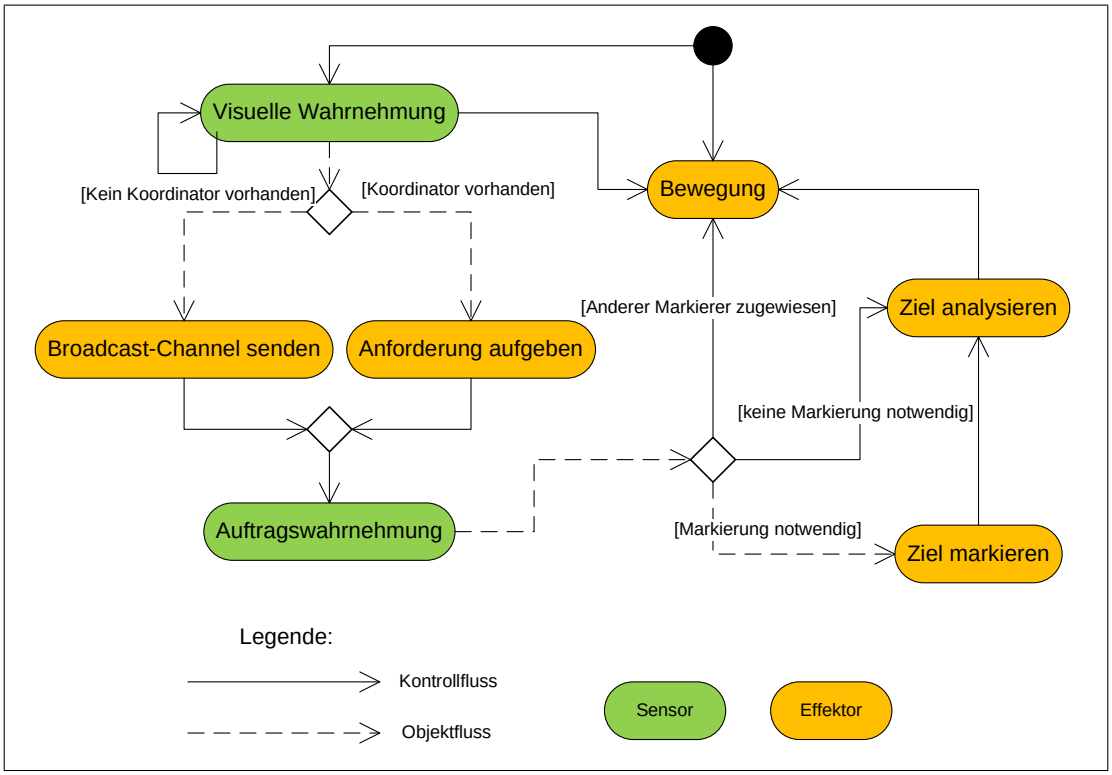

Abbildung 2.8: Diese Abbildung zeigt, welche Sensoren und Effektoren voneinander abhängig sind und nacheinander aktiv werden. Zu beachten ist, dass der Sensor Visuelle Wahrnehmung ständig aktiv ist.

Status

Der Status dieses Agenten gibt an, auf welcher Position er sich gerade befindet und ob er auf Patrouille ist (also sich bewegt) oder an seiner Position wartet, um ein Ziel zu markieren.

$$Status_{\text{Aufklärer}} = \text{Position}(X, Y),\ [\text{Patrouille/Warten/Markieren}]$$

Verhalten

In Abbildung 2.8 ist dargestellt, welche Sensoren und Effektoren voneinander abhängig aktiv werden. Zum Simulationsstart bewegt sich der Aufklärer zunächst von seinem Startpunkt zum ersten Wegpunkt der Liste seiner Wegpunkte. Anschließend bewegt er sich immer iterativ zum nächsten Wegpunkt dieser Liste. Nach dem letzten geht er automatisch wieder direkt zum ersten Wegpunkte zurück und beginnt diese Liste von vorn. Abhängig von dem Attribut Sichtweite kann ein Aufklärer Ziele wahrnehmen

und darauf aufbauend eine Anforderung erstellen. Wenn er sein Wahrnehmungsgebiet verändert, indem er seine Position innerhalb des Einsatzraumes durch den Effektor Bewegung ändert, so kann es sein, dass sich nun ein Objekt vom Typ Ziel innerhalb seines Sichtfeldes befindet. Wenn dies der Fall ist, so besteht abhängig vom Attribut Entdeckungswahrscheinlichkeit des Ziels die Wahrscheinlichkeit, dass der Aufklärer dieses Ziel entdeckt. Wenn er jedoch das Ziel nicht entdeckt, so bewegt er sich ebenfalls weiter. Sollte das Ziel dabei entweder seinen Sichtbereich verlassen und ihn wieder betreten oder innerhalb des Wahrnehmungsgebietes seine Position innerhalb des Einsatzraumes verändern, so besteht wieder die gleiche Wahrscheinlichkeit, dass das Ziel entdeckt wird.

Wenn der Aufklärer ein Ziel identifiziert, so stellt er durch den Sensor Visuelle Wahrnehmung die Eigenschaften des Ziels fest; diese sind:

- Position zum Zeitpunkt der Identifizierung
- Gefährlichkeit
- Bewegungsrichtung
- Geschwindigkeit
- Plattform
- Zeitpunkt der Identifizierung

Anschließend gibt er über den Effektor Anforderung aufgeben die Meldung mit diesen Daten an den zuständigen Koordinator weiter. Welcher dies ist, weiß der Agent durch sein Attribut Geografie. Sollte es für das Gebiet keinen zuständigen Koordinator geben, so gibt der Aufklärer diese Anforderung in den Broadcast-Channel auf. Sobald für diesen Auftrag ein Wirkmittel ausgewählt wurde, erhält der Aufklärer über den Sensor Auftragswahrnehmung den Befehl für sein weiteres Vorgehen. Diese Anweisung erhält er von dem zuständigen Koordinator, oder von dem Wirkmittel, welches den Auftrag durch den Broadcast-Channel angenommen hat. Wenn der Aufklärer markieren kann, die Art des Wirkmittels eine Markierung erfordert oder das Ziel sich bewegt, so wird der Auftrag darin bestehen, das Ziel zu markieren und anschließend zu analysieren. Sollte keine Markierung notwendig sein, so analysiert der Aufklärer dennoch das Ziel. Ist jedoch eine Markierung notwendig, der Aufklärer dazu aber nicht in der Lage, so wird er auch keine Zielanalyse vornehmen, sondern mit seiner Patrouillentätigkeit fortfahren, da ein weiterer Leistungserbringer abgestellt werden wird, um das Ziel zu markieren. Dieser übernimmt dann auch die Wirkungsanalyse. Spätestens nachdem die Wirkung analysiert wurde, setzt der Aufklärer seine Patrouille fort.

Formale Beschreibung von Submodell Eingabewerten

- Bewegungsgeschwindigkeit - $[v]$ = Koordinaten/ZE; $v \in \mathbb{N}$; $v \geq 0$
- Startpunkt - Koordinatentupel $[X, Y]$
- Plattform - [Schiff/Flugzeug/Fahrzeug/Infanterie]

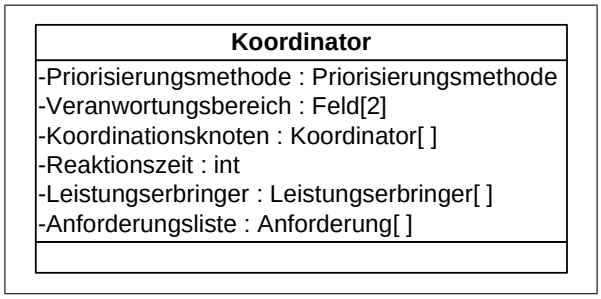

Abbildung 2.9: Das Klassendiagramm des Agenten Koordinator.

- Markierungsfähigkeit - [Ja/Nein]
- Koordinationsknoten - $(k_1,\ldots,k_i)$; $i \in \mathbb{N}$
- Wegpunkte - $(w_1,\ldots,w_i)$; $i \in \mathbb{N}$) ; w_i = Koordinatentupel (X, Y).

Beschreibung von Interaktionen zwischen Submodellen

Ein Zustandswechsel erfolgt durch das Objekt Ziel und anschließend durch den Agenten Koordinator. Interaktionen finden dabei durch das Abgeben einer Meldung über Position, Bewegungsgeschwindigkeit, Richtung, Plattform, Gefahrenpotential und Zeitpunkt der Identifizierung an einen Koordinator oder falls es keinen gibt an gleichberechtigte Leistungserbringer statt.

2.2.4 Agent Koordinator

Beschreibung von Struktur und internem Verhalten

Attribute

- Priorisierungsmethode - Hier kann zwischen drei verschiedenen Methoden der Auswahl des geeignetsten Wirkmittels unterschieden werden (Wertebereich [Entfernung/ Gefahrenpotential/Mischpriorisierung]). Abhängig von der Art des Ziels kann es notwendig sein, dass für ein Ziel zwei Leistungserbringer ausgewählt werden müssen: Ein LE zur Wirkung und ein zweiter zur Markierung und Wirkungsanalyse oder der erste LE zur Wirkung und ein zweiter nur zur Wirkungsanalyse, falls keine Markierung notwendig ist. Ist sie nicht erforderlich, so nimmt der besser geeignete die Wirkung vor, während der andere die Analyse durchführt. Wenn jedoch eine Markierung erforderlich ist, so müssen beide LE am frühestmöglichen Zeitpunkt in Zielreichweite sein. Deshalb wird der schnellere LE zur Markierung eingesetzt, um dafür zu sorgen, dass der zweite LE sofort ohne Wartezeit wirken kann. Die drei Priorisierungsmethoden werden im Folgen beschrieben:
  1 Entfernung - Es wird der LE ausgewählt, der das Ziel am schnellsten bekämpfen kann. Auch bereits in einem Auftrag gebundene LE werden betrachtet,

Zeit zum Ziel (Angabe in ZE)	Gefahr -gering- (10 Pkt.) Priorität= Entfernung X Gefahr	Gefahr -mittel- (5 Pkt.) Priorität= Entfernung X Gefahr	Gefahr -hoch- (1 Pkt.) Priorität= Entfernung X Gefahr
10	100	50	10
20	200	100	20
30	300	150	30
40	400	200	40
100	1000	500	100
500	5000	2500	500
1000	10000	5000	1000
t	10*t	5*t	t

Abbildung 2.10: Die Priorisierungsmethode Mischpriorisierung. Bei dieser Priorisierungsmethode erfolgt eine Gewichtung anhand der Entfernung (benötigte Zeit) zum Ziel und dessen Gefährlichkeit. Der Agent mit dem niedrigsten Wert ist der am meisten Geeignete.

wobei die Zeit, die sie brauchen, um ihren aktuellen Auftrag abzuschließen, mit in der Zeit berücksichtigt wird, die sie für die Bewegung zum neuen Wirkungsort benötigen. (Ein bereits gebundener LE, der sich in geringer Entfernung zum neuen Ziel befindet, kann also durchaus besser geeignet sein als ein ungebundener LE, der sehr weit vom Ziel entfernt ist.)

2 Gefahrenpotential - Es wird der LE ausgewählt, der aufgrund seiner Bekämpfungskapazität am besten geeignet ist. Wird zum Beispiel ein Ziel mit niedrigem Gefahrenpotential identifiziert, so erfolgt zunächst die Zuteilung eines Leistungserbringers mit Bekämpfungskapazität niedrig. Das ist unabhängig von der Entfernung zum Ziel und hat den Zweck, LE mit hoher Kapazität für hohe Gefahrenpotentiale frei zu halten. Erst wenn alle LE mit niedriger Kapazität gebunden sind, wird über die mit hoher Kapazität verfügt.

3 Mischpriorisierung - Es wird eine Kombination aus den beiden oben genannten Methoden angewandt, die die Vorteile beider nutzen soll.

Dabei wird für jeden LE ein Faktor berechnet, der seine Eignung für das jeweilige Ziel angibt (Abb. 2.10). Je niedriger dieser Wert, umso geeigneter ist der Leistungserbringer für die Bekämpfung. Es wird also der LE mit dem geringsten Eignungsfaktor für die Bekämpfung eines Ziels ausgewählt. Bei dieser Methode wird sowohl die nötige Zeit für bereits laufende Aufträge beachtet (falls der LE gebunden ist), als auch die Eignung des Leistungserbringers für die Bekämpfung des Ziels (siehe zweite Priorisierungsmethode).

- Verantwortungsbereich - Einteilung anhand von Koordinaten (Wertebereich: zwei Koordinatentupel [X,Y],[X,Y]). Der Verantwortungsbereich ist immer rechteckig.
- Reaktionszeit - Der Parameter, den der Koordinator durchschnittlich für die Bearbeitung eines Auftrags von Meldungseingang bis Wirkmittelzuteilung benötigt (in Zeiteinheiten).
- Koordinationsknoten - Ordnet dem Koordinator weitere übergeordnete Koordinatoren der Anzahl 0..n zu (Wertebereich: $(k_1,\ldots,k_n)$; $n \in \mathbb{N}$).
- Unterstellte Leistungserbringer - Dem Koordinator ist eine Liste mit zugehörigen Leistungserbringern und auch weiteren Koordinatoren zugeordnet. Diesen Einheiten ist dieser Agent vorgesetzt.
- Anforderungsliste - Diese Liste beinhaltet alle Anforderungen, die zwar eingegangen, aber noch nicht bearbeitet wurden.
- Verfügbarkeitsmatrix - In dieser Tabelle sind alle unterstellten Leistungserbringer aufgelistet. Für jeden Leistungserbringer wird der laut Priorisierungsmethode zuletzt ermittelte Wert aufgeführt.

Sensoren

- Anforderungswahrnehmung - Mit diesem Sensor kann der Koordinator eine Anforderung zur Zielmarkierung, zur Wirkungsanalyse oder zur Wirkung erhalten. Dabei handelt es sich um einen globalen Sensor, d.h. er ist nicht an den Standort oder Status des Koordinators gebunden, bzw. wird durch diese nicht beeinflusst. Nachdem eine Anforderung durch diesen Sensor eingegangen ist, wird sie mit dem Effektor Anforderung speichern in das Attribut Anforderungsliste eingetragen. Dies dauert 1 ZE.
- Status abrufen - Durch diesen endogenen Sensor nimmt der Koordinator die Position und den Zustand seiner unterstellten Leistungserbringer wahr. Dabei wird durch diesen Sensor der Priorisierungswert eines unterstellten Leistungserbringers ermittelt und in das Attribut Verfügbarkeitsmatrix eingetragen. Dafür wird eine Dauer von 1 ZE benötigt.
- Broadcast-Channel lesen - Befähigt den Koordinator den Broadcast-Channel zu empfangen. Dadurch kann er ebenfalls Anforderungen annehmen und diese mit

dem Effektor Anforderung speichern in seinem Attribut Anforderungsliste aufnehmen. Dies dauert 1 ZE.

Effektoren

- Anforderung aufgeben - Durch diesen Effektor gibt der Koordinator ein ihm bekanntes Ziel an den zuständigen, nächsthöheren Koordinator oder alle benachbarten Wirkmittel weiter; es wird also ein Ereignis Anforderung von Wirkung erstellt. Dies dauert 1 ZE.
- Verfügbarkeitsmatrix aktualisieren - Mit diesem endogenen Effektor überprüft der Koordinator alle seine unterstellten Leistungserbringer und kann sie miteinander vergleichen. Er kontrolliert dabei sein Attribut Verfügbarkeitsmatrix und wählt den Leistungserbringer aus, der am besten geeignet ist. Dafür wird 1 ZE benötigt.
- Auftrag erteilen - Entfernt die erste Anforderung des Attributes Anforderungliste und bearbeitet diese. Dabei kann ein Aufklärer damit beauftragt werden, ein Ziel zu markieren und zu analysieren, nur zu analysieren oder mit der Patrouille fortzufahren. Einem Leistungserbringer kann der Befehl zur Wirkung, zur Markierung oder zur Wirkungsanalyse gegeben werden. Der Auftrag enthält dabei die Daten der Zielidentifizierung und, falls benötigt, den Zeitpunkt der Wirkung und Markierung. Die Dauer dieses Effektors ergibt sich aus dem Attribut Reaktionszeit.
- Anforderung speichern - Mit diesem endogenen Effektor nimmt ein Koordinator eine eingegangene Anforderung in das Attribut mit dem Effektor Anforderungsliste auf. Dies dauert 1 ZE.
- Priorisierungswert berechnen - Mit diesem Effektor berechnet der Leistungserbringer seine Eignung für die Erfüllung eines Auftrages. Dafür wird 1 ZE benötigt.
- Broadcast-Channel senden - Befähigt den Koordinator im den Broadcast-Channel Meldungen zu veröffentlichen, indem er ein Ereignis Anforderung von Wirkung erstellt. Dies dauert 1 ZE.

Status

Der Status dieses Agenten gibt an, ob er gerade eine Anforderung bearbeitet oder untätig ist.

$$Status_{\text{Koordinator}} = \text{[ja/nein]}$$

Verhalten

In Abbildung 2.11 ist dargestellt, welche Sensoren und Effektoren voneinander abhängig aktiv werden. Der Agent Koordinator wartet zunächst auf eine eingehende Anforderung. Sobald er über den Sensor Anforderungswahrnehmung eine Anforderung empfängt, speichert

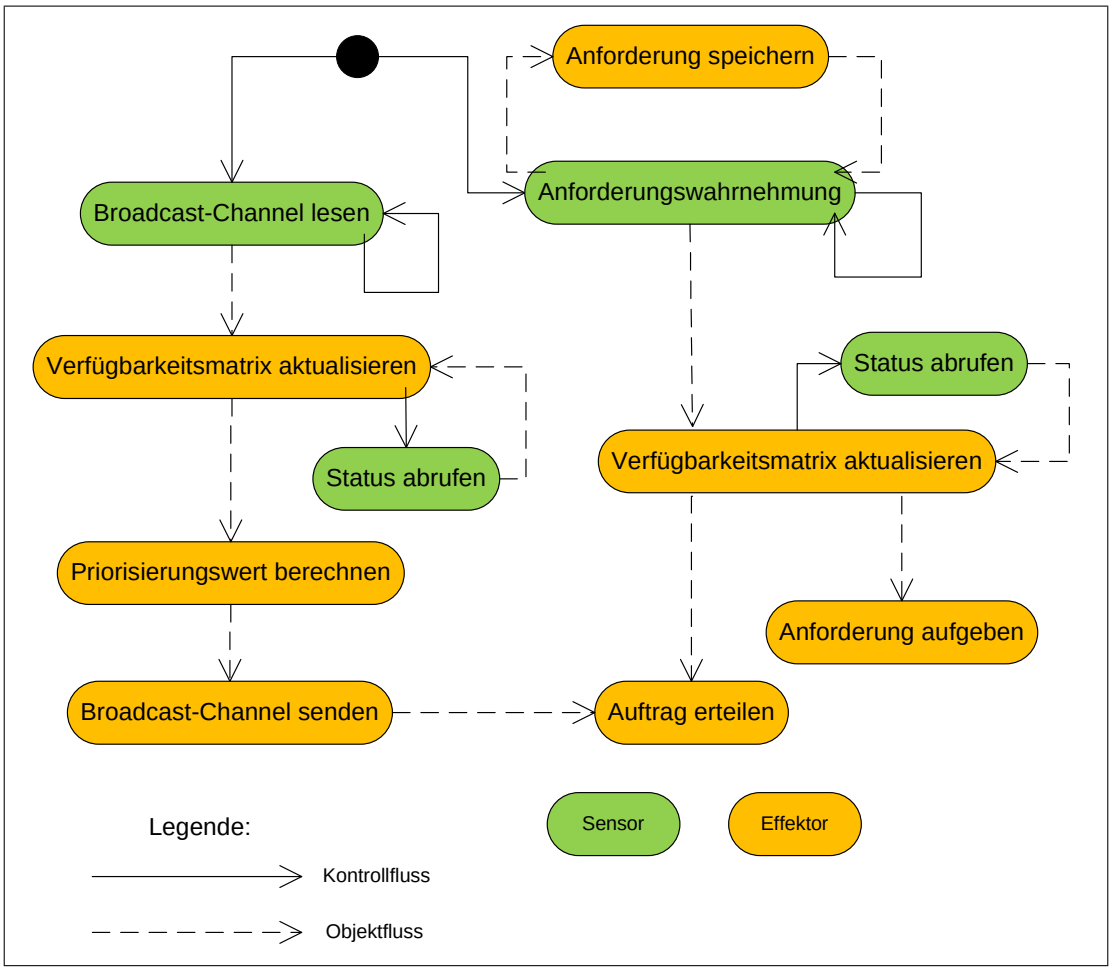

Abbildung 2.11: Diese Abbildung zeigt, welche Effektoren und Sensoren in welcher Reihenfolge aktiv werden. Zu beachten ist, dass die beiden Sensoren Anforderungswahrnehmung und Broadcast-Channel lesen ständig aktiv sind und Anforderungen aufnehmen können.

er sie mit dem endogenen Effektor Anforderung speichern in seinem Attribut Anforderungsliste ab. Der Koordinator wird immer die erste Anforderung in dieser Liste bearbeiten. Er benötigt dafür eine Reaktionszeit, die um sein Attribut Reaktionszeit (Erwartungswert) bei einer Standardabweichung von 20% der Reaktionszeit normalverteilt ist. Dabei benutzt er den endogenen Sensor Status abrufen, um von jedem seiner unterstellten Leistungserbringer einen Wert gemäß dem Attribut Priorisierungsmethode des Koordinators zu bestimmen. Eine detaillierte Beschreibung dieser Methode ist in der Attributbeschreibung zu finden. Hat der Koordinator den geeigneten Leistungserbringer ermittelt, so prüft er, ob weitere für die Erfüllung dieser Anforderung benötigt werden. Falls ja, dann ermittelt er diese auf dem gleichen Weg. Sobald genügend Leistungserbringer für die Erfüllung des Auftrages zur Verfügung stehen, übermittelt er mit dem exogenen Effektor Auftrag erteilen den Befehl an die Leistungserbringer. Dieser beinhaltet:

- Position zum Zeitpunkt der Entdeckung
- Geschwindigkeit des Ziels
- Bewegungsrichtung des Ziels
- Zeitpunkt der Entdeckung
- Zeitpunkt der Markierung und Wirkung.

Letztgenannter Zeitpunkt ist der früheste Zeitpunkt, an dem alle Leistungserbringer das Ziel in ihrer Sicht- (Markierung, Wirkungsanalyse) bzw. Wirkreichweite (Wirkung) haben können. Anschließend nutzt der Koordinator den gleichen Effektor um dem Aufklärer, der die Anforderung aufgegeben, hat sein weiteres Vorgehen, ob Markierung und Zielanalyse, nur Zielanalyse oder Fortsetzen der Patrouille, zu befehlen. Ist dies getan, so bearbeitet der Koordinator die nächste Anforderung seiner Anforderungsliste oder wartet auf weitere Anforderungen, falls diese Liste leer ist. Sollte der Koordinator bei dem Aktualisieren seiner Verfügbarkeitsmatrix feststellen, dass er die Anforderung nicht erfüllen kann, so nutzt er den Effektor Anforderung aufgeben, um diese an seinen nächsthöheren Koordinator oder an benachbarte Leistungserbringer im Broadcast-Channel weiterzuleiten. Der Koordinator kann auch mithilfe des Sensors Broadcast-Channel lesen Anforderungen empfangen. Sollte dies der Fall sein, so aktualisiert er seine Verfügbarkeitsmatrix wie oben beschrieben und wählt den besten Wert mit dem Effektor Priorisierungswert berechnen aus. Dieser wird dann im Broadcast-Channel mithilfe des Effektors Broadcast-Channel senden herausgegeben. Falls der Koordinator den besten Wert veröffentlicht hat, so nimmt er den Auftrag an und vergibt ihn weiter an den besten Leistungserbringer wie oben beschrieben.

Formale Beschreibung von Eingabewerten

- Priorisierungsmethode - [Entfernung/Gefahrenpotential/Mischpriorisierung]
- Verantwortungsbereich - zwei Koordinatentupel ($[X_1,Y_1]$,$[X_2,Y_2]$)
- Reaktionszeit - $[t] = \mathsf{ZE}$; $t \in \mathbb{N}$; $t \geq 0$
- Koordinationsknoten - $(k_1,\ldots,k_i)$; $i \in \mathbb{N}$

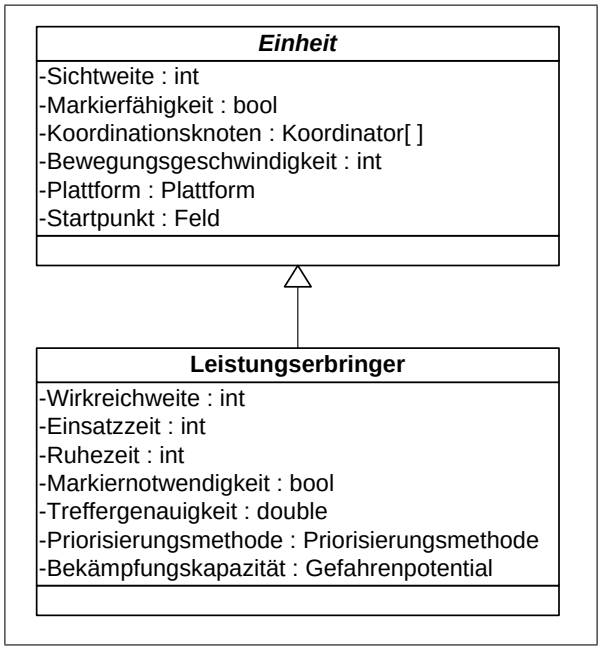

Abbildung 2.12: Das Klassendiagramm des Agenten Leistungserbringer zusammen mit seiner Superklasse.

Beschreibung von Interaktionen zwischen Submodellen

Ein Zustandswechsel erfolgt zum einen durch die Agenten Aufklärer, Leistungserbringer oder Koordinator, indem eine Anforderung von Wirkung, Markierung und/oder Wirkungsanalyse eingeht. In dieser Anforderung ist die Position des Ziels zum Zeitpunkt der Entdeckung, die festgestellte Bewegungsgeschwindigkeit, das Gefahrenpotential und der Zeitpunkt der Entdeckung enthalten. Ebenfalls findet eine Interaktion mit dem Submodell Leistungserbringer statt, wenn ein Auftrag zur Markierung, Wirkung oder Wirkungsanalyse vergeben wird. Dabei wird Position während der Identifizierung, Bewegungsgeschwindigkeit, Zeitpunkt der Identifizierung und falls Markierung notwendig ist, der frühestmögliche Zeitpunkt der Markierung und Wirkung (da diese in der gleichen Zeiteinheit erfolgen müssen) übergeben. Falls der Auftrag nicht durch diesen Koordinator durchgeführt werden soll, wird er an einen anderen Koordinator weitergeleitet oder falls es keinen anderen gibt, an gleichberechtigte Leistungserbringer.

2.2.5 Agent Leistungserbringer

Beschreibung von Struktur und internem Verhalten

Attribute

Neben denen unter 2.2.2 im Sinne der Vererbung von Einheit übernommen Attributen hat dieser Agent noch folgende weiteren Attribute:

- Markiernotwendigkeit - Gibt an, ob die Art der erbrachten Leistung immer eine

Markierung notwendig macht (Wertebereich: boolescher Wert [Markiernotwendigkeit ja/nein]).

- Wirkreichweite - Die Anzahl der Koordinaten, die ein LE zum Wirken maximal vom Ziel entfernt sein kann (Wertebereich: Anzahl Koordinaten $n \in \mathbb{N}$). Die Wirkreichweite folgt den selben Restriktionen wie die Sichtweite (siehe dazu Abbildung 2.5). Das bedeutet, dass ein LE mit einer Wirkreichweite von eins nur auf seiner eigenen Koordinate wirken kann.

- Einsatzzeit - Die maximale Zeit, die ein LE im Einsatz bleiben kann (Wertebereich: $t \in \mathbb{N}; t \geq 0$).

- Ruhezeit - Die Zeit, die ein LE nach Erreichen seiner Einsatzzeit am Startpunkt bleiben muss. (Wertebereich: $t \in \mathbb{N}; t \geq 0$)

- Treffergenauigkeit - Gibt die Wahrscheinlichkeit an, mit der der Einsatz des Wirkmittels zur erfolgreichen Bekämpfung des Ziels führt (Wertebereich: $0\% \leq X \leq 100\%$; $X \in \mathbb{R}$]).

- Bekämpfungskapazität - Bei der Bekämpfungskapazität handelt es sich um das Gegenstück zum Gefahrenpotential des Ziels. Ein LE kann nur gegen ein Ziel wirken, wenn die Bekämpfungskapazität gleich dem Gefahrenpotential oder höher ist (Wertebereich: [gering/mittel/hoch]).

- Priorisierungsmethode - Dies ist das gleiche Attribut wie das unter 2.2.4 des Agenten Koordinator beschriebene. Es unterscheidet sich nur darin, dass der Koordinator diese Methode für untergeordnete Agenten anwendet. Der Leistungserbringer dagegen verwendet sie für sich selbst, falls er keinen übergeordneten Koordinator besitzt.

- Aktueller Auftrag - In diesem Attribut wird die Anforderung gespeichert, die der Leistungserbringer gerade erfüllt.

- Aktuelle Einsatzzeit - Die Zeit, die sich ein Leistungserbringer nun schon in Einsätzen befindet, ohne seinen Startpunkt wieder aufgesucht zu haben (Wertebereich: $t \in \mathbb{N}$).

Sensoren

- Visuelle Wahrnehmung - Durch diesen Sensor kann der Leistungserbringer Ziele in seiner Umgebung wahrnehmen und identifizieren. Die Reichweite dieses Sensors ist von seinem Attribut Sichtweite abhängig. Wenn durch diesen Sensor ein Ziel entdeckt wird, so wird dieses Ziel im Attribut Geografie als Inhalt einer Koordinate (x,y) gespeichert. Wenn dieser Leistungserbringer dieses Ziel vorher noch nicht identifiziert hat, dann resultiert daraus das Erstellen einer Anforderung. Die Dauer dieses Sensors beträgt 1 ZE.

- Auftragswahrnehmung - Mit diesem Sensor kann der Leistungserbringer einen Auftrag zur Zielmarkierung, zur Wirkungsanalyse oder zur Wirkung erhalten. Wenn ein Auftrag angenommen wird, so wird dieser im Attribut Aktueller Auftrag gespeichert. Hier handelt es sich um einen globalen Sensor, d.h. er ist nicht an den Standort oder Status des Leistungserbringers gebunden, bzw. wird durch diese nicht beeinflusst. Die Dauer des Sensors beträgt 1 ZE.

- Broadcast-Channel lesen - Befähigt den Leistungserbringer den Broadcast-Channel zu empfangen. Dabei kann der Leistungserbringer einen Auftrag annehmen und durch den Sensor Auftragswahrnehmung speichern. Dies benötigt 1 ZE.

- Einsatzzeit ausreichend - Dieser endogene Sensor stellt fest, ob die Einsatzzeit des Leistungserbringers ausreicht, um einen weiteren Auftrag zu erfüllen oder ob er zum Startpunkt zurückkehren muss. Sie reicht nicht aus, wenn der Wert des Attributes Aktuelle Einsatzzeit den Wert des Attributes Einsazzeit übersteigt. Dafür wird 1 ZE benötigt.

- Status abrufen - Durch diesen Sensor nimmt der Leistungserbringer seine Position und seinen Zustand wahr und speichert diesen in seinem Attribut Geografie. Dies benötigt 1 ZE.

Effektoren

- Bewegung - Der Leistungserbringer hat die Fähigkeit, sich von seinem Standort zu einer anderen Position zu bewegen. Er wählt dabei immer den schnellsten Weg (minimale Anzahl Zeiteinheiten). Die Dauer dieses Effektors ergibt sich dabei aus der Zeit, die nötig ist, um das Ziel zu erreichen.

- Anforderung aufgeben - Durch diesen Effektor gibt der Leistungserbringer ein gesichtetes Ziel an den zuständigen Koordinator oder alle Wirkmittel weiter; es wird also ein Ereignis Anforderung von Wirkung erstellt. Dafür wird 1 ZE benötigt.

- Ziel markieren - Markiert ein Ziel, sodass es durch einen anderen Leistungserbringer parallel bekämpft werden kann. Dies benötigt 1 ZE.

- Ziel bekämpfen - Versucht ein Ziel zu bekämpfen. Dieser Effektor ist vom Attribut Wirkreichweite abhängig. Wenn dieser Versuch erfolgreich ist, wird ein Ereignis Bekämpfung eines Zieles erzeugt. Dafür wird 1 ZE benötigt.

- Ziel analysieren - Analysiert durch das Ereignis Analysieren eines Zieles, ob ein Ziel erfolgreich bekämpft wurde. Dadurch aktualisiert der Leistungserbringer die Koordinate (x,y) innerhalb seines Attributes Geografie, welche das Ziel enthält. Das Resultat, also ob das Ziel noch vorhanden ist oder ob die Koordinate wieder leer ist, ist das Ergebnis dieses Effektors und wird an den vorgesetzten Koordinator weitergegeben. Dieser Effektor hat die Dauer 1 ZE.

- Priorisierungswert berechnen - Mit diesem Effektor berechnet der Leistungserbringer seine Eignung für die Erfüllung. Er nutzt dabei sein Attribut Priorisierungsmethode. Dafür benötigt er 1 ZE.
- Broadcast-Channel senden - Befähigt den Leistungserbringer in den Broadcast-Channel Meldungen zu veröffentlichen, indem er ein Ereignis Anforderung von Wirkung erstellt. Dies dauert 1 ZE.
- Eignungsanalyse - Mit diesem endogenen Effektor kann der Leistungserbringer feststellen, ob er grundsätzlich in der Lage ist, ein Ziel zu bekämpfen. Er stellt dabei zunächst seinen Standort über den Sensor Status abrufen fest und prüft anschließend, ob er zum Ziel gelangen kann und ob er in der Lage ist, es zu bekämpfen. Dafür wird 1 ZE benötigt.

Status

Der Status dieses Agenten gibt seine Position, seinen Auftrag, seinen Startpunkt und seine verbleibende Einsatzzeit an.

$Status_{\text{Leistungserbringer}} =$
Position(X, Y), [Kein/Wirkung/Analyse/Markierung],
Startpunkt(X, Y), Einsatzzeit t

Verhalten

In Abbildung 2.13 ist dargestellt, welche Sensoren und Effektoren voneinander abhängig aktiv werden. Jeder Leistungserbringer ist zunächst verfügbar. Durch den Sensor Auftragswahrnehmung kann er einen Auftrag von einem übergeordneten Koordinator erhalten. In diesem Fall ist der Auftrag ein konkreter Befehl, der dem Agenten Leistungserbringer sagt, welche Art der Leistung (Wirkung, Markierung, Wirkungsanalyse) er erbringen muss. Zusätzlich beinhaltet dieser Befehl an welchem Ort sich das Ziel zum Zeitpunkt seiner Entdeckung befand, seine Geschwindigkeit, seine Richtung und den Zeitpunkt, an dem Markierung und Wirkung stattfinden sollen. Wenn dieser Agent keinen vorgesetzten Koordinator besitzt, so empfängt er den Inhalt des Broadcast-Channels über den Sensor Broadcast-Channel lesen. Diese Aufträge enthalten:

- Position des Ziels zum Zeitpunkt der Identifizierung,
- Gefährlichkeit des Ziels
- Bewegungsrichtung des Ziels
- Plattform des Ziels und
- Zeitpunkt der Entdeckung.

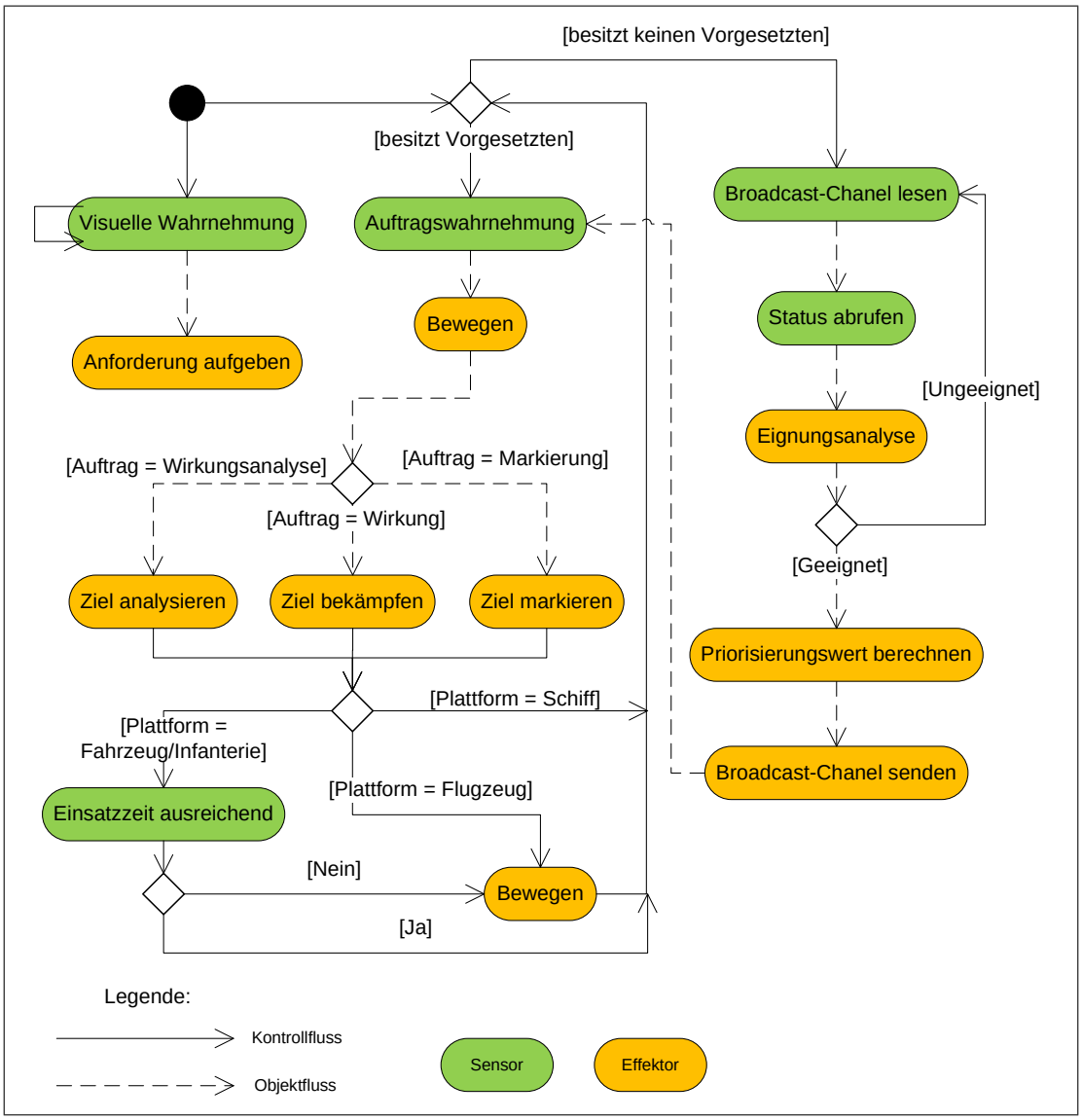

Abbildung 2.13: Diese Abbildung zeigt, welche Sensoren und Effektoren voneinander abhängig sind und nacheinander aktiv werden. Zu beachten ist, dass der Sensor Visuelle Wahrnehmung ständig aktiv ist.

Nach Empfang dieser Anforderung muss der Leistungserbringer zunächst feststellen, ob er grundsätzlich in der Lage ist, ein Ziel zu bekämpfen. Dafür bezieht er zunächst die Daten des Sensors Status abrufen. Nun überprüft er mit dem Effektor Eignungsanalyse, ob er dieses Ziel bekämpfen könnte. Dabei wird zunächst kontrolliert, ob das Attribut Bekämpfungskapazität mindestens genauso hoch ist wie die Gefährlichkeit des Ziels. Ist dies der Fall, wird kontrolliert, ob die Einsatzzeit des Leistungserbringers noch ausreicht, um das Ziel zu bekämpfen und anschließend zum Startpunkt zurückzukehren. Wenn dies der Fall ist, dann ist der Leistungserbringer grundsätzlich in der Lage, das Ziel zu bekämpfen. Dann berechnet er gemäß dem Attribut Priorisierungsmethode mit dem Effektor Priorisierungswert berechnen einen Wert und veröffentlicht diesen und zusätzlich den Wert des Attributes Markierungsfähigkeit mithilfe des Effektors Broadcast-Channel senden in den Broadcast-Channel. Wenn der Agent nun mit dem Sensor Broadcast-Channel lesen feststellt, dass er den höchsten Wert hatte, so erteilt er sich selbst den Auftrag zur Leistung. Sollte er gemäß seinem Attribut Markiernotwendigkeit immer auf Markierung angewiesen sein, so erteilt er sich den Auftrag nur, wenn es einen Leistungserbringer mit Markierungsfähigkeit gibt, der einen besseren Wert hat als er selbst. Wenn er eine Anforderung über den Broadcast-Channel empfangen hat, so kann diese abhängig von den Attributen des Agenten, der die Anforderung erstellt, aus den Teilaufträgen Markierung, Wirkung und Wirkungsanalyse bestehen. Im Rahmen der Selbstkoordination erteilt sich der Leistungserbringer zunächst den Auftrag zur Wirkung. Sollte er gemäß seinem Attribut Markiernotwendigkeit immer auf Markierung angewiesen sein, so fügt er den Teilauftrag Markierung hinzu, zusammen mit dem Zeitpunkt, an dem der Leistungserbringer am Zielort sein kann, bzw. ändert ihn entsprechend, falls er schon vorhanden ist. Diese übrigen Teilaufträge gibt er wieder als neue Anforderung auf. Durch die Auswahl des Wirkungsauftrages durch den am zweitbesten geeigneten Leistungserbringer wird sichergestellt, dass der Markierer immer vor dem Wirkmittel am Zielort sein kann. Falls der Leistungserbringer einen übergeordneten Koordinator besitzt, so übernimmt dieser für ihn die Koordination. In beiden Fällen ist der Agent Leistungserbringer irgendwann in dem Zustand, dass er einen Auftrag auf Wirkung, Markierung oder Wirkungsanalyse und die dazugehörigen Daten besitzt. Mithilfe des Effektors Bewegung verändert der Agent nun seine Position, bis sich das Ziel in seiner Wirkreichweite, für den Auftrag Wirkung, oder in seiner Sichtweite, für die Aufträge Markierung oder Wirkungsanalyse befindet. Sobald er diese Position erreicht hat, nutzt er die Effektoren Ziel bekämpfen, Ziel markieren oder Ziel analysieren, um seinen Auftrag auszuführen. Das weitere Verhalten ist abhängig vom Attribut Plattform: fliegende Einheiten bewegen sich nun zurück zum Ausgangspunkt, seegehende Einheiten verbleiben an ihrer Position (Versorgung in See durch Replenishment at Sea - Manöver). Handelt es sich jedoch um eine Bodeneinheit, so muss der Agent prüfen, ob die verbleibende Einsatzzeit noch ausreicht. Dafür wird zunächst die benötigte Zeit für diesen nun abgeschlossenen Einsatz berechnet und auf den Wert des Attributes Aktuelle Einsatzzeit addiert. Ist nun dieser Wert kleiner als die Einsatzzeit, ist der Leistungserbringer sofort wieder verfügbar. Andernfalls kehrt er zum Startpunkt zurück. Am Startpunkt muss der Leistungserbringer für eine Zeit entsprechend dem Attribut Ruhezeit warten, danach ist er wieder verfügbar und setzt das Attribut Aktuelle Einsatzzeit gleich null. Während der gesamten Simulation nimmt jeder

Agent vom Typ Leistungserbringer seine Umwelt durch den Sensor Visuelle Wahrnehmung wahr. Entdeckt er dabei ein Ziel ähnlich wie ein Agent vom Typ Aufklärer, verhält er sich auch wie dieser, mit dem Unterschied, dass der Leistungserbringer seinen bisherigen Auftrag dabei keinesfalls abbricht.

Formale Beschreibung von Eingabewerten

- Bewegungsgeschwindigkeit - $[v]$ = Koordinaten/ZE; $v \in \mathbb{N}$; $v \geq 0$
- Startpunkt - Koordinatentupel [X, Y]
- Plattform - [Schiff/Flugzeug/Fahrzeug/Infanterie]
- Markierungsfähigkeit - $[Ja/Nein]$
- Koordinationsknoten - $(\mathrm{k}_1,\ldots,\mathrm{k}_\mathrm{i})$; $i \in \mathbb{N}$
- Markiernotwendigkeit - $[Ja/Nein]$
- Wirkreichweite - $[r]$ = Koordinaten; $r \in \mathbb{N}$; $r \geq 0$
- Einsatzzeit - $[t_\mathrm{e}]$ = ZE; $t_\mathrm{e} \in \mathbb{N}$; $t_\mathrm{e} \geq 0$
- Ruhezeit - $[t_\mathrm{r}]$ = ZE; $t_\mathrm{r} \in \mathbb{N}$; $t_\mathrm{r} \geq 0$
- Treffergenauigkeit - $[p]$ = %; $0 \leq p \leq 100$; $p \in \mathbb{R}]$
- Bekämpfungskapazität - [gering/mittel/hoch]
- Priorisierungsmethode - [Entfernung/Gefahrenpotential/Kombination]

Beschreibung von Interaktionen zwischen Submodellen

Ein Zustandswechsel erfolgt durch die Agenten des Typs Aufklärer, Leistungserbinger oder Koordinator (dezentraler Fall) oder nur durch den Koordinator (zentraler Fall). Wenn dieser Agent ein Objekt vom Typ Ziel entdeckt, so wird durch die erstellte Anforderung ebenfalls ein Zustandswechsel ausgelöst. Der Leistungserbringer kann, falls er einen Vorgesetzten hat (zentraler Fall) drei verschiedene Aufträge erhalten:

- Auftrag zur Markierung - Wenn ein anderer Leistungserbringer Markierung benötigt, so erhält ein weiterer Leistungserbringer einen Auftrag, falls der Anforderer nicht selber markieren kann. Dieser Auftrag beinhaltet neben der Position zum Zeitpunkt der Identifizierung und der Bewegungsgeschwindigkeit auch den Zeitpunkt, an dem das Wirkmittel das Ziel in Wirkreichweite hat. An diesem Zeitpunkt muss dann auch die Markierung erfolgen, um das Ziel schnellstmöglich zu bekämpfen.
- Auftrag zur Wirkung - Diese Art von Auftrag beinhaltet Position zum Zeitpunkt der Identifizierung, Bewegungsgeschwindigkeit und Gefahrenpotential des Ziels. Es ist kein Zeitpunkt zum Wirken notwendig, da der Markierer sich bereits vor Ort befindet, denn es wird immer der schnellere LE als Markierer ausgewählt.

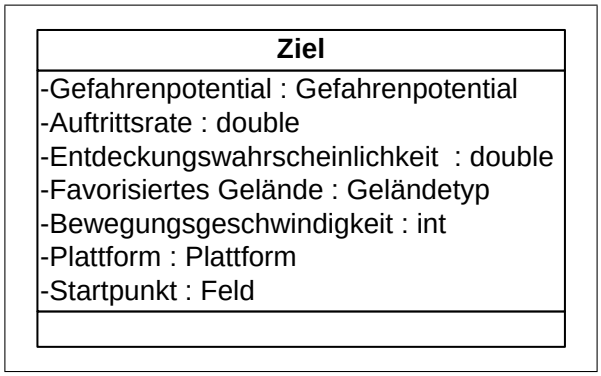

Abbildung 2.14: Das Klassendiagramm des Objektes Ziel

- Auftrag zur Wirkungsanalyse - Falls der Anforderer das Ziel nicht mehr im Sichtfeld hat, kein Markierer notwendig ist und die Wirkreichweite des Leistungserbringers seine Sichtweite überschreitet, so ist ein zweiter Leistungserbringer zur Wirkungsanalyse einzuteilen. Dieser erhält die Position des Ziels zum Zeitpunkt der Identifizierung und die Bewegungsgeschwindigkeit. Da er von den zwei LE, die mit dem Ziel beauftragt werden, immer der langsamere ist, ist kein Zeitpunkt zur Analyse notwendig, da das Ziel bereits bekämpft ist.

Hat der Leistungserbringer keinen Vorgesetzten (dezentraler Fall), erhält er diese Aufträge aus dem Broadcast-Channel, den er sich mit gleichgesetzten Knoten (Koordinatoren, Aufklärer und Leistungserbringer) teilt. Er kann dann ebenfalls Aufträge in den Broadcast-Channel abgeben.

2.2.6 Objekt Ziel

Beschreibung von Struktur und internem Verhalten

Attribute

- Gefahrenpotential - Die Gefährlichkeit des Zieles. Diese ist nötig für die Auswahl des passenden Wirkmittels und wird durch den Anforderer festgestellt (Wertebereich: [gering/mittel/hoch]).
- Auftrittsrate - Der durchschnittliche Abstand zwischen dem Auftauchen zweier Ziele dieser Art (Wertebereich: $t \in \mathbb{N}$; $t \geq 0$). Die angegebene Auftrittsrate dient dabei nur als Erwartungswert einer Normalverteilung mit einer Standardabweichung von 20% der Auftrittsrate.
- Favorisiertes Gelände - Hier kann für Objekte der Plattform Fahrzeug und Infanterie ein Geländemuster gewählt werden. Auf diesem Geländetyp wird ein Ziel, fünf mal wahrscheinlicher auftreten als auf einem anderen Geländemuster (Wertebereich: [Ebene/Binnengewässer/Stadt/Wald/Gebirge]).

- Entdeckungswahrscheinlichkeit - Gibt die Wahrscheinlichkeit an, mit der ein Ziel durch eine Einheit entdeckt wird. Dieser Wert wird einmalig für jede Einheit betrachtet, in dessen Sichtfeld das Ziel kommt (Wertebereich: $0\% \leq X \leq 100\%$; $X \in \mathbb{R}$]).
- Bewegungsgeschwindigkeit - in [Koordinaten/ZE]. Dies ist die Geschwindigkeit, mit der sich das Objekt bewegen kann. Es kann nicht wählen, ob es schneller oder langsamer fahren will. Die endgültige Geschwindigkeit ist jedoch auch von den Bewegungseinschränkungen abhängig, die sich durch den Untergrund ergeben (siehe Umwelt).
- Startpunkt - Die Koordinate, auf der das Objekt bei Simulationsstart platziert wird (Koordinatentupel [X, Y]).
- Plattform - das Medium, dem das Objekt angehört (Wertebereich: [Schiff/Flugzeug/ Fahrzeug/Infanterie]).

Status

Der Status dieses Objektes gibt seine Position zum Zeitpunkt des Auftretens, seine Richtung, seine Geschwindigkeit, sein Gefahrenpotential und den Zeitpunkt seines Auftretens an.

$Status_{\text{Ziel}} =$
Position (X, Y), [Nord/Ost/Süd/West], Geschwindigkeit v,
[niedrig/mittel/hoch], Zeitpunkt t_0

Verhalten

Dieses Objekt wird während der Simulation an einem beliebigen Zeitpunkt erzeugt. Die Regelmäßigkeit des Auftretens hängt von dem Attribut Auftrittsrate ab. Die Position innerhalb der Karte ist zufällig, sie ist jedoch in den Gebieten, die mit dem Attribut Favorisiertes Gelände identisch sind, fünfmal wahrscheinlicher wie in anderen Gebieten. Sobald dieses Objekt erzeugt wurde, bewegt es sich in eine zufällige Richtung, falls seine Geschwindigkeit größer null ist. Diese wird eingehalten bis die detaillierte Karte verlassen wurde. In diesem Fall wird das Ziel von der Simulation aus der Welt entfernt. Während es sich innerhalb der Karte bewegt, kann es durch einen Agenten vom Typ Aufklärer oder Leistungserbringer entdeckt werden. Dies ist abhängig von dem Wert des Attributs Entdeckungswahrscheinlichkeit. Die Agenten können dann seinen Status feststellen. Wenn das Objekt Ziel dann erfolgreich bekämpft wurde, so wird es aus der Simulation entfernt.

Formale Beschreibung von Eingabewerten

- Bewegungsgeschwindigkeit - $[v] =$ Koordinaten/ZE; $v \in \mathbb{N}$; $v \geq 0$
- Startpunkt - Koordinatentupel [X, Y]

- Plattform - [Schiff/Fahrzeug/Infanterie] (keine Luftziele!)
- Gefahrenpotential - [gering/mittel/hoch]
- Auftrittsrate - $[p_\mathrm{r}] = \%;\ 0 \leq p_\mathrm{r} \leq 100;\ p_\mathrm{r} \in \mathbb{R}]$
- Favorisiertes Gelände - [Ebene/Binnengewässer/Stadt/Wald/Gebirge]
- Entdeckungswahrscheinlichkeit - $[p_\mathrm{i}] = \%;\ 0 \leq p_\mathrm{i} \leq 100;\ p_\mathrm{i} \in \mathbb{R}]$

Beschreibung von Interaktionen zwischen Submodellen

Ein Zustandswechsel erfolgt durch den Agenten Leistungserbringer oder durch dieses Objekt selbst.

2.3 Vergleich mit konzeptuellem Modell

Das konzeptuelle Modell konnte sehr gut in das formale Modell transferiert werden. Es waren keine besonderen Änderungen oder Einschränkungen notwendig. Das Modell wurde lediglich um die Forderungen des GRAMS-Referenzmodell erweitert, zum Beispiel die Definition von Events.

3 Ausführbares Modell

Dieses Kapitel beschreibt die konkrete Implementierung des Modells zum Szenario taktische Feuerunterstützung. Es wird eine Übersicht über die erstellten Pakete und Klassen, sowie eine genauere Beschreibung ausgewählter Methoden gegeben.

3.1 Simulationsumgebung

Die Implementierung geschah in Java, das bedeutet, dass es plattformunabhängig ist und lediglich eine lauffähige Java Runtime Environment benötigt. Bei der Programmierung wurde die Entwicklungsumgebung Eclipse genutzt. Das Modell wurde unter Nutzung des GRAMS-Referenzmodells erstellt. Die Entwicklung eines Simulators war nicht notwendig, denn es werden die Simulatoren des Referenzmodells genutzt.

3.2 Struktur und Verhaltensbeschreibung

Beschreibung der Herkunft der Pakete

Alle Elemente dieses Modells wurden im Rahmen dieser Bachelorarbeit erstellt, nutzen dabei aber Interfaces und abstrakte Klassen des GRAMS-Referenzmodells um die Lauffähigkeit auf dessen Simulatoren zu gewährleisten.

3.2.1 Identifikation der Pakete mit dem Kapitel des entsprechenden FM

Die Bestandteile des formalen Modells wurden unterschiedlich in den Paketen des ausführbaren Modells umgesetzt. Zum einen fiel die Zeit weg, da diese durch das Referenzmodell schon bereitgestellt wurde. Andere Elemente wurden wiederum in der Umwelt zusammengefasst. Wie die einzelnen Abschnitte des formalen Modells konkret umgesetzt wurden, zeigt folgende Tabelle.

Kapitel des formalen Modells	Realisierung im ausführbaren Modell
Submodell Zeit (Kap.2.2.2)	Eine Definition der Zeit entfällt, da diese Bestandteil der Referenzimplementierung ist.
Submodell Umwelt (Kap.2.2.2)	Paket 3.2.2 Umwelt
Submodell Objekte (Kap.2.2.2)	Die Abstrakte Klasse Unit wird in der Umwelt abgebildet, ansonsten erfolgt die genaue Definition der Objekte in ihren entsprechenden Paketen.
Submodell Ereignisse (Kap.2.2.2)	Alle Ereignisse werden ebenfalls in der Umwelt aufgeführt und erläutert.
Submodell Rahmenbedingungen (Kap.2.2.2)	Die Rahmenbedingungen sind auch Bestandteil des Paketes Umwelt
Agent Aufklärer (Kap.2.2.3)	Paket 3.2.3 Aufklärer
Agent Koordinator (Kap.2.2.4)	Paket 3.2.4 Koordinator
Agent Leistungserbringer (Kap.2.2.5)	Paket 3.2.5 Leistungserbringer
Objekt Ziel (Kap.2.2.6)	Paket 3.2.6 Ziel

Insgesamt wurden die implementierten Klassen auf sieben Pakete aufgeteilt. Diese sind in Abbildung 3.1 abgebildet. Zwischen den Paketen existieren eine Vielzahl von Beziehungen. So befinden sich zum Beispiel die Actions, die von den Agenten genutzt werden, im Paket des entsprechenden Agenten, die dazugehörigen Events befinden sich jedoch in einem anderen Paket, da sie von verschiedenen Paketen genutzt werden.

3.2.2 Umwelt

Struktur und internes Verhalten

In Abb. 3.2 werden sieben Kernklassen und Enumerationstypen der Umwelt dargestellt:

- Die Umweltklasse JFSEnvironment als solches, in der die Karte, die verschiedenen Zieltypen und eine globale Priorisierungsmethode gespeichert werden. Diese Priorisierungsmethode findet Anwendung, wenn kein Koordinator vorhanden ist und die Leistungserbringer selber priorisieren müssen. Weiterhin bietet sie Methoden, die mit der Karte in Verbindung stehen. So liefert getValidFieldsInRange() alle Felder, die innerhalb eines bestimmten Radius um ein Feld liegen. Durch createRandomTarget() wird ein Ziel an einer zufälligen Position erzeugt, dabei wird jedoch bevorzugt dessen favorisiertes Gelände genutzt. Weiterhin implementiert diese Klasse die durch das Interface vorgegebenen Methoden findCoordinate() und isValidCoordinate(), die zum Auffinden von Agenten und Objekten dienen.

- Die gesamte Karte ist schachbrettartig aus vielen Feldern der Klasse Field aufgebaut. Jedes dieser Felder zeichnet sich durch ein bestimmtes Landschaftsmuster des Enums Landscape aus, welche für unterschiedliche Bewegungseinschränkungen sorgen. Auf einem Feld können sich mehrere Agenten oder Ziele aufhalten.

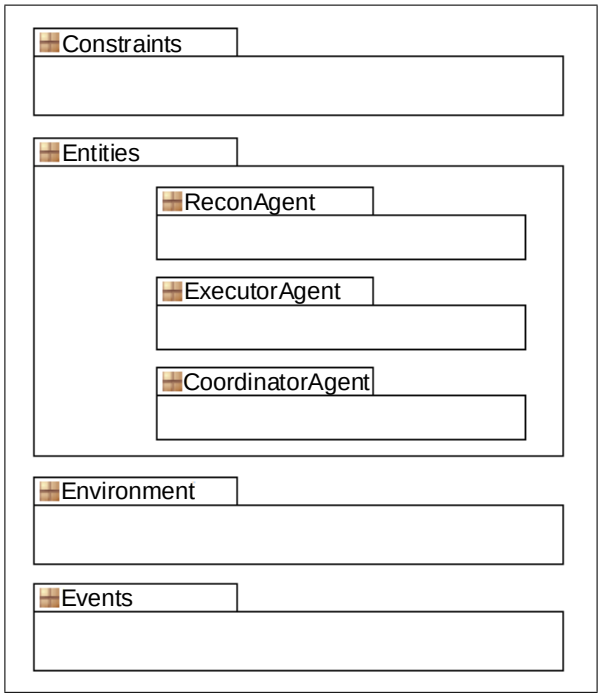

Abbildung 3.1: Übersicht über alle Pakete des implementierten Modells. Die Bezeichnung und Gruppierung orientiert sich am GRAMS.

- Das Enum Priorizitation bietet die notwendigen Priorisierungsmethoden für dieses Modell. Es sind drei Priorisierungsmethoden implementiert: Mit TIME wird der Leistungserbringer ausgewählt, der am schnellsten das Ziel bekämpfen kann und mit CAPABILITY der Leistungserbringer, der aufgrund seiner Bekämpfungsfähigkeit am besten geeignet ist, um das Ziel zu bekämpfen. Besitzt das Ziel also ein niedriges Bedrohungspotential, so wird versucht, erst einen Leistungserbringer mit ebenfalls niedriger, oder falls nicht vorhanden, mit mittlerer Bekämpfungskapazität auszuwählen. Dies hat den Vorteil, dass Leistungserbringer mit besonders hoher Bekämpfungskapazität für gefährlichere Ziele zur Verfügung stehen, sollten diese auftauchen. Die dritte Priorisierungsmethode HYBRID ist, wie der Name schon sagt, eine Mischung der beiden ersten Methoden. Siehe dazu Abbildung 2.10 im formalen Modell. Es können hier jederzeit neue Priorisierungsmethoden ergänzt werden, da die Lauffähigkeit durch mehrere abstrakte Methoden, die implementiert werden müssen, gewährleistet wird.

- Die eben erwähnte Kapazität wird durch das Enum Capability abgedeckt. Es existiert hier bisher LOW, MEDIUM und HIGH und eine abstrake Methode isEqualOrGreater(), mit deren Hilfe zwei Werte dieses Enums verglichen werden können.

- Die eigenen Einheiten und die auftauchenden Ziele können verschiedenen Plattformen angehören. Dieser Wert des Enums Platform ist wichtig für die Art der

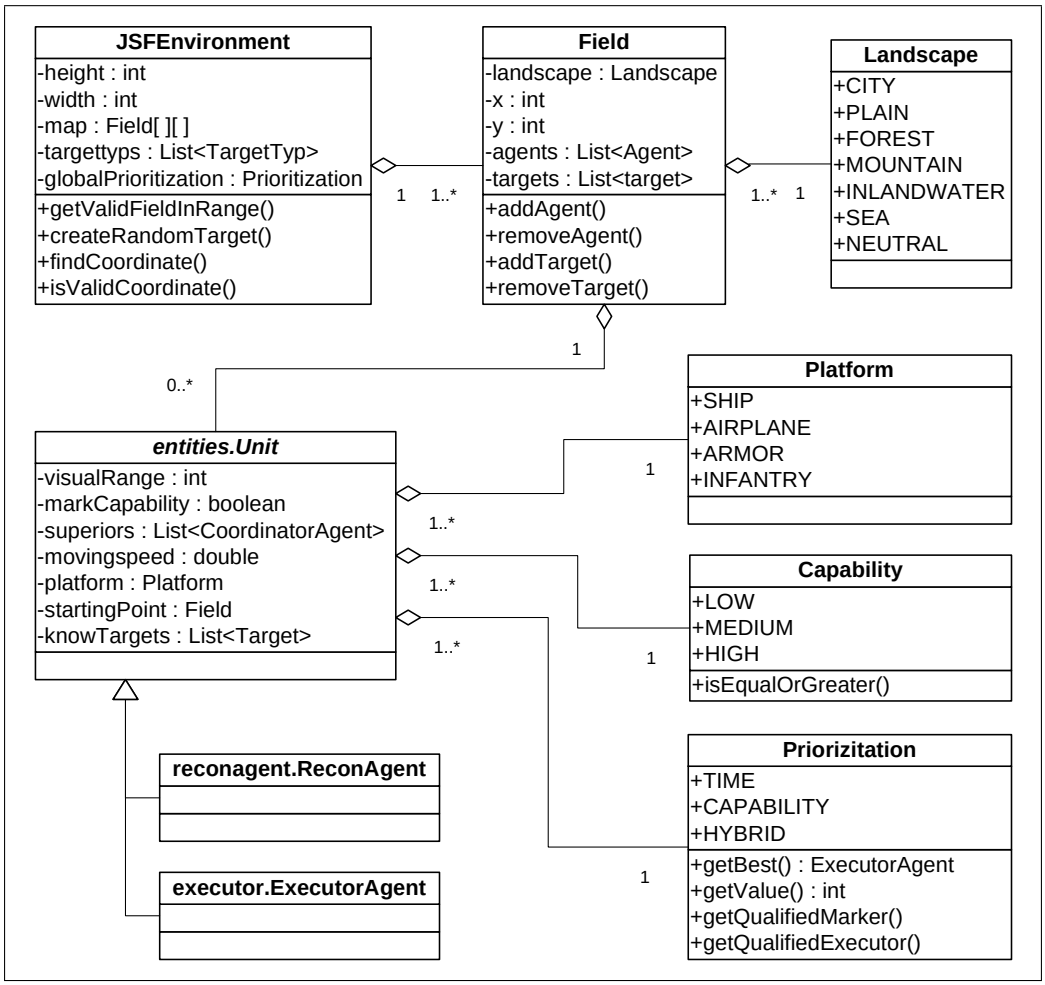

Abbildung 3.2: Eine Übersicht über relevante Klassen des Paketes Environment

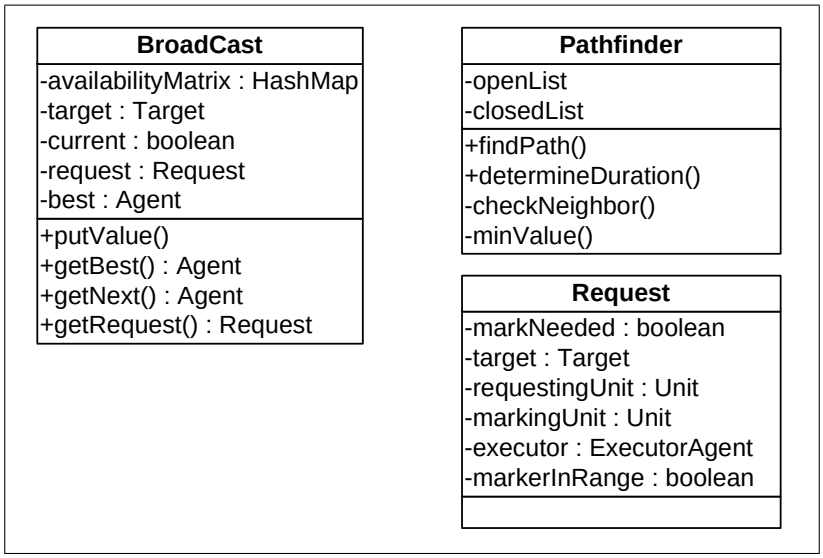

Abbildung 3.3: Eine Übersicht über relevante Klassen des Paketes Environment

Bewegungseinschränkung, der der Agent oder das Ziel unterliegt.

- Die abstrakte Klasse Unit ist die Superklasse des Aufklärers und des Leistungserbringers. Sie vereint im Sinne der Vererbung gemeinsame Attribute der Agenten.

In Abb 3.3 werden drei weitere wichtige Klassen dargestellt.

- Eine besonders wichtige Rolle innerhalb der Umwelt nimmt die Klasse Pathfinder ein. Sie wird von allen Aufklärern und Leistungserbringern genutzt, um den optimalen Weg zwischen zwei Feldern der Karte zu finden. Dabei wird beim Aufklärer immer der direkte Weg gewählt (sein Patrouillenweg) und bei Leistungserbringern immer der schnellste (Weg zum Ziel oder Startpunkt). Bei der Ermittlung des optimalen Pfades wird der A-Stern Algorithmus [6] verwendet, da er immer den optimalen Pfad findet, falls einer existiert. Bei der Implementierung wurde jedoch auf eine vorherige Schätzfunktion verzichtet, da aufgrund der verschiedenen Landschaftsmuster sich der schnellste Pfad wesentlich vom direkten Weg unterscheiden kann. Ein eventueller Laufzeitgewinn ist hier also fragwürdig und auch nicht Ziel der Arbeit.

- Die Klasse Request bildet eine Anforderung einer Einheit ab. In ihr werden alle benötigten Daten über die beteiligten Agenten gesammelt und auch an diese weitergerecht, sobald sie sie benötigen.

- Sollte ein Ziel nicht durch einen verantwortlichen Koordinator bekämpft werden können oder falls es gar keinen Verantwortlichen gibt, so wird die Anforderung dezentral an alle ungebundenen Agenten aufgegeben, die in der Lage sind, das Ziel zu bekämpfen. Dazu dient die Klasse BroadCast.

Bestandteil der Umwelt sind auch alle Ereignisse, die innerhalb des Modells auftreten können. Es werden dabei vier verschiedene Arten unterschieden [7], die hier aufgelistet und kurz beschrieben werden. Auffällig ist hier, dass es kein Ereignis gibt, dass von einem Objekt ausgelöst wird und direkt ein anderes beeinflusst. Das liegt daran, dass immer der Umweg über die Umwelt gegangen werden muss. Deshalb ist es oft der Fall das solche Events doppelt aufgelistet sind.

i. Ereignisse, die in der Umwelt entstehen und nur diese direkt beeinflussen:

- *BroadCastControllEvent* - Nach einer gewissen Zeit wird kontrolliert, ob der diesem Event zugeordnete BroadCast erfüllt wurde. Wenn das nicht der Fall ist, wird er neu ausgelöst.
- *TargetCreateEvent* - Durch dieses Ereignis wird ein Ziel auf der Karte erzeugt.
- *TargetDeleteEvent* - Wenn ein Ziel nicht innerhalb einer bestimmten Zeit bekämpft wurde, so wird es durch dieses Ereignis wieder entfernt.

ii. Ereignisse, die aus der Umwelt kommen und direkt Objekte beeinflussen:

- *BroadCastEvent* - Lässt den Agenten überprüfen, wie sehr er für die Erfüllung der Anforderung geeignet ist und anschließend auf diesen BroadCast antworten.
- *EffectEvent* - Entfernt ein Ziel von der Karte.
- *MarkingEvent* - Markiert ein Ziel, damit es bekämpft werden kann.
- *OrderEvent* - Dieses veranlasst einen oder mehrere Leistungserbringer zur Erfüllung eines Auftrages (Markierung oder Wirkung).
- *RequestEvent* - Durch dieses Ereignis wird der Koordinator eine neue Anforderung annehmen.

iii. Ereignisse, die von Objekten ausgelöst werden und nur das gleiche Objekt beeinflussen:

- *BroadCastEvent* - Sobald der Agent seinen Wert an den BroadCast gegeben hat, überprüft er erneut, ob er jetzt der am besten Geeignete ist.
- *DetectionEvent* - Wenn eine Einheit ein Ziel entdeckt, wird es mit diesem Ereignis seine Weitermeldung auslösen.
- *MoveEvent* - Ein Leistungserbringer bewegt sich nach diesem Ereignis weiter auf seinem Pfad zum Ziel.
- *PatrolEvent* - Ein Aufklärer bewegt sich nach diesem Ereignis weiter auf seinem Patrouillenpfad zum nächsten Wegpunkt.

- *OperatingTimeCheckEvent* - Diesen Ereignis veranlasst den Leistungserbringer, seine Einsatzzeit zu kontrollieren.
- *PerceiveEvent* - Nach diesem Ereignis wird die Einheit ihr Sichtfeld auf Ziele kontrollieren.
- *PlanEvent* - Anschließend wird der Koordinator oder Leistungserbringer seine weiteren Schritte planen. Der Koordinator wird eventuell eine neue Anforderung bearbeiten und der Leistungserbringer wird sich weiter auf sein Ziel zubewegen, es markieren, bekämpfen oder warten.
- *RequestProcessEvent* - Dieses Ereignis löst die Bearbeitung einer offenen Anforderung durch den Koordinator aus.
- *ReturnEvent* - Mit diesem Ereignis löst ein Leistungserbringer seine Rückkehr zum Startpunkt aus.
- *WaitEndEvent* - Nach dem Ablauf der Wartezeit macht dieses Ereignis den Leistungserbringer wieder verfügbar für neue Aufträge.

iv. Ereignisse, die von von Objekten ausgelöst werden und die Umwelt beeinflussen:

- *BroadCastInitEvent* - Wenn die zentrale Bekämpfung eines Ziels nicht möglich ist, erstellt ein Agent dieses Ereignis, welches einen BroadCast generiert, der dann bei jedem möglichen Empfänger ein BroadCastEvent auslöst.
- *EffectEvent* - Ein Leistungserbringer wirkt mit diesem Ereignis gegen ein Ziel.
- *MarkingEvent* - Eine Einheit markiert mit diesem Ereignis ein Ziel, sodass es bekämpft werden kann.
- *OrderEvent* - Dieses Ereignis wird von einem Koordinator oder Leistungserbringer ausgelöst und veranlasst einen oder mehrere Leistungserbringer zur Erfüllung eines Auftrages (Markierung oder Wirkung).
- *RequestEvent* - Mit diesem Ereignis gibt eine Einheit eine Anforderung von Wirkung auf ein bestimmtes Ziel auf.

Weiterhin findet in diesem Paket die Definition der Einschränkungen, welche zu jedem Zeitpunkt der Simulation gelten, statt. Die Anzahl dieser ist im Vergleich zum formalen Modell viel niedriger, da sich viele Einschränkungen durch die Art der Programmierung oder dem Erstellen der Konfigurationsdatei ergeben. Weiterhin gilt jedoch folgende Einschränkung:

- *MarkerDifferentFromEffectorConstraint* - Durch diese Einschränkung wird sichergestellt, dass der Markierer einer Anforderung niemals der gleiche ist, wie das Wirkungsmittel. Ansonsten wird die Simulation abgebrochen.

Eine wichtige Einschränkung im formalen Modell ist die, dass auf einem Feld, auf dem sich bereits ein Aufklärer oder Leistungserbringer befindet, kein Ziel erzeugt werden kann. Diese Einschränkung lässt sich nicht unter Nutzung des Constraint-Interfaces umsetzen, da Einschränkungen nach dem Referenzmodell immer an Aktionen gebunden sind. Das erzeugen von Zielen ist jedoch ein Umweltereignis. Die Einschränkung wurde daher ohne Nutzung des Constraint-Interfaces umgesetzt und beim Erzeugen von Zielen beachtet.

Interaktion mit anderen Paketen

Die Umwelt ist sehr stark verflochten mit allen anderen Agenten. Jeder Agent, der sich bewegen kann, bewegt sich auf der Karte und nutzt dabei die Klasse Pathfinder. Auf dieser Karte erscheinen Ziele, die anschließend von Agenten identifiziert und durch eine Instanz der Request-Klasse weitergemeldet werden. Als Folge davon nutzen Koordinatoren oder auch Leistungserbringer die Klasse Priorizitation, um die Anforderung bestmöglich zu erfüllen.

3.2.3 Agent Aufklärer

Struktur und internes Verhalten

In Abbildung 3.4 werden die fünf für den Aufklärer relevanten Klassen abgebildet.

- Der Aufklärer selbst wird durch die Klasse ReconAgent abgebildet, welche von der abstrakten Klasse Unit viele wesentliche Attribute erbt.
- Solange der Aufklärer kein Ziel entdeckt hat, patrouilliert er mit der PatrolAction auf seinem vorbestimmten Pfad und kontrolliert ständig seine Umwelt mit Perceive-Action.
- Wenn er dabei ein Ziel entdeckt, so meldet er dies mit der ReportAction. Je nachdem, wie die weitere Verarbeitung dieser Anforderung sich entwickelt, kann es sein, dass der Aufklärer der Markierer des Zieles wird. Wenn das der Fall ist, dann markiert er das Ziel zum richtigen Zeitpunkt durch die MarkReconAction. In jedem Fall setzt er anschließend seine normale Patrouillentätigkeit fort.

Interaktion mit anderen Submodellen

Der Agent erhält den Auftrag zur Zielmarkierung von seinem vorgesetzten Koordinator. Dabei erhält er gleichzeitig den richtigen Zeitpunkt für diese Aktion. Ansonsten führt lediglich das Objekt Ziel durch seine Entdeckung einen Zustandswechsel herbei.

3.2.4 Agent Koordinator

Struktur und internes Verhalten

Die für den Koordinator wesentlichen Klassen sind in Abbildung 3.5 dargestellt.

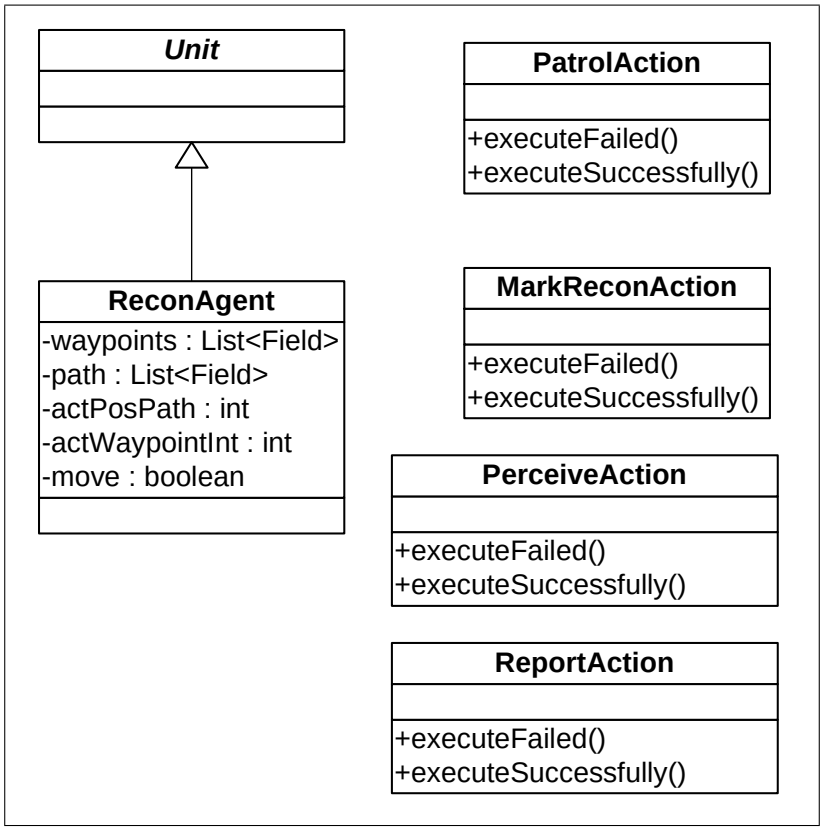

Abbildung 3.4: Eine Übersicht über alle relevanten Klassen des Paketes ReconAgent

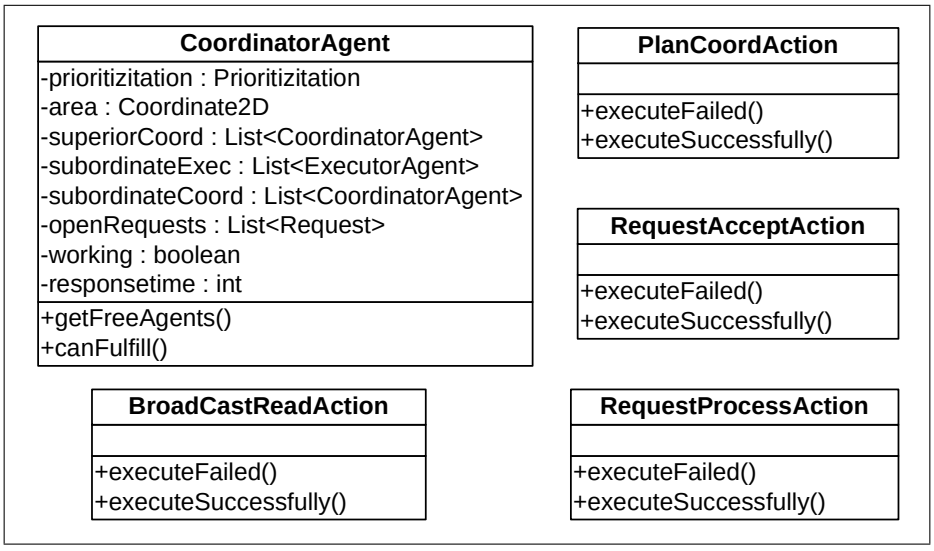

Abbildung 3.5: Eine Übersicht über relevante Klassen des Paketes CoordinatorAgent

- Der Koordinator wird durch die Klasse CoordinatorAgent abgebildet. Diese Klasse enthält alle für den Koordinator wesentlichen Attribute. Durch die Methode canFulfill() kann kontrolliert werden, ob ein Koordinator überhaupt in der Lage ist, eine Anforderung zu erfüllen. Sollte das nicht der Fall sein und sollte er auch keine weiteren Vorgesetzten haben, so kann er mithilfe der Methode getFreeAgents() alle ungebundenen Agenten ermitteln, an die er die Anforderung abtreten könnte.

- Ein Koordinator kontrolliert zu jedem Zeitpunkt der Simulation durch die PlanCoordAction, ob für ihn eine offene Anforderung vorliegt. Weiterhin kann er durch die RequestAcceptAction und die BroadCastReadAction Anforderungen annehmen.

- Durch die RequestProcessAction versucht ein Koordinator eine Anforderung zu erfüllen. Wenn der Anforderung durch einen anderen Agenten schon ein Leistungserbringer als Wirkmittel zugewiesen wurde, so sucht dieser Koordinator noch einen passenden Markierer aus seinen Untergebenen aus. Wenn die Anforderung jedoch noch komplett unbearbeitet ist, so sucht er zunächst das passende Wirkmittel aus der Menge seiner Untergebenen aus. Anschließend prüft er, ob dieses Wirkmittel einen zusätzlichen Markierer benötigt und ermittelt diesen falls nötig. Wenn der Koordinator nun zwei Leistungserbringer ausgewählt hat, so übermittelt er diesen den passenden Befehl (OrderEvent).

Interaktion mit anderen Submodellen

Eine Zustandsänderung kann zum einen durch einen Aufklärer oder Leistungserbringer erfolgen, wenn diese eine Anforderung aufgeben, die der Koordinator durch die RequestAcceptAction annimmt. Zusätzlich kann jedoch auch durch die BroadCastReadAction eine Zuständsänderung durch jeden Agenten der Simulation eintreten.

3.2.5 Agent Leistungserbringer

Struktur und internes Verhalten

In Abbildung 3.6 sind alle für den Leistungserbringer wichtigen Klassen aufgezeigt.

- Der Leistungserbringer selbst wird durch die Klasse ExecutorAgent abgebildet, welche von der abstrakten Klasse Unit viele wesentliche Attribute erbt.

- Zu jedem Zeitpunkt der Simulation überprüft der Leistungserbringer durch die PerceiveAction, ob sich innerhalb seines Sichtfeldes Gegner befinden. Ist dies der Fall, so meldet er sie durch die ReportAction an den zuständigen Koordinator weiter, bzw. löst ein BroadCastInitEvent aus, wenn es keinen Koordinator gibt.

- Wenn der Leistungserbringer einen Auftrag annimmt, so wird in der PlanExecAction zunächst der Pfad zu dem Ziel ermittelt. Auf diesem Pfad bewegt der Agent sich dann durch die MoveToTargetAction entlang, bis sich das Ziel in seiner Sichtweite, falls er der Markierer ist, oder in seiner Wirkreichweite, wenn er das Wirkmittel ist,

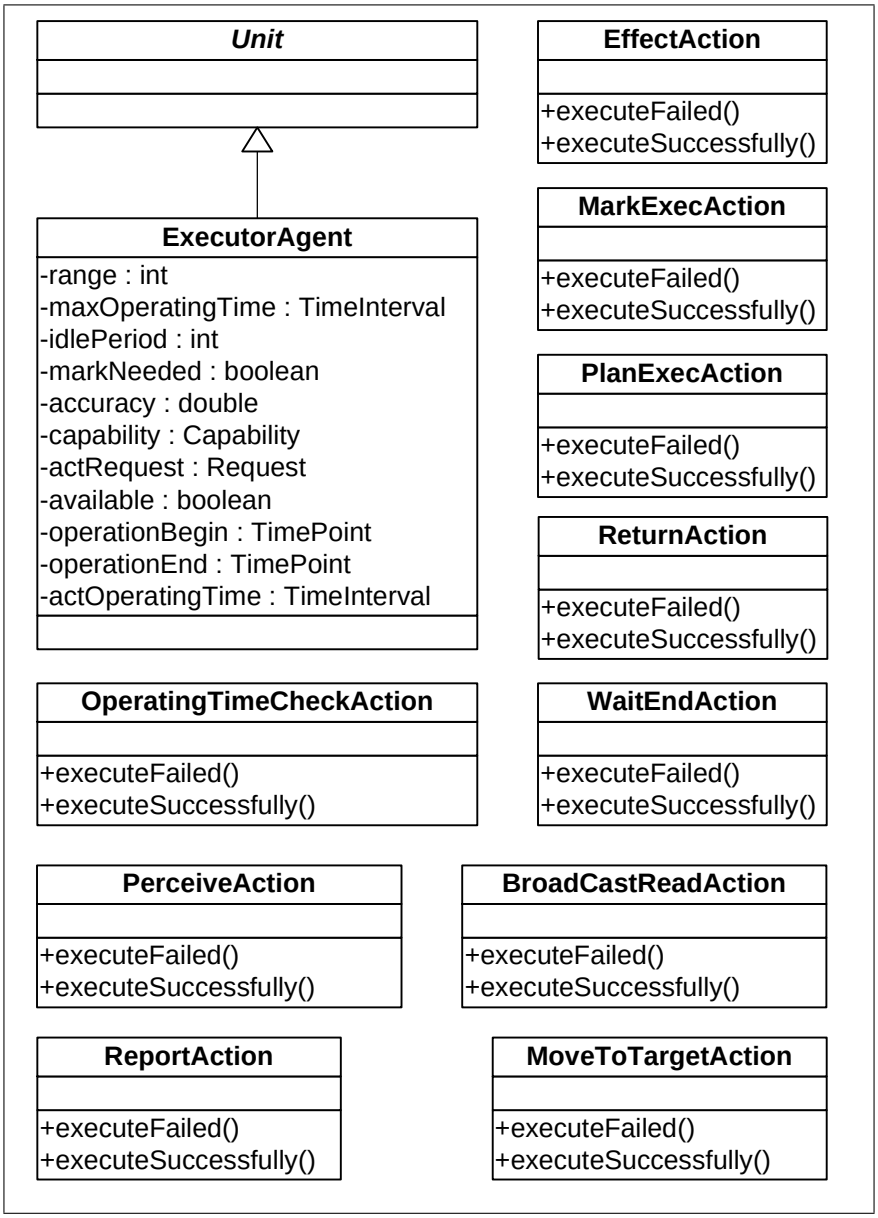

Abbildung 3.6: Eine Übersicht über relevante Klassen des Paketes ExecutorAgent

befindet. Anschließend beendet er mit der MarkExecAction oder der EffectAction seinen Auftrag.

- Nach jedem erfüllten oder abgebrochenen Auftrag (ein Auftrag wird abgebrochen, falls das Ziel verschwindet) kontrolliert der Leistungserbringer mit der OperatingTimeCheckAction, ob seine Einsatzzeit überschritten wurde. Wenn dies der Fall ist, oder er der Plattform Airplane angehört, so kehrt er sofort durch die ReturnAction zu seinem Startpunkt zurück. Wenn er der Plattform Schiff angehört, so bleibt er an seiner Position. Dann und wenn er nach Rückkehr wieder an seinem Startpunkt angelangt ist, wartet er entsprechend seiner Ruhezeit (Attribut idlePeriod). Ist diese abgelaufen, ist er durch die WaitEndAction wieder verfügbar. Sollte die Einsatzzeit jedoch nicht überschritten worden sein, so ist er sofort wieder verfügbar.

- Wenn der Leistungserbringer keinen vorgesetzten Koordinator besitzt, so ist er in der Lage durch die BroadCastReadAction Anforderungen durch den BroadCast-Channel zu empfangen. Wenn er unter den Empfängern eines Broadcastes der Bestgeignetste ist, so wird er die Anforderung annehmen und sich, falls notwendig, selbstständig noch einen Markierer suchen, den er aus den anderen Empfängern des Broadcastes auswählt. Anschließend übermittelt er sich selbst und unter Umständen dem Markierer durch ein OrderEvent den Auftrag und arbeitet diesen wie einen normalen Auftrag beginnend mit der PlanExecAction ab.

Interaktion mit anderen Submodellen

Im Regelfall nimmt zunächst ein Koordinator eine Zustandsänderung an diesem Agent vor, in dem er einen Auftrag übermittelt, den der Leistungserbringer dann erfüllen wird. An dessen Ende interagiert dieser Leistungserbringer noch mit dem Ziel, in dem es zerstört oder markiert wird. Es kann jedoch auch eine Zustandsänderung durch ein Ziel vorgenommen werden, in dem es entdeckt wird. In so einem Fall wird der Leistungserbringer es weitermelden. Wenn der Leistungserbringer keinen vorgesetzten Koordinator hat, so ist er in der Lage den BroadCastChannel zu empfangen. Aus diesem kann er Anforderungen annehmen, welche grundsätzlich von jedem anderen Agenten aufgegeben werden können.

3.2.6 Objekt Ziel

Struktur und internes Verhalten

In Abbildung 3.7 sind die zwei für das Ziel wesentlichen Klassen gezeigt.

- Jedes Ziel hat einen festen Typ. Die Eigenschaften dieses Types werden in der Klasse TargetType zusammengefasst.

- Ein einzelnes Ziel wird durch die Klasse Target abgebildet. Diese Klasse bietet die Methode fight(), durch die ein Ziel bekämpft und aus der Karte entfernt wird.

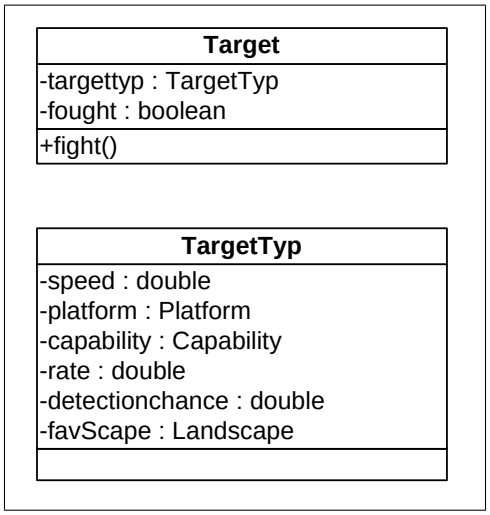

Abbildung 3.7: Eine Übersicht über relevante Klassen des Objektes Target

Interaktion mit anderen Submodellen

Ein Zustandswechsel erfolgt durch den Agenten ExcutorAgent oder durch das Event TargetDeleteEvent.

3.3 Softwarebeschreibung

Die gesamte Implementierung geschah in Java. Dabei wurde das Referenzmodell GRAMS sowie log4j verwendet.

Das ausführbare Programm, eine Beispielkonfiguration, sowie die JavaDoc befindet sich im Anhang.

3.4 Simulationsinfrastruktur

Um eine erfolgreiche Simulation durchführen zu können, ist eine Umweltdatei (karte.env) und eine Konfigurationsdatei (konfiguration.xml) notwendig. Die Umweltdatei enthält die komplette Karte und kann durch den beiliegenden Kartengenerator. Die Konfigurationsdatei muss manuell erstellt werden. Dabei ist zunächst das Festlegen der Zeit notwendig:

<Time resolution=`Auflösung` end=`Simulationsdauer`/>

Dabei ist

- `Auflösung` die gewählte Auflösung des Modells, zum Beispiel `"TIME_STEPS"` und
- `Simulationsdauer` muss ein Integer sein, zum Beispiel `"1500"`.

Anschließend erfolgt die Festlegung von Umweltdaten:

<Environment width=`Breite` height=`Höhe` envFile=`Pfad` priorisierung=`Priorisierung` />

Dabei ist:

- `Breite` die Breite der Karte, zum Beispiel `"100"`,
- `Höhe` die Höhe der Karte, zum Beispiel `"30"`,
- `Pfad` der Pfad zur Umweltdatei „`karte.env"` und
- `Priorisierung` die Priorisierungsmethode, die von allen Agenten im dezentralen Fall angewendet werden soll, zum Beispiel „`HYBRID"`.

Darauf folgt die Spezifikation aller in der Simulation auftretenden Agenten:

<Agents>
<`Agenttyp` name=`Name` id=`id` ...`weitere notwendigen Daten`/>
weitere Agenten
⋮
</Agents>

Dabei ist:

- `Agenttyp` die Art des Agenten, zum Beispiel `"Koordinator"`,
- `Name` ein beliebig gewählter Name, dieser hat keinen Einfluss auf den Ablauf der Simulation, zum Beispiel `"Linker Koordinator"` und
- `id` muss eine fortlaufende Nummer sein, mit der der Agent modellintern identifiziert wird.
- `Weitere notwendige Daten` des beschriebenen Agenten müssen hier folgen. Ein Realisierungsbeispiel bietet hierzu die Implementierung.

Anschließend werden die verschiedenen Zieltypen ähnlich der Agenten beschrieben:

<Ziele>
<Ziel id=`id` .../>
weitere Ziele
⋮
</Ziele>

Zum Schluss müssen noch alle Bewegungseinschränkungen beschrieben werden:

<Bewegung>
<`Plattform Landschaftsmuster=Einschränkung` />
⋮
</Bewegung>

Dabei gilt:

- `Plattform` ist eine der Plattformen, die in diesem Modell existieren. Es muss für jede Plattform eine extra Zeile definiert werden.
- `Landschaftsmuster` ist das Landschaftsmuster, für die die folgende Einschränkung gilt. Dabei muss für jedes Landschaftsmuster jeder Plattform eine Einschränkung definiert werden.
- `Einschränkung` ist ein Doublewert, der die Bewegungseinschränkung zwischen „`0.0`" und „`1.0`" angibt. „`0.0`" steht dabei für unpassierbar, während „`1.0`" bedeutet, dass hier keine Bewegungseinschränkung vorliegt.

Nachdem diese Konfigurationsdatei erstellt wurde, kann die Simulation durchgeführt werden. Anschließend werden die Ergebnisse in einer Statistik präsentiert, welche zum späteren Vergleich gespeichert wird.

3.5 Weitere Aspekte

3.5.1 Unterschiede zum formalen Modell

Im Verlauf der Implementierung haben sich einige Unterschiede zum formalen Modell ergeben. Grund dafür war entweder eine zu große Komplexität, die nicht im Verhältnis zu den Zielparametern steht, oder technische Unmöglichkeit.

- Im ausführbaren Modell ist eine Mehrfachbekämpfung ausgeschlossen. Daher wurde dieser Zielparameter entfernt, denn um ihn zu realisieren, hätten absichtlich logische Fehler in den Programmcode produziert werden müssen.
- Die erweiterte Karte wurde entfernt. Ursprünglich sollte diese zur Simulation von Leistungserbringern, die ihren Startpunkt außerhalb der Karte haben, dienen. Sie wurde entfernt, da sie die Kartengröße vervierfacht hätte und dies die technisch mögliche Kartengröße stark eingeschränkt hätte. Mit erweiterter Karte würden OutOfMemoryExceptions bereits bei relativ kleinen Karten sehr wahrscheinlich werden.
- Es wurden die Bewegungsfähigkeit der Ziele nicht implementiert. Dies sollte ursprünglich dazu dienen, Ziele nach einer gewissen Zeit zu entfernen, um somit eine ineffektive Aufklärung oder Koordination zu identifizieren. Die Implementierung

dieses Aspektes hätte die Komplexität des Modells sehr stark gesteigert und dies steht in keinem Verhältnis zu den zu untersuchenden Zielparametern. Stattdessen wurde ein neues Umweltereignis eingeführt, welches Ziele nach einer gewissen Zeit selbstständig entfernt und somit das gewünschte Negativergebnis erzeugt.

- Im formalen Modell existiert eine Einschränkung, dass kein Ziel auf einem Feld erzeugt werden darf, auf dem sich bereits ein Agent befindet. Die Umsetzung dieser Einschränkung ist technisch nicht möglich, da Einschränkungen laut GRAMS immer an eine Action gebunden sind. Die Ziele werden jedoch durch ein Event erzeugt. Die Einschränkung wurde daher als Abfrage direkt bei dem Erzeugen des Zieles umgesetzt.

3.5.2 Erweiterung

Obwohl das vorliegende Modell bereits sehr komplex ist, ist eine Erweiterung dennoch möglich. Zum einen können die verwendeten Enums ohne Methoden (Landschaftsformen, Plattformen, Kapazitäten) leicht erweitert werden. Das Enum Priorizitation, in dem die Priorisierungsmethoden definiert werden, kann ebenfalls erweitert werden. Hierbei helfen abstrakte Methoden, um die Lauffähigkeit zu garantieren.

Weiterhin ist es möglich, Einschränkungen, die sich aufgrund der zu hohen Komplexität ergeben haben, noch umzusetzen. Ein Beispiel hierfür wären fliegende oder bewegliche Ziele.

4 Simulationsergebnisse

4.1 Ausführungsumgebung

Die Simulation ist mit dem erstellten Programm durchführbar. Benötigt wird dafür nur eine Java Runtime Environment der Version 5 oder höher. Um eine fehlerfreie Simulation zu gewährleisten dient folgende 3-Schritte-Anleitung:

1. Schritt Im gleichen Ordner muss sich eine Kartendatei „karte.env“ befinden. Diese kann entweder durch das beiliegende Tool *Kartengenerator* generiert werden oder per Hand.

2. Schritt Der Ordner, der das Simulationsprogramm enthält, muss sich außerdem eine Konfigurationsdatei befinden. Diese muss „konfiguration.xml“ heißen und wie in Kapitel 3.4 aufgebaut sein. Als ein Anhaltspunkt kann dabei die schon vorhandene Beispielkonfiguration dienen. Bei den Angaben innerhalb der Konfigurationsdatei ist auf folgendes zu achten:

   - Alle Koordinaten müssen innerhalb der durch `width` und `height` definierten Grenzen liegen. Diese beiden Werte müssen die tatsächliche Kartengröße der in der „karte.env“ definierten Umgebung wiederspiegeln!
   - Wenn die Startpunkte der Agenten oder die Wegpunkte der Aufklärer festgelegt werden, so ist darauf zu achten, dass diese auch durch den Agenten erreicht werden können. Wenn ein Wegpunkt also auf einem Feld des Types `MOUNTAIN` liegt, der Aufklärer der Plattform `ARMOR` angehört und in den Bewegungseinschränkungen festgehalten ist, dass Fahrzeuge Gebirge nicht betreten können, so wird das zu einem Fehler führen. Ebenfalls muss jeder dieser Wegpunkte erreichbar sein. Unter Umständen kann es also erforderlich sein, dass die Kartendatei per Hand nacheditiert werden muss.
   - Wenn für bestimmte Plattformen ein Gebiet nicht mehr betretbar sein soll, so müssen ebenfalls alle Wegpunkte und Startpunkte kontrolliert werden. Beispiel: In der Beispielkonfiguration ist das Waldgebiet für Fahrzeuge befahrbar. Wenn nun das Waldgebiet nicht mehr befahrbar sein soll, so muss anschließend bei allen Agenten der Plattform `ARMOR` kontrolliert werden, ob ihr Startpunkt nicht von diesem Typ ist und ob diese immernoch von ihrem Startpunkt zu allen Wegpunkten gelangen können, falls die ein Aufklärer sind.

   Achtung! Diese Bedingungen sind sehr wichtig, denn wenn gegen eine davon verstoßen wird, so funktioniert der Simulator fehlerhaft. Es wird nicht zwangsläufig zum Scheitern einer Simulation führen, sondern eher zu verfälschten Ergebnissen.

3. Schritt Wenn die vorangegangen Schritte erfolgt sind, kann durch Ausführen des Programmes die Simulation gestartet werden. Nachdem diese automatisch mehrfach durchgeführt wurde, wird eine CSV Datei im gleichen Ordner erstellt, in der die Ergebnisse der Simulation aufgelistet sind. Diese CSV Datei enthält alle Simulationsergebnisse, welche durch ein Semikolon getrennt werden und kann zum Beispiel mit MS Excel geöffnet und dann per Hand ausgewertet werden. Wichtig! Das Programm muss hierfür auf dem Datenträger schreiben können. Es ist also zuerst nötig das Programm und alle benötigten Dateien von dem beiliegenden Datenträger zu kopieren, sonst kann keine Ausgabedatei erstellt werden.

4.2 Durchführung der Experimente

Es werden insgesamt neun Exerimente durchgeführt. Diese Experimente dienen dabei nur einem ersten Testen des Modells und lassen nicht zwangsläufig Rückschlüsse auf die Realität zu. Dies liegt daran, dass die Eingabegrößen des Modells frei gewählt sind und noch nicht validiert wurden. Die ersten drei Exerimente finden alle mit einer zentralen Koordinationsstrukur und jeweils einer der drei verschiedenen Priorisierungsmethoden statt. Anschließend folgen drei weitere Experimente mit einer dezentralen Koordinationsstruktur und ebenfalls je einer der drei Priorisierungsmethoden. Diese ersten sechs Experimente dienen dem direkten Vergleich der zwei Koordinationsstrukturen bei verschiedenen Priorisierungsmethoden. Das siebte Experiment benutzt die gleiche Konfiguration wie das erste Experiment mit zentraler Koordination, jedoch wird hier die Anzahl der Leistungserbringer und Aufklärer, die zur Verfügung stehen, verdoppelt. Ähnlich wie im achten Experiment, welches bei dezentraler Koordination und ebenfalls doppelt so vielen Leistungserbringern und Aufklärern stattfindet. Im neunten Experiment liegt eine ähnliche Situation wie im ersten Experiment vor, jedoch sind hier alle Koordinationsbereiche aufgeteilt in je zwei einzelne Bereiche, sodass doppelt so viele Koordinatoren vorhanden sind. Jeder einzelne hat jedoch nur noch halb so viele Leistungserbringer.

Jede der neun Konfigurationen wird 100 mal durchgeführt und aus den Zielparametern wird der Durchschnitt gebildet.

Im folgenden Kapitel wird die Startkonfiguration der Experimente noch etwas genauer beschrieben. Die konkrete Umsetzung in einer XML-Konfigurationsdatei ist in den angehängten Szenarien zu sehen.

4.3 Dokumentation der Eingabedaten

4.3.1 Experimente 1-3

In diesen ersten drei Experimenten liegt eine zentrale Koordinationsstruktur vor. In dieser wird das gesamte Einsatzgebiet von zwei Koordinatoren kontrolliert, die einen weiteren gemeinsamen Vorgesetzten haben.

Koordinatoren:

- *Linker Koordinator*

  Dieser Koordinator kontrolliert den linken Kartenbereich, in welchem sich ein Seegebiet befindet. Dieses wird von einem Seeaufklärer (MPA) überwacht. Zur Zielbekämpfung stehen dem Koordinator drei Schiffe für Seeziele, sowie eine Einheit Infanterie für Landziele zur Verfügung. Wenn dieser Koordinator ein Ziel nicht bekämpfen kann, so gibt er die Anforderung an seinen Vorgesetzten, den *Super Koordinator* ab.

- *Rechter Koordinator*

  Diesem Koordinator untersteht der Großteil der Landfläche. Zu dessen Überwachung setzt er zwei Infanteriepatrouillen ein. Wenn diese ein Ziel entdecken, so stehen dem Koordinator zwei Infanterieeinheiten und zwei Einheiten gepanzerte Fahrzeuge zur Verfügung. Wenn ein Ziel nicht bekämpft werden kann, so gibt er die Anforderung an seinen Vorgesetzten, den *Super Koordinator* ab.

- *Super Koordinator*

  Beide Koordinatoren besitzen diesen Koordinator als übergeordneten Vorgesetzten. Er kann keine Anforderungen von Aufklärern direkt annehmen, sondern bekommt diese nur von seinen untergebenen Koordinatoren, wenn diese eine Anforderung nicht erfüllen können. Zur Auftragserfüllung untersteht diesem Koordinator ein Kampfflugzeug. Wenn er damit ein Ziel nicht bekämpfen kann, so prüft er, ob nun einer seiner untergebenen Koordinatoren das Ziel bekämpfen kann und gibt die Anforderung eventuell ab. Kann weder er noch seine Untergebenen die Anforderung erfüllen, so wird er dies so lange weiterprüfen, bis wieder freie Kapazitäten zur Erfüllung bereit stehen oder das Ziel verschwunden ist.

4.3.2 Experimente 4-6

Diese Experimente nutzen eine dezentrale Koordinationsstruktur. Es sind dabei sowohl Karte, Ziele, Aufklärer als auch Leistungserbringer die selben wie in den ersten drei Experimenten. Der einzige Unterschied besteht darin, dass kein Koordinator existiert, sondern die Agenten über einen Broadcast-Channel miteinander kommunizieren.

4.3.3 Experimente 7-8

In diesen beiden Experimenten stehen die doppelte Anzahl Leistungserbringer und Aufklärer zur Verfügung. Die Karte und die Konfiguration der Ziele sind dabei identisch mit denen der bisherigen Experimente. Die Koordinatoren besitzen im Vergleich zu den Experimenten 1-3 jeden Leistungserbringer und Aufklärer doppelt. Die Routen der Aufklärer sind ebenfalls die selben, mit dem einzigen Unterschied, dass sie in gegensätzlichen Richtung beschritten werden.

4.3.4 Experiment 9

Das neunte Experiment nutzt ebenfalls die gleiche Konfiguration wie das erste Experiment. Hier wurden lediglich das Verantwortungsgebiet der zwei Koordinatoren halbiert und jeweils ein weiterer Koordinator eingeführt. Die Leistungserbringer der bisherigen Verantwortungsgebiete werden gleichmäßig auf die neuen verteilt. Dies bedeutet, dass einem Koordinator des linken Bereiches zwei Schiffe und dem anderen ein Schiff und eine Einheit Infanterie unterstehen. In dem rechten Bereich unterstehen jedem Koordinator eine Einheit Infanterie und eine Einheit gepanzerte Fahrzeuge. Der übergeordnete Koordinator wird nicht geändert.

4.4 Dokumentation der Experimentergebnisse

Es wurde jedes der genannten Experimente 100 mal durchgeführt und der Durchschnittswert ermittelt. Diese Durchschnittswerte sind in folgender Tabelle dokumentiert. Dabei sind die Leistungserbringer in Gruppen eingeteilt. Gruppe 1 sind die Leistungserbringer des linken Bereiches der Karte, in welchem sich ein Seegebiet und angrenzende Küste befindet, diese Leistungserbringer sind in der Regel Koordinator 1 zugeordnet. In Gruppe 2 sind alle Leistungserbringer zusammengefasst, die sich zum Start der Simulation im rechten Bereich der Karte aufhalten und Koordinator 2 unterstellt sind. Die dritte Gruppe wird aus einem globalen Leistungserbringer gebildet, der einem dritten vorgesetzten Koordinator 3 unterstellt ist.

Zielparameter	Experimentnummer								
	1	2	3	4	5	6	7	8	9
Auslastung der LE Gruppe 1 [%]	28,2	27,5	28,7	22,3	32,7	29,8	11,7	9,1	25,3
Auslastung der LE Gruppe 2 [%]	33,0	36,0	35,5	29,1	38,7	29,1	26,7	12,2	32,1
Auslastung der LE Gruppe 3 [%]	22,2	25,7	36,3	56,9	35,6	42,1	1,8	52,4	42,4
Anzahl der identifizierten Ziele: Seeaufklärer	27,5	28,4	27,4	18,9	17,6	19,2	36,2	33,7	40,8
Anzahl der identifizierten Ziele: Patrouille 1	7,2	7,5	7,8	6,5	6,9	6,6	11,8	11,9	8,2
Anzahl der identifizierten Ziele: Patrouille 2	6,9	8,0	7,9	6,5	6,6	6,9	11,9	10,7	7,5
Benötigte Zeit pro Zuweisung: Koordinator 1 (relativ zu seiner Bearbeitungszeit)	1,36	1,44	1,38	—	—	—	1,02	—	3,73
Benötigte Zeit pro Zuweisung: Koordinator 2 (relativ zu seiner Bearbeitungszeit)	1,21	1,40	1,36	—	—	—	1,03	—	1,5
Benötigte Zeit pro Zuweisung: Koordinator 3 (relativ zu seiner Bearbeitungszeit)	8,64	9,44	7,79	—	—	—	1.15	—	7,0
Zeitbedarf bis zur Identifizierung: Koordinator 1 [ZE]	17,0	18,1	17,5	—	—	—	8,9	—	17,8
Zeitbedarf bis zur Identifizierung: Koordinator 2 [ZE]	111,5	190,2	107,6	—	—	—	85,1	—	105,3
Zeitbedarf bis zur Bekämpfung: Koordinator 1 [ZE]	70,7	77,2	71,9	—	—	—	49,9	—	87,9
Zeitbedarf bis zur Bekämpfung: Koordinator 2 [ZE]	105,7	118,3	114,5	—	—	—	103,5	—	109,3
Gesamtanzahl erzeugte Ziele	48,7	48,0	48,7	49,0	48,3	48,8	48,9	48,0	49,0
Davon wurden identifiziert [%]	66,4	68,6	69,3	67,7	70,2	70,3	78,7	77,8	68,0
Davon wurden bekämpft [%]	77,0	73,6	74,4	68,4	61,1	62,2	80,3	83,6	74,7
Gesamter Zeitbedarf Identifizierung bis Bekämpfung [ZE]	83,1	91,1	86,7	69,5	89,3	77,2	73,7	58,5	94,9

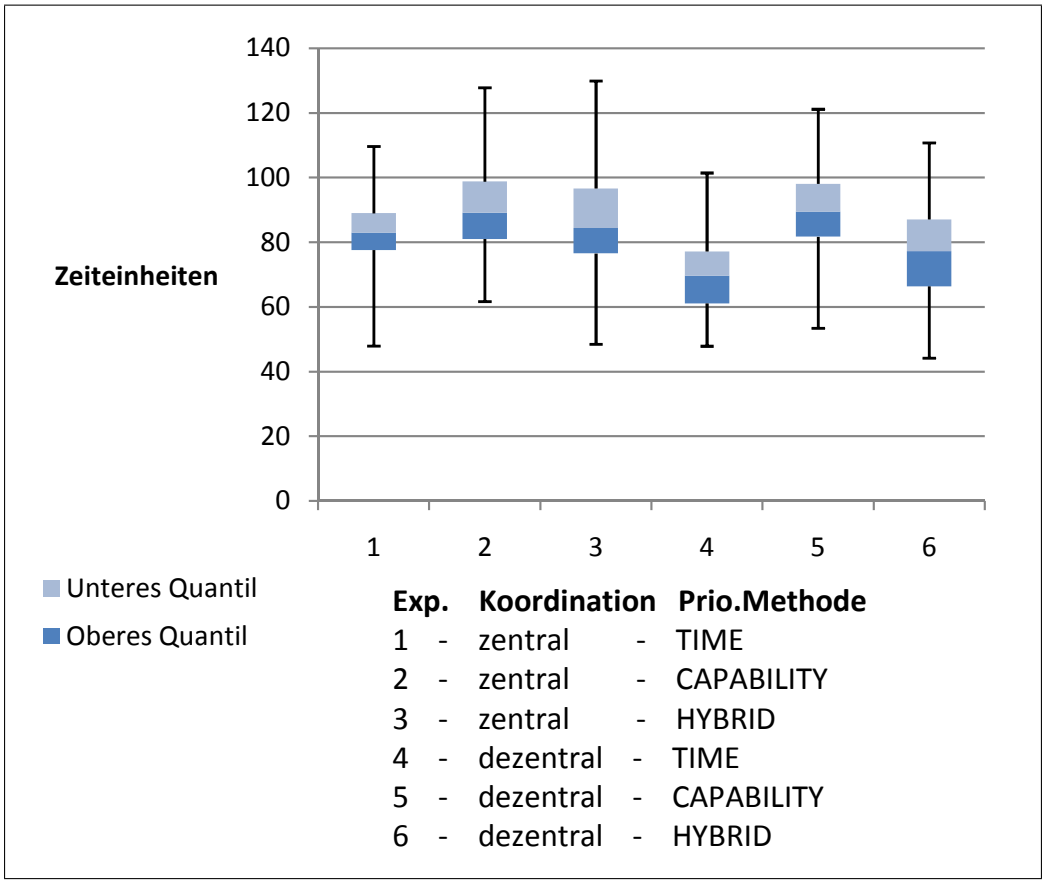

Abbildung 4.1: In diesem Boxplotdiagramm ist die durchschnittliche Gesamtdauer der Identifizierung bis zur Bekämpfung eines Zieles dargestellt. Es sind sowohl die minimalen und maximalen Werte, als auch das obere und untere Quantil abgebildet, in dem sich jeweils 25% der Werte befinden.

4.5 Analyse der Ergebnisse

Bereits bei der Analyse der ersten drei Experimente fällt auf, dass die Wahl der Priorisierungsmethode nur beschränkt Einfluss auf die Zielparameter hat. Bei allen Experimenten war immer die Priorisierungsmethode TIME diejenige, welche die kürzeste durchschnittliche Dauer zwischen Identifizierung und Bekämpfung eines Ziels verursachte (siehe Abbildung 4.1). Der Grund hierfür ist offensichtlich, dass immer der Leistungserbringer ausgewählt wird, der die Ziele auch am schnellsten bekämpfen kann.

Die Wahl der Priorisierungsmethode hat jedoch kaum Einfluss darauf, wieviele Ziele insgesamt bekämpft werden können (siehe Abbildung 4.2). Dies liegt daran, dass unabhängig von der Priorisierungsmethode alle Anforderungen früher oder später bearbeitet werden können. Sollte durch eine schlechte Wahl der Priorisierungsmethode ein unpassender Leistungserbringer ausgewählt werden und deshalb ein Ziel nicht bekämpft werden

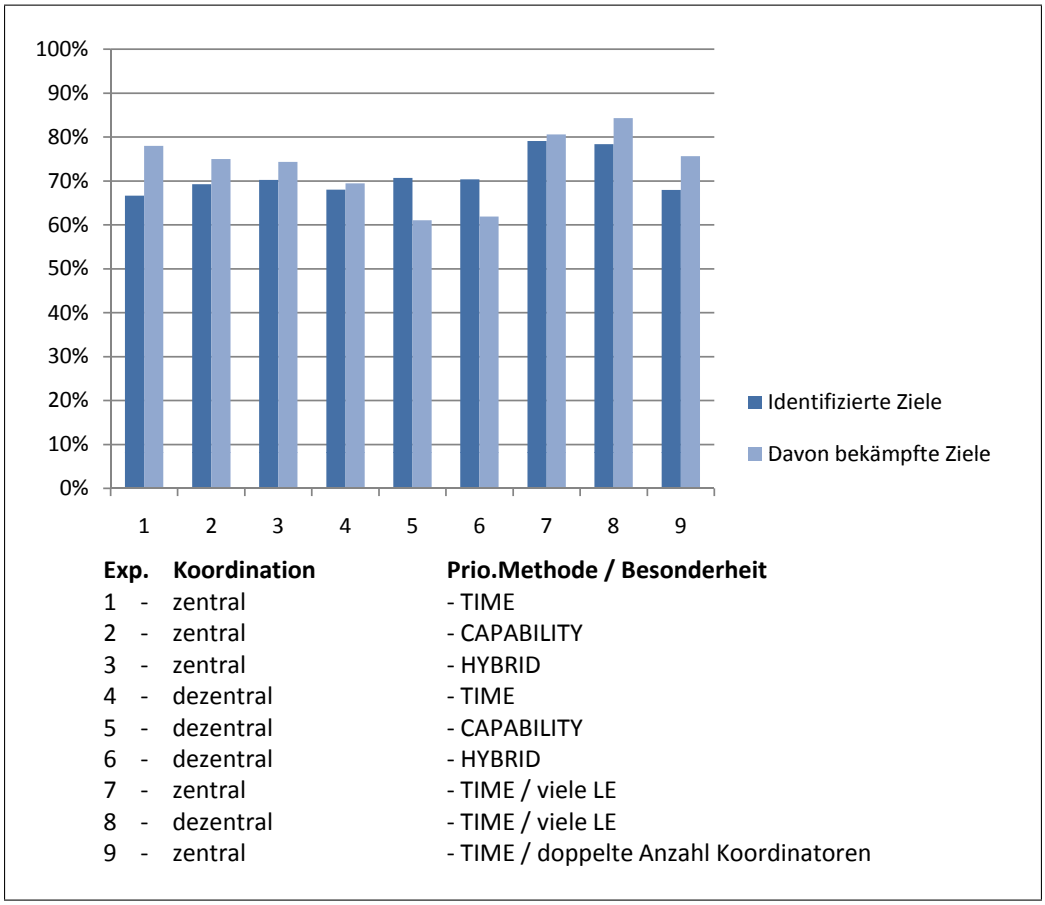

Abbildung 4.2: In diesem Balkendiagramm ist dargestellt, wieviele der erzeugten Ziele identifiziert wurden (linker Balken) und wieviele davon bekämpft werden konnten (rechter Balken).

können, so wird dieses Ziel entweder später bekämpft oder gar nicht. Dafür konnte jedoch eine andere Anforderung erfüllt werden, welche bei einer anderen Priorisierungsmethode eventuell nicht zeitgerecht bearbeitet werden konnte. Die Wahl der Priorisierungsmethode hat also hauptsächlich Einfluss auf die Reihenfolge. Die Auswirkung auf die entgültigen Zielparameter sind jedoch unwesentlich.

Es hat sich herausgestellt, dass eine flache Hierarchie nur bedingt besser geeignet ist. Wie in Abbildung 4.1 zu sehen, haben alle Experimente, in denen eine dezentrale Koordination gewählt wurde, eine kürzere Gesamtdauer von Identifizierung bis Bekämpfung als ihre entsprechenden zentral koordinierten Varianten. Betrachtet man jedoch Abbildung 4.2, so erkennt man, dass in den dezentral koordinierten Szenarios weniger Ziele bekämpft werden konnten, als in denen mit zentraler Koordination. Der Grund dafür ist, dass Anforderungen viel schneller erfüllt werden können und somit ein Ziel schneller bekämpft werden kann. Sollte die Bearbeitung einer Anforderung aber zu keinem Ergebnis führen, weil gerade

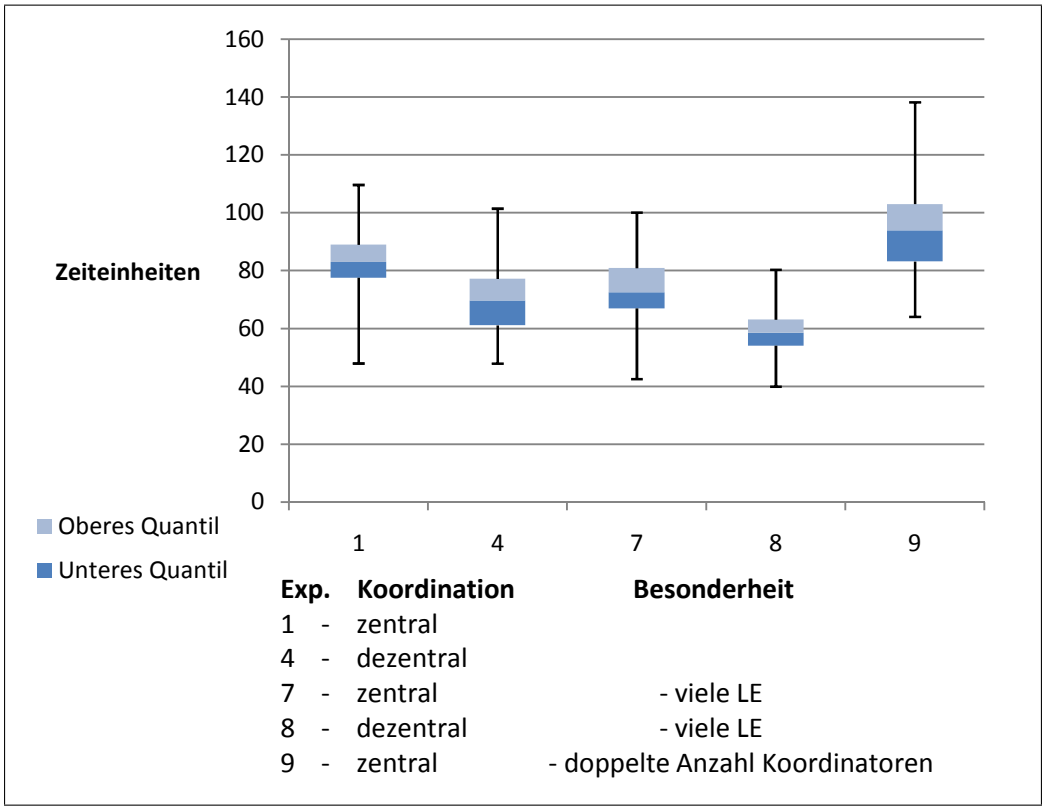

Abbildung 4.3: In diesem Boxplotdiagramm ist die durchschnittliche Gesamtdauer von Identifizierung bis zur Bekämpfung eines Zieles dargestellt. Es sind sowohl die minimalen und maximalen Werte, als auch das obere und untere Quantil abgebildet, in dem sich jeweils 25% der Werte befinden.

kein passender Leistungserbringer verfügbar ist, so wird im zentralen Koordinationsfall die Anforderung abgegeben oder später geprüft. Im dezentralen Fall aber wird sie immer später noch einmal geprüft, dabei kann es jedoch sein, dass mittlerweile das Ziel verschwunden ist. Der Anteil der identifizierten Ziele, die bekämpft werden konnten, ist also im zentralen Fall immer höher, da ein Koordinator länger versucht, eine Anforderung zu erfüllen, während sie im dezentralen Fall verworfen und erst später neu überprüft wird. Eine Ausnahme ist Experiment 8, in dem bei einer dezentralen Koordination mehr Leistungserbringer zur Verfügung standen als bei den ersten sechs Experimenten. In diesem achten Experiment war die durchschnittlich benötigte Zeit von Identifizierung bis Bekämpfung wieder kürzer als im entsprechenden zentralen Experiment (7), es konnten jedoch auch mehr Ziele bekämpft werden.

Ebenfalls sind die Ergebnisse in allen Experimenten, welche zentral koordiniert werden, umso besser, je weniger Koordinatoren vorhanden sind (siehe Abbildung 4.3). Der Grund hierfür ist ähnlich wie bei der vorangegangen Schlussfolgerung: Die Auswahl eines passenden Leistungserbringers ist sehr zeitaufwendig und nicht immer erfolgreich, sodass

ein Ziel viel eher wieder verschwindet, als dass es bekämpft werden kann.

Auffällig ist, dass bei den Experimenten 4, 5 und 6 deutlich weniger Ziele durch die Aufklärer entdeckt werden. Das ist jedoch nicht schlecht; im Gegenteil: Im zentralen Koordinationsfall findet keine Koordination zwischen den verschiedenen Aufklärern oder Leistungserbringern statt, das heißt, dass Ziele sehr oft mehrfach identifiziert werden und erst der Koordinator feststellt, dass dieses Ziel schon einmal identifiziert wurde.

Mit dem Ergebnis von Experiment 8 wird die Bedeutung der Anzahl an Leistungserbringern und Koordinatoren deutlich (siehe Abbildung 4.3). Ob dezentral oder zentrale Koordination, unabhängig von der Priorisierungsmethode ist die Dauer der Identifizierung bis Bekämpfung von Zielen immer zwischen 74 und 95 Zeiteinheiten. Doch bei Experiment 8, in dem es doppelt so viele Leistungserbringer wie normal und keine Koordinatoren gibt, ist diese benötigte Zeit nur 58,5 Zeiteinheiten. Das ist das beste Ergebnis der Experimentreihe und bedeutet also, dass die Ergebnisse umso besser werden, wenn keine Koordinatoren vorhanden sind und die Anzahl der Leistungserbringer möglichst hoch ist.

Die Auswertung der Ergebnisse der Experimentreihe bedeutet also für die Realität, dass eine dezentrale Koordination grundsätzlich einer zentralen zu bevorzugen ist, jedoch nur, wenn auch viele Leistungserbringer zur Verfügung stehen. Dies ist schwer zu realisieren, da die eigenen Mittel immer stark begrenzt sind. Zudem verlangt eine vollständig dezentrale Koodination, dass es für jeden Leistungserbringer und Aufklärer möglich sein muss, mit alle anderen eigenen Einheiten zu kommunizieren. Gerade in ungünstigem Gelände verlangt das eine sehr teure und teilweise auch schwer zu transportierende Ausrüstung. Zusätzlich steigert es das Risiko der eigenen Einheiten sehr stark, denn wenn ein Teil der Kommunikationstechnik in die Hand von gegnerischen Kräften gerät, geht der eigene Informationsvorteil verloren. Zudem ist die Anzahl der eigenen Einheiten meistens stark beschränkt (Mandate, Personalmangel, Kostengründe, etc.). Die ideale Lösung ist also sehr schwer umzusetzen. Wenn zu wenige eigene Einheiten vorhanden sind, ist eine zentrale Koordination zu empfehlen, da ein Koordinator sich auf das Verwalten der knappen Ressourcen spezialisieren kann und die Leistungserbringer entlastet. Die Hierachie sollte jedoch sehr flach sein, damit es vermieden wird, dass Koordinatoren sich bei der Erfüllung von Anforderungen untereinander abstimmen müssen. Im Idealfall ist also ein Koordinator für alle Leistungserbringer im Einsatzgebiet verantwortlich.

5 Fazit

5.1 GRAMS-Referenzmodell

Das GRAMS-Referenzmodell war für die Erstellung des Modells und dessen Implementierung sehr hilfreich. Die Einarbeitungszeit in das Referenzmodell war relativ kurz, da viele Definitionen und Festlegungen intuitiv sind. Ebenfalls musste keine eigene Simulationsumgebung implementiert werden, da durch die Referenzimplementierung schon mehrere verschiedene Simulatoren bereitgestellt werden.

Durch die strikte Trennung in Actions und Events wurde die Wiederverwendung vereinfacht, da Events die Auslöser von mehreren Actions sein können und auch von mehreren unterschiedlichen Actions erstellt werden können. Durch diese Festlegung lässt sich die Kommunikation zwischen den einzelnen Agenten sehr gut abbilden, da ein Agent lediglich ein Event initieren muss. Dieses Event ist dann der Auslöser für die Action eines anderen Agenten.

Die referenzmodellseitige Festlegung, dass lediglich ein Event der Auslöser für eine Action sein kann, erweist sich teilweise jedoch auch problematisch, da es die Actions schwer kontrollierbar macht. Es ist nicht ohne das Einführen weiterer Attribute möglich, festzustellen, ob gerade eine Action eines Agenten aktiv ist. Ebenfalls können so Actions auch mehrfach zum gleichen Zeitpunkt aktiv sein, ohne das dies leicht abzuprüfen ist. Besonders wenn ein Agent eine lange Kette von internen Actions und Events besitzt, wird es schwer, zu kontrollieren an welcher Stelle dieser Kette sich der Agent gerade befindet. Falls sich verschiedene Actions gegenseitig ausschließen, ist es sehr aufwendig, dies in der Implementierung umzusetzen und zu kontrollieren. Ein Beispiel: Seien A, B und C verschiedene Actions. Action A kann jederzeit durch ein externes Event aktiviert werden, außer es ist gerade C aktiv. Wenn A ausgeführt wird, so folgt daraus immer B. Durch diese Action B kann nun entweder wieder A oder aber C ausgelöst werden. Sollte Action C ausgeführt werden, so folgt daraus immer ein externes Event oder aber wieder A. Diese Kette lässt sich sehr gut durch Events und Actions abbilden. Die Bedingung für die Action A umzusetzen, ist jedoch schon schwerer: Dafür ist das Einführen eines Booleans notwendig, welcher durch eine Constraint beim Aktivieren von A abgeprüft wird. Das Setzen dieses Booleans muss nun jedoch durch alle Events geschehen, welche C aktivieren, denn sonst kann A und C theoretisch zu gleich aktiv sein. Vor allem wenn diese Kette aus A-B-C später erweitert wird, muss sichergestellt werden, dass wieder alle Events, die C auslösen sollen, auch den entsprechenden Boolean setzen. Gerade bei Projekten, die über eine längere Zeitdauer laufen und bei dem mehrere Entwickler mitarbeiten, kann dies zu ärgerlichen und schwer zu identifizierenden Fehlern führen.

Grundsätzlich ist die Kombination aus Events und Actions jedoch sehr nützlich, da so sehr viele verschiedene oder gleiche Agenten modelliert werden können, welche durch die Wiederverwendung die gleichen Actions nutzen.

5.2 Streitkräftegemeinsame Feuerunterstützung

Im Hinblick auf das betrachtete Szenario, die streitkräftegemeinsame Feuerunterstützung, hat sich herausgestellt, dass flache Koordinationsstrukturen zu bevorzugen sind. Es sollte entweder nur einen einzigen Koordinator geben, der alle verfügbaren Kräfte verwaltet oder gar keinen Koordinator, falls die Anzahl der Leistungserbringer sehr groß ist. Wenn es keinen Koordinator gibt, so muss sichergestellt werden, dass jede eigene Einheit mit jeder anderen kommunizieren kann und dass zu jedem Zeitpunkt ein geeigneter Leistungserbringer zur Verfügung steht, sonst können Ziele nicht effektiv bekämpft werden. Wie bereits in Kapitel 4.5 erläutert, ist dies jedoch sehr schwer zu realisieren. Es sollte deshalb angestrebt werden, auf sehr umfangreiche Hierachien zu verzichten.

Ebenfalls ist es ideal, wenn immer möglichst viele geeignete Kräfte im Einsatzraum vorhanden und in diesem verteilt sind, um somit kurze Wege zu den Zielen zu erreichen.

5.3 Ausblick

Auch wenn das Modell in dem sehr knappen Zeitraum, der zum Erstellen dieses komplexen Modells verfügbar war, zwar abgeschlossen werden konnte, ist es damit noch nicht beendet. An vielen Stellen ist sowohl eine Erweiterung des Modellrahmens, als auch des Modells selber möglich.

Eine wesentliche Erweiterungsmöglichkeit des Modellrahmens besteht in der Benutzerfreundlichkeit. Besonders das Erstellen einer validen XML-Konfigurationsdatei ist sehr aufwendig, da nicht nur eine Vielzahl von Parametern beachtet werden muss, sondern diese auch aufeinander abgestimmt werden müssen. So ist es zum Beispiel zwar möglich, einem Aufklärer vom Typ Fahrzeug einen Wegpunkt auf eine Koordinate des Types Gebirge zu legen. Wenn jedoch an anderer Stelle festgelegt wird, dass Gebirge für Fahrzeuge unpassierbar ist, so wird dies beim Start der Simulation zwar nicht zu einem Abbruch, wohl aber zu verfälschten Zielparametern führen. Ebenfalls werden durch die Simulation nur Rohdaten in Form einer CSV-Datei ausgegeben. Es ist jedoch notwendig, diese Daten per Hand zu analysieren. Es würde sich hier anbieten, ein weiteres Programm zu entwickeln, welches die Ein- und Ausgabe des Modells im Hinblick auf die Benutzerfreundlichkeit verbessert.

Das Modell selber lässt sich ebenfalls sehr gut erweitern. Dazu trägt vor allem die in Kapitel 5.1 angesprochene Wiederverwendbarkeit der Events und Actions bei. Es wurden jedoch auch sehr oft Enums genutzt (Landscape, Platform, Capability und Priorizitation), welche sich sehr gut erweitern lassen. Besonders das Enum Priorizitation kann um weitere Priorisierungsmethoden erweitert werden. Deren Lauffähigkeit wird durch abstrakte Methoden sichergestellt.

Bisher sind alle Eingangsparameter des Modells willkürlich gewählt. Für eine sinnvolle Nutzung ist also zuvor noch eine Verifikation und Validierung des Modells und dessen Eingangsparameter notwendig.

Mit dieser Arbeit liegt also ein abgeschlossenes Modell vor, welches sich jederzeit erweitern lässt und an dessen Ein- und Ausgabeschnittstellen sich die Benutzerfreundlichkeit noch wesentlich steigern ließe.

6 Eigenständigkeitserklärung

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die Stelle der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Neubiberg, den 05.01.2010, Christian Gerstner

Literaturverzeichnis

[1] Bel Haj Saad, Sameh ; Best, Michael ; Köster, Andreas ; Pohl, Junlan ; Waldner, Carmen ; Wang, Zhongshi ; Xu, Zhongfu ; Lehmann, Axel: Leitfaden für Modelldokumentation / ITIS Institut für Technik Intelligenter Systeme e. V. 2005. – Abschlussbericht. BMVg – Studienauftrag Nr. M/GSPO/2A024/2A924, SKZ 12 990 2 114

[2] Bloch, Joshua: *Effective Java - Second Edition*. 6. Auflage. Verlag: Addison Wesley, 2008. – ISBN-10: 0-321-35668-3

[3] Bundesministerium der Verteidigung - Generalinspekteur der Bundeswehr / Fü S II 2: *Teilkonzeption Aufklärung der Bundeswehr*. – Az: 09-02-05/VS-NfD

[4] Bundesministerium der Verteidigung - Generalinspekteur der Bundeswehr / Fü S VI 2: *Teilkonzeption Vernetzte Operationsführung in der Bundeswehr*. – Az: 09-02-05/VS-NfD

[5] Schleupner, Susann ; Dietrich, Stefan ; Gerstner, Christian: *Taktische Feuerunterstützung - Konzeptuelles Modell*. Universität der Bundeswehr München - Fakultät für Informatik - Institut für Technische Informatik und iABGmbh Ottobrunn VG 62. 2009. – Praktikumsbericht - Industriepraktikum

[6] Schmidt, Thorsten ; Fuchs, David: *A-Stern Algorithmus*. http://www.geosimulation.de/methoden/a_stern_algorithmus.htm. – Universität Tübingen - Geographisches Institut

[7] Siegfried, Robert: A General Reference Model for Agent-Based Modeling and Simulation. In: *EUMAS 2009*, 2009. – 7th European Workshop on Multi-Agent Systems

[8] Siegfried, Robert ; Lehmann, Axel ; El Abdouni Khayari, Rachid ; Kiesling, Tobias: A Reference Model for Agent-Based Modeling and Simulation. In: *Proceedings of the Spring Simulation Multiconference* SCS, 2009. – Agent-Directed Simulation Symposium