FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Entwicklung und Visualisierung von Filterregeln in autonomen Software-Agenten

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Wirtschaftsingenieurwesen Informatik

eingereicht von

Ivalina Jordakieva
Matrikelnummer 0106532

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer/in: Associate Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Michael Wimmer
Mitwirkung: Dipl.-Ing. Dr.techn. Friedrich Gelbard

Wien, 21.08.2017

(Unterschrift Verfasser/in) (Unterschrift Betreuer/in)

Kurzfassung

SiMA (Simulation of the Mental Apparatus & Applications) ist ein interdisziplinäres Projekt bei dem Techniker und Psychoanalytiker ein implementierbares Modell einer Entscheidungsfindungseinheit entwickeln, welches auf dem psychischen Apparat des Menschen beruht. Dabei wird besonderes Augenmerk darauf gelegt, dass das entwickelte System in seinen Funktionen (und seinem Verhalten) dem menschlichen Vorbild nahe kommt. Dazu gehört auch das Umsetzen sozialer Regeln und Abwehrmechanismen. Diese bilden gemeinsam einen bionisch inspirierten Filtermechanismus für die Sensordaten des Agenten. Der Filtermechanismus ermöglicht es Konflikte zwischen Wahrnehmung und Zielen des Agenten sowie sozialen Regeln zu erkennen und aufzulösen.

Einem *Use-Case driven development*-Ansatz folgend, wird das SiMA Projekt in der vorliegenden Arbeit iterativ weiterentwickelt. Grundlage sind psychoanalytische Use-Cases, die das menschliche Verhalten abbilden. Der aktuell zu erfüllende Use-Case befasst sich mit der Verfeinerung der Abwehrmechanismen. Zur Erfüllung dieses Use-Cases ist es notwendig die Abwehr auf die nächste Entwicklungsstufe zu heben. Dies beinhaltet das Formulieren der Filter-Regeln, eine Wissensrepräsentation um diese abzubilden und Visualisierungsmöglichkeiten um Filterregeln und ihren Einfluss auf die Entscheidungsfindung validieren zu können. Des Weiteren muss die Infrastruktur zum Umsetzen des aktuellen Use-Cases weiter entwickelt werden, um eine Validierung mit dem Gesamtsystem zu ermöglichen. Dies fordert die Entwicklung eines Inventarsystems und neuer Aktionen für den Agenten.

Nach der Analyse verschiedener Eingabemöglichkeiten und Wissensrepräsentationen bezüglich ihrer Eignung für das SiMA-Modell hat sich gezeigt, dass keine existierende Variante die spezifischen Anforderungen an Flexibilität und Lesbarkeit erfüllt. Aus diesem Grund wurde ein eigenes Eingabe- und Speicherformat für die Filterregeln entwickelt und in das existierende Modell integriert. Der Ansatz wurde in die existierende Java-Implementierung des SiMA-Modells implementiert und in der Multi-Agenten Simulationsumgebung MASON getestet. Die Simulationsergebnisse zeigen die Eignung des neu entwickelten Regelformats und die Möglichkeiten die sich durch den Einsatz des bionisch inspirierten Filtermechanismus ergeben.

Abstract

SiMA (Simulation of the Mental Apparatus & Applications) is an interdisciplinary project, in which technicians and psychoanalysts cooperatively develop an implementable model of a decision-making unit based on the human psychological apparatus. Particular attention is attributed to designing the developed system akin to the human model in its functions (and its behavior). This includes the implementation of social rules and defense mechanisms. Together they form a bionically inspired filter mechanism for the agent's sensory data. The filter mechanism allows the user to identify and resolve conflicts arising between the agent’s perception and goals or implemented social rules.

In this work, the SiMA project is further developed iteratively, following a use-case-driven methodological approach. The basis is psychoanalytic use cases, which depict human behavior. The current use case revolves around the refinement of defense mechanisms. To fulfill this use case it is necessary to enhance the defense mechanisms to the next development stage. This includes the formulation of filter rules, knowledge representation for mapping these filter rules and validating their visualization possibilities and their influence on decision making. Additionally, the infrastructure for implementing the current use case has to be advanced further in order to enable validation with the entire system. This necessitates the development of an inventory system and new actions for the agent.

After analyzing different input possibilities and knowledge representations regarding their suitability for the SiMA model, it could be shown that no existing model meets the specific requirements for flexibility and legibility. As a result, a separate input and memory format was developed for the filter rules and integrated into the existing model. The approach was embedded into the existing Java implementation of the SiMA model and tested in a multi-agent simulation environment (MASON). The simulation results supported eligibility of the newly developed rule format and highlighted the potential applications for the use of this bionically inspired filter mechanism.

Danksagungen

Als allererstes möchte ich meinem Assistenz-Betreuer Dr. Friedrich Gelbard ganz herzlich für die liebevolle zur Ende Betreuung nach meiner Karenz-Pause bedanken, trotz, dass sein Vertrag an der Universität bereits abgelaufen war!

Danke an Prof. Dr. Michael Wimmer, dass er mich so kurzfristig vor der Deadline aufgenommen hat. Für die absolut schnelle Response-Zeit bei eMail- Anfragen und für die Mühe, dass die Arbeit einen Abschluss findet!

Danke an Dir Stefan Kollmann dafür, dass Du mich angetrieben hast und Dein Zuvorkommen bezüglich jeglicher Fragen im Projekt!

Danke an meinem Partner Dr. Stefan Malainer für das Korrektur-lesen und für die Ermöglichung, dass ich an dem Projekt weiter bleiben kann!

~*~.~*~

Inhaltsverzeichnis

Abkürzungen

ACT-R	Adaptive Control of Thought-Rational
AGI	Artifical General Intelligence
AI	Artifical Intelligence
ARS	Artifical Recognition System
BDI	Belief, Desire and Intention
Cog-MAS	Cognitive Multi-Agent System
CPU	Central Processing Unit
DM	Drive Mesh
EBNF	Erweiterte Backus-Naur-Form
EOS	Evolutionären Objektorientierten Software-Entwicklung
GUI	Graphical User Interface
GWT	Global Workspace Theory
IDA	Intelligent Distribution Agent
KI	Künstliche Intelligenz
LIDA	Learning Intelligent Distribution Agent
MASON	Multi-Agent Simulator of Neighbourhoods
SiMA	Simulation of the Mental Apparatus & Applications
SOAR	State Operator Apply Result
SPARQL	SPARQL Protocol and RDF Query Language
SW	Software
TP	Thing Presentation Mesh
WM	Working Memory
WP	Word Presentation
WPM	Word Presentation Mesh
WPS	Wortvorstellungssequenz
XML	Extensible Markup Language

1. Einführung

Um die menschliche Arbeit zu unterstützen und zu erleichtern, werden seit jeher Techniken erfunden und weiterentwickelt. Der Wunsch, Roboter zu erschaffen, welche alleine komplexere Aufgaben erfüllen können wird immer stärker. Einen Roboter-Agenten zu erschaffen, der ohne Hilfe in einer unbekannten Umgebung fungieren und Aufgaben erfüllen kann, stellt eine große Herausforderung dar. Die Herausforderung besteht nicht nur in der großen Anzahl an Datenpunkten, die von den Sensoren erfasst und behandelt werden sollen, sondern auch darin einen Roboter in die Lage zu versetzen selbstständig Aufgaben zu erledigen und situationsbezogen zu reagieren. Zur Erschaffung eines Roboters bedarf es nicht nur mechanischer Kenntnisse, sondern muss auch eine Art von Intelligenz, eine so genannte künstliche Intelligenz (KI), entwickelt werden. Eine solche KI wird oft in Form eines Software-Agenten entwickelt. Dabei handelt es sich um ein Programm, das ohne physischen Körper in einer Simulationsumgebung fungiert. Der zu entwickelnde Agent muss die Fähigkeit besitzen, seine Umgebung wahrzunehmen. Er soll, als Endziel, aufgrund der wahrgenommenen Situation menschenähnliche Entscheidungen selbstständig treffen und mit seiner Umgebung interagieren können.

Diese Diplomarbeit beschäftigt sich mit dem Abschnitt der Entscheidungsfindung im Prozess zur Entwicklung des angesprochenen Agenten. Es gibt verschiedene Ansätze (eine Auswahl von Architekturen mit verschiedenen Ansätzen wird im [Kapitel 2.3] behandelt), die versuchen das Problem zu lösen, wie ein Agent seine Entscheidungen trifft und in einer unbekannten Umgebung interagiert, ohne sich und seine Umgebung zu verletzen:

Die Herangehensweise, mit der sich die vorliegende Diplomarbeit befassen wird, sind kognitive Architekturen [Kapitel 2.1.3]. Es existieren verschiedene kognitive Architekturen, welche aber nicht ein konstantes und einheitliches kognitives Modell befolgen. Als bekannte Beispiele kann man hierfür SOAR [LNR87] und ACT-R [ABB+04] nennen. Im Rahmen ihrer gemeinsamen Funktionsweise stellen sie formale Modelle von kognitiven Prozessen dar. Kognitive Architekturen werden in Software- oder Hardware-Agenten (wie z. B. Roboter) implementiert, die in der Folge als kognitive Agenten bezeichnet werden.

Als Gegensatz zu diesen Ansätzen ist das „*Simulation of Mental Aparatus and Application*“ (SiMA) [BDD+15] zu nennen, das einem einheitlichen kognitiven Modell folgt, um eine kognitive Architektur zu konstruieren. Es ist ein interdisziplinäres Projekt, bei dem Techniker und Psychoanalytiker ein implementierbares Modell einer Entscheidungsfindungseinheit entwickeln, das auf dem psychischen Apparat des Menschen beruht. Dafür wurden bionische Ansätze, Methoden und Muster der menschlichen Wahrnehmung evaluiert und in einer künstlichen Lebenssimulation getestet. Für die Implementierung wurden Prinzipien der Neuropsychoanalyse angewendet. Eines der Prinzipien ist die Unterscheidung

zwischen dem bewussten- und unbewussten Vorgang, im Kontext der Psychoanalyse als Primär- und Sekundärprozess genannt. Damit dieses Modell für eine technische Umsetzung genutzt werden kann, müssen die Bedingungen mit Hilfe der Psychoanalyse in einer technischen Sprache übersetzt werden. Diese wird im SiMA-Projekt der Technischen Universität Wien von einem Team, das aus Ingenieuren und Psychoanalytikern besteht, getan.

Zwei wichtige Aspekte, die im Zuge des SiMA Projekts in einem Agenten umgesetzt werden sollen, sind soziale Regeln und Abwehrmechanismen. Diese bilden gemeinsam einen Filter-Mechanismus für die Sensordaten des Agenten und sind Gegenstand der vorliegenden Arbeit [GBD+11].

1.1 Motivation

Bei der Konstruktion einer kognitiven Architektur, ist der menschliche Geist die beste Inspirationsquelle [Scha12, p.1], da er das einzige uns bekannte, voll funktionsfähige kognitive System ist. Aufgrund der Komplexität des menschlichen Geistes wäre es unrealistisch, anzunehmen, als Techniker dessen Funktionsweise vollständig erschließen und vor allem konstruktiv umsetzen zu können, weshalb das SiMA Modell sich neuer und bewährter Modelle der menschlichen Entscheidungsfindung als Grundlage bedient. So versucht SiMA beispielsweise das Konzept von „Ich“, „Es“ und „Über-Ich“ nach Sigmund Freuds zweitem Topischen Modell nachzubilden. Dabei ist das „Es“ für die menschlichen Antriebe zuständig, das „Über-Ich“ für die Verbote sowie Einschränkungen und das „Ich“ für die Verbindung zur Außenwelt sowie die Integration der Anforderungen von „Es“ und „Über-Ich“ [Deut11, pp.71-73].

Ähnliche Konzepte existieren in klassischen KI-Systemen bereits in Form von Produktionsregeln. Diese Produktionsregeln würden dem „Über-Ich“ entsprechen. Die zusätzlichen Ansprüche des „Es“ und die Vermittlung durch das „Ich“ macht die SiMA Herangehensweise gegenüber von Produktionsregeln überlegen, weil sich dadurch vielfältigere und komplexere Entscheidungsmöglichkeiten für den Agenten ergeben. Produktionsregeln sind linear, das heißt, eine Bedingung führt zu einem Ergebnis. Beim SiMA-Ansatz mit „Über-Ich“-Regeln entsteht die Entscheidung hingegen als Reaktion auf eine Kombination aus verschiedenen Ansprüchen und Gegebenheiten. Die „Über-Ich“-Regeln können allgemeiner gehalten werden als es in Produktionssystemen üblich wäre, weil neben den „Über-Ich“-Regeln auch das „Ich“ existiert, dass die „Über-Ich“-Regeln mit dem „Es“ in Einklang bringt. Die Über-Ich Regeln können somit auf einer abstrakteren Ebene definiert werden. Im Gegensatz zu den Produktionsregeln die z.B. sagen „Wenn Stimulus A, dann Aktion B“ lassen die Über-Ich Regeln auch die Formulierung komplexerer proaktiver Zusammenhänge wie z.B. „mache jeden Tag eine gute Tat“ zu. Aus diesem Beispiel ergibt sich, dass sich mit solchen abstrakten Regeln mehr Situationen mit weniger Regeln abdecken lassen, als mit Produktionsregeln, was erlauben würde mit einer geringeren Anzahl an Über-Ich Regeln einen robusteren und flexibleren Agenten zu entwickeln, – als in einem Produktionssystem. So gesehen bildet das SiMA-System nicht nur eine Infrastruktur um das Produktionssystem herum, die es erlaubt, die gesetzten Regeln abstrakt zu halten, sondern kann der Rest des SiMA Systems, z.B. die assoziative Wahrnehmung, die Triebe, die Realitätsprüfung, die Planung, usw. [siehe das Kapitel „Ein Zyklus im Rahmen des SiMA-Modells“] die Implementierung der Regeln im aktuellen Kontext vornehmen.

Die Motivation meiner Arbeit ist es, das SiMA Projekt einen Schritt menschenähnlicher zu machen.

1.2 Problemdefinition

Der SiMA-Agent wird regelmäßig nur über eine kurze Zeitspanne in Betrieb genommen. Dh, dass die Daten, die sonst bei einem Menschen in der Kindheit und über viele Jahre in Form von Erfahrungen gelernt werden, hier von den Entwicklern angefertigt und dem SiMA-Agenten vorgegeben werden müssen. Problematisch ist dabei, dass dem Entwickler als Techniker in der Regel das psychologische Hintergrundwissen fehlt, um diese Daten anlegen zu können, während der Psychoanalytiker, in dessen fachlichen Anwendungsbereich die psychologischen Wissensinhalte fallen, wenig zur technischen Implementierung beitragen kann. Es ist also notwendig, eine Schnittstelle zu schaffen, die es Nicht-Technikern erlaubt, die notwendigen Informationen in das SiMA System einzubringen. Dazu zählen eben auch die Über-Ich Regeln, die beim menschlichen Vorbild, meist in der Kindheit gelernt wurden. Eine weitere Herausforderung des SiMA-Projektes liegt darin, dass die Abläufe, die durch die eingepflegte Information ausgelöst werden, von technischen und nicht-technischen Experten evaluiert werden müssen.

Die zentrale Aufgabe dieser Diplomarbeit ist die Erweiterung der existierenden Abwehr [siehe Kapitel 3.2.1] um ein regelbasiertes Kontrollsystem, das den psychoanalytischen Anforderungen genügt und mit dem Rest des SiMA Modells kompatibel ist. Das beinhaltet die Erweiterung der existierenden Abwehr um ein Modell der Regel-Repräsentation, deren Einspeisung im System und deren Anwendung auf die existierenden Daten. Darüber hinaus ist für die Evaluierung der Abwehr die Entwicklung neuer Visualisierungen erforderlich. Um zusätzliche Aktionen ausführen zu können, die für die Abwehr relevant sind, muss auch der simulierte Körper um ein Inventarsystem erweitert werden. Die Funktionalität des Inventar-Systems soll auch grafisch validiert werden können.

1.3 Task Setting

Die Herausforderung besteht darin, die eben erwähnten Aufgabenstellungen im SiMA Modell umzusetzen, wobei besonders auf die Anforderungen der anderen aktiven Entwickler am genannten Modell Rücksicht genommen werden muss. Das SiMA Modell hat nämlich aufgrund seiner Komplexität starke Abhängigkeiten zwischen den einzelnen Komponenten, was die Integration neuer Komponenten in Abstimmung mit den Arbeiten anderer Entwickler besonders herausfordernd macht. Aus diesem Grund besteht die erste Aufgabe darin, ein Entwicklungsmodell zu wählen, das eine solche Arbeitsweise unterstützt. Für diese Diplomarbeit wurde deshalb das so genannte Spiralmodell gewählt [siehe Kapitel 1.4].

Die nächste Herausforderung besteht darin, eine Regelrepräsentation zu entwickeln, die dem bionischen Vorbild Menschen entspricht, aber trotzdem mit vertretbarem Aufwand auf die existierenden Daten angewendet werden kann. Hierfür wird es zuerst notwendig sein, die existierenden Datenflüsse und ihre Schnittstellen zu analysieren, um entsprechende Ansatzpunkte für einen Filtermechanismus,

der diese Regeln verarbeiten kann, heraus zu kristallisieren. Der Fokus wird dabei voraussichtlich auf der Analyse der Abwehr und ihrer internen Abläufe liegen.

Als nächstes wird es notwendig sein, Möglichkeiten zur Evaluierung der Erweiterungen zu entwerfen und umzusetzen. Dafür müssen zuerst die existierenden Visualisierungen analysiert werden, um bestimmen zu können, wie diese am Sinnvollsten erweitert werden können, um ein umfassendes Bild der internen Abläufe zu bekommen.

Um die Entwicklung tatsächlich zu evaluieren, müssen schließlich die neuen Visualisierungen im Kontext neuer Szenarien, die entworfen werden, um die Funktionsweise der neuen Filterregeln zu zeigen, eingesetzt werden. Dies wird die Erweiterung des Agenten um einige neue Aktionen und eine Inventarfunktionalität notwendig machen, wobei die genannte Erweiterung wiederum die Notwendigkeit neuer Visualisierungen nach sich zieht.

1.4 Methode

Aus der bisherigen Darstellung geht bereits hervor, dass die Komplexität des SiMA-Modells, die Zusammenarbeit mit den anderen Experten und die Integration in SiMAs-Top-Down Entwicklungsansatz besondere Herausforderungen darstellen. Aus diesen Gründen wird – wie bereits in [Kapitel 1.3] erwähnt, für den Entwicklungsprozess das abgewandelte Spiralmodell eingesetzt. Das Spiralmodell ist ein Hybridmodell, das von B. Boehm ursprünglich 1986 entwickelt wurde [Sarf03, pp.12-13]. Für die Zwecke dieser Arbeit wird der im ursprünglichen Modell spezifizierte Schritt der Risikoanalyse durch das Abklären der aktuellen Fortschritte mit den anderen wissenschaftlichen Mitgliedern des SiMA-Projektes ersetzt.

Abbildung 1.1 zeigt die Phasen der „Evolutionären, objektorientierten Software-Entwicklung“ (EOS), die in den Iterationen des Spiralmodells zur Erreichung der nächsten Entwicklungsstufe abgehandelt werden. Das EOS stellt eine Weiterentwicklung der klassischen Software-Entwicklungsmodelle (SW-Entwicklungsmodelle) dar, das die Lücke zwischen Theorie und Praxis in der SW-Entwicklung überbrücken sollte, indem es den evolutionären Charakter der SW-Entwicklung im gesamten Modell abbildet [Sarf03, p.24].

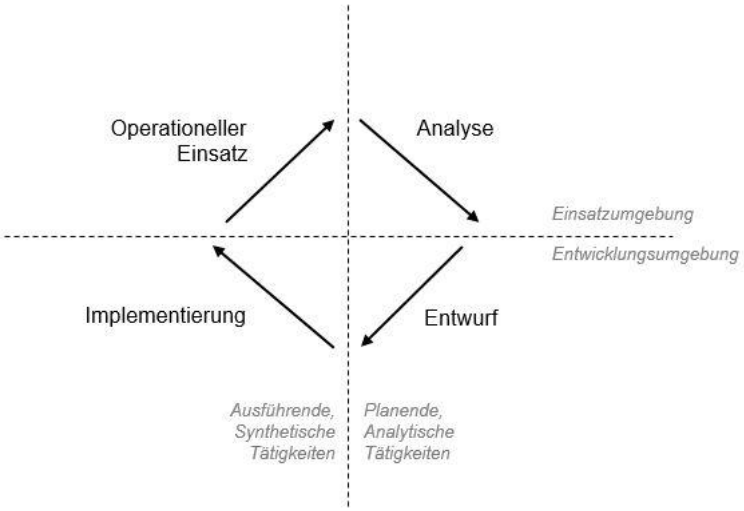


Abbildung 1.1: Phasen eines Entwicklungszyklus [Sarf03, p24], nachgezeichnet

In Folge werden nun die geplanten Inhalte der Phasen des abgeänderten Spiralmodells Abbildung 1.2 erklärt.

In der **Planungsphase** sollen die Anforderungen an die jeweiligen Modulerweiterungen festgelegt werden, die dann in der Phase **Verifikation und Validation** mit den anderen am SiMA arbeitenden Experten abgeklärt werden, um Konflikte oder Synergien frühzeitig erkennen zu können. Die voraussichtliche Herausforderung dieser Phase wird angesichts der hohen Komplexität des Modells das Erarbeiten der SiMA-Inhalte und -Abläufe sein. Als Nächstes werden im Zuge eines **Grobentwurfs** die Umsetzungsmöglichkeiten für die einzelnen Aufgabenpakete entwickelt, die dann nochmals in einer weiteren Verifikation und Validierungsphase erneut mit dem SiMA-Team anhand von beispielhaften Abläufen abgeklärt werden. In der **Testplanung** soll der Test für die neuen Blöcke in den SiMA-Testplan integriert werden. In dieser Phase wird voraussichtlich eine starke Koordination mit den als Psychoanalytiker beim SiMA-Projekt tätigen Kollegen stattfinden müssen. Im Zuge des **Feinentwurfes** soll der Grobentwurf basierend auf dem vorherigen Feedback von dritter Seite konkretisiert und um Schnittstellen zum SiMA-Modell erweitert werden. Ein geplantes Artefakt dieser Phase sind erste – nicht lauffähige – Prototypen, die zur erneuten Abklärung mit dem SiMA-Team herangezogen werden sollen. Daraufhin kann mit der **Implementierung und Integration** begonnen werden, die abschließend in der **Testphase** anhand von Use-Cases validiert werden.

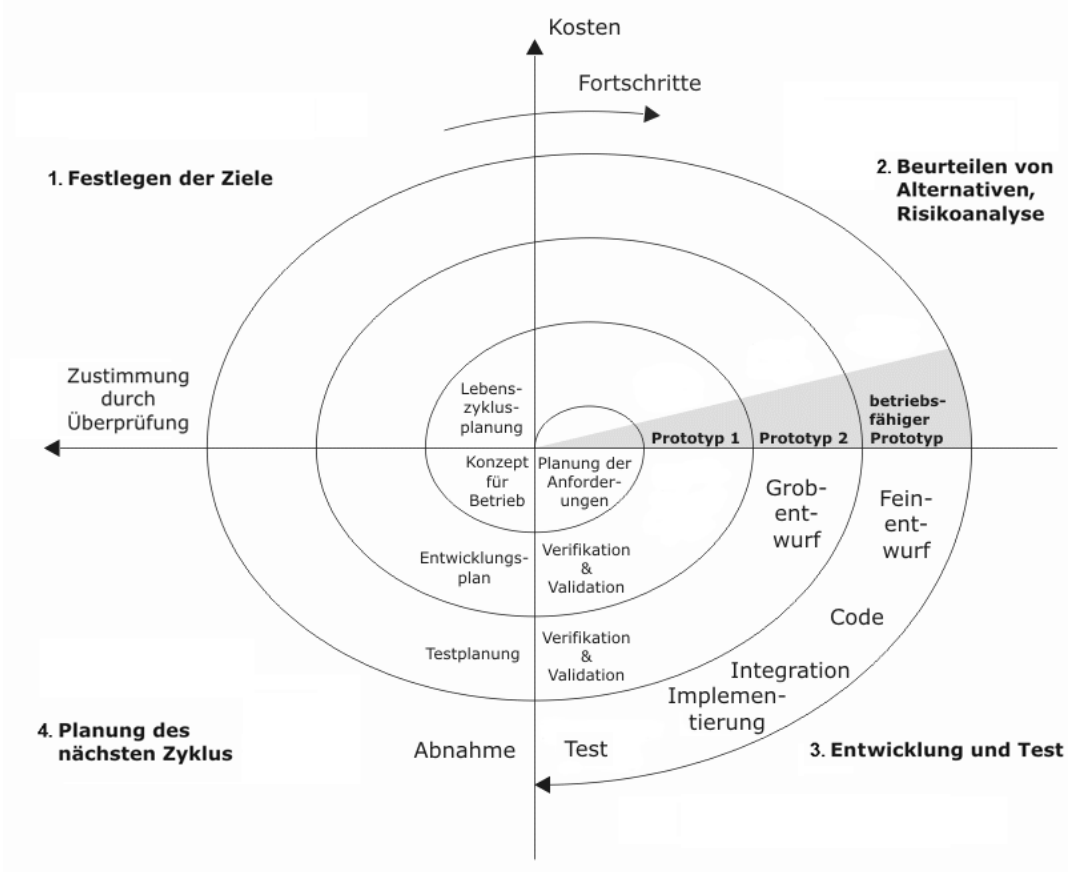


Abbildung 1.2: Spiralmodell nach Boehm, abgeändert nachgezeichnet

2. State of the Art and Related Work

Nach einer Einführung in das Thema dieser Arbeit folgt in diesem Kapitel ein Überblick über den Stand der Technik im Bereich *cognitive artificial general intelligence* (cognitive AGI). Grundlage dieser Diplomarbeit bildet – wie in der Einleitung erwähnt – das kognitive SiMA-Modell.

Das Ziel dieses Kapitels ist es, zum einen andere kognitive Modelle zu beschreiben, die genauso wie das SiMA-Modell Mechanismen zur autonomen Konfliktlösung beinhalten, zum anderen einen Überblick über den aktuellen Stand des SiMA Modells und den aktuellen Stand der Wissenschaft in Bereich kognitive Architekturen zu liefern.

Da kognitive Architekturen auf kognitiven Modellen basieren, die wiederum die Existenz einer AGI voraussetzen, ist es vor der erwähnten inhaltlichen Auseinandersetzung mit den einzelnen kognitiven Architekturen notwendig, die Definition und den Unterschied zwischen AI und AGI herauszuarbeiten sowie sich mit der Natur kognitiver Architekturen auseinandersetzen.

Daran anschließend wird das SiMA-Modell vorgestellt und detailliert beschrieben. Anschließend werde ich andere kognitive Architekturen darstellen und dabei ihre Vor- und Nachteile gegenüber dem SiMA-Modell herausarbeiten.

2.1 AI, AGI und kognitive Architekturen

In diesem Kapitel werden die grundlegenden Begriffe und Konzepte eingeführt und erklärt welche für das Verständnis für das im folgenden Kapitel notwendig sind. Zuerst wird der Unterschied zwischen der generellen künstlichen Intelligenz und er allgemeinen künstlichen Intelligenz beschrieben. Danach wird das für diese Arbeit relevante Konzept AGI näher erklärt und letztlich die Grundlagen einer kognitiven Architektur beschrieben.

2.1.1 Unterschiede zwischen AI und AGI

AI steht für *artificial intelligence*, übersetzt künstliche Intelligenz – dabei ist Intelligenz nicht genau definiert. Die künstliche Intelligenz (KI) stellt die Automatisierung intelligenten Verhaltens dar. Wenn man von einer AI spricht sind normalerweise Lösungen von spezifischen Arbeiten gemeint.
Der Unterschied zwischen einer AI und einer AGI besteht hauptsächlich in der angezielten Anwendungsdomäne. AI versucht Intelligenz zu einer spezifischen Aufgabe zu lösen z.B. Schachspiel, eine AGI versucht domänenübergreifende Konzept zur Entwicklung zu entwickeln.

AGI steht demgegenüber für *artifical general intelligence*. Diese ist eine Erweiterung der AI und kann zusätzlich zu den Fähigkeiten der AI verschiedene komplexe Probleme lösen und sich an verschiedene Anforderungen anpassen: Im Unterschied zur AI, die bloß für eine Anwendung programmiert wird, wird die AGI nicht zur Lösung einer vorgegebenen Aufgabe, sondern zur selbstständigen Bewältigung einer vielschichtigen Problemstellung eingesetzt. Daher wird vorausgesetzt, dass sich AGI-Programme selbst kontrollieren können, demnach autonom sind und eigene Gedanken, Sorgen, Gefühle, Stärken, udgl. aufweisen [PG07, p.1]. Jede wirkliche AGI braucht die Fähigkeit, Konflikte mit ihrem Wissenstand erkennen und selbständig lösen zu können [GOS14, p.33]. In vorliegender Diplomarbeit werden nur Projekte beschrieben, die als AGI zu qualifizieren sind.

2.1.2 Merkmale einer AGI

Im Sinne der eben beschriebenen Unterscheidung erfordert die Qualifikation eines Programmes als AGI Projekt – im Gegensatz zu z.B. AI-Projekten oder AGI-Konzepten – kumulativ folgende Kriterien [WG06, pp.2-3]:

- AGI-Projekte brauchen eine Theorie von „Intelligenz" und nicht eine ausprogrammierte sachbezogene Intelligenz, wie sie etwa in AI-Programmen integriert ist;
- AGI-Projekte benötigen einen technischen Plan für die Implementierung der angewendeten Theorien; und
- Ergebnisse von AGI-Projekten müssen publiziert und evaluierbar sein.

2.1.3 Beschreibung kognitiver Architekturen

Kognitive Architekturen erfüllen alle unter Punkt 2.1.2 herausgearbeiteten Merkmale einer AGI. Das Ziel einer kognitiven Architektur ist der Versuch, den menschlichen Erkenntnisprozess darzustellen. Ein Erkenntnisprozess sollte die Möglichkeiten bieten, das Wissen zu repräsentieren, relevantes Wissen abzuspeichern und dieses zu verarbeiten. [LC06, p.1469]

Kognitive Architekturen können symbolisch, konnektionistisch und hybrid ausgestaltet sein.

Bei der symbolischen kognitiven Architektur wird Objekten aus der Wahrnehmung ein konkretes Symbol zugeordnet. Im Zusammenhang mit der konnektionistischen kognitiven Architektur bekommt der Nutzer eine Vielzahl an Sensor-Daten über das Untersuchungsobjekt und ist dieses dadurch kein Symbol, sondern eine Summe aus Sensorwerten. Die hybride kognitive Architektur ist die die Kombination aus den ersten beiden Architekturvarianten.

Basierend auf diesen Anforderungen, können bei [Lair08, p.2] folgende Punkte identifiziert werden, welche für eine kognitive Architektur erforderlich sind:

- eine Speichermöglichkeit für Wissen;
- eine Verarbeitungseinheit zum Extrahieren, Auswählen, Kombinieren und Abspeichern von Wissen;
- die Definition eines Formates (oder einer Sprache), in dem das Wissen abgespeichert und verarbeitet wird.

Kognitive Architekturen unterscheiden zwischen dem Wissen, das während der Ausführung angesammelt wird, und grundlegendem Wissen, das allen Aufgaben zu Grunde liegt. Diese Unterscheidung beim Entwurf der kognitiven Architektur zu berücksichtigen, ist bei [Lair08, p.2] als besondere Herausforderung hervorgehoben. Die Schwierigkeit dabei liegt darin, die Balance zwischen einem stabilen System, das das existierende Wissen anwenden kann, um ein Problem zu lösen, und der Flexibilität, die notwendig ist, damit der Agent sich anpassen und neue Lösungswege erlernen kann, zu finden. Um dies zu erreichen, brauchen kognitive Architekturen ordentliche Hypothesen über die Beschaffenheit des kognitiven Prozesses (dies ist der stabile Teil) und darüber, wie zusätzliches Wissen erlangt, abgespeichert und verarbeitet wird.

2.2 Darstellung des SiMA-Modells und Ableitung der Aufgabenstellung dieser Diplomarbeit

SiMA verkörpert gemessen an den unter Punkt 2.1.2 dargestellten Anforderungen eine AGI, es kann als kognitive Architektur verstanden werden: Zum einen besitzt es eine Theorie der „Intelligenz“ basierend auf psychoanalytischen Theorien, zum anderen wurde das SiMA-Modell in verschiedenen Projekten bereits implementiert. Weiteres wurden die Ergebnisse dieser Projekte in verschiedenen Papers publiziert [5].

Im Rahmen der Betrachtung des SiMA-Modells werden in den folgenden Unterkapitel neben der Einführung in das SiMA-Projekt, im Unterpunkt 2.2.1, auch dessen Anfänge, im Unterpunkt 2.2.2, und das aktuelle SiMA-Modell dargestellt, Unterpunkt 2.2.3. Ferner werden die Datenflüsse vom Eingang bis zum Ausgang beschrieben. Die Simulationsumgebung, in die das Modell eingebettet ist, wird im Unterpunkt Punkt 2.2.4 beschrieben, sowie ein Use-Case im Unterpunkt 2.2.5 dargestellt.

2.2.1 Einführung in das SiMA-Modell

Das SiMA (*Simulation of the Mental Apparatus & Applications*; dazu Dietrich et alt. [DBZ+09]) ist ein interdisziplinäres Projekt bei dem Informatiker zusammen mit Psychoanalytikern ein Computermodell entwickeln, das auf dem psychischen Apparat des Menschen beruht. Als Grundlage hierfür wurde Freuds zweites topisches Modell herangezogen. Der Grund dafür ist, dass dieses Modell eine holistische funktionale Beschreibung des menschlichen Entscheidungsprozesses liefert; d.h. mit anderen Worten, dass die gesamte menschliche Entscheidungsfindung erklärt wird und somit gut in funktionale Einheiten zerlegbar ist, wobei beim Modell Freuds keine Bereiche ausgelassen werden. Das besagte Modell erklärt nicht nur den gesamten Entscheidungsfindungsprozess, sondern es ist in sich auch schlüssig und bildet die Grundlage für andere, in der psychoanalytischen Praxis angewendete Modelle.

Im SiMA-Projekt wird besonderes Augenmerk darauf gelegt, dass das entwickelte System in seinen Reaktionen dem menschlichen Vorbild nahe kommt. Diesem bionischen Ansatz zu folgen, bedeutet die Funktion der menschlichen Psyche vollständig, und unabhängig von einzelnen Aufgabestellungen, abzubilden. Deshalb handelt es sich bei SiMA – wie bereits im Zuge der Einleitung in dieses Kapital festgestellt – um ein AGI und keine einfache AI. Ein weiter Effekt ist, dass Funktionen unabhängig

von ihren Nutzen übertragen werden, sprich auch Funktionen, die bei der menschlichen Entscheidungsfindung Probleme verursachen, werden mit übernommen. Um das zu bewerkstelligen, wurde bei der Modellentwicklung ein Top-down-Ansatz verfolgt Abbildung 2.1, bei dem die psychischen Funktionen erst grob und dann schrittweise (*breath first* – d.h. erst nachdem alle Funktionen auf der gleichen Ebenen ausreichend beschrieben wurden) weiter konkretisiert werden.

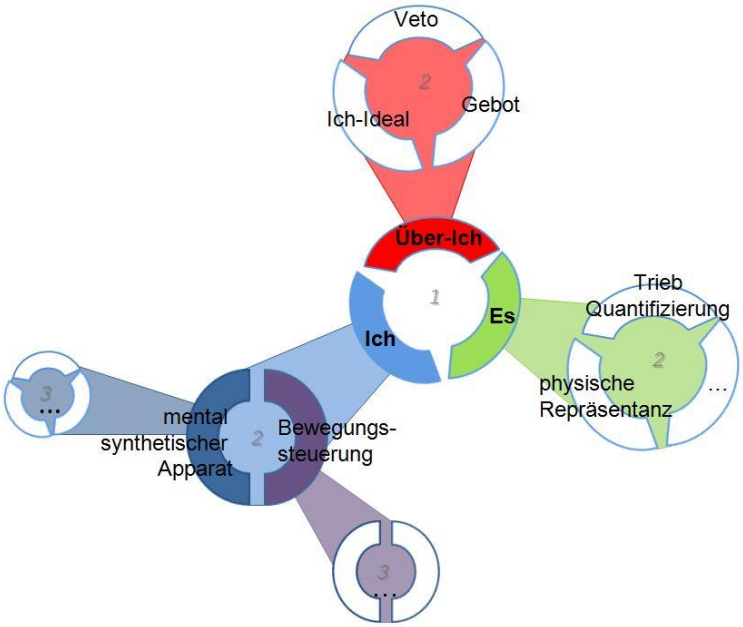


Abbildung 2.1: Top Down Ansatz

Im Ergebnis lässt sich SiMA als ein holistisches, bionisches und funktionales Modell der menschlichen Entscheidungsfindung beschreiben. Während der Modellierung wird bereits auf die spätere Verifizierbarkeit geachtet, indem die Modellierung „Use-Case"-gesteuert erfolgt [BGSW13, pp.3-6]. Das bedeutet, dass der definierte „Use-Case" die Anforderungen an das Modell festlegt.

Entwickelt und getestet wird das Projekt aktuell als ein in Java geschriebener, kognitiver Agent in der Simulationsumgebung MASON [3].

2.2.2 Anfänge

Das SiMA-Projekt startete im Jahre 1999. Zu dem Zeitpunkt wurde es unter dem Namen ARS – *artificial recognition system* – geführt und ab dem 11.02.2015 unter dem Namen SiMA weiterbetrieben. Das Projekt entstand im Kontext der Gebäudeautomatisierung als eine Idee: Das Problem, das SiMA zugrunde lag, war die steigende Sensoranzahl in einem Gebäude und dadurch auch die erhaltene Datenvielfalt. Klassische AI Projekte waren nicht imstande, mit derart vielen Sensoren umzugehen, sodass dringender Bedarf nach einer neuen Lösung bestand.

Eine der ersten Installationen von SiMA war in einer wahrnehmungsfähigen Küche (Smart Kitchen [Much13, p59]). Diese war in der Lage, aus Sensordaten Szenarien zu erkennen und sie zuzuordnen.

Beispielsweise konnte die künstliche Intelligenz im Falle einer heißen Herdplatte erkennen, ob sich ein Besucher eine Tasse Kaffee zubereitete oder ob ein Kind der heißen Herdplatte zu nahe kam.

Um eine Vielzahl solcher Anwendungsfälle bewältigen zu können, entstand der Gedanke als Lösungsweg den menschlichen Entscheidungsapparat auf Basis der psychoanalytischen Theorien nachzubauen [PPDB05, p.259 und PP05, p.5].

Im Jahr 2009 wurde das SiMA-Modell erstmalig in seiner jetzigen Form entworfen [DFZB09, p178]. Ein früheres Konzept eines psychoanalytischen Wahrnehmungsprozesses ist in Abbildung 2.2 skizziert, angelehnt an Palensky et al. Für detaillierte Beschreibung siehe [DFZB09, p199-201]

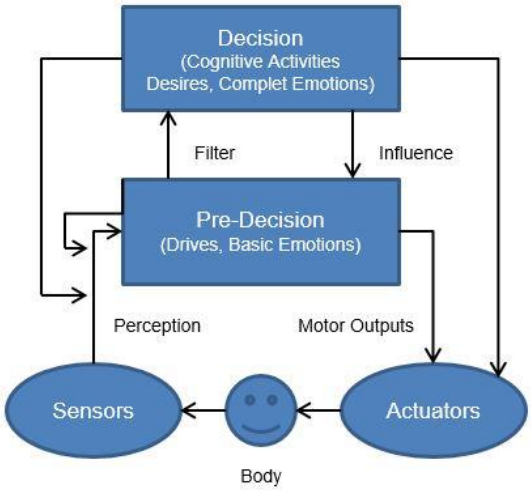


Abbildung 2.2: Eine alte SiMA Modellübersicht [Much13, p.61] modifiziert nachgezeichnet

In den darauffolgenden Jahren wurde das SiMA-Modell von verschiedenen Wissenschaftlern mit- und weiterentwickelt. Im Jahre 2010-2011 verbesserten [Lang10] und [Deut11] die funktionale Beschreibung des Modells, einem strikten Top Down Ansatz folgend, anhand der Konzepte der Metapsychologie: In [Lang10, pp.50-64] wird erstmalig die Entscheidungsfindungseinheit (*Decision Unit*) für einen autonomen Agenten benutzt, wohingegen bei [Zeil10, pp.80-88] der Fokus auf der Informationsrepräsentations-Schicht (*Information Representation Layer*) liegt. [Deut11, pp.79-85] erweitert das Triebkonzept und die Entscheidungsfindungseinheit.

Die erste Implementierung in einer artificial live Simulation wurde von Deutsch et al. [DZL07 pp.995-999] entwickelt.

2.2.3 Inhaltliche Beschreibung des SiMA-Modells

Schichtenmodell

Als Ergebnis des eben kurz umrissenen Entwicklungsprozesses entstand ein funktionales Modell, das die Funktionen, also das „Warum“, und nicht das Verhalten, also das „Was“ beschreibt. Das eben erwähnte funktionale Modell ist als Schichtenmodell aufgebaut [Much13, p.63], Abbildung 2.3: Schicht – oder „Layer“ – 1 ist die neuronale Schicht, in der Informationen aus einer Kombination aus Sensordaten existieren. In Schicht 2, der neurosymbolischen Schicht, werden aus diesen Sensordaten

Neurosymbole gebildet, die in Schicht 3, der Psyche, verarbeitet werden (der Vollständigkeit halber sei erwähnt, dass die Umwelt als Schicht 0 verstanden wird). Das Schichtmodell lässt sich grafisch folgendermaßen darstellen:

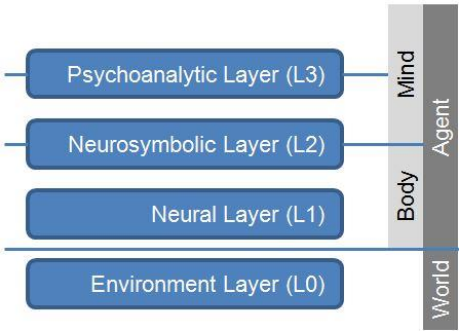


Abbildung 2.3: Das SiMA Schichtenmodell, [Much13, p.63] nachgezeichnet

In dem aktuellen SiMA Modell bestehen Schicht 1 (L1) und Schicht 2 (L2) nur aus Interfaces. In der Simulation werden sie abstrahiert, indem die Umgebungsdaten direkt auf Symbole gemapt werden. Mit anderen Worten: der Programmierer weiß, welches Objekt im Simulator z.B. ein Apfel ist und verknüpft es mit der entsprechenden Information. Trotz dieser Lücken der Implementierung kann SiMA als eine hybride kognitive Architektur klassifiziert werden, da die Schnittstellen zur subsymbolischen Schicht klar definiert sind.

Entscheidungsfindungseinheit

Die Entscheidungsfindungseinheit (*Decision Unit*) wird in Abbildung 2.4 dargestellt. Sie stellt den Informationsfluss zwischen äußerer und innerer Welt des psychischen Apparates in technischer Hinsicht dar. Der psychische Apparat hat die Aufgabe zwischen drei Hauptinstanzen (Es, Ich und Über-Ich) und den Anforderungen, die von ihnen kommen, zu vermitteln.

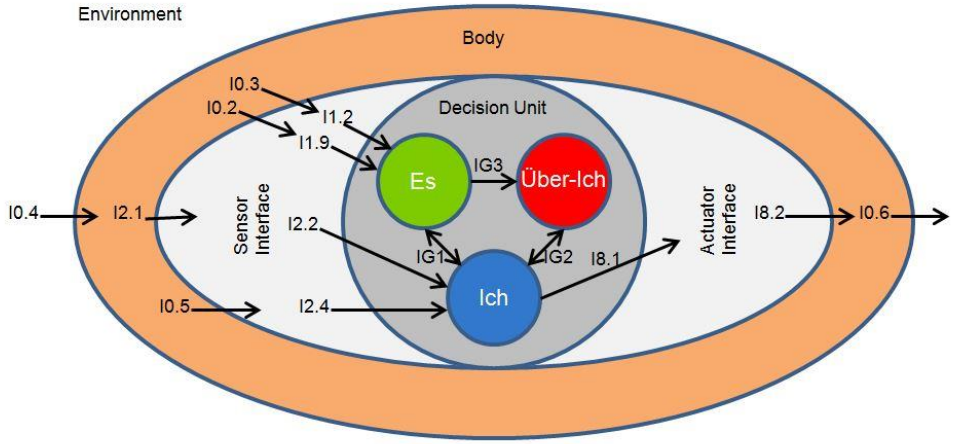


Abbildung 2.4: Entscheidungsfindungseinheit [Much13, p.64] nachgezeichnet

Die Entscheidungsfindungseinheit empfängt Informationen aus der Umgebung durch die Interfaces I0.4 sowie I2.1 und aus dem Körper durch die Interfaces I0.2, I0.3 sowie I0.5. Die Interfaces I1.2, I1.9, I2.2 und I2.4 zeigen den Informationsfluss von der äußeren zur inneren Welt, bestimmt durch die Sensoren. Das Interface I8.1 zeigt den Informationsfluss von der inneren zur äußeren Welt und mündet in die Aktuatoren. Die Interfaces I8.2 und I0.6 stellen die Verbindung zwischen der inneren Welt und der Umgebung zur Handlungsausführung dar.

Die erste Instanz innerhalb der Entscheidungsfindungseinheit, das ES, generiert die Trieb-Forderungen ausgelöst durch körperlichen Bedürfnisse. Dementsprechend ist das ES die psychische Einheit, die unbewussten Daten, wie etwa verdrängte Inhalte oder körperliche Bedürfnisse, enthält. Die Bedürfnisse werden in Form von Triebrepräsentanzen und Affektbeträgen repräsentiert, wobei die einschlägig beteiligten Funktionen operieren anhand des Lustprinzips, welches die unmittelbare Befriedigung aller Bedürfnisse verlangt. Das ES befindet sich im Primärprozess und beschäftigt sich mit unbewussten Informationen. Detaillierte Informationen darüber lassen sich in [Deut11, pp.79-85] nachlesen.

Die zweite Instanz innerhalb der Entscheidungsfindungseinheit, das ICH, ist verantwortlich für die Realitätsanforderung. Sie kombiniert dafür das Wissen über die Realität, dessen Möglichkeiten, Einschränkungen und die Auswirkung der eigenen Handlungen. In diesem Sinn vermittelt das ICH zwischen den Ansprüchen des ES, des ÜBER-ICH und der Realität. Es funktioniert nach dem Realitätsprinzip, das als Gegensatz zum Lustprinzip des ES angesehen werden kann. Die Inhalte werden entsprechend den sekundären Prozessprinzipien organisiert und haben somit die Möglichkeit, bewusst zu werden. Einige Funktionen des ICH sind Teil des Primärprozesses und deren Inhalte bleiben unbewusst. Die Funktionen des ICH mit funktionalen Unterteilungen werden im Detail in [Lang10, pp.66-76] diskutiert.

Die dritte Instanz innerhalb der Entscheidungsfindungseinheit repräsentiert das ÜBER-ICH. Dieses beinhaltet Forderungen sowohl von sozialen und kulturellen Regeln als auch von Annahmen. Das ÜBER-ICH ist somit verantwortlich für die Befolgung von sozialen und kulturellen Regeln und fungiert als Gegenspieler zum ES. Der Konflikt zwischen der Forderung des ÜBER-ICH und den Anforderungen der Triebe des ES löst Abwehrmechanismen aus. Diese können als ein Filter für unbewusste Informationen angesehen werden, die zur Anwendung gelangen, bevor Handlungsmaßnahmen ergriffen werden. Eine Diskussion über die Aspekte des ÜBER-ICH findet sich in [DTM+09, p376-382].

Die Schnittstellen IG1, IG2 und IG3 stellen den in [Zeil10, p.63] beschriebenen Informationsfluss dar.

Ein Zyklus im Rahmen des SiMA-Modells

Ein Zyklus wird im SiMA-Modell stets sequenziell abgehandelt, jeder Zyklus durchläuft alle Module in einer fixen Abfolge. Der Anlauf eines Zyklus wird in der in Abbildung 2.5 angeführten Grafik von links oben nach links unten dargestellt.

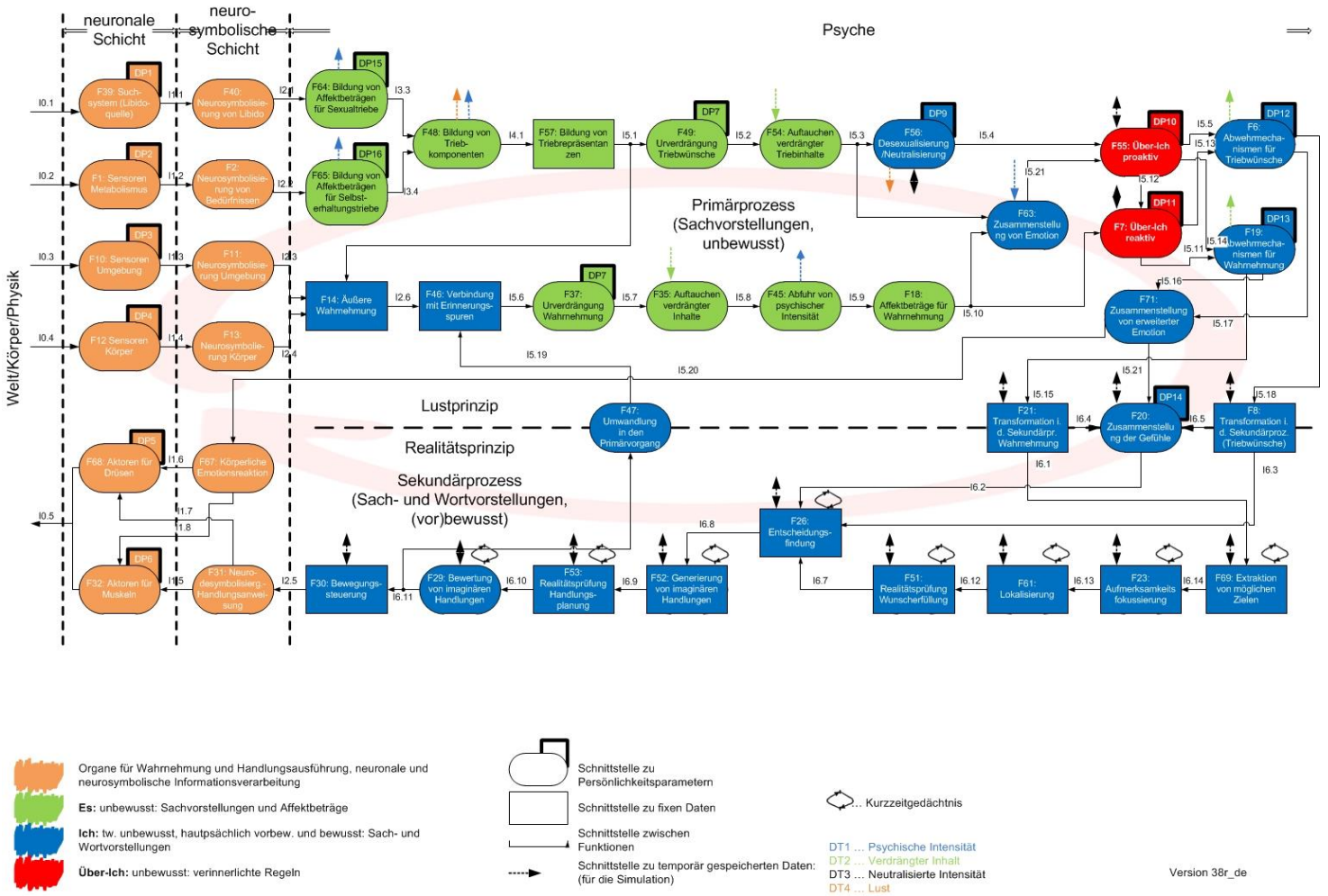


Abbildung 2.5: Das aktuelle SiMA-Modell in der Ebene 1 [DBD+14, p.81]

Um das SiMA-Modell übersichtlicher zu gestalten, ist es ratsam, den Datenfluss des Modells anhand der jeweiligen Schienen, die der Abbildung 2.6 zu Grunde liegen, zu betrachten. Für genauere Beschreibung der einzelnen Module siehe [DBD+14, pp.101-147].

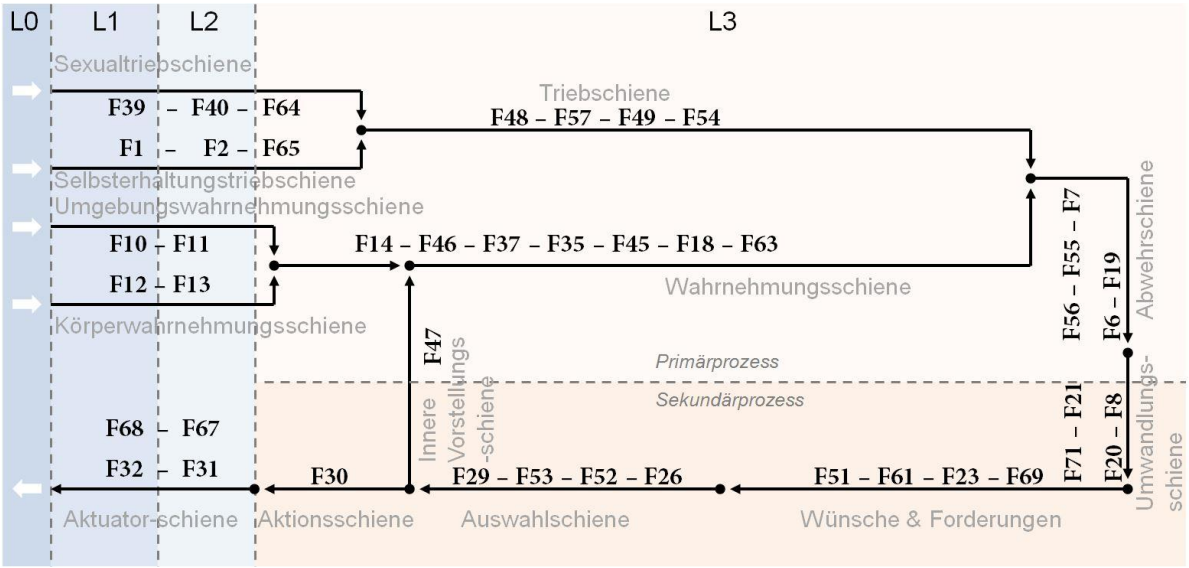


Abbildung 2.6: Das aktuelle SiMA Modell - vereinfacht

Zu Beginn eines Zyklus empfängt der Agent Daten zur aktuellen Trieblage über die *Sexualtriebschiene* und die *Selbsterhaltungstriebschiene*. Die erhaltenen Daten werden in der sogenannten *Triebschiene* zusammengeführt. Parallel dazu erhält der Agent Wahrnehmungen von der *Umgebungswahrnehmungsschiene* und der *Körperwahrnehmungsschiene*, welche er, angereichert mit phantasierten Inhalten durch die *Innere Vorstellungsschiene*, an die *Wahrnehmungsschiene* weiterleitet. Beide Schienen (die der Triebe und die der Wahrnehmung) werden anschließend in die *Abwehrschiene* gespeist, wo sie anhand von Abwehrmechanismen gefiltert werden. Die Abwehrschiene wiederum reicht ihre Daten an die *Umwandlungsschiene* weiter, wo sie für die Verwendung im Sekundärprozess vorbereitet werden. Ab dem Zeitpunkt kann von (vor-)bewussten Daten gesprochen werden. Diese Daten werden in der *Wünsche- und Forderungsschiene* mit Wünschen, Forderungen und Erfahrungen angereichert und daraufhin in die *Auswahlschiene* weitergeleitet. Die erwähnte Schiene wählt aus den aktuell erwogenen Handlungsoptionen eine aus und übergibt diese an die *Aktionsschiene*. Die Handlungsoptionen, die nicht ausgewählt wurden, werden als phantasierte Inhalte über die *Innere Vorstellungsschiene* in den unbewussten Prozess (Primärprozess) zurück geschickt damit sie unter Umständen in späteren Zyklen doch noch zu Handlungen führen können. Die Aktionsschiene gibt die gewählte Handlung nachfolgend an die *Aktuator Schiene* weiter, die sie in Befehle umwandelt, durch die der Agent mit seiner Umwelt interagiert.

An dieser Stelle scheint es angebracht, eine genauere Beschreibung der einzelnen Schienen vorzunehmen:

2.2.3.1.1 Sexualtriebschiene

Über die Sexualtriebschiene gelangen Triebe in die Entscheidungsfindungseinheit, die für die Erhaltung des Körpers nicht notwendig sind. Dabei darf die Sexualtriebschiene nicht mit dem Bedürfnis nach Reproduktion verwechselt werden, sondern handelt es sich dabei um körperliche Triebspannungen, die durch die Interaktion mit der Umwelt befriedigt werden können. Als Beispiel wäre hier etwa „Kaugummikauen" zu nennen: Das Kauen wirkt sich stimulierend auf die Mundschleimhaut aus, wodurch die Sexualtriebe stimuliert werden, wohingegen der Verzehr von Nahrung auf die Selbsterhaltungstriebe, nicht aber auf die Sexualtriebe, einwirkt.

Diese Schiene existiert in allen drei Layern und umfasst die drei Module F39 – „Suchsystem", F40 – „Neurosymbolisierung von Libido" und F64 – „Bildung von Affektbeträgen für Sexualtriebe". F39 kombiniert alle libido-bezogenen Informationen aus den verschiedenen körperlichen Quellen und erogenen Zonen und sendet sie an F40, wo sie in Neurosymbole umgewandelt werden. In F64 werden die erhaltenen Informationen schließlich in Partialtriebe aufgeteilt (oral, anal, phallisch, genital) und für jede Triebquelle wird eine libidinöse und eine aggressive Triebrepräsentanz gebildet.

2.2.3.1.2 Selbsterhaltungstriebschiene

Neben der Sexualtriebschiene existiert die Selbsterhaltungstriebschiene, über die Triebe, die für die Funktion des Körpers notwendig sind, wie Essen, Trinken, Entleeren von Darm und Blase, in die Entscheidungsfindungseinheit gelangen. An dieser Stelle sind die eben bezeichneten Triebe noch kein Plan, Wunsch oder Gedanke, sondern äußern sie sich in Spannungen, die im SiMA in Triebrepräsentanz, also das körperliche Gefühl nach einer Befriedigung des Triebes, umgewandelt werden.

Der Selbsterhaltungstrieb existiert in allen drei Layern (entsprechend Abbildung 2.3) und umfasst die drei Module F1 – „Sensoren Metabolismus", F2 – „Neurosymbolisierung von Bedürfnissen" und F65 – „Bildung von Affektbeträgen für Selbsterhaltungstriebe". Die Sensoren des Moduls F1 liefern Daten über den Zustand der körperlichen Organe, wie zB Information über Interbolismus, die Magenspannung, Blutdruck, Energieverbrauch, Magenfüllstand ect. Die erhaltenen Daten werden im Modul F2 in Neurosymbole umgewandelt, die dann in F65 dergestalt weiterverarbeitet werden können, dass für jedes körperliches Bedürfnis eine aggressive und libidiöse Triebrepräsentanz, die wiederum aus Triebobjekt, Triebquelle, Triebziel und Affektbetrag[1] (englisch Quota of Affect = QoA) zusammengesetzt sind, erstellt wird.

2.2.3.1.3 Umgebungswahrnehmungsschiene

[1] Ist eine psychoanalytische Größe, die in der Technik einer Bewertung entspricht.

Über die Umgebungswahrnehmungsschiene kommen Informationen über die Umwelt des Agenten in die Entscheidungsfindungseinheit. Hierbei handelt es sich nicht nur um visuelle Eindrücke, sondern auch um olfaktorische und sensorische Informationen. Als Beispiel wird auf der ersten Ebene die Farbe „rot“ wahrgenommen und weitergeleitet. In der zweiten Stufe wird ein Symbol „rot“ erstellt und in der dritten und letzten wird dieses Symbol mit einem Apfel assoziiert.

Die Umgebungswahrnehmungsschiene deckt im Unterschied zu den vorgenannten Schienen lediglich die Layer 1 und 2 ab. Sie besteht aus F10 – „Sensoren Umgebung“ und F11 – „Neurosymbolisierung Umgebung“. F10 verwandelt die Veränderungen physikalischer und chemischer Variablen aus der Umgebung in messbare Werte, wobei die verfügbaren Daten auf den fünf Sinnen Sehen, Hören, Riechen, Tasten und Schmecken basieren. Die gesammelten Sensordaten werden von F10 an F11 weitergegeben, wo sie in Neurosymbole für die weitere Verarbeitung im psychischen Apparat umgewandelt werden.

2.2.3.1.4 Körperwahrnehmungsschiene

Über die Körperwahrnehmungsschiene kommen schließlich körperinterne Reize, wie z.B. die Herzschlagrate, Schmerz oder Völlegefühl, in die Entscheidungsfindungseinheit.

Die Körperwahrnehmungsschiene deckt wie die Umgebungswahrnehmungsschiene lediglich die Schichten 1 und 2 ab und besteht aus F12 – „Sensoren Körper“ und F13 – „Neurosymbolisierung Körper“. F12 ist verantwortlich für das Erkennen des eigenen körperlichen Zustandes wie etwa die Stellung der Extremitäten oder das Bestehen von Schmerzreizen. F13 wandelt diese Informationen wiederum in Neurosymbole für die weitere Verarbeitung im psychischen Apparat um.

2.2.3.1.5 Triebschiene

Die Informationen aus der Sexual- und die Selbsterhaltungstriebschiene werden in der so genannten Triebschiene zusammengeführt, um in weiterer Folge mit Informationen aus der Wahrnehmungsschiene bereichert werden zu können.

Die Triebschiene setzt sich zusammen aus den Modulen F48 – „Bildung von Triebkompomenten“, F49 – „Urverdrängung Triebwünsche“, F54 – „verdrängter Triebinhalte“ und F57 – „Bildung von Triebrepräsentanzen“. Die Sexual- und Selbsterhaltungstriebrepräsentanzen werden in F48 zusammengeführt und die Triebreduktion (im Verhältnis zum letzten Zyklus) wird als Lust abgeführt und im Libido-Speicher abgespeichert. In F57 wird die Triebrepräsentanz um ein Triebobjekt und ein Triebziel erweitert, wobei die Auswahl darauf basiert, welches Objekt und welche Aktion (repräsentiert durch das Triebziel) für den jeweiligen Trieb mit der höchsten Befriedigung erinnert wird. Das Resultat dieses Prozesses ist die Anordnung der involvierten Datenstrukturen als Baum, der die Triebe mit den erinnerten Objekten und Aktionen assoziiert. In der Funktion F49 werden diese assoziierten Inhalte mit dem Speicher urverdrängter Inhalte abgeglichen und markiert. In F54 versuchen bereits verdrängte Inhalte wieder bewusst zu werden, indem sie sich an passende Strukturen anhängen.

2.2.3.1.6 Wahrnehmungsschiene

In der Wahrnehmungsschiene wird die Wahrnehmung durch subjektive Inhalte wie Erinnerungen zusammengeführt, dadurch ergibt sich eine erste Bewertung der in der Wahrnehmung verfügbaren Möglichkeiten anhand von Affektbeträgen und auch Emotionen.

In technischer Hinsicht werden die Umgebungswahrnehmungsschiene und die Körperwahrnehmungsschiene in der Wahrnehmungsschiene zusammengefasst und mit Erinnerungen assoziiert. Dies dient dem Zweck, eine erste Auswahl treffen zu können, welche Wahrnehmungen für den Agenten wichtig sein könnten. Die derart selektierten Wahrnehmungen werden danach mit der inneren Vorstellungsschiene des Agenten zusammengeführt. Auf Ebene der inneren Vorstellungsschiene können sich die bereits vorhandenen Wahrnehmungen mit den Wunschvorstellungen des Agenten überlappen. Wenn dies der Fall ist, wird diese Information an die Wahrnehmungsschiene weitergeleitet und ihr dort in weiterer Folge die erste Relevanz zugewiesen. Im Rahmen dieses Prozesses versuchen Inhalte, die zuvor im Rahmen der Abwehr, siehe dazu Kapitel 3.2.1 *Innere Vorstellungsschiene*, verdrängt wurden, sich wieder in die Wahrnehmungssphäre und damit ins Bewusstsein zu drängen.

Die Wahrnehmungsschiene setzt sich aus den Modulen F14 – „Äußere Wahrnehmung“, F18 – „Affektbeträge für Wahrnehmung“, F35 – „Auftauchen verdrängter Inhalte“, F37 – „Urverdrängung Wahrnehmung“, F45 – „Abfuhr von psychischer Intensität“, F46 – „Verbindung mit Erinnerungsspuren“ und F63 – „Zusammenstellung von Emotionen“ zusammen.

F14 generiert eine Sachvorstellung aus den neurosymbolischen Inhalten sodass sie die Daten aus dem Körper und die Umgebungssensoren vom mentalen Apparat verarbeitet werden können. F46 generiert Assoziationen zwischen den in F14 generierten Sachvorstellungen und erinnerten Inhalten. F37 und F35 haben Zugriff auf Erinnerungen verdrängter Inhalte. In dem Modul F37 werden diese assoziierten Inhalte mit dem Speicher verbotener Inhalte abgeglichen und markiert. In F35 versuchen bereits verdrängte Inhalte wieder bewusst zu werden, indem sie sich an passende Strukturen anhängen.

Die Funktion des Moduls F45 liegt darin, dass Libido abgeführt und Lust erzeugt wird, wenn Aktionen wahrgenommen werden, die die aktuellen Triebe befriedigen, wobei die Libidoabfuhr einen Lustgewinn bedeutet, der zu F18 weitergeleitet wird. F18 summiert die Lust und die Unlust für alle Triebe desselben Typs.

In F63 werden auf Basis der Triebrepräsentanzen, der Wahrnehmung und der phantasieaktivierten Inhalte Basisemotionen erzeugt. Die Zusammenstellung dieser Basisemotionen ist abhängig von vier Emotionsskalaren: Lust, Unlust, aggressiver Anteil der Affektbeträge und der libidinöser Anteil der Affektbeträge. In SiMA existieren sechs Basisemotionen, die anhand der nachfolgenden Liste aus diesen vier Skalaren gebildet werden:

- Wut ergibt sich aus einer Dominanz von Unlust + aggressiver Anteil der Affektbeträge
- Trauer ergibt sich aus einer Dominanz von Unlust + libidinöser Anteil der Affektbeträge
- Sättigung ergibt sich aus einer Dominanz von Lust + libidinöser Anteil der Affektbeträge
- Hochgefühl ergibt sich aus einer Dominanz von Lust + aggressiver Anteil der Affektbeträge
- Angst ergibt sich aus einer Dominanz von Unlust
- Freude ergibt sich aus einer Dominanz von Lust

2.2.3.1.7 Abwehrschiene

Durch die Zusammenführung von Trieb- und Wahrnehmungsschiene entstehen Emotionen, die über aktuelle interne und externe Wahrnehmungen verknüpft werden. Emotionen sind im SiMA-Modell ein grundlegender Bewertungsmechanismus der vorhandenen Inhalte, siehe dazu Kapitel 3.1.4. Alle erhaltenen Informationen, angereichert durch Erinnerungen und eben Emotionen, werden in der Folge in der Abwehrschiene anhand der Anforderung des Über-Ichs gefiltert. Hier könnte z.B. eine bestimmte Information verdrängt werden, die dann über die Wahrnehmungsschiene in den folgenden Zyklen wieder versuchen könnte, abermals bewusst zu werden.

Diese Schiene besteht aus den Modulen F56 – „Desexualisierung/Neutralisierung“, F55 – „Über-Ich proaktiv“, F7 – „Über-Ich reaktiv“, F6 – „Abwehrmechanismen für Triebwünsche“ und F19 – „Abwehrmechanismen für Wahrnehmung“.

Das Modul F56 berechnet, abhängig von den aktuellen Trieben, einen Faktor namens neutralisierter Intensität, der verwendet wird, um zu beeinflussen, wie viel Aufwand andere Module in die Erfüllung ihrer Aufgabe stecken dürfen (z.B. plant ein Agent mit viel neutralisierter Intensität weiter in die Zukunft als ein mit weniger neutralisierter Intensität ausgestattetes Pendant). Während F55 für das proaktive Befolgen internalisierter Regeln verantwortlich ist, betrifft F7 die reaktive Durchsetzung internalisierter Regeln. F6 ist verantwortlich für das Auflösen entstehender triebbezogener Konflikte, F19 wiederum ist für das Auflösen von wahrnehmungs- und emotionsbezogenen Konflikten zuständig.

Für eine ausführliche Beschreibung der Funktionsweise der Abwehr siehe Kapitel 3.2.1.

2.2.3.1.7.1 Bereits vorhandene Abwehrmechanismen

Der SiMA Agent bedient sich aktuell der folgenden vordefinierten Mechanismen welche in zukünftige Versionen im Langzeitgedächtnis abgespeichert werden sollen, aber zur Zeit hardgecodet sind.

- Affektverkehrung (*Reversal of Affect*). Hierbei werden bei der Bildung von Emotionen Affektbeträge aus aggressiven Quellen wie Affektbeträge aus libidinösen Quellen behandelt und umgekehrt.
- Rationalisierung (*Rationalization*). Hierbei werden Triebziele in einer hardgecodeten Liste, welche Triebziele razionalisierteren Alternativen zuordnet, gesucht und durch die jeweilige Alternative ersetzt.
- Projektive Identifizierung (*Projective Indentification*). Hierbei werden DM’s oder Emotionen welche von den Über-Ich Regeln als konflikthaft markiert wurden und mit dem Agenten selbst assoziiert sind beim TPM aus der Wahrnehmung, welches einen anderen Agent repräsentiert, als dessen DM oder Emotion assoziiert.
- (*Splitting Depreciation*) Hierbei werden TPM’s die mit konflikthaften Inhalten assoziiert sind mit einer hardgecodeten Liste von die als positiv betrachteten TPM’s abgeglichen und wenn Übereinstimmung gefunden wurde, werden die TPM’s entfernt.
- (*Splitting Idealization*). Hierbei werden TPM’s die mit konflikthaften Inhalten assoziiert sind mit einer hardgecodeten Liste von die als negativ betrachteten TPM’s abgeglichen und wenn Übereinstimmung gefunden wurde, werden die TPM’s entfernt.

- (*Denial or Disavowal*) Hierbei werden Inhalte welche von den Über-Ich Regeln als konflikthaft markiert wurden von der Abwehr herausgefiltert und entfernt.
- (*Projection*). Hierbei wird die Assoziation von konflikthaften DM oder Emotionen die mit dem Agenten selbst assoziiert sind entfernt und durch Assoziationen zu einem TPM aus der Wahrnehmung ersetzt.
- Verdrängung (*Repression*). Hierbei werden Inhalte welche von den Über-Ich Regeln als konflikthaft markiert wurden von der Abwehr herausgefiltert. Das bedeutet, dass in der entsprechenden Datenstrukturen der Wert von Triebquelle, Triebziel und/oder Triebobjekt entfernt wird.
- Wendung gegen das selbst (*Turning Against the Self*). Hierbei werden Triebwünsche auf den Agenten selbst gerichtet. Das bedeutet, dass in der konflikthaften DM das konflikthafte Triebobjekt durch den Agenten selbst ersetzt wird.
- Intellektualisierung (*Intellectualization*). Hierbei werden Triebziele in einer hardgecodeten Liste, welche Triebziele intellektualisierteren Alternativen zuordnet, gesucht und durch die jeweilige Alternative ersetzt.
- Reaktionsbildung (*Reaction Formation*). Hierbei werden Triebziele in einer hardgecodeten Liste, welche Triebziele ihre Gegenstücke zuordnet, gesucht und durch die jeweilige Alternative ersetzt.
- Verschiebung (*Displacement*). Hierbei werden Triebziele in einer hardgecodeten Liste, welche für jedes Triebziel sozial gleichgestellte Alternative bereitstellt, gesucht und durch die jeweilige Alternative ersetzt.
- Sublimierung (*Sublimation*). Hierbei werden Triebziele in einer hardgecodeten Liste, welche für jedes Triebziel sozial und kulturell höher gestellte Alternative bereitstellt, gesucht und durch die jeweilige Alternative ersetzt.
- Verkehrung ins Gegenteil (*Reversal into the Opposite*). Hierbei werden Triebziele in einer hardgecodeten Liste, welche für jedes aktives Triebziel ein passives Triebziel und für jedes passive Triebziel ein aktives Triebziel bereitstellt, gesucht und durch die jeweilige Alternative ersetzt.
- Projektion (*Projection*). Hierbei werden bei Triebzielen welche von den Über-Ich Regeln als konflikthaft markiert wurden und bei denen der Agent selbst die Triebquelle ist, die Triebquelle durch eine andere Person oder ein anderes Objekt ersetzt.

2.2.3.1.8 Umwandlungsschiene

In der, der Abwehrschiene nachfolgenden Umwandlungsschiene erfolgt die Umwandlung vom unbewussten primären Prozess zum bewussten sekundären Prozess. Im Zuge dieser Umwandlung werden die Daten um Wortvorstellungen und strukturierende Konzepte (wie z.B. Reihenfolge) erweitert. Als Wortvorstellung lässt sich ein Komplex von zueinander assoziierten Vorstellungen, die ein Wort der natürlichen Sprache in der Psyche repräsentieren, beschreiben. Dazu gehören etwa Klangbild, Lesebild, Schriftbild, Bewegungsbild, wobei in Hinblick auf die übliche Sprachpraxis das Klangbild überwiegt [DBD+14, p.133]. Im sekundären Prozess passiert zusammengefasst das, was wir Menschen als bewusstes Denken kennen.

Die Umwandlungsschiene setzt sich aus den Modulen F71 – „Zusammenstellung von erweiterter Emotion“, F21 – „Transformation in den Sekundärprozess für Wahrnehmung, F20 – „Zusammenstellung der Gefühle“ und F8 – „Transformation in den Sekundärprozess für Triebwünsche“ zusammen.

Im Primärprozess findet alles zeitgleich statt, es gibt sozusagen kein Davor und kein Danach und es gibt vor allem auch keine Kausalität. Die Umwandlungsschiene erweitert die existierenden Inhalte um diese Informationen: In F71 werden basierend auf Änderungen, die die Abwehr an den Basis-Emotionen vorgenommen hat, erweiterten Emotionen erzeugt. F21 transformiert die Daten des Primärprozesses zu Daten des Sekundärprozesses, wobei die Sachvorstellungsnetze aus dem Primärprozess um sekundärprozesshafte Datenstrukturen wie Wortvorstellungsnetze erweitert werden. Hier werden auch die logischen Zusammenhänge hinzugefügt, wie z.B. die zeitliche und hierarchische Reihung. Unter zur Hilfenahme der sekundärpozesshaften Informationen können aus einzelne zusammengehörige Images logisch strukturierte Akte gebildet werden. In F20 werden aus vorbewussten[2]-erweiterten Emotionen Gefühle erzeugt. Dabei wird der erweiterten Emotion ein Wortvorstellungsnetz zugeordnet, das es ermöglicht das Gefühl sprachlich zum Ausdruck zu bringen. In F8 wird schließlich aus den Triebrepräsentanzen eine erste Liste möglicher Ziele des Agenten erstellt. Diese Liste repräsentiert die grundlegendsten Befriedigungsmöglichkeiten der aktuellen Triebe.

2.2.3.1.9 Wünsche- und Forderungsschiene

In der Schiene für Wünsche und Forderungen wird die Liste an Handlungsmöglichkeiten anhand von Kriterien der langfristigen Planung und auf Basis sozialer Regeln angereichert.

Die Schiene setzt sich aus den Modulen F51 – „Realitätsprüfung Wunscherfüllung“, F61 – „Lokalisierung“, F23 – „Aufmerksamkeitsfokusierung“ und F69 – „Extraktion von möglichen Zielen“ zusammen.

F51 überprüft, ob die aktuellen Ziele angesichts der aktuellen Umgebung umsetzbar sind. F61 ist für die geographische Orientierung und für die Bestimmung, wo sich der Agent in dieser Umgebung befindet, verantwortlich. F23 hat die Aufgabe die Wahrnehmung und das Kurzzeitgedächtnis zu fokussieren. Dabei wird abhängig von den aktuellen Plänen priorisiert, welche Objekte aus der Wahrnehmung und / oder aus den psychischen Inhalten[3] fokussiert werden. Die Fokussierung passiert, indem der Affektbetrag des betreffenden Objektes oder Inhaltes erhöht wird. F69 schließlich extrahiert alle wählbaren Ziele, die für den Agenten verfügbar sind. Wenn für ein bestimmtes Triebziel kein wählbares Ziel existiert, werden mögliche Ziele daraus erstellt, um dem Agenten anschließend die Möglichkeit zu geben, danach zu suchen“ [DBD+14, p.136].

2.2.3.1.10 Auswahlschiene

[2] Unter vorbewussten Inhalten werden im gegebenen Zusammenhang Inhalte, die unbewusst sind, aber bewusst werden können, verstanden. Sie befinden sich an der Grenze vom Primär- zum Sekundärprozess.

[3] Laut psychoanalytische Definition beschreibt der Begriff „psychischer Inhalt“ alle Daten/Informationen der Psyche, die mit Bewertungsgrößen versehen werden können.

Die Auswahlschiene befasst sich schließlich damit, die Liste an Handlungsmöglichkeiten anhand der zu erwartenden Befriedigung und des zu erwartenden Aufwands zu reihen, wobei die Auswahl die zu erwartende Eignung unter Berücksichtigung der aktuellen Wahrnehmungs- und Trieblage wiederspiegelt.

Die genannte Schiene setzt sich aus den Modulen F26 – „Entscheidungsfindung", F29 – „Bewertung von imaginären Handlungen", F52 – „Generierung von imaginären Handlungen" und F53 – „Realitätsprüfung Handlungsplanung" zusammen.

Das Modul F26 ist der erste Reduktionsmechanismus. Das bedeutet, dass bis zu diesem Prozess relativ einfache, nicht sonderlich rechenaufwendige Funktionen angewendet wurden. Mit anderen Worten: Im Modul 26 wird erstmals eine Vorauswahl getroffen, um die Anzahl der genauer zu untersuchenden Möglichkeiten zu reduzieren, was den Rechenaufwand minimieren soll. Die zentrale Entscheidungsmaxime für diese Reduktion sind die Triebe, da im Rahmen des Reduktionsprozesses entschieden wird, welcher Trieb angesichts der aktuellen inneren Situation und Außenwelt die beste Befriedigung verspricht. An dieser Stelle wird aber noch nicht über den Modus der Triebbefriedigung entschieden, sondern erstmals nur festgelegt welche Triebe befriedigt werden sollen. Das Modul F52 simuliert verschiedene Handlungspläne, ohne sie an den Körper weiterzuleiten, um das Ergebnis besser bewerten zu können. Die simulierten Handlungspläne des Moduls F52 werden von Modul F53 anhand von semantischem Wissen um die Durchführbarkeit und auf Basis der Anforderungen überprüft. Das Modul F29 bekommt schließlich vom Modul F53 die evaluierten Handlungspläne weitergereicht und wählt einen dieser Pläne nach dem Prinzip der Lustmaximierung und Unlustminimierung aus. Der am besten geeignete wird ausgewählt und zusammen mit den anderen eine gewisse Zeitlang im Kurzzeitgedächtnis abgespeichert.

2.2.3.1.11 Innere Vorstellungsschiene

Alle Handlungspläne, die im Rahmen der Aktionsschiene nicht unmittelbar verfolgt werden, werden über die Innere Vorstellungsschiene oder Imaginationsschiene in den Primärprozess zurückgeschickt, um mehr einschlägige Informationen aus den Erinnerungen zu aktivieren. Für das Zurückschicken müssen die Handlungspläne umgewandelt werden, was zu einer Umkehrung des Prozesses der Umwandlungsschiene führt.

Die Imagionationsschiene besteht aus dem Modul F47 – „Umwandlung in den Primärvorgang", dieses reduziert die Daten um Wortvorstellungen und strukturierende Konzepte.

2.2.3.1.12 Aktionsschiene

Die Aktionsschiene extrahiert aus den weiterverfolgten Handlungsplänen die nächste auszuführende Handlung und gibt sie an die so genannte Aktuator-Schiene weiter.

Die Aktionsschiene besteht aus dem Modul F30 – „Bewegungssteuerung". Dieses Modul zerlegt die Aktion des Handlungsplanes in viele Einzelaktionen, welche der Körper ausführen kann.

2.2.3.1.13 Aktuator Schiene

In der Aktuator-Schiene passiert nun – den Prozess abschließend – die Rückumwandlung in die Schicht 1 des oben beschriebenen funktionalen Modells, also in die neuronale Schicht.

Das Modul F31 – „Neurodesymbolisierung- Handlungsanweisung“ wandelt die Vorgaben von F30 in Bewegungsbefehle um. So ähnlich verwandelt das Modul F67 – „Körperliche Emotionsreaktion“ die aktuelle Emotionslage in körperliche Reaktionen. Diese Informationen werden letztendlich von den Modulen F32 und F68 –„Aktuator für Drüsen“ in Hardwarebefehle umgewandelt, die zu einer Reaktion des Agenten führen.

2.2.4 Aktueller Stand der Implementierung – MASON

Das SiMA-Modell wird derzeit in unterschiedlichem Umfang in verschiedenen Projekten implementiert. Die Implementierungsarbeit konzentriert sich auf eine *artificial life simulation* mit dem Ziel, menschenähnliches Verhalten zu erforschen. Alternative Einsatzgebiete von SiMA sind im Bereich der Nutzersimulation für ein Marketing-tool im Bereich Grünstrom vorhanden, siehe Projekt CogMAS [SMW+15]. Ein anderes Anwendungsbereit ist der Einsatz zur automatisierter Optimierung einer Gebäudesteuerung im Projekt KORE [ZWSS16].

Es existieren einige Simulationsplattformen, um Agenten in *artifical life simulations* zu testen. Die Projektentwickler von SiMA haben sich für die Simulationsumgebung MASON [Luke11] entschieden, um ihr Modell zu evaluieren. Eine detaillierte Beschreibung des Auswahlverfahrens findet sich in [Lang10, pp.36-42]. MANSON ist ein flexibles Simulationstoolkit, hat ein gutes Scheduling und weist eine gute Physics-Engine-Performance auf. Ferner ist es leicht erweiterbar und kann für eine große Breite an Simulationsszenarien verwendet werden. Ein anderer Vorteil ist, dass es eine sehr große Anzahl von Agenten unterstützt. MANSON ist ein Single Prozess System, seine Bibliothek ist Open Source und es wurde komplett in Java entwickelt [Luke11].

Die Simulationsumgebung, in der die SiMA *Decision Unit* entwickelt wird, setzt sich aus zwei Teilen zusammen. Ein Controller zur Steuerung der Simulation, sowie eine Physics-Engine. Diese zwei Teile werden vom MANSON-Projekt als Bibliotheken eingebunden. Die eigentliche *artificial life simulation* wurde gemeinsam mit der *Decision Unit* in Java vom Team des SiMA-Projektes entwickelt. MANSON ist für die Ausführungszyklen im SiMA verantwortlich, d.h. es sorgt dafür, dass die Module zur richtigen Zeit aufgerufen werden.

Das MANSON-Toolkit besteht aus drei Schichten:

- *Utility Layer*
  Der Utility Layer besteht aus Klassen, die allgemeine Tools beinhalten. Als Beispiele wären etwa Datenzugriffswerkzeuge, Methoden zur Bilder- und Videoerzeugung oder Zufallsgeneratoren zu nennen.
- *Visualization Layer*
  Der *Visualization Layer* beinhaltet eine GUI zur Steuerung der Simulation und zur visuellen Darstellung (2d oder 3d) der Daten des Modells. Da der Layer unabhängig von der Simulation ist, kann die Visualisierung durch andere Visualisierungen ersetzt werden, ohne dabei die Simulation zu beeinflussen.

- *Model Layer*
  Der Model Layer ist komplett unabhängig vom *Visualization Layer*. Im Bereich des Model Layers wird ein Entscheidungsfindungsmodell eingebettet, das es möglich macht, die Simulation ohne oder mit einer eigenen Visualisierung auszuführen.

Der Hauptvorteil von MANSON ist die Laufgeschwindigkeit. MANSON wurde mit dem Fokus entwickelt, Modelle mit vielen Agenten und zahlreichen Interaktionen zu ermöglichen. Die Möglichkeit, die Simulation willkürlich zu stoppen, zu starten oder fortzusetzen, bietet die perfekte Grundbasis für lange Simulationen in MANSON.

2.2.5 Darstellung eines Use-Case und der Aufgabenstellung dieser Diplomarbeit

Da das SiMA-Modell auf einem Use-Case Development Design basiert, wird hier der Use-Case 1 AsS (Adam sucht Schnitzel) vorgestellt, [DBD+14, pp.49-52]. Er soll

- das Aufzeigen eines Konflikts zwischen inneren Bedürfnissen,
- das Assoziieren von Erinnerungen,
- die Auswahl von Abwehrmechanismen und
- eine einfache Handlungsplanung

durchspielen können.

Der Use-Case 1 beinhaltet zwei Agenten, die um eine Nahrungsquelle konkurrieren, Abbildung 2.7. Der eine SiMA-Agent namens Adam wird von der SiMA Entscheidungsfindungseinheit gesteuert, während der andere Agent namens Bodo ein passiver Träger des mentalen Apparates ist. Im Bild befinden sich Bodo (in roter Farbe oben), Adam (in grüner Farbe unten) und der Schnitzelteller. Die blauen Linien sind ist die jeweilige Wahrnehmung der Agenten.

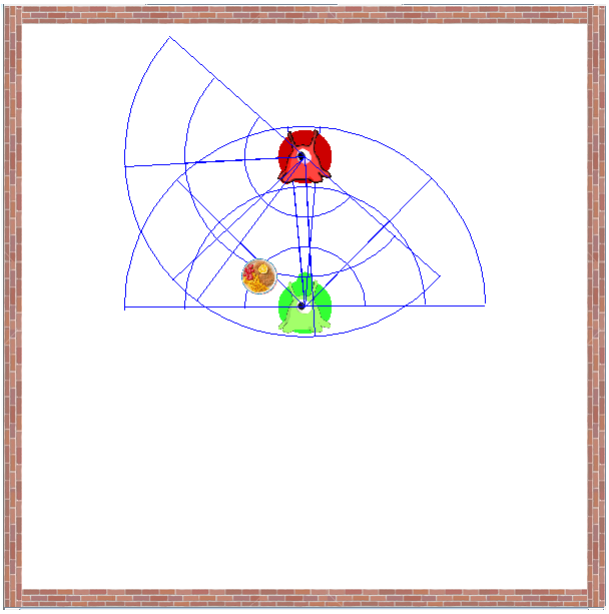


Abbildung 2.7: Screenshot Use-Case 1

Zu der angeführten Ausgangssituation im Abbildung 2.7 wurden mehrere Abläufe definiert:

- Adam holt sich das Schnitzel und teilt es mit dem anderen Agenten;
- Adam gibt Bodo das Schnitzel;
- Adam holt sich das Schnitzel und isst es auf;
- Adam wird aggressiv, macht sich Bodo unterwürfig und isst das Schnitzel auf;
- Adam macht keinen einzigen Schritt, weil er durchgehend nachdenkt, was er tun soll;
- Adam flieht.

Adam soll sich für den Plan entscheiden, der ihm längerfristig die meiste Lust bringt bzw. die wenigste Unlust bereitet und diesen ausführen.

Der erste, zweite und dritte Ablauf, „das Teilen“, „das Geben“ und „das Holen“, kann zurzeit aus zweierlei Gründen nicht ausgeführt werden: Zum einen besitzt der Agent derzeit nicht die Möglichkeit, etwas aufzuheben, weshalb er weder „Teilen“ noch „Geben“ kann. Zum anderen sollen diese drei Abläufe entsprechend der psychoanalytischen Fall-Beschreibung durch Anwendung verschiedener Abwehrmechanismen erreicht werden, wobei es in der aktuellen Version jedoch noch keinen Möglichkeit gibt um die Anwendung von Abwehrmechanismen in den einzelnen Anwendungsfällen zu konfigurieren.. In diesem Bereich knüpft die vorliegende Diplomarbeit an:

Wie im Zuge der einleitenden Darstellung des SiMA-Modells aufgezeigt, wird das erwähnte Modell top-down entwickelt. Das führt dazu, dass einige Funktionen derzeit – wie ebenfalls bereits herausgearbeitet – nur definiert, aber noch nicht umgesetzt sind, andere sind noch nicht ganz fertig definiert. Besonders hervorzuheben ist in diesem Zusammenhang die Abwehrschiene. Dort werden aktuell nur beispielhafte Filterregeln zu Demonstrationszwecken hardgecodet verwendet. Die Abläufe der Abwehr wurden nur debugg-geeignet visualisiert.

Die Interaktionsmöglichkeiten des Agenten mit seiner Umwelt sind in diesem Sinne daher aktuell auf das Essen, Bewegen und Ausscheiden beschränkt.

Diese Diplomarbeit befasst sich mit der Erweiterungen des SiMA-Modells. Der SiMA-Agent soll im Rahmen meiner Ausarbeitung in die Lage versetzt werden, Sachen aufzuheben, loszulassen bzw. in sein Inventar zu übernehmen. Ferner soll die Konfiguration des Abwehrverhaltens der Psyche durch Definition von Regeln zur Erkennung von konflikthaften Inhalten für Anwender ohne Programmierkenntnisse ermöglicht werden. Darüber hinaus sollen mehrere neue Inspektoren für die Visualisierung der neuen Funktionalitäten entwickelt und die bereits existierende Visualisierung der Abwehr erweitert werden, um sichtbar zu machen, welche Änderungen die aktivierten Abwehrmechanismen herbeigeführt haben.

2.3 Vergleich von KI-Architekturen

Nach Darstellung des SiMA-Modells werden in diesem Kapitel die kognitiven Architekturen SOAR, LIDA, BDI und Volitron vorgestellt und inhaltlich erläutert. Bei jeder Architektur werde ich versuchen, aufzuzeigen, in welchen Bereichen „Verbesserungspotential“ besteht und wie SiMA hier im Vorteil ist.

2.3.1 SOAR

Kurzdarstellung des SOAR-Modells

SOAR (*state operator apply result*) ist – ebenso wie das SiMA-Modell – eine auf einer kognitiven Architektur basierende AGI. Sie wurde ursprünglich in den 1980er Jahre von Allen Newell, John Laird und Paul Rosenbloom entwickelt und 1996 durch Laird und Rosenbloom verbessert [Lair08, p.1]. SOAR ist ein regelbasiertes System, das die Problemstellung durch Zustände und Ziele beschreibt, wobei durch Regeln aktivierte Operatoren genutzt werden, um diese Zustände zu beeinflussen und ans Ziel zu kommen.

Die Vorgehensweise ist derart strukturiert, dass ein Problem in einem hypothetischen Raum, der alle Informationen enthält, die das Problem beschreiben, gesucht wird. Der Produktionsspeicher beinhaltet dabei das dauerhafte Wissen in Form von Produktionsregeln. Im Arbeitsspeicher wird durch Objekte das temporäre Wissen repräsentiert.

Die aktuelle Version – SOAR 9 – nutzt zwei Arten von Erinnerungen, die symbolische Langzeit- und die symbolische Kurzzeit-Erinnerung:

- Die symbolischen Langzeiterinnerung unterteilt sich in drei Bereiche: (1) eine “*Procedural Memory*”, die das Wissen des Agenten als Regel darstellt; (2) die “*Semantic Memory*”, die das Wissen als Fakt(en) repräsentiert; und (3) die “*Episodic Memory*”, die Erfahrungen speichert.
- Das symbolische Kurzzeitgedächtnis beinhaltet demgegenüber die Einschätzung der aktuellen Situation und Wahrnehmung, abgeleitet vom Langzeitgedächtnis. Es besteht aus symbolischen Kurzzeiterinnerungen, die als Graphenstruktur abgebildet werden. Der Graph beinhaltet die Objekte der aktuellen Szene bzw. des aktuellen Zustandes mit ihren bzw. seinen Eigenschaften und Relationen sowie Beziehungen zueinander.

Neu in der Version 9 ist die Funktionalität, die automatisch Symbole aus statistischen Regelmäßigkeiten bildet. Diese Erneuerung erweitert die ursprünglich rein symbolische kognitive Architektur um subsymbolische Elemente.

Chunking ist der klassische Lernmechanismus in SOAR. Dieser eignet sich verfahrenstechnisches Wissen an, bricht Probleme in Teilziele und versucht diese durch Regeln zu lösen. Er lernt das Ergebnis, um den zu bearbeitenden Prozess nicht noch einmal in Teilziele abbrechen zu müssen. [Lair08, p.4]

Reinforced-Learning ist ein zusätzlicher Lernmechanismus ab der Version 9 der mit einer Belohnungsfunktion arbeitet, d.h. wenn z.B. ein Zustand eine gute Bewertung hat, werden Aktionen, die zu diesem Zustand führen, eher ausgewählt.

Bevor eine Handlung ausgeführt wird, reduziert SOAR die Anzahl an möglichen Aktionen durch verschiedene Mechanismen, welche in [Lair08, p.10] beschrieben sind. Wenn mehrere Handlungen als Ausführungsresultat in Frage kommen, führt SOAR trotzdem nur eine dieser Handlungen durch, um ans gewünschte Ziel zu kommen. Wenn hingegen keine Handlung generiert werden kann, erstellt SOAR eine neue Grundlage und wiederholt den Ausschlusszyklus.

Ein Ausschlusszyklus in SOAR lässt sich wie folgt grafisch darstellen Abbildung 2.8:

Abbildung 2.8: SOAR Ausschlusszyklus [Lair08, p.4] nachgezeichnet

Die Wahrnehmung aktiviert als Input Regeln im Working Memory. Diese durchlaufen Operatoren, die auf die Zustände im Working Memory angewendet werden können. Dadurch bilden die Operatoren zusätzliche Informationen, die das in den Regeln abgespeicherte Wissen mit zusätzlicher Information anreichern. Operatoren können ihrerseits neue Regeln auslösen. Diese revolvierende Aktion wiederholt sich solange, bis keine weiteren Regeln mehr anschlagen. Danach werden die aktuellen Operatoren evaluiert und anhand dessen werden die möglichen Aktionen, die von den Operatoren vorgeschlagen wurden, für die momentane Situation ausgewählt. Wenn es eine klare und als beste Option bewertete Aktion gibt, wird diese ausgeführt, falls aus den verfügbaren Aktionen keine als die beste bestimmt werden kann, kommt es zum einem *Subgoaling*. Dabei erstellt SOAR ein neues Ziel mit dem Inhalt, die bestehende ausweglose Situation aufzulösen. Nachdem dieser Vorgang abgeschlossen ist, kommt es zum bereits beschriebenen „*chunking*“, das aus einem erfolgreich gelösten *Subgoal* eine Regel erschafft, die in weiterer Folge im Langzeitgedächtnis als eine solche abgelegt wird.

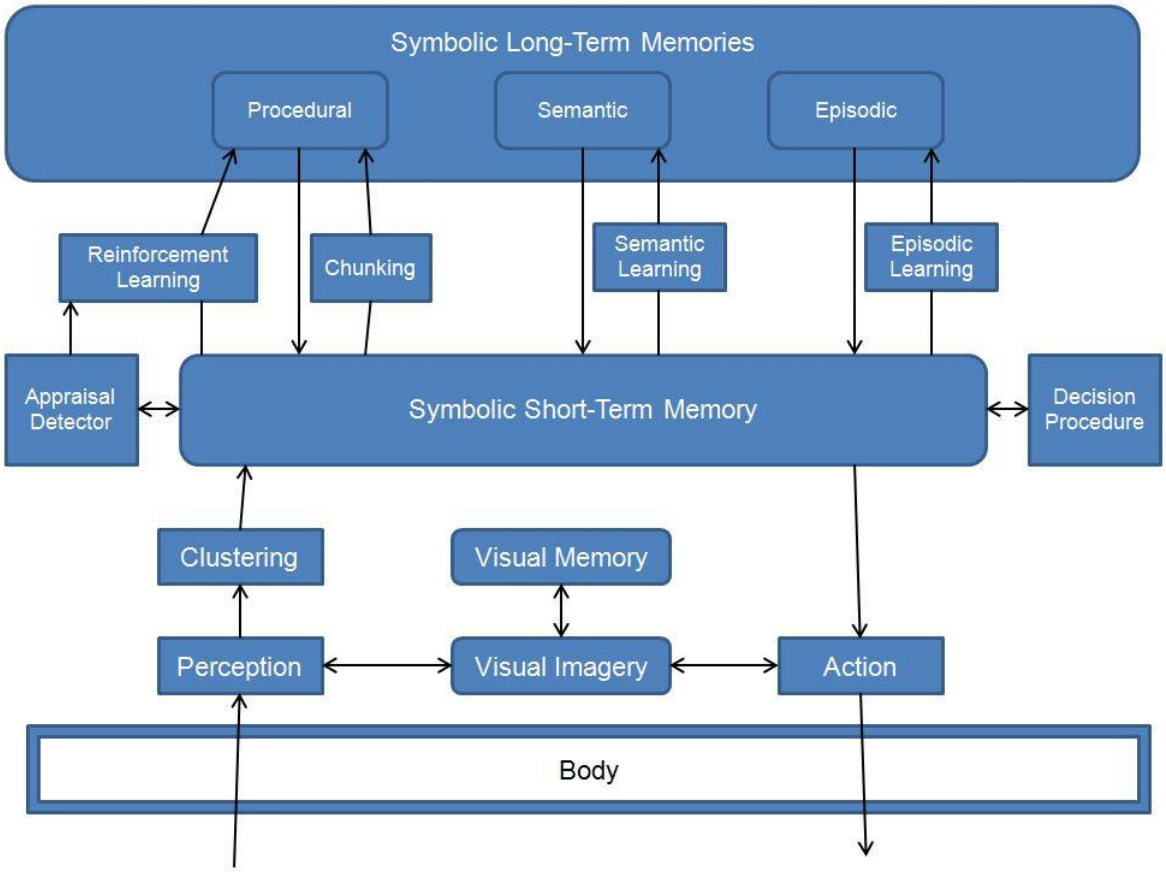


Abbildung 2.9: SOAR9 Architektur [Lair08, p.3] modifiziert nachgezeichnet

Seit Version 9 gibt es ein *Visual Imagery Modul*, siehe Abbildung 2.9, das mit einem eigenen visuellen Speicher verknüpft ist, um Aufgabestellungen aus der Bildverarbeitung (z.B. das Auswerfen aller Buchstaben mit einem senkrechten Strich, wobei die Eigenschaft "senkrecht" nicht als Regel beschrieben werden kann) zu meistern, da diese nicht durch Regeln lösbar sind. Das Modul ersetzt, wenn angewendet, den normalen SOAR Zyklus mit einem speziellen auf Bildverarbeitung zugeschnittenen Zyklus.

Zusammenfassend kann festgehalten werden, dass Regeln in SOAR spezifisch auf bestimmte Aufgaben abgestimmt sind und daher viel Input brauchen. Die Art und Weise, wie SOAR Regeln kombiniert, sorgt dafür, dass sehr kurze Regeln benötigt werden. Das bedeutet, dass die Abbildung eines sehr komplexen Zusammenhangs einer Vielzahl an Regeln bedarf, die zusätzlich auch wieder über Regeln miteinander funktionieren. Da die Definition der Regeln und der damit verbundene Aufwand exponentiell wachsen, erscheint eine Umsetzung unmöglich.

Unterschiede zum SiMA-Modell

Im Gegensatz zu SOAR wird die Entscheidungsfindung im SiMA-Modell in nur einem Zyklus erreicht. Die Abwehrmechanismen bewerkstelligen dies so, dass die durch einen Verstoß gegen eine

Regel verursachten Konflikte durch Anwendung verschiedener Abwehrmechanismen[4] gelöst werden. SiMA ist dadurch schneller und effizienter als SOAR.

SiMA hat überdies, anders als SOAR, kein Regelsystem, sondern basiert auf einer Ontologie, d.h. das Wissen ist nicht in der Form von Regeln beschrieben, sondern in Form von Zusammenhängen. Dadurch ist SiMA deutlich flexibler und aus diesem Grund besser für gewisse Situationen wiederverwendbar.

2.3.2 LIDA

Kurzdarstellung des LIDA-Modells

LIDA steht für „Learning Intelligent Distribution Agent“ und ist eine Erweiterung des IDA-Modells, das für die US Marine entwickelt wurde. Das System war ursprünglich dazu gedacht, die Personalzuweisung von Seemännern nach abgeschlossenen Touren zu automatisieren. Im Zuge dessen mussten nämlich Daten aus verschiedenen Quellen, z.B. aus Email-Verkehr und Notizen von Vorgesetzten, verknüpft werden, sowie die Befähigungen und die Ausbildung der einzelnen Mitarbeiter in Betracht gezogen werden, was eine durchaus komplexe Planung erforderte [FP06, p.1].

Ursprünglich wurde LIDA [FMDS14, p.1] von Stan Franklin und Kollegen von der Universität Memphis entwickelt. Die LIDA Architektur basiert auf einer Kombination aus Erkenntnissen aus verschiedenen Teilbereichen der Kognitionswissenschaft und ist in der kognitiven Neurowissenschaft empirisch fundiert. Sie ist eine hybride Architektur, bei der automatisch gruppierte Sensordaten mit symbolischen Speicherinhalten verknüpft werden. LIDA verfügt über Lernmechanismen, die in den einzelnen Modulen spezifisch für die jeweilige Aufgabe implementiert wurden. LIDA kann Aufmerksamkeit schenken, Handlungen selbständig auswählen und verfügt über menschenähnliches Lernvermögen. Sie kombiniert eine komplexe Handlungsauswahl mit Selbstmotivation durch Emotionen, einem zentralen Aufmerksamkeitsmechanismus, multimodalen Anweisungen und selektivem Lernen.

LIDA bedient sich der „Global Workspace Theory“ (kurz „GWT“). Die GWT ist das Konzept einer flüchtigen Speichermöglichkeit, in dem nur ein konsistenter Inhalt dominant sein kann. Dieser Ansatz ist besonders nützlich für neuartige Probleme, die nicht über eine bekannte Lösung verfügen. Diese Probleme können durch die Vernetzung zahlreicher spezialisierter Netze, von denen jedes einen Teilschritt darstellen kann, zu einer Lösung führen [BF09, p.2].

LIDA durchläuft kognitive Zyklen, Abbildung 2.10. Ein Zyklus kann in drei Phasen unterteilt werden [BF09, p.5]:

1) Verständnis-Phase (*understanding phase*)
   Während der Verständnis-Phase aktivieren ankommende Signale die Erkennung in der S*ensory Memory* bezüglich der Erkennungsmerkmale auf Low-Level Basis. Die Ausgabe von der

[4] Unter Abwehrmechanismen können Funktionen, die die Daten so verändern, dass sie nicht mehr gegen Regeln verstoßen, verstanden werden.

Sensory Memory wird der *Perceptual Associated Memory* weitergereicht und durchläuft die Erkennung eine höhere Ebene. Die Ausgabe davon kann in Form von Kategorien, Handlungen, Objekten und Ähnlichem erfolgen.
Das daraus resultierende Wahrnehmungsobjekt wird in den Workspace gestellt, wo es die *Transient Episodic Memory* und die *Declarative Memory* dazu bringt, lokale Assoziationen zu erzeugen. Aus all dem ergibt sich ein Modell der momentanen Situation. Aufmerksamkeits-Codlets schalten die Bewusst-Werden-Phase ein.

2) Bewusst-Werden-Phase (*consciousness phase*)
Aufmerksamkeits-Codlets gruppieren die hervorstechenden Elemente und leiten diese bewusst gewordenen Objekte, Handlungen, Kategorien udgl. dem Global Workspace (GW) weiter.
Im GW durchlaufen die Daten einen Wettbewerb der Aufmerksamkeit. Das mehrfache Ergebnis – da verschiedene Kriterien und Eigenschaften darauf Einfluss haben – wird verbreitet (broadcasted).

3) Handlungsauswahl- und Lern-Phase (*action selection phase*)
In der Handlungsauswahl- und Lern-Phase laufen mehrere Prozesse gleichzeitig ab: Wenn die Daten neu aus der „Bewusstwerden Phase“ ankommen, werden neue Entitäten und Assoziationen erzeugt, die alten werden bei Wiederholung verstärkt. Mögliche Handlungsschemata werden zusammen mit ihren Kontexten und den erwarteten Ergebnissen aus der Bewussten-Werden Phase in das Procedural Memory gelernt. Eine Kopie jedes Handlungsschemas wird mit konkreten Werten instanziiert und zu der Handlungs-Phase (*Action Selection*) geschickt. Dort wird aus den konkurrierenden Schemata eines als Basis für das Verhalten dieses kognitiven Zyklus ausgewählt. Das gewählte Verhalten wählt im sensorisch-motorischen Gedächtnis einen geeigneten Algorithmus aus, dieser erzeugt das Verhalten, das für die Ausführung zuständig ist. Die Ausführung schließt den kognitiven Zyklus ab.

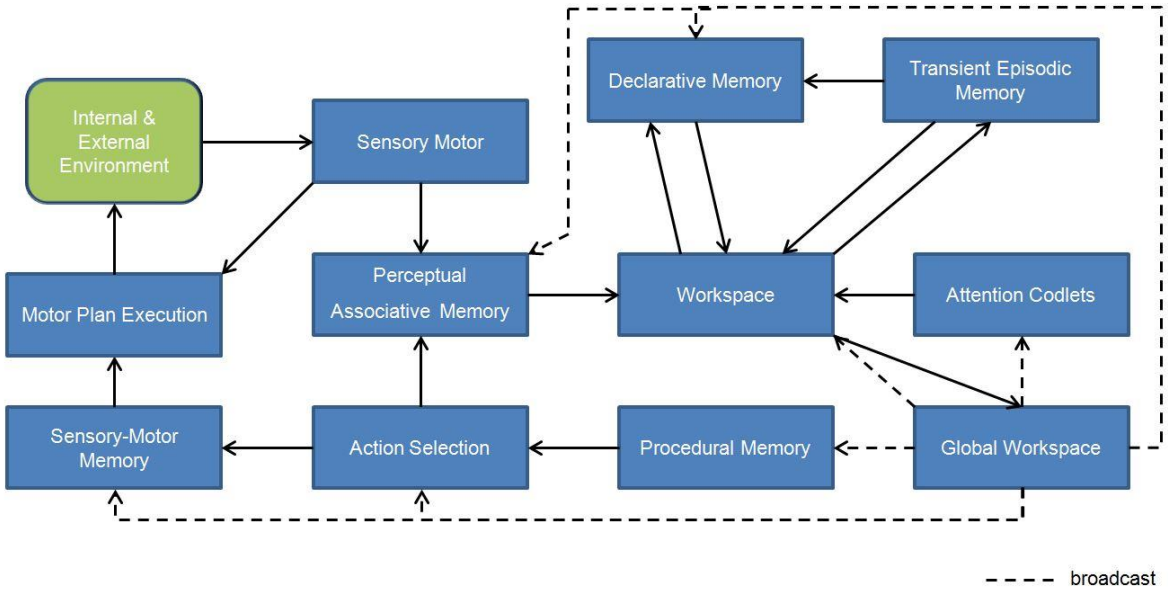


Abbildung 2.10: LIDA Zyklus [BF09, p.4] - modifiziert nachgezeichnet

LIDA verwendet Konzepte wie Emotionen und Gefühle für die Bewertung einer Situation. Verglichen zu andere Architekturen beinhaltet LIDA eine vorgelagerte und bewusste Stelle im System, bei der Daten vorverarbeitet werden: Im Rahmen der erwähnten Vorverarbeitung wird eine Teilmenge der Daten durch einen Aufmerksamkeits-Mechanismus „bewusst“. Diese Auswahl wird nun an alle angeschlossenen Module gesendet (broadcast) um mögliche Optionen für eine Handlung zu aktivieren [GOS14, pp.156 -157].

Unterschiede zum SiMA Modell

Eine Stärke von SiMA im Vergleich zu LIDA sind schließlich die höher entwickelten, bzw. menschenähnlicher definierten und konzeptionierten kognitiven Funktionen: LIDA basiert hauptsächlich auf der Anwendung verschiedener Memory-Arten (Erinnerungsarten) im GW, während SiMA ausgereiftere Konzepte wie Abwehrmechanismen, Triebe, erweiterte Emotionen und (noch nicht implementierte) Fantasy Actions ermöglicht.

Verglichen zum SiMA-Modell hat LIDA eine Menge an irrelevanten Bewusstseinsinhalten, die ständig mit den relevanten Inhalten um die Auswahl konkurrieren. Im SiMA-Modell gibt es keinen irrelevanten Bewusstseinsteil, sodass auch die beschriebene Konkurrenzsituation wegfällt. Der irrelevante Bewusstseinsinhalt wird im SiMA-Modell insofern durch Abwehrmechanismen verhindert, als Regeln deklariert werden und irrelevante Inhalte beim Durchlauf dieser Regeln wegfallen.

Als systemischer Unterschied wäre anzuführen, dass SiMA eine fixe Zyklussequenz hat. LIDA ist demgegenüber von Haus aus für einen asynchronen Ablauf konzipiert, d.h. die einzelnen Prozesse werden abhängig von Bedingungen aktiviert. SiMA verwendet im Unterschied zu LIDA die GWT nicht und nutzt somit keinen broadcast, sondern leitet die Daten von Modul zu Modul.

2.3.3 BDI

Kurzdarstellung des BDI-Modells

BDI ist ein Software Agent und steht für „*Belief Desire and Intention*“. Dies sind die drei Hauptbestandteile der hier beschriebenen Architektur, siehe Abbildung 2.11. Der Agent bzw. die BDI-Agenten werden mit Annahmen über ihre Umwelt und Wissen über den Zielzustand inkl. der Absichten, wie dieser Zustand zu erreichen ist, ausgestattet. Das Hauptziel eines BDI-Agenten ist, ein Handeln zu wählen, um seine Begierde zu befriedigen [RG95, p. 312]. Um dies zu erreichen analysiert der Agent zuerst seine Ziele und leitet ab, welche noch nicht befriedigt wurden. Danach bestimmt der Agent, welche Ziele als nächstes folgen würden und wählt einen passenden Plan von den für verschiedene Situationen vorhandenen Plänen aus. Ein solcher Plan beinhaltet eine Vorstellung darüber, wie die jeweilige Absicht erzielt werden kann und welche Schritte dafür gesetzt werden müssen. Letztendlich verifiziert der BDI Agent, ob das geplante Vorgehen mit seinen Wünschen übereinstimmt [BACR08, p. 38].

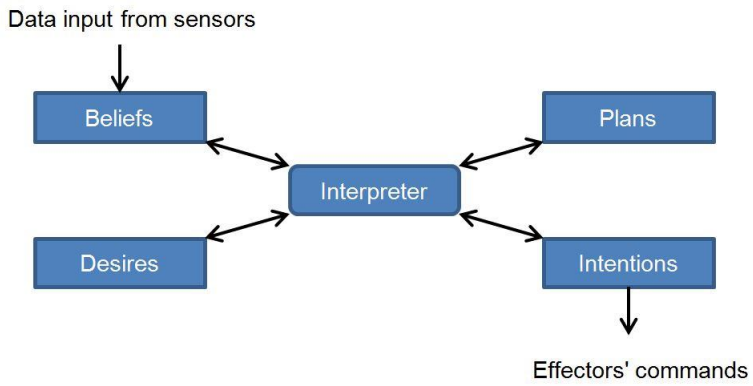


Abbildung 2.11: BDI Architektur [Wool96, p.2] nachgezeichnet

Wie bereits erwähnt, besteht die BDI-Architektur aus drei Bausteinen: (1) Der Baustein „Belief“ liefert dem BDI Agenten wesentliche Informationen über sich selbst, über die Umgebung und – wenn vorhanden – über andere Agenten. (2) Der Baustein „*Desire und Plans*“ beinhaltet die notwendigen Motivationen des BDI Agenten, um eine Aktion in Gang zu setzen und damit das gewünschte Ziel zu erreichen. (3) Der Baustein „Intention“ wird situationsbedingt gewählt. Der Agent besitzt diesbezüglich verschiedene Pläne für verschiedene Situationen. Die erwähnten Pläne beschreiben, wie die gefasste Absicht erreicht werden kann und welche Schritte konkret gesetzt werden müssen.

Unterschiede zum SiMA Modell

Die Entscheidungsfindung in BDI basiert auf die Verifizierung aller eingetragenen Pläne. Dies macht das System „Plananzahl“-abhängig und kostet im Vergleich zum SiMA-Modell sehr viel CPU-Zeit. Verglichen zu SiMA kann BDI ferner mit Situationen, die nicht vorgesehen oder beschrieben wurden, nicht umgehen.

2.3.4 VOLITRON

Kurzdarstellung des VOLITRON-Modells

Volitron ist eine Steuerung für Roboter und wurde von A. Buller entwickelt. Diese Steuerung steigert die Fähigkeit einer selbstinszenierten Erforschung der Umwelt und unterstützt die neue Zielsetzung, Planung und Ausführung von Handlungen unter Berücksichtigung der Verhaltensmuster der Zielobjekte [Bull02, p.1].

Die Volitron-Struktur besteht aus vier Hauptelementen.

1) Ein Modell der wahrgenommenen Realität (*Perceived Reality*)
2) Ein Modell der angestrebten Realität (*Desired Reality*)
3) Ein Modell der idealen Realität (*Ideal Reality*) und
4) Ein Modell der erwarteten Realität (*Anticipated Reality*)

Ein Konflikt entsteht aus den Unterschieden zwischen der wahrgenommenen und der angestrebten Realität. Er stellt die Hauptquelle der Motivation in Richtung Handlung (Action) dar. Die endgültige Entscheidung, eine Handlung auszuführen, ist auf den Vergleich mit dem Modell der idealen Wirklichkeit mit dem der erwarteten Wirklichkeit zurückzuführen.

Einer der Aufgaben des Arbeitsspeichers des Roboters ist es, sich selbst in die Rolle des anderen zu versetzen und das als Ziel der angestrebten Realität zu analysieren.

Volitron bedient sich dabei mit fünf Arten von Erinnerungen.

1) das *Perceptual Representation* System,
2) die *Procedural Memory*,
3) die *Semantic Memory*,
4) die *Episodic Memory* und
5) die *Working Memory* („WM“)

Während die ersten vier Erinnerungsarten dafür zuständig sind, Informationen über längere Dauer abzuspeichern, dient die WM in erster Linie zur Änderung des Inhaltes des Speichers von den anderen vier Arten von Erinnerungen. Buller schlägt vor, dass die Informationsstücke „Memes“ genannt werden. Memes treten in der WM als mehrfachen Kopien auf. Ein Bestand der Memes dient etwa der Zufriedenheit, während ein anderer Bestand der Unzufriedenheit dient. Die Informationsstücke werden als Nebenprodukte der WM Aktivität erzeugt. Basierend auf dem numerischen Gleichgewicht des Zufriedenheits- und Unzufriedenheitsbestandes wird ein Maß an Spannung berechnet. Volitron versuchet kontinuierlich dieses Maß an Spannung so niedrig wie möglich zu halten [Bull02, p.1].

Der Unterschied zwischen der wahrgenommenen Realität und der angestrebten Realität wird als Hauptgrund für das Auftreten der Meme im Bereich der Unzufriedenheit vorgeschlagen. Alle erworbenen Kenntnisse der empfundenen und erwünschten Realität sind im semantischen Gedächtnis gespeichert, welches statisch ist [Bull02, p.1].

Volitron bedient sich Abwehrmechanismen, um die Memes in der WM zu verändern. Sie treten in Kraft, wenn die Veränderungen der Umwelt zu keiner Reduktion der Spannung geführt haben oder erst gar keine umzusetzende Aktion ausgewählt werden kann. Das erschafft eine Verzögerung in der Ausführung. Gemäß der Psychoanalyse werden Verteidigungsmechanismen eingesetzt, um Konflikte zu lösen. Deswegen müssen Regeln in Volitron deklariert werden, wobei festzuhalten ist, dass ihre Verletzung Konflikte verursacht, die zur Aktivierung von Verteidigungsmechanismen führen. Die von [Bull02, pp.2-3] implementierten Abwehrmechanismen sind: Verdrängung, Projektion, Verleugnung, Rationalisierung und Sublimierung. Sexuelle oder aggressive Triebe wurden nicht in die Architektur implementiert. Die Identität und Art des jeweils aktivierten Mechanismus und die Abhängigkeit untereinander werden nicht angegeben.

Unterschiede zum SiMA-Modell

Volitron ist im Gegensatz zum SiMA-Modell, nicht in der Lage ein internes Modell einer vollständigen Szene im WM zu kreieren. Ferner sind keine Simulationsergebnisse oder Ansätze zur Implementierung des Modells vorhanden [Bull02, p.4].

3. Modell und Konzepte

Im vorigen Kapitel wurden das SiMA Projekt und andere als kognitiv geltende AGI's-Architekturen vorgestellt. Es wurde darauf eingegangen, welche Inhalte bereits bei den einzelnen Architekturen umgesetzt wurden und wo alle diese kognitiven Architekturen ihre Grenzen haben. Das nun folgende Kapitel wird die im Zuge der vorliegenden Diplomarbeit neu entwickelten Modelle und Konzepte zur Verbesserung dieser existierenden Strukturen vorstellen. Es wird aufgezeigt werden, an welche existierenden Konzepte angeknüpft wird und wie der Entwicklungsprozess zu den neuen Ansätzen ausgestaltet ist.

Dieses Kapitel beinhaltet drei Unterkapitel. Im Unterkapitel „Grundlegende Konzepte im SiMA Modell", werden die grundlegenden und für die Arbeit relevanten Konzepte im SiMA Modell vorgestellt. Im zweiten Unterkapitel „Anknüpfungspunkte im SiMA-Modell", wird erklärt, wo in die Architektur angeknüpft wird. Das letzte Unterkapitel „Neue Konzepte" beschreibt die neu einzuführenden Konzepte.

3.1 Grundlegende Konzepte im SiMA Modell

Wie im Abbildung 2.6 visualisiert, ist das SiMA-Modell in einen Primärprozess und einen Sekundärprozess aufgeteilt. Diese grundlegende Dualität spiegelt sich auch in den hier vorgestellten Konzepten wieder und soll vorab kurz erläutert werden. Danach werden die Datenstrukturen des Primärprozesses und Sekundärprozesses vorgestellt, gefolgt von einer Erläuterung der Bewertungsmechanismen in beiden Prozessteilen. Detaillierte Informationen können im [DBD+14] und in aktuellen Dissertationen [4] nachgelesen werden.

3.1.1 Unterscheidung Primärprozess und Sekundärprozess

Die Entscheidungsfindungseinheit im SiMA-Modell, siehe Abbildung 2.5, ist in zwei Bereiche aufgeteilt, den Primärprozess und den Sekundärprozess. Die Daten durchlaufen immer zuerst den Primärprozess, werden danach durch den Sekundärprozess erweitert und führen gegebenfalls zu einer Handlung. Die Daten im Primärprozess sind unbewusst, teilweise schon vorbewusst, die Daten im Sekundärprozess sind ebenfalls teilweise noch vorbewusst oder bereits bewusst.

Im ***Primärprozess*** haben die Daten weder eine zeitliche noch eine hierarchische Ordnung, sie haben auch keine Kausalität. Schlussfolgerungen von einer Tatsache auf die nächste sind daher in diesem Stadium noch nicht möglich. Die im Modell verwendeten Funktionen sind dementsprechend auch sehr einfach gehalten, woraus sich ergibt, dass der Primärprozess sehr schnell ist und einen Verhaltensvorschlag zu einer Situation fast unmittelbar liefert. Diese Schnelligkeit führt zu einem undurchdachten,

impulsiven Vorschlag. Dieser Umstand ist aus psychologischer Sicht damit zu erklären, dass der Primärprozess nach dem Lustprinzip funktioniert, d.h. dass sich der Agent im Primärprozess immer für das, was ihm unmittelbar am meisten Lust bringt, entscheiden würde, auch wenn diese Wahl auf lange Sicht keine gute Idee darstellen würde.

Der ***Sekundärprozess*** erweitert die Ergebnisse dieser grundlegenden, im Primärprozess vorgenommenen Bewertung dahingehend, dass er zusätzliche Informationen wie die bereits erwähnte Kausalität, die hierarchische Ordnung und die zeitliche Abfolge heranzieht und diese danach durch komplexere Funktionen und auch nach anderen Kriterien – wie z.B. der Unlust[5] – bewertet. Dadurch entstehen deutlich komplexere Strukturen wie langfristige Pläne und ist der Agent mitunter bereit, vorab Unlust zu verspüren, wenn er in der Folge mit einem Zugewinn der Lust rechnen kann. Die Funktionen, die im Sekundärprozess verwendet werden sind umständlicher, komplexer und dadurch auch langsamer.

Zusammenfassend kann man sagen, dass der Primärprozess eine erste Auswahl trifft, die der Sekundärprozess genauer bewertet und mitunter auch verwirft.

In den folgenden Unterkapiteln werden die Datenstrukturen des Primärprozesses und des Sekundärprozesses sowie ihre Bewertungsmechanismen genauer vorgestellt.

3.1.2 Daten des Primärprozesses

Der Primärprozess besteht aus Funktionen, die mit unbewussten Daten arbeiten. Diese haben – wie im vorigen Kapitel erwähnt – keine Hierarche und keine zeitlichen Zusammenhänge, lediglich die Unterscheidung zwischen Gleichzeitigkeit oder Ungleichzeitigkeit spielt eine Rolle. Tabelle 1 zeigt eine Zusammenfassung der in diesem Unterkapitel vorgestellten Konzepte und Datenstrukturen.

Modellebene (Konzepte)	Kurze Beschreibung (Modellebene)	Implementierungsebene (Datenstrukturen)
Image	eine statische Szene (ein Moment der Realität)	Thing Presentation (TP) und Thing Presentation Mesh (TPM)
Szenario	eine dynamische Abfolge von Szenen	Thing Presentation Mesh (TPM)
Triebsymbol	ein körperliches Bedürfnis	Drive Mesh (DM)

Tabelle 1: Übersicht von Modell- und Datenebene im Primärprozess

Beschreibung der Konzepte auf Modellebene

[5] Wie die Unlust zu einer Aktion führen kann, wird hier an einem Beispiel näher gebracht: der Agent ist hungrig und sieht einen Apfel. Im Primärprozess wird die Schlussfolgerung getroffen -> iss den Apfel. Der Sekundärprozess sieht sich die Umgebung an und stellt fest, dass der Apfel einem anderen Agenten gehört, der viel größer und stärker ist. Würde Agent1 den Apfel von Agent2 nehmen, könnte ihn Agent2 deswegen bestrafen, was zu großer Unlust führen würde.

Ein ***Image*** ist eine psychische Repräsentanz einer statischen Szene. Dies kann nicht nur visuelle Elemente enthalten, sondern auch olfaktorische, auditive, gustatorische und taktile. Es ist unveränderlich und hat keine zeitliche Ausdehnung. Auf der Implementierungsebene wird das Image als Netz aus Sachvorstellungsnetzen (TPM's) und Sachvorstellungen (TP's) umgesetzt.

Ein ***Szenario*** ist ein sehr kurzer Bewegungsablauf. Damit ist ein Symbol gemeint und nicht eine zeitlich ausgedehnte Bewegung. Dies wäre etwa mit der grafischen Darstellung von Bewegungen in Comics vorstellbar. Ein Szenario kann mit anderen Szenarien und Images assoziiert werden.

Ein ***Triebsymbol*** repräsentiert immer ein körperliches Bedürfnis. Dies kann auf zwei verschiedene Arten eingesetzt werden, entweder um ein aktuelles Bedürfnis wie z.B. „Hunger" zum Ausdruck zu bringen oder – verknüpft mit einer Aktion – um die erlebte Befriedigung eines Bedürfnisses darzustellen.

Beschreibung der Datenstruktur-Konzepte

Eine ***Sachvorstellung*** (TP – *Thing Presentation*) ist eine atomare Größe, die einen nicht weiter unterteilten Informationsblock repräsentiert. Als Beispiele wären etwa die Farbe „grün", das Symbol für die Bewegung „Springen" oder der Geruch „blumig" zu nennen.

Ein ***Sachvorstellungsnetz*** (TPM – *Thing Presentation Mesh*) setzt sich aus mehreren TP's und / oder TPM's zusammen. Es hat einen so genannten Identifier und einen Typ, der angibt, welche Art von Informationen es enthält. Das Sachvorstellungsnetz fügt die atomaren Größen, die zusammen gehören, zusammen und definiert eine gemeinsame Sache.

Eine ***Triebrepräsentanz*** (DM – *Drive Mesh*) repräsentiert immer einen Triebanspruch. Dies kann auf zwei Arten eingesetzt werden. Wenn sie alleine steht, ist es die Darstellung eines körperlichen Bedürfnisses wie z.B. „Hunger". Wenn die Triebrepräsentanz dagegen mit einem Sachvorstellungsnetz verknüpft ist, stellt sie dar, wie gut die jeweilige Sachvorstellung einen bestimmten Trieb befriedigen kann (laut Erinnerungen und Erwartungen).

Die eben dargestellten Datenstrukturen bilden im Primärprozess ein großes zusammenhängendes Netz, dessen Aufbau in Abbildung 3.1 beispielhaft veranschaulicht wird.

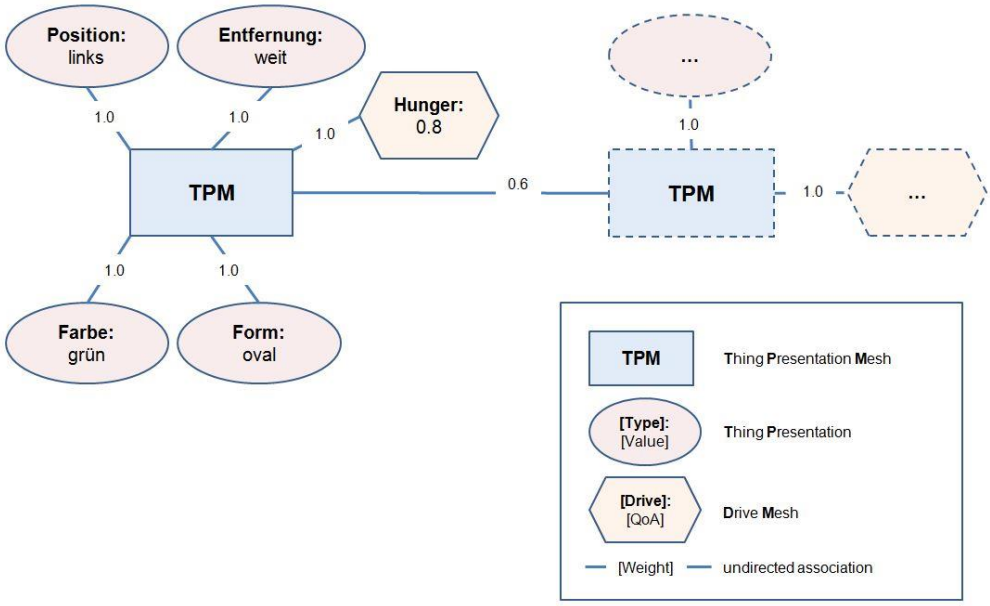


Abbildung 3.1: [DBD+14, p.87], Sachvorstellungsnetz - modifiziert nachgezeichnet

Die obige Grafik stellt exemplarisch ein TPM für ein Objekt Apfel in der Wahrnehmung dar. Das Wort „Apfel“ kommt deswegen nicht vor, weil SiMA im Primärprozess noch keine Wortvorstellungen verwendet. Das Objekt ist also nur durch seine Eigenschaften (im obigen Beispiel grün und oval) definiert. Die Verbindungen zwischen den einzelnen Elementen sind Assoziationen. Diese verbinden die Elemente mit einer bestimmten Verbindungsstärke von 0.0 bis 1.0. Im obigen Beispiel sind die meisten Verbindungsstärken 1.0, da es sich um ein TPM aus der Wahrnehmung handelt, bei dem die Eigenschaften unzweifelhaft zugeordnet werden können. Abweichende Verbindungsstärken symbolisieren eine geringere Zugehörigkeit, das kann z.B. als Unsicherheit interpretiert werden. Ein Beispiel dazu wäre etwa der Wurm, der in dem Apfel stecken könnte.

3.1.3 Daten des Sekundärprozesses

Die Datenstrukturen des Sekundärprozess sind immer eine Erweiterung der Datenstrukturen des Primärprozesses. Das bedeutet, dass sie niemals allein für sich stehen können und sich immer auf Daten des Primärprozesses beziehen müssen. Der Sekundärprozess verarbeitet vorbewusste und bewusste Informationen und weist – im Gegensatz zum Primärprozess – Hierarchien und komplexere zeitliche Zusammenhänge wie z.B. „hintereinander“ oder „bedingt durch“ auf. Tabelle 2 zeigt eine Zusammenfassung der in diesem Unterkapitel vorgestellten Konzepte und Datenstrukturen.

Modellebene (Konzepte)	Kurze Beschreibung	Implementierungsebene (Datenstrukturen)
Labelled Image	Statische Szene mit Wortvorstellungen	Word Presentation (WP) Word Presentation Mesh (WPM)

Akt	Abfolge von Szenen	Word Presentation Mesh (WPM)
Ziel	Befriedigungsmöglichkeit	Word Presentation Mesh (WPM)
Wortvorstel-lungssequenz (WPS)	Phrasen und Sätze	Word Presentation Mesh (WPM)

Tabelle 2: Übersicht von Modell und Daten im Sekundärprozess

Beschreibung der Konzepte auf Modellebene

Ein ***labelled Image*** entsteht, wenn alle Sachvorstellungen und Sachvorstellungsnetze eines Images mit ihren Wortvorstellungen (WP) verknüpft werden.

Ein ***Akt*** ist ein logisch nachvollziehbarer Planungsablauf welcher Images, Szenarien und andere Akte enthalten kann. Ein Akt kann darüber hinaus Ziele und markante Orte enthalten, die für die Situation relevant sind.

Ein ***Ziel*** ist die sekundärprozesshafte, stärker verknüpfte Variante der primärprozesshaften Triebrepräsentanz. Ähnlich wie im Primärprozess kann sie ebenfalls auf zwei Arten eingesetzt werden: alleinstehend repräsentiert sie das Bedürfnis nach Befriedigung, in Assoziation mit einem *labelled Image* repräsentiert sie darüber hinaus die Möglichkeit der Befriedigung eines Triebwunsches.

Eine ***Wortvorstellungssequenz*** (WPS) repräsentieren Sätze oder Phrasen der natürlichen Sprache. Die Elemente der Sequenz haben eine zeitliche und oder logische Abfolge (Vorgänger, Nachfolger).

Beschreibung der Datenstruktur-Konzepte

Eine ***Wortvorstellung*** (WP – *word presentation*) ist das Sekundärprozessanalog zur Sachvorstellung und repräsentiert eine nicht weiter unterteilte Bezeichnung für eine Sachvorstellung z.B. „das Symbol für das Wort Hunger“[6].

Das ***Wortvorstellungsnetz*** (WPM – *word presentation mesh*) ist das sekundärprozesshafte Analog des Sachvorstellungsnetzes TPM im Primärprozess, d.h. es kann ebenfalls nicht alleine für sich selbst stehen und fasst zusammengehörige Wortvorstellungen zusammen.

Die folgende Abbildung 3.2 zeigt beispielhaft wie eine Erweiterung von Abbildung 3.1 im Sekundärprozess aussehen könnte:

[6] Beispiel zum Verständnis zur Unterscheidung von Wort- und Sachvorstellung: das Objekt Apfel in der Realität erzeugt die Sachvorstellung Apfel. Die Sachvorstellung des Apfels ist mit der Wortvorstellung „Apfel“ verknüpft, das aber vom Wort „Apfel“ unterschieden werden muss. Das Wort Apfel (Klang und die dazugehörigen Muskelbewegungen) sind eine Sachvorstellung, welche wiederum ihre eigene Wortvorstellung hat „das Wort Apfel“.

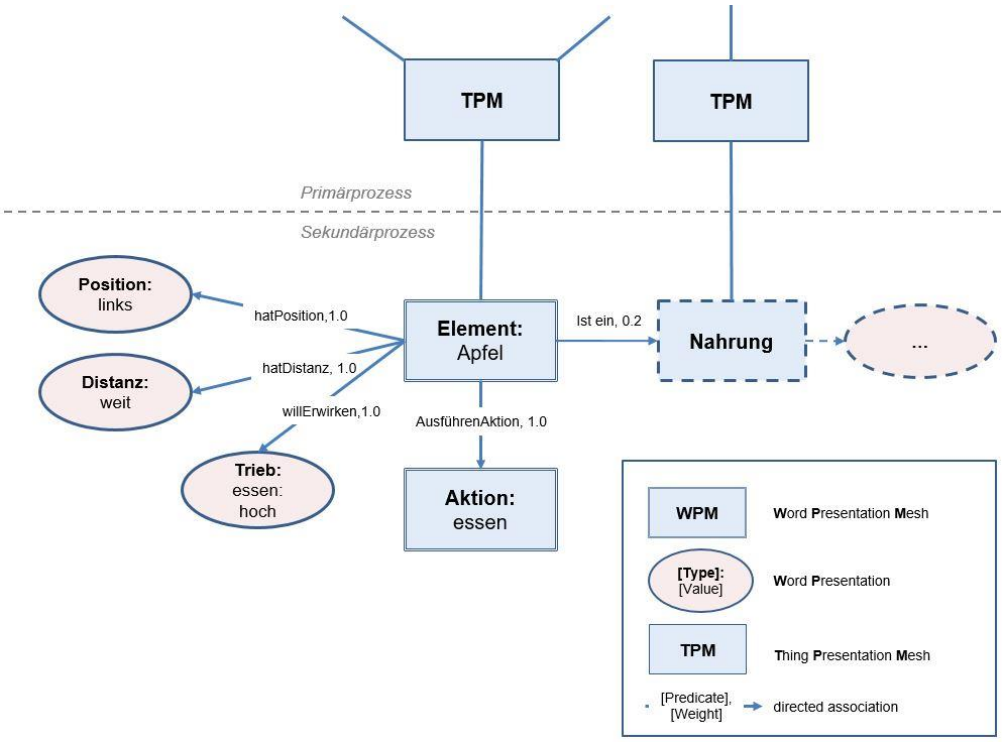


Abbildung 3.2: WPM (*Word-Presentation Mesh*)

Aus den ungerichteten Assoziationen des Primärprozesses wurden, wie die obige Abbildung zeigt, im Sekundärprozess gerichtete Assoziationen mit Prädikaten. Diese Prädikate beschreiben die Bedeutung der Assoziation z.B. „*has position*“. Im Beispiel wird gezeigt, dass der Apfel zur Gruppe Nahrung gehört, aber nicht der prominenteste Vertreter dieser ist, weshalb die Ass-Stärke 0,2 ist.

3.1.4 Bewertungsmechanismen des Primärprozess und Sekundärprozess

Bewertungsmechanismen bilden die Grundlage für die Entscheidung eines SiMA Agenten. Es gibt verschiedene Arten von Bewertungsmechanismen in SiMA, die verschieden ausgereift sind. Für die vorliegende Arbeit sind die Bewertungsmechanismen „Affektbetrag“, „Basis-Emotion“ und „Gefühl“ von Bedeutung. Über-Ich-Regeln können diese Bewertungsmechanismen als Bedingungen nutzen.

Die ***Affektbeträge*** bewerten Inhalte entsprechend des Lustprinzips. Sie repräsentieren wie viel Lust der Umgang mit dem Objekt ungeachtet möglicher Unlust oder Machbarkeit verspricht. Ein Beispiel für eine Über-Ich Regel, die diesen Bewertungsmechanismus nutzt wäre: „du darfst nicht gierig essen“. D.h. der Agent darf sich in seiner Wahrnehmung nicht zu viel Lust vom Essen versprechen, d.h. der Affektbetrag vom Essen soll niedrig gehalten werden.

Die ***Basis-Emotionen***, wie in Abbildung 3.3 dargestellt, erweitern die Bewertung durch den Affektbetrag um eine Repräsentanz des Gesamtzustands des Agenten. Dies wird erreicht, indem die aktuell aktiven Affektbeträge zur Bildung eines Emotionsvektors herangezogen werden. Dieser Vektor umfasst die Dimensionen „Lust/Pleasure“, „Unlust/Unpleasure“, „Aggressiv/Aggressiv“ und „Libidi-

nös/Libidinous“. Gespeist wird er spezifisch aus den Trieben, den Wahrnehmungen und den Vorstellungen. Bei der Wahrnehmung und der Vorstellung werden die entsprechenden vier Dimensionen direkt aus der Erinnerungen übernommen. Bei den Trieben bildet die Summe aller Affektbeträge aus aggressiven Trieben den aggressiven Teil des Vektors, die Summe der libidinösen Affektbeträge den libidinösen Teil des Vektors, die Summe der Affektbeträge aller Triebe die Unlust (d.h., wenn z.B. der Hunger steigt, steigt damit die Unlust) und die Summe aller abgeführten Affektbeträge (d.h. um wie viel sich der Affektbetrag im Vergleich zum letzten Zyklus verringert hat) die Lust.

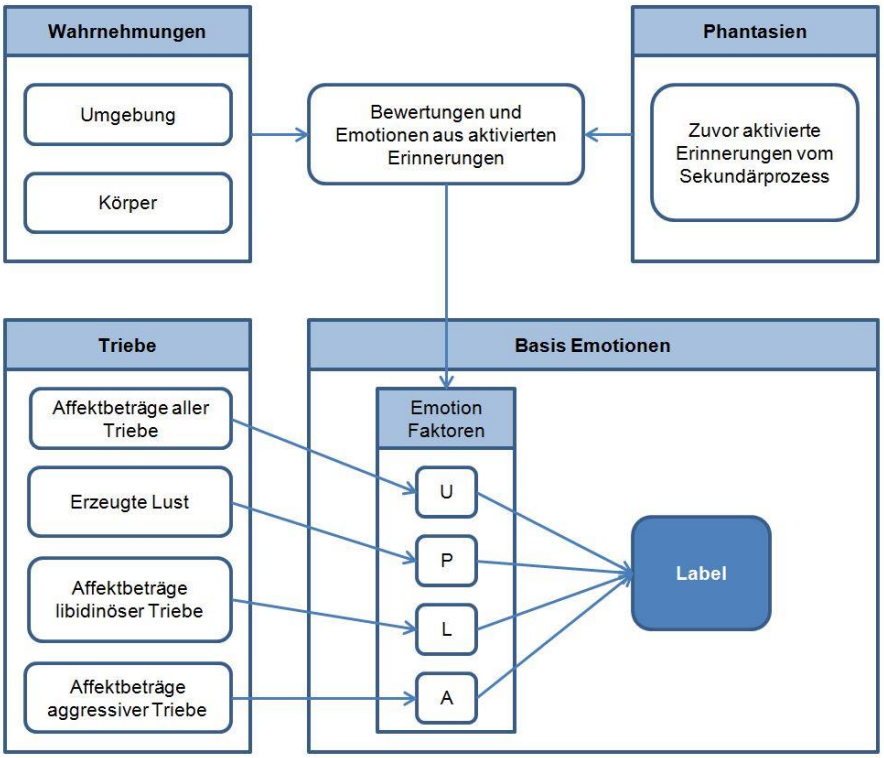


Abbildung 3.3 Bildung von Basis-Emotionen - nachgezeichnet [Scha16, p.106]

Basierend auf diesen vier Dimensionen wird der Basis-Emotion ein Label zugewiesen. Hierfür wird folgende Methode verwendet.

Dominanz von	Label
Lust	Freude
Unlust	Angst
Unlust + aggressiv	Wut
Unlust + libidinös	Trauer
Lust + libidinös	Sättigung
Lust + aggressiv	Hochgefühl

Tabelle 3: Zuweisung von Label einer Basis-Emotion

Ein Beispiel für die Verwendung dieses Bewertungsmechanismus in einer Über-Ich Regel ist: „du darfst nicht wütend sein“. Das impliziert, dass die Basisemotion „Wut“ nicht signifikant ausgeprägt sein darf.

Die ***Gefühle*** sind eine Erweiterung der Basis-Emotionen und sind die sekundärprozesshafte Repräsentation der Emotionen, die um sekundärprozesshafte-Informationen wie Wortvorstellungen und logische Zusammenhänge angereichert wurden. Durch das Bewusstwerden der Emotion als Gefühl kann über den emotionalen Zustand reflektiert und dieser Zustand verbal kommuniziert werden.

Ein Beispiel für die Verwendung dieses Bewertungsmechanismus in einer Über-Ich Regel ist: „du darfst deinen Bruder nicht hassen“. Dies impliziert, dass das Gefühl „Hass“ nicht auf den Bruder gerichtet sein darf.

Die Bewertungsmechanismen sind konzeptionell unterschiedlich, werden aber alle in der Implementierung auf einen Zahlenwert zwischen 0.0 und 1.0 transformiert. Im Falle von Basis-Emotionen handelt es sich hierbei um einen Vektor aus mehreren Zahlenwerten zwischen 0.0 und 1.0.

3.2 Anknüpfungspunkte im SiMA-Modell

Nachdem zuvor die grundlegenden Konzepte des SiMA-Modells vorgestellt wurden, werden in diesem Kapitel ausgewählte weitere spezifische Konzepte vorgestellt, die für das Verständnis der in dieser Arbeit im Kapitel 3.3 neu zu entwickelnden Konzepte relevant sind. Das vorliegende Kapitel ist in zwei Unterkapitel aufgeteilt. In Unterkapitel 3.3.1 werden die existierenden Abwehrstrukturen und ihre Interfaces zum existierenden SiMA-Modell vorgestellt, auf denen die neuen Über-Ich-Regeln aufbauen. In Unterkapitel 3.3.2 wird auf die existierenden Anknüpfungspunkte für eine neue Inventar-Funktion eingegangen.

3.2.1 Abwehr

Abwehr bezeichnet in SiMA ein bionisch inspirierten Filtermechanismus, der dafür verantwortlich ist, die fest vorgegebenen und verinnerlichten Regeln mit den Wünschen/Zielen des Agenten in Einklang zu bringen.

Das SiMA-Modell ist in funktionale Schienen unterteilt. Diese wurden bereits in Unterkapitel 2.2.3 beschrieben. In diesem Kapitel werden die Module und Schnittstellen genauer beschrieben, die für die vorliegenden Verbesserungen relevant waren. Die ausführlichen psychoanalytischen Grundlagen der vorgestellten Konzepte und eine genauere Beschreibung aller Module können in [DBD+14, pp.60-96] nachgelesen werden. Die Erweiterung der Abwehr, als eine der funktionalen Schienen, ist der Kernpunkt dieser Arbeit, weshalb im Folgenden alle Module und Interfaces der Abwehrschiene als zentraler Ausgangspunkt für die neuen Konzepte genauer vorgestellt werden.

In der folgenden Abbildung 3.4 wird nochmal in vereinfachter Form der bereits vom State of the Art, Kapitel 2.2.3 bekannte Zyklus der Positionierung der Abwehr im Gesamtmodell angezeigt. Die Abwehr befindet sich im hell hervorgehobenen Bereich im Layer 3 (L3).

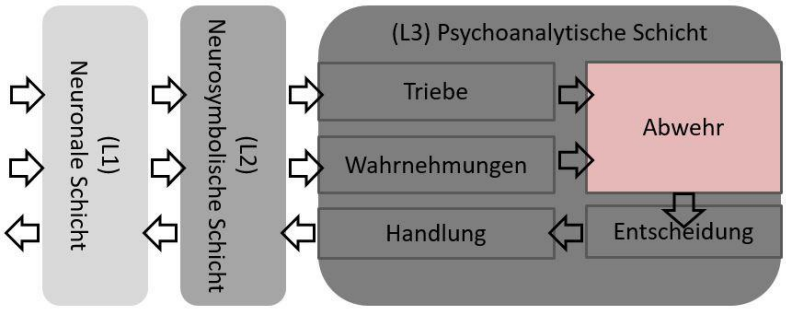


Abbildung 3.4: Den kognitiven SiMA-Prozess - stark vereinfacht

Die nächste Abbildung 3.5 enthält eine Detaildarstellung der Module der Abwehrschiene (rosa Bereich in der Abbildung 3.4), ihrer Interfaces sowie der Anknüpfungspunkte an die anderen Schienen. Darin wird die Stellung der Abwehrschiene als Vereinigung der Trieb- und der Wahrnehmungsschiene, sowie ihre Positionierung als letzte Schiene im Primärprozess vor der Umwandlung in den Sekundärprozess dargestellt.

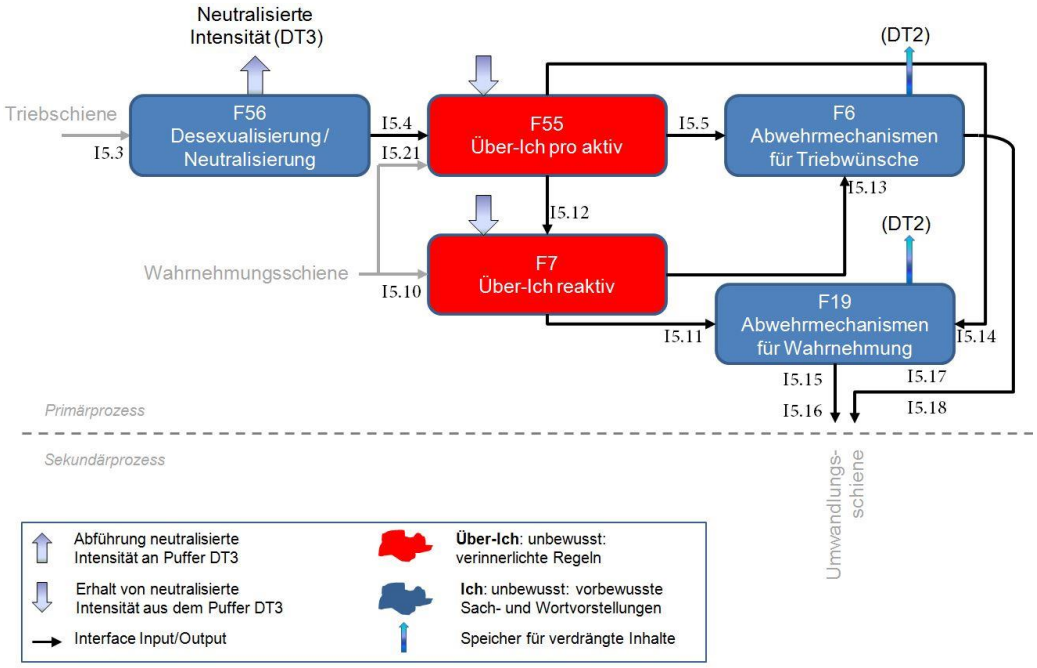


Abbildung 3.5: Module und Interfaces der Abwehrschiene

In erster Stufe lassen sich die dargestellten Module in Abbildung 3.5 in drei Kategorien aufteilen: Die erste Kategorie ist die Verarbeitung der Trieb-Daten in Modul F56, danach erfolgt eine Konflikterkennung in den Modulen F55 und F7, gefolgt von der Konfliktbehebung in den Modulen F6 und F19. Die zuletzt angesprochenen Konfliktbehebungs-Module sind bereits Teil vom ICH.

Verarbeitung

Das **Modul F56** führt abhängig von der aktuellen Trieblage, einen Teil der Affektbeträge von den Triebrepräsentanzen entsprechend zweier Persönlichkeitsparameter ab. Dieser abgeführte Wert nennt sich „neutralisierte Intensität“ und wird in dem Puffer DT3 abgespeichert (siehe Abbildung 3.5). Die darin abgespeicherte Intensität wird bei jedem Zyklus nach einem Algorithmus auf verschiedene Module, abhängig von ihrer Wichtigkeit und der noch vorhandenen Intensität neu verteilt.

Konflikterkennung

Das **Modul F55** repräsentiert den proaktiven[7] Teil des Über-Ich’s. Es ist verantwortlich für die Befolgung internalisierter Regeln, die – um Konflikte zu vermeiden – mit den Triebwünschen, Emotionen und Wahrnehmungen abgeglichen werden. Solche Regeln könnten lauten „sei immer freundlich zu anderen“ und sind momentan nur als hardgecodete, logische Operatoren implementiert.

Im **Modul F7** wird die Konfliktspannung berechnet, es repräsentiert den reaktiven[8] Teil des Über-Ich’s. Es ist verantwortlich für die Befolgung internalisierter[9] Regeln, d.h. hier werden Triebwünsche, Emotionen und Wahrnehmungen mit den internalisierten Regeln abgeglichen.

Konfliktbehebung

Das **Modul F6** beinhaltet die Abwehrmechanismen für die Daten, die aus der Triebschiene kommen. Es erhält eine Liste konflikthafter Triebwünsche von F7 und entscheidet – basierend auf Faktoren welche bei [Gelb15. pp.142-144] beschrieben sind – ob und wie der Konflikt aufgelöst werden soll. Es gibt folgende Arten, wie konflikthafte Triebwünsche[10] behandelt werden können:

- keine Abwehr – konflikthafte Triebwünsche können die Abwehr unverändert passieren;
- Abwehr durch Veränderung der Triebwünsche und/oder Affektbeträgen; oder
- teilweise oder vollständige Unterdrückung der Triebwünsche. Die unterdrückten Triebe, werden zum Modul F54 „Auftauchen verdrängter Triebinhalte“ zurückgeschickt.

Konkret sind folgende Abwehrmechanismen für Triebwünsche verfügbar [Gelb15, pp.50-51]:

- Verdrängung (*Repression*)
  Die Verdrängung ist eine ICH-Funktion, die Triebwünsche, die gegen die aktuellen Über-Ich Regeln verstoßen, ins Unterbewusste verschiebt und sie dort vom Bewussten fernhält. Als

[7] Die hier geprüften Regeln basieren auf Geboten. Im Gegensatz zu den reaktiven Regeln, die auf Verstößen basieren. Reaktiv Beispiel: „du darfst nicht beißen“, proaktives Beispiel: „jeden Tag eine gute Tat.“

[8] Die hier geprüften Regeln basieren auf Verstößen. Im Gegensatz zu den proaktiven Regeln, welche auf Geboten basieren. Reaktiv Beispiel: „du darfst nicht beißen“, proaktives Beispiel: „jeden Tag eine gute Tat.“

[9] Dies sind Regeln, die in der Kindheit geprägt wurden. Sie sind ausformuliert und sind so tief verwurzelt, dass der Betreffende nicht mehr darüber nachdenkt.

[10] Bezeichnet die naheliegenden, noch unbewussten Aktionen zur unmittelbaren Befriedigung eines Triebs.

Konsequenz daraus können die damit verbundenen Erfahrungen nicht bewusst wahrgenommen werden.

- Wendung gegen das selbst (*Turning Against the Self*)
  Dabei handelt es sich um einen „Vorgang, durch welchen ein Triebobjekt durch die eigene Person ersetzt wird." [SSK97, p55].
- Intellektualisierung (*Intellectualization*)
  Dieser Abwehrmechanismus versucht sich nur auf den intellektuellen, kognitiven Teil des Triebes zu konzentrieren und ignoriert den Affekt: Der Agent erzeugt verschiedene intellektuelle Erklärungen für den Trieb und vermeidet es dadurch, mit dem gefährlichen oder den vom ICH unerwünschten Teilen des Triebs konfrontiert zu werden.
- Reaktionsbildung (*Reaction Formation*)
  Im Zuge der Reaktionsbildung wird der Affektbetrag von der konflikthaften Handlung zu einer entgegengesetzten Handlung übertragen, um die konflikthafte Handlung dadurch im Bewusstsein zu überschatten.
- Verschiebung (*Displacement*)
  Der Abwehrmechanismus der Verschiebung lenkt den Triebwunsch auf ein neues Triebziel. Hier wird also auf ein Ziel der gleichen sozialen und kulturellen Ebenen umgelenkt.
- Sublimierung (*Sublimation*)
  Bei diesem Abwehrmechanismus wird der Triebwunsch auf ein neues, kulturell oder sozial akzeptableres Ziel umgelenkt.
- Verkehrung ins Gegenteil (*Reversal into the Opposite*)
  Bei diesem Vorgang wird das Triebziel in sein Gegenteil verkehrt. Dies geschieht entweder durch einen Übergang zwischen aktiv und passiv, oder durch eine inhaltliche Verkehrung von z.B. „Liebe" in „Hass".
- Projektion (*Projection*)
  Im Rahmen der Projektion werden konflikthafte Aspekte des Triebwunsches anderen Personen oder Objekten zugeschrieben und diesen bewusst unterstellt. Diesen Abwehrmechanismus gibt es sowohl für die Triebe als auch für die Wahrnehmung.

Das **Modul F19** ist auch ein Teil der Abwehrschiene und ist dem eben dargestellten Modul F6 ähnlich. Der Hauptunterschied ist, dass F19 die verbotenen Emotionen und Wahrnehmungen abwehrt. Es entscheidet somit, welche Wahrnehmungen und/oder Gefühle[11] in welcher Form bewusst oder vorbewusst werden dürfen.

Es gibt folgende Arten wie konflikthafte Triebwünsche behandelt werden können [Gelb15, pp.50-51]:

- keine Abwehr – konflikthafte Wahrnehmungen und oder Emotionen können die Abwehr unverändert passieren;
- Abwehr durch Veränderung der Wahrnehmungen und oder Emotionen; oder

[11] Die Bezeichnung meint in diesem Fall die bewusst werdende Form von Emotionen.

- teilweise oder vollständige Unterdrückung der Wahrnehmungen und oder Emotionen. Die unterdrückten Inhalte, werden zum Modul F35 „Auftauchen verdrängter Inhalte“ zurückgeschickt.

Konkret sind folgende Abwehrmechanismen für Emotionen verfügbar [Gelb15, p.51]:

- Affektverkehrung (*Reversal of Affect*)
  Bei der Affektverkehrung vollzieht sich die Umwandlung von aggressiven Affektbeträgen in libidinöse Affektbeträge und umgekehrt. Die Summe der Affektbeträge aus den libidinösen und aggressiven Quellen wird benutzt um die Erzeugung von Emotionen zu bestimmen. Affektverkehrung beeinflusst diesen Prozess, indem nicht der ursprüngliche Trieb verändert, sondern nur das Aufsummieren der entsprechenden Affektbeträge verändert wird.
- Wenden gegen das selbst (*Turning against the Self*)
  Bei diesem Vorgang wird ein Triebobjekt durch die Selbstrepräsentanz[12] ersetzt.

Konkret sind folgende Abwehrmechanismen für Wahrnehmungen verfügbar [Gelb15, pp.52-53]:

- Rationalisierung (*Rationalization*)
  Bei der Rationalisierung wird für einen konflikthaften Inhalt eine scheinbar rationale Erklärung konstruiert, die jedoch einer objektiven Betrachtung nicht standhalten würde. Mit anderen Worten versteht man unter Rationalisierung die Findung einer rationalen, logischen Erklärung für einen zur Wahrnehmung assoziierten Trieb, der eigentlich nicht erlaubt ist. Die Rationalisierung unterdrückt somit den Konflikt nicht inhaltlich, sondern findet eine passende Erklärung zur Rechtfertigung des Inhalts.
- Teilung (*Splitting*)
  Die Teilung ist ein primitiver Abwehrmechanismus, der auch beim Menschen in frühen Phasen der Persönlichkeitsentwicklung auftritt. Dieser Mechanismus hält die verschiedenen Introjektionen[13] eines Objekts getrennt. D.h. Splitting trennt die guten von den schlechten Eigenschaften eines Objekts. Auf diese Art und Weise kann ein Objekt oder eine Person als nur gut oder nur schlecht wahrgenommen werden. Die Wahrnehmung der ausschließlich guten Eigenschaften nennt sich „Idealisierung“, die Wahrnehmung der ausschließlich schlechten Eigenschaften nennt sich „Abwertung“. Wenn eine Person beispielsweise heute nett und morgen wütend ist, sorgt dieser Abwehrmechanismus für die Illusion, es mit zwei vollkommen unterschiedlichen Personen zu tun zu haben, da, es nicht möglich ist, bewusst die Eigenschaften freundlich und wütend zur selben Person zu assoziieren.
  Der Abwehrmechanismus verursacht in der frühen menschlichen Entwicklung etwa den Effekt, dass ein Baby die freundliche und die böse Mutter als zwei verschiedene Personen wahrnimmt. Splitting tritt gemeinsam mit anderen Abwehrmechanismen auf.

[12] Unter Introjektion wird die realitätsgerechte Vorstellung über sich selbst verstanden. Aus den Selbstrepräsentanzen bezieht ein Mensch seine Selbstdefinition.

[13] Ist eine Integration von Attributen eines Objekts oder Indiums in das unbewusste.

- Projektive Identifizierung (*Projective Indentification*)
  Mit projektiver Identifizierung ist gemeint, dass die eigenen konflikthaften Inhalte in eine andere Person projiziert werden.
- Abwertung (*Depreciation*)
  Die Abwertung ist – wie bereits beschrieben – die Variante des Abwehrmechanismus „Teilung“, bei dem nur die negativen Eigenschaften eines Objekts wahrgenommen werden.
- Idealisierung (*Idealization*)
  Die Idealisierung ist – wie ebenfalls bereits beschrieben – die Variante des Abwehrmechanismus „Teilung“, bei dem nur die positiven Eigenschaften eines Objekts wahrgenommen werden.
- Verleugnung (*Denial or disavowal*)
  Im Rahmen der Verleugnung wird ein unerwünschtes Stück der äußeren Realität mit Hilfe einer wunscherfüllenden Phantasie oder durch äußeres Verhalten zu unterdrücken versucht.
- Projektion (*Projection*)
  Die Projektion ist ein psychischer Abwehrvorgang in dessen Verlauf Gefühle, Wünsche und sogar innere Objekte, die anstoßerregend sind, aus dem subjektiven psychischen Raum eines Menschen und damit auch aus dem Bewusstsein ausgeschlossen werden. Die dergestalt ausgeschlossenen Inhalte werden dann einer anderen Person oder einem nicht belebten Objekt der Außenwelt zugeschrieben.

Interfaces

In diesem Unterkapitel werden die in Abbildung 3.5 dargestellten Verbindungen zwischen den Modulen aufgelistet und erklärt.

Das Interface **I5.3** ist ein Output von dem Modul F54 – „Auftauchen verdrängter Treibinhalte“ zum Modul F56 – „Desexualisierung/Neutralisierung“ und beinhaltet die aktuellen verdrängten Triebinhalte als DM.

Das Interface **I5.4** ist ein Output vom Modul F56 – „Desexualisierung/Neutralisierung“ zum Modul F55 – „Über-Ich proaktiv“ und beinhaltet die aktuellen Triebrepräsentanzen als DM.

Das **Sende-Interface zum Puffer DT3** ausgehend vom Modul F56 – „Desexualisierung/Neutralisierung“ führt neutralisierte Intensität an diesen ab.

Das **Empfang-Interface vom Puffer DT3** ist die Schnittstelle für die neutralisierte Intensität und liefert die Neutralisierungsrate zur Bestimmung der Ich-Stärke, die die Auswahl der Abwehrmechanismen beeinflusst.

Das Interface **I5.5** ist ein Output vom Modul F55 – „Über-Ich proaktiv“ zum Modul F6 – „Abwehrmechanismen für Triebwünsche“ und transferiert die aktuellen Triebrepräsentanzen als DM’s.

Das Interface **I5.10** ist ein Output vom Modul F18 – „Affektbeträge für Wahrnehmungen“ zum Modul F7 – „Über-Ich reaktiv“ und transferiert die aktuellen Wahrnehmungen als TPM aus der Wahrnehmungsschiene.

Das Interface **I5.11** ist ein Output vom Modul F7 – „Über-Ich reaktiv“ und leitet die erzeugte Liste von erkannten Emotions-Konflikten und eine Liste von erkannten Wahrnehmungs-Konflikten an das Modul F19 – „Abwehrmechanismen für Wahrnehmung“ weiter. Außerdem werden Wahrnehmungen als TPM und Emotionen nach ihrer Verwendung zur Konfliktbestimmung über das Modul F19 – „Abwehrmechanismen zur Wahrnehmung“ zur Umwandlungsschiene durchgereicht.

Das Interface **I5.12** transferiert die aktuellen Triebrepräsentanzen als DM mit assoziierten Wahrnehmungen und Erinnerungen als TPM’s aus der Triebschiene, weitergeleitet durch das Abwehr-Modul F55 – „Über-Ich proaktiv“ und die aktuelle Emotion aus dem Modul F63 – „Zusammenstellung von Emotionen“ (, das für die Emotionsgenerierung zuständig ist) an das Modul F7 – „Über-Ich reaktiv“.

Das Interface **I5.13** transferiert eine Liste der aktuell erkannten Triebkonflikte als Konfliktobjekte aus dem Modul F7 – „Über-Ich reaktiv“ und die aktuellen Triebrepräsentanzen als DM’s an das Modul F6 – „Abwehrmechanismen für Triebwünsche“. Außerdem werden die aktuellen Triebrepräsentanzen nach ihrer Verwendung zur Konfliktbestimmung über das Modul F6 „Abwehrmechanismen für Triebwünsche“ zur Umwandlungsschiene durchgereicht.

Das Interface **I5.14** ist ein angedachtes Interface ausgehend vom Modul F55 – „Über-Ich proaktiv“ zum Modul F19 – „Abwehrmechanismen für Wahrnehmungen“ und derzeit noch nicht implementiert.

Das Interface **I5.15** ist ein Output vom Modul F19 – „Abwehrmechanismen für Wahrnehmungen“ und schickt die durch die Abwehr veränderte aktuelle Wahrnehmung als TPM und Emotion an das Modul F21 – „Transformation in den Sekundärvorgang Wahrnehmung“ weiter.

Das Interface **I5.16** ist ein Output vom Modul F19 – „Abwehrmechanismen für Wahrnehmungen“ und schickt die durch die Abwehr veränderte aktuelle Emotion an dem Modul F71 – „Zusammenstellung von erweiterter Emotion“ weiter.

Das Interface **I5.17** ist reserviert für eine zukünftige Verbindung vom Modul F6 – „Abwehrmechanismen für Triebwünsche“ mit dem Modul F71 – „Zusammenstellung von erweiterten Emotionen“.

Das Interface **I5.18** ist ein Output vom Modul F6 – „Abwehrmechanismen für Triebwünsche“ und schickt die durch die Abwehr veränderten aktuellen Triebrepräsentanzen als DM‘s an das Modul F8 „Transformation in den Sekundärprozess (Triebwünsche)“ weiter.

Das Interface **I5.21** ist ein Output vom Modul F63 – „Zusammenstellung von Emotionen“ welches eine Liste von Basisemotionen an das Modul F55 „Über-Ich proaktiv“ schickt. Die möglichen Basisemotionen sind Freude, Trauer, Angst, Wut, Sättigung und Hochgefühl.

Das **Interface zum Puffer DT2** ist die Schnittstelle, über die das Modul F6 – „Abwehrmechanismen für Triebwünsche“ und das Modul F19 – „Abwehrmechanismen für Wahrnehmung“ verdrängte Inhalte zum Speicher für verdrängte Inhalte schicken. Der Speicher wird von den Modulen F54 – „Auftauchen verdrängter Triebinhalte“ und F35 – „Auftauchen verdrängter Inhalte“-Wahrnehmung ausgelesen.

Das hier vorgestellte Konzept ist eine erste Version, die in Zukunft für den Einsatz mit proaktiven Über-Ich Regeln erweitert werden kann.

3.2.2 Inventarkonzept

Auch wenn sich das Projekt SiMA auf die Entscheidungsfindung des Agenten konzentriert, ist es für mache Szenarien, wie im Kapitel 6.3, unerlässlich, dass auch Objekte im Simulator gewisse Funktionen bieten. Zum Beispiel ist es notwendig, dass Objekte (dies wird meistens von Objekten vom Typ „Agent“ vorkommen) die Möglichkeit haben, andere Objekte mit sich herumzutragen und zu sammeln. Da es sich beim Inventar um eine Funktionalität des virtuellen Körpers aus der Simulation handelt, ist es in Layer 0 definiert, siehe dazu Kapitel 2.2.3, Abbildung 2.3, und interagiert mit der Entscheidungsfindungseinheit nur über die Interfaces von Layer 1 und Layer 2. Die Interaktion von der Entscheidungsfindungseinheit zu den Objekten erfolgt über Aktionen, die Kommunikation von den Objekten zu der Entscheidungsfindungseinheit erfolgt über die Wahrnehmung. Da auch nicht-Agenten Inventar-Funktionen benutzen können sollen, ist es vorgesehen die Inventar-Funktionalität in der Oberklasse für alle mobile Entitäten einzugliedern. Die entsprechenden Anknüpfungspunkte sind bereits vorhanden. Die Nutzung der einzelnen Komponenten ist in der Abbildung 3.6 skizziert. Innerhalb einer Entität liefert der Memory (Erinnerungen) die Grundlage für die Entscheidungsfindungseinheit. Diese Formuliert Befehle an den Körper, welche wiederum das Inventar, nach Bedarf, zur Erfüllung seiner Aufgaben nutzt.

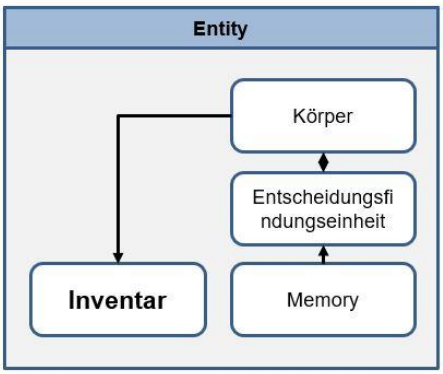


Abbildung 3.6: Positionierung des Inventars beim Agenten

Die Interaktion zwischen der Entscheidungsfindungseinheit und den Objekten im Inventar lässt sich folgendermaßen genauer darstellen:

- Die Rückmeldung vom Inventar zur Entscheidungsfindungseinheit erfolgt über dieselben Mechanismen wie bei allen Simulationsobjekten über die Wahrnehmung. Dies beinhaltet die Umgebungsinformationen aus dem Sensoren im Modul F10 „Sensoren Umgebung“ (Layer1) und die Umwandlung in Neurosymbole im Modul F11 „Neurosymbolisierung Umgebung" (Layer2), siehe dazu das Unterkapitel „*Umgebungswahrnehmungsschiene*“.
- Die Interaktion von der Entscheidungsfindungseinheit zum Simulationsobjekten erfolgt über Befehle an den simulierten Körper wie etwa. „gehe gerade aus“ oder „hebe auf“. Der betreffende Mechanismus ist vorgesehen, aber die ausführenden Befehle sind noch nicht implementiert. Wenn diese realisiert wurden, werden sie zuerst im Modul F31 „Neurodesymbolisierung-Handlungsanweisung“ (Layer 2) in Neurosymbole umgewandelt und dann im Modul

F32 „Aktoren für Muskeln“ (Layer 1) als Befehle für die simulierten Muskeln ausgegeben, siehe dazu das Unterkapitel „*Aktuator Schiene*“.

3.3 Neue Konzepte

In diesem Unterkapitel werden die im Zuge dieser Diplomarbeit neu entwickelten Konzepte zur Verbesserung und Erweiterung des SiMA-Modells detaillierter vorgestellt. Das Kapitel ist in drei Teilbereiche aufgeteilt. Der erste Teilbereich, das Kapitel „Über-Ich-Regeln“, ist der zentrale Punkt dieser Diplomarbeit und darin wird ein neues Konzept zur Definition von Über-Ich-Regeln in einer neu entwickelten Syntax beschrieben, die es erlaubt die Struktur natürlich sprachigen Regeln auf einfacher weise, für nicht Programmierer, nachzubilden. Es werden die Anknüpfungen zu existierenden Abwehrkonzepten, sowie die neu entwickelte Regelsyntax beschrieben. Im zweiten Teilbereich, das Kapitel „Inventar“, wird das neue Konzept zur Handhabung und Lagerung von Objekten beschrieben. Im dritten Teilbereich des Kapitels „Visualisierungs-Konzepte“ werden neu entwickelte Visualisierungen sowohl der neu hinzugekommenen Konzepte, als auch Erweiterungen existierender Visualisierungen beschrieben.

3.3.1 Über-Ich-Regeln

Die Über-Ich-Regeln sind ein Teil der Abwehr. Diese stellt in SiMA einen bionisch inspirierten Filtermechanismus dar. Die Aufgabe dieses Filtermechanismus ist das Lösen von Konflikten die aus dem Zusammenspiel der verschiedenen Anforderungen, die auf das System einwirken, entstehen. Sie teilen sich in zwei Teile, wobei der eine Teil Konflikte anhand von Über-Ich Regeln (siehe dazu weiter unten) identifiziert und der andere Teil sich mit dem Lösen dieser Konflikte mittels Abwehrmechanismen beschäftigt. Vor dieser Arbeit gab es im SiMA keine Schnittstelle zur Definition von Über-Ich Regeln, weshalb sie bisher nur als Programmcode definiert werden konnten.

Über-Ich Regeln stellen gelernte internalisierte[14] Regeln dar und werden von Psychoanalytiker für jedes Simulations-Szenario gesondert definiert. Die Simulation mit einem SiMA-Agenten läuft stets nur eine vordeterminierte kurze Zeit, weshalb der Agent sich keine Regeln internalisieren kann. Der Internalisierungsvorgang würde über Jahren hinweg andauern. Eine Internalisierte Regel könnte lauten „du darfst nicht schlagen“. Zurzeit sind nur eine Handvoll Über-Ich-Regeln von Programmierer direkt im Programmkode implementiert. Wie im Paper [SWK+15] beschrieben, ist das Übertragen von psychoanalytischen Inhalten in die technische Domäne nicht trivial. Aus diesem Problem ergibt sich die Anforderung, dass ein Über-Ich-File menschlich lesbar und intuitiv formulierbar sein soll, damit auch Psychoanalytiker die Regeln selbst eintragen können ohne sich an einem Informatiker wenden zu müssen. Dies führt dazu, dass die Über-Ich-Regeln außerhalb vom Programm festgeschrieben sein

[14] Die gelernten sozialen Regeln passieren bewusst, die internalisierte Regeln hingegen sind so eintrainiert, dass sie bereits im Unbewussten wirken.

müssen. Im Zuge dieser Diplomarbeit werden die hardgecodete Über-Ich Regeln durch eine flexible dateibasierte Speicherform ersetzt.

Konzeptionelle Anforderungen an die Über-Ich Regeln

Aus dem beschriebenen Sachverhalt und Vorgaben aus dem SiMA-Projekt ergeben sich folgende Anforderungen:

a. keine Notwendigkeit für zusätzliche Tools

b. menschlich lesbares Speicherformat

c. mit minimaler Einarbeitung editierbar

d. Syntax muss mit den psychoanalytischen Anforderungen kompatibel sein, also müssen
   - die Regeln mit einer Stärke versehen sein;
   - die Regeln möglichst ähnlich der natürlichen Sprache formuliert und gespeichert sein (um auch offline Review durch Psychoanalytiker zu ermöglichen) und
   - es muss die Möglichkeit gegeben sein, die Inneren-(Emotionen und Triebe) und Äußeren-(sensorische Eindrücke) Wahrnehmungen in einer Regel gemeinsam behandeln zu können.

Abgrenzungskriterien:

Um den Umfang der Arbeit einzugrenzen, werden folgende Rahmenbedingungen festgelegt

e. In dieser ersten Version keine Anforderungen an die Performance

f. Regeln brauchen nur beim Systemstart eingelesen zu werden

g. Der Agent lernt selbstständig keine neuen Regeln

h. Keine Sekundärprozess Operationen, d.h. keine zeitliche oder hierarchische Abfolge (wenn A gesehen wird und danach B), keine Negation (wenn kein rosa Elefant). Siehe Unterkapitel 3.1.2

Beschreibungssprachen

Als erster Schritt zur Umsetzung wurde eine Liste von potentiellen Beschreibungssprachen erstellt. Die mit der Aufgabenstellung am Besten übereinstimmenden Ansätze waren:

- **Regeln als eigenständige Java-Klassen** abspeichern und beim Programmstart festlegen, welche Klassen für die konkrete Simulation als Regeln verwendet werden sollen. Diese Option erlaubt flexible Regeln, die alle Aspekte von SiMA verwenden können. Als Nachteile konnten identifiziert werden, dass die Vorgehensweise gute JAVA-Kenntnisse erfordert hätte, nicht offline lesbar wäre und für Laien nicht nachvollziehbar da es keine Ähnlichkeit mehr zur natürlichen Sprache aufweist.
- **Lisp** – Dieser Programmcode findet in der klassischen KI-Programmierung immer wieder Verwendung. Als Programmiersprache könnten damit alle für SiMA relevanten Regeln aus-

formuliert werden [McCa60, pp.8-22, Grah05]. Allerdings müsste dafür ein Interpreter eingebaut werden, der dem Lisp-Programmcode Zugriff auf die java-Datenstrukturen erlaubt [McCa60, p.22]. Darüber hinaus sind Programmierkenntnisse für die Regeleingabe notwendig, das Regelformat hat keinen Bezug mehr zur natürlichen Sprache und ist auch nicht offline lesbar.

- **SPARQL Query** ist eine grafenbasierte Abfragesprache [8]. Die gesamte Information im SiMA existiert als Graph zur Ausführungszeit, somit würde sich das Beschreiben des zu filternde Graphen durch SPARQL anbieten. Hier würde sich die Möglichkeit zur Nutzung existierender Anfrage-Engines ergeben, was eine gute Performance bringen würde. SPARQL ist gut dokumentiert und erlaubt das Beschreiben aller in SiMA gängigen Graphen-Konstellationen. Gerade aufgrund der Vielseitigkeit und Flexibilität ist die Syntax allerdings für Nicht-Informatiker sehr schwer nachvollzieh- und erlernbar. Ferner haben SPARQL-Abfragen kaum Bezug zur natürlichen Sprachform.
- **XML-File** hätte schließlich zwar den Vorteil einer guten Tool Unterstützung. Das XML-Format ist auch bekannt, gut dokumentiert und XML [10] erlaubt eine gute Strukturierung von Regeln, außerdem ist ein Editor bereits in der für SiMA benutzen Entwicklungsumgebung Eclipse enthalten. Aber XML erlaubt es zum einen nicht, die Regel ähnlich einer natürlichen Sprachform zu betrachten, zum anderen wäre das Format für Nicht-Informatiker mit einem hohen Einarbeitungsaufwand verbunden.

Da alle anderen Optionen vorwiegend an der im konkreten Fall zwingenden Anforderung, die Regel ähnlich einer natürlichen Sprache zu formulieren, gescheitert sind, wurde es notwendig eine eigene Syntax spezifisch für das Formulieren von Regeln in einer „wenn -> dann“-Form für die Anwendung zu entwickeln: Die Regeln, die vom Anwender als Text-Dateien abgespeichert werden sollen, sollen später im Programm in eine eigenen Regel-Klasse geladen werden, die in weiterer Folge während einer konkreten Simulation den Datenfluss entsprechend filtert (siehe Kapitel 5.1).

Struktur einer Über-Ich Regel

Über-Ich-Regeln werden von den Psychoanalytikern definiert. Sie stellen eine Beschreibung von Inhalten, die nicht gleichzeitig auftreten dürfen, dar. Ein Entwurf der Textdatei könnte so aussehen, dass der erste Teil der Regel die Regelstärke festlegt (siehe Abbildung 3.7 „SuperEgo“). Der zweite Teil wäre die Beschreibung von den Inhalten (siehe in der Abbildung 3.7 „Bedingung“), die nicht gemeinsam auftreten dürfen (die Bedingungen sind so als wären sie mit einem logischen UND verknüpft), und der letzte Teil soll beschreiben, welche Inhalte beim Zutreffen des zweiten Teiles (Bedingungen) als konflikthaft markiert werden sollten (siehe in der Abbildung 3.7 „Konsequenzen“). Die letzte Zeile in der Abbildung 3.7 zeigt die formale Beschreibung des Aufbaues einer Regel-Zeile.

```
Ziffern = "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"
Wert = "1" | 0,{Ziffern}
SuperEgoStrength = "SuperEgoStrength"
eDriveComponent = "eDriveComponent"
eDriveComponent_Wert = "AGGRESSIVE", "LIBIDIDINOUS", "UNDEFINED"
eOrgan = "eOrgan"
eOrgan_Wert = "STOMACH", "RECTUM", "BLADDER", "STAMINA", "LIBIDO", "UNDEFINED"
QoA = QoA
eEmotionType = "eEmotionType"
eEmotionType_Wert = "ANGER", "ANXIETY", "LOVE", "JOY", "SATURATION", "GUILT", "HATE", "CONFLICT"
eContentType = "eContentType"
eContentType_Wert = "ENTITY"
Objekt = "BODO", "CAKE"

Triebblock = eDriveComponent eDriveComponent_Wert; eOrgan eOrgan_Wert[; QoA Wert [Wert]]
Wahrnehmungsblock = eContentType eContentType_Wert Objekt
Emotionsblock = eEmotionType eEmotionType_Wert[; QoA Wert [Wert]]

Trieb =  eDriveComponent eDriveComponent_Wert; eOrgan eOrgan_Wert
Wahrnehmung = eContentType eContentType_Wert Objekt
Emotion = eEmotionType eEmotionType_Wert

Bedingung = Triebblock, Wahrnehmungsblock, Emotionsblock
Konsequenz = Trieb, Wahrnehmung, Emotion

Regel = [SuperEgoStrength Wert,] Bedingung {; Bedingung}# Konsequenz {# Konsequenz}
```

Abbildung 3.7: Die Struktur der Regeldatei in EBNF (Erweiterte Backus-Naur-Form)

Die in Abbildung 3.7 angeführten „Bedingungen“ und „Konsequenzen“ sind Beschreibungen psychischer Inhalte. In der aktuellen Version der Über-Ich-Regeln ist die Formulierung von Bedingungen und Konsequenzen auf die Bereiche „Wahrnehmung“, „Emotionen“ und „Triebe“ beschränkt. In jedem Bereich erfolgt die Formulierung von „Bedingung“ und „Konsequenz“ durch ein geordnetes Paar, wobei der erste Term des Paars den Typ und der zweite den Inhalt beschreibt. In der finalen Version sollen die „Bedingungen“ und „Konsequenzen“ unter zur Hilfenahme von semantischem Wissen mit den momentan aktiven psychischen Inhalt abgeglichen werden. In SiMA ist noch kein automatisches, sematisches Schlussfolgern implementiert, weshalb die „Typ“ und „Inhalt“, zur Zeit, nur auf low level Attributen definiert werden können. Das heißt, statt dem Konzept „Hunger“, muss die Kombination „eOrgan STOMACH“ angeben werden. In der finalen Version würde die Zuordnung von „Hunger“ zu „eOrgan STOMACH“ durch die Ontologie automatisch erfolgen. In der aktuellen Version muss der „Typ“ einen bestimmten String entsprechen und der Inhalt dynamisch mit den aktuell verfügbaren Inhalte abgeglichen wird. Die folgende Tabelle ergänzt die in der Abbildung 3.7 angeführte EBNF und zeigt die aktuell nutzbaren Inhalte.

Typ	Mögliche Werte
eDriveComponent	AGGRESSIVE, LIBIDINOUS
eOrgan	STOMACH, RECTUM, BLADDER, STAMINA, LIBIDO, UNDEFINED

QoA	Ist ein Bewertungsbereich zwischen zweier Affektbeträgen, angegeben als ein oder zwei Double-Werte. Ein einzelner Double-Wert spezifiziert den Bereich von minus Unendlich bis zu dem angegebenen Wert. Bei Angabe von zwei Werten reicht der Bereich von ersten (inclusive) bis zum zweiten (inclusive) Wert.
eEmotionType	ANGER, JOY, GUILT, SHAME, HATE, LOVE,… (bereits über 20 Werte)
eContentType	ENTITY, EMTPYSPACE, BITE, ISALIVE, COLOR DISTANCE,… (bereits über 20 Werte)

Tabelle 4: Filterparameter für die Über-Ich Regeln

Anhaltend an die beschriebene Regelstruktur könnten zwei Über-Ich Regel Beispiele folgendermaßen aussehen:

- Wenn ein aggressiver Magentrieb mit einer Triebstärke von höchstens 0,23678 vorkommt UND die Emotion ANXIETY („Angst“) vorherrscht, soll der Agent die Torte nicht essen:
  `eDriveComponent AGGRESSIVE; eOrgan STOMACH; QoA 0,23678; eEmotionType ANXIETY; QoA 0,94 #eContentType ENTITY CAKE`
- Wenn das vorgegebene Super-Ego größer als 0,8 ist, der Agent Hunger mit einem Wert im Bereich 0,6 – 0,9 UND wenn ein anderer Agent (Bodo) in Sichtweite ist, soll der Agent die Torte nicht essen:
  `SuperEgoStrengh 0,8;eContentType ACTION EAT;QoA 0.6 0.9; eContentType ENTITY CAKE; eContentType ENTITY BODO#ACTION EAT`

An Schnittstellen zum existierenden Modell würden durch die neu implementierten, programmierten Über-Ich-Regeln folgende Module angesprochen werden, für einen Überblick siehe Abbildung 3.5: Die Über-Ich-Regeln beeinflussen das Modul F7 „Über-Ich reaktiv“. Die anhand der Über-Ich-Regeln erkannten Konflikte beeinflussen ferner die Module F6 „Abwehrmechanismen für Triebwünsche“ und F19 „Abwehrmechanismen für Wahrnehmung“.

3.3.2 Inventar

Das Inventar erlaubt es allen mobilen Entitäten in der MASON Simulationsumgebung, siehe dazu Kapitel 2.2.4 und Kapitel 4.1, Objekte aufzuheben, abzulegen oder mit sich zu führen, entweder in den Händen oder in einem Container mit begrenzter Kapazität. Das Inventar ist besonders für Szenarien von Interesse, bei denen Objekte geteilt, gegessen oder etwa aufbewahrt werden müssen, z.B. eine nicht fertig verzehrte Nahrungsquelle. Ohne Inventar waren die Interaktionen des Agenten mit Objekten vereinfacht, indem er beispielsweise bei der Aktion „Essen“ die Nahrungsquelle ohne Berührung verzehrt hat. In der ersten Version ist das Inventar nur bezüglich des Gewichts und der Anzahl an Objekten, jedoch nicht bezüglich des Volumens beschränkt.

Anforderungen

Seitens der SiMA-Entwickler gab es folgende Anforderungen an das Inventar:

- Objekte in die Hand nehmen können
- Aktuell in der Hand getragene Objekte in das Inventar transferieren
- Inventarobjekt als aktuell getragenes Objekt transferieren
- Konfigurierbares Maximalgewicht
- Statusinformation zurückgeben
    - Maximalles Gewicht
    - Aktuelles Gewicht aller Objekte
    - Die Referenzen auf alle Objekte im Inventar
- Konfigurierbarer Einfluss des getragenen Gewichtes auf die Ausdauer

Zur Erfüllung der obigen Anforderungen wurden folgende neue Konzepte angedacht:

- Neue Befehle für den Körper für die Interaktion mit Objekten:
    - Für das Anlegen neuer Befehle wird es notwendig sein, das Mapping der Neurosymbole zum Simulator „Befehlen“ zu erweitern und die neuen Aktionen in der Erinnerung des Agenten verfügbar zu machen. Neue Befehle könnten sein:
    - „Nimm Objekt in die Hand“;
    - „Transferiere das Objekt aus der Hand in das Inventar“;
    - „Transferiere das Objekt aus dem Inventar in die Hand“ und
    - „Lege das Objekt aus der Hand ab“.
- Neuer statischer Einflussfaktor auf die maximale Ausdauer:
    - Die neuen statischen Einflussfaktoren auf die maximale Ausdauer bedürfen nur einer kleinen Erweiterung der aktuell verwendeten Ausdauerberechnung.
- Die neuen Konfigurationsparameter müssen dem bereits existierenden Konfigurationskonzept hinzugefügt werden. Neue Konfigurationsparameter für:
    - die Inventargröße;
    - die maximale Traglast; und
    - den Faktor zur Berechnung der Auswirkung des Gewichtes auf die Ausdauer.

- Neue Visualisierungsschnittstellen müssen an das existierende Visualisierungskonzept angeknüpft werden.

Schnittstellen zum existierenden Modell

Aus den oben erwähnten Anforderungen ergeben sich folgende Anknüpfungspunkte an das existierende SiMA Modell:

- Die Komponente „Körper“ im Entity (siehe Abbildung 3.8) ist für die Berechnung des Ausdauerverbrauches verantwortlich und muss somit um die Berechnung eines statischen, auf dem Inventargewicht basierenden Faktors erweitert werden. Sie braucht dafür eine passive Schnittstelle zum Inventar für das Auslesen des Gewichtes.
- Die Komponente „Körper“ ist außerdem für die Weiterleitung von Aktionen an die Spielwelt verantwortlich z.B. beim Transferieren von Objekten aus dem Inventar zum „getragenen Objekt“. Sie bräuchte dafür eine Schnittstelle zum
    - Lesen und Setzen des aktuell getragenen Objektes;
    - Ablegen des aktuell getragenen Objektes;
    - Transfer des getragenen Objektes an eine bestimmte Position;
    - Auslesen der aktuellen Objekte im Inventar; und
    - Ablegen des aktuell getragenen Objektes im Inventar.
- Um auf Parameter aus den Konfigurationsdateien zugreifen zu können, müsste der von SiMA angebotene Parametriesierungsmechanismus – angesiedelt in der Konfigurationskomponente – nur um neue Parameter erweitert werden. Hier bräuchte man keine zusätzlichen Schnittstellen.
- Die Visualisierung benötigt eine passive Schnittstelle zum Auslesen des Gewichtes und der aktuellen Objekte im Inventar.

Abbildung 3.8 fasst die für das Inventar relevanten Komponenten des SiMA-Systems und ihre Interaktionen zusammen.

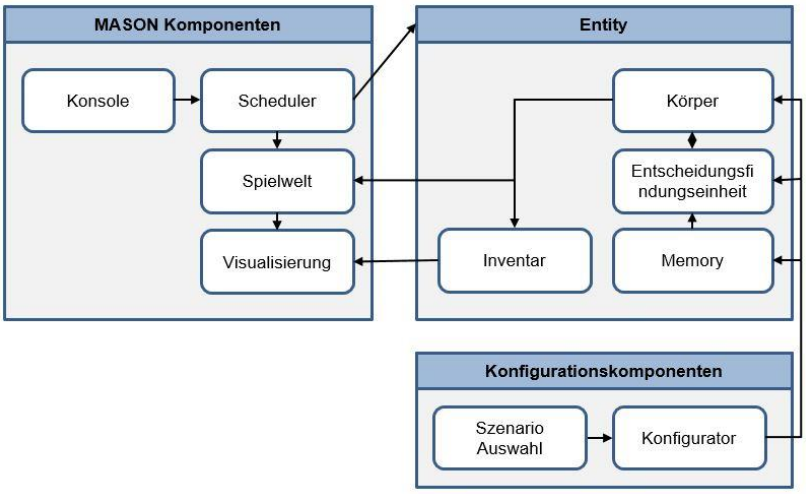


Abbildung 3.8: Übersicht der für das Inventar relevanten Komponenten

3.3.3 Visualisierungs-Konzepte

Das SiMA-Projekt ist mittlerweile zu einer Komplexität herangewachsen, bei der einfache Konsolen-Ausgaben, wie sie bei anderen Java-Applikationen, die zum Debuggen verwendet werden, nicht mehr ausreichen. Aus diesem Grund bietet die SiMA-Implementierung verschiedene grafische Visualisierungsmöglichkeiten. Im Zuge dieser Arbeit wurden zwei Bereiche um neue Visualisierungen erweitert. Das Inventar hatte eingangs gar keine Visualisierung, also sollte eine solche in diesem Bereich ganz neu eingeführt werden. Im Bereich der Abwehr existieren bereits grundlegende Visualisierungen, die Aufschluss darüber gaben, welche Abwehrmechanismen zum jeweiligen Zeitpunkt aktiv waren. Da bisher aber unbekannt war, warum es in der Struktur der Über-Ich-Regeln zum Konflikt kam und wie die Abwehrmechanismen die konflikthaften Inhalte verändert haben, sollen im Bereich der Abwehr zwei neue Inspektoren eingeführt werden. Einer für die Visualisierung der Über-Ich Regeln und einer für die Visualisierung der Konfliktbehebung.

Dieses Kapitel unterteilt sich in drei Unterkapiteln, nämlich „Über-Ich Regeln“, „Abwehr“ und „Inventar“, und beschreibt die angedachte Visualisierungsumsetzung dieser drei Bereiche.

Über-Ich Regeln

Der Inspektor für die Über-Ich Regeln soll als separater Tab bei den Visualisierungen des Moduls F07 umgesetzt werden und soll die aktuellen Regeln als Graph zeigen. Ziel hierbei ist es, die Struktur der Bedingungen und der daraus folgenden Konsequenzen übersichtlich darzustellen. Zu diesem Zweck sollen alle Regeln als zusammenhängende Baumstruktur abgebildet werden. Wie in Abbildung 3.7 beschrieben, gibt es bei den Über-Ich Regeln gemeinsam auftretende Bedingungen und gemeinsam auftretende Konsequenzen, die jeweils als Knoten repräsentiert werden sollen. Zur Abbildung der Regelstruktur sollen eigene Strukturknoten eingesetzt werden. Alle validen Regeln sollen in einer gemeinsamen Visualisierung abgebildet werden können. Daraus ergibt sich eine Struktur wie in Abbildung 3.9 exemplarisch aufgezeigt wird.

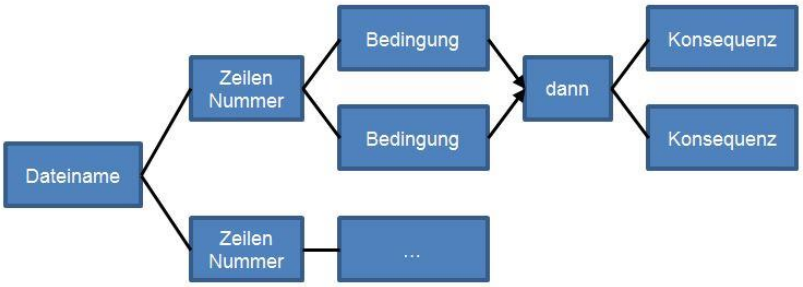


Abbildung 3.9: Exemplarischer Über-Ich-Regeln Graph von zwei Regeln

Wie bei Abbildung 3.7 beschrieben sind aktuell drei Arten von Bedingungen angedacht, entsprechend dazu gibt es drei Arten von Bedingungsknoten, die unterschieden werden müssen, siehe Abbildung 3.10. Der Bedingungstyp soll in der Knotenfarbe und Knotenbezeichnung abgebildet werden. Die Triebbedingung soll in rosa dargestellt werden und in der Beschreibung die Triebkomponente, die Triebquelle und den minimalen und maximalen Affektbetrag (QoA – *Quota of Affect*) angeben. Die

Emotionsbedingung soll in braun dargestellt werden und die Emotionsbezeichnung mit den minimalen und maximalen QoA in der Beschriftung angeben. Die Wahrnehmungsbedingung soll in gelb dargestellt werden und den Typ (Content-Type) und die Bezeichnung des konflikthaften Objektes in der Beschriftung enthalten.

Abbildung 3.10: Die drei Arten von Bedingungsknoten

Bezüglich der Darstellung der Konsequenzen sind auch drei Arten angedacht. Die Trieb-Konsequenz soll ebenfalls, wie die Triebbedingung, rosa dargestellt werden und soll die Triebkomponente und Triebquelle des zu verbietenden Triebes in der Beschriftung enthalten.

Die Emotions-Konsequenz soll wie die Emotions-Bedingung ebenfalls in braun dargestellt werden und den Namen der zu verbietenden Emotion in der Beschriftung enthalten.

Die Wahrnehmungs-Konsequenz soll wie die Wahrnehmungs-Bedingung ebenfalls gelb sein und den Objekt-Typ und die Bezeichnung der zu verbietenden Wahrnehmung in der Beschriftung enthalten.

Zu der Visualisierung der Über-Ich Regeln muss eine Schnittstelle zu F07 „Über-Ich reaktiv“ eingeführt werden, über die eine Repräsentation der aktuellen Über-Ich-Regeln zur Visualisierung übertragen wird.

Abwehr

Die existierenden Visualisierungen für die Abwehr erlauben es bis dato nicht, die Veränderungen, die durch die Abwehr zustande gekommen sind, nachzuvollziehen. Ein Verständnis dieses Effekts, zusätzlich zu der bereits existierenden Visualisierung, welche Abwehr aktuell aktiv ist, ist aber für die Entwicklung und Evaluierung am SiMA-Projekt wichtig. Ziel soll es dabei sein, mit einfachen Mitteln, übersichtlich zu repräsentieren, wie und wo sich die Abwehrmechanismen auf Triebe ausgewirkt haben. Zur Erinnerung: ein Trieb wird definiert durch Triebquelle, Triebkomponenten, Triebziel und Triebobjekt sowie den Partialtrieb im Falle eines Sexualtriebes. Die Triebquelle und der Partialtrieb ändern sich in der aktuellen Implementierung der Abwehr nicht, weshalb sie in der Visualisierung nicht berücksichtigt werden müssen. Bei Änderungen der Triebkomponente handelt es sich immer um eine Verschiebung von libidinöser zu aggressiver Triebkomponente und auch umgekehrt. Am komplexesten, und aktuell auch am häufigsten auftretend, ist die Änderung in Triebziel und Triebobjekt. Hier soll nicht nur nachvollzielbar sein in welchem Trieb die Änderung passiert, sondern auch welches das ursprünglich dominante Triebziel bzw. die ursprünglichen dominanten Triebobjekte waren und welches das neue dominante Triebziel bzw. die neuen dominanten Triebobjekte sind. Da das Verhalten des Agenten aktuell durch die jeweils dominanten Triebziel/Triebobjekt-Kombination beeinflusst wird, ist es ausreichend, nur die Änderungen in der Dominanz darzustellen. Des Weiteren soll leicht erkennbar sein, welche Abwehrmechanismen bei den verschiedenen Trieben wie oft aktiv waren;, aktuell existiert lediglich eine Übersicht zu den Abwehraktivitäten, welche jedoch nicht nach Trieben aufgeschlüsselt ist.

Um die angeführten Informationen zu transportieren, werden zwei neue Visualisierungen angedacht: Eine piktographische Zeitreihe (siehe dazu in der Abbildung 3.11 gelber Bereich), um darzustellen, wie sich die Triebe während der Abwehr verändert haben, siehe Abbildung 3.12, und mehrere Tortendiagramme (siehe dazu in der Abbildung 3.11 brauner Bereich), die das Verhältnis der angeschlagenen Abwehrmechanismen in der Vergangenheit der jeweiligen Triebe visualisieren sollen.

Die Piktographische Zeitreihe wird, der Übersicht halber, Icons zur Identifizierung der jeweiligen Triebe nutzen (siehe dazu in der Abbildung 3.11 blauer Bereich) und dann für jeden Trieb, wieder unterteilt nach Triebkomponente (aggressiv und libidinös), eine Icon-Reihe anlegen (für die Zusammensetzung einzelner Reihen-Icon siehe Abbildung 3.12), wobei sowohl Triebziel als auch Triebobjekt einfach durch die bereits im Simulator eingesetzten Bilder repräsentiert werden. Ursprüngliche und neue Triebziel/Triebobjekt-Kombinationen sollen gemeinsam in einem verkleinerten Icon auf der Zeitreihe präsentiert werden. Da alle aktuell existierenden Abwehrmechanismen nur vollständige Änderungen vornehmen (also z.B. das Triebziel „Essen“ vollständig durch das Triebziel „Teilen“ ersetzen), ist eine solche Visualisierung aktuell noch ausreichend. Eine Änderung der Triebkomponenten wird durch ein spezielles „Austausch“-Symbol zwischen den Zeitreihen dargestellt.

Die Tortendiagramme werden dagegen benutzt, um in einfacher und gut etablierter Form die Häufigkeitsverteilung der einzelnen Abwehrmechanismen im jeweiligen Trieb zu visualisieren. Dabei soll bei allen Diagrammen die gleiche Farbgebung verwendet werden, um die Vergleichbarkeit zu erhöhen. Auch das Ausbleiben von Konflikten im jeweiligen Trieb soll im Tortendiagramm aufscheinen, um eine Übersicht der generellen Abwehraktivität im jeweiligen Trieb zu ermöglichen.

Abbildung 3.11: Visualisierung für die Abwehr

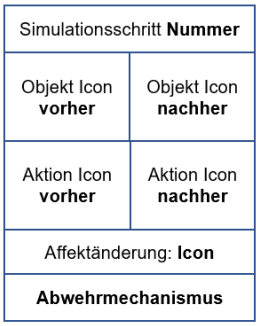


Abbildung 3.12: Einzelner Zeitreihen-Icon-Eintrag im Detail

Inventar

Da das Inventar erst im Zuge dieser Arbeit verfügbar gemacht wird, existiert noch keine Inventar – Visualisierung, auf der aufgebaut werden könnte. Im Zuge der Verwendung des Inventars ist es von Interesse, zu wissen, welche Objekte sich darin befinden, welches Objekt gerade getragen wird, wieviel vom maximal tragbaren Gewicht verwendet wird, wie sich das Gewicht zusammensetzt und wie sich die Traglast auf die Ausdauer auswirkt. Dazu werden zwei neue Inspektoren angedacht.

Durch ein **Tortendiagramm** soll die Zusammensetzung des Gesamtgewichts des Inventars visualisiert werden. Die Visualisierung soll den Anteil jedes Objektes an Maximalgewicht als eingefärbter Tortenabschnitt darstellen. Damit wird sowohl die Gewichtsverteilung der Objekte, als auch die noch offene Kapazität angezeigt.

Durch ein **Liniendiagramm** soll die zeitliche Entwicklung des getragenen Gewichtes, der maximalen Tragekapazität des Agenten und der aktuellen Ausdauer (Stamina) ausgegeben werden. Dabei soll ersichtlich werden, wie sich das Gewicht entwickelt, wie sich das Gewicht auf die Ausdauer auswirkt und wie das aktuelle Gewicht im Verhältnis zur Traglast steht.

Zusätzlich sollen auch noch über eine **Text-Konsole** Informationen über die ID des aktuell getragenen Objektes, die maximale Traglast des Agenten, das aktuelle Gesamtgewicht und die ID's aller aktuell im Inventar befindlicher Objekte ausgegeben werden.

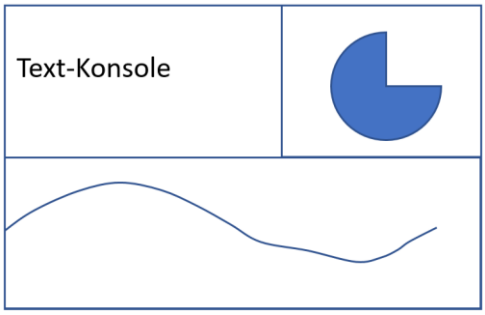


Abbildung 3.13: Inventar-Visualisierung. Text-Konsole, Torten- und ein Liniendiagramm

Die Visualisierung benötigt Daten aus verschiedene Teile des Systems. Um die Flexibilität des Systems und dessen zukünftige Weiterentwicklung zu gewährleisten, werden aber keine Schnittstellen für die spezifischen Systemteile eingeführt, sondern sollen die Daten von der Visualisierungsklasse über die Agenten-Referenz extrahiert werden.

4. Umsetzung und Use-Cases

In diesem Kapitel werden die Umsetzungen der im Kapitel 3.3 vorgestellten neue Konzepte präsentiert und die Use-Cases, die zur Validierung dieser Konzepte benutzt werden, spezifiziert. Damit wird gezeigt wie die neuen Ansätze entsprechend der Anforderungen der vorliegenden Diplomarbeit umgesetzt wurden und diese Umsetzung validiert werden kann. Hierfür ist es notwendig zuerst in Unterkapitel 4.1 das Konzept der Inspektoren in MASON und die grundlegende Navigation innerhalb der Simulationsumgebung zu erklären, da die Ausgabe der Inspektoren, die Grundlage für die hier beschriebenen Use-Cases, sind. Nach diesem Einführungskapitel werden die Umsetzung und die dazugehörigen Use-Cases in den Unterkapiteln „4.2 Über-Ich Regeln“, „Abwehr-Inspektor“ und „Inventar“ beschrieben.

4.1 Einführung in die Simulationsumgebung

Zum Testen des SiMA-Modells wird die Simulationsumgebung MASON verwendet. MASON ist – wie bereits an anderer Stelle ausgeführt - eine Multi-Agenten-Simulationsbibliothek für Java und bietet ein physisches Simulationsmodell mit Visualisierungsunterstützung in Form von verschiedenen Inspektoren. Über das Zusammenwirken und die Interaktion von MASON und SiMA siehe [Herr14, pp.66-67].

Das User-Interface von MASON Abbildung 4.1 setzt sich aus einer Visualisierung der simulierten Welt (gelb umrandet) und aus einem Steuerungsfenster (rot umrandet) zusammen. Die simulierte Welt ist durch eine braune Mauer eingegrenzt, in deren Innerem die in der Simulationswelt vorhandenen Elemente und Objekte dargestellt werden. Für eine Beschreibung der Bedeutung der Visualisierung siehe Kapitel 4.3.1.

Im Steuerungsfenster ist ein Inspektor-Tab geöffnet. Dieses Instrument ist für die Entwickler eminent, da damit die SiMA spezifischen Detailvisualisierungen erreichbar sind. Durch einen Mausklick in die Visualisierung der simulierten Welt werden in dem oberen Bereich des Steuerungsfensters alle Objekte aus der simulierten Welt in der Nähe der Klickposition angezeigt. Im unteren Bereich des Steuerungsfensters besteht die Möglichkeit, für jedes oben ausgewählte Objekt verschiedene Inspektoren aufzurufen.

Am unteren Ende des Steuerungsfensters befinden sich die Elemente, mit denen das „Starten“, „Pausieren“ und „Stoppen“ der Simulation möglich ist. Detaillierte Beschreibung des Steuerungsfensters sind unter [7] nachzulesen.

Im Zuge der vorliegenden Diplomarbeit werden ausgewählte Inspektoren in den Bereichen „Brain Details“ und „ARSin Overview“ betrachtet, siehe die Buttons im roten Bereich in Abbildung 4.1.

MASON unterstützt die Entwickler bei der Visualisierung mit einem klar definierten Ablauf, indem es eine klare Trennung von Simulation und Visualisierung ermöglicht, siehe dazu den „gelben“ und „roten“ Bereich in Abbildung 4.1. Dies wird dadurch erreicht, dass jeder Ausführungsschritt in zwei Phasen unterteilt wird, nämlich in eine Simulations- und in eine Visualisierungs-Phase. MASON stellt sicher, dass alle Elemente der Simulation, inkl. Physik, abgehandelt sind, bevor die Inspektoren aktualisiert werden. Außerdem bietet MASON eine Basis-Klasse für die Erzeugung von Visualisierungsfenstern. Diese Fenster nennen sich in MASON „Inspektoren“. SiMA erweitert dieses Konzept und stellt sicher, dass die einzelnen Inspektoren nacheinander in einer klar definierten Reihenfolge aufgerufen werden.

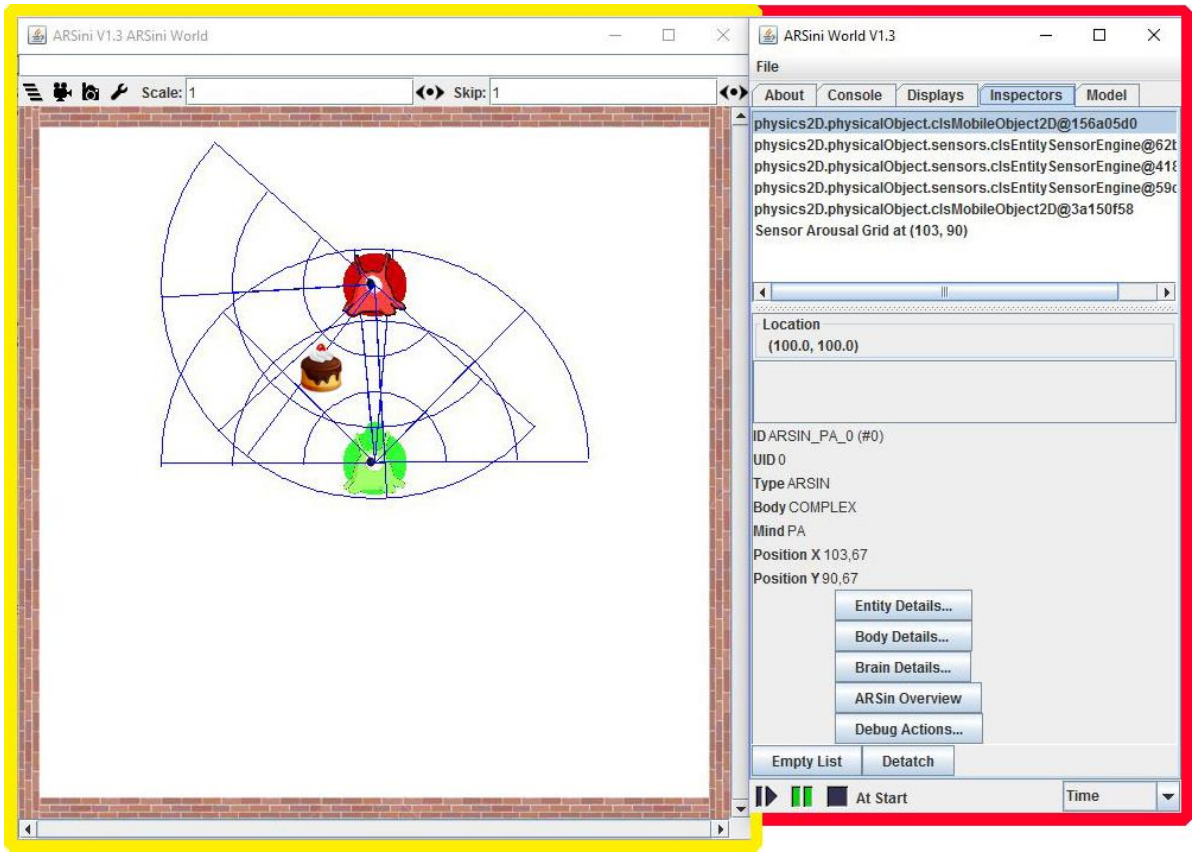


Abbildung 4.1: Links sieht man die Visualisierung der simulierten Welt (in der Farbe „gelb“) mit zweier Agenten, begrenzt durch eine Mauer. Zwischen den Agenten ist eine Torte platziert. Die Halbkreise (in blauen Linien), die an die Agenten angrenzen, repräsentieren ihr Sichtfeld. Die Aufteilung der Abschnitte ist der menschlichen Wahrnehmung angepasst. Rechts (in der Farbe „rot“ umrandet) sieht man das Steuerungsfeld. Oben wird im aktuellen Tab eine Liste aller Objekte in der Spielwelt angezeigt. Darunter befinden sich Detailinformationen des aktuell gewählten Objektes und die Schaltfläche zum öffnen der Inspektoren. Ganz unten befinden sich die Schaltflächen zum „Starten“, „Pausieren“ und „Stoppen“ der Simulation.

4.2 Über-Ich Regeln

Der Inspektor für die Über-Ich-Regeln ist als separater Tab bei den Visualisierungen des Moduls F07 umgesetzt und visualisiert die aktuellen Regeln als Graph unter Zuhilfenahme der Bibliothek JGraph [2]. Ziel hierbei ist es, die Struktur der Bedingungen und der daraus folgenden Konsequenzen übersichtlich darzustellen. Zu diesem Zweck werden alle Regeln als zusammenhängende Baumstruktur abgebildet. Ein Strukturknoten, der die Zeilennummer in der die Regel im Über-Ich-File steht, angibt, bildet den Anfang für die regelspezifischen Untergraphen. Auf den Strukturgraphen folgt je ein Knoten für jede Bedingung in dieser Regel. Um die Bedingungen von den Konsequenzen zu separieren, wird ein weiterer Strukturknoten mit der Beschriftung „then“ eingefügt, der alle Konsequenzen – d.h. Psychische Inhalte welche als konflikthaft markiert werden, wenn die Regel zutrifft – als Blätter enthält.

Wie in dem Unterkapitel „Über-Ich Regeln“ im Kapitel 3.3.3 beschrieben wird, sind aktuell drei Arten von Bedingungen umgesetzt. Entsprechend dazu gibt es drei Arten von Bedingungsknoten, die sich durch den Text und die Farbe voneinander unterscheiden (für ein Beispiel, der alle Arten von Bedingungs- und Konsequenz-Knoten enthält, siehe Abbildung 4.3):

- Der Triebbedingungs-Knoten wird in rosa dargestellt und gibt die Triebkomponente, die Triebquelle und den minimalen und maximalen Affektbetrag (QoA) an.
- Der Emotions-Bedingungsknoten wird in braun dargestellt und gibt die Emotionsbezeichnung mit dem minimalen und maximalen Affektbetrag an.
- Der Wahrnehmungs-Bedingungsknoten wird in gelb dargestellt, beinhaltet den Content-type und die Bezeichnung des konflikthaften Objektes.

Bei den Konsequenzen sind ebenfalls folgende drei Kategorien umgesetzt:

- Die Trieb-Konsequenz ist ebenfalls rosa wie die Triebbedingung und beinhaltet die Triebkomponente und Triebquelle des zu verbietenden DM’s.
- Die Emotions-Konsequenz ist ebenfalls braun wie die Emotions-Bedingung und beinhaltet den Namen der zu verbietenden Emotion.
- Die Wahrnehmungs-Konsequenz ist ebenfalls gelb wie die Wahrnehmungs-Bedingung und beinhaltet den Content-Type und die Bezeichnung der zu verbietenden Wahrnehmung.

Sowohl bei den Bedingungen als auch bei der Konsequenz-Spezifikation können folgende Kriterien zur Filterung angegeben werden:

- Triebquelle (das Körperteil dem der Trieb zugeordnet ist);
- Triebkomponente (aggressiv oder libidinös);
- Affektbetrag des Triebes (ist die Intensität des Triebes);
- Wahrnehmungs-Objekt (beliebige Objektbezeichnung wie Torte, Bodo, udgl.);
- Emotionstyp; und
- Affektbetrag der Emotion (ist die Intensität der Emotion).

Siehe dazu Tabelle 4 für Details an möglichen Werten von Bedingungen und Konsequenzen.

Im Rahmen zweier Use-Cases soll die Funktionalität der Programmierung aufgezeigt werden - nicht jedoch wie sinnhaft die psychoanalytischen Regeln sind. Dafür wird die Definition der Filter-Regel, ab der textuellen Beschreibung bis hin zur Visualisierung dargelegt. Der neu entwickelte Über-Ich-Inspektor befindet sich im Steuerungsfenster unter „Brain Details“ -> Modul F7 -> Tabulator „Rules“.

Zur Validierung des Ansatzes werden im Folgenden zwei Beispielen vorgestellt: Das erste Beispiel zeigt eine einfache Über-Ich-Regel, die nur Trieb und Wahrnehmungsinhalte filtert. Das zweite Beispiel erweitert das erste um eine Filterung nach Emotionsinhalten.

4.2.1 Use-Case: Regel 1

Beschreibung der Regel: Wenn der libidinösen Magentrieb höher ist als ein bestimmten Schwellenwert und der Agent sowohl Bodo als auch eine Nahrungsquelle wahrnimmt, dann wird der libidinöse Magentrieb als konflikthaft markiert.

Natürlichsprachliche Regel: Sei in Gegenwart anderer nicht hungrig!

Repräsentation in der Über-Ich Regel Syntax – im Regelfile (hier farblich dem Über-Ich-Inspektor angeglichen):

```
eDriveComponent LIBIDINOUS;eOrgan STOMACH;QoA 0.2 1;eContentType ENTITY CAKE;eContentType ENTITY BODO#eDriveComponent LIBIDINOUS#eOrgan STOMACH
```

Über-Ich-Inspektor Ausgabe:

Die interne Repräsentation der Über-Ich-Regel wird durch den neuentwickelten Regel-Inspektor „Rule“ im Modul F7 visualisiert, siehe Abbildung 4.2. Die Regel besteht aus einer Zeile, repräsentiert durch den linken grünen Knoten mit der Beschriftung „Rule 1:“. Die Regel kombiniert drei Bedingungen. Der rosa-Knoten oben mittig repräsentiert die Bedingung an den libidinösen Magentrieb mit einem Affektbetrag zwischen 0.2 und 1.0. Die beiden gelben Knoten repräsentieren die Anforderung, dass ein Objekt vom Typ Torte „CAKE“ und ein Objekt vom Typ „Bodo“ in der Wahrnehmung vorhanden sein müssen. Der rosarote-Konsequenzknoten ganz rechts visualisiert den zu markierenden libidinösen Magentrieb, wenn die Bedingungen davor zutreffen.

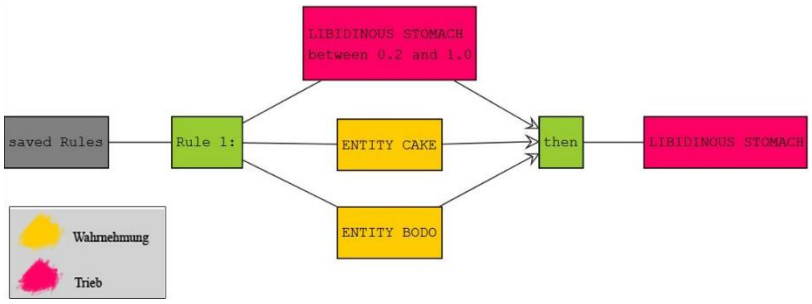


Abbildung 4.2: Darstellung der Grafen-Ausgabe von "Regel 1".

4.2.2 Use-Case: Regel 1 + 2

Regel 2 erweitert das bereits im Zusammenhang mit der Regel 1 dargestellte Szenario um ein Emotions-Matching und zusätzliche Konfliktmarker für Emotion und Wahrnehmung. Dies führt dazu,

dass von der Abwehr des Agenten zusäztliche Emotions- und Wahrnehmungskonflikte gelöst werden müssen, wenn Regel 2 anschlägt.

Beschreibung der Regelerweiterung: Wenn der Agent keinen großen Hunger hat und mit Bodo in Konkurrenz um eine Nahrungsquelle steht, wird er unbewusst – solange er auch nur die geringste Angst verspürt – die Nahrungsquelle und seinen Hunger unterdrücken, was seine Angst und Wut verringert.

Natürlichsprachliche Regel: Solange du nicht hungrig bist, konkurriere nicht mit deinem Bruder ums Essen!

Repräsentation in der Über-Ich Regel Syntax – im Regelfile (hier farblich dem Über-Ich-Inspektor angeglichen):

```
eDriveComponent AGGRESSIVE;eOrgan STOMACH;QoA 0.2;eEmotionType ANXIETY;eContentType ENTITY CAKE;eContentType ENTITY BODO#eContentType ENTITY CAKE#eDriveComponent AGGRESSIVE#eOrgan STOMACH#eEmotionType ANGER#eEmotionType ANXIETY
```

Über-Ich Inspektor Ausgabe:

Abbildung 4.3 zeigt, wie die interne Repräsentation inklusive der Erweiterung um die eben beschriebene zweite Regel aussieht. Die zweite Regel ist in der Grafik durch den Subgraph beginnend beim grünen Knoten „Rule 2 visualisiert. Die neu eingefügte Regel 2 kombiniert vier Bedingungen:

(1) Der rosarote Knoten oben mittig repräsentiert die Bedingung an den aggressiven Magentrieb mit einem Affektbetrag kleiner 0.2.
(2) Der braune Knoten visualisiert die Bedingung der Existenz der Emotion „anixiety“, unabhängig von der Höhe des Affektbetrags.
($3^{+}4$) Die beiden gelben Knoten repräsentieren die Anforderung, dass ein Objekt vom Typ Torte „CAKE“ und ein Objekt vom Typ „Bodo“ in der Wahrnehmung vorhanden sein müssen.

Die Konsequenzen der Regel sind dargestellt durch vier Knoten:
(1) Der rosarote Konsequenzknoten visualisiert den zu markierenden aggressiven Magentrieb,
(2+3) die beiden braunen Konsequenzknoten visualisieren die zu markierenden Emotionen „anger“ und „anxiety“ und
(4) der gelbe Konsequenzknoten visualisiert die zu markierende Wahrnehmung „CAKE“.

Zu beachten ist, dass die beiden Regeln bei der Filterung getrennt voneinander überprüft werden, wohingegen die Bedingungen innerhalb der Regeln alle gleichzeitig zutreffen müssen. Die Konsequenzen treten nur ein, wenn die entsprechende Bedingung erfüllt ist.

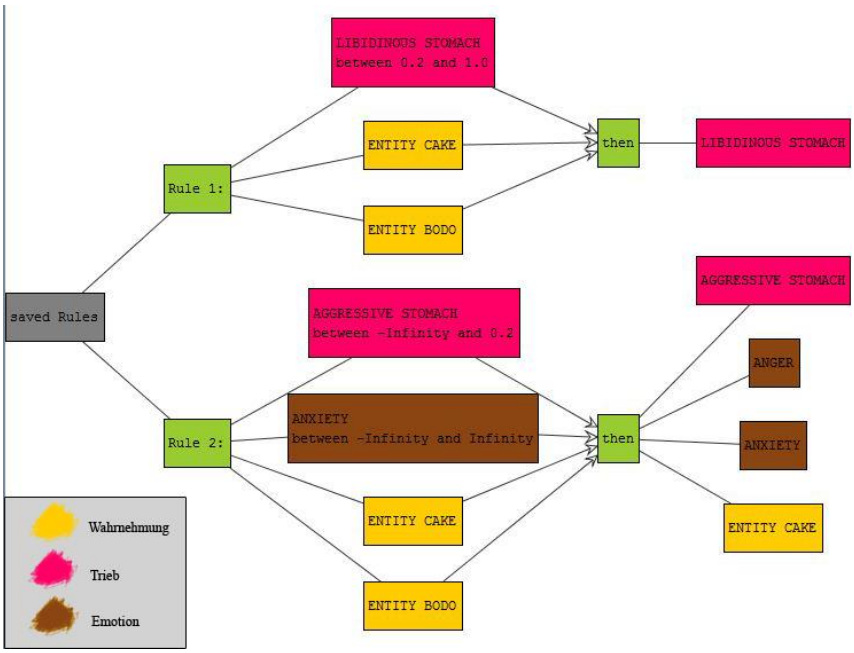


Abbildung 4.3: Darstellung der Grafen-Ausgabe von „Regel 1“ und „Regel 2“.

4.3 Abwehr-Inspektor

Eine wesentliche Neuerung von der Diplomarbeit ist die Einführung einer Abwehrvisualisierung, die es auch Nicht-Informatikern erlaubt, die Funktionsweise und die Ergebnisse der Abwehr nachzuvollziehen. Hierfür wurden piktogrammbasierte Zeitreihen benutzt, die die Arbeitsweise der Abwehrmechanismen der zuletzt gesetzten Schritte aufzeigen, wobei ein Simulationsschritt den Durchlauf eines im Kapitel „Ein Zyklus im Rahmen des SiMA-Modells“ beschriebenen Modells bedeutet.

Die Funktion des Piktogramm-Inspektors wird anhand von Variationen eines spezifischen Use-Cases demonstriert. Im ersten Unterkapitel wird der zu Grunde liegende Use-Case erklärt, danach wird auf die Grundlagen der Inspektor-Ausgaben eingegangen. Die restlichen Unterkapitel betrachten die verschiedenen Variationen des Use-Cases, um die unterschiedlichen Visualisierungsarten aufzuzeigen, die vom Visualisierungsinspektor beherrscht werden. Möglich sind Änderungen des Triebziels, Änderungen des Triebobjekts, Änderungen des Affektbetrags und eine Kombination daraus.

4.3.1 Gemeinsame Konfiguration für Use-Cases 1-3

Die Use-Cases basieren auf der in Abbildung 4.1 gezeigten simulierten Welt. Diese beinhaltet zwei Agenten, Adam (grün) und Bodo (rot), in der Mitte ist eine Torte platziert. Die Halbkreise (in blauen Linien), die an die Agenten angrenzen, repräsentieren ihr Sichtfeld. Dieses ist in Quadranten unterteilt. Dabei können Entfernungen „nah“, „mittel“, „fern“ und die relativen Positionen „links außen“, „links“, „mittig“, „rechts“ und „rechts außen“ bestimmt werden. Die Aufteilung der Abschnitte ist der menschlichen Wahrnehmung angepasst.

Der grundlegende Ablauf ist bei allen Use-Cases gleich: Adam sieht Bodo und die Torte, dadurch entstehen Konflikte. Der Agent Bodo ist passiv, der Agent Adam ist so konfiguriert, dass er Hunger und eine leicht aggressive Trieblage hat, außerdem hat er eine Über-Ich-Regel implementiert, die lautet: „Du musst dein Essen, wenn jemand dabei ist, mit diesem teilen“. Adam hat in diesen Szenarien einen hohen Hungertrieb und die Kombination „Torte essen“ ist in seine Erinnerungen als bestes Befriedigungsobjekt für Hunger hinterlegt.

Im Originalszenario wäre der Ablauf so, dass Adams Wunsch, zu essen durch die erwähnte Über-Ich-Regel zum Teilen sublimiert wäre, worauf hin er die Torte in zwei Hälften zerteilt hätte, die zweite Hälfte in der Nähe von Bodo abgelegt und die eigene Hälfte verzehrt hätte. In den nachfolgend dargestellten Use-Cases liegt der Fokus nicht auf dem Verhalten des Agenten, sondern dient dieser nur zur Veranschaulichung der Visualisierung, weshalb das tatsächliche Agenten-Verhalten in den einzelnen Use-Cases nicht weiter beschrieben wird.

4.3.2 Allgemeine Beschreibung des Piktogramm-Inspektors

Die Triebe des Agenten sind zeilenweise repräsentiert. Jede Zeile unterteilt sich in drei Bereiche. Links ist die Triebidentifizierung, in der Mitte werden die Änderungen der Abwehr der letzten vier Zyklen im Detail dynamisch visualisiert und rechts werden alle Änderungen in einem Kreisdiagramm zusammengefasst. Jeder Trieb teilt sich in einen aggressiven und einen libidinösen Anteil, wobei in der dynamischen Detail-Visualisierung oben der aggressive Anteil und unten der libidinöse Anteil dargestellt sind, während bei der Kreisdiagrammvisualisierung links der aggressive und rechts der libidinöse Anteil dargestellt werden. Für ein Beispiel dieser Visualisierung, siehe Abbildung 6.5.

Das SiMA-Modell weist vier Selbsterhaltungstriebe in aggressiver und/oder libidinöser Form auf. Zur besseren Veranschaulichung werden die Triebe durch Symbole repräsentiert. Der Ausdauertrieb wird durch eine Batterie, der Magen, die Blase und das Rektum werden durch entsprechende anatomische Bilder dargestellt.

In der Detailvisualisierung, Abbildung 4.4, wird immer der Zustand des DM vor der Abwehr gegenüber dem Zustand des DM nach der Abwehr dargestellt. Dabei werden Änderung im Triebziel und Triebobjekt durch Piktogramme des Vorher- und Nachherzustands aufgezeigt. Die Änderungen im Affektbetrag werden durch einen nach oben oder unten gerichteten Pfeil veranschaulicht. Zusätzlich wird bei jeder Darstellung der Name des angewendeten Abwehrmechanismus textuell ausgegeben. Jeder Eintrag in der Detailvisualisierung besteht somit aus drei Bereichen:

- der Zyklus-Nummer – ganz oben in der Mitte
- dem grafischen Teil in der Mitte. Dieser ist zeilenweise unterteilt in
    - Triebobjekt vor der Abwehr in blass, links, und Triebobjekt nach der Abwehr rechts;
    - Triebziel vor der Abwehr in blass, links, und Triebziel nach der Abwehr rechts;
    - Änderung des Affektbetrags durch einen grünen Pfeil nach oben, einen roten Pfeil nach unten oder durch ein blaues ‚=‘, wenn keine Änderung stattgefunden hat
- Name des Abwehrmechanismus ganz unten

Der Betrachtungszeitraum der Visualisierung bezieht sich auf dem Bereich vom ersten Aufruf des Inspektors bis zum aktuellen Schritt.

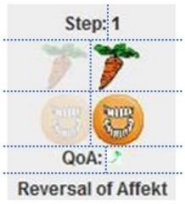


Abbildung 4.4: Detailvisualisierung

Eine Ausnahme dieser Darstellungsform ist das Fehlen des Bereichs der Abwehr. Hier ist erstmals zu unterscheiden zwischen dem Ausbleiben eines Konfliktes und dem Ausbleiben eines Abwehrmechanismus beim bestehenden Konflikt. Aus Triebsicht sind diese beiden Vorkommnisse äquivalent, sprich das Triebziel, das Triebobjekt und der Affektbetrag bleiben unverändert, aber für den User ist die Unterscheidung relevant und muss klar erkennbar sein. Um hier eine für den User erkennbare Unterscheidung treffen zu können, wird beim Ausbleiben des Konfliktes gar keine Detailvisualisierung ausgegeben. Wenn dagegen ein Konflikt auftritt, der nicht abgewehrt werden kann, wird eine Detailvisualisierung angelegt, bei der Triebziel, Triebobjekt und der Affektbetrag unverändert bleiben und als Abwehrname „No Defense“ ausgegeben wird.

Nach den zum Verständnis notwendigen Erläuterungen werden die spezifischen Use-Cases, in denen verschiedenen Aspekte der Visualisierung exemplarisch dargestellt werden sollen, betrachtet. Die Bilder sind Screenshots und stammen direkt aus dem tatsächlich bei der SiMA-Entwicklung verwendeten Inspektor. Die Entwicklung wurde anhand von vier Use-Cases evaluiert. Der erste zeigt eine Änderung des Triebziels, der zweite eine Änderung des Triebobjektes, der dritte eine Änderung des Affektbetrags und der vierte kombiniert verschiedene Abwehrmechanismen in verschiedenen Trieben teilweise parallel. Alle Use-Cases benutzen das gleiche, bereits dargestellte Szenario als Ausgangsbasis. Wie bereits mehrmals erwähnt, ist das SiMA-Modell sehr komplex und somit nur schwer für spezifische Abläufe konfigurierbar. Besonders die Auswahl von Abwehrmechanismen war zum Zeitpunkt der Implementierung dieser Arbeit noch in Entwicklung. Als Konsequenz wurde der Programmablauf in allen vier Use-Cases angepasst. In den Use-Cases 1-3 wurde der Algorithmus zur Auswahl eines Abwehrmechanismus durch einen direkt parametrisierten Algorithmus ersetzt, mit anderen Worten: welcher Abwehrmechanismus angewendet wird, wurde – um es demonstrieren zu können – für jeden Use-Cases spezifisch festgelegt.

4.3.3 Use-Case 1: Triebziel ändert sich

Der erste Use-Case veraunschaulicht die Visualisierung der Änderung des Triebziels. Hierfür wurde der Abwehrmechanismus „Sublimierung“ parametrisiert. Die parametrisierte Sublimierung entspricht in diesem Szenario der unbewussten Entscheidung, statt den eigenen Hunger zu empfinden, mit dem anderen Agenten teilen zu wollen.

In Abbildung 4.5 wird ein Screenshot der relevanten Zeile im Piktogramm-Inspektor angezeigt. Hierbei handelt es sich um den libidinösen Magentrieb. Links ist das Symbol der Triebquelle „Magen“ zu

sehen. Mittig werden in der unteren Zeile (da libidinös) die letzten vier Schritte im Detail visualisiert. Ganz rechts zeigt das dazugehörige Kreisdiagramm in der Spalte für libidinöse Triebkomponenten die Verteilung der Abwehrmechanismen im aktuell angezeigten Schritt.

In der Detailvisualisierung lässt sich, von oben nach unten und zeilenweise von links nach rechts gelesen, ersehen, dass das Triebobjekt „Torte“ unverändert blieb. Das Triebziel „Essen“, repräsentiert durch Messer und Gabel, wurde jedoch ersetzt durch das Triebziel „Zerteilen“, repräsentiert durch den angerissenen Kreis. Der Affektbetrag blieb unverändert und wird deshalb durch das blaue „=“repräsentiert. Als Abschluss wird der Name des angewendeten Abwehrmechanismus „Sublimierung“ ausgegeben.

Aus dem rechts befindlichen Kreisdiagramm lässt sich herauslesen, dass im Betrachtungszeitraum im konkreten Fall von Schritt 1 bis Schritt 4 nur eine Art von Abwehrmechanismus, nämlich „Sublimierung“ zur Anwendung kam.

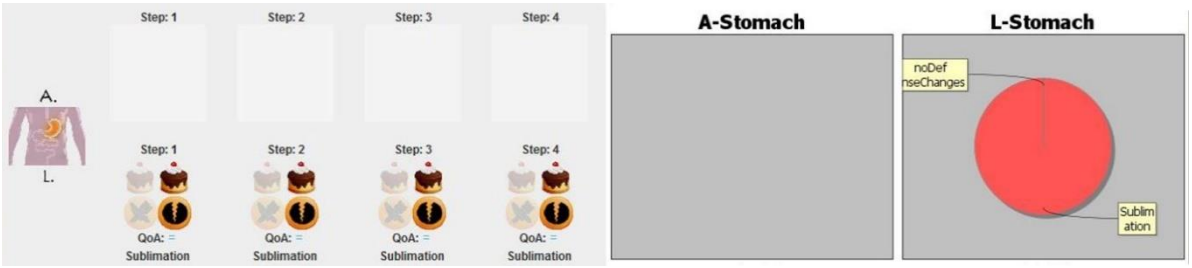


Abbildung 4.5: Darstellung von der Visualisierungsausgabe des Abwehrmechanismus "Sublimierung"

4.3.4 Use –Case 2: Triebobjekt ändert sich

Der zweite Use-Case veranschaulicht die Visualisierung der Änderung des Triebobjektes. Hierfür wurde der Abwehrmechanismus „*Displacement*“ parametrisiert. Dieser Abwehrmechanismus entspricht einer Neuausrichtung des Triebwunsches auf ein anderes verfügbares Objekt. Der Agent hat Hunger auf eine Torte, es ist keine Torte vorhanden, sondern ein Schnitzel, also ersetzt er die Objekte.

Abbildung 4.6 zeigt ein Screenshot der relevanten Zeile im Piktogramm-Inspektor. Hierbei handelt es sich ebenfalls wieder um den libidinösen Magentrieb: Links zeigt sich das Symbol der Triebquelle „Magen“, mittig werden in der unteren Zeile (da libidinös) die letzten vier Schritte im Detail visualisiert. Ganz rechts zeigt das dazugehörige Kreisdiagramm in der Spalte für die libidinösen Triebkomponenten die Verteilung der Abwehrmechanismen im aktuell angezeigten Schritt.

In der Detail-Visualisierung sieht man, von oben nach unten, zeilenweise von links nach rechts gelesen, dass das Triebobjekt „Torte“ geändert wird auf das Triebobjekt „Schnitzel“. Das Triebziel „Essen“, repräsentiert durch Messer und Gabel, bleibt ebenso unverändert wie der Affektbetrag, repräsentiert durch das blaue „=“. Als Abschluss wird der Name des angewendeten Abwehrmechanismus „*Displacement*“ ausgegeben.

Aus dem Kreisdiagramm lässt sich herauslesen, dass im Betrachtungszeitraum im konkreten Fall von Schritt 1 bis Schritt 4 nur eine Art von Abwehrmechanismus, nämlich „*Displacement*“, zur Anwendung kam.

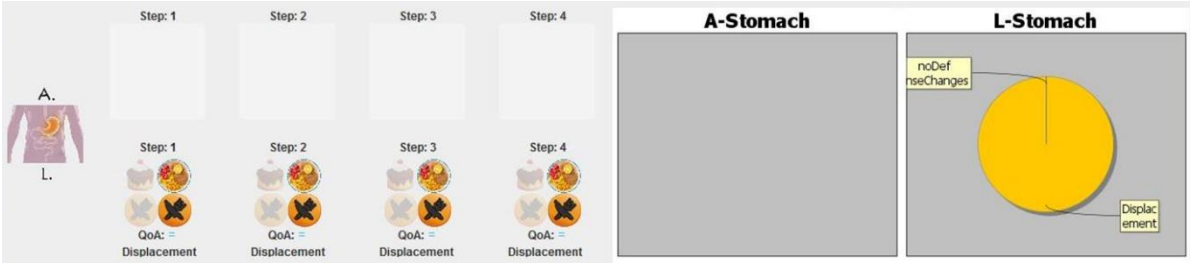


Abbildung 4.6: Darstellung von der Visualisierungsausgabe des Abwehrmechanismus "*Displacement*"

4.3.5 Use-Case 3: Der Affektbetrag ändert sich

Der dritte Use-Case veranschaulicht die Ausgabe der Visualisierung, wenn sich der Affektbetrag ändert. Diese Veränderung wird ausgelöst durch den Abwehrmechanismus „Affektverkehrung“. Dies funktioniert laut Psychoanalyse niemals nur auf einer Triebkomponente, sondern kommt es im Rahmen dieses Abwehrmechanismus zu einer Verschiebung der Affektbeträge von einem libidinösen zu einer aggressiven Komponente oder umgekehrt. Im konkreten Use-Case erfolgt die Verschiebung von libidinös zum aggressiv. Der Abwehrmechanismus entspricht einer Neuausrichtung des Triebwunsches auf ein anderes verfügbares Objekt. In diesem Fall würde der Agent auf Grund seines Hungers essen wollen, aber nach der Abwehr hat er stattdessen das Bedürfnis auf etwas herum zu kauen, ohne das Objekt wirklich zu verzehren.

Abbildung 4.7 zeigt ein Screenshot der relevanten Zeile im Piktogramm-Inspektor. Hierbei handelt es sich wieder um den aggressiven und libidinösen Magentrieb: Links lässt sich das Symbol der Triebquelle „Magen“ ersehen, mittig werden in beiden Zeilen (aggressiv und libidinös) der letzten vier Schritte im Detail visualisiert. Ganz rechts zeigen die dazugehörigen Kreisdiagramme in den jeweiligen Spalten die Verteilung der Abwehrmechanismen innerhalb der aktuell angezeigten Schritte.

Aus der Detailvisualisierung geht hervor, von oben nach unten, zeilenweise von links nach rechts gelesen, dass das Triebobjekt in beiden Triebkomponenten unverändert blieb – beim Zustand „Aggressiv“ handelt es sich um eine Karotte, im libidinösen Zustand um eine Torte. Dasselbe gilt für das Triebziel, im aggressiven Bereich will der Agent „beißen“, während er im libidinösen Bereich „essen“ will. Im Unterschied zu den beiden vorangehenden Use-Cases hat sich der Affektbetrag im konkreten Fall bei beiden Komponenten verändert. Im Fall der aggressiven Komponente ist er gestiegen, im Fall der libidinösen Komponente ist er gesunken (jeweils im gleichen Ausmaß). Als Abschluss wird der Name der angewendete Abwehrmechanismus „*Reversal of Affekt*“ ausgegeben.

Aus den Kreisdiagrammen links und rechts geht hervor, dass im Betrachtungszeitraum im konkreten Fall von Schritt 1 bis Schritt 4 nur eine Art von Abwehrmechanismus, nämlich „*Reversal of Affekt*“, zur Anwendung kam.

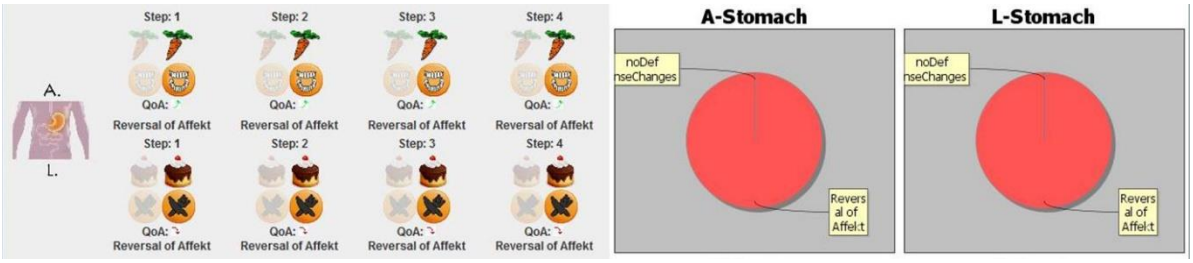


Abbildung 4.7: Darstellung von der Visualisierungsausgabe des Abwehrmechanismus "Affektverkehrung"

4.4 Inventar

Dieses Unterkapitel zeigt zuerst wie die Interaktion vom Agent und Welt umgesetzt wird, anhand einer Beispielhaften Interaktion. Danach werden die durch diese Arbeit verfügbar gemachten Funktionen des Inventares beispielhaft anhand der neu erstellten Aktion „Aufheben und ins Inventar stecken“ mithilfe des ebenfalls neu entwickelten Inventar-Inspektors. Der eben erwähnte Inspektor lässt sich durch einen Doppelklick auf den Agenten und die Auswahl „ARSin overview“ sowie den Tab „Inventory“ im Steuerfenster anwählen.

4.4.1 Interaktion

Eine Beispielhafte, stark abstrachierte, Umsetzung der Interaktion zwischen dem Agent und der Umwelt wird in Abbildung 4.8 gezeigt. Dabei sieht man, dass die Interaktion vom Agenten durch das Kommando „pick-up“ initialisiert wird und die Welt das geforderte Objekt zuerst aus dem Spielfeld entfernt und dann eine Referenz an den Agenten zurückliefert.

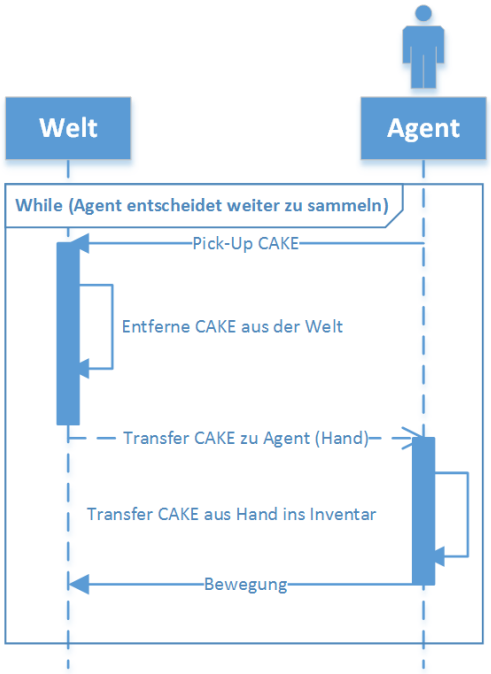


Abbildung 4.8: abstrahierte Interaktion zwischen Welt und Agent beim „sammeln“

4.4.2 Use-Case

Zur Evaluierung dieser Funktion wird ein Use-Case verwendet, indem der Agent die gesamte Spielwelt nach Nahrungsquellen absucht und gefundene Nahrungsquellen einsammelt, siehe Abbildung 4.9 für die Ausgangslage.

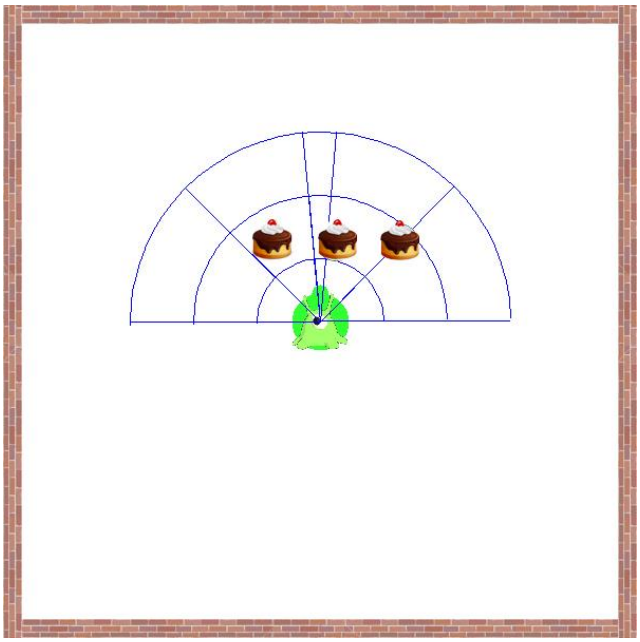


Abbildung 4.9: Ausgangslage in dem Simulations-Szenario

Adam (der grüne Agent) wird sich nach dem Start der Simulation die erste Torte nähern, es aufnehmen und sofort in das Inventar verschieben, siehe Abbildung 4.10.

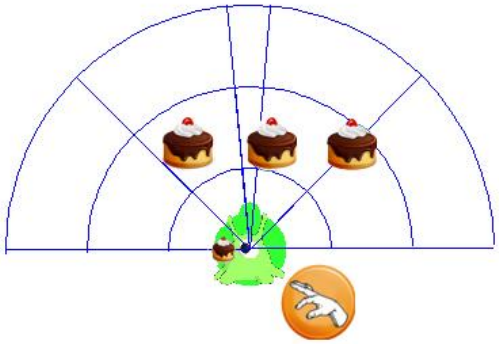


Abbildung 4.10: Symbol "Aufheben" und Mini-Torte in der Hand

Nachdem der Agent die Aktion „pick-up“ ausgeführt hat, hat er erstmals das Objekt als getragenes Objekt in der Hand, siehe Abbildung 4.11. Diese textuelle Ausgabe entspricht den Bereich „Text-Konsole“ aus der Abbildung 3.13 im Abschnitt „Inventar“ im Kapitel 3.3.3.

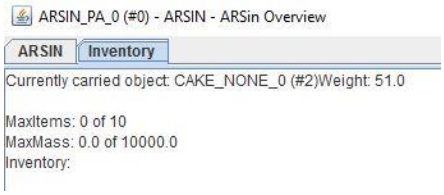


Abbildung 4.11: Textueller Ausgabe-Teil des Inventar-Inspektors beim ersten Aufheben

In dem hier benutzten Szenario packt der Agent die getragenen Objekte sofort in das Inventar. Nachdem er dies für alle drei Torten gemacht hat, zeigt Abbildung 4.12, dass der Agent alle drei Torten aufgehoben und verstaut hat. Die Torten haben nicht viel Auswirkung auf das Gewicht. Die Tragekapazität ist nicht nur durch das Gewicht beschränkt, sondern auch durch die Anzahl der tragbaren Objekte.

Im Liniendiagramm zeigt sich, wie sich das Inventargewicht auf die Ausdauer auswirkt. Abschnitt 1 zeigt das Gehen des Agentes ohne die Torten. Die Ausdauer fällt nur langsam ab. Abschnitt 2 zeigt das Gehen mit nur einer Torte. Die Ausdauer sinkt dabei bereits etwas schneller ab. Abschnitt 3 zeigt das Gehen mit zwei Torten. Die Ausdauer sinkt noch etwas schneller. Abschnitt 4 zeigt die Bewegung mit drei Torten im Inventar. Die Auswirkungen des Gewichtes ist hier nicht gut zu erkennen, da der Agent sich immer nur kurze Strecken bewegt, während er nach weiteren Torten sucht. In der Zeit, in der er nach weiteren Objekten Ausschau hält, ruht er, was den Anstieg der Ausdauer erklärt. Im Abschnit 5 findet die eigentliche Bewegung mit drei Torten statt, im Abschnitt 6 ruht der Agent wieder, um weiter Ausschau nach Nahrung zu halten.

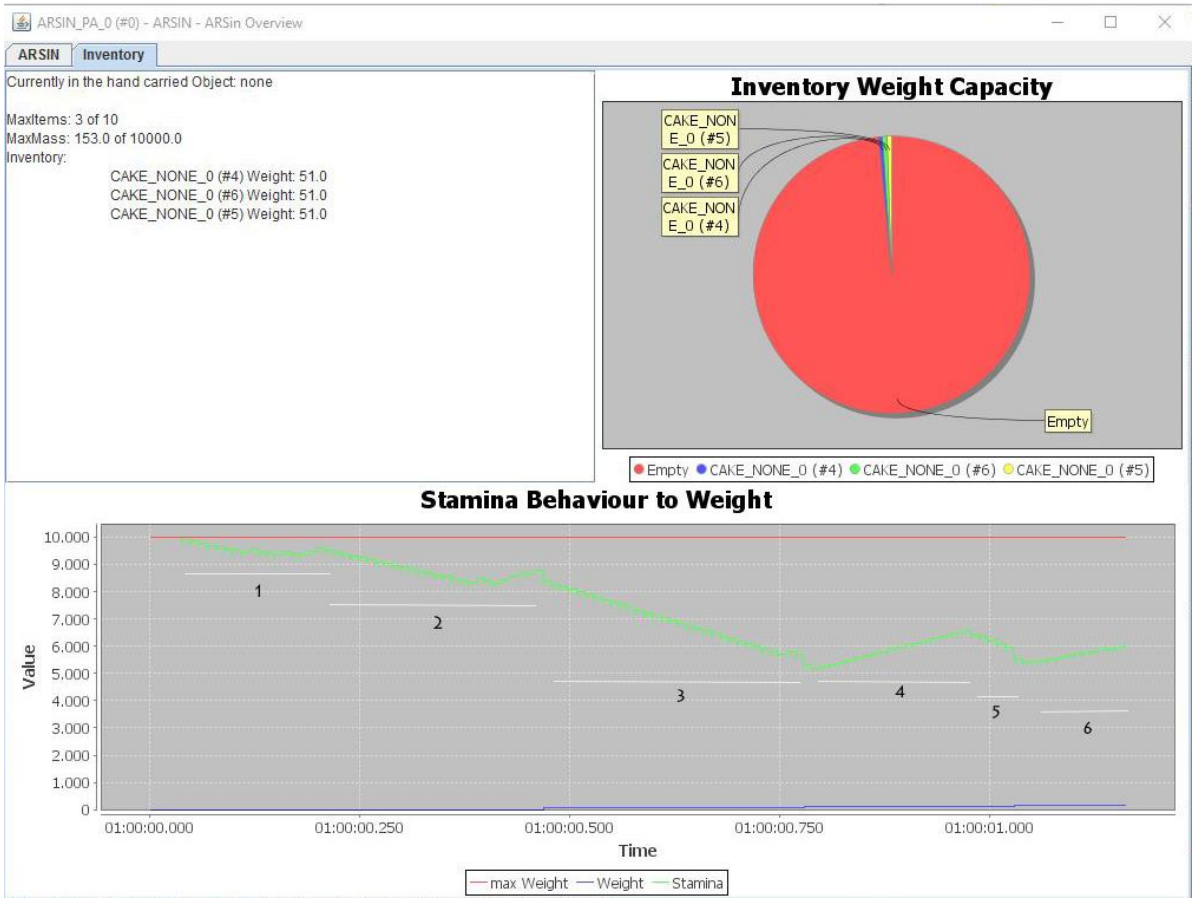


Abbildung 4.12: Darstellung des Inventar-Inspektors.
Anmerkung: die Torte ist System-Intern unter dem Namen „CAKE_NONE_0“ referenziert.

5. Implementierung

Basierend auf den in Kapitel 3.3 vorgestellten Modulen und Modellen der neuen Konzepte wird in diesem Kapitel auf deren technische Realisierung eingegangen. Das Kapitel wird sich entsprechend der Zielsetzungen dieser Diplomarbeit in drei Unterkapiteln aufteilen:

Zuerst wird der neu implementierte Mechanismus zur Formulierung von Über-Ich-Regeln im Unterkapitel „Über-Ich Regeln“ vorgestellt. Dabei wird auch die Anbindung der neuen Regel-Syntax, aus Kapitel 3.3.1, und der dazugehörige Visualisierung an die existierenden SiMA-Komponenten erklärt.

Im nächsten Unterkapitel „Abwehr“ wird auf die Implementierung der neuen Abwehrvisualisierung eingegangen. Dabei werden die Aspekte zur Datenaufbereitung seitens SiMA und die Visualisierung seitens der Inspektoren beleuchtet, mit besonderem Fokus auf die verwendete Layout-Struktur.

Im letzten Unterkapitel „Inventar“ wird die Implementierung des neuen Inventar-Systems vorgestellt. Dies beinhaltet die Neuerungen im Bereich der Aktionen in der Simulationsumgebung, Neuerungen an der Inventar-Klasse selbst und dem Inventar-Inspektor.

5.1 Über-Ich Regeln

Wie im Kapitel Modelle und Konzepte 3.2.1 beschrieben, befinden sich die Über-Ich-Regeln in der Abwehrschiene in den Modulen „F7, F6 und F19“ und sind für das Filtern von psychischen Inhalten anhand von internalisierten Über-Ich-Regeln zuständig. Im Zuge dieser Diplomarbeit wird die existierende Lösung, bei der die Regeln hard-gecodet in den entsprechenden Modulen implementiert sind, durch eine flexible file-basierte Lösung mit eigener Regel-Syntax ersetzt. Da die neuen Entwicklungen im bereits existierenden Code integriert werden müssen, unterliegt die Implementierung verschiedenen Einschränkungen, die sich aus dem bereits existierenden Programmcode ergeben. Um all diesen Punkten gerecht zu werden, wird im vorliegenden Unterkapitel die Interaktion der beteiligten Module erläutert und anschließend die Funktionalität aufgeteilt in „Einlesen“, „Parsen“ und „Markieren“ detailliert beschrieben.

5.1.1 Ablauf

Das SiMA-Modell besteht aus verbundenen Modulen, die – anders als beim bionischen Vorbild – sequenziell ausgeführt werden. Somit ergibt sich der aus der Grafik „Abwehr Detail“ in der Abbildung 5.1 abgebildete Ablauf, der den konzeptionellen Programmablauf der Abwehr zeigt. Das Modul F56

initialisiert die Abwehr durch das Bereitstellen von „neutralisierter Intensität“. Die neutralisierte Intensität ist ein Indikator, wieviel Rechenleistung für das Lösen von Konflikten aufgewendet werden darf. Soll beispielsweise wenig Rechenleistung für die Abwehr eingesetzt werden, können Konflikte durch extrem einfache Abwehrmechanismen wie z.B. „Verleugnung“ gelöst werden. Hierbei werden die konflikthaften Elemente solange gelöscht, bis kein Konflikt mehr besteht. Wenn mehr Rechenleistung eingesetzt werden darf, kann etwa der elaborierte Abwehrmechanismus „Sublimierung“ ausgeübt werden. Dieser versucht zuerst gesellschaftlich anerkannte Alternativen zum konflikthaften Inhalt zu finden und integriert diese Alternativen in weiterer Folge in die existierenden Inhalte.

Danach erfolgt die Konflikterkennung in proaktiver und reaktiver Form. Beide Module kommunizieren die konflikthaften Inhalte an die Konfliktlösungsmodule, die abhängig von den Typen des Inhaltes (Triebe oder Wahrnehmungen und Emotionen) versuchen, den Konflikt aufzulösen.

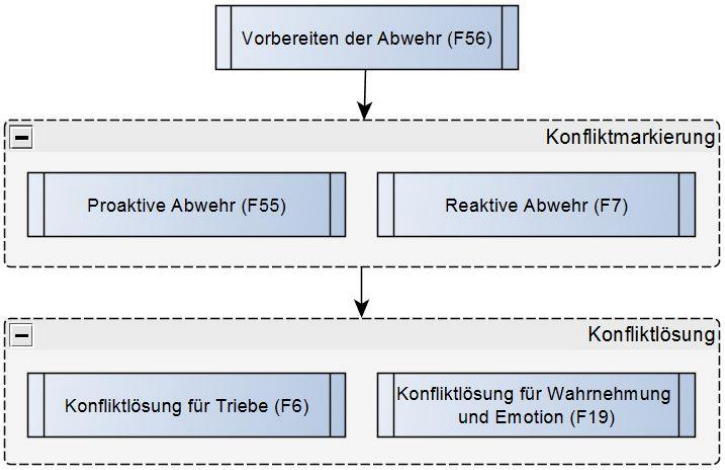


Abbildung 5.1: Schritte der Abwehr

Die neu entwickelten Mechanismen zum Definieren von Über-Ich-Regeln sind in das Modul F7 integriert. Sie unterteilen sich in die Schritte „Einlesen“, „Parsen der Über-Ich-Regeldatei“ und „Erzeugen der Regelinstanzen“, sowie „Durchsuchen und Markieren der aktuellen Information anhand der Regelinstanzen nach Konflikten“. Abbildung 5.2 zeigt ein Aktivitätsdiagramm incl. des Objektflusses für die angeführten Schritte. Das Modul F7 liest dabei die reaktiven Über-Ich-Regeln aus einem text-File (welches der Struktur in Abbildung 3.7 folgt) heraus. Welches File verwendet wird, wird im Zuge der Persönlichkeitsparameter für jeden Agenten vor dem Start der jeweiligen Simulation einzeln konfiguriert. Aus dem Inhalt des Files werden Zeilenweise, mittels einen selbstgeschriebenen Top-Down-Parsers (LL(1)), Instanzen vom Typ `clsSuperEgoRulesCheck` erzeugt. Die erzeugten Instanzen durchsuchen die eingehenden Trieb-, Wahrnehmungs- und Emotionsrepräsentationen anhand der Therme ihrer zugrundeliegenden Regeln. Für jeden Regelverstoß wird ein Konfliktobjekt erzeugt, das die Charakteristika der konflikthaften Inhalte ermittelt, die Konfliktstärke berechnet und Beides abspeichert. Die Konfliktobjekte werden anhand ihrer Quelle in Listen aufgeteilt. Die Liste mit den verbotenen Trieben wird zu F6, die Liste mit der verbotenen Wahrnehmungen und die Liste mit den verbotenen Emotionen an F19 geschickt. Diese Listen werden getrennt von den Inhalten (Triebe,

Wahrnehmung und Emotionen) transportiert, d.h. die Inhalte werden in diesem Modul noch nicht verändert.

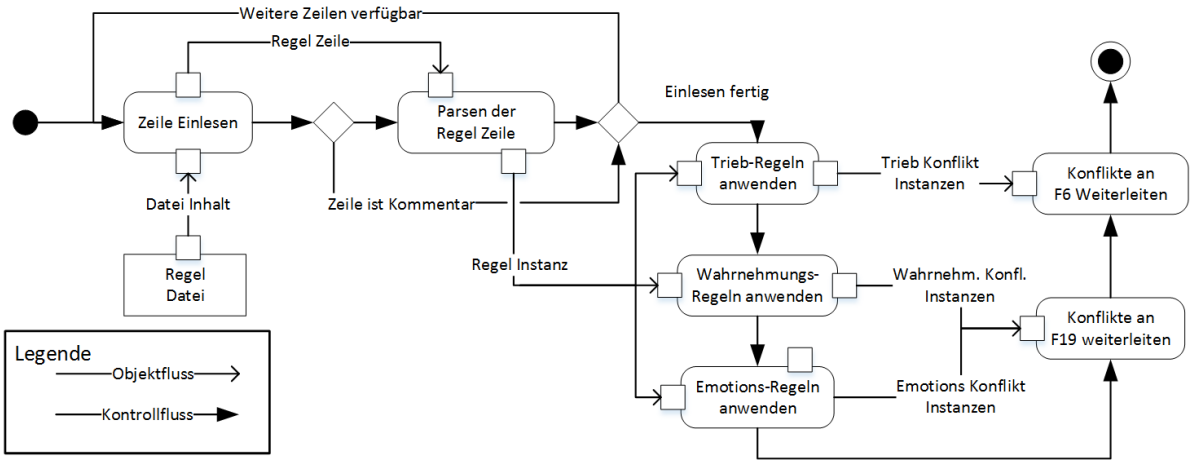
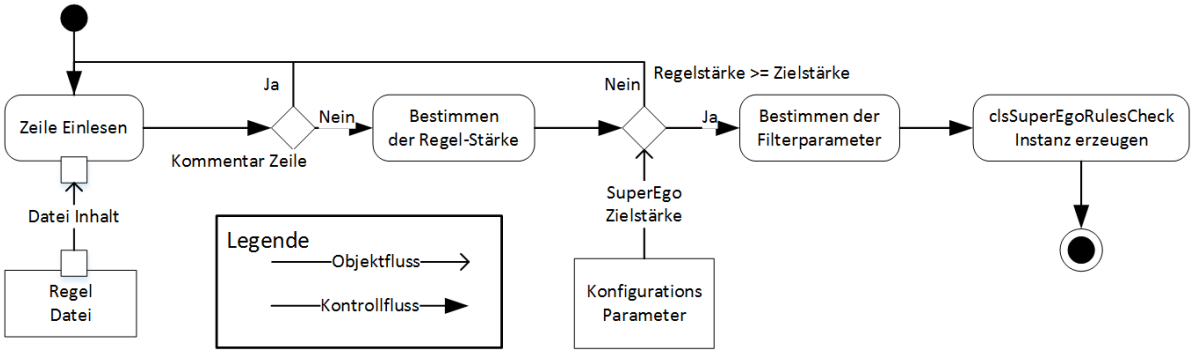


Abbildung 5.2: Aktivitätsdiagramm für die Konflikterkennung im Modul F7

Das Einlesen, Parsen und Instanzen erzeugen wurde in der Klasse `clsSuperEgoRulesCheck` gekapselt. Aktuell wird diese nur im Modul F7 verwendet, in späterer Folge sollen ähnliche Abläufe auch für die proaktiven Regeln im Modul F55 implementiert werden. Der übliche Ablauf in der Verwendung von `clsSuperEgoRulesCheck` sieht folgendermaßen aus:

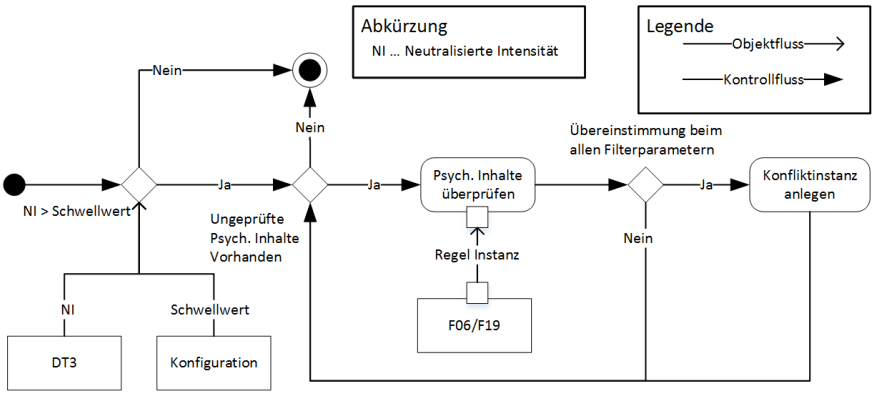
1. **Erzeugen einer Liste von `clsSuperEgoRulesCheck`-Instanzen entsprechend der in der Über-Ich-Datei definierten Regeln** mittels der statischen Methode `List<clsSuperEgoRulesCheck> fromFile(double prSuperEgoStrength, String poFileName)`: Diese Methode vergleicht anfangs die aktuelle Über-Ich-Stärke mit der in der Regel definierten Stärke. Sie verwirft Regeleinträge, sofern deren Zielstärke nicht erreicht wird. Im anderen Fall – also bei Erreichen der Zielstärke – wird die Regel (entspricht der ganzen Zeile) in einer `clsSuperEgoRulesCheck`-Instanz gekapselt, die alle Bedingungen und Konsequenzen dieser einen Regel beinhaltet.



2. **Registrieren der aktuellen Triebe, Wahrnehmungen und Emotionen** über die statische Methode `setCheckingSuperEgoRuleParameters(clsThingPresentationMesh poPerceptionalMesh, ArrayList<clsEmotion> poEmotions_Input, ArrayList<clsDriveMesh> poDrivesInput)`. Die Registrierung hat ihre eigene Funktion,

zum einen, um die Flexibilität des Ablaufes zu erhöhen, und zum anderen, weil es für das Debugging nützlich ist, den Zustand der Emotionen, Wahrnehmungen und Triebe auch außerhalb der Methode `checkInternalizedRules` verfügbar zu machen. Letzteres beispielsweise um auch nach der Abwehr feststellen zu können, welche Triebe den Konflikt ausgelöst haben.

3. **Starten des Filterns** durch Aufruf der Methode `checkInternalizedRules()`. In diesem Schritt wird vor dem eigentlichen Check die neutralisierte Intensität überprüft, und nur, wenn diese höher als ein konfigurierter Schwellwert ist, überhaupt die Prüfung auf Konflikte begonnen. Mit anderen Worten: Wenn der Wert zu gering ist, werden überhaupt keine Konflikte erkannt. Dieser Vorgang steht im Gegensatz zur Abwehr, bei der ein Mangel an neutralisierter Intensität dazu führt, dass die erkannten Konflikte nicht gelöst werden können. Die Grundlage für die Prüfung sind die `clsSuperEgoRulesCheck`-Instanzen, die vorher erzeugt wurden. Diese Instanzen beinhalten immer nur die identifizierenden Charakteristika der verbotenen Kombination aus Inhalten (z.B. bei Trieben die Triebkomponente als `eDriveComponent` und die Triebquelle als `eOrgan`, für nähere Informationen zu den Datenstrukturen siehe [DBD+14, pp.86-97]). Diese Charakteristika werden in den aktuellen Wahrnehmungen, Emotionen und Trieben gesucht und wenn sie gemeinsam auftreten, werden in der `clsSuperEgoRulesCheck`-Klasse entsprechende Konfliktinstanzen angelegt.



4. **Auslesen der vorher angelegten Konfliktinstanzen** als Klassen des Typs `clsSuperEgoConflictPerception`, `clsSuperEgoConflictEmotion` oder `clsSuperEgoConflictDrive` über die `clsSuperEgoRulesCheck`-Methoden `getForbiddenPerception()`, `getForbiddenEmotions()` und `getForbiddenDrives()`.

Der Grund für das Trennen der Prozesse des File-Lesens und der Übergabe der aktuellen Trieb- und Emotions-Lage vom eigentlichen Check ist, dass die Über-Ich-Regeln an zwei verschiedene Stellen in SiMA überprüft werden müssen: einmal vor und einmal nach der Abwehr. Vor der Abwehr muss die Überprüfung durchgeführt werden, um festzustellen, welche Konflikte es gibt. Nach der Abwehr ist eine Überprüfung dagegen notwendig, um feststellen zu können, ob alle Konflikte gelöst worden sind, da durch das Lösen von den Konflikten wieder neue andere Konflikte entstanden sein könnten. Die Konfliktstärke nach der Abwehr wird benutzt, um erweiterte Emotion zu erzeugen [Scha16, pp.106-107].

Da das Markieren und Lösen von Konflikten in der Abwehr in zwei verschiedenen Modulen abgehandelt wird, werden die konflikthaften Inhalte, repräsentiert durch die oben genannten Klassen, über die entsprechenden Interfaces an den Rest der Abwehr verteilt.

5.1.2 Inspektor

Im Zuge der vorliegenden Diplomarbeit wurde eine neue Visualisierungs-Klasse (`clsGraphForRules`) und ein dazugehöriges Interface (`itfInspectorForRules`) entwickelt. Außerdem wurden zwei bestehende Klassen erweitert (`clsInspectorTab_Modules, clsGraph`).

- `clsGraphForRules` erbt von `clsGraphWindow`, einer abstrakten Fensterklasse aufbauend auf dem MASON's Inspektorsystem, und ist für die Erzeugung einer neuen Graphen-Instanz verantwortlich. Außerdem transferiert sie die Regeln vom aufrufenden Modul zum Graphen, siehe `itfInspectorForRules`.
- `itfInspectorForRules` definiert das Interface vom Datenaustausch zwischen dem Inspektor `clsGraphForRules` und den zugeordneten Modulen. Das Interface erzwingt im implementierenden Modul eine Methode `getDriverules ()`, über die Regeln ausgelesen werden können
- `clsInspectorTab_`Modules wurde erweitert, um für alle Module, die das Interface `itfInspectorForRules` implementieren, eine Regelinspektor-Instanz (`clsGraphForRules`) zu erzeugen (dies entspricht der typischen SiMA Vorgehensweise).
- `clsGraph` ist die Klasse für die Visualisierung von Graphen. Hier findet das eigentliche Zeichnen statt. Die Klasse enthält für alle darstellbaren Datentypen eine Methode, die den jeweiligen Datentyp visualisiert. `clsGraph` ruft diese Methoden in der jeweiligen Reihenfolge, abhängig von der Struktur des Eingangs-Graphen, auf. Im Zuge der vorliegenden Diplomarbeit wurde eine zusätzliche Methode zur Visualisierung des Datentyps `clsSuperEgoRulesCheck` erstellt.

5.1.3 Schnittstellen/Kommunikation

Aus den oben beschriebenen Anforderungen und Abläufen ergeben sich die nachfolgend dargestellten Änderungen in den Schnittstellen des SiMA-Modells, siehe Abbildung 5.3. Wir unterscheiden im Folgenden zwischen SiMA-Schnittstellen und Visualisierungs-Schnittstellen. Die SiMA-Schnittstellen beziehen sich auf den im SiMA etablierten Mechanismus zur Kommunikation zwischen Modulen. Der Mechanismus basiert auf Paaren von java-Interfaces, die von den Kommunikationspartnern implementiert werden. Dies wurde eingeführt, um die ausgetauschten Daten in automatisch erstellten Inspektoren visualisieren zu können. Für Details siehe [DBD+14].
Die Visualisierungs-Schnittstellen beziehen sich ebenfalls auf einen standardisierten Mechanismus zum Datenaustausch zwischen den SiMA-Modulen und MASON-basierten Inspektoren (in der aktuellen Implementierung sind alle Inspektoren MASON-basiert, d.h. abgeleitet von einer MASON-Klasse). MASON stellt dabei die Grundklassen für Inspektoren bereit – also das Fenster, in das Darstellungen mit eigenen Visualisierungsmethoden eingefügt werden können. SiMA hat dieses Konzept

um einen Automatismus für das Anlegen von Inspektoren erweitert. Dieser wurde eingeführt, um automatisiert entscheiden zu können, welche Inspektoren für welches Modul erzeugt werden müssen. Dafür wird ein java-Interface registriert und dieses Interface in allen Modulen, die diesen Inspektor verwenden sollen, implementiert. Die grundlegende SiMA-Infrastruktur sorgt dann dafür, dass alle Inspektoren in der entsprechenden Reihenfolge während der Inspektor-Phase im MASON-Simulationszyklus aufgerufen werden. Die Inspektoren sind in weiterer Folge dafür verantwortlich, die Daten über das registrierte Interface abzuholen und darzustellen.

SiMA-Schnittstellen:

Nach dem Einlesen der Regeln und dem Erkennen von Konflikten im Modul F7 werden die Markierungen für die konflikthaften Inhalte (als `clsSuperEgoConflictDrive`, `clsSuperEgoConflictEmotion`, `clsSuperEgoConflictPerception`) an die für die Konfliktlösung zuständigen Module F6 und F19 gesendet. Dafür werden die Interfaces I5.11 (geht zum Modul F19) und I5.13 (geht zum Modul F6) erweitert. I5.11 wird um eine Liste aus `clsSuperEgoConflictPerception`-Instanzen und eine Liste aus `clsSuperEgoConflictEmotion`-Instanzen erweitert. I5.13 wird um eine Liste aus `clsSuperEgoConflictDrive`-Instanzen und `clsSuperEgoConflictEmotion`-Instanzen erweitert.

Visualisierungs-Schnittstellen:

Für die Regelinspektoren wurde das Interface `itfInspectorForRules` als Schnittstelle zwischen Visualisierung durch die Klasse `clsGraphForRules` und dem Modul F7 erstellt. Das Interface beschreibt eine einzelne Methode namens `ArrayList <clsSuperEgoRulesCheck> getDriverules ()`, die eine Liste aller aktuell aktiven Über-Ich-Regeln als `clsSuperEgoRulesCheck`-Instanzen zurückgibt und in der Klasse F07 implementiert werden musste.

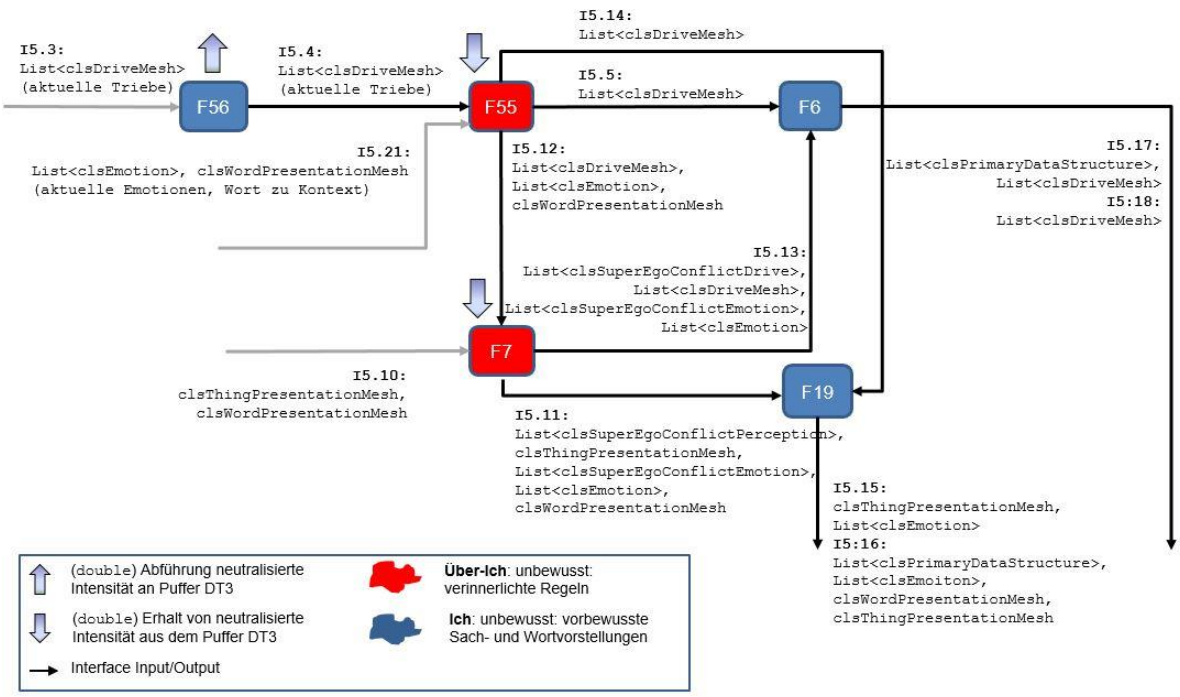


Abbildung 5.3: Diese Abbildung veranschaulicht die Module und Interfaces der Abwehr mit den Typen der Datenübergabe.

5.2 Abwehr

Die Abwehrvisualisierung visualisiert die letzten Schritte der Abwehr als Piktogrammen. Dafür wurde eine neue Visualisierungstechnik eingeführt, die in SiMA noch nicht verwendet worden ist. Darüber hinaus wurde eine neue Schnittelle zwischen dieser neu erstellten Klasse `clsInspectorImageDrives` und dem SiMA-Modell in Form einer neuen Interface-Klasse eingeführt.

5.2.1 Daten Aufbereitung

Der Inspektor ist so konstruiert, dass er in allen Modulen verwenden kann, in denen sich die Trieb-Inhalte ändern. Dafür wurde ein neues Interface entwickelt, siehe Kapitel 5.2.2, das aktuell von Modul F06 implementiert wird. Um das Interface übersichtlich zu gestalten, war es notwendig, dass die zu visualisierenden Daten aufbereitet werden, bevor sie über ein Interface zur Visualisierungsklasse geschickt werden.

Zur Verallgemeinerung des Interfaces von SiMA zum Inspektor wurde eine Datenstruktur (`clsChangedDrives`) definiert, die zwei Zustände desselben Triebes, nämlich das Vorher und das Nachher, abbildet. Dafür werden paarweise Triebziel, Triebobjekt und Affektbetrag zusammen mit der Identifizierung des zugrundeliegenden Triebes und des für die Änderung verantwortlichen Abwehrmechanismus abgespeichert. Zur Vereinfachung wurde in F06 die Hilfsfunktion `aufbereitungInterface(clsDriveMesh, String)` erstellt. Diese wird in den Abwehrmethoden der jeweiligen Ab-

wehrmechanismen einmal mit dem ursprünglichen Triebzustand und ein zweites Mal mit dem veränderten Triebzustand aufgerufen. Die Methode stellt fest, ob der übergebene Trieb als `clsDriveMesh` der Vorher- oder Nachherzustand ist und speichert die Information entsprechend in einer `clsChangedDrive` Instanz ab.

5.2.2 Interface

Wenn MASON die eigentliche Visualisierungsklasse `clsInspectorImageDrives` updatet, greift der Inspektor über die Methode `ArrayList<clsChangedDrives> processList()` der Interfaces `itfInspectorModificationDrives` auf das Modul F06 zu, um eine Liste aller Trieb-Änderungen als `clsChangedDrive` Instanzen zu erhalten.

Zum Instanziieren der Visualisierungs-Klasse wird wieder die SiMA-übliche Methode zum automatisierten Generieren von Inspektoren verwendet. Dabei wird in der Methode `addAutocreatedInspectors(TabbedInspector, clsPsychicApparatus, String)` der Klasse `clsInspectorTab_Modules` bei allen Modulen, die das Interface `itfInspectorModificationDrives` implementieren, eine Instanz von `clsInspectorImageDrives` angelegt.

5.2.3 Visualisierungs-Klasse

Die neu erstellte Visualisierungs-Klasse `clsInspectorImageDrives` ist von der MASON–Inspektor-Basisklasse `Inspector` abgeleitet und hat die Funktionalität, Änderungen in der Triebstruktur zu visualisieren. Die MASON-Klasse `Inspector` stellt das Fenster und eine update-Methode `updateInspector ()`, die von MASON in der Visualisierungsphase aufgerufen wird, zur Verfügung. `clsInspectorImageDrives` überlädt die update-Funktion und nutzt sie, um die Daten des letzten Schrittes über das Interface `itfInspectorModificationDrives` aus dem SiMA-Modell in die Visualisierungs-Klasse zu kopieren und anzuzeigen.

Die Funktionalität der Visualsierungs-Klasse kann grob in zwei Phasen aufgeteilt werden, nämlich in die sogenannte Initialisierungsphase und die sogenannte dynamischePhase. In der Initialisierungsphase werden verschiedene vorbereitende Schritte durchgeführt. Als Erstes werden die Pfade für die Grafiken der Triebinhalte festgelegt. Danach werden die visualisierungsspezifischen Grafiken geladen, z.B. die Bilder für „ansteigend“ und „abfallend“. Als nächstes wird das Fensterlayout festgelegt, in dem die Fenstermanager entsprechend initialisiert werden.

Abbildung 5.4 zeigt die verwendeten Layout-Manager. Dabei werden die GUI-Toolkits java.awt und javax.swing parallel eingesetzt. Die Layouts sind von einem JScrollPane (javax.swing) umschlossen, wodurch es möglich sein soll, über den Inhalt des gesamten Fensters zu scrollen, falls der Anzeigebereich zu klein ist. Es erscheint unsinnig, das Fenster unbegrenzt verkleinern zu lassen, da die verwendeten Piktogramme erst ab einer gewissen Größe erkennbar sind. Als Top-Layout wird ein Border-Layout (java.awt) verwendet, das den Anzeigebereich in drei Spalten unterteilt. Alle drei Spalten werden weiteres vom separaten GridLayout (java.awt) in je vier Zeilen unterteilt. In der linken Spalte werden später die Bilder zur Triebidentifizierung angezeigt. In der rechten Spalte kommen später die Tortendiagramme, die das Auftreten der Abwehrmechanismen dynamisch abbilden. In der mittleren

Spalte wird in jeder Zeile von GridLayout jeweils ein BoxLayout (javax.swing) eingefügt, das später dynamisch die Piktogramme der Simulations-Schritte als Sequenz von links nach rechts darstellt.

Abbildung 5.4: Fenstermanager-Layout

Im Zuge der Initialisierungsphase wird das Interface `itfInspectorModificationDrives` benutzt, um die Referenz der im F06 erzeugten Liste aus `clsChangedDrives` zu erhalten. Die Liste funktioniert als geteiltes Objekt, da beide Klassen eine Referenz darauf haben, und wird benutzt, um Informationen über den Vorher- und Nachher-Zustand aller Triebe zwischen den SiMA-Modulen und der Visualisierung zu transportieren.

In einem nächsten Schritt werden die Tortendiagramme in den oben erwähnten Zellen des GridLayouts angelegt und mit ihren Datenquellen vom Typ `org.jfree.data.general.DefaultPieDataset` verbunden. Die Datasets werden im dynamischen Teil innerhalb der update-Methode benutzt, um die Häufigkeit der Abwehrmechanismen abzuspeichern. Für die Tortendiagramme wird das Diagrammframework JFreeChart [1] verwendet. Als Diagramm wird ein `org.jfree.chart.plot.PiePlot` genutzt. Danach werden Bilder zur Repräsentation der Triebquelle geladen, angepasst und in den entsprechenden Zellen des GridLayouts eingefügt.

Die dynamische Phase wird von MASON durch den Aufruf der `updateInspector ()` – Methode eingeleitet. Als Erstes wird die oben erwähnte Liste aus `clsChangedDrives`-Instanzen auf Änderungen im letzten Schritt überprüft und aktualisiert, mit der Datenquelle der Tortendiagramme wird dementsprechend verfahren. Als Nächstes wird für den aktuellen Schritt, veranschaulicht in Abbildung 5.5, ein JPanel mit fixer Größe und GridLayout mit zwei Zeilen und einer Spalte erstellt und abhängig davon, ob eine Änderungen der Trieblage gefunden wurde oder nicht, entweder ein Piktogramm mit der Schrittnummer und einem Leerbild, siehe Abbildung 5.7, oder das in Abbildung 5.6 gezeigte Änderungspiktogramm in die Zellen eingefügt.

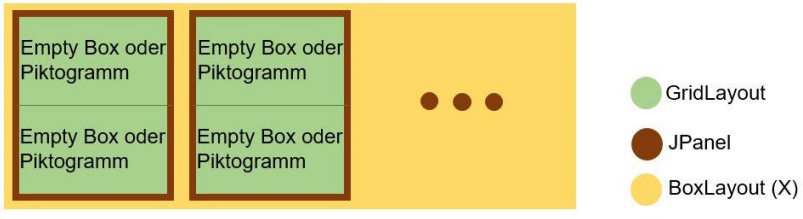


Abbildung 5.5: Anknüpfung vom statischen Layout, **Abbildung 5.4**, zum dynamischen. Dieser Ausschnitt zeigt eine der gelben BoxLayout-Zeilen im **Abbildung 5.4**

Zum Erstellen eines Piktogramms wie in Abbildung 5.6 wird zuerst eine Zeichenfläche in Form eines JPanel in der einer vorgegebenen Größe und einem vertikal orientieren BoxLayout erstellt. Das BoxLayout bewirkt, dass alle ins Panel eingefügten Komponenten untereinander angeordnet werden. Als Erstes wird ein neues JPanel mit dem default Layout und einem einzelnen JLabel, das die Schritt-Nummer ausgibt, hinzugefügt. Die nächsten zwei Einträge sind JPanels mit horizontal orientierten BoxLayouts, die jeweils wieder zwei JLabels mit ImageIcons enthalten. Diese beiden JPanels werden später die Veränderung des Triebobjektes und die Veränderung des Triebzieles visualisieren. Als Nächstes wird in das ersterwähnte BoxLayout wieder ein JPanel eingefügt, das ein horizontal orientiertes BoxLayout mit zwei JLabels aufweist. Das linke Label wird später die textuelle Ausgabe des Affektbetrages enthalten und im rechten Label wird ein ImageIcon mit einem Tendenzindikator implementiert werden. Als finales Element wird ein JPanel mit defaultLayout eingefügt, das das JLabel mit einer textuellen Ausgabe des Abwehrmechanismus enthält.

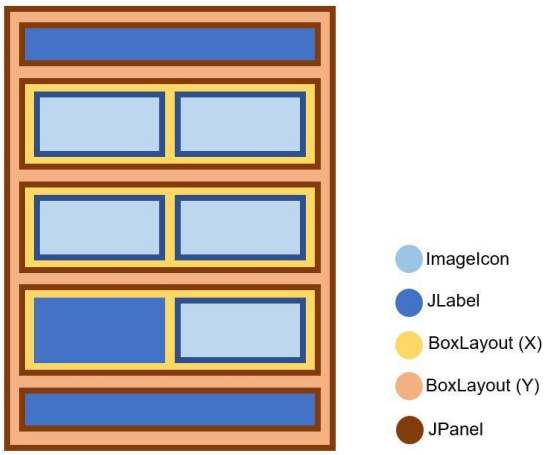


Abbildung 5.6: Aufbau des Piktogramms

Das Piktogramm „Empty Box“ visualisiert in der Abbildung 5.7 ist signifikant einfacher als das zuvor beschriebene. Hier wird ein JPanel mit der selben Größe wie das im Bereich der „Full Box“ in Abbildung 5.6 erstellt. Als Layout Manager wird ein BoxLayout mit vertikaler Orientierung verwendet. In diesem Layout werden zwei JLabels eingefügt. In das obere Label kommt eine textuelle Ausgabe der Schrittnummer, das untere Label wirdein ImageIcon von einem leeren Bild enthalten.

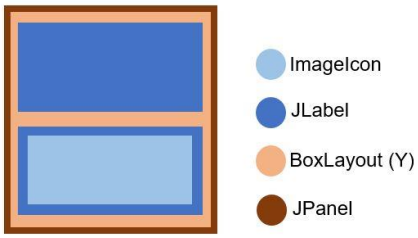


Abbildung 5.7: Empty Box

5.3 Inventar

Die Grundfunktionalität des Inventars ist im SiMA-Modell als Klasse `clsInventory` bereits vorhanden, wird aber auf Grund mangelnder Anknüpfung zum Rest des Modells nicht verwendet, konkret fehlt dem SiMA-Agenten aktuell die Möglichkeit, Objekte ins oder vom Inventar zu transferieren. Im Zuge der vorliegenden Diplomarbeit soll das Inventar nutzbar gemacht werden, also durch Implementierung der notwendigen Aktionen „pick-up", „drop" und „move to/from inventory" verwendbar und mittels eines Inspektors visualisierbar sein. In den folgenden Unterkapiteln wird beschrieben, wie die bereits existierende Inventarumsetzung aussieht, welche Änderungen und Erweiterung im Code vorgenommen werden mussten, welche Änderungen an den Schnittstellen sich daraus ergaben und wie der neue Inventar-Inspektor umgesetzt worden ist.

5.3.1 Umsetzung

Das Inventar ist als Teil der Superklasse für „Bewegliche Objekte" implementiert. Das bedeutet auch, dass das Inventar nicht nur für Agenten verfügbar ist, sondern für alle mobilen Objekte (`clsMobile`). Diesbezüglich ist anzumerken, dass es dem Zweck der vorliegenden Diplomarbeit entsprechend nur notwendig war, das Inventar des Agenten zugänglich zu machen, weshalb auf das Inventar unbelebter Objekte weiterhin nicht zugegriffen werden kann.

Das Inventar ist als eigene Klasse (`clsInventory`) realisiert. Die Referenzen auf alle aktuell mitgeführten Objekte sind lokal in einer `java.util.HashMap` dargestellt, die Referenz auf das aktuell getragene Objekt als private member-Variable von Typ `clsMobile`. Das Gewicht der getragenen Objekte wirkt sich direkt auf die aktuelle und die maximale Traglast (zugreifbar über `clsInventory::getMass()` und `clsInventory::getMaxMass ()`) des Agenten aus. Diese Werte werden zur Bestimmung des aktuellen Ausdauerverbrauches (berechnet in `clsActionProcessor::ConsumeBindingEnergy()`) und in weiterer Folge des Energieverbrauchs herangezogen. Im Körper des Agenten wird zwischen Ausdauer [(1)] und Energie [(2)] unterschieden. Ausdauer wird benutzt um den Agenten in der Spielwelt zu bewegen, das bedeutet, wenn die Ausdauer auf 0 fällt, bleibt der Agent unbeweglich, wodurch die Ausdauer wieder regenerieren kann. Die Energie beschreibt die „Grundversorgung" des Agenten (z.B. mit Nährstoffen), fällt diese auf 0, stirbt der Agent.

Beides wird anhand folgender Formeln ermittelt:

$$\boldsymbol{Verbrauch_{Ausdauer} = \frac{Gewicht_{akteull}}{Gewicht_{maximal}} * Einflussfaktor_{Ausdauer}} \tag{1}$$

$$\boldsymbol{Verbrauch_{Energie} = Verbrauch_{Ausdauer} * Einflussfaktor_{Energie}} \tag{2}$$

Sämtliche bereits existierenden Funktionalitäten wurden nun durch die Implementierung neuer Aktionen und ihrer Exekutoren für den Agenten verfügbar gemacht. Die Aktionsausführung ist im `clsActionProcessor` angesiedelt, wobei die vom Agenten ausgewählte Aktion (Subklasse von `clsActionCommand`) bestimmt, welcher Exekutor (Subklasse von `clsActionExecutor`) ausgeführt wird, um mit dem Spielfeld zu interagieren.

Es wurden folgende Aktionen neu implementiert

„Aufheben": entfernt ein Objekt aus der Spielwelt und referenziert es als aktuell getragenes Objekt „in der Hand". Ist „die Hand" gerade belegt, wird das aktuell gehaltene Objekt vorher automatisch in die Spielumgebung abgelegt.

- `clsActionPickUp` erbt von `clsActionCommand`
- `clsExecutorPickUp` erbt von `clsActionExecutor`

„Ins Inventar ablegen": transferiert das aktuell gehaltene Objekt „aus der Hand" ins Inventar.

- `clsActionMoveToInventory` erbt von `clsActionCommand`
- `clsExecutorMoveToInventory` erbt von `clsActionExecutor`

„Aus dem Inventar nehmen": transferiert ein Objekt aus dem Inventar „in die Hand". Ist „die Hand" gerade belegt, wird das aktuell gehaltene Objekt vorher automatisch in die Spielumgebung abgelegt.

- `clsActionMoveFromInventory` erbt von `clsActionCommand`
- `clsExecutorMoveFromInventory` erbt von `clsActionExecutor`

„Das Objekt ablegen": legt das aktuell „in der Hand" gehaltene Objekt in die Simulationswelt an der Stelle ab, wo sich der Agent gerade befindet.

- `clsActionDrop` erbt von `clsActionCommand` (diese Klasse war bereits da, konnte aber noch nicht verwendet werden) und
- `clsExecutorDrop` erbt von `clsActionExecutor`

Der technische Ablauf, wenn ein Agent das Inventar verwendet, gestaltet sich wie folgt:

Welche Aktionen verfügbar sind, wird durch die Erinnerung des Agenten bestimmt, die in einer Wissensdatenbank abgelegt sind. Im Zuge der Entscheidungsfindung werden (vereinfacht ausgedrückt) die bekannten Aktionen anhand ihrer kurz und langfristigen Vorteile für den Agenten bewertet und ausgewählt. Diese beiden Aspekte, also die Repräsentation in den Erinnerungen und der Vorgang in der Entscheidungsfindungseinheit, werden in dieser Arbeit nicht erläutert. Für Details siehe [Zeil10, pp.96-109, Wend16 pp.180-185].

Wenn sich der Agent entschieden hat, ein Objekt in das Inventar zu transferieren, erfolgt dies in zwei Stufen: Zuerst muss der Agent die Aktion „pick-up" und danach die darauffolgende Aktion „put-to-inventory" ausführen, siehe dazu Abbildung 4.8. Der Ablauf ist in beiden Fällen der gleiche, weshalb er im nächsten Absatz für die Aktion „pick-up" genauer beleuchtet wird:

Der Grund für die Aktion „pick-up" ist immer die Verfolgung eines Zieles. Wenn beispielsweise in einem Teilszenario Adam sein Essen mit Bodo teilen will, wäre Adams erster Schritt, das Essen in die Hand zu nehmen und zu Bodo zu tragen. Die gewollte Aktion ist also immer als nächster Schritt zum

Erreichen des aktuellen Ziel geplant, dem die Aktion „pick-up“ logisch vorausgeht. In den Modulen F29 und F30 wird die Aktion „pick-up“ als Word Presentation Mesh (WPM) aus dem Ziel extrahiert und an F31 weitergeleitet. F31 befindet sich in der Schicht zwei, siehe dazu Unterkapitel „Schichtenmodell“, und ist dafür verantwortlich das WPM in eine für den Simulator verständlichen Befehl umzuwandeln und für den Transfer zum Simulator aufzubereiten [Herr14, pp.52-53]. Im Simulator wird der Befehl in der Klasse `clsComplexBody` in eine von `clsActionCommand` abgeleitete Klasse umgewandelt (in unserem Fall `clsActionPickUp`) und an den `clsActionProcessor` weitergereicht. Hier wurde bereits beim Initialisieren festgelegt, welche von `clsActionCommand` abgeleiteten Klassen durch welche von `clsActionExecutor` abgeleiteten Klassen im Simulator umgesetzt werden sollen. Für die Aktion „pick-up“ erfolgt die Verknüpfung von der Aktion „pick-up“ zum Exekutor über folgenden Aufruf:

```
addCommand(clsActionPickUp.class, new clsExecutorPickUp(…));
```

Nachdem alle Aktionen des aktuellen Zyklus vom Action-Processor weitergereicht wurden, ruft dieser die Execute-Methode der entsprechenden, von `clsActionExecutor` abgeleiteten Exekutor-Klassen (in unserem Fall `clsExecutorPickUp`) auf. In den Exekutor-Methoden wird in weiterer Folge auf die Simulationsumgebung zugegriffen, im Fall vom „pick-up“ das Objekt aus der Simulationsumgebung entfernt und die entsprechender Referenz im Inventar als „gehaltenes Objekt“ gespeichert. Derselbe Prozess erfolgt für die Aktion „move to inventory“ mit der einzigen Abweichung, dass im `clsExecutorMoveToInventory` die Execute-Methode die Referenz des gehaltenen Objektes in die Liste der eingepackten Objekte verschiebt. Auch das Ablegen von Objekten erfolgt äquivalent mit den Aktionen `clsActionMoveFromInventory` und `clsActionDrop`.

5.3.2 Schnittstellen

Die Schnittstellen des Inventars lassen sich grob anhand ihrer Kommunikationspartner in Schnittstellen zu Exekutoren, Schnittstellen für die Visualisierung und Schnittstellen für die Parametrisierung unterteilen. Die Schnittstelle zu den Exekutoren wird – wie oben beschrieben – genutzt, um die physische Interaktion zwischen dem Inventar und die Simulationsumgebung zu ermöglichen. Die dabei involvierten Methoden des Inventars sind:

- `void setCarriedEntity(clsMobile)`. Die angesprochene Methode übergibt dem Inventar die Referenz auf ein Objekt, das vom Inventar als aktuell getragenes Objekt abgelegt wird. Wenn sich die Referenz in der Liste der mitgeführten Objekte befindet, wird sie von dort entfernt. Falls der Agent bereits ein getragenes Objekt in der Hand hält, wird dieses automatisch abgelegt.
- `void dropCarriedItem()`. Diese Methode transferiert die Referenz des aktuell getragenen Objektes zurück in die Simulationsumgebung an die aktuelle Position des Agenten.
- `void moveCarriedEntity(Double2D)`. Diese Methode transferiert die Referenz des aktuell getragenen Objektes an einer bestimmten Stelle in der Simulationsumgebung.
- `void moveCarriedToInventory()`. Diese Methode transferiert die Referenz des aktuell getragenen Objektes in die Liste der mitgeführten Objekte.

- `clsMobile getCarriedEntity()`. Diese Methode retourniert die Referenz des aktuell getragenen Objektes, ohne es zu entfernen. Diese Methode wird verwendet um festzustellen, ob der Agent ein Objekt in der Hand hält.
- `clsMobile getInventoryItem(int)`. Diese Methode retourniert die Referenz eines mitgeführten Objektes an einer bestimmten Stelle in der Liste der mitgeführten Objekte, ohne es zu entfernen. Sie wird zusammen mit `setCarriedEntity` verwendet, um ein mitgeführtes Objekt in die Hand zu transferieren.

Die Schnittstelle zur Visualisierung wird genutzt, um den aktuellen Zustand des Inventars für die grafische und die textuelle Aufbereitung in den Inspektoren abzurufen. Sie nutzt die Funktionen:

- `int getMaxItems()`. Diese Funktion retourniert die Anzahl der maximal mitführbaren Objekte, exklusive des getragenen Objektes.
- `int getItemCount()`. Diese Funktion retourniert die Anzahl der aktuell mitgeführten Objekte.
- `double getMaxMass()`. Diese Funktion retourniert das festgelegte maximale Tragegewicht.
- `double getMass()`. Diese Funktion retorniert das aktuelle Tragegewicht.
- `clsMobile getCarriedEntity()`. Diese Funktion retourniert die Referenz des aktuell getragenen Objektes, ohne es zu entfernen. Diese Methode wird verwendet, um die Daten des aktuell getragenen Objekts für die Visualisierung auszulesen.
- `clsMobile getInventoryItem(int)`. Diese Funktion retourniert die Referenz eines mitgeführten Objektes an einer bestimmten Stelle in der Liste der mitgeführten Objekte, ohne es zu entfernen. Diese Methode wird verwendet, um die Daten der aktuell mitgeführten Objekte für die Visualisierung auszulesen.

Die Schnittstelle zum Körper wird ausschließlich verwendet, um den Einfluss des Inventars auf die Ausdauer und Energie des Agenten zu berechnen. Dafür werden die oben beschriebenen Funktionen `getMaxMass()` und `getMass()` verwendet [(1) und (2)].

Die Parametrisierung erfolgt über den Konstruktoraufruf aus `clsMobile`.

5.3.3 Inspektor

Das Inventar war bislang nicht nutzbar, weshalb noch keine Visualisierung benötigt wurde. Aufgrund der Erweiterungen ist es jetzt aber von Interesse, zu wissen, welche Objekte sich im Container befinden, welches Objekt gerade getragen wird, wie viel vom maximal tragbaren Gewicht verwendet wird, wie sich das Gewicht zusammensetzt und wie sich das auf die Ausdauer auswirkt.

Um die notwendigen Informationen unter Verwendung einer Visualisierung zu erhalten, wurde ein neuer Inspektor namens „inventory“ im „Arsin overview“-Tab erstellt. Um die Anforderungen an die Visualisierung zu erfüllen, wurden ein Bereich für die textuelle Ausgabe, ein Tortendiagramm und ein Liniendiagramm in einem Inspektor implementiert. Für die grafischen Darstellungen der Diagramme wird JFreeChart [1] verwendet, für die textuelle Ausgabe und das Fenster-Management wird Swing [9] genutzt.

Das Tortendiagramm visualisiert die Zusammensetzung und Gesamtgewichtsverteilung des Inventars. Die Visualisierung iteriert dabei durch alle Objekte im Inventar, bestimmt ihren Anteil am Maximalgewicht und erzeugt einen entsprechenden farbigen Slice („Tortenstück“). Sofern das Maximalgewicht nicht ausgeschöpft ist, wird das verbleibende Potential ebenfalls durch einen Slice repräsentiert. In jedem Slice wird auch die Identität des dazugehörigen Inventar-Objektes ausgegeben.

Durch das Liniendiagramm wird die zeitliche Entwicklung des aktuellen Gewichtes und der aktuellen Ausdauer (Stamina) ausgegeben. Dabei kann beobachtet werden wie sich das Gewicht des Inventars auf die Ausdauerentwicklung auswirkt. Um das aktuelle Gewicht besser in Relation setzten zu können, wird auch die maximale Traglast im Liniendiagramm angezeigt.

Der Bereich für die textuelle Ausgabe zeigt zusätzlich Informationen zu dem getragenen Objekt wie etwa dessen Identität sowie das Maximalgewicht, das aktuelle Gewicht und eine Liste der Identitäten aller aktuell im Inventar befindlichen Objekte an.

Für die visuelle Ausgabe, siehe Abbildung 4.12: Darstellung des Inventar-Inspektors.

6. Ergebnisse

In diesem Kapitel werden die Ergebnisse der Use-Cases aus Kapitel 4 zusammengefasst und ihr Beitrag zum SiMA-Projekt durch Gegenüberstellung mit dem Status vor der Diplomarbeit validiert. Dieses Kapitel unterteilt sich in die Unterkapiteln für die Evaluierung der Neuerungen bei der Eingabe der Über-Ich Regeln, die Evaluierung der Neuerungen beim Abwehr-Inspektor und der Evaluierung der Neuerungen im Inventar. Abschließend wird der Fragebogen zur formalen Evaluierung des Systems und seine Auswertung präsentiert.

6.1 Über-Ich Regeln

Dieses Unterkapitel evaluiert die neuen Methoden zur Spezifizierung von Über-Ich-Regeln anhand der Use-Cases die in Kapitel 4.2 vorgestellt wurden. Dies basiert auf zwei Aspekten. Der eine ist die Gegenüberstellung des neu entwickelten Über-Ich-Inspektors mit den alten text-basierten-Inspektoren. Der zweite Aspekt ist die flexiblere Eingabe anhand der neu entwickelten Regel-Syntax deren Verbesserung im Kapitel 6.4 validiert hätten werden sollen.

6.1.1 Regel 1

Als Ausgangslage wird die textuelle Ausgabe des Inspektors vom Modul F07 betrachtet. Der Screenshot in Abbildung 6.1 kombiniert die ursprünglichen Ausgaben (in grau umrandet) mit den neuen Ausgaben (in blau umrandet). Die farbliche Hinterlegung im Inspektor wurde zum Zwecke der Lesbarkeit eingeführt und im eigentlichen Inspektor nicht verfügbar.

Die ursprüngliche Ausgabe beinhaltet überhaupt keine Darstellung der aktiven oder inaktiven Über-Ich-Regeln, oder der aktuell konflikthaften Inhalte, da die mangelnde Flexibilität der hard-gecodeten-Regeln dies nicht erforderte. Die Erweiterungen des textuellen Inspektors bietet eine Übersicht der aktuellen Filterparameter jeder Regel, gruppiert nach den bereits bekannten Kategorien: Triebe, Wahrnehmung und Emotionen. Die neuen grafischen Inspektoren, deren Ausgaben bereits in Kapitel 4.2.1 dargelegt wurden, visualisieren die selben Inhalten in einer klassischen Graphen-Darstellung. Des Weiteren werden die Identifikatoren, der aktuelle als konflikthaft markierten Triebe, Wahrnehmung und Emotionen textuell beschrieben. Durch die Verknüpfung der neuen und alten Ausgaben ist auch nachvollziehbar, welche der psychischen Inhalte mit welchen Regelteilen übereinstimmen. Z.B. ist die Bedingung „libidinös“ und „stomach“ zwischen 0.2 und 1.0 in der Abbildung 6.1 grün markiert, ebenso wie alle Triebkandidaten im oberen Teil des Inspektors. Man sieht, dass „libidinös“ vier Mal vorkommt, „stomach“

zweimal und nur bei einer Zeile trifft Beides zu. Da sich der QoA (Affektbetrag) ebenfalls im vorspezifizierten Rahmen bewegt, schlägt die vordefinierte Regel an. Der resultierende Trieb-Konflikt mit der Konfliktstärke ca. 0.376 ist ebenfalls grün markiert.

Abbildung 6.1: Regel und Konflikte in der textuellen Ausgabe. Grau umrandet ist die ursprüngliche Ausgabe, blau umrandet sind die neuen Ausgaben. Der dazugehörige Visualisierungs-Graph ist hier blass dazugegeben, kann man in **Abbildung 4.2** nachsehen.

Die Auswirkung hier ist, dass der Trieb „libidinös stomach“ als Konflikthaft markiert wird. Dieser Konflikt wird in weiterer Folge von der Abwehr behandelt.

6.1.2 Regel 1+ 2

Dieser Use-Case ist eine Erweiterung des vorhergehenden im Kapitel davor. Der textuelle Screenshot in Abbildung 6.2 ist gleich aufgebaut wie bei Regel 1 beschrieben. Die Ausführungen des Kapitel 6.1.1 gelten auch hier. Zusätzlich sind folgende Neuerungen zu beobachten: Am oberen Ende des Screenshots ist eine Liste der Triebe zu sehen. Diese wird nun durch eine neue Liste der aktuellen Emotionen ergänzt. Danach kommt die bereits bekannte textuelle Auflistung der Regel-Terme und im unteren Bereich eine textuelle Ausgabe des markierten Konfliktes. Zur genaueren Veranschaulichung wurden die Terme auch im vorliegenden Screenshot grün eingefärbt. Die Übereinstimmungen aus Regel 1 bleiben bestehen, zusätzlich lässt sich eine Übereinstimmung im Bereich der Emotion bei „anxiety“ erblicken.

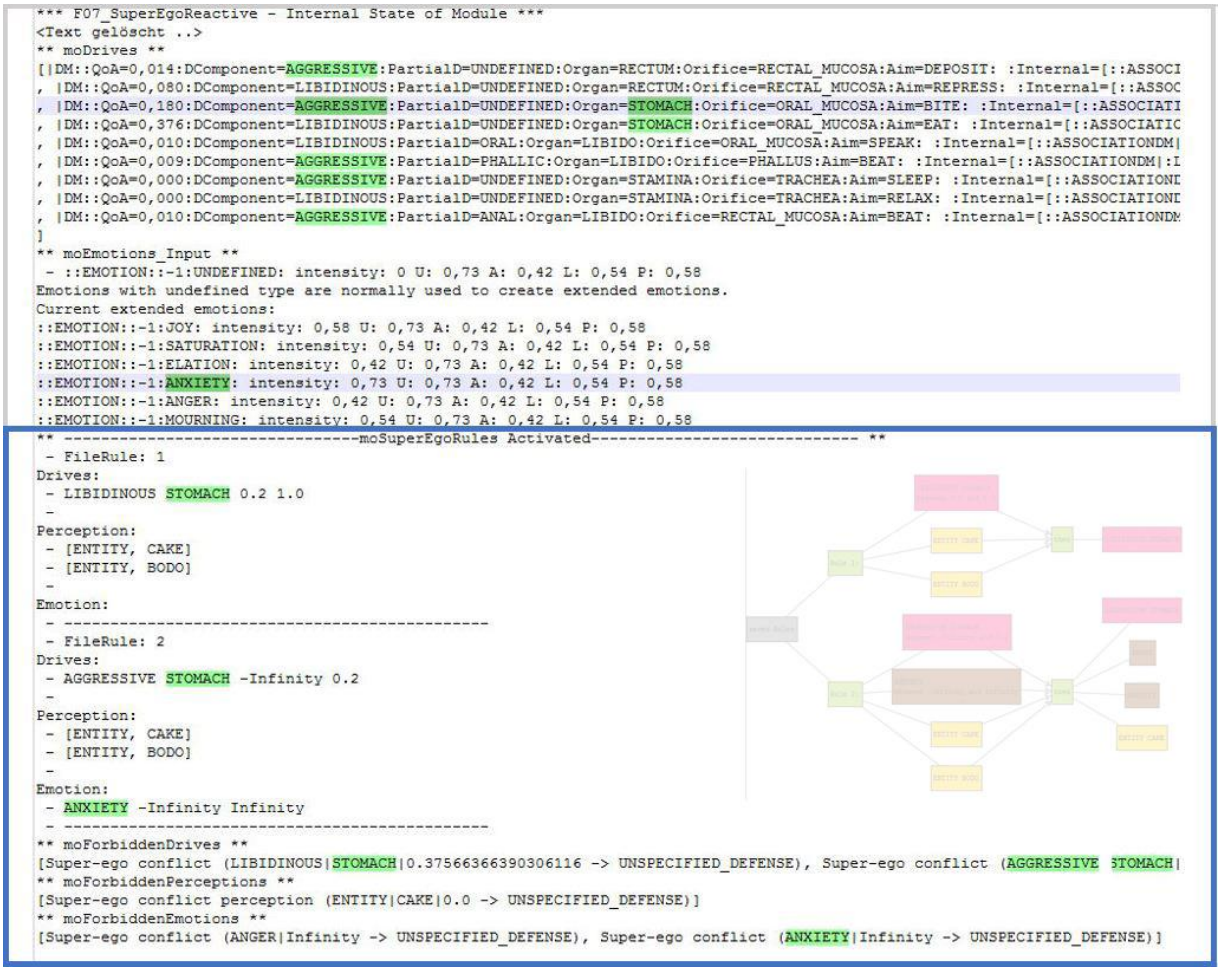


Abbildung 6.2: Regel und Konflikte in der textuellen Ausgabe. Grau umrandet ist die ursprüngliche Ausgabe, blau umrandet sind die neuen Ausgaben. Der dazugehörige Visualisierungs-Graph ist hier blass dazugegeben, kann man in **Abbildung 4.3** als original ansehen.

Da in diesem Beispiel beide oben beschriebene Regeln zutreffen, werden die Triebe „libidinös stomach“ und „aggressive stomach“, das Wahrnehmungsobjekt „CAKE“ und die Emotionen „anger“ und „anxiety“ als konflikthaft markiert. In weiterer Folge wird der entstandene Konflikt der Abwehr des Agenten zur weiteren Behandlung zugeführt.

6.2 Abwehr-Inspektor

Dieses Unterkapitel evaluiert die neue Visualisierungsmethode für Abwehr-Abläufe. Dies stellt eine Erweiterung der existierenden Abwehrinspektoren, erstmals vorgestellt in [Lotfi14, p.55], dar. Die ur-

sprünglichen Inspektoren sind Abwehr-zentriert. Abbildung 6.3 zeigt einen dieser ursprünglichen Inspektoren der einen Überblick über die Aktivität eines spezifischen Abwehrmechanismus während der Laufzeit des Agenten liefert.

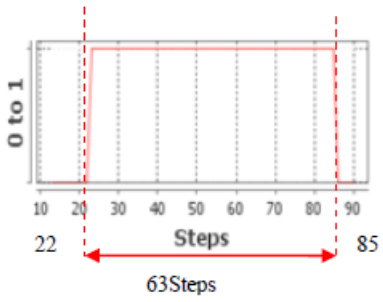


Abbildung 6.3: Abwehrmechanismus „Sublimierung“ über die Laufzeit des Agenten, aus [Lotfi14, p.55]

Der zweite ursprüngliche Inspektor bietet einen Überblick über den Einsatz aller verfügbaren Abwehrmechanismen über die gesamte Laufzeit, siehe Abbildung 6.4.

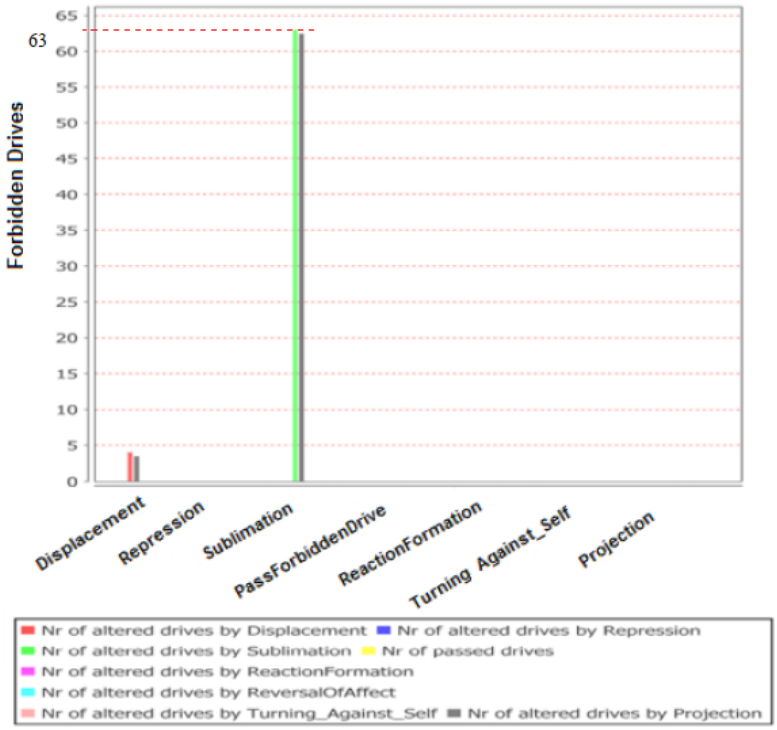


Abbildung 6.4: Anzahl der angeschlagenen Abwehrmechanismen über die gesamte Laufzeit des Agenten, aus [Lotfi14, p.55]

Keiner dieser beiden Visualisierungen erlaubt ein Aufschlüsseln nach Trieben. Ebenso ist es nicht möglich die konkreten Auswirkungen der Abwehrmechanismen abzuleiten. Die neuen Visualisierungen, wie z.B. in Abbildung 6.5, ermöglichen dies jedoch nur über einen begrenzten Zeitrahmen. Die Visualisierung umfasst piktographische Darstellungen der Änderung von Triebzieles, Triebobjektes und des Af-

fektbetrages, aufgeschlüsselt nach Triebquelle (Ausdauer, Magen, Blase und Rektum) und Triebkomponenten (aggressiv und libidinös). Das Tortendiagramm entspricht inhaltlich der Abbildung 6.4, aber wieder aufgeschlüsselt nach Triebquelle und Triebkomponente.

Das Beispiel in Abbildung 6.5 zeigt den neuen Abwehr-Inspektor im Fall einer komplexen, vielfachen Abwehr. Zu der Zeit der Erstellung der vorliegenden Diplomarbeit war die Abwehr im SiMA-Modell noch nicht ausgereift, um diese Art von Komplexität mit vernünftigem Aufwand erzeugen zu können, weshalb die Abwehr hier vollständig durch einen eigens geschriebenen Test-Code ersetzt wurde. Dieser generiert nur die entsprechenden Inputs der Visualisierung unabhängig von der tatsächlichen SiMA-Abwehr. Ein reiner Unit-Test ganz ohne SiMA war nicht möglich, weil der Zusammenhang für das Erstellen der Datenstrukturen einen vielfachen Aufwand bedeutet hätte.

In Abbildung 6.5 lässt sich ganz links die ganze Palette an Trieben, repräsentiert durch ihre Symbole, sehen. Die Batterie symbolisiert den Ausdauertrieb des Agenten. Der Magen repräsentiert den Hunger. Die Blase repräsentiert den liquiden Ausscheidungstrieb, das Rektum repräsentiert den rektalen Ausscheidungstrieb. In den Detailvisualisierungen kommen die Abwehrmechanismen die bereits in den use-Cases aus Kapitel 4.3 noch einmal vor. Hintergrund der gegenständlichen Abbildung soll die Präsentation der Möglichkeit, sich einen Überblick zu verschaffen, sein. Dafür sind die Kreisdiagramme gut geeignet.

Bei Betrachtung der entsprechenden Spalten von oben nach unten zeigt sich sofort, dass in der libidinösen Ausdauer keine Abwehr passiert und in der aggressiven Komponente „Sublimation“ und „Displacement“ im gleichen Ausmaß Anwendung fanden. Aus der nächsten Zeile ergibt sich, dass beim aggressiven Magentrieb eine abwechslungsreiche Abwehr stattgefunden hat, während im Bereich des libidinösen Magentriebs nur eine Art vom Abwehrmechanismus zugeschlagen hat. Das Kreisdiagramm bei der Blase ist wie bei der Ausdauer nur auf den libidinösen Teil ausgelegt. Beim Rektum sieht man sofort, dass bei der aggressiven Triebkomponente die „Sublimierung“ überwogen hat.

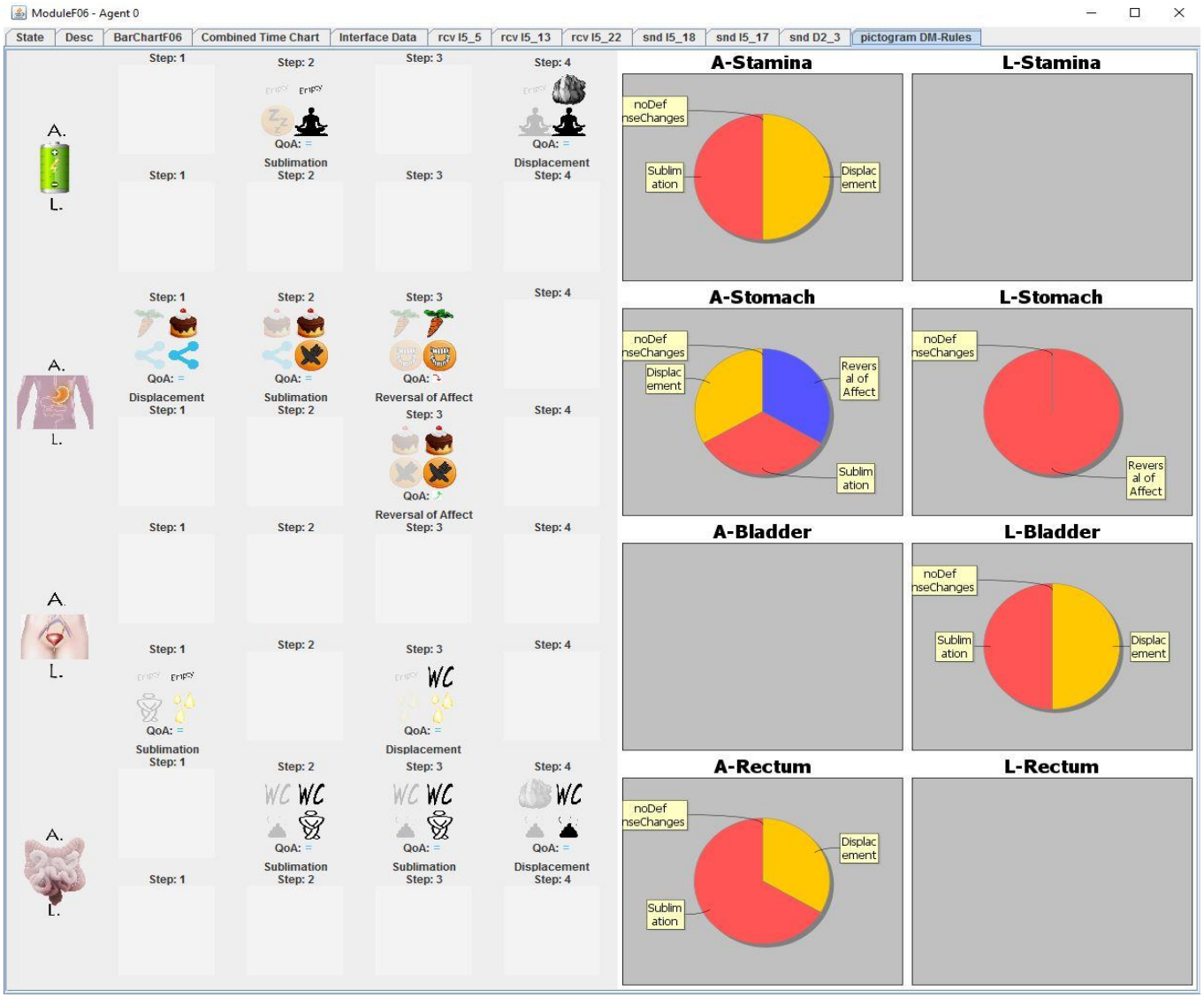


Abbildung 6.5: Visualisierungs-Inspektor von die Piktogramme

6.3 Inventar

Das Inventar wurde gänzlich neu entwickelt, weshalb kein direkter Vergleich möglich ist. Das neue Inventar-System erweitert die Agenten-Funktionalität um die im Kapitel 3.3.2 beschriebenen Konzepte. Zur Validierung, wurden die in Kapitel 4.4.2 beschriebenen Use-Cases in der MASON Simulationsumgebung implementiert und getestet. Die verschiedenen Screenshots in Kapitel 4.4 zeigen die Umsetzung.

6.4 Feedback

Dieses Kapitel beschreibt den Entwurf von Fragen für die formale Evaluierung des Systems durch den psychoanalytischen und die technischen Mitarbeiter des SiMA-Projektes. Aufgrund der Mitarbeiter-Fluktuation im Projekt und mangelnder Förderung des Grundlagenforschungsprojektes, waren zur Evaluierungsphase der Diplomarbeit nur zwei Nutzer für eine formale Prüfung verfügbar.

Die Fragen aus denen der Fragebogen sich zusammenstellt, sind unterteilt in drei Kategorien:

- Allgemeine Fragen,
- Kategorie spezifische Fragen und
- Fragen nur für die Psychoanalytiker

Die Kategorie spezifische Fragen wiederholen sich für die Kategorien „Über-Ich Regeln“, „Abwehr“ und „Inventar“. Die Fragen in der untenstehende Tabelle 5: Fragenkatalog entsprechen ihrer Kategorien.

Allgemein Fragen					
Ausbildung					
Altersgruppe	Bis 20	21 - 30	31 - 40	ab 40	
Computererfahrung	Wenig	Mittel	Viel	Experte	
Farbschwäche	keine	(angeben)			
Kategorie spezifische Fragen					
Wie leicht waren die relevanten Inspektoren zu finden	Offensichtlich	leicht	schwer	Gar nicht	
Waren alle textuellen Ausgaben gut lesbar	Ja	Nein			
War die Strukturierung der Daten verständlich	Ja	Nein			
Waren alle Texte in den Icons lesbar	Ja	Nein			
Waren alle Icons identifizierbar	Ja	Nein			

Waren alle Farben gut unterscheidbar	Ja	Nein			
Wie hoch war die Verzögerung bei der Erstellung der Ausgaben	Irrelevant	Akzeptabel	Enervierend	Unzumutbar	
(Im Fall von Live-anzeigen) War die Anzeige Ruckelfrei	Ja	Nein			
Wurden alle relevanten Informationen angezeigt	Ja	Nein			
Waren Zusammenhänge zwischen den Daten nachvollziehbar	Ja	Nein			
Sind alle Abkürzungen erklärt	Ja	Nein			
Ist die Applikation abgestürzt	Ja	Nein			
Zeigen die Inspektoren nur relevant Informationen	Ja	Größtenteils	Teilweise	Kaum	
Ist die Über-Ich Regel im File Verständlich	Ja	Nein			
Welche Features haben Ihnen gefehlt?					
Was hat Sie bei der Nutzung gestört?					
Was hat Sie bei der Ausgabe gestört?					
Psychoanalytiker-Fragen					
Entsprechenden die verwendeten Farben ihren Erwartungen (in Hinsicht auf die Emotionale Bedeutung des repräsentierten Inhalts)	Ja	Nein			
Entspricht die Über-Ich Regel dem Psychoanalytischen Vorbild	Ja	Nein			1
Ist die Über-Ich Regel im File für Sie lesbar	Ja	Nein			1
Wie vollständig ließ sich die geplante Regel beschreiben	Komplett	Größtenteils	Kaum	Gar nicht	1
Wie akkurat ließ sich die geplante Regel beschreiben	Exakt	Größtenteils	Kaum	Gar nicht	1

Tabelle 5: Fragenkatalog

1 … nur für die Über-Ich-Regel-File-Kategorie

Auswertung:

Die User stammen aus der Altersklasse 31-40 und 40+, beide stufen sich selbst Computer-Experten ein und weisen keine Farbschwäche auf.

Im Folgenden werden nur die Anregungen zur Verbesserung des Systems angeführt. Alle anderen nicht erwähnten Punkte wurden von den Nutzern als „JA“ bzw. mit dem bestmöglichen Eintrag bewertet.

Über-Ich Regeln:

Einer der Usern fand die textuelle Ausgabe, spezifisch im Fall schwarz auf rosa, nicht gut lesbar. (Änderung ist nicht möglich um die Konsistenz zu den Farben in anderen Inspektoren beizubehalten)

Einer der User hätte eine relevante Information vermisst, spezifisch die Zeilennummer der visualisierten Regel im File. (ist bereits als Feature angedacht)

Einer der User fand die Syntax zur Eingabe der Über-Ich Regel immer noch zu umständlich.

Einem der User fehlt die Farblegende im Inspektor.

Einer der User hat von sich aus folgende Aspekte positiv schriftlich vermerkt. Er mochte besonders die übersichtliche Darstellung der Regeln, die Möglichkeit die Regeln aus dem Inspektor auch drucken zu können, die farbliche Codierung und damit die einhergehende gute Unterscheidbarkeit der einzelnen Komponenten.

Abwehr-Visualisierung:

Einer der User bekritzelt die fehlende Abkürzungserklärung beim QoA (für den Affektbetrag), A. (für die aggressive Triebkomponente) und L. (für die libidinöse Triebkomponente).

Einer der User würde sich eine Verlinkung von den der Abwehr-Aktionen zu den auslösenden Über-Regel wünschen.

Einer der User bemängelt fehlende visuelle Abgrenzungen zwischen den verschiedenen Triebquellen.

Inventar:

Einer der User wünscht sich eine kombinierte Aktion aus „Pick-Up“ und „Move-To Inventory“ für einen einfacheren Aufbau der Erinnerungen.

Einer der User bemängelte die fehlende Kompaktheit vom Inventory-Inspektor und die Vermischung aus Grafischerausgabe und Debugg-Ausgabe in der Text-Konsole.

7. Conclusio

Basierend auf der im Kapitel 1.2 dargestellten Problemstellung wurde mit Hilfe der im Kapitel 1.4 beschriebene Vorgangsweisen das SiMA-Modell erweitert und die im Kapitel 3.3 vorgestellten neuen Konzepte entwickelt. Die Anwendbarkeit der Konzepte wird im Kapitel 6 unter zur Hilfenahme der im Kapitel 5 beschriebenen Implementierung aufgezeigt. Um die Arbeit abzuschließen, wird in diesem Kapitel über die gewonnenen Ergebnisse diskutiert und ein Ausblick auf zukünftige Aufgabenstellungen in diesem Gebiet gegeben.

Die Arbeit behandelt mehrere unabhängige Aufgabenstellungen im SiMA-Kontext. Der zentrale und komplexeste Aspekt dabei ist die Weiterentwicklung der Abwehr durch Einführung eines LL(1) Regelparsers inklusive einer psychoanalytisch plausiblen Repräsentationen von Über-Ich-Regeln (Filtermechanismen) sowie die Erstellung einer innovativen Visualisierungsform zur Veranschaulichung der durch die Abwehr herbeigeführten Änderungen. Ein weiterer, in Umfang kleinerer und weniger komplexer Aspekt ist die Erweiterung und Integration eines bestehenden Inventar-Konzeptes für die Nutzung in komplexen Use-Cases. Diese Aufteilung der Aufgabestellung spiegelt sich in der Struktur der Kapiteln 3, 4 und 6 wieder.

7.1 Zusammenfassung

Das Modell SiMA wird seit über 15 Jahren als interdisziplinäre Arbeit von Psychoanalytiker, Psychologen, Philosophen und Technikern verschiedener Sparten entwickelt. Ziel der jahrelangen Bemühungen ist es, die menschliche Entscheidungsfindung, in all ihre Stärken und Schwächen, zu simulieren. Als Ausgangspunkt dienen psychoanalytische Modelle, primär das zweite topografische Modell von S. Freud. Die Entwicklung geschieht nach dem Top-Down Ansatz, bei dem sich die Simulation bei jeder Entwicklungsiteration weiter dem psychoanalytischen Vorbild annähert. Dabei ist es von besonderer Bedeutung, dass sich die Simulation nicht nur in ihren Ergebnissen dem menschlichen Vorbild annähert, sondern vor allem in den internen Abläufen. Das bedeutet, dass im SiMA die Funktionen modelliert werden, nicht aber das Verhalten des Agenten. Das Verhalten bezeichnet in diesem Zusammenhang die Antwort (Output) auf einen bestimmten Input. Da nicht alle Reaktionen des Systems (Outputs) auf alle Inputs beobachtet werden können, wäre eine Modellierung anhand des Verhaltens nie vollständig. Deshalb werden die dem Verhalten zu Grunde liegende Funktionen im Software-Agenten programmiert, aus denen sich dann das Verhalten des Agenten ohne weiteren Eingriff ergibt. Ob die Funktionen das korrekte Verhalten erzeugen, wird anhand von Simulationen überprüft. Bei Verhaltensabweichungen werden die zu Grunde liegenden Funktionen entsprechend modifiziert oder

erweitert, um das neue Verhalten zu integrieren. Die Annäherung wird anschließend durch immer komplexer werdende, weitere Simulationsabläufe verifiziert. In der aktuellen Iteration ist das bestehende System von Filterregeln an seine Grenzen gestoßen und musste durch ein dynamisches System zu Formulierung von Über-Ich-Regeln ersetzt werden. Dies ist ein Hauptaspekt der vorliegenden Diplomarbeit. Darüber hinaus war es notwendig zur Verifikation der neuen Funktionalität eine neue Visualisierung dieser Abläufe zu entwickeln. Zusätzlich musste eine neue Visualisierung der Abläufe innerhalb der Abwehr entwickelt werden, die erkennbar macht, ob die Abwehr sich aus den „richtigen Gründen" korrekt verhält. Ein weiterer Beitrag dieser Diplomarbeit ist die Inbetriebnahme einer Inventarfunktionalität und ihre Visualisierung. Aufgrund der hohen Komplexität von SiMA haben die einzelnen Experten jeweils ihren eigenen Fokus innerhalb des Modells. Diese unterschiedlichen Fokusse beeinflussen einander sehr stark, da alle Module mit ihrer Arbeit einen gemeinsamen Datenstrom erwirken. Aus diesem Grund war es im Zuge der Diplomarbeit immer wieder notwendig, die nächsten Schritte mit den anderen Teammitgliedern abzustimmen. Dadurch kam für die Planung das Spiralmodell, das auf die speziellen Anforderungen des SiMA-Modelles angepasst wurde, als beste Lösung in Frage. Eine besondere Rolle hat dabei die Abstimmung der einzelnen Entwicklungsphasen mit den SiMA-Experten gespielt.

In der Planungsphase wurden die Anforderungen an die Über-Ich-Regeln, ihre Visualisierungen und an das Inventar festgelegt. Die Anforderungen der Über-Ich-Regeln kamen vorwiegend von den Psychoanalytikern, für die der zentrale Punkt war, dass die Regeln ähnlich wie ausgesprochene Befehle – in der Version auf low-level – formuliert und gespeichert werden können. Hinsichtlich der Visualisierung wurden neben der Aufgabenstellung, die Änderungen in der Abwehr darzustellen, kaum Anforderungen gestellt. In Bezug auf das Inventar hat sich herausgestellt, dass die Klassen-Struktur für das Inventar bereits vorhanden war und nur noch mit Funktionalität versehen werden musste.

In der Phase Verifikation und Validation wurden die ursprünglich herausgearbeiteten Anforderungen mit den anderen am SiMA arbeitenden Experten abgeklärt, um Konflikte oder Synergien frühzeitig erkennen zu können.

Zeitlich war die Erstellung des Entwicklungsplans ein wesentlicher Aufgabenbereich. Hier war es für das Verständnis der Schnittstellen und Anknüpfungspunkte der einzelnen Aufgabenblöcke notwendig, sich intensiv mit dem SiMA-Modell und der aktuellen Implementierung auseinander zu setzen. Als Ergebnis dieser Phase wurden die im Kapitel 3.2 aufgezeigten Modell-Schnittstellen identifiziert, aus denen dann die in Kapitel 5 verwendeten Implementierungs-Schnittstellen hergeleitet wurden.

Im Zuge eines ersten Grobentwurfs wurden in der Folge die Umsetzungsmöglichkeiten für die einzelnen Aufgabenpunkte entwickelt. Für die Über-Ich-Regeln wurden verschiedenen Regel-Repräsentationen durchdacht und gegeneinander abgewogen, mit der Schlussfolgerung eine eigene Darstellungsform zu entwickeln, siehe dazu Kapitel 3.3.1. Für die Abwehrvisualisierung wurde ein Konzept basierend auf piktografischen-Darstellungen der „Vorher-" und „Nachher-" Zustände entworfen, welches die selben Piktogramme verwendet, die auch im Spielfeld genutzt werden, um die Position der Objekte anzuzeigen, siehe dazu Kapitel 3.3.3 Unterpunkt 2. Für die Anknüpfung des Inventars wurde in dieser Phase eine Liste der notwendigen Aktionen für die Erfüllung der aktuellen Use-Cases zusammengestellt.

Der Grobentwurf wurde in der Phase der Verifikation und Validation nochmals sowohl mit dem Psychoanalytiker als auch mit dem SiMA-Team anhand von beispielhaften Abläufen durchgegangen.

Die Testplanung wurde in den SiMA-Testplan integriert. Der SiMA-Testplan nutzt psychoanalytische Use-Cases, um aufzuzeigen, dass das Modell menschenähnlich entscheidet. Für die in dieser Diplomarbeit behandelten Aufgabenbereiche wurden eigene Komponententests entworfen, um die Funktionalitäten, die von den SiMA-Use-Cases nicht abgedeckt werden, validieren zu können.

Im nächsten Schritt wurde im Rahmen der Erstellung eines Feinentwurfs die Syntax für die Über-Ich-Regeln konkretisiert, siehe dazu im Kapitel 3.3.1 die Abbildung 3.7 und die nötigen Schnittstellen zu Modell und Visualisierung ausformuliert. Außerdem wurde eine Filter-Klasse mit ihren Transferdatenstrukturen entworfen. Für die Abwehrvisualisierung wurden händische Prototypen erstellt und in Abstimmung mit den anderen Teammitgliedern fixiert. Für das Inventar wurde die Funktionalität der neuen Aktionen fixiert und die dafür notwendigen Erinnerungen in der SiMA-Ontologie entworfen.

In weiterer Folge wurde mit der Implementierung und Integration begonnen. Dafür wurden mit den Komponenten für die Aufgabenpakete direkt an die SiMA-Implementierung angeknüpft. Da das SiMA-Modell eine komplexe, grafenbasierte Datenstruktur als zentrale Datenstruktur verwendet und diese ohne Verwendung des Gesamtmodells nicht nachgebildet werden kann, war es nicht möglich die Komponenten alleinstehen zu entwickeln,. Die Komponenten wurden anschließend nacheinander entwickelt.

Im Rahmen der Testphase wurden die in der Planungsphase entworfenen Komponententests ausgeführt und ihre Ergebnisse im Kapitel „Ergebnisse“ dokumentiert. Die Abnahme erfolgte anhand der Ergebnisse der einzelnen Use-Cases nach Integration der neuen Komponenten durch die jeweiligen Experten. Die am Anfang der Arbeit formulierten Anforderungen konnten allesamt vollumfänglich erfüllt werden, was positiv zur Weiterentwicklung des SiMA-Modells beigetragen hat.

7.2 Schlussfolgerungen

Im Kapitel 3.3.1 wurden verschiedener Speicherformate im Vergleich dargelegt und schlussgefolgert, dass die existierende Darstellungsformen von Regeln nicht für die Darstellung von Über-Ich-Regeln geeignet sind, da sie für Nicht-Techniker meist schwer zu verstehen sind. Im Zuge dieser Arbeit, besonders anhand der Ergebnisse des Kapitels 6, wurde jedoch gezeigt, dass ein Speicherformat, welches Psychoanalytiker lesen und verstehen können, durchaus realisiert werden konnte, ohne auf Flexibilität der Filtermechanismen verzichten zu müssen. Die Integration des neuen Über-Ich-Parserkonzeptes hat ebenfalls friktionsfrei funktioniert. Die Auswertung der in Kapitel 6.1zusammengefassten Auswertungen, der in Kapitel 4.2 vorgestellten Use-Cases, zeigen, dass es möglich ist, realistisch und psychoanalytisch inspirierte Regeln abbilden zu können. Es wurde auch gezeigt, dass die existierende Abwehr mit den neu erzeugten, dynamischen Regeln funktioniert. Die Speicherung im neuentwickelten Fileformat hat sich im Zuge der Tests bewährt. Einige dieser Tests wurden in Kapitel 4.2 exemplarisch dargestellt.

Die Verwendung von Piktogrammen für die Visualisierung von Abwehrabläufen hat gut funktioniert. Sie ermöglicht auch Nicht-Technikern, die Details der Veränderung durch die Abwehr schnell und

akkurat wahrnehmen zu können. Kombiniert mit den bereits existierenden Inspektoren trägt der neue Inspektor zu einem besseren Überblick darüber, wie sich die Abwehr zuletzt verhalten hat, bei. Als problematisch hat sich der Platzbedarf der Piktogrammdarstellung erwiesen, wodurch es schwierig wird, die Entwicklung der Abwehr über lange Zeit zu dokumentieren.

Die Integration des existierenden Inventars war problemlos möglich. Die erfolgreiche Umsetzung der in Kapitel 4.4 vorgestellten Use-Cases, weißt darauf hin, dass die eingeführten neuen Aktionen für die Aufgaben der aktuellen SiMA Use-Cases ausreichend sind. Die neuen Inspektoren haben sich bezüglich des Inventars ebenfalls sehr gut bewährt. Als problematisch hat sich dabei die Berechnung der Ausdauer erwiesen, die nicht Teil der vorliegenden Diplomarbeit war. Diesbezüglich war zu bemerken, dass der Ausdauer durch die Integration des Inventars eine deutlich wichtigere Rolle zukommen wird. Das resultierende Problem wurde an den entsprechenden Entwickler kommuniziert. Da das Problem zum Zeitpunkt der Fertigstellung nicht gelöst wurde, hat der Liniengraf des Inventar-Inspektors derzeit in der Praxis wenig Aussagekraft.

Zusammenfassend lässt sich festhalten, dass die in Kapitel 1 formulierte Aufgabenstellung im Zuge dieser Diplomarbeit erfolgreich erfüllt wurde.

7.3 Ausblick

Abschließend sollen noch die Möglichkeiten, zusätzliche Funktionalitäten zur Erweiterungen der im Rahmen der vorliegenden Diplomarbeit in Kapitel 3.3 vorgestellten Modelle und Konzepte zu entwickeln, aufgezeigt werden:

Bei den Über-Ich-Regeln wäre es für die Zukunft sinnvoll, eine Funktionalität einzurichten, mit der der Abwehr mehrere statt nur eine Über-Ich Regel-Datei mitgegeben werden kann. Dadurch wäre es möglich, die Regeln – geordnet nach ihrem Sinn – in separate Dateien aufzuteilen, etwa nach Regeln, die für alle Agenten gelten, Regeln für spezifische Szenarien, Regeln für die einzelnen Simulationsläufe, Regeln für bestimmte Stimmungszustände eines Agenten, und diese an verschiedene Use-Cases übergeben. Das neue Konzept müsste in der Folge natürlich auch in der Regelvisualisierung berücksichtigt werden, beispielsweise indem alle Regeln aus derselben Datei an einen gemeinsamen Anfangs-Knoten, der mit den Dateinamen beschriftet ist, geknüpft werden (statt des aktuellen und einzigen Anfangsknotens „saved Rules“). Zusätzlich wäre es sinnvoll, die Angabe der Zeilennummer, in der die Regel in der Datei auftritt, anzugeben, anstatt bloß die fortlaufende Regelnummer. Dies wäre etwa sinnvoll, um dem Anwender die Kontrolle zu erleichtern, ob und wo die Regel vom Parser aufgenommen wurde. Derzeit besteht nämlich das Problem, dass manche Regeln wegen zu kleiner Super-Egos gar nicht berücksichtigt werden.
Ein weiterer Anknüpfungspunkt für die Weiterentwicklung des SiMA-Modells wäre die Erweiterung des Parsers und der Regelsyntax um zusätzliche Bedingungstypen, wie z.B. erinnerungsbasierte Bedingungen.

Bei der Abwehrvisualisierung mittels des Piktogramm-Inspektors wäre eine zusätzliche Visualisierung zur Betrachtung der Abwehraktivität über längere Zeiträume sinnvoll. Der aktuelle Inspektor würde bei einer Betrachtung von mehr als 20 Schritten sehr langsam und unübersichtlich werden. Momentan kann im SiMA außerdem pro Konflikt nur ein Abwehrmechanismus aktiv sein. In Zukunft

wird das SiMA-Modell auch multible Abwehrmechanismen kombinieren können, um ein Konflikt zu lösen. Die jetzige Visualisierung ist dafür aber nicht ausgelegt und müsste grundlegend überarbeitet werden, um dies zu erlauben.

Das Inventar ist ein relativ abgeschlossener Bereich, weshalb Erweiterungen nur in Form einer weiteren Angleichungen an das physische Vorbild vorgenommen werden könnten, etwa durch die Berücksichtigung der Objektgröße oder der Handlichkeit eines Objektes.

Abschließend kann jedoch festgehalten werden, dass das SiMA-Modell bereits in seiner jetzigen Konfiguration geeignet ist, auch in Zukunft durch die weitere Entwicklung eine immer realitätsgetreuere Simulation des menschlichen Verhaltens auf Basis psychologischer Wertungen zu bieten.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[ABB+04] Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. und Qin, Y. – An integrated theory of the mind. Psychological review, 111(4):1036-1060, 2004.

[BACR08] Burmeister, B., Arnold, M., Copaciu, F., und Rimassa, G. - BDI-agents for agile goal-oriented business processes. International Foundation for Autonomous Agents and Multiagent Systems (pp. 37-44), Mai 2008.

[BDD+15] Brandstätter, C., Dietrich, D., Doblhammer, K., Fittner, M., Fodor, G., Gelbard, F., Huber, M., Jakubec, M., Kollmann, S., Kowark, D., Schaat, S., Wendt, A. und Widholm R. - Natural Scientific, Psychoanalytical Model of the Psyche. Technische Universität Wien, 2015.

[BF09] Baars, B. J. und Franklin, S. - Consciousness is computational: The LIDA model of global workspace theory. International Journal of Machine Consciousness, (pp. 23-32), 2009.

[BGSW13] Bruckner, D., Gelbard, F., Schaat, S. und Wendt, A. - Validation of cognitive architectures by use cases: Examplified with the psychoanalytically-inspired ARS model implementation. In Industrial Electronics (ISIE), 2013 IEEE International Symposium on (pp. 1-6). IEEE, Mai 2013.

[Bull02] Buller, A. - Volitron: On a psychodynamic robot and its four realities. Kyoto 619-0288 Japan,2002.

[DBD+14] Dietrich, D., Brandstätter, C., Doblhammer, K., Fittner, M., Fodor, G., Gelbard, F., Huber, M., Jakubec, M., Kollmann, S., Kowarik, D., Schaat, S., Wendt, A. und Widholm, R. - Naturwissenschaftliches, psychoanalytisches Modell der Psyche für Simulation und Emulation. wissenschaftlicher Bericht, Institute of Computer Technology, Technische Universität Wien, 2014.

[DBZ+09] Dietrich, D., Bruckner, D., Zucker, G., Muller, B. und Tmej, A. - Psychoanalytical model for automation and robotics. In AFRICON, 2009. AFRICON'09. (pp. 1-8). IEEE. September 2009.

[Deut11] Deutsch, T. - Human Bionically Inspired Autonomous Agents. Phd thesis, Technische Universität Wien, 2011.

[DFZB09] Dietrich, D., Fodor, G., Zucker, G., und Bruckner, D. - Simulating the mind. Wien: Springer, 2009.

[DTM+09] Deutsch, T., Tmej, A., Muchitsch, C., Zucker, G., Riediger, C. und Lang, R. - Failsafe aspects of a decision unit inspired by cognitive sciences-the id without ego and super-ego. In Human System Interactions, 2009. HSI'09. 2nd Conference on (pp. 376-382). IEEE, Mai 2009.

[DZL07] Deutsch, T., Zeilinger, H. und Lang, R. - Simulation results for the ars-pa model. In Industrial Informatics, 2007 5th IEEE International Conference on (Vol. 2, pp. 995-1000). IEEE, Juni 2007.

[FMDS14] Franklin, S., Madl, T., D'Mello, S. und Snaider, J. - LIDA: A systems-level architecture for cognition, emotion, and learning. IEEE Transactions on Autonomous Mental Development, 6(1):19-41, 2014.

[FP06] Franklin, S. und Patterson Jr, F. G. - The LIDA architecture: Adding new modes of learning to an intelligent, autonomous, software agent. 703:764-1004, 2006.

[GBD+11] Gelbard, F., Brandstätter, C., Doblhammer, K., Hinterleitner, I., Kohlhauser, S., Kovacs, Z. und Zeilinger, H. - Comparison of technical filter mechanisms and defense mechanisms of the human mind. In AFRICON, 2011 (pp. 1-6). IEEE, September 2011.

[Gelb15] Gelbard, F - Psychoanalytic defense mechanisms in artificial intelligence: exemplified implementation in complex technical systems. Phd thesis, Technische Universität Wien, 2015.

[GOS14] Goertzel, B., Orseau, L. und Snaider, J. -
Artificial General Intelligence: 7th International Conference, AGI 2014. Quebec City, Canada, August 2014.

[Grah05] Graham, P. - The roots of lisp. Retrieved October, 5, 2005.

[Herr14] Herret, L. - Schnittstellendesign einer kognitiven Architektur.
Technische Universität Wien, 2014.

[Lair08] Laird, J. E. - Extending the Soar cognitive architecture. Frontiers in Artificial Intelligence and Applications, 2008.

[Lang10] Lang, R. - A Decision Unit for Autonomous Agents Based on the Theory of Psychoanalysis. PhD thesis, Technische Universität Wien, 2010.

[LC06] Langley, P. und Choi, D. - A unified cognitive architecture for physical agents. Stanford University, 2006.

[LNR87] Laird, J. E., Newell, A. und Rosenbloom, P. S. - Soar: An architecture for general intelligence. Artificial intelligence, 33(1):1-64, 1987.

[Lotfi14] Lotfi, Y., Implementation of psychoanalytic defense mechanisms in artifical intelligence. Diplomarbeit Technische Universität Wien, 2014.

[Luke11] Luke, S. - Multiagent simulation and the MASON library. George Mason University, 2011.

[McCa60] McCarthy, J. - Recursive functions of symbolic expressions and their computation by machine, Part I. Communications of the ACM, 3(4), 184-195, 1960.

[Much13] Muchitsch, C. - Human-like Perception for Psychoanalytically Inspired Reasoning Unit. PhD thesis, Technische Universität Wien, 2013.

[PG07] Pennachin, C. und Goertzel, B. - Contemporary approaches to artificial general intelligence. Springer Berlin Heidelberg, pp. 1-30, 2007.

[PP05] Pratl, G. und Palensky, P. - Project ARS-the next step towards an intelligent environment (pp. 55-62), 2005.

[PPDB05] Pratl, G., Penzhorn, W. T., Dietrich, D. und Burgstaller, W. - Perceptive awareness in building automation. In Computational Cybernetics, 2005. ICCC 2005. IEEE 3rd International Conference on (pp. 259-264). IEEE, April 2005.

[RG95] Rao, A. S. und Georgeff, M. P. - BDI agents: From theory to practice. ICMAS (Vol. 95, pp. 312-319), Juni 1995.

[Sarf03] Sarferaz, S. - Methoden- und Werkzeugunterstützung für evolutionäre, objektorientierte Software-Projekte. Phd thesis, Universitätsbibliothek Marburg, 2003.

[Scha12] Schaat, S. - Integrated Drive Object Categorization in Cognitive Agents. Diplomarbeit Technische Universität Wien, 2012.

[Scha16] Schaat, S. - Simulation of Foundational Human Information-Processing in Social Context. PhD thesis, Technische Universität Wien, 2016.

[SMW+15] Schaat, S., Miladinović, A., Wilker, S., Kollmann, S., Dickert, S., Geveze, E., und Gruber, V. - Emotion in consumer simulations for the development and testing of recommendations for marketing strategies. In Proceedings of the 3rd Workshop on Emotions and Personality in Personalized Systems 2015 (pp. 25-32). ACM, September 2015.

[SSK97] Schuster, P. und Springer-Kremser, M. - Bausteine der Psychoanalyse: eine Einführung in die Tiefenpsychologie (Vol. 3). WUV-Universitätsverlag, 1997.

[SWK+15] Schaat, S., Wendt, A., Kollmann, S., Gelbard, F., und Jakubec, M. - Interdisciplinary Development and Evaluation of Cognitive Architectures Exemplified with the SiMA Approach. In EAPCogSci, 2015.

[Wend16] Wendt, A. - Experience-Based Decision-Making in a Cognitive Architecture. Phd thesis, Technische Universität Wien, 2016.

[WG06] Wang, P. und Goertzel, B. - Introduction: Aspects of artificial general intelligence. IOS Press (pp. 1-16), 2006 .

[Wool96] Wooldridge, M. - Practical reasoning with procedural knowledge. Practical Reasoning, 663-678, 1996.

[Zeil10] Zeilinger, H. - Bionically Inspired Information Representation for Embodied Software Agents. PhD thesis, Technische Universität Wien, 2010.

[ZWSS16] Zucker, G., Wendt, A., Siafara, L., und Schaat, S. - A cognitive architecture for building automation. In Industrial Electronics Society, IECON 2016-42nd Annual Conference of the IEEE (pp. 6919-6924). IEEE, Oktober 2016.

Internet Referenzen

[1] JFreeChart. Homepage, 31.12.2015.
http://www.jfree.org/jfreechart/

[2] JGraph. Homepage, 31.12.2015.
https://www.jgraph.com/about-jgraph-company.html

[3] MASON Framework. Homepage, 27.04.2015.
cs.gmu.edu/~eclab/projects/mason/

[4] SiMA Dissertationen. Homepage, 14.05.2015.
http://sima.ict.tuwien.ac.at/publications/?cat=Dissertation#headline

[5] SiMA Papers. Homepage, 14.05.2015.
http://sima.ict.tuwien.ac.at/publications/

[6] SiMA - Simulation of the Mental Apparatus & Applications. Homepage, 20.04.2017.
http://sima.ict.tuwien.ac.at/

[7] SiMA Steuerungsfenster. Homepage, 23.01.2017.
http://sima.ict.tuwien.ac.at/wiki/index.php/Inspektoren

[8] SPARQL Query. Homepage, 10.05.2015.
https://www.w3.org/TR/rdf-sparql-query/

[9] Swing. Homepage, 31.12.2015.
http://www.java-tutorial.org/swing.html

[10] XML-File. Homepage, 10.05.2015.
http://www.xmlfiles.com/

Erklärung zur Verfassung der Arbeit

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 21. Aug. 2017

Ivalina Jordakieva, BSc