FFI-rapport 2008/01565

Representation of Human Behavior in Military Simulations

Vidar Engmo

Forsvarets forskningsinstitutt/Norwegian Defence Research Establishment (FFI)

25.08.2008

FFI-rapport 2008/01565

419801

P: ISBN 978-82-464-1415-7
E: ISBN 978-82-464-1416-4

Keywords

Modellering og simulering

Menneskelig atferd

Agenter (Databehandling)

Kognitiv teknologi

Panservogner

English summary

This report contains the master thesis of Vidar Engmo, which investigates human behavior representation in military simulations. The thesis reviews several agent and cognitive frameworks, and evaluates these for use in military simulations. In addition, the thesis describes a case study. The case study consists of integrating a Beliefs-Desires-Intention (BDI) agent framework with a simulation framework. The case study focused on simulating synthetic teammates in a main battle tank troop.

Sammendrag

Rapporten inneholder masteroppgaven til Vidar Engmo, som omhandler representasjon av menneskelig oppførsel i militære simuleringer. Masteroppgaven beskriver flere agent- og kognitive rammeverk, og evaluerer disses brukbarhet i militære simuleringer. Oppgaven beskriver også en tilfelle-studie hvor et Beliefs-Desires-Intention (BDI) agentrammeverk ble integrert med et simuleringsrammeverk. Studien fokuserte på å simulere syntetiske lagkamerater i en stridsvogntropp.

Contents

Preface

The following page contains the assignment text. The subsequent pages contain the master thesis in full.

Problem Description

A challenge in military simulations is to include human factors and behavior. One way of avoiding this problem is to use real time human-in-the-loop simulations, however when simulating larger operations this requires too much resources in terms of time and personnel.

Lately some frameworks are made for computer generated forces or CGF. These frameworks covers modeling of physical systems in a good way, but human behavior can be a problem. Within the military also other frameworks are created for modeling and simulating human behavior, some of these frameworks are built on cognitive theories. Also within civil industry (such as games) there are interests in these frameworks for representing human behavior.

The candidate shall describe existing frameworks for human behavior, and build a demonstrator in a chosen framework. The candidate must also do a literature survey of theories for human behavior in military operations, and describe their features and limitations.

Assignment given: 04. February 2008
Supervisor: Lill Kristiansen, ITEM

NTNU
Norwegian University of
Science and Technology

Representation of Human Behavior in Military Simulations

Vidar Engmo

Master of Science in Communication Technology
Submission date: July 2008
Supervisor: Lill Kristiansen, ITEM
Co-supervisor: Karsten Bråthen, Forsvarets forskningsinstitutt

Norwegian University of Science and Technology
Department of Telematics

Preface

> *"Why waste time on a summer vacation,*
> *when you can spend the time on writing*
> *your master thesis!"*
>
> *The thought crawled in my head around*
> *midnight a Saturday evening in July, when I*
> *was still hammering away on my keyboard.*
> *What can I say? I just like it, Vidar Engmo.*

I would like to thank my father and mother for giving me the courage to always seek new challenges and believe in myself.

I would like to thank the officers at the Norwegian Army Transformation and Doctrine Command for all the help regarding the development of the demonstration software. My visit to Rena Camp and the demonstration of the Combined Combat Simulator gave me valuable inputs on how to create simple but effective behavior in Human-in-the-Loop simulations.

I would like to thank my professor at the Norwegian University of Science and Technology (NTNU), Prof. Lill Kristiansen for the guidance and valuable feedback during the period of writing this master thesis.

Finally I would like to thank my teaching supervisors, researchers Karsten Braathen and Odd-Martin Staal, at the Norwegian Defense Research Establishment for always looking out for my best interests and providing me with guidance, software and advice on how to solve the task of writing the master thesis.

Vidar Engmo

Vidar Engmo

Wednesday, July 30, 2008

Abstract

The purpose of this master thesis was to investigate the psychological and computational basis for human behavior representation (HBR) in military simulations and identify problem areas of existent software agent frameworks that provide computer generated forces (CGF) with human like cognitive abilities.

The master thesis identifies psychological properties that influence human cognition in an operational environment through a theoretical study of operational and cognitive psychology. The psychological properties of human cognition are then connected to artificial intelligence through a theoretical study of agents and multi-agent systems and form the foundation for identifying general HBR properties. The HBR properties are used as evaluation markers that constitute the basis for constructing an evaluation of relevant agent frameworks thereby visualizing their strengths and weaknesses.

The problem areas of incorporating artificial intelligence into CGF are further concretized by the development of a demonstrator that interacts with a synthetic environment. The demonstrator is an implementation of a tank platoon in the agent framework Jadex. The synthetic environment is provided by VR-Forces which is a product by MÄK technologies.

The thesis makes a distinction between the conceptual structure of agent frameworks and their actual implementation. According to this master thesis it is the output of the agent framework that is the most important feature not how the output came into being. Producing the correct output requires the selection of the correct tools for the job. The selection of an agent framework should be taken on the background of an evaluation of the simulation requirements.

A large portion of the development time is consumed by the development of application and communication interfaces. The problem is a result of lacking standardization and that most cognitive agent frameworks are experimental in nature. In addition the artificial intelligence (AI) in such simulations is often dived into levels, where the synthetic environment takes care of low-level AI and the agent framework the high-level AI. Tight synchronization between low and high-level AI is important if one wishes to create sensible behavior. The purpose of an agent framework in conjunction with CGF is thereby ensuring rapid development and testing of behavior models.

Table of Contents

List of Figures

List of Tables

Abbreviations

ACL	Agent Communication Languages
AI	Artificial Intelligence
ANN	Artificial Neural Networks
AOI	Area of Interest
AOR	Area of Responsibility
BDI	Belief, Desire and Intention
C2	Command & Control
C2IS	Command & Control Information Systems
CCS	Combined Combat Simulator
CGF	Computer Generated Forces
CNS	Central Nervous System
CO	Commanding Officer
COA	Course of Action
COP	Common Operation Picture
CSP	Constraint Satisfaction Problem
DIS	Distributed Interactive Simulation
DoD	US Department of Defense
EBO	Effect Based Operations
EDF	Earliest Deadline First
EU	Expected Utility
FFI	Forsvarets Forskningsinstitutt / Norwegian Defense Research Establishment
FFOD	Forsvaret Felles Operative Doktrine / Norwegian Forces Joint Operational Doctrine
FIPA	Foundation for Intelligent Physical Agents
FOL	First Order Logic
FOM	Federation Object Model
FPS	Fixed Priority Scheduling
FSM	Finte State Machines
GIS	Graphical Information System
GA	Genetic Algorithm
GP	Genetic Programming
HBM	Human Behavior Modeling
HBR	Human Behavior Representation
HD	Hard Drive
HITL	Human-in-the-Loop
HLA	High Level Arhitecture
HMI	Human Machine Interface
ICT	Information and Communication Technology
IPIP NEO	International Personality Item Pool NEO
IQ	Intelligence Quotient
JADE	Java Agent Development Environment
JCC	Jadex Control Center

KB	Knowledge Base
KIF	Knowledge Interchange Format
KQML	Knowledge Query and Manipulation Language
LOP	Levels-of-Processing
LTM	Long-Term Memory
MEU	Maximum Expected Utility
MO	Military Organization
NATO	North-Atlantic Treaty Organization
NBD	Network Based Defense
NDM	Naturalistic Decision-Making
NoAF	Norwegian Armed Forces
NTNU	Norges Teknisk-Naturvitenskalige Universitet
OODA	Observe, Orient, Decide, Act
PEAS	Performance, Environment, Actuator, Sensor
PNS	Peripheral Nervous System
PMF	Performance Moderator Function
RPD	Recognition Primed Decision-Making
PTSD	Post Traumatic Stress Disorder
RAM	Random Access Memory
ROE	Rules of Engagement
RTI	Run-Time Infrastructure
SA	Situational Awareness
SE	Synthetic Environment
SEU	Subjective EU
SOP	Standing Operating Procedure
SOR	Sector of Responsibility
STM	Short-Term Memory
STRIPS	Stanford Research Institute Problem Solver
TCL	Tool Command Language
TOD	Task Oriented Domain
TRADOC	The Norwegian Army's Transformation and Doctrine Command
UN	United Nations
UT	Unreal Tournament
UTC	Unified Theory of Cognition
VBS	Virtual Battle Space
WM	Working Memory
XO	Executive Officer
XML	Extensible Markup Language

Chapter 1 Introduction

1.1 Background

The ways in which things are done all over the world is changing due to advances in Information and Communication Technology (ICT). The change will affect all aspects of society also the national security policies and military strategies. The ability to collect and process information brings along a change in the methods of how to conduct warfare and as a result military doctrines are rewritten. The change in military doctrine and organization has a deep impact on how military operations are conducted and increases the need for competent personnel. The demand for well trained and educated officers and soldiers is getting higher due to the increased complexity of both the equipment and the battle space (1).

The Norwegian Armed Forces (NoAF) ongoing transformation process is an acknowledgement of the changing operational situation. Time spent on research that might increase the flexibility, responsiveness and cost efficiency of the military organization is therefore considered to be of great value. One of the activities that the Norwegian Defense Research Establishment emphasizes in its latest research plan is modeling and simulation of military capabilities in synthetic environments (2).

A wide spectrum of concepts for military simulations has been developed, but the amount of personnel and time needed to perform large scale simulations of military operations is still too large. By introducing intelligent software modules that impersonate human behavior into the simulation environment, the need for human involvement decreases. *The question is how to represent human behavior?*

1.2 Problem Description

The Norwegian Defense Research Establishment, hereafter referred to as FFI, is investigating computational models for human behavior with the intent to develop or integrate existing intelligent software modules that act and think similarly to human beings.

There are several commercial frameworks for simulating large scale battle scenarios. Simulation frameworks made for CGF, often cover most aspects of modeling physical systems including terrain, structures and vehicles. The problem becomes evident when a human contestant is interacting with a synthetic entity in the simulation. The built in AI is, if existent, too limited to give a believable representation or model of human behavior. The result is irrational and predictable entities, which by most definitions does not give the desired combat experience for humans participating in the virtual exercise.

Introducing and incorporating frameworks for human behavior representation (HBR) or human behavior modeling (HBM) into large scale battle space simulations presents an opportunity to create realistic simulation environments. FFI therefore needs an overview and analysis of relevant HBR frameworks and a general description of their features.

Real insights into problem areas around the concept of distributed simulations do not become evident before one try to implement a demonstrator. Based on a case study, where the case is provided by FFI, a conceptual demonstrator for human behavior will be developed. FFI will place at disposal frameworks and simulators for development of a demonstrator which uses one or several HBR techniques identified by the master thesis.

1.3 Scope

The nature of military simulations is diverse due to the fact that the environment and the intelligent entities within it constitute a complex simulated world. There are many environmental variables that need to be considered in the reasoning process of a human being. Some examples are the nature of the mission, the operational situation, the topology of the battlefield, time of day, time of year and amount of resources available. The problem is how one can create a model that captures the essence of human behavior in a complex operational environment (3 p. 16).

The creation of an artificial human being needs input from several areas of science. Either one is going to create a physical embodiment or a virtual animated embodiment of a human, one have to consider the biological features to be implemented. An example could be the movement of the joints and muscular contractions both facial and corporal. Recreation of human behavior requires insight into both psychology and culture (Figure 1). Psychology and culture is closely connected, because culture affects the ways in which we reason and think. Culture can be said to be the framework in which we rely on to make sense of the world. Psychology aims to amongst others, describe the individual's cognitive and social processes within the dynamic boundaries of the cultural framework (4).

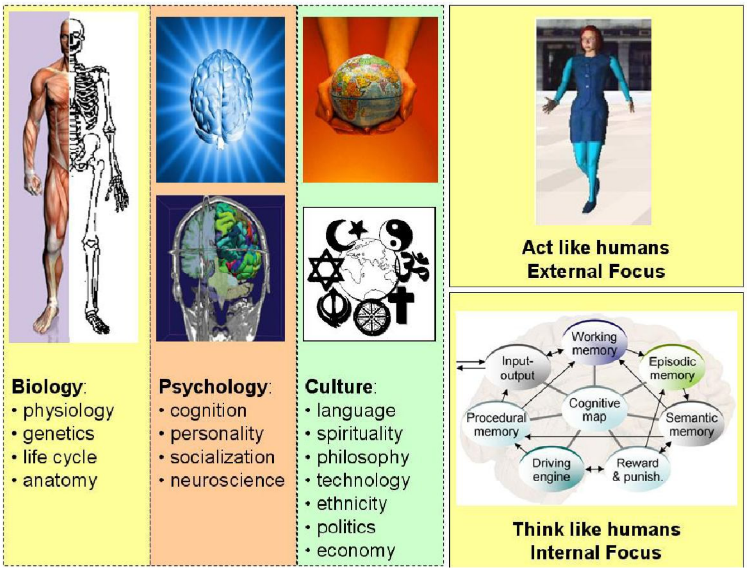


Figure 1 Human Representation

There are many ways of engineering software modules that constitutes a close resemblance to a human being. The question is how one should build a module that is both believable and realistic. A module that simulates human behavior can be constructed on the basis of two different approaches:

act like humans or think like humans (Figure 1). The first approach is built on the module's ability to impersonate a human being, while the latter approach is built on theories of human cognition (5).

Military simulations tend to involve both approaches. The reason for this is that the simulation of human behavior in a military setting constitutes the need of a simulated environment, a simulation of the body and a simulation of the mind. The cognitive approach emphasizes the simulation of the mind, while military simulations extents to not only recreate human thought processes, but also the observable interaction with the environment (Figure 2). The main focus of this master thesis is the representation of the human mind.

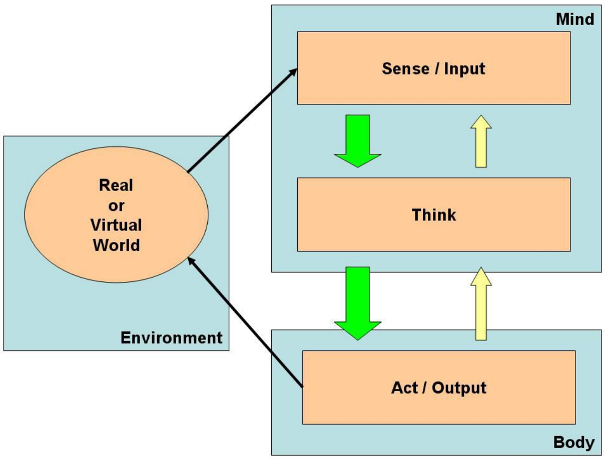


Figure 2 Sense-think-act Cycle, from (6)

People interact with the environment in several ways, for instance the ability to move and affect physical objects and the ability to express oneself by the means of natural and body language. Interaction between a virtual human and a real human is considered to be a complex problem to solve, because it involves processing of natural language both speech recognition and conversational dialogue. This area of research together with the area of non-verbal sensing and body language is considered to be immature (6). These research areas are not within the scope of this master thesis.

Figure 3 gives an overview over areas of AI. The parts on a red background are considered to be outside the scope of the master thesis while the part on a green background is considered to be a part of the master thesis. The master thesis does not elaborate on issues concerning interactions in the real world. The focus is on the simulated world or environment. Human-in-the-loop (HITL) simulations demands a human machine interface (HMI), which makes it possible for a person to interact with entities in the simulated world. The master thesis takes a closer look at problems that arises when humans interact with artificial intelligent entities in a synthetic environment.

The abstract objects of HBR receive sensory information both through the HMI and from the simulated world. The representation of the simulated world and the entity's body manifestation is not a part of the master thesis. The intelligent module is connected to the body via an interface. The body or virtual human then functions as the basis for sensory input and visualize the output or behavior. The master thesis takes a closer look at how behavior is created both from a psychological and computational view point.

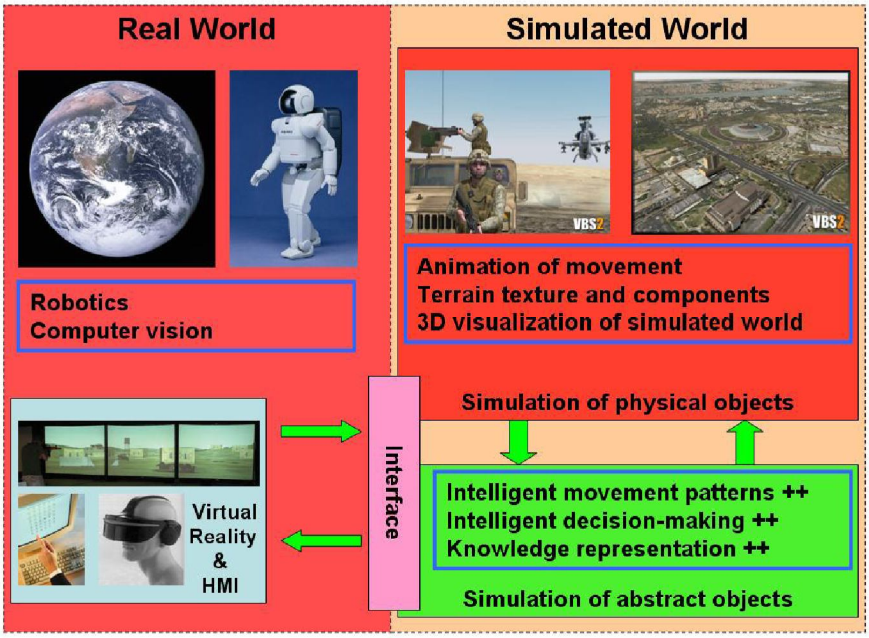

Real World
Simulated World
VBS2
VBS2
Robotics
Computer vision
Animation of movement
Terrain texture and components
3D visualization of simulated world
Simulation of physical objects
Interface
Virtual
Reality
&
HMI
Intelligent movement patterns ++
Intelligent decision-making ++
Knowledge representation ++
Simulation of abstract objects

Figure 3 Thesis Focus Area

The master thesis focuses on internal HBR issues, meaning the internal processes of processing information. This information is the basis for all elaborations that the intelligent entity makes. The information needs to be processed and turned into a specified representation of knowledge that can be stored and retrieved when needed. Actions performed in the simulated world are the result of a reasoning process, where the entity has retrieved the necessary information and concluded with proper behavior.

1.4 Structure

The background setting for the master thesis is situations where military forces are involved. This means that some of the theoretical concepts of a military organization must be known. Military concepts and organization is presented in Chapter 2: The Military Domain.

The master thesis attach importance to creating a theoretical foundation for understanding what is meant by HBR and the problems involved when trying to incorporate such elements into a synthetic environment. The theoretical foundation for human behavior is to be found in the world of psychology. The computational counterpart AI similarly uses the notion of a software agent to produce behavior in computer systems. Background theory on operational psychology and multi-agent systems are to be found in Chapter 3: Background Theory.

The actual presentation of HBR is to be found in Chapter 4: Human Behavior Representation. The HBR chapter gives a definition of HBR and an overview of research areas that fall under this category. The HBR chapter assembles concepts and theoretical models of human performance factors and cognitive architectures.

HBR structures deliberate on information perceived in the simulated world and impose changes to the world state either indirectly by affecting the reasoning process of the agent or directly by performing actions. Chapter 5: Simulation Frameworks presents virtual environments and gives an introduction to distributed simulation. In addition it presents frameworks for creating agents and multi-agent systems, with emphasize on frameworks that have their foundation in cognitive theories.

The demonstrator is based on a case described in Chapter 6: Case Studies. The chapter presents 3 current study areas which FFI find relevant for further investigative research. On the basis of one case study a demonstrator has been developed. The demonstrator software and concept is presented in Chapter 7: Development of Demonstrator.

The results and conclusions of the discussed topics of HBR, simulation frameworks and the demonstrator are presented in Chapter 8: Discussion and and Chapter 9 Conclusion, Recommendations and Further Work.

Chapter 2 The Military Domain

The purpose of this chapter is to create a theoretical foundation for understanding the military organization and military operations. The chapter defines terms and abbreviations to be used in the master thesis. The overall policy or guidance in how to conduct operations and exercise command and control is usually written in a military doctrine. The Norwegian Armed Forces Joint Operational Doctrine (FFOD) describes amongst others the concepts to be used in military operations home and abroad (7 p. 7).

2.1 Military Levels of Command

Operations are controlled through a hierarchy of several levels. Usually one distinguishes between the strategic, operational and tactical levels. These are known as the military levels of command. There is also a strategic level that is non-military. This level is the grand strategic level, which is the political level. At this level a nation decides amongst others whether to engage in a conflict or not. The forming of alliances like for instance NATO is a part of the strategic thinking of a nation (Figure 4).

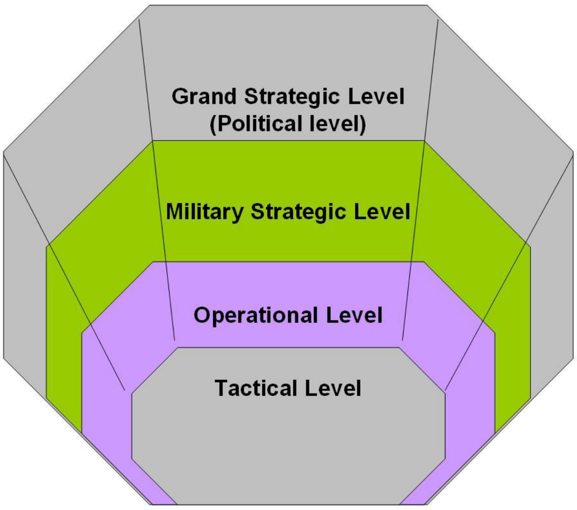


Figure 4 Levels of Command, from (7 p. 31)

At the military strategic level decisions are made on how armed forces will be used to achieve the aims of national policy. The operational level is responsible for shaping the political and military strategic goals into feasible plans and operations. The level actively link military strategies to tactics. The tactical level on the other hand resolves the concrete, tactical tasks in conjunction with the operational goals. At the tactical level forces are deployed to confront the enemy in the battle space (7 p. 32).

2.2 Structural Composition of a Military Force

The military is a hierarchical organization. The purpose of this composition is to be able to compose structures of forces according to operational needs. Each component possesses specialized capabilities that can be moved from one structure to another. The components are tools for composing an organization that can respond to highly dynamic situations.

A military force is an aggregation of several combat groups, which have a distinct expertise in one or more fields. On the lowest level there are the sections or the squads. The group is lead by the squad leader, which in most cases would be a sergeant. Several squads are grouped together in a troop or platoon, which often is lead by a lieutenant. The number of squads in a platoon depends on the mission and the equipment, for instance a mechanized infantry platoon would be equipped with armored vehicles. Several platoons would be organized into a company, squadron or battery. The name is dependent on the specialty of the group. A battery usually is some kind of artillery while a squadron could be groups of aircrafts. The levels continue up to army group, but in Norway one usually does not go further up than division. Table 1 gives an overview over unit names, the number of personnel involved in each unit, the APP-6A unit symbol and the rank of the commanding officer (8).

Name English	Name Norwegian	Force	Symbol	Military Rank English	Military Rank Norwegian
Squad	Lag	8 – 16	•	Corporal/Sergeant	Korporal/Sersjant
Platoon	Tropp	25 – 60	•••	1st or 2nd Lieutenant	Fenrik/Løytnant
Company	Kompani	70 – 250	|	Captain/Major	Kaptein/Major
Battalion	Bataljon	300 – 1000	| |	Lieutenant colonel	Oberst Løytnant
Regiment	Regiment	2000 – 3000	| | |	Colonel	Oberst
Brigade	Brigade	3000 – 5000	X	Brigadier	Brigader
Division	Divisjon	10000 – 20000	XX	Major General	General Major

Table 1 Military Organizational Structure

NATO countries usually comply with the NATO standard APP-6A, which is a standard for chart symbols (9). The chart symbols give information about own and observed units. The units could either be identified as friendly, neutral or hostile. If the unit has been observed but not identified it is labeled as unknown. APP-6A specifies different symbols for unit types and contains information about size and identification. In a war theatre units are given an area of responsibility (AOR). The responsibility is delegated all the way down to the squads. Symbols for different types of areas also exist in APP-6A and is denoted tactical graphics (9).

NoAF has several different tools for planning and producing a picture of the current operational situation. The most know tool is MARIA, which is a geographic information system (GIS) that uses APP-6A symbols. Figure 5 shows a situation displayed by MARIA, where the units are pictured using APP-6A symbols. Each unit is identified by the specific symbol, the tag numbers and the color coding. Blue forces are categorized as friendly forces. The battle space is divided into different areas of interest (AOI). Some areas are categorized as AORs others are areas of strategic importance or distributions points for combat support missions.

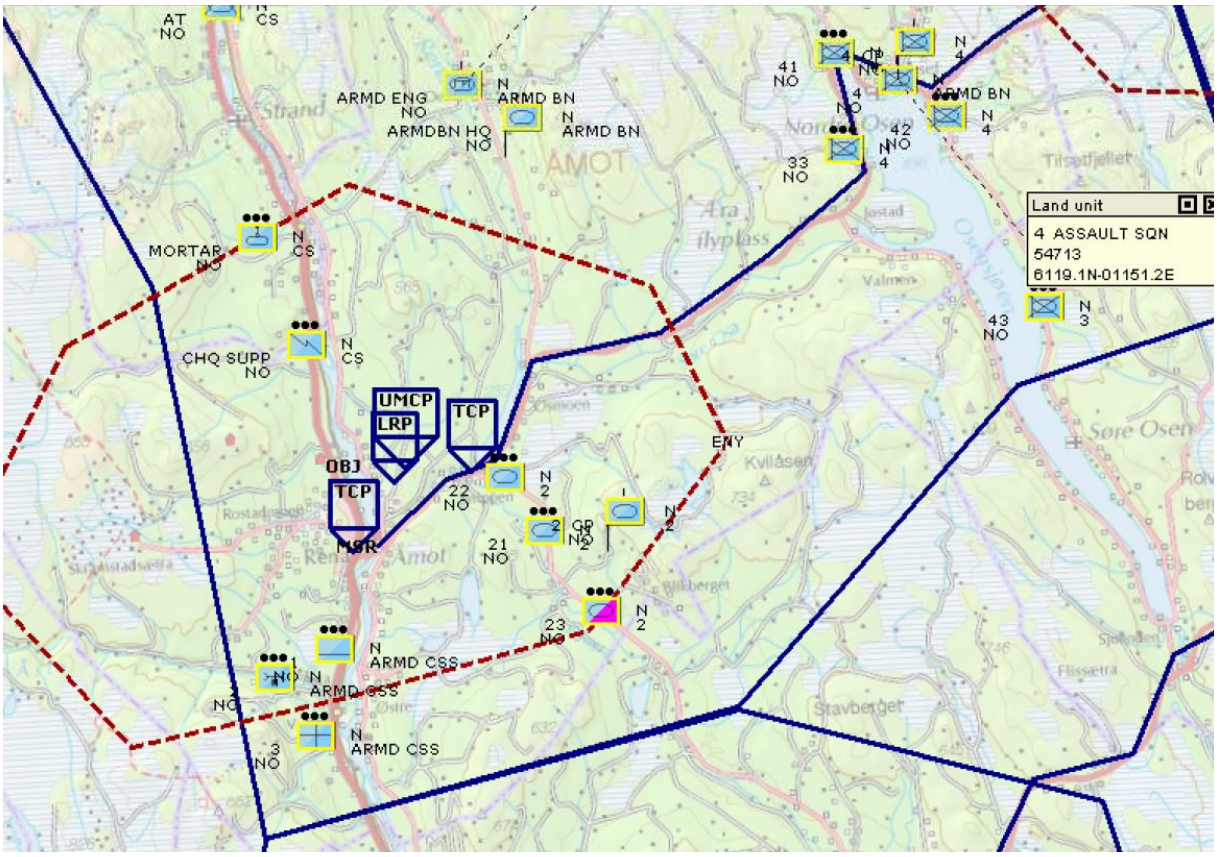


Figure 5 Use of Military Symbols, from (10)

2.3 Military Operations

The use of excessive force has been the domain of the military around the world for centuries and the use of violence has been the last resort of action for politicians if diplomacy failed. Through the charters of the United Nations (UN) one has agreed on some ground rules of conducting warfare and this has lead to definitions that define different types of conflicts based on the conflict level (7 p. 33). NoAF uses these definitions to describe different types of operations.

An operation in the way the military understands it, is a series of planned events aiming at reaching a specified goal. There are many kinds of operations that are in the domain of a military force and NoAF concentrates on supporting national and international operations. Participation in international operations is the result of the international obligations Norway has committed itself to. Operations abroad can be divided into categories dependent on the intensity of the conflict (Figure 6).

- **Stabilization operations** - Low intensity conflicts demands a force that is capable of keeping the conflicting parties in check and would therefore be a stabilizing factor in the conflict area. Examples are peace support operations (7 p. 25).
- **Attrition warfare** - Low to average high intensity conflicts often are recognized by combat against irregular forces. An irregular force is a group of armed individuals that act on behalf of themselves, meaning there is no body of state to govern them (7 p. 41). Examples are local militia and drug lords. The operational strategy is to use exhaustive techniques to

continuously weaken the capabilities of the opponent which in this case is likely to run out of resources.

- **Maneuver warfare** – Average high to high intensity conflicts are recognized by combat with regular forces. This means that one is combating a force that has been formed by a body of state (7 p. 41). When seeking to end a conflict with regular forces one actively uses defensive and offensive methods for influencing the opponent's will and determination to keep fighting. The purpose is to actively engage the opponent in operations with the intent of ensuring that one's own forces keep the initiative.

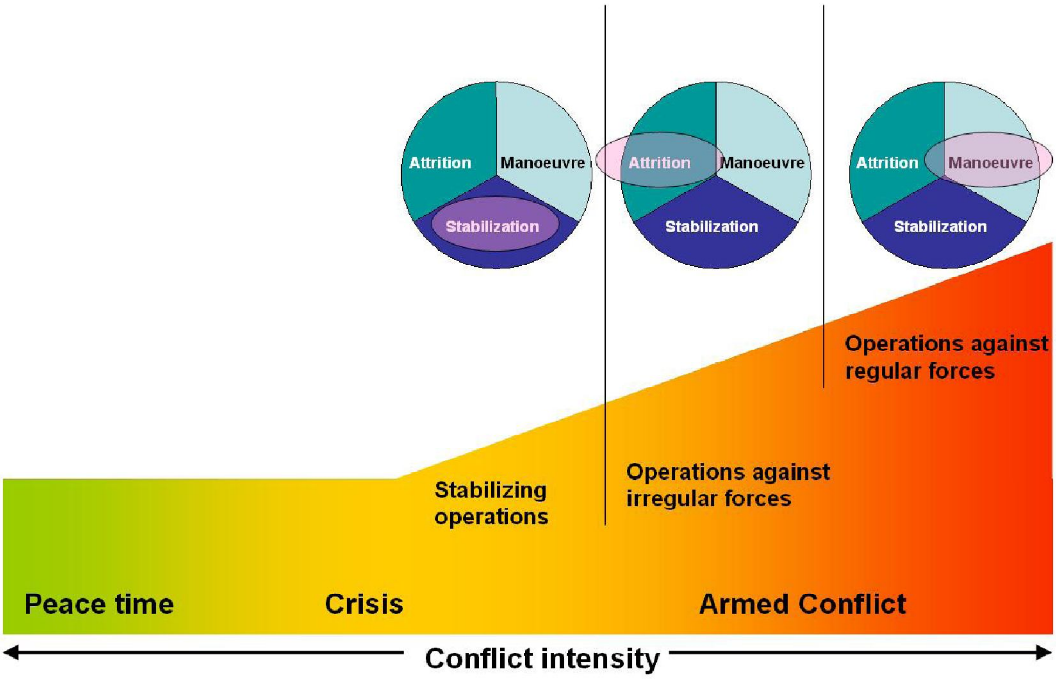


Figure 6 Operations and Conflict Intensity, from (7 p. 24)

The battle space is a multidimensional space, which can be influences by different kinds of operations. A conflict or battle is not only a fight between two physical forces it is also a battle between wills and ideas. This notion of a conflict or battle introduces several domains in which one can influence the opponent's will and determination to keep fighting: the cognitive domain, the informational domain, the social domain and the physical domain. Combined efforts seek to induce effects on all domains at the same time. Military operations are therefore a collection of activities in several domains that seek to end the conflict (7 p. 70).

2.4 Command and Control

A military force is organized into a hierarchical structure to ensure control and ease of command. Command can be defined as the formal authority an operational leader has been assigned in the interest of being able to administrate, coordinate and control available military resources. Command is the matter of assignment while control is the access to actively execute orders and directives. Command and control (C2) is the means in which the operational leader relies on to make effective and correct decisions and implement orders. C2 is dependent on several basic functions: battle resources, intelligence information, logistics, mobility and protection of force (7 p. 73).

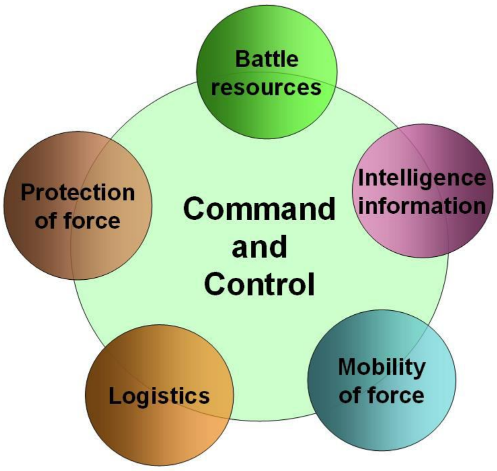


Figure 7 C2 Basic Functions, from (7 p. 73)

Several systems and models have been developed, denoted command and control information systems (C2IS), the area is still undergoing extensive research. The most known decision model used by the Norwegian military is the OODA-loop by John Boyd (Figure 8). The OODA-loop is a description of a four step decision cycle and the factors that influence the decisions made in an operational environment (7 p. 80). The quality of the decision is dependent on the assessment of the situation, which have lead to the notion of situational awareness.

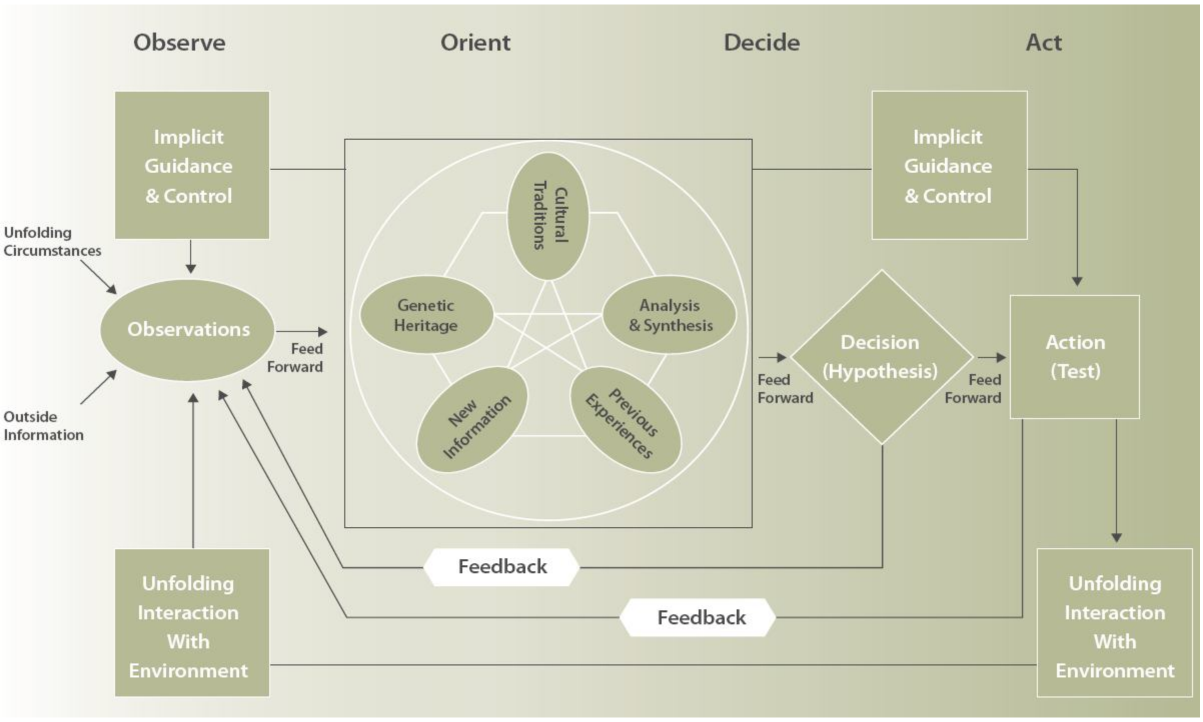


Figure 8 John Boyd's OODA-Loop, from (7 p. 80)

2.5 Network Based Defense

The concept of Network Based Defense (NBD) is to exploit the beneficial effects of sharing huge amounts of information. The concept revolves around advances in ICT the last twenty years and the purpose is to interconnect military units and enable networked collaboration with the intent of increasing operational efficiency. A robust network consisting of military sensors and units share information about the battle space. Typical information shared is the location of own and enemy troops and tactical analyses of the situation. Sharing of information enhances the Common Operational Picture (COP) and leads to common situational awareness. When military units share the same understanding of the situation communication between the same units is more effective, therefore increased common situational awareness improve the ability to collaborate (7 p. 91). An example of COP is the map in Figure 5, where one has created an operational picture of the current operational situation.

Situational awareness (SA) is a key element in the concept of NBD. The most used model of SA was defined by M. R. Endsley who described SA as consisting of three levels or aspects of a situation (11). First one has to identify that there is a change in situation and that something is happening. This is known as situation consciousness (L1-SA). Secondly one needs to understand which consequences the change in situation has for the military units in question. This is known as situation comprehension (L2-SA). Last but not least one should be able to make predictions about how the situation will evolve. This is known as situation prediction (L3-SA) (7 p. 95).

Chapter 3 Background Theory

3.1 Introduction

The purpose of this chapter is to shed light on the psychological background for grounding and sensemaking in a military context, in addition to connecting psychological theories to computational theories through software agents.

HBR is a term created to describe modeling of behavior in military simulations (3 p. 11). The success of a military operation is dependent on that many different levels of command and tactical units are able to coordinate their strategic and tactical dispositions in order to maximize the effect of operations. In order to do so the military aims at looking at the battle space from different angles or domains (chapter 2.3). Through communication one seeks to create a common understanding of the battle space and achieve a state of common situational awareness (chapter 2.5).

Whether the exchange of information is performed through direct dialogue, tactical radio channels or some other means of communication is irrelevant, it is the process of exchanging information that creates the common understanding not the means of communication. The exchange of information is in reality a conversation that has a goal of deriving explicitly or implicitly a common set of beliefs about the tactical activities. The result of merging the information collected through multiple conversations is a drive towards mutual understanding of the battle field and the units that operates within it. The process of creating mutual understanding between operational elements by clarifying their intentions and actions through conversation is known as *grounding* (12). Military orders, regulations and directives are formed with the objective of speeding up the grounding process and create a common foundation for interpreting and making sense of the operational situation.

Sensemaking is an activity or a process that aims at structuring the unknown. The process of grounding is to create mutual understanding while sensemaking would be the activity that makes grounding possible. An interpretation is the result of evaluating a situation, while sensemaking on the other hand constitutes the framework in which humans make interpretations. Sensemaking is grounded in the construction of identity. The personal and social identity is the result of past experiences and social interactions, which means that making sense of a situation is dependent on the individual's retrospective reflections of own experiences and social abilities. Sensemaking can be interpreted as a lifelong ongoing process of structuring the belief system of an individual (13 pp. 1-18).

The battle space is highly dynamic something which makes it hard to obtain a general view over the situation, this is known as the fog of war. Making sense about the battle space demands experience and training. During training one seeks to influence the sensemaking process in such a way that one creates a common belief system for the organization that enhances the ability to read the intentions and actions of other members of the organization, hence the grounding process speeds up. This chapter will look upon psychological theories that constitute the foundation for grounding and sensemaking.

The software agent constitutes the main building block of recreating human behavior in simulations. It constitutes a framework in which one can create computational models of psychological theories. Similarly to an individual the agent is an autonomous unit that can sense and act in the given environment. People interact through membership of groups: the group can be your family or the population of a whole nation, the point being that you can be a member of multiple groups at the same time. There exists multiple frameworks for building multi-agent systems, but in most cases the framework supports communication and establishment of conversations, while the content of the agent is in most cases up to the programmer. Figure 9 gives an overview of how different aspects of psychology on both an individual and group level is transferable to computational models through software agents.

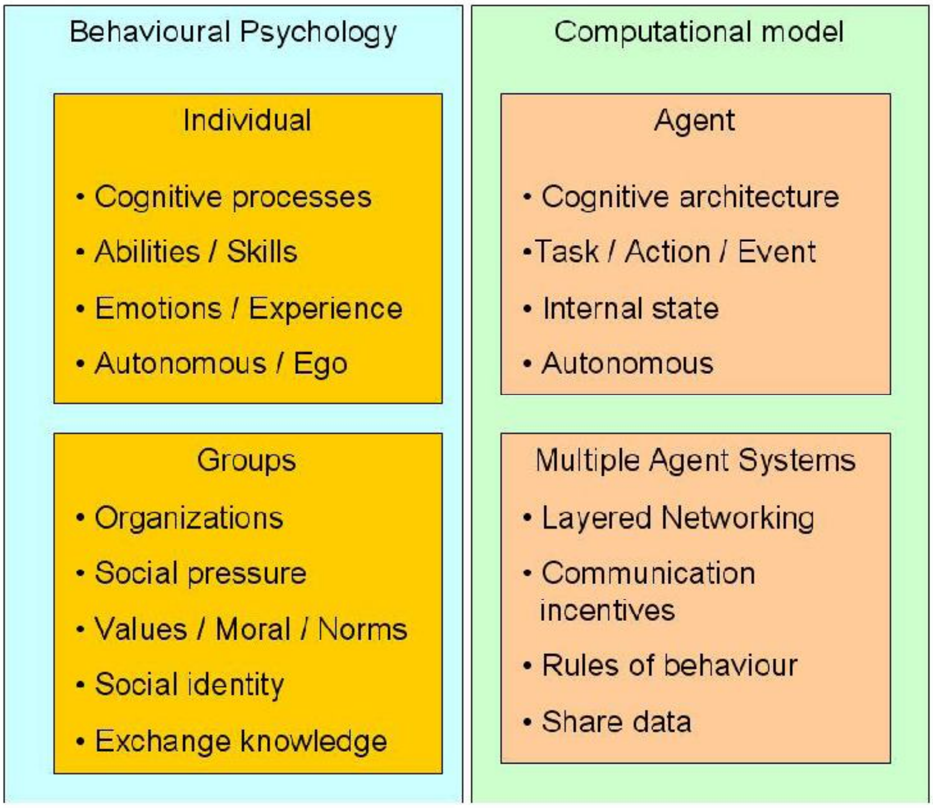


Figure 9 Prerequisites for HBR

3.2 Operational Psychology

3.2.1 Introduction

Operational psychology is a term describing the psychological issues of operating in environments which induce a considerable amount of pressure and stress. For military units this is highly relevant, because the consequences of miscalculating the seriousness of a situation could be catastrophic. Operational psychology is about investigating and understanding, under which circumstances the ability to take correct actions in an operational environment is reduced (Definition 1).

Definition 1: Operational Psychology

Operational psychology gives systematic knowledge about individual and contextual factors, which influence human behavior in operational environments and operational situations where life, health or basic values are threatened (14 p. 16).

Operational psychology is the foundation which the military organization relies on for executing successful operations. Most tasks performed on a daily basis are tasks that make the military organization ready to participate in or lead a variety of operations. Peace time tasks normally evolve around selecting new suitable candidates for military service, training and educating personnel, and performing daily routines and maintenance. Norwegian officers and soldiers have to undertake an examination before they can be enrolled into the military service. The examination consists of a number of physical and abstract tests, where the individuals can be categorized on the basis of the result. On the basis of the test scores individuals are either approved or rejected. The eligible officers also undergo a period of intensive physical and psychological strain. The purpose is to rank the individuals according to their potential for leading men and women into battle. When selected the next step is specialization and further education. Exercises and simulations further enhance the ability to make the correct decisions in stressful situations (14 pp. 13-30).

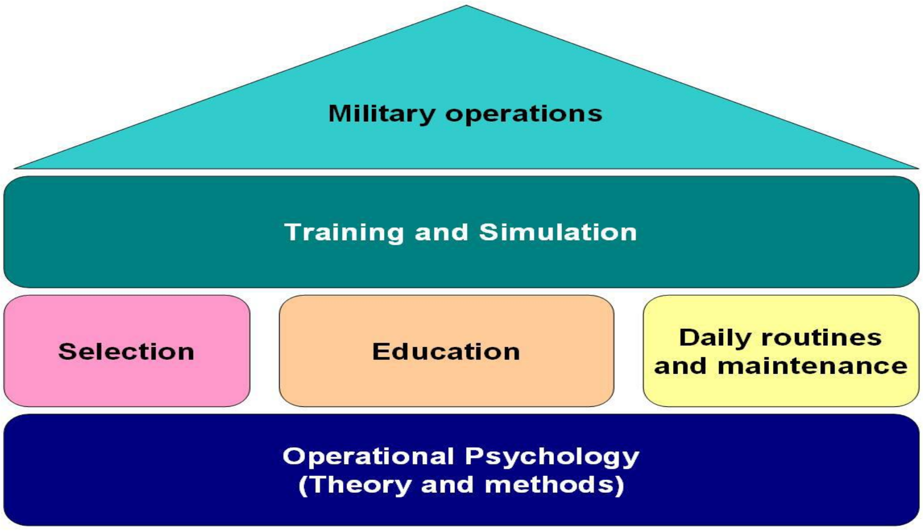

Military operations
Training and Simulation
Selection
Education
Daily routines and maintenance
Operational Psychology (Theory and methods)

Figure 10 Grounding Military Operations, from (14 p. 17)

Figure 11 gives an overview of research areas in operational psychology. The input constitutes the internal factors of a military unit and the throughput is what the unit with its internal resources can succeed in doing under the effect of environmental influences. The output is the degree in which one feels that the mission was a success or not. The degree of success can be measured in time or some other performance measure (14 pp. 20-24).

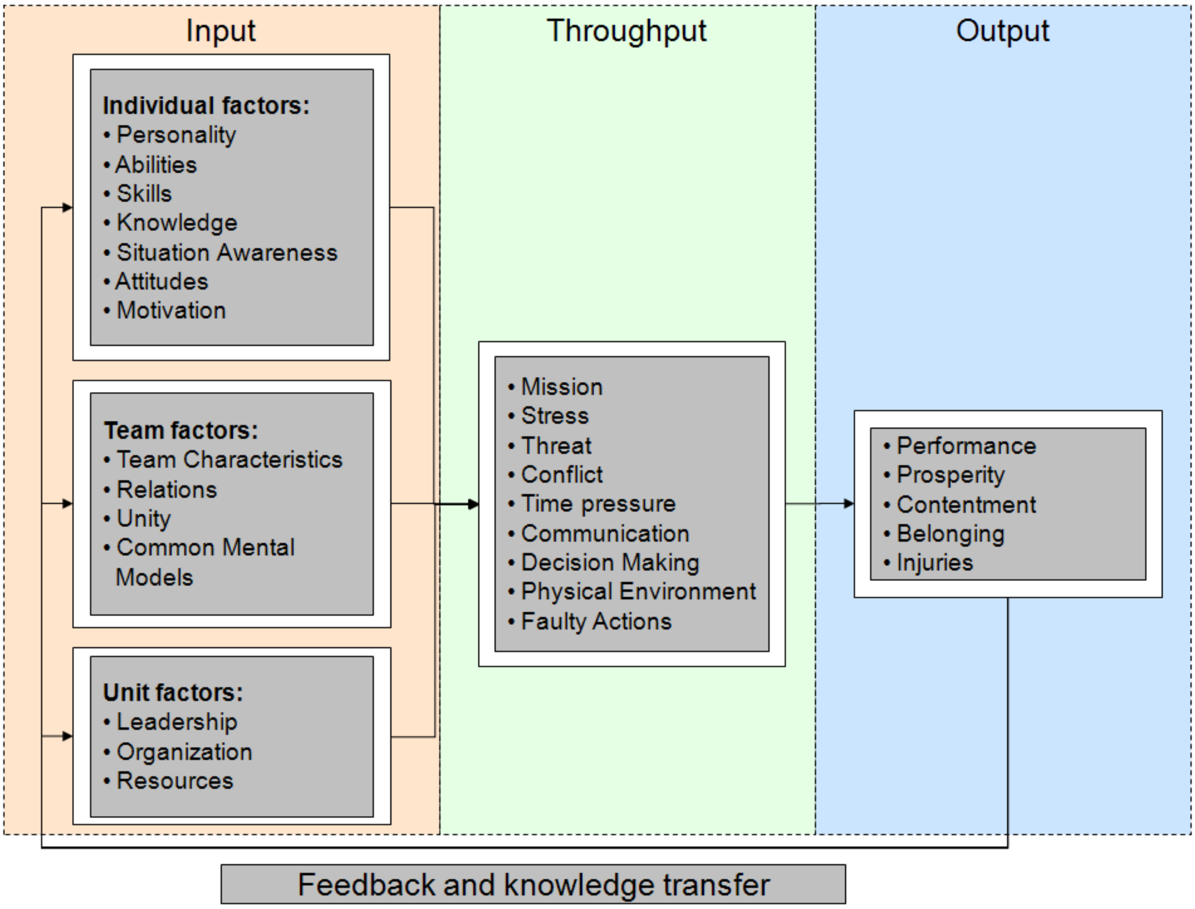


Figure 11 Research Areas of Operational Psychology, from (14 p. 25)

Operational psychology concentrates mainly on the following research areas: biological psychology, cognitive psychology, personality psychology and social psychology. This thesis will concentrate on areas that account for HBR in military simulations.

3.2.2 Biological Processes and Cognitive Neuroscience

The nervous system is the key component in human behavior. It channels all signals to and from the brain through interlinked bundles of neurons. In fact the brain itself consists of millions of neurons, something which makes the neuron the building block of the nervous system. The neuron is a specialized cell for transferring electrical signals and signal transfer from one neuron to another is done through chemical substances called neurotransmitters (15 pp. 31-34).

First and foremost we have the Central Nervous System (CNS) which is composed of the brain and the spinal cord. Secondly we have the Peripheral Nervous System (PNS) which controls muscles and regulates the function of inner organs. The processes within the PNS are to a large degree automated, for instance it activates organs in situations of stress with the intention of mobilizing reserves in the organism and on the other hand stabilize the performance of organs in other situations (14 pp. 38-40).

The most interesting part of the CNS is the brain. The brain controls our thoughts, emotions, motivations and deliberations. The brain can be divided into parts based on function which is denoted as lobes (Figure 12). The division into lobes shows that the brain is structured around some important functions, for instance sight and hearing. This composition can be found in many of the cognitive architectures to be represented in Chapter 4 about HBR.

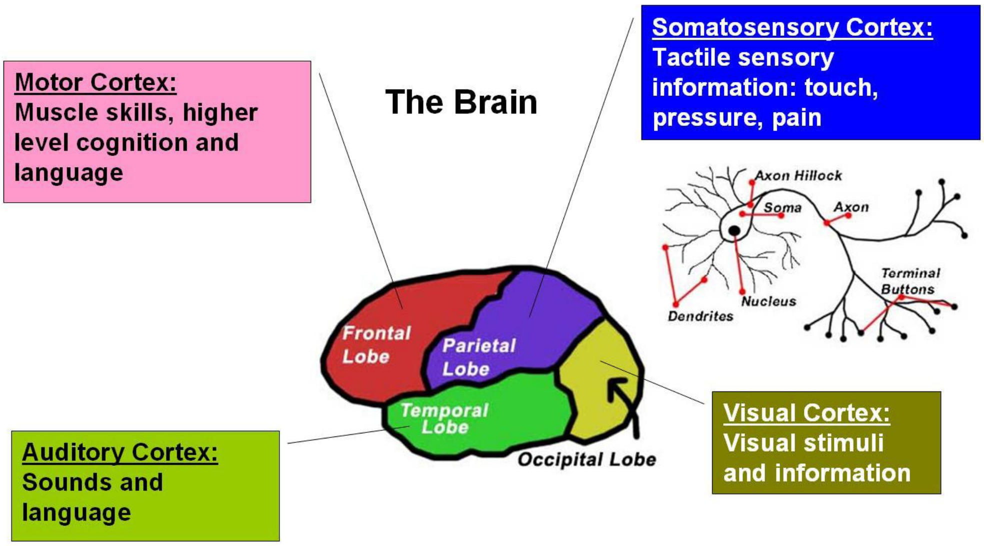


Figure 12 The Brain[1]

Learning is a process of growing neural connections within the brain. New connections between neurons mean that new knowledge is being stored. Areas of the cerebral hemispheres and cortex not covered by the lobes are considered to be association areas and constitute about 75% of the cerebral cortex. The association area has the responsibility to link sensory and motor cortices (15 pp. 57-61).

For the brain to function properly the most fundamental biological needs must be satisfied. The state of mind is influenced by the state of the body. In a military setting this is interesting because one can use biological needs as weapons. For instance can sleep deprivation cause irrational behavior and ultimately reduce the ability to fight. Other examples are the use of chemical substances to affect the nervous system (14 pp. 40-48).

3.2.3 Perception

The ways in which human beings perceive the world is often fundamentally different from individual to individual. Perception is not only given by the sensory inputs, but also the experiences that one has collected through one's existence (sensemaking). For instance the context one evaluates oneself in is usually based on the norms and rules of the society one is living in and the local culture.

[1] The figure is composed from several pictures downloaded from internet, namely the brain and the neuron. The composition is then organized in PowerPoint and text is added from (15 pp. 57-60)

The number of physiological senses that a human possess are considered to be six, and these are: sight, hearing, smell, taste, touch and the vestibular. The vestibular sense is basically a motion and balance registration system (14 p. 83). A perception process is how the organism form or shape an inner image of the outside world (Definition 2). The image is created on the background of the information collected and processed by the brain (Figure 13).

Definition 2 Perception

Perception is the set of processes by which we recognize, organize and make sense of the sensations we receive from environmental stimuli (15 p. 109).

The perception process can either be considered as a bottom-up or top-down process. The bottom-up process or direct perception is for instance when you see a cat you perceive the image of a cat and in your mind you make the associations necessary to identify and connect the notion of a cat with the image. If you already have stored the information this process is almost instantaneous. The top-down process or constructive perception is on the other hand focusing on the high-level cognitive processes. This is often the case when one is placed into a new and unfamiliar situation. The individual needs to process the stimuli by using stored knowledge and experiences in order to relate to the situation. The process is time consuming compared to the bottom-up process (14 pp. 84-87; 15 pp. 126-139).

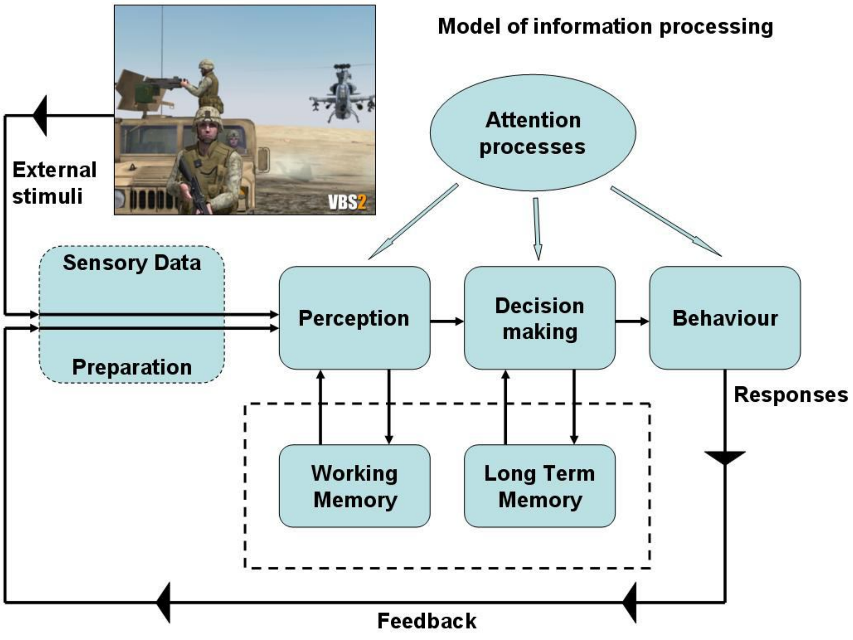


Figure 13 Processing of Information, from (14 p. 82)

In an operational setting it is important to understand the behavior of oneself and others in the interest of achieving a high degree of SA. Being able to read a situation and predict the reaction patterns of the enemy depends on collection of information. Identifying factors that influence perception is therefore of great importance when it comes to creating the correct image of the operational situation. Information is collected through our senses and filtered through the notion of our personal and cultural belief system (14 p. 81). In addition one of the reasons why military personnel use drill as a learning technique is to embed a certain action to a certain situation, which in return enhance the ability to react quickly. Drill is imprinting a certain action to a certain perceived situation (14 p. 59).

3.2.3.1 Attention

Theories of perception claim that perception is selective. People seem to only focus on relevant aspects of the world around them (Definition 3). Information not considered to be relevant is filtered out (Figure 14). The human brain has the ability to supply incomplete information with the missing pieces. Some sort of prediction or substitution of missing information makes partial information seemingly complete. In some cases the substitute could be wrong and the result is a misinterpretation of the situation. The analysis or interpretation of perceptions is performed according to a set of principles, including pattern recognition, chunking and sequencing of events (14 pp. 82-85).

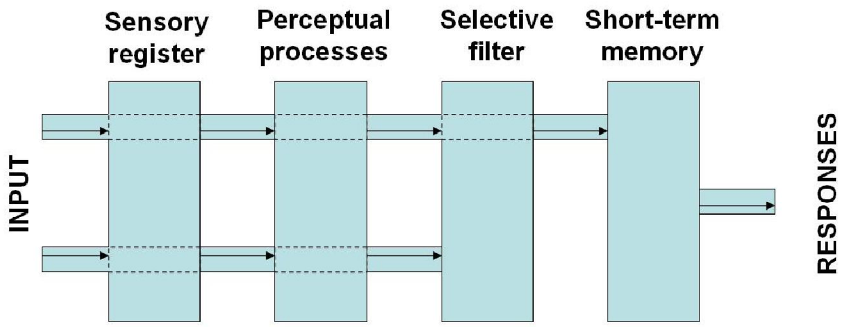


Figure 14 Attention: A Selective Filter, from (15 p. 93)

Definition 3 Attention

Attention is the means by which we actively process a limited amount of information from the enormous amount of information available through our senses, our stored memories and other cognitive processes (15 p. 66).

Attention can span over multiple or single tasks, and is divided into selective, divided and sustained attention. By selective attention is meant the ability to focus the mental capacity on following or understanding a certain number of stimuli from the environment. Divided attention means that one is concentrating on more than one task at the same time. If one tries to handle too many tasks at the same time one can end up suffering from dual task interference, where faults are committed due to that the handling of one task is disturbing the handling of another task. Sustained attention is the notion of an operator keeping focus on the same task over a longer time period. Usually the operator's ability to maintain attention on the same task is decreasing over time (15 pp. 91-100).

Most military operations are exhaustive in nature and the ability to keep focus is reduced over time. Making correct decisions in an operational situation is often life saving and the importance of keeping a high degree of alertness over time becomes crucial. To keep focus on the events or tasks which presumably could have the greatest impact on the situation becomes a priority (14 pp. 90-91).

3.2.4 Memory

With memory is meant the organism's ability to store, retain and retrieve information. The questions one is trying to answer is how information is encoded and registered into memory, how the information is stored by creating some sort of temporal or permanent record and finally how information can be retrieved from that storage as a response triggered by a cue.

Definition 4 Memory

Memory is the means by which we retain and draw on our past experiences to use this information in the present. As a process, memory refers to the dynamic mechanisms associated with retaining and retrieving information about past experiences (15 p. 149).

The three-store model prepared by Atkinson and Shiffrin supports two main dimensions of memory, namely that memory can be either structural or functional. The structural memory is permanent and determined biologically (Figure 15). Included in a structural memory are sensory registers, short-term memory (STM) and long-term memory (LTM).

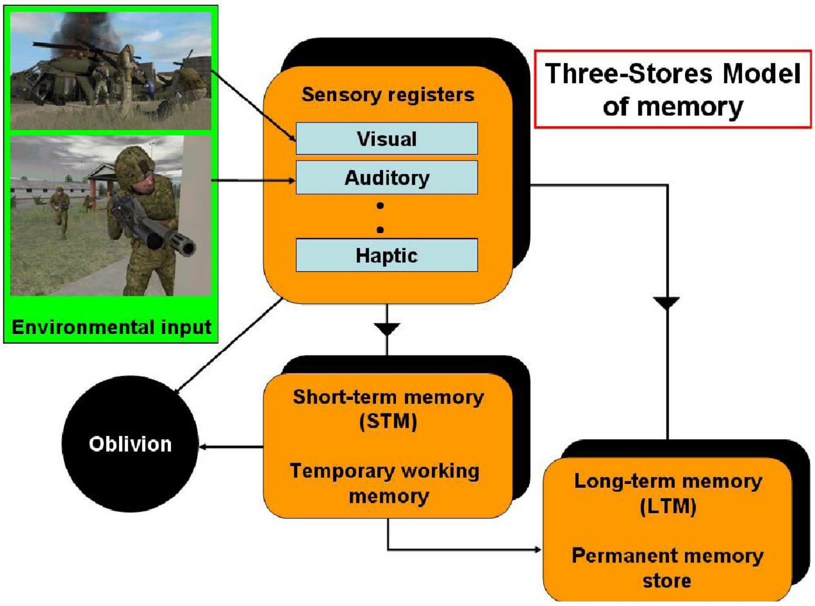


Figure 15 Three Store Model of Memory, adapted from (14 p. 67; 15 p. 153)

The functional memory on the other hand is more controllable by the individual itself. This includes control operations like encoding, repetitive operations and search strategies. The theory states that the functional dimension is the control entity of the structural dimension. The individual has influence over which perceived inputs that are going to be stored, which implies that selected information is copied between storages. Information that is not stored in LTM will pass into oblivion (14 pp. 65-70).

- **Sensory memory** - contains information immediately conceived from the environment. Often one distinguishes between iconic and echoic memory. The storage capacity of the memory is perceived to be large, but the storage time frame is limited to a few milliseconds. The contents in sensory memory are either copied or erased (14 p. 67).

- **Short-Term Memory** - contains information that is currently being focused on. The storage capacity is limited for instance typically seven items (give or take two items). The storage time frame is limited to a few seconds and up to a couple of minutes. A typical method for retaining information is by rehearsing; repeating the information until it sticks (14 p. 68).
- **Long-Term Memory** - contains information collected over a life time. The storage capacity is considered be large. Storage is considered to be permanent and information is presumably organized into categories and patterns for easy access and recollection (14 p. 69).

To capture the notion of different memory types one has created a taxonomy (Figure 16) based on the information contents of the memory (15 p. 165). A skill is considered to be stored in procedural memory (implicit memories). These are memories where one associates the task with earlier experiences on an unconscious level. For instance when you have learned to ride a bike you will not forget how you do it, and you do not need to think about it either. Declarative memory on the other hand is memories one specifically are trying to recall and recognize (explicit memory), and can be further divided into semantic and episodic memory. Declarations have the characteristics of containing facts (semantic memory) and events or situations (episodic memory) (15 pp. 163-164).

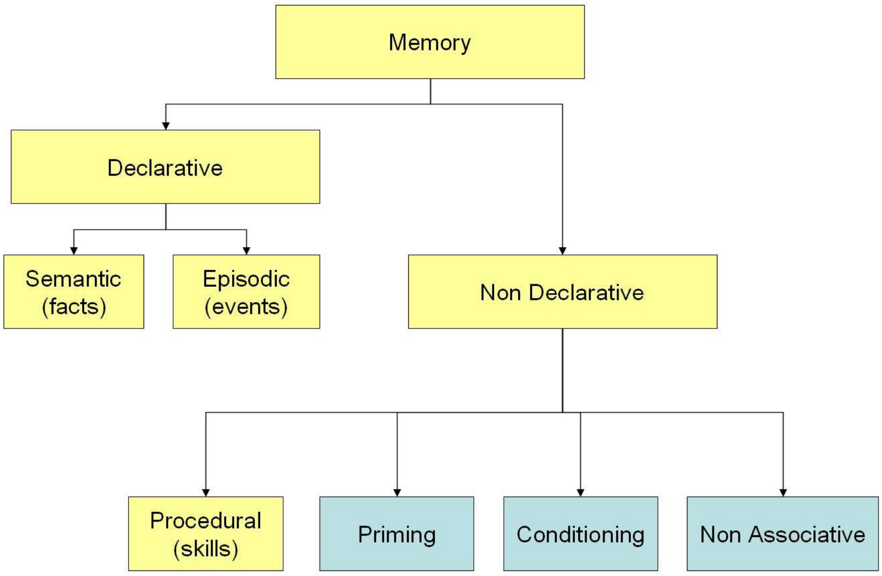


Figure 16 Taxonomy of Memory, from (15 p. 165)

Figure 16 shows 3 non-declarative memory types that have not been elaborated on. These types are strongly connected to the learning process and will therefore be discussed in chapter 3.2.5.

The model presented so far is known as a traditional model of memory, but has been challenged by other suggestions on memory structures, one being based on an ever increasing depth of processing. The theory is known as Levels-of-Processing (LOP). Memory is in this theory considered to be structured as a long continuous dimension where the complexity of the perception decides how deep into the memory one needs to process it. The deeper one goes into the levels of processing the larger the probability is to retrieve it. This implies that larger depth means higher degree of semantic or cognitive analysis (15 p. 159).

A supplement to the LOP theory is the notion of working memory (WM). The WM contains the information most recently activated from LTM and STM. It is a temporal storage buffer for being able to maintain information in storage simultaneously with processing new information (14 p. 71). You might make a comparison with the cache memory in a computer, where information stored on the HD or in RAM is retrieved for easier access to the processor. The principle being that information recently used has a high probability of being reused within a short time frame.

There are several different factors that affect the memory in some way. The use of drugs and stress has negative effect on memory while some smart drugs and regular exercise have positive effects on memory. The ability to store and retrieve information is important when one needs to assess an operational situation. The mental capacity is influenced by stress factors which typically are noise, heat and time pressure in a military operation. In addition experiments have shown that both overexposure and underexposure to sensory stimulation is perceived to be stressful for a human being. Sensory deprivation can cause anxiety and insecurity (14 pp. 86-96). Stress factors moderate human behavior and is therefore often used as performance moderator functions (chapter 4.5).

3.2.5 Learning

The study of learning seeks to identify universal laws of how one acquires knowledge. The definition of learning is dependent on the view point of the beholder. One view point is to see learning as a product or an outcome of a process and another view point would be seeing learning as a process (Figure 17).

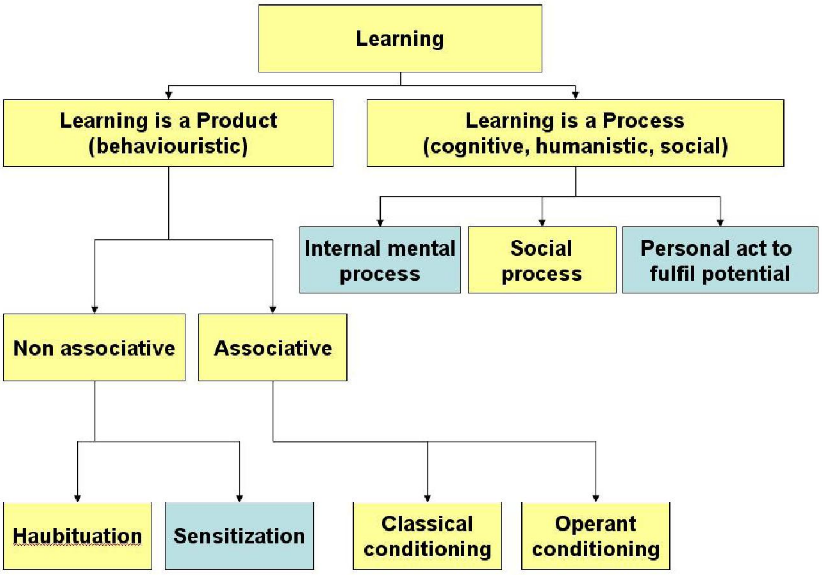


Figure 17 Taxonomy of Learning[2]

Behaviorists look upon learning as change in behavior. A change in behavior includes establishment of new behavior or modification of existing behavior. In addition the occurrence of certain behavior can increase or decrease (Definition 5).

[2] The figure is a result of visualizing topics of learning fetched from Wikipedia and (14 pp. 51-64).

Definition 5 Behavioristic Learning

Learning is a relatively permanent change of behavior which is caused by a concrete experience the individual has gained (14 p. 51).

Non-associative learning is also called single event learning and is a type of learning that includes recognizing an event as unrelated or irrelevant. For instance a sound that is unfamiliar could keep you up all night, but when you get used to it, the sound will fade away and you will not pay attention to it anymore. In this case one gets accustomed to the sound and this is known as *habituation* (14 pp. 51-52).

Associative learning on the other hand is a learning process in which a new response becomes associated with a specific stimulus:

- The psychologist Ivan P. Pavlov studied learning on the bases of how dogs associate stimulus to certain expectations of events and is known as ***classical conditioning***. The dogs learned to associate the bell with food and adopted the same response to the bell as to the food itself (14 p. 53).
- ***Operant conditioning*** originates from the psychologist Burrhus F. Skinner. The principle is that a stimulus presented after the individual has displayed certain behavior can reinforce the behavior. Reinforcement of a behavior yields that the probability of repeating behavior will increase. The reinforcer exists in the environment and as a result the environment controls the individual (14 p. 56).

In contrary to the behavioristic approach to learning, theories on social learning claims that individuals can influence their behavior by either creating or seeking out different environments. The notion is that processes within an individual like expectations, reasoning and emotions will influence and be influenced by the behavior. The psychologist Albert Bandura introduces a social point of view of explaining learning processes. Observational learning is when a person learns by observing a behavioral pattern. If the individual copies the behavior without understanding why the behavior is functional, it is called *imitation* (14 pp. 61-63). Bandura identified three characteristics of social learning:

- **Characteristics of pattern**: Attractive patterns or patterns perceived to be close to one's own behavioral pattern are more easily learned. This can be a result of a relation to for instance a role model.
- **Characteristics of observer**: Individuals with low self-confidence will more easily adapt or incorporate patterns into their own behavioral pattern, facilitating imitation.
- **Consequence of behavior**: Behavior patterns conceived to be superior or functional are more easily adapted and incorporated into one's own behavioral pattern.

The Social learning theory states that by adapting and incorporating behavioral patterns one obtain a set of standards. A standard implies personal goals in which the individual assess its own behavior against. The theory therefore explains how persons might obtain values and a set of moral codes. Self efficacy reflects self-examination of one's capacity of mastering or implementing a specific behavior. The self-perception of one's capability to master this behavior affects the ability to learn. If you think you can do it you probably can. The expectations of the outcome affects the probability of learning, Bandura called this for outcome expectancies (14 pp. 61-63).

The training and education of personnel is important in any setting where one wish to perform well. A crew has the responsibility to learn and maintain skills, knowledge and appropriate conduct. The effect of *habituation* is to eliminate unnecessary responses. In an operative sense this is important to be aware of, since decreased awareness is not a wanted state of affairs. Measures taken to prevent habituation are called *dishabituation*. Classic conditioning describes unconscious learning and is relevant when it comes to coping with stress and fear. Traumatic situations can trigger Post-Traumatic Stress Disorder (PTSD). Operant conditioning on the other hand is relevant for most training aspects in the military. A specific behavior can be motivated through use of positive or negative reinforcement measures. The education of soldiers and officers is conducted by experienced personnel that form role models. This is an example of a social learning technique where the experienced guide the inexperienced (14 p. 63).

3.2.6 Emotions

Emotions are a set of functions to help evaluating the situation an individual is in. Situations that are perceived to be favorable are rewarded by emotions that encourage pursuing actions that ensure reoccurrence of the same situation or similar situations (Definition 6). Bad emotions are an encouragement to avoid getting into the same situation again. The utility value of emotions is to act as a heuristic function, helping the individual to evaluate the environment and take correct decisions (14 p. 98).

Definition 6 Emotion

An emotion can be defined as a limited and incorporated cognitive, psycho-physiological and behavioral response (14 p. 98).

An emotion consists of three different response systems: a certain physical or behavioral response, a pattern of physiological changes and certain subjective experiences or sensations (Figure 18). This constitutes facial expressions, body language, heart frequencies, blood pressure and respiration. Emotions change behavior by bypassing the high-level cognitive layer of reasoning. Amongst basic emotions are: fear, anger, joy, curiosity, disgust and neutrality. The emotional expressions will have influence on decision-making and the relation between members of operational teams (14 pp. 98-99).

The individual has different approaches to how to handle stress. There have been identified three individual strategies for mastering stress: task-oriented, emotion-oriented and avoidance. Task-oriented persons would try to solve the problem in an orderly fashion, while emotion-oriented persons would use energy on working through one's own frustrations. The avoidance strategy includes denial and withdrawal from the problem. Studies show that persons using a strategy of avoidance are vulnerable to chronic stress related illnesses (14 pp. 112-113).

In military operations it is important to suppress some of the normal emotions in order to be effective in the job. Stress induces a number of effects on the body (Figure 18). There is a relation between externally induced load and individual stress reactions (14 p. 103).

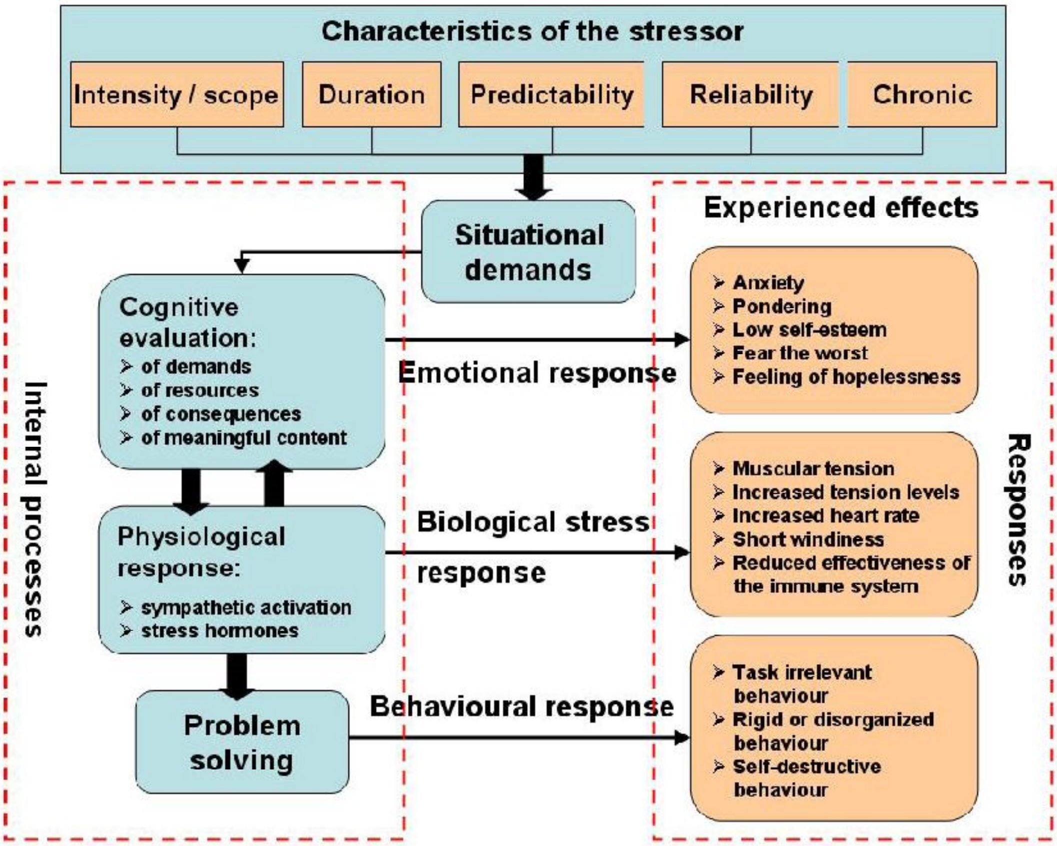


Figure 18 Stressors and Effects of Emotions, from (14 p. 105)

A factor that has an effect on leadership is emotional intelligence. Experiments have shown that the ability to understand one's own and other's emotions and use this knowledge in reasoning and problem-solving is an advantage when leading teams and organizations. The leader is able to discover or read emotional expressions in others and regulate own emotions that are perceived to be irrelevant or hurtful for the organization. This is especially important in high context cultures, where one emphasize the use of expressions. For instance show submission or compassion etc. An example of a high context culture would be Japan and a low context culture would be Norway (14 pp. 101-102).

Emotions are ideal for use when it comes to human factors and performance moderator functions (chapter 4.5). Emotions create motivation and intention, which in many cases does not need to be considered rational. In a combat situation many emotional manifestations are considered to be unwanted, for instance being unable to move due to fear. In a war scenario the performance of a human is limited by the amount of stress the human organism can withstand, for instance can such an overload manifest itself as loss of vision known as hysterical blindness.

3.2.7 Problem Solving and Decision-Making

In cognitive psychology one is interested in how people in general solve problems and makes decisions. Generally when one is trying to solve a problem one goes through a cyclic process, which structures the problem-solving into phases. The path from having a problem to finding a solution is covered with intersections where one needs to make a decision about which path to follow next.

Meaning that the problem-solving process includes several decision stages where one needs to choose a direction toward the solution (Figure 19).

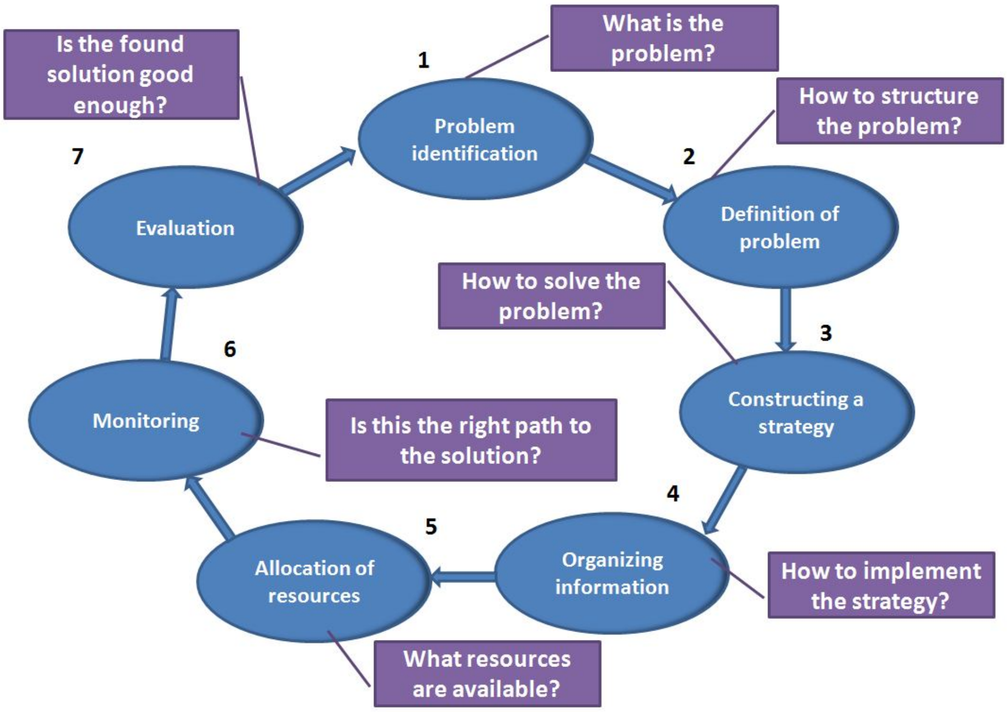

What is the problem?
1
Problem identification
Is the found solution good enough?
7
Evaluation
How to structure the problem?
2
Definition of problem
How to solve the problem?
3
Constructing a strategy
6
Monitoring
Is this the right path to the solution?
4
Organizing information
How to implement the strategy?
5
Allocation of resources
What resources are available?

Figure 19 Problem-solving Cycle, from (15 p. 361)

The space between the problem state and the solution state is called the problem space (Definition 7). The amount of pathways to the solution can be many and is dependent on the structure of the problem. Well-structured problems have clear and sometimes obvious paths to the solution, while ill-structured problems have no clear solution paths. Research on problem-solving often uses computational models for testing theories. An algorithm is generally conceived to be a recipe of how to solve a specific problem or a specific category or group of problems while a heuristic is a computational short-cut to an effective solution. Human beings process their problems in the same way, people develop procedures and recipes for solving problems based on their mental model of the world and the knowledge they possess. Heuristics are in this aspect a mental short-cut to a satisfying solution and can be an emotion (15 pp. 364-378).

Definition 7 Problem Space

The problem space is the universe of all possible actions that can be applied to solving the problem (15 p. 369).

People use many different techniques in their problem-solving, and there are several different approaches or strategies of how to attack a problem (Figure 20). The most obvious approaches are either to work oneself backwards from the solution state to the problem state or working forward from the problem toward the solution (top-down or bottom up approaches). Another approach is to generate solutions and test if they are valid, the process repeats itself until a valid solution has been found. It is also possible to use combinations of the methods, a method that is based on reducing the gap between the problem state and the solution state is called means-ends reasoning (a continuous comparing of current state and goal state) (15 p. 368).

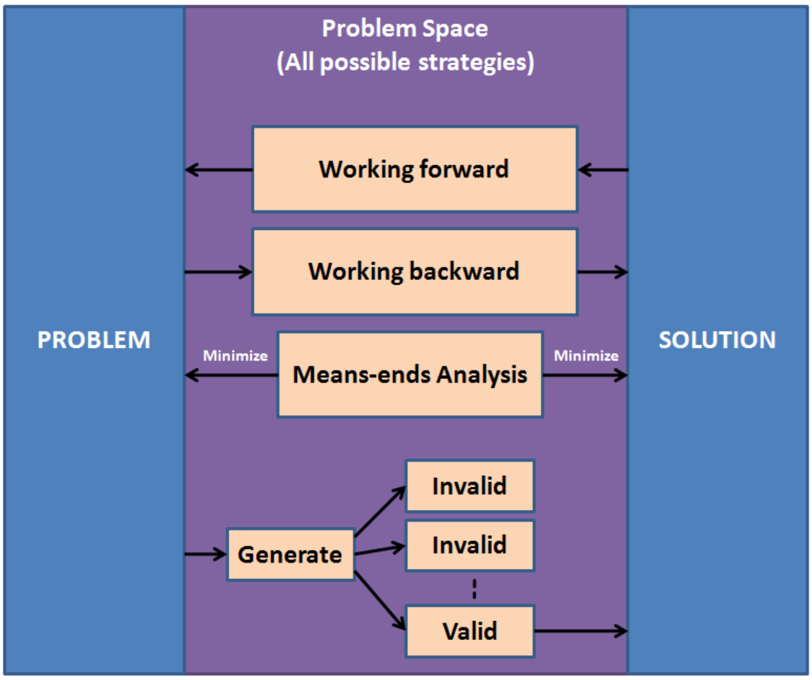


Figure 20 The Problem Space, from (15 p. 369)

Humans are predisposed of using problem-solving methods that have worked in the past. Procedures can in many instances hinder the discovery of more optimal solutions, because they covert new ways of looking at a problem. Problems are represented as a mental set in the human mind and this is the current context in which the individual perceives the problem. The problem is identifying knowledge and skills that is transferable to another situation (15 p. 369).

The problem-solving process involves several stages of decision-making. The purpose of decision-making is selecting amongst a diverse variety of choices and opportunities. The decision is based on the knowledge one possesses of the situation. The amount of information available will affect the result. For instance a fully informed decision relies on that fact that all information about the problem is available, and it is possible to evaluate all possible outcomes in the problem space. To be able to reason about pros and cons one must be able to distinguish between the directions to take and rank them according to some criteria. The notion of a utility function is biased, because each individual for most parts perceive the world differently and therefore one is talking about a subjective utility function and bounded rationality (15 pp. 404-410).

The most known and wide spread notions of decision-making are deductive and inductive reasoning. Deductive reasoning is based on propositions and use propositional logic to find a soundness of a decision, meaning that it is theoretically possible to find a solution that is deductively valid. Inductive reasoning on the other hand is of a different nature, it is based on past experiences and observations, where logically conclusions cannot be considered possible. One could look upon these two opposing reasoning theories as either an intuitive or analytical approach to reasoning (15 pp. 419-438).

Military forces have guides to help them to solve problems and speed up the decision-making process. Three such supporting documents are the Standing Operating Procedure (SOP), the Rules Of Engagement (ROE) and the FFOD (FFOD uses John Boyd's OODA-loop as an example of a decision-making cycle, chapter 2.4). The planning of military operations involves evaluating risks and effects of the different operational options available. The evaluation strategies one can use are either compensatory strategies or non-compensatory strategies. In the first case one compensates weak attributes against strong attributes in for instance an operational plan. In the latter case one evaluates according to cut-off values. If below the minimum or above the maximum the plan is not carried through. On a tactical level the situation often is highly dynamic and one has to change the approach to the problem according to the changing situation. Extensive training and drills help the soldiers in their task of coping with the dynamic and rapid changing environment (14 pp. 155-157).

3.2.8 Personality Psychology

The preliminary sections have described characteristics of the individual that are considered to be equal from individual to individual. Cognitive psychology describes how a human being learns, organizes knowledge, perceives and feels. These are structures considered to be equal in all humans; the mechanisms explain how a human being functions. Personality psychology on the other hand focuses on the characteristics that are different from individual to individual. Individual differences influence the processes of grounding and sensemaking (chapter 3.1). This chapter takes a closer look at intelligence and personality theories.

A personality feature studied in detail has been the study of intelligence. There are many definitions on the notion of intelligence; some mean it is a group of specific abilities while others mean that it is one ability altogether (Definition 8).

Definition 8 Intelligence

Intelligence is the ability to learn from one's experiences and to adapt one's behavior according to the environment.

The definition makes the notion of intelligence dependent on culture and environment. It is not enough to use oneself and one's own culture as a reference point, because if you operate in another environment under another context or culture the notion of intelligent behavior has changed.

The psychologist John Bissell Carroll presented in 1993 a model called the three-stratum theory of cognitive abilities. The model is a hierarchy of depth three where you have the g-factor which is the general intelligence factor and eight specific abilities (Figure 21).

In the military the measurement of intelligence is used in the selection process, meaning that soldiers in the end are given assignments that are suited for their cognitive capacity and personal skills. Intelligence is measured through intelligence tests, measuring the Intelligence Quotient (IQ). The tests are standardized and must be valid and reliable. The tests are based on the normal distribution where a score of a 100 is average. The model by Carroll shows that there is a dependency between general intelligence and cognitive factors like memory and attention (Figure 21). Reaction time is one example of this. Persons that have reaction times considered to be above normal usually also have high scores on the IQ-test (14 p. 124).

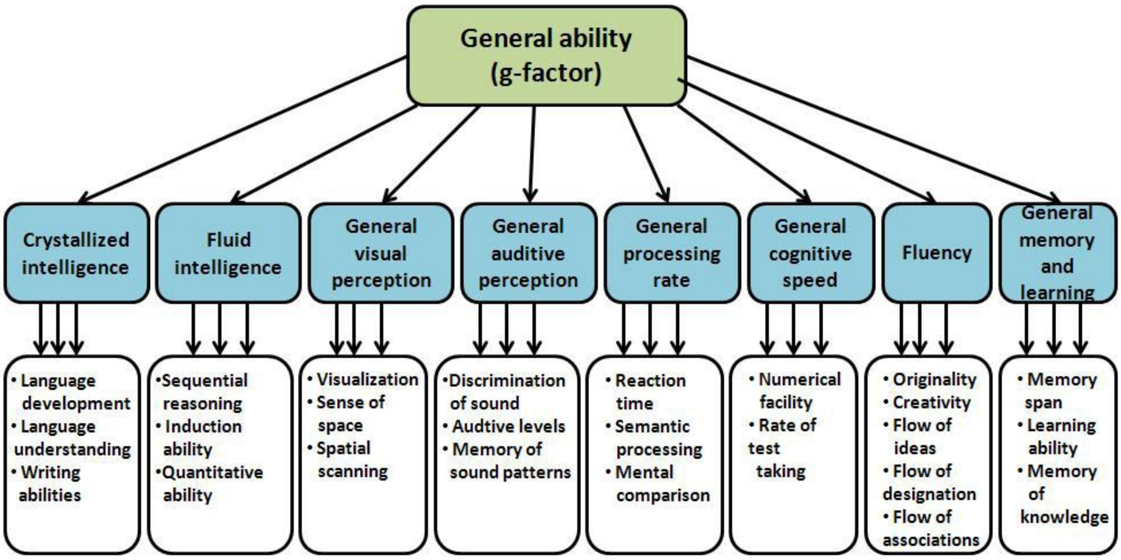


Figure 21 Structure of Intelligence by John B. Carroll, from (14 p. 123)

Some personality theories involve theories of learning, while other theories are based on categorizing individuals into types or traits. Learning theories look upon personality as behavior or habit that is learned through encounters with different factors in the environment, the consequence being that a person will change behavior over time. During a lifetime the behavior will have changed considerably. In this theory personality is a matter of a learning process (14 p. 127).

Theories based on types classify a person as being in a personality category, which is something you are or are not. The division is based on identification of certain personality characteristics and recognizing the absence of other characteristics. Types are considered to be too dogmatic which has lead to the notion of trait theory. In this case one identifies characteristics that all individuals possess and measure to which degree they correspond to the characteristic (14 p. 127). The result can be presented as a personality graph (Figure 22).

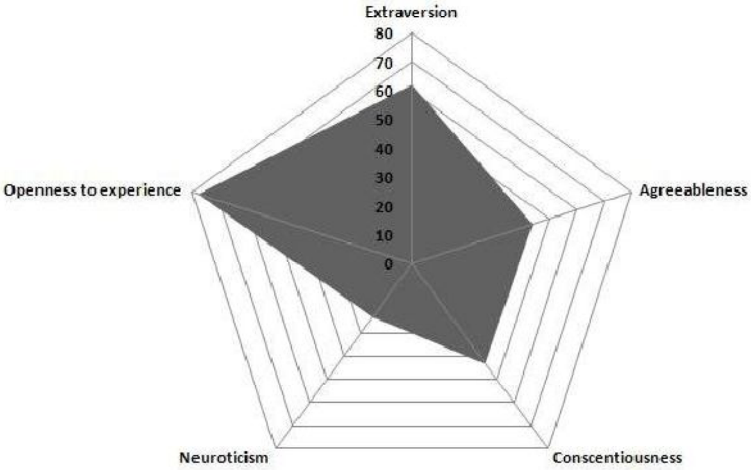


Figure 22 Five Factor Personality Graph, constructed from IPIP-NEO personality test

There are dependencies between personality and the ability to lead teams or groups. A person's ability to get a team to cooperate towards a common goal will increase the effectiveness of the group. Different situations demands different leadership styles and an individual's personality often influence the choice of preferred style. The personality also gives a picture of which individuals that are more likely to cooperate and work in teams. The military and the police perform selection processes before they employ people. The process makes it possible to select individuals with the correct attitudes and preferred preferences (14 p. 132).

3.2.9 Social Psychology

Social psychology is the study of how people interact. The social behavior is explained through theories of how people through interaction change the behavior of each other. Allport's definition is based on the idea that social psychology is the study of social influences (Definition 9). The term social influence encapsulates some of the major areas in social psychology such as: persuasion, change in attitude and conformity. The social context to a large degree influences the processes of grounding and sensemaking (chapter 3.1).

Definition 9 Social Psychology

Social psychology is an attempt to understand and explain how the thought, feeling and behavior of individuals is influenced by the actual, imagined or implied presence of others (14 p. 137).

Social contexts can be described as memberships in different groups. A group consists of a set of members (Definition 10). Groups can be categorized through the time frame they exist and the characteristics of the connections between the individuals.

Definition 10 Group

A group is defined as two or more individuals who are connected to one another by social relationships (16 p. 3).

Groups consists of individuals, therefore it is important to consider personality when talking about groups (chapter 3.2.8). Some people seek to be a member of groups more often than others. According to social identity theories the individual performs different cognitive unconscious elaborations based on self-evaluation within the frame of the group (17). The individual will in these processes classify people into categories and accept the group as an extension to oneself. The result of such categorization can be seen in for instance the tendencies to create stereotypes (16 p. 92).

The individual's need to belong to a group results in the action of joining a group. Individuals join groups for different reasons, which could amongst others be the need for affiliation, need for intimacy or need for power. One other aspect is the attraction of a group. Some groups seem to be more appealing than others, and the attraction factor is based loosely on some principles presented in Figure 23 (16 pp. 123-132).

The structure of the group is created through interaction between its members. The development of a group goes through several phases over time. The duration of a phase depends on which basis the group was formed (16 pp. 146-152).

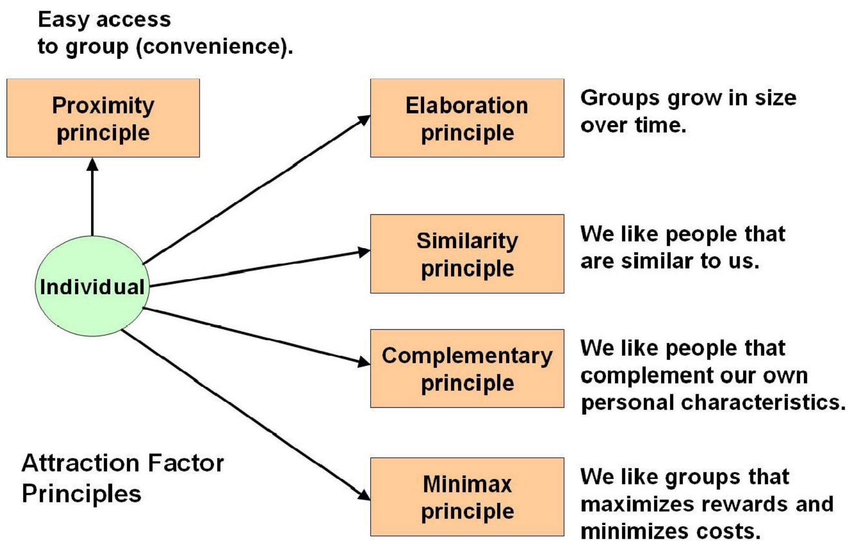

Easy access
to group (convenience).
Proximity principle
Elaboration principle
Groups grow in size over time.
Similarity principle
We like people that are similar to us.
Individual
Complementary principle
We like people that complement our own personal characteristics.
Attraction Factor Principles
Minimax principle
We like groups that maximizes rewards and minimizes costs.

Figure 23 Principles of Group Attraction[3]

The connection between the individuals in a group can be said to have a degree of strength. Groups that contain individuals with a strong bonding are said to be cohesive. Cohesion is a term that describes the attraction factor in the group or the group unity. Group structure is made up of roles and norms (16 pp. 136-145).

The dynamics in a group decides the power of a person or in which degree a person has influence over another person. Social influence is a interpersonal process of exerting influence, while majority and minority influences are based on who is exerting influence on the group, either it is a majority vote or in the case of a jury one person might exert pressure to re-evaluate the votes of the many. Typical ways of exerting power in a group are through reward and punishment (16 pp. 208-227).

Conformity is an expression that is describing how eager an individual is to please the group. People have certain tendencies to be conforming to the rules and norms of the group. Conformity can therefore be conceived to be the measure of an individual's disposition to surrender to group pressure without directly involving the group. Obedience on the other hand is people's tendency to comply with explicit demands, often a well-known authority figure (16 p. 208).

The military need individuals with certain characteristics to operate effectively. The characteristics can be different depending on the job that the person is going to do, for instance there is a large difference between a fighter pilot and a soldier on guard duty. Individual factors affect the ability to lead others, to make sound and logic decisions and to cope with stress, but an operation usually involves a large number of people working together. The teams have to cooperate effectively and have coinciding mental models (grounding chapter 3.1). The structure of the organization, the leadership exercised and the amount of resources available will have an impact on the execution of the operation, therefore building cohesive teams is of great importance (14 pp. 148-152).

[3] Visualization created on the background of text in (16 pp. 123-132).

3.3 Artificial Intelligence

3.3.1 Introduction

Artificial intelligence (AI) is the notion of creating or building an entity that is capable of acting intelligently. Understanding what AI is all about, is understanding what is meant by the notion of intelligence. There are many ways of looking at intelligence, but the most elaborated definitions involve the capabilities of reasoning, learning and understanding behavior. To be able to address the complicated task of resolving these problems the AI community relies on scientific discoveries from many other scientific fields. The notion of AI involves theories from amongst others philosophy, mathematics, economics, neuroscience, psychology, computer engineering, control theory, cybernetics and linguistics (5 pp. 1-16). The basis of an intelligent entity is the notion of an agent. An agent has several distinct characteristics or properties that coincide with intelligent behavior (Definition 11).

Definition 11 Agent

An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives (18 p. 15).

Rationality involves elaboration on how to solve a given task. It states the need of performance functions, memory, data and information representation and the means to perceive and act in a given environment (Definition 12). Agents can operate in any given environment, which means that the agent can be made entirely in software an act in a software environment or be embodied in hardware and act in both a software and real world environment. Acting in the real world further adds to the complexity of the agent.

Definition 12 Rational Agent

For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has (5 p. 36).

The more complex tasks an agent needs to perform the more complex the agent gets. It is important to be aware of that there exists several definitions of agents, but in most aspects the above definitions gives the same basic idea of what is meant by the word agent and what we are talking about in this context.

3.3.2 Agent Program Types

Agents can be categorized into types according to the complexity of the agent program. For the sake of the simplicity we will concentrate on the notion of software agents acting in a software environment. Figure 24 shows a general purpose model of an agent and how it interacts with the surrounding environment. The agent type is given by the definition of the contents that process perception through sensors and choose an action to impose on the real world through actuators.

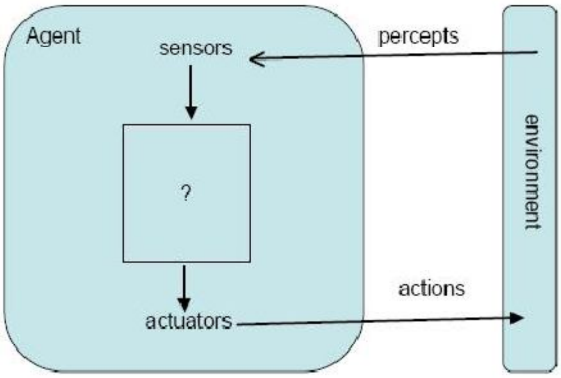


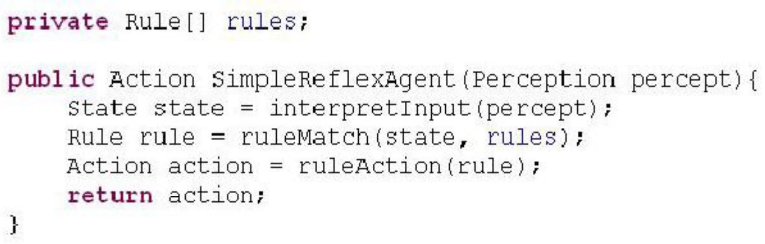
Figure 24 General Purpose Agent, from (5 p. 33)

The software agent basically consists of the agent program and its sensors and actuators. The agent program is situated in the box marked with a question mark. The program contains an agent function that describes the agent's behavior through mapping a perception or a sequence of perceptions to an action. The sensors and actuators constitute the architecture of the agent (5 pp. 44-46). The types we are going to take a closer look at are simple reflex agents, model-based agents, goal-based agents, utility-based agents and learning agents. The progression in complexity is shown through giving coded examples. The examples show a steady evolution of the agent.

3.3.2.1 Simple Reflex Agents

These agents are purely reactive in the sense that they do not store any history of previous decision-making and therefore base the action on a direct mapping between the perception and the action (Figure 25). The observed state or condition triggers directly the action. The mapping is typically done through condition-action rules and is therefore a rule-based decision-making program (5 pp. 46-48).

The simplicity is one of the advantages of the agent program, but it is overshadowed by the fact that it has great limitations on range of use. The agent needs to have full overview of the state of the environment if it is to be capable of rational behavior. To avoid loops of reoccurring actions randomization of actions is a possibility, but this exclude intelligent behavior.



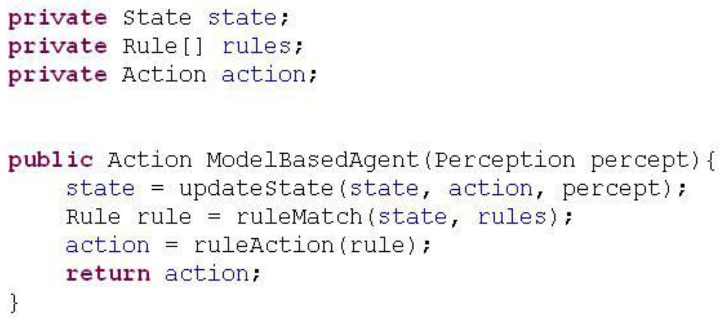
```
private Rule[] rules;

public Action SimpleReflexAgent(Perception percept){
    State state = interpretInput(percept);
    Rule rule = ruleMatch(state, rules);
    Action action = ruleAction(rule);
    return action;
}
```

Figure 25 Code Example Simple Reflex Agent, adapted from (5 p. 47)

3.3.2.2 Model-Based Agents

Model-based agents can store information of previous perceptions in memory and therefore can elaborate on which action to take next. This basically means that the agent can keep an internal state. The internal state can either be a history of previous actions or a model of the outside world or both. The internal state makes it possible to make presumptions or predictions of how the world is going to evolve and base the action upon this knowledge (5 pp. 48-49).

In contrast to the simple reflex agent the model-based agent have the advantage of basing its decision-making on its experiences of the perceived world (Figure 26). The perceived world is not necessarily the same as the actual world state, which means that the agent is capable of making decisions based on partial information of the world state. The limitations are that the agent cannot learn because it cannot evaluate its performance. The program is still limited by the fact that it can only prepare one action for a certain state.

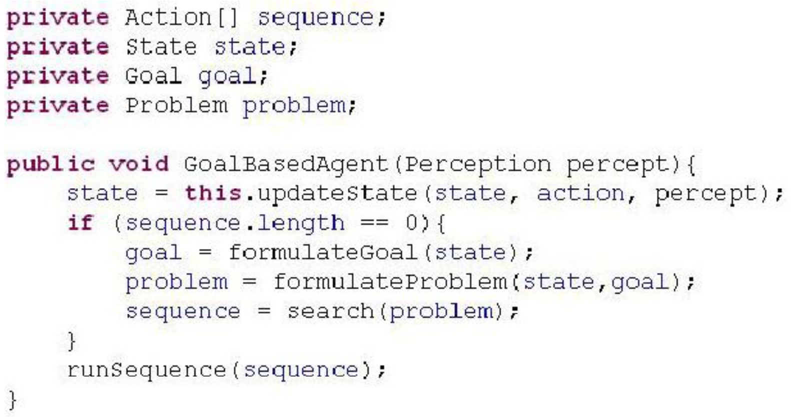


```
private State state;
private Rule[] rules;
private Action action;


public Action ModelBasedAgent(Perception percept){
    state = updateState(state, action, percept);
    Rule rule = ruleMatch(state, rules);
    action = ruleAction(rule);
    return action;
}
```

Figure 26 Code Example Model-Based Agent, adapted from (5 p. 49)

3.3.2.3 Goal-Based Agents

A problem with the simple reflex agent and the model-based agent is that it does not know the purpose behind its actions. The purpose is given by an intention or goal of what you try to accomplish. The goal-based agent solves this problem my introducing goals. A goal is a description of a desirable situation or state that the agent can adapt its behavior to. A goal is a state that you intent to satisfy and on the way on to satisfying the goal you may choose from different sequences of actions to get there (5 pp. 49-50).



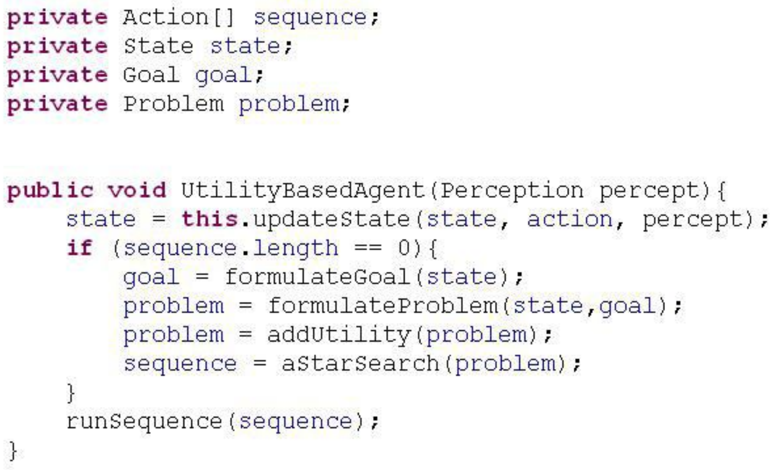
```
private Action[] sequence;
private State state;
private Goal goal;
private Problem problem;

public void GoalBasedAgent(Perception percept){
    state = this.updateState(state, action, percept);
    if (sequence.length == 0){
        goal = formulateGoal(state);
        problem = formulateProblem(state,goal);
        sequence = search(problem);
    }
    runSequence(sequence);
}
```

Figure 27 Code Example Goal-Based Agent, adapted from (5 p. 61)

Goal-based agents are capable of using planning, which is a capability that gives the agent the means to find a sequence of actions in order to fulfill the goal. Figure 27 shows an example java code that illustrates the typical problem-solving agent using a goal state. The problem is formulated as the problem space between the current state and the goal state. The sequence of actions is given by the search through the problem space. The problem of being able to evaluate the solution is missing in this type of agent program.

3.3.2.4 Utility-Based Agents

The agent presented in Figure 28 is basically equal to the one presented for the goal-based agent example. The only difference is that the problem is added a utility value and then the A*search is performed on the problem. The A*search minimizes the total estimated solution cost based on a cost function. The cost function uses the utilities added to the problem when calculating the solution. Adding the utility function gives the agent a capability to evaluate solutions and choose amongst them, but it still cannot learn from its experiences or from making observations (5 p. 51).



```
private Action[] sequence;
private State state;
private Goal goal;
private Problem problem;


public void UtilityBasedAgent(Perception percept){
    state = this.updateState(state, action, percept);
    if (sequence.length == 0){
        goal = formulateGoal(state);
        problem = formulateProblem(state,goal);
        problem = addUtility(problem);
        sequence = aStarSearch(problem);
    }
    runSequence(sequence);
}
```

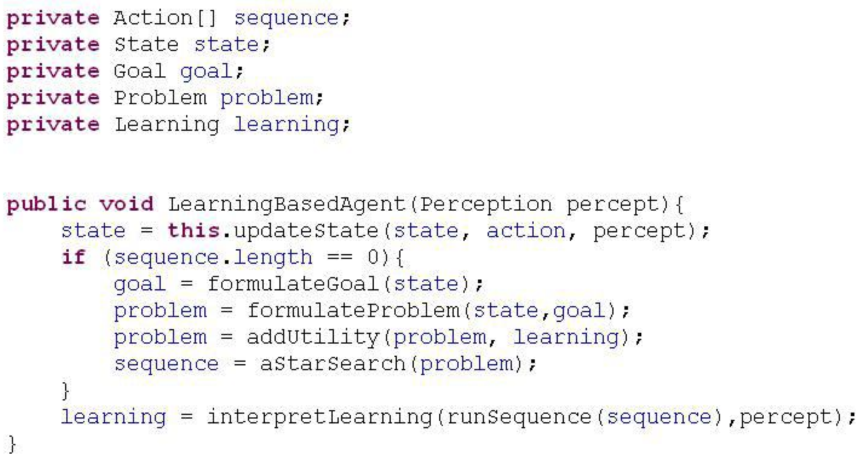
Figure 28 Code Example Utility-Based Agent[4]

3.3.2.5 Learning Agents

The learning agent is a type of agent that can both learn from previous decisions or observations of desired behavior and adapt to it. The agent has a learning element that can suggest improvements on performance, based on an evaluation or critique on past and present measurements of performance. Learning is in this case a result of feedback on past and current experiences with a problem space (5 pp. 51-52).

The agent's ability to distinguish between good and a bad solution suggests that the agent has exploratory capabilities that introduce informative experiences to the agent learning element. The result could be formed as actions that perform tweaking of the utility function itself or the finding of a new utility function suitable for a special type of problems. The learning element operates under a learning strategy or regime that describes how the agent learns (5 pp. 52-54). Examples are reinforcement learning (behaviorism) and learning from observations (socialization).

[4] The figure is a further development of Figure 27.



```
private Action[] sequence;
private State state;
private Goal goal;
private Problem problem;
private Learning learning;


public void LearningBasedAgent(Perception percept){
    state = this.updateState(state, action, percept);
    if (sequence.length == 0){
        goal = formulateGoal(state);
        problem = formulateProblem(state,goal);
        problem = addUtility(problem, learning);
        sequence = aStarSearch(problem);
    }
    learning = interpretLearning(runSequence(sequence),percept);
}
```

Figure 29 Code Example Learning Agent[5]

Figure 29 shows an example of how calculation of utility values can be influenced by learned experiences and how learning can be the result or feedback from the running of action sequences and interpretation of perceptions. Figure 29 gives a highly simplified picture of the problem of implementing learning capabilities to an agent.

3.3.3 Agent Architectures

Architecture is a word describing the design and construction of a system. The notion agent architecture is thereby a description of the components of an agent and the dependencies between them. There exist several definitions of what is meant by agent architecture, but the most common perceptions are that it is a methodology for building agents either in hardware or software with distinct descriptions of internal modules and modes of external interactions (18 pp. 16-103).

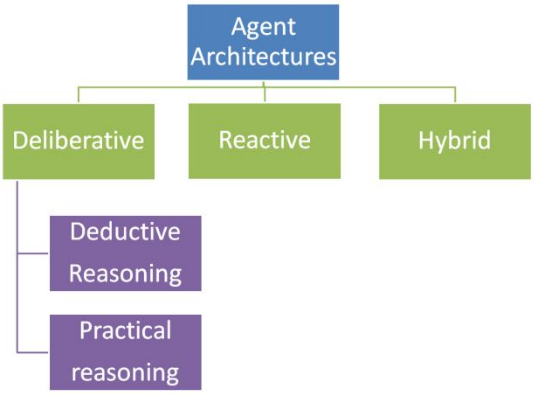


Figure 30 Agent Architecture Taxonomy[6]

Agent architectures can be put into two distinct categories, namely deliberative and reactive agent architectures. Agents with deliberative agent architectures are also known as symbolic reasoning agents. This means that agents of this category contains and maintain a knowledge-base that stores symbolic representations which the agent can manipulate. Reactive architectures on the other hand

[5] The figure is a further development of Figure 28.

[6] The figure is a visualization of agent architectures presented in (18).

generate behavior without any explicit internal symbolic representation. The third kind of agent architecture is a combination of the two and is called hybrid architectures (18 pp. 47-103).

3.3.3.1 Deliberative Agent Architectures

Deductive reasoning agents use logic in their reasoning process and make decisions based on inference in logic. The decision-making problem is a problem of theorem proving. Logic is based on propositions or predicates and there exists many different logical structures. One of the most known are First Order Logic (FOL). The reason for having more than one structure is that in many cases there is a need for using logic where time is a factor (temporal logic) or where one needs to express uncertainty (modal logic). Some of the problems of using logic are how to represent images or video and when the complexity decreases the deliberation process gets more time consuming (18 pp. 48-64).

Practical reasoning is reasoning directed towards action. The role model for practical reasoning agents is the study of how human beings solve their everyday problems. A human being seldom has the time or cognitive capacity to prove all aspects of their actions, thereby facilitating a practical approach to solving a problem and choosing action according to a best-fit algorithm. The approach is to weight conflicting actions up against each other and choose according to the agent's internal convictions or goals. The process is twofold, meaning that the agent has to deliberate on available options and use means-ends reasoning to decide which option to choose (18 pp. 66-86).

Two examples of agent architectures that use a practical approach are the STRIPS (Stanford Research Institute Problem Solver) and BDI (Belief, Desire and Intention) agent architectures. STRIPS has later also been associated with the formal language developed for the input to this planner (5 pp. 378-408). STRIPS language gives a receipt on how to formalize descriptions of states. The BDI architecture is closer described in chapter 4.7.

3.3.3.2 Reactive Agent Architectures

Reactive agent architecture is the result of trying to overcome the problems of operating in time-constrained environments. These agents react on changes to the environment without explicitly reason about it. Two examples of reactive architectures are the *subsumption architecture* of Brooks and the *agent network architecture* by Maes (18 pp. 90-96).

The subsumption architecture introduces a new way of looking at intelligence. This view point on AI claims that intelligence is an emergent property of certain complex systems and that intelligence is undisputedly connected to the real world and not disembodied. Instead of using the traditional decomposition in functional modules Brooks introduce an experimental driven design where one combines layers of behavior or competence. The building blocks of the competence layer are finite-state machines (FSM) named processors that fulfills a specific task. All processors are interconnected in a network of wires. The coordination mechanism between processors is either the suppression of inputs or the inhibition of outputs. By adding a layer you add new behavior or competence (19).

The agent network architecture is conceptually different in the sense that you introduce goals. Instead of processors Maes introduce the concept of competence modules. Each module represents an action described with variables containing a list of preconditions, two lists that either add or subtract effects of the action and an activation level. Except from the activation level the representations of an action has a close resemblance to the STRIPS language (20).

The modules are interlinked in a network through three different types of links: a successor link, a predecessor link and a conflicter link. Communications flow through these links according to specific rules, in which the purpose is to spread activation energy through the network. Activation can be spread as a result of a change in situation, goals or by passing activation energy from one module to another. Inhibition is achieved through protection of achieved goals or by inhibition through a conflicter link. Inhibition is in most aspects achieved by reducing the activation energy (20).

Actions are selected when all preconditions are met and the activation energy of the module is above a given threshold. If more than one module is eligible for action selection the module with the highest activation score is chosen (20).

3.3.3.3 Hybrid Agent Architectures

By combining deliberative and reactive agent architectures it is possible to build a more dynamic agent. Symbolic AI provides capabilities like planning and learning, while the reactive AI handles events that are under time-constraint or does not need complex reasoning, but when combining two different architectural approaches into one there is one question that needs to be answered. How will the agent control and coordinate the two components? The answer is to introduce a layered architecture with control structures (18 pp. 97-102). Examples of layered architectures are TouringMachines (not to be confused with Turing machine) (21), Contention Scheduling (22) and InteRRaP (23).

3.3.4 Building Agents

Building agents is a task that demands an analysis of the tasks or actions the agent is going to perform. There are many different techniques which are used to accomplish the desired result. This sub-chapter will present some techniques that are often used in agent building. The topics discussed are problem-solving, knowledge representation, planning, decision-making and learning. One thing to keep in mind when choosing technique is the definition of the task environment in question.

Creating agents that maintain an internal state and an internal representation of the world is a challenging task. There are three main problems that one needs to overcome before the agent can be implemented.

- **Transduction problem** - the problem of translating the world in the form of perceptions to accurate and adequate symbolic description. In addition the description must be available to the agent within a certain time frame for it to be useful. If the agent is controlling a car, the road block must be identified before the car hits it (18 p. 48). The transduction problem is the problem of translating real-world information into the model.
- **Representation problem** - the problem of how to represent information about real-world entities. The state of the world must be represented in such a way that the agent is able to

reason about it. Creating such knowledge structures is called ontology or knowledge engineering. The representation problem is the problem of making the model (18 p. 48).

- **Reasoning problem** - the problem of how to reason with the information available to the agent. The reasoning problem is the problem of using the model (18 p. 48).

3.3.4.1 The Agent Environment

The agent environment is important when it comes to choosing strategies for implementing the agent. If the operating environment of the agent is complex then the agent will also be complex. The operating environment is the sub-elements of the world that the agent needs to reason about. The analysis of the operating environment is therefore one of the first tasks when one wish to create an agent. There are some general properties of agent environments that have been identified and they are represented in Table 2.

Property	Description
Fully Observable	The state of the environment is known to the agent.
Partially Observable	Only some aspects of the environment state are available to the agent.
Deterministic	The next state of the environment is determined by the current state of the environment.
Stochastic	The next state of the environment is uncertain.
Episodic	The experience of the agent is divided into atomic episodes. One episode is a cycle of perceiving and then performing a single action. The agent's action in one episode is independent of the action in another episode.
Sequential	The current decision affects all future decisions. There is a dependency between actions.
Static	The environment is not changing while the agent is deliberating.
Dynamic	The environment is changing while the agent is deliberating.
Discrete	The number of states is finite, meaning that there exist a finite number of situations. Example a chess game.
Continuous	The number of states is not clear, and one situation seems different from another. Example driving a car.
Single Agent	Only one agent is operating in the environment.
Multiple Agents	More than one agent is operating in the environment.

Table 2 Properties of an Agent Environment, from (5 pp. 40-44)

The property of the environment must be seen from the view point of the agent. The environment could in reality be deterministic, but if it is partially observable to the agent it could very well seem to be stochastic.

3.3.4.2 Problem-Solving with Search

Search is a technique for identifying solutions in a problem space. The search types can be categorized according to the information flow available to the agent. Uninformed algorithms are only given the problem definition while informed algorithms have additional information about where to search. Adversarial search on the other hand is search problems arising on the account of multi-agent environments. For both informed and adversarial search it is normal to use utility functions which provide supplemental information about the problem space. An adversarial search space is also known as a game, which means that it is closely connected to game theory. Table 3 gives an overview of search techniques.

Uninformed Search	Informed Search	Adversarial Search
Breadth-first search (BFS) Depth-first search (DFS) Depth-limited search Iterative Deepening Bidirectional search	Best-first search Greedy Best-first search A*search Iterative deepening A* search (IAD*) Memory Bounded A* search (MA*) Recursive best-first search (RBFS) Simplified MA* (SMA*) Hill-climbing search Local Beam search Simulated annealing search Genetic algorithms (GA) Linear programming	Search mini-max algorithm Alpha-Beta pruning Cutting-off search Expect mini-max

Table 3 Search Techniques, abstract from (5 pp. 59-189)

Search is used to deal with many problems that the agent needs to solve. Well-known examples are the constraint satisfaction problem (CSP) (5 pp. 137-160), the planning problem (5 pp. 375-416) and management of events and actions through scheduling (5 pp. 417-461).

3.3.4.3 Knowledge Representation

The agents need information in order to be able to reason about its surroundings. The information needs to be categorized and given meaning in order to derive new knowledge about the world. This is what symbolic AI has given us through the notion of logics. An agent that relies on symbolic representation of the world contains a storage facility that constitutes its knowledge base (KB). The symbolic language used is also known as a knowledge representation language (5 pp. 320-374).

There exist several types of logics to deal with different problem areas. One of the simplest forms of logic is propositional logic and its successor First Order Logic (FOL). An extension to FOL is called second-order-logic which makes it possible to express upper bound properties, express finite domains and countable cardinalities. Time properties are introduced in temporal logic and in modal logic it is possible to express properties of uncertainty. Reasoning with logic is based on laws of inference and a complete inference algorithm constitutes a resolution (5 pp. 194-319).

The process of connecting meaning to information is done through knowledge engineering or in a grander scale by ontology engineering. Making a complete description of the world is not feasible, but by defining the agent it is possible to derive a set of knowledge representations that the agent can base its reasoning around. What ontology engineering does is that it is an organization process that categorizes objects into taxonomies. This is applicable to both abstract and concrete objects. Abstract objects are often attended with mental constructions, while concrete objects are defined as objects that represent physical constructions (5 pp. 320-374).

Knowledge-based agents define a set of actions, situations and events that it can reason about. Situation calculus is based on the notion of how an action changes the situational state. Two often encountered problems in situation calculus are the frame problem and the qualification problem. The frame problem is the problem of representing all the situations that stay the same, while the latter is the problem of making sure that all necessary conditions, that ensure the success of an action, have been identified and specified. Mental events and objects are usually represented through theories of beliefs. A belief is not a fact, but a picture of a fact that the agent maintains (5 p. 329).

An agent maintains a set of beliefs of the world and often there are connected uncertainties to these beliefs. Techniques for handling uncertainty in knowledge are shown in Table 4.

Technique	Description
Basic probability	Use of statistical probability distributions.
Joint distributions	Probabilistic inference and marginalization.
Bayesian Networks	Represent dependencies between variables.
Dempster-Shafer Theory	Representing distinction between uncertainty and ignorance.
Fuzzy sets and fuzzy logic	Approximation techniques.
Hidden Markov Models (HMM)	Temporal probabilistic reasoning using a single discrete random variable.
Kalman filters	The identification of objects in noisy environments.
Dynamic Bayesian Networks (DBN)	Bayesian Networks that represent a temporal probability model.

Table 4 Handling Uncertainty, abstract from (5 pp. 462-583)

3.3.4.4 The Planning Problem

Planning is defined as the task of deriving a sequence of actions leading from the current state to the goal state. Classical planning environments are fully observable, deterministic, finite, static and discrete. These kinds of problems are usually solved with search techniques, which have been presented earlier in this chapter.

The planning problem is extended when the environment becomes dynamic. In a dynamic environment the environmental state changes with time, which leads to time scheduling of tasks. In addition the access to resources can be limited. There exist a number of scheduling algorithms which can be used. Scheduling has been an area of interest for operative system engineers for years. Examples are earliest-deadline-first scheduling (EDF) and fixed-priority scheduling (FPS). Other methods for planning are hierarchical task network planning, conditional planning, continuous planning and multi-agent planning (5 pp. 417-461).

3.3.4.5 Decision-Making

Decisions made by agents are in most aspects taken by evaluating the available solutions up against each other. Probability theory describes to which degree an agent should rely on the belief that it has in its KB, while the evaluation of available solutions is based on a utility function. The utility function states to which degree the choice is favorable to the agent. The basis for utility functions is to be found in utility theory. Two basic utility principles are expected utility (EU) and maximum expected utility (MEU). The combination of probability theory and utility theory constitute decision theory. Examples of utility functions are money and risk evaluation. The two examples are based on a single attribute, while in contrast some utility functions are dependent on multiple attributes (5 pp. 584-648). Table 5 gives an overview of decision-making techniques.

Name	Description
Decision Network	Also known as influence diagram. Combines Bayesian networks with additional node types for actions and utilities.
Information Value	Decision is based on which information is considered to most valuable.
Expert System	A system that use statistical and expert information to estimate best-fit decisions. Possess the ability to assess the information value and sense changes in probability and utilities.
Markovian Decision Process (MDP)	Specification of a sequential decision problem for a fully observable environment with a Markovian transition model and additive rewards.
Value Iteration (Bellman equations)	Utility calculation for all states, and selects optimal action in each state.
Partially observable MPD (POMPD)	Specification of a sequential decision problem for a partially observable environment with a Markovian transition model and additive rewards.

Table 5 Decision-making Techniques, abstract from (5 pp. 584-648)

Multi-agent environments base decisions on these techniques and use aspects from game theory to choose decision strategies. One of the most profound problems of decision-making is to identify or create the utility function perceived to be correct for the environment the agent is going to operate in.

3.3.4.6 Learning

Learning is an exceedingly important component to incorporate into an agent framework. Learning is dependent on a performance element that monitors and decides which actions to take and a learning element that modifies the performance element with the intent to increase overall performance (5 pp. 51-54).

We can divide the area of agent learning according to the learning methods. Inductive learning is learning by observations, statistical learning methods create hypotheses and reinforcement learning introduces a system of rewards. Table 6 gives a short overview of learning techniques for agents (5 pp. 650-789).

Learning method	Description
Pure inductive inference	Given collection f return function h so that h ≈ f
Learning decision Trees	Decision tree induction.
Ensemble learning	Select collection of hypotheses and combine their predictions. Make a new decision tree induction.
Bayesian learning	Calculates probability of hypothesis and give predictions on the basis of those data.
Maximum-likelihood learning	Find numerically parameters for probability model.
Neural networks	Nodes resemble neurons; learning is based on creating connections.
Kernel functions	Also know as support vector machines (SVM). A high dimensional weight space constructed of multi-layered networks, which can represent complex nonlinear functions.
Passive reinforcement learning	The agent's policy is fixed and the agent learns the utilities of states in the environment. Also see utility function.
Active reinforcement learning	Exploration of state space to find the reward.
Q-Learning	Uses a function approximation to avoid one input value for every state in the problem space.

Table 6 Learning Methods, abstract from (5 pp. 650-789)

3.4 Distributed Artificial Intelligence

There are two distinct directions within distributed artificial intelligence (DAI), namely Multi-Agent Systems (MAS) and Distributed Problem Solving (DPS). MAS concentrates on the behavior of a collection of autonomous agents trying to solve a problem, while a DPS concentrates on how different modules can divide tasks among themselves to solve a given problem. The master thesis focuses on MAS.

Definition 13 Multi-Agent System

MASs contain a number of agents which interact with one another through communication. The agents are able to act in an environment, where each agent will act upon or influence different parts of the environment (18 p. 105).

3.4.1 Agent Interaction

Interaction is series of events where agents are in direct or indirect contact with each other. Indirect contact can be realized through other agents or through the environment. The contact can either be intended or unintended. Before interaction can be a reality the agents need capabilities, and the environment need properties, that facilitate interaction. An example of an agent capability is communication. Properties of the environment could be structures that introduce physical impacts or restricts access to resources. Interaction is a result of creating temporal or definitive relationships between agents (18 pp. 105-111).

There are different strategies for overcoming the problem of creating agent interactions. To be able to make decisions under uncertainty when interacting with other agents one uses game theory. For reaching agreements one uses other forms of mechanisms. Typical mechanisms for reaching agreements are auctions and the use of domain theory.

Game theory makes some assumptions about the agents, which are that they are rational and that they consider the other agents' decisions. Game theory sets up and environment for creating strategic games, which is defined by the number of agents, number of actions and utility functions. An important assumption is that all agents are aware of the game definition. The game definition is common knowledge (18 pp. 111-119).

Auctions are used to allocate goods to agents based on bids. You have one agent being the auctioneer and a group of agents being the bidders. Types of auctions are English, Dutch, first-price sealed-bid and Vickrey auctions. The difference lies in the method of passing a bid and the determination of price. Auctions are susceptible to lies and collusion (18 pp. 129-137).

Domain theory identifies the main object for an agent to deliberate on. Examples of different domains are task oriented domains, state oriented domains and worth oriented domains. A task oriented domain (TOD) is defined by an environment where the agents have non-conflicting jobs to perform. To increase efficiency the agents can negotiate to redistribute the task for mutual benefit. The job of the agent is then to convince the other agent through compelling arguments to redistribute the task (18 pp. 137-146).

3.4.2 Agent Communication

In an environment where agents interact, coordination of actions is often necessary. A prerequisite to coordination is being able to communicate. Communication is important in situations where one needs to resolve a conflict or negotiate a course of action. Agents that work together as a team need to coordinate their actions in order to facilitate cooperation (18 pp. 163-164).

Agents that operate in a software environment might look at communication as an action. Actions are operations that change the world state and when an agent communicates it changes the beliefs of the agent it is communicating with, ultimately changing the intentions of the other agent. Communication between software agents is performed by sending formatted messages using already well known network infrastructures (18 pp. 163-164).

Communication requires a set of rules that both receiver and sender have agreed upon in advance. If two communicating parties are to understand each other they both need to have the same understanding of the content in the messages and know the rules for conducting the conversation. This has lead to a formalization of agent communication languages and conversation protocols. The agent communication structure follows the same layered concept as any other networked communication structure (18 pp. 164-168).

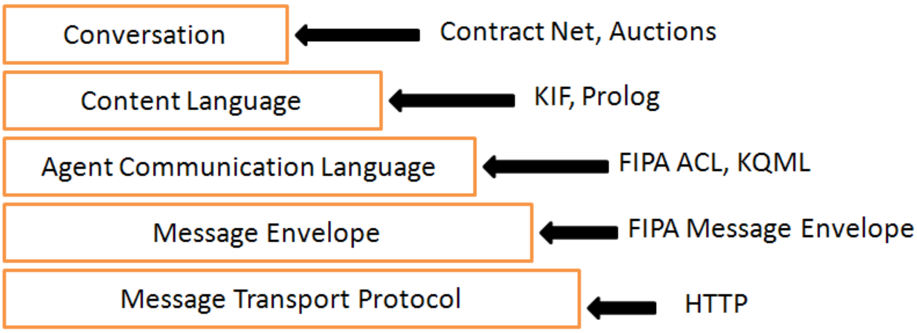


Figure 31 Layered Agent Communication Model, from (24)

The conversation layer describes the formal exchange of messages in a conversation and constitutes the interaction layer. The communication layer describes the communication acts, content and ontology (Figure 31). The communication acts are described in Agent Communication Languages (ACL), which define a framework for messages sent between agents. The content can be specified in a specific format like for instance XML. The content is often described using a specified ontology (chapter 3.3.4.3). The Foundation for Intelligent Physical Agents (FIPA) maintains standards for agent communication through the FIPA ACL standard (18 pp. 175-183).

Chapter 4 Human Behavior Representation

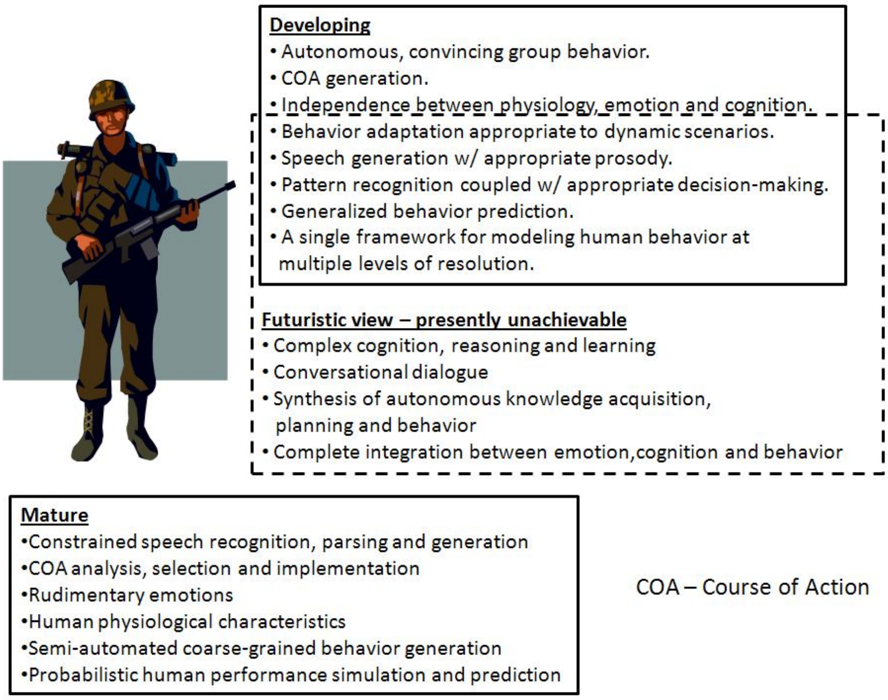
The purpose of this chapter is threefold: primarily it explains what is meant with the term HBR and how it is related to operational psychology and AI. Secondly it identifies properties of HBR by presenting conceptual models of human cognition based on theory presented in chapter 3.2. Thirdly it presents a selection of architectures that imitate cognitive processes within humans. These architectures use techniques and mechanisms from AI (chapter 3.3 and 3.4) in their structures. This master thesis makes a distinction between the conceptual structures of agent architectures and their actual implementation. This chapter presents the concepts of the agent architectures while chapter 5.4 takes a closer look at the implementation which constitutes the agent framework.

4.1 Introduction

A computational model is a model that is implemented in software or hardware or both with the purpose of playing out represented behavior in a real or a simulated world environment. Human Behavior Representation (HBR) is the term of representing human behavior by computational models. The US Department of Defense (DoD) refer to HBR as the modeling of human behavior or performance that needs to be represented in military simulations (3 pp. 10-14). The HBR community has adopted the definition given by Pew and Mavor (Definition 14).

Definition 14 Human Behavior Representation

HBR refers to computer-based models that mimic either the behavior of a single human or the collective actions of a team of humans (3 p. 11).



Developing
- Autonomous, convincing group behavior.
- COA generation.
- Independence between physiology, emotion and cognition.
- Behavior adaptation appropriate to dynamic scenarios.
- Speech generation w/ appropriate prosody.
- Pattern recognition coupled w/ appropriate decision-making.
- Generalized behavior prediction.
- A single framework for modeling human behavior at multiple levels of resolution.

Futuristic view – presently unachievable
- Complex cognition, reasoning and learning
- Conversational dialogue
- Synthesis of autonomous knowledge acquisition, planning and behavior
- Complete integration between emotion,cognition and behavior

Mature
- Constrained speech recognition, parsing and generation
- COA analysis, selection and implementation
- Rudimentary emotions
- Human physiological characteristics
- Semi-automated coarse-grained behavior generation
- Probabilistic human performance simulation and prediction

COA – Course of Action

Figure 32 HBR Characteristics and Development Status, constructed from (25)

The scope of HBR embraces most aspects of AI and operational psychology in its attempt to recreate the artificial human being or organization of human beings (Figure 32). The fields of science involved in such an undertaking are broad and the amount of research is extensive. Characteristics of HBR cover areas from speech and image recognition to mental processes and social interaction (25).

The focus area for this chapter will be to cover those aspects of HBR which is needed to complement the simulation frameworks (chapter 5.2) already in use by FFI. Existing frameworks cover the notion of artificial life (AL) and need a component that adds higher-level functioning through AI (Figure 33).

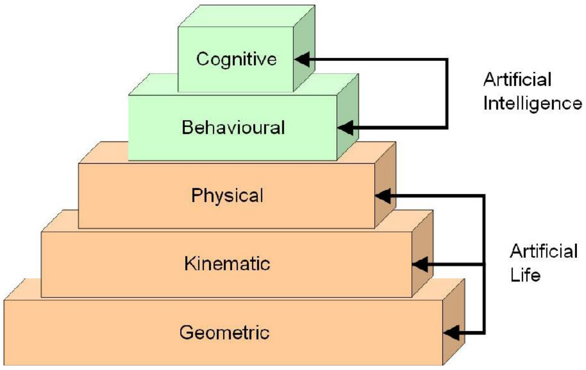


Figure 33 Categories of HBR, from (26)

Restricting the notion of HBR to the higher-level functioning domain makes it possible to categorize the term of HBR through operational psychology. The theories of operational psychology look at different aspects of the individual, groups and internal and external group interactions. A model therefore needs to cover cognitional, personal and social aspects of either an individual or a group of individuals. The abstraction or resolution level of the simulation influences the nature of the model. If you are going to simulate a crowd of people one needs to evaluate if one wants to simulate each and every individual or groups of individuals within the crowd or the whole crowd as one simulation entity.

There exist many models that cover aspects of human cognition or human factors. A model that try to simulate a complete range of human mental processes from perception to abstract problem-solving and decision-making often is described as a Unified Theory of Cognition (UTC), while models that describe fixed or small ranges of covert or overt human behavior often are referenced to as performance models (27).

Alternatives to cognitive modeling are the use of genetic programming (GP) and BDI models. GP models base their computations on theories of evolution. Solutions are produced on the basis of mutations where solutions with the best performance survive to the next iteration. The area of usage for evolutionary computations is at the moment too independent from human behavior constraints to be considered (27). Even though BDI architectures are not based on human cognition, but rather is and effective way of doing practical reasoning, the use of the architecture has increased when it comes to military simulations. For this reason BDI architectures are highly relevant.

4.2 Representing Human Behavior in Synthetic Environments

The Norwegian military needs a framework that can ease the introduction of intelligent behavior in their existing simulation frameworks. These frameworks provide a synthetic environment (SE) for performing simulated military operations, but have limited capabilities for facilitating simulation of human behavior. Simulations are regarded as a cost effective and time-saving method for training personnel on all levels of the military organization (MO). This includes training of leaders, training on joint operations and training of support functions (3 pp. 33-38).

The SE is the operating environment of intelligent software agents that contains the HBR structure. A SE will be the playing ground for injected real forces, artificially induced synthetic and semi-automated forces (28 pp. 2-4). Figure 34 describes the relationship between SE and the containment unit of the HBR functionality. The data compilation unit is a layer that gathers data about the environment and creates representations of for instance terrain advantages and risk areas for use in the tactical deliberation process of the agent. The agent framework can also be a multi-agent system (MAS), which can resemble an aggregation of entities or groups.

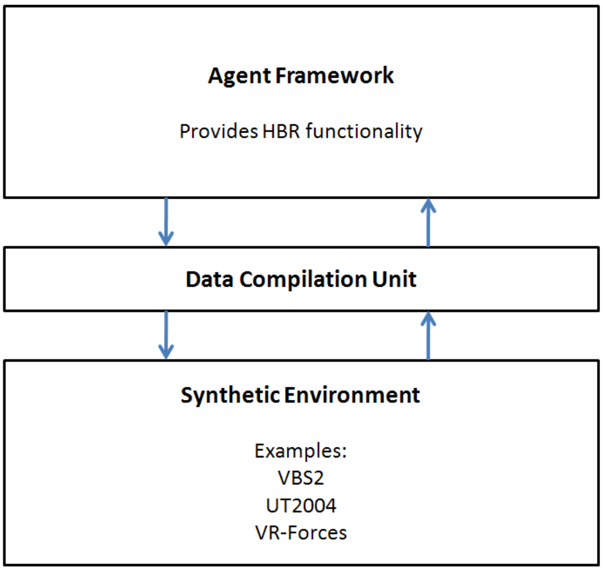


Figure 34 Agent Situated HBR

The agent framework constitutes the implementation of HBR, which is why one needs to distinguish between the model of cognition, which provides the cognitive architecture, and the model of knowledge which one implements within the cognitive architecture. Integrated cognitive architectures are modeled on the background of two different approaches, namely rule-based and connectionist cognitive modeling, the latter also known as artificial neural networks (ANN) (27).

To conclude a SE constitutes the simulated physical and low-level cognitive components of HBR, while the agent constitutes the high-level cognitive component of HBR. The physical components contain the states of the simulated object or platform within the SE, while the behavioural aspects determine the actual actions the component is going to perform. The SE is in most respects covered by environment simulation frameworks for computer generated forces (CGF) like for instance Virtual Battle Space 2 (chapter 5.2).

4.3 Identification of HBR properties

HBR can be broken down into different focus areas based on operational psychology. If we imagine the internal structure of the mind as a black box where stimuli are the input and the responses are the output, we can incorporate the biological, cognitive, personality and group theories into a structure which constitutes the computational basis for observed behavior. This chapter develops a conceptual model from operational psychology to identify important properties of HBR. The properties which are identified will constitute the basis for the evaluation of the presented cognitive architectures in chapter 4.6.

The starting point for developing a conceptual model is to identify the conceptual components that the model will contain. The general model will be concretized through an analysis of structural and functional elements in the human cognitive process. Every model can be looked upon as a module within the mental framework of the human mind.

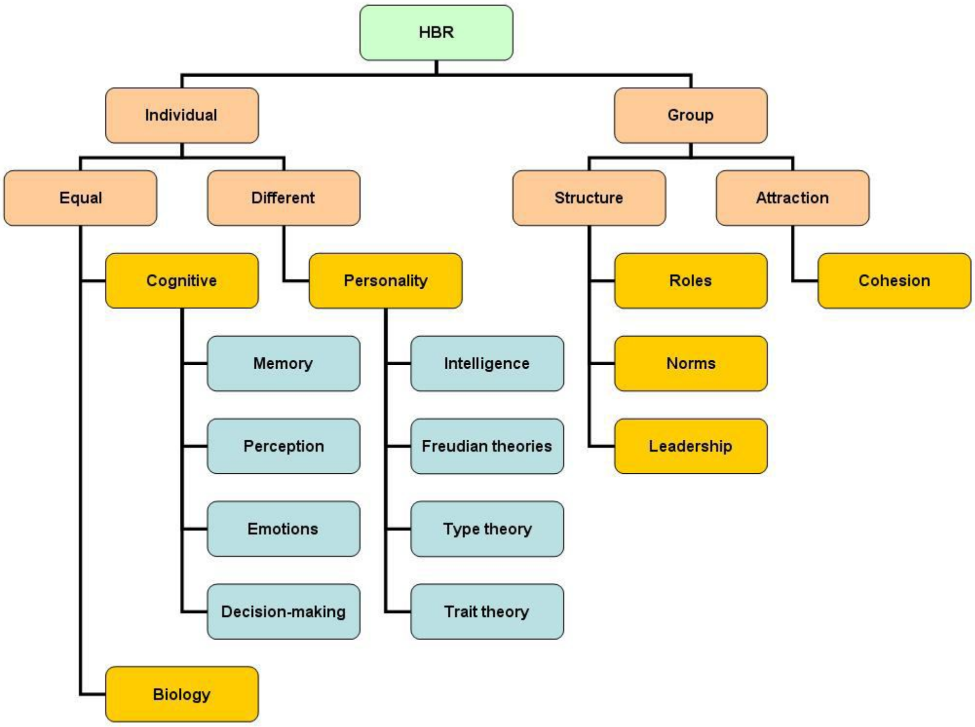


Figure 35 HBR Related to Operational Psychology

Every module needs to identify the contents of the black box, the sources of inputs and the destination of the outputs. In many cases the operational modus of the module is determined by a set of control data, which implies a third line into the black box. Each module adds or subtracts information from other modules and they are all interlinked and dependent on each other to process information of different forms and categories. Cognitive control is perceived to be exercised by a meta-reasoning process. This process relies on input from the majority of the identified modules.

This behavior analysis is inspired from the ideas of Prof. Barry G. Silverman who has initiated an investigation into performance moderator functions (26; 29; 30; 31; 32; 33). Support material has been found in (13) , (14) and (15).

4.3.1 The Biological Module

The biological module contains information about the state of the physical body. Information can either be received through the perception module or be the result of an internal generated event. The state of the physical body is influenced by emotional distress, which means that emotions trigger physiological responses. The type of information received from the perception module depends on how the biological module is organized. Status of vital internal organs is given by the PNS to the CNS. If the perception module is extended to include the PNS information then it would be perceived through the perception module else it would be modeled internally in the biological module.

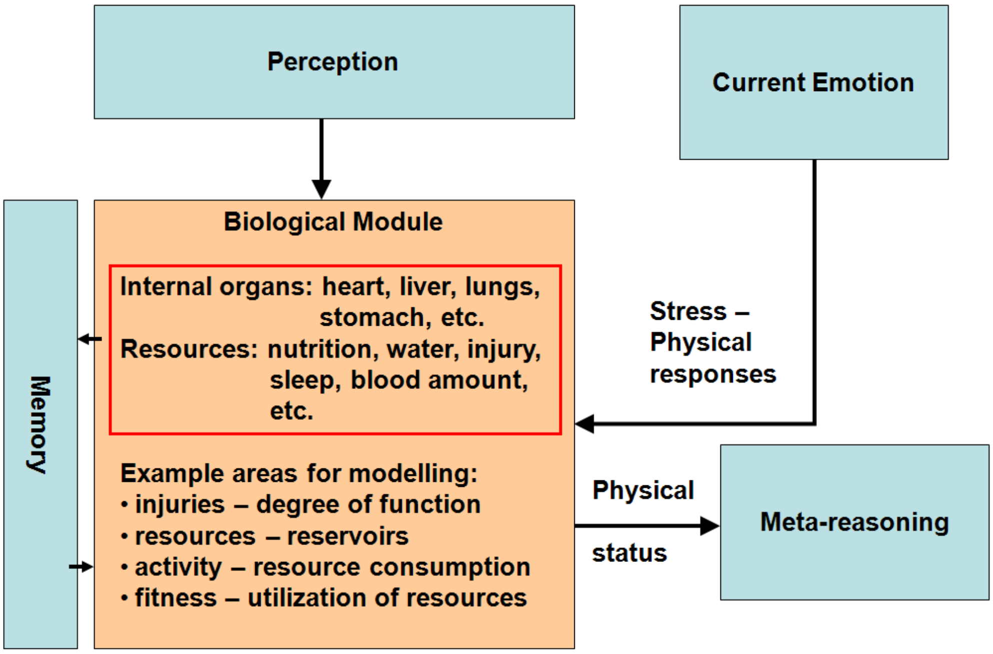

Perception
Current Emotion
Memory
Biological Module
Internal organs: heart, liver, lungs, stomach, etc.
Resources: nutrition, water, injury, sleep, blood amount, etc.
Example areas for modelling:
• injuries – degree of function
• resources – reservoirs
• activity – resource consumption
• fitness – utilization of resources
Stress – Physical responses
Physical status
Meta-reasoning

Figure 36 The Biological Module

Type of information that is relevant to be maintained in a biological module is the status of vital organs, the status of resources and damage assessment (Figure 36). Based on certain conditions the biological module should generate physical needs. Needs are used for meta-reasoning in the reasoning module. Information is stored in memory and shared with other modules through shared memory space (31).

The biological module needs to contain several models of bodily functions, which functions to implement depends on the purpose of the simulation. In a military simulation you might get shot by an enemy combatant. This inflicts pain and reduces mobility which again causes fear and anxiety. The biological module should include or support a damage model that includes impact on different vital organs. Resources are elements that make it possible to operate as a human being, for instance physical needs like food, water and sleep. Resources are consumed by the organism and needs to be refilled. The conscious mind does often not have full control over the status of the body. One often does not know that something is wrong before a symptom appears. The symptom can be a result of physically induced stress or mentally induced stress or the result of an illness. This can be modeled through activation values in resource reservoirs (30).

Figure 36 clearly shows that the input is taken from the perception module. Perceived information can also be conditionally fetched from memory. Injuries induce strong memories, which means that experiences that involves injuries or other physical pain is stored into memory. You are likely to remember the times you where ill and can use those memories to recognize similar symptoms (sequential memory chapter 3.2.4).

Control information to the biological module is in this case reduced to the current emotion. The current emotion would induce physiological responses and is therefore important for the internal functional models of the biological module. Emotions can change the physiological status of the body. The output of the module is physiological status which is sent to the meta-reasoning module. The physiological status influences the prioritization of tasks to deal with. When you get hungry or thirsty you start to spend more time on finding and preparing food than you normally would.

Biological models can in many aspects be seen as a collection of performance moderator functions that influence the thought processes within the human mind. Important properties to a biological module are listed in Table 7.

Property	**Description**
Resource model	Resources are reservoirs that are consumed by the simulated individual. Examples of such reservoirs are energy level, fluid level, sleep level etc. Each resource model constitutes a performance model that affects both physical and mental performance.
Activity model	Activity is a measurement of physical and mental load. The load affects the consumption rates of identified resources. Mental load will in itself influence the mental processing.
Damage model	Registration of injuries and to which degree they affect both physical and mental abilities constitute a damage model.
Fitness model	A fitness model describes the individual's ability to utilize available resources. This model would be a source for implementing individual differences into the biological model.
Environmental model	External factors that influence the performance of human beings can be said to be environmental influences. Examples of such influences are heat and noise. These factors will affect the consumption of resources, which is controlled by the fitness model.

Table 7 Biological Module Properties

4.3.2 The Perception Module

Perception is closely connected to memory. Perceived information is stored in either sensory memory, STM or WM depending on which memory perspective one uses, some theories even use all. Perceived impressions are stored in temporal memories for further processing. Most simulations constrict the number of sensors to implement or make simplified models of the human sensatory spectrum. The reason for this is the difficulties of transforming raw sensory data to recognizable representational data formats in a computational model (25).

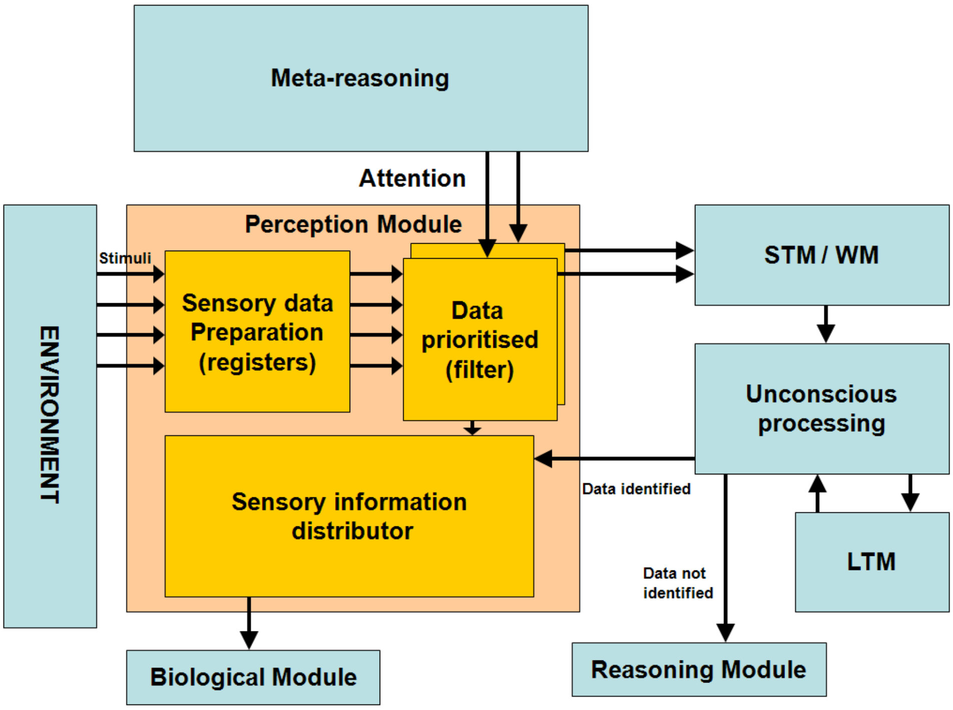


Figure 37 The Perception Module

Figure 37 shows a stream of sensory stimuli flowing into a sensory register that pre-process the perceived impressions. Sensory input is according to most theories images that we have connected certain expectations to when it comes to what they mean. The process of connecting the term and the image is the processes of linking associated memories or concepts to the input data. The identification of the image or sound seems to be automated over time, meaning that the process is passed from the conscious mind to the unconscious mind when the connection between the term and the image seems to have been permanently established (15 pp. 211-283).

Figure 37 shows that data are unconsciously processed and if the data is directly identified it is distributed to the appropriate models and if not there is a need for a constructive approach for identification. If there are no familiar data to relate the image to then one need to extend the process to conscious deliberation and synthesize about what it can mean. Deliberation acquires a process that involves problem solving and decision-making, which is why the figure shows that unidentified data should be handled by the reasoning module.

The senses provide an abundant stream of information to be processed by the mind. It is fair to presume that some information is more important than other information. The information is most likely pre-processed in the filter process. Theories on attention have introduced the notion of a selective filter or bounded resources, where only impressions of a certain value are stored into STM or WM (chapter 3.2.3.1). Figure 37 shows this by introducing a control line from the meta-reasoning module to the perception module.

A model of perception should support the most incorporated views on perception. We can divide into two main categories of properties, namely the perception mode and the perception registers. The registers have the task of transforming sensation into representation. Table 8 gives a list of important properties of perception.

Property	Description
Visual Register	Stores the visual stimuli in a format that is understandable for further processing, included performance moderation.
Auditory Register	Stores the auditory stimuli in a format that is understandable for further processing, included performance moderation.
Tactile Register	Stores the tactile stimuli in a format that is understandable for further processing, included performance moderation.
Olfactory Register	Stores the olfactory stimuli in a format that is understandable for further processing, included performance moderation.
Gustatory Register	Stores the gustatory stimuli in a format that is understandable for further processing, included performance moderation.
Direct perception mode	Perception mode that facilitates direct association of registry representation data to data in long-term memory with performance moderator functions.
Constructive perception mode	Perception mode that facilitates a deliberation process for identifying the perceived image with performance moderator functions.
Sensory filter function	A function that is capable of distinguishing between important and unimportant information on the background of selective attention.

Table 8 Perception Module Properties

4.3.3 The Memory Module

The memory module mostly involves the perception and reasoning modules, but is significant for all modules because all modules need to be able to retrieve information. The contents of the memory module are dependent on which theories that one wishes to emphasize. The module presented in Figure 38 emphasizes the three-store and the WM model. Both models support the notion of LTM as being the foundation of permanent storage of information in some form.

All modules draw information from memory. There are therefore several ways of looking at how the different modules interact with memory, for instance most likely all modules draw information from memory concurrently and in parallel. STM is in this picture a temporal storage facility for information sensed during a short time period. WM can fulfill the need for temporal fetching of memories that needs processing. It is a mixture of information from STM and LTM. Memory is basically not active but passive, since other modules use memory in their processing (15 pp. 149-206).

The most important input to memory is environmental information processed by the perception module. The perception module filters out irrelevant information and pre-processes the raw data into a recognizable format and stores the information in STM, what data to focus on is decided by the meta-reasoning module. Meta-reasoning is important when it comes to the WM, which holds information that is vital for working out an interpretation of bundled data formats. A bundle of data formats means that the WM has simultaneous access to sensory information and permanently stored information in the LTM. Information fetched from LTM is related to the data from external sensors. The contents of the WM are processed by the reasoning module.

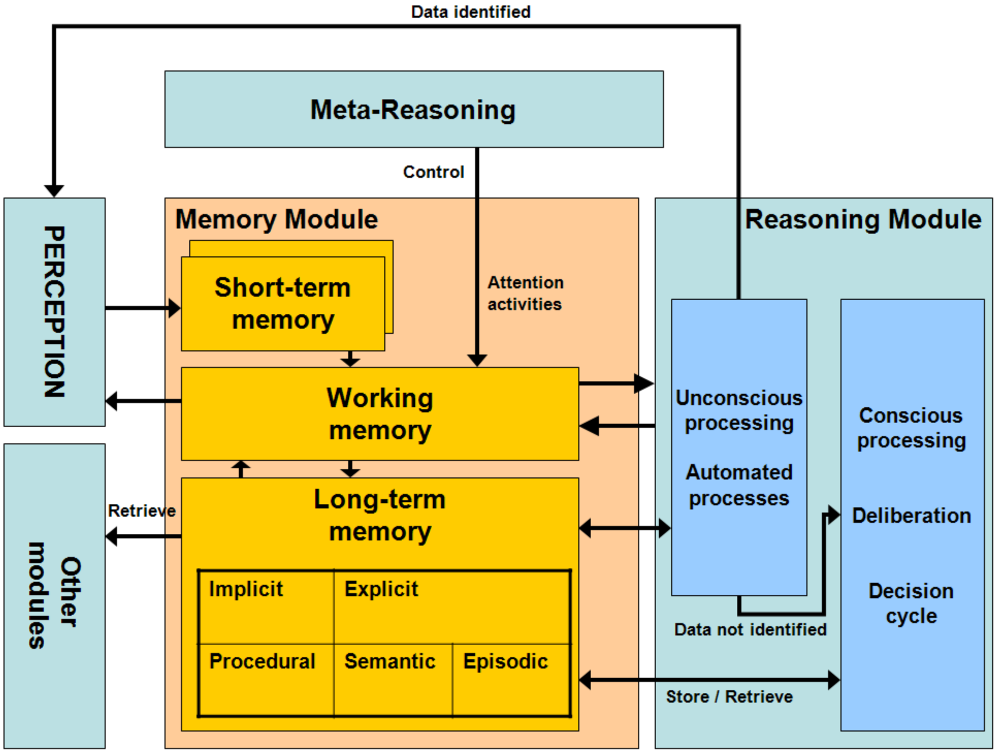


Figure 38 The Memory Module

Information becomes knowledge when it has been processed and structuralized. Knowledge is information which has been added meaning and ultimately creates competence which has the potential to generate action. The integration process of storing new information into stored information is termed consolidation, which is seemed to be controlled by some elementary functions for knowledge categorization. Another way of looking at it is that the cognitive process of incorporating new information can be said to be the process of engineering an individual ontology or knowledge-base (34).

The knowledge stored in LTM is the result of a learning process where new knowledge is obtained by processing sensory stimuli. New knowledge is added to LTM by repeatedly being stored in STM. Memorizing and rehearsals are familiar techniques for remembering images and sounds. Processes of deliberation are more likely to be remembered than automated processes. Events or episodes that generate special feeling or emotions are more likely to stay with you than everyday routines, which is why the conscious processing module has a store and retrieve line to LTM in Figure 38.

Memory contains all experiences and facts, sequential and procedural memories. Other types of memories are declarative memories like semantic and episodic memory. Properties of a memory module are these memory types and performance moderators that for instance describe the probability of retrieving a memory or give a measurement of knowledge importance. Table 9 describes the most important memory properties.

Property	Description
LTM	This property describes the type of long-term storage the model has. It should support both implicit and explicit memory types.
STM	Memory type for storing immediately received sensations and thoughts.
WM	Memory type for storing immediately received sensations and thoughts in addition to recently activated long-term memory knowledge.
Retrieval PMF	To retrieve information from memory one relies on functions that gives the probability of successful retrieval based on degree of association and usefulness. Time plays a role when it comes to forgetting and some memories are harder to activate than others.
Storage PMF	Information is transferred from temporal storage to long-term memory with different speed and method. Depends on which kind of LTM one is trying to access.
Representation	How knowledge is represented in the model.

Table 9 Memory Module Properties

4.3.4 Module for Creating Emotions

Emotions influence many aspects of life. Psychology describes it as being a short-cut heuristic for speeding up the process of finding an appropriate action. It short circuit the normal loop of making decisions and is a foundation for reactive behavior. Emotions are triggered by external events and the individual's mental and physical health. The physical health is given by the biological module while the mental health is given by the personality and social modules in conjunction with the reasoning module (Figure 39). Reasoning and calculated behavior is the result of a conscious evaluation process while reactive behavior is instinctive behavior to certain stimuli and perceptions.

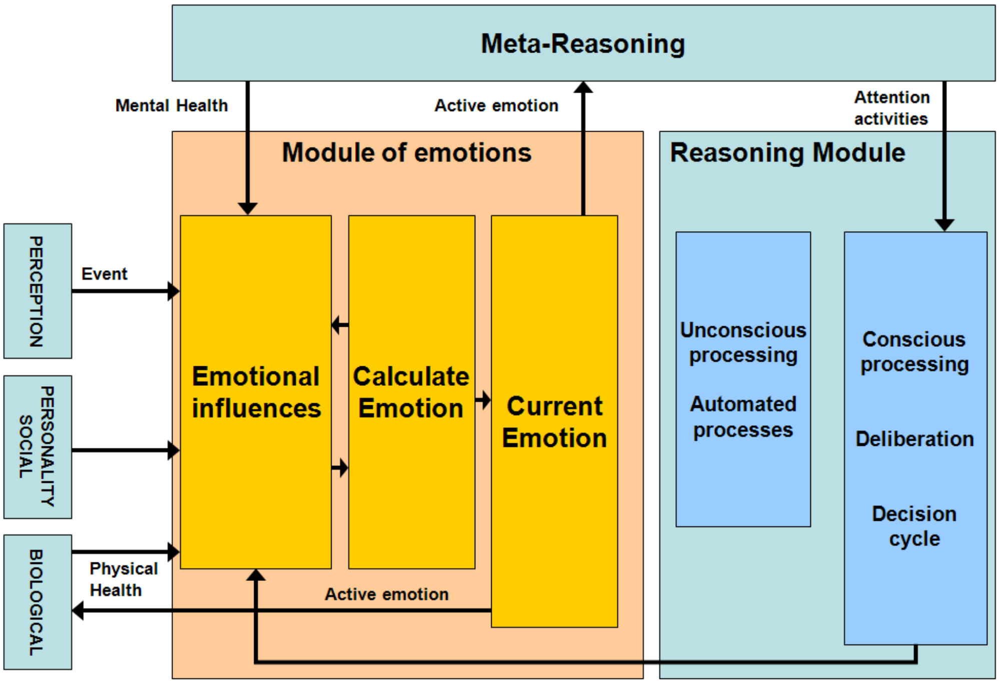


Figure 39 Module of Emotions

Emotions are highly important in the meta-reasoning module. The current emotion is one of the variables that controls where to direct attention and in this aspect creates motivation and intentions.

The degree in which an emotion influences the decision-making process depends on the strength of the emotion. Reactive behavior is a fact when the emotion set aside all conscious processes and as a result the individual is acting instinctively, for instance rescuing a person who is drowning involves avoiding physical contact with the individual, because the rescuer risks being pushed under water by the panicking person. In this sense the reasoning module is influenced by a dynamic shifting emotional spectrum (35). The spectrum of emotions can be divided into general emotional affects and reactive emotions based on the intensity of the emotion. General emotional affects are emotions that one are able to reason about, while the latter are emotions that surpasses common sense and are border lining instinctive.

Properties for a module of emotions are presented in Table 10.

Property	Description
Theory of Emotion	The theoretical foundation of the model. Examples are Cognitive Complexity and Control Theory (CCC), PSI Theory and Desire-Belief Coherence (DEBECO) (35)
Emotion performance function	The foundation of calculating an emotion based on biological stress, personal values and social interaction.

Table 10 Module of Emotions Properties

There are a few architectures that incorporate an emotional model. The micro-PSI architecture and PMFserv incorporates aspects of emotional influences on reasoning. The micro-PSI agent architecture is built around the notion of that emotions are modulations of cognitive and motivational processes (36). This means that emotions can be viewed upon as modes of thinking or as a special form of thinking, and not as being a separate cognitive process.

4.3.4.1 The Reasoning Module

The reasoning module contains important functions like problem-solving and decision-making. By the word reasoning is meant the process of evaluating acquired knowledge in the interest of obtaining a conclusion or solution to a given task, which means that the model contains elements of planning and learning. In this case the reasoning module is divided into meta-reasoning (Definition 15) and a general reasoning module.

Definition 15 Meta-reasoning

Meta-reasoning is the process of reasoning about reasoning itself. It is composed of both the meta-level control of computational activities and the introspective monitoring of reasoning to evaluate and to explain computation. Meta-level control is the ability of an agent to efficiently trade off its resources between object level actions (computations) and ground level actions to maximize the quality of its decisions (37).

Meta-reasoning controls and monitors the processes within the different models. Certain values are used for estimating and calculating priorities of tasks and if there are more than one solution to a problem the meta-reasoning module chooses a course of action that is appropriate. The concept of meta-reasoning is shown in Figure 40.

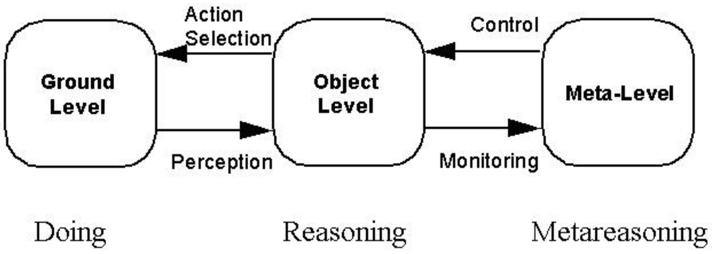


Figure 40 Levels of Reasoning, from (37)

Meta-reasoning is in most aspects a question of creating the motivational force of the module. Motivation is generated mainly through biological, emotional, personal and social needs (Figure 41). The normal human mind is capable of performing a limited number of tasks at the same time and this has lead to the impression of that the mind has a limited set of resources at its disposal. The meta-reasoning module deliberates on the importance of restricting access to resources and make sure that the attention is focused on important tasks and downgrade other tasks. Effective deliberation on motivational factors is dependent on inputs from several modules (26).

The meta-reasoning module can also contain elements that calculate mental induced stress. If the cognitive element is overloaded by information or tasks then the meta-reasoning element should create a measure that quantifies the load factor.

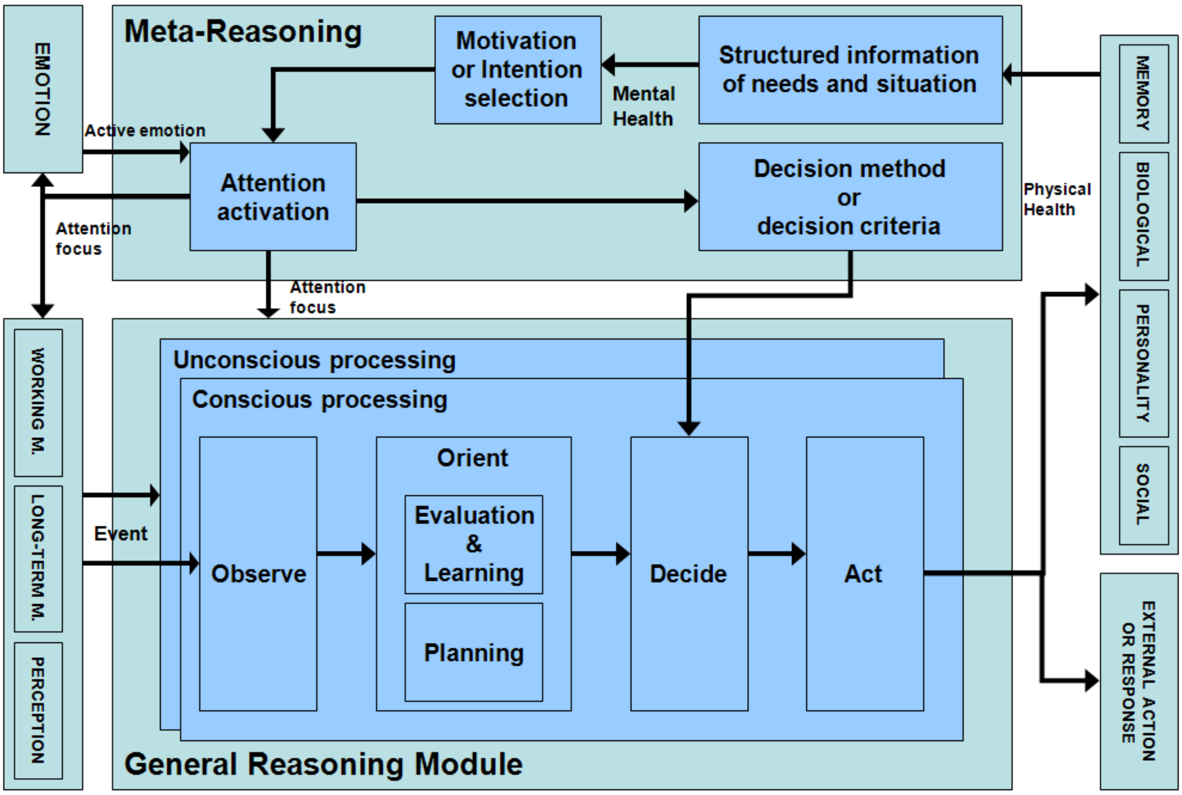


Figure 41 Object and Meta Level Reasoning Modules

The attention or task focus is determined by the current need or needs. The needs can be formed as temporal and long term goals that need to be fulfilled. These goals can be divided into types that describe motivation, task, time pressure etc. The goals that have the highest rating are processed first, meaning that the attention in this case is focused by goal attributes.

General reasoning is in this case build upon the notion of decision cycles. Some known decision cycles are ABCD (acquire, best-fit, choose, direct), OODA (observe, orient, decide, act), SHOR (stimuli, hypothesis, options, response) (26). General reasoning is the result of one such loop. Humans usually are able to process more than one task at the time, so it is reasonable to conclude with multiple resources for making decisions. The OODA-loop was presented in chapter 2.4 and will be the decision cycle used in this section.

The reasoning module covers aspects of problem-solving and decision-making (Figure 41). A problem that needs to be solved can be described in many ways. Usually you know the desired end state and your current state. The end state can be called a goal state of affairs. Strategies for fulfilling the goal state could be either deductive or practical approaches (chapter 3.2.7).The model uses the *observe* module and the *orient* module to define the problem and synthesize a strategy for solving it. If the problem space is too wide the goal needs to be divided into a number of sub-goals that can be worked on individually. If there are several paths to choose from, which is usually the case in real life, the *decision* module decides which action to perform. The *act* module then performs the action (31).

The action can introduce state changes both internally and externally. Internally a reasoning process can change the internal world model of the agent. Internal events can change information that is organized by the biological, emotional, personality and social modules. External events on the other hand are agent expressions directed towards the environment. This can be communication acts or simulated physical acts.

The reasoning module is divided in two levels. The meta-level controls the object-level by evaluating the data in the different modules. The properties of a reasoning engine should cover both meta-level and object-level reasoning. Properties of a reasoning module are presented in Table 11.

Property	**Description**
Problem-solving method	The property describes how problems are handled.
Problem representation	The property describes the representation of problems.
Decision-Making method	The property describes the methods used for making decisions. Examples of decision-making regimes are Naturalistic Decision-Making (NDM) and Recognition Primed Decision-Making (RPD).
Reasoning method	The property describes the approach to reasoning whether it is deductive or inductive. The methods are closely linked to methods used by agents.
Learning mechanisms	The property describes mechanisms that are used for giving the architecture learning capabilities, including performance moderator functions.
Planning mechanisms	The property describes mechanisms that are used to add the capability for reasoning about sequences of acts, including moderator functions.
Serial or Parallel processing	The property describes if the architecture promotes serial or parallel processing of tasks or events.

Table 11 Reasoning Module Properties

4.3.5 The Personality Module

The personality module promotes individually stated variances in behavior. As an individual one has experienced different situations and learned different things than the individuals around you. In addition people are born with certain abilities or tendencies that are biologically determined (chapter 3.2.8). Some people might have larger STM than other people, and because of this they have fewer problems learning theoretical subjects. There exist many models that represent individual variations, which emphasize different aspects of personality theory. Some use biologically determined variances, while others use variances in knowledge or both (28 pp. 17-18).

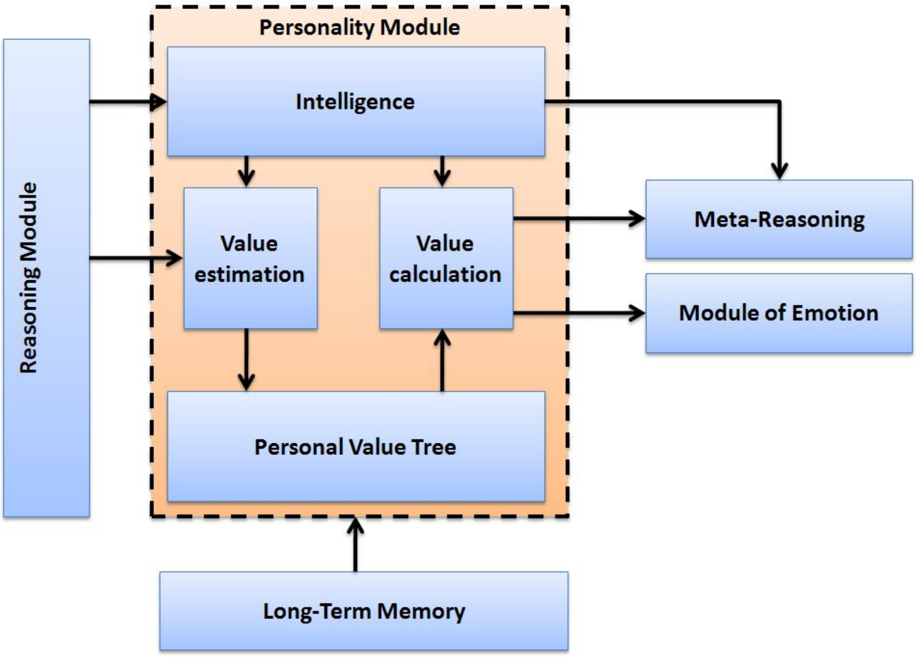


Figure 42 The Personality Module

Intelligence is a factor that is considered to be influenced by both biological determined properties and knowledge. The g-factor is the general ability that does not change, while other kinds of intelligence can be improved by exercises and training (chapter 3.2.8). Intelligence influences the cognitive abilities of the individual and therefore a model of intelligence would influence the speed and accuracy of the cognitive processes (Figure 42).

Most modern theories look upon personality as a result of inherited characteristics and life-long experiences. This means that the personality consists of a permanent and unchangeable biological determined component and a changeable socially learned component that evolves over time. To which degree one component overrides the other is still an ongoing discussion amongst psychologists (chapter 3.2.8).

The learned component is the set of values that a person evaluates himself or herself up against. The values are influenced by cultural, religious, political, economical status etc. To visualize such values one often uses personality graphs. The values give an overview of certain individual tendencies of behavior (30; 32).

The personality properties influence both meta-reasoning and emotions. Meta-reasoning is based on what is most important for the individual and what is the motivation of the individual. The motivation is highly influenced by the personal moral code, and rules of conduct. Some cultures weigh the importance of social interaction higher than other cultures (chapter 3.2.8).

The individual's values are constantly under evaluation due to environmentally induce experiences. Group pressure is normal in all cultures and all parts of the world. Even though the demand to comply with social norms and rules are different in each culture, the individual is suspect to group induced biases. This means that social interaction alters personal values over time. The question is to understand which values are my own and which values are inherited from one's own culture or group (32). Properties of a personality module are listed in Table 12.

Property	Description
Intelligence model	Intelligence stipulate to a certain degree inherited individual variances in cognitive abilities and parts of intelligence that can be improved by learning.
Value model	A value model covers some basic notions of what an individual belief system would contain. A value system would give an idea of tendencies toward for instance participation in corruption or terrorist attacks. Example IPIP NEO IP-R.

Table 12 Personality Module Properties

4.3.6 The Social Module

The social module covers aspects of group relations. Members of groups play a role in that group, which is either imposed on them by a group consensus or by promoting oneself for a specific role. A group is a network of relationships, with varying strength of the connections. The connection strength is a measurement of the trust one direct towards another member of the group (chapter 3.2.9).

One individual can be a member of multiple groups and the attraction level of the group explains the individual's dedication to that group. A group has a set of norms and values that are different from the values that the individuals themselves have, with increasing dedication to a group to a larger degree will the individual adapt his own values in such a way that they comply with the values of the group. The dedication level is based on rules of attraction (chapter 3.2.9).

A role that influences the group to a large degree is the role of the leader. The leader's personal abilities to read the emotions of the members and provide a productive and including environment will influence the group cohesion (33).

A social module therefore must model group memberships, roles of the group members, the attraction levels within the group and the attraction levels between the group itself and the individual (Figure 43). The latter case can be a calculation of the internal attraction levels, where the result represents the attraction level to the group. For every group membership the norms and rules that follow that membership needs to be concretized and comparable to the personal values (30; 32; 38).

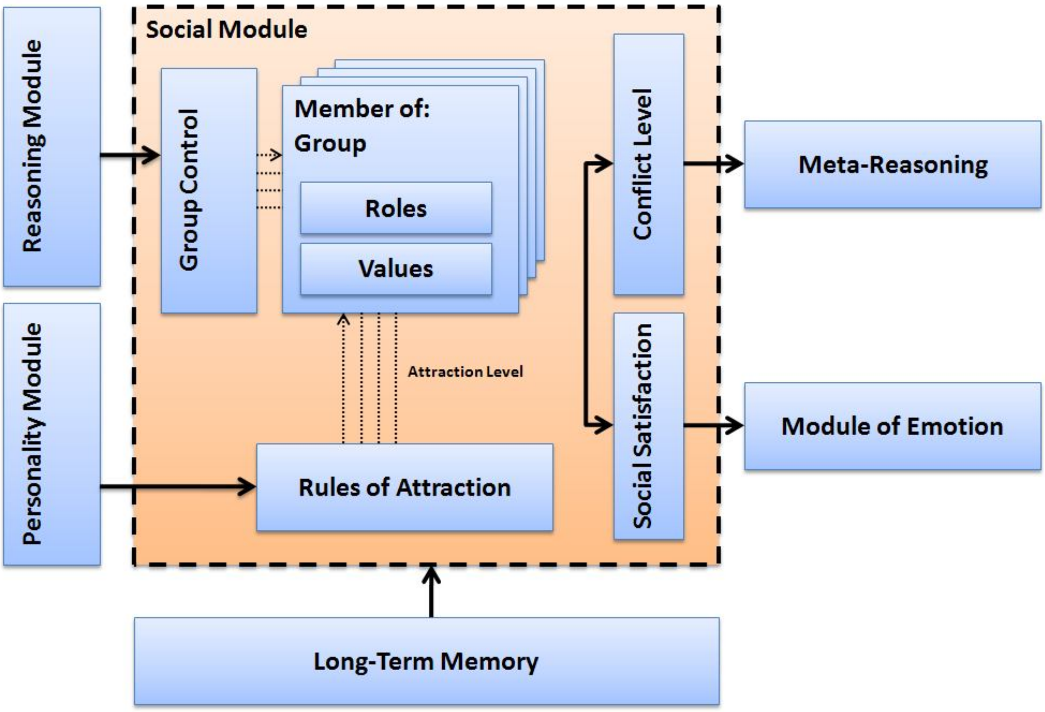


Figure 43 The Social Module

The personality module will influence the degree in which the individual will be attracted to a group or not. Personal values that strongly contradict the values of the group would in most cases lead to low attraction levels (Figure 43). Other properties that influence group membership are the time span in which an individual has been a member. Treatment of new members is often very superficial in contrast to members that have been a part of the group since the group was established. The division of power within the group decides alliances and is the foundation for estimating the conflict level of the group. Power struggles are very common in organizations, and constitute disruptive elements in a group (33).

Properties that a social module should support are the creation and termination of social relationships, together with mutually adjusting personal values and group values. The conflict level of the group determines the priority the group has in the cognitive task processing hierarchy. Table 13 lists properties of a social module.

Property	Description
Group model	The method for implementing groups and organizational hierarchies.
Rules of attraction	The method for calculating attraction levels.
PMF for social satisfaction	Functions that calculate the individual's satisfaction level for social interaction.
PMF for conflict level	Functions that calculate the individual's social conflict levels. The conflict level and social satisfaction level are dependent of each other.

Table 13 Social Module Properties

4.4 Preliminary Evaluation of Agent Architectures

The master thesis identifies a long range of architectures that support high-level HBR, but were only a handful can be said to fall within the area of interest. The number of architectures is too large to give an extensive evaluation of them all, which is why a preliminary screening of the architectures is necessary in the interest of excluding most of them from the in-depth evaluation. The screening is based on personal observations and earlier work done by (3), (27), (28) and (39). Personal observations include reading information on the internet sites, downloading and trying out the different implementations of the agent frameworks, browsing through white-sheets and manuals and contacting key persons in some of the projects.

The architectures have been categorized in four different levels based on maturity and to which degree they fit into the scope of being usable as a HBR in a SE. Each level is defined according to the list given below and are named mature, experimental, expired and out-of-scope. To be chosen for evaluation the architecture must be realized in software and considered to be mature. A mature architecture would include all sub-criteria (AND), while for the other categories it is enough that one sub-criterion is fulfilled (OR).

MATURE	Extensive amounts of support documentation AND
	Extensive amount of articles written AND
	High degree of activity both on software publications and internet site AND
	Successfully used in SE before AND
	Profoundly connected theory.
EXPERIMENTAL	Low degree, incomplete or lacking support documentation OR
	Low activity on written articles or out-of-scope articles OR
	Incomplete or no software publications OR
	Mixed-up or confusing internet site with few updates OR
	Few or non-existing attempts to incorporate in SE before OR
	It is a theoretical model only.
EXPIRED	More than 8 year old documentation OR
	Only of historical relevance and not possible to obtain software OR
	New, improved or enhanced versions have taken over.
OUT-OF-SCOPE	Model is not suited for SE OR
	It is a language for specialized functions OR
	Targeted use is too narrow and put limitations and restrains on choice of SE OR
	The model or framework is restricted due to governmental publication clauses.

Models that are highly specialized and only applicable to a limited area cannot be said to cover all the basic needs of HBR in a SE, and are therefore considered to be out of scope. Some special notifications have been made when it comes to architectures using Artificial Neural Networks (ANN). ANNs are considered to be experimental, due to need of extensive training data and that it is put to limited use. This means that ANN architectures are considered to generally be on an experimental stage due to limitations of the technology itself. The result being that they are evaluated as not fitted to use in a complex SE (27).

Architectures that are chosen for evaluation are marked in Table 15. The table also lists some extra information about the architectures. This information gives a short description of the architecture. The second column is denoted type and the abbreviations used here are shown in Table 14. The third is the SW column, which informs about software being obtainable or not. The last column is the reference column, which points to a webpage in the bibliography.

Type Long Name	Short Name
Analysis Method	AM
Artificial Neural Networks	ANN
Belief, Desire, Intention	BDI
Cognitive Architecture	CA
Cognitive Layer	CL
Development Environment	DE
Expert System	ES
Graphical Development Environment	GDE
Programming Description Language	PDL
Runtime Environment	RE
Symbol Learning System	SLS
Theory of Cognition	TC
Unavailable	UN

Table 14 Category Type for Architectures

The identified architectures are only referenced to with their acronyms in Table 15. Additional information is available in the appendix. Appendix B: Frameworks for Modeling HBR Abbreviations shows the architecture's full name and Appendix C: Frameworks for HBR and their Original Purpose gives a short description of the architecture and its original purpose. Table 15 presents the preliminary evaluation.

Acronym/ Abbriviation	Type	SW	State	Reference
ART	ANN	YES	Experimental	(40)
APEX	CA	YES	Mature	(41)
ACT-R	CA	YES	Mature	(42)
Brahms	CA	YES	Mature	(43)
CHREST	CA	YES	Experimental	(44)
CogAff	CA	YES	Experimental	(45)
COGNET	CA	YES	Expired	(27)
iGEN	CA	YES	Out-of-Scope	(46)
CCT	PDL	NO	Expired	(27)
CES	UN	NO	Expired	(28)
CF	UN	NO	Experimental	(47)
CoJACK	CL	NO	Experimental	(48)
COGENT	GDE	YES	Out-of-Scope	(27)
CREAM	AM	NO	Out-of-Scope	(28)
COSIMO	UN	NO	Expired	(28)
CAPS	CA	NO	Expired	(27)
4CAPS	CA	YES	Experimental	(49)
CLARION	CA	YES	Experimental	(50)
C-I Theory	SLS	NO	Out-of-Scope	(27)
DCOG	UN	NO	Experimental	(27)
EPAM	CA	YES	Experimental	(51)
EPIC	CA	YES	Mature	(52)
GOMS	PDL	NO	Out-of-Scope	(53)
GLEAN	RE	YES	Out-of-Scope	(53)
HOS	CA	NO	Expired	(27)
IMPRINT	CA	NO	Out-of-Scope	(54)
IPME	CA	NO	Out-of-Scope	(55)
JACK	BDI	YES	Mature	(56)
JADEX	BDI	YES	Experimental	(57)
Jason	BDI	YES	Experimental	(58)
JESS	ES	YES	Out-of-Scope	(59)
LSA	SLS	NO	Out-of-Scope	(27)
LICAI	SLS	NO	Out-of-Scope	(27)
MIDAS	CA	NO	Out-of-Scope	(60)
Micro SAINT	CA	YES	Out-of-Scope	(61)
PDP++	ANN	YES	Expired	(62)
Emergent	ANN	YES	Experimental	(63)
OMAR	DE	YES	Out-of-Scope	(64)
PMFserv	CA	YES	Experimental	(65)
PRS	BDI	NO	Expired	(27)
PSI	TC	NO	Experimental	(66)
Micro PSI	CA	YES	Experimental	(67)
R-CAST	CA	NO	Experimental	(68)
SIM_AGENT	DE	YES	Experimental	(45)
SMoC	UN	NO	Expired	(28)
SAMPLE	CA	NO	Out-of-Scope	(69)
Soar	CA	YES	Mature	(70)

Table 15 Preliminary Architecture Evaluation

4.5 Performance Moderator functions

Performance models are described earlier in the chapter as cognitive models that cover a narrow field of human cognition. Performance Moderator Functions (PMF) describes concrete performance factors that influence human behavior. A computer can do many million calculations per second. In contrast a human being can only complete a certain amount of tasks within a specified time frame and often the result is dependent on several human factors that moderate the performance. The PMF ensures that performance is moderated according to biological, personality and social factors. In addition the PMF ensures that the simulated entity does not exceed the prevailing theoretical limits of human cognitive performance.

4.5.1 Introduction to PMFs

Table 16 lists some PMFs that describe human performance under certain conditions. The functions in the table focus on human performance under physical stress. The PMF server project, which is lead by Barry G. Silverman, has developed an application that integrates several PMFs into an unified behavior architecture (30). Elements of this architecture will be presented due to the fact that it is an architecture that has based its design on integrating state of the art PMFs into a cognitive architecture. The PMFs in the table make a representative selection for explaining the notion of a PMF on the background that some of them recently have been evaluated for use in military applications (71).

Function Name	Abbreviations
Circadian Neurobehavioral Performance & Alertness	CNPA
Fatigue Degradation	FADE
Physiological Stress Index	PSI
Sleep, Activity, Fatigue, and Task Effectiveness	SAFTE
Sleepiness-Induced Lapsing and Cognitive Slowing	SILCS

Table 16 Example Performance Moderator Functions

Circadian rhythm is the daily 24 hour cycle in the physiological, biochemical or behavioral processes of living beings. CNPA looks at how human beings have fluctuating levels of performance and alertness during the daily cycle. There are several masking factors that make it hard to measure circadian rhythmicity in a human being, for instance environmental factors that induce stress (72).

Maintaining alertness over periods of time is often difficult for human beings. The operating environment often demands a high state of alertness and readiness:

"FADE is a fatigue algorithm that predicts human response capabilities over an extended period of sleep and wake cycles. The focus point of the algorithm being the interaction of prolonged sleep deprivation or fragmented sleep with circadian disruption on crew performance and on sleep recovery from fatigue" (73).

Constructing computer models that simulate human behavior need quantifiable values which are computational sane. PSI creates a physiological strain value that is based on rectal temperature and hart rate. The index ranges from 0-10 and gives a picture of stress induced by heat (74).

Models that incorporate a full range of performance factors ease the construction of human behavior models and SAFTE constitutes such an integrative model. The model simulates sleep and performance through several interacting modules. The performance module calculates the cognitive effectiveness based on inputs from a circadian oscillator and a sleep reservoir. A sleep regulation module computes the quality of sleep and controls the accumulation of sleep in the sleep reservoir (75). The sleep reservoirs are reduced during waking hours and refilled during sleeping hours. The Fatigue Avoidance Scheduling Tool (FAST) constitutes the user interface that makes it possible to interact with the SAFTE model simulation (76).

In the military one often need to deal with situations where the amount of sleep is limited. Sleep deprivation results in certain recognizable behavior characteristics. The SILCS model predicts behavior patterns based on sleepiness. Examples of typical characteristics are slowing of responses, increasing occurrence of delayed responses, increases in duration of delayed responses and increases in error production in a task environment (77).

4.5.2 Integrating PMFs with PMF server

The PMF server (PMFserv) is an architecture that in principle interconnects several independent PMFs. The problem of unifying several PMFs is to determine how they should be connected and which dependencies should be established between them. A PMF constitute a specific effort to model a small aspect of human cognition, how it relates to other PMFs is in most cases not the focus point of different research efforts.

PMFserv contains several subsystems that simulate the physical, cognitive, social and individual aspects of human behavior and where the memory is the foundation for knowledge retrieval for all subsystems (Figure 44). Each subsystem is constructed of state of the art PMFs, which are interconnected. A subsystem provides inputs and outputs for other subsystems. Stimuli from the simulated world are given through the perception module and responses through an expression module (30).

PMFserv does not maintain an internal representation of the simulated world. Instead the perception module receives a perception object that contains all the needed knowledge about the environmental object including how the object can be perceived and used. The object contains multiple perceptual types (representations) of itself, which need to be interpreted by the PMFserv agent architecture. Finding a suitable action is a matter of interpreting affordance values using the current active perceptual type of the agent. The object contains a rule base that describes rules for making perceptual shifts that compel the agent to change its current perceptual type (30).

Several demonstration applications have been developed with PMFserv as the basis structure for creating human behavior. In 2003 PMFserv was used to generate behavior of civilian population in an operational environment such as Mogadishu, which is a scenario known from the movie “Black Hawk Down” (78). More recently PMFserv has been used to simulate crowd behavior, world political leaders and socio-cultural interactions (33; 32).

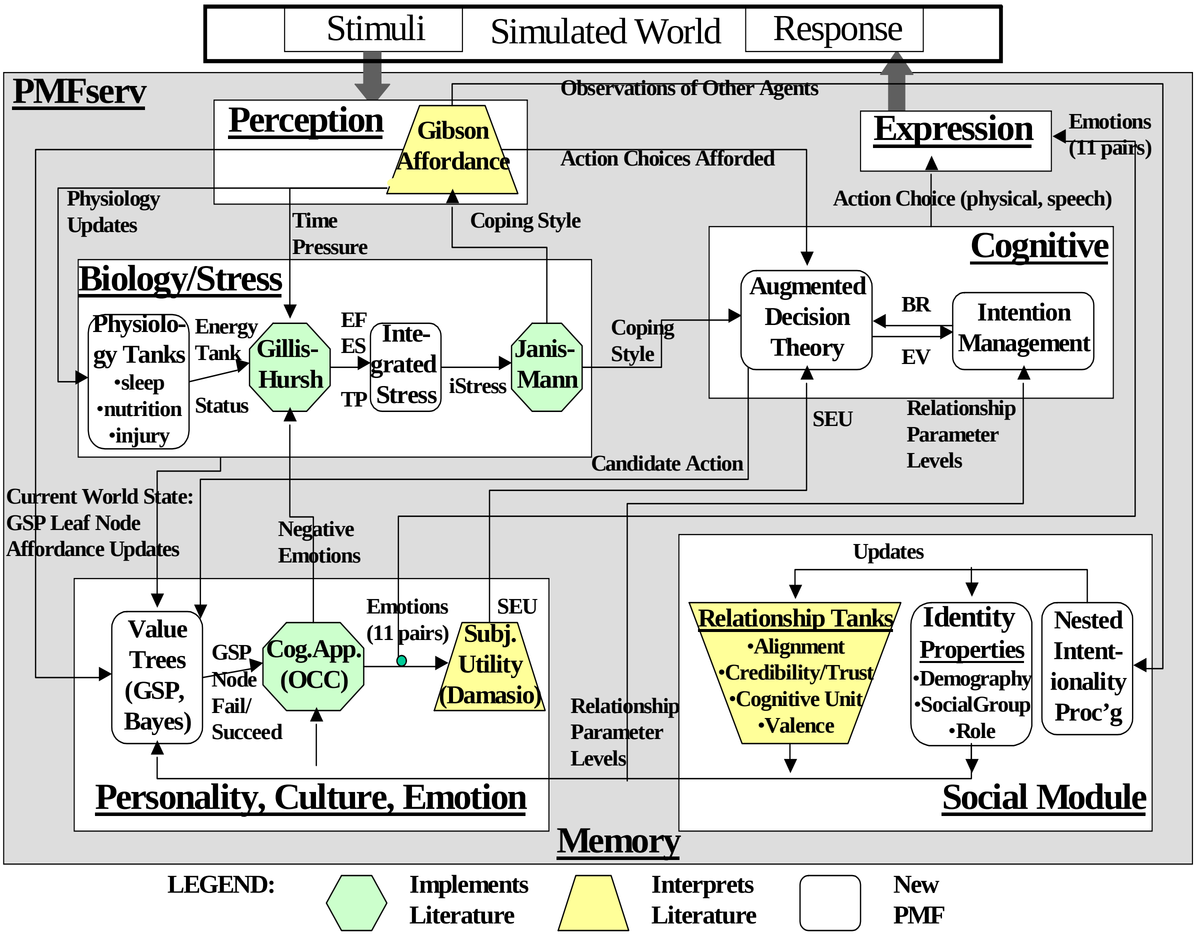


Figure 44 The PMFserv Architecture, from (30)

The PMFs in PMFserv are shown both in the Figure 44 and listed in Table 17. The color coding on the figure tells which PMFs that have been implemented directly from literature, which PMFs that have been developed as a result of interpreting literature and which PMFs are considered to be created solely for the purpose of satisfying the architectural needs (new PMF).

Table 17 gives an overview over the different PMFs incorporated into the PMFserv architecture. The function has the name of the reference given in the PMFserv reports (30; 78) and is therefore not in the reference list of this master thesis. It is important to understand that the research project group for PMFserv has made many compromises and adaptations when incorporating the theoretical foundations into a comprehensive cognitive architecture.

One large problem was how to ensure the validity of the PMFs used and how to determine to which extent one was capable of recreating human behavioral characteristics in the architecture. The PMFserv project has revealed a theoretical shortage on the subject of interconnecting PMFs, which is why one had to develop a range of new PMFs. This revelation has exposed new research areas of HBR and AI in general (30 p. 23).

More information on the PMFserv project and HBR can be found on the webpage of Prof. Barry G. Silverman at the University of Pennsylvania (65).

Function	Description
Janis & Mann (1977)	Methodology for decision strategies for coping under stress, time pressure and risk. PMFserv use a modifier to insert values to this module called integrated stress (iSTRESS) (30).
Gillis-Hursch (1999)	A model for decision-making in combat situations that focus on effectiveness under stress (30).
OCC model (Ortony et al. 1988)	A model of emotions, which states that there are 11 pairs of opposing fundamental emotions. For instance pride-shame and hope-fear. A value tree is used for preference to the model (30).
Gibson Affordance (1979, Toth 1995)	The theory states that people perceive objects in the environment in terms of their affordances. The perception of the object is connected to past experiences and the functionality that a human being associates with the object (30).
Subjective Expected Utility (SEU) (Damasio 1994)	A theory which states that there is a connection between emotions and practical decision-making. The state of the body in conjunction with emotions, constructs intuitional markers that guide decision-making (30).

Table 17 PMFserv Main PMFs

4.5.2.1 HBR properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix D: Properties of PMFserv. The pie diagram shows which model groups PMFserv concentrates on fulfilling in the architecture. PMFserv mainly focus on biological, perceptional, emotional and social aspects of human behavior (Figure 45).

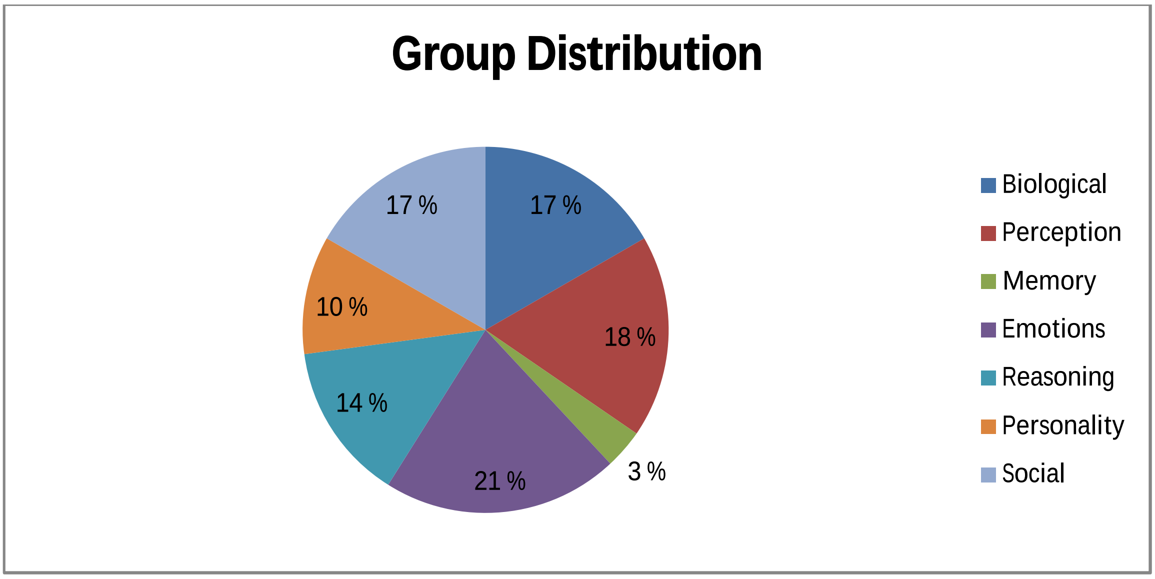


Figure 45 PMFserv Architecture Module Group Distribution

4.6 Cognitive Architectures

Computational models that try to create an overall model of cognition can be said to be cognitive architectures (CA). The notion of CA includes architectures that are candidates for UTC, although being a CA does not mean that it is an UTC. A more complementary definition of cognitive architectures is given in the definition below.

Definition 16 Cognitive Architecture

A CA is the overall, essential structure and process of a domain-generic computational cognitive model, used for a broad multiple-level, multiple-domain analysis of cognitive behavior (79).

Cognitive architectures that were chosen for in-depth evaluation in the previous sub-chapter (4.4) are listed in Table 18. EPIC could have been eliminated on the background of being overly focusing on sensory and motor processes, but due to the success of using EPIC as a sensory pre-processor for Soar and ACT-R it is on the evaluation list.

Architecture Name	Abbreviation
Architecture for Procedure Execution	APEX
Atomic Components of Thought – Rationale	ACT-R
Business Redesign Agent-Based Holistic Modeling System	Brahms
Executive Process/Interactive Control	EPIC
State, Operator and Result	Soar

Table 18 Cognitive Architectures to Evaluation

4.6.1 Architecture for Procedure Execution (APEX)

4.6.1.1 Description

The APEX architecture comprises two main components, namely the action selection architecture (ASA) and the human resource architecture (HRA) (Figure 46). The ASA is the structure element that performs the scheduling, prioritizing and interpretation of tasks, while HRA provide access to resources defined in modules. The methods for processing tasks are consistent with the reactive action package (RAP), which introduce goal elements that set the agenda of the agent (27 pp. p. II-10).

The agenda of the ASA is set through prioritization of goals, which can be reached in multiple ways. Task monitors are created to handle wait for conditions. The tasks can be said to form production rules that need certain conditions to be fulfilled before they are executed. The ASA can receive tasks from bootstrap (initialization procedures), from procedures and from itself. The task processing cycle constitute a problem-solving cycle (80), which resembles the theoretical problem-solving cycle in chapter 3.2.7.

The HRA on the other hand receives inputs from the task environment and passes them on to the ASA in an understandable format, while the outputs from ASA are passed on to the actuators defined in a HRA module. HRA consists of different modules (also known as templates in APEX) that specify perceptual, cognitive and motor skills or resources. APEX comes with several built in templates that constitute a library of resources available for the modeler. At the moment the templates cover human and robot resources. Examples on human templates are memory, vision, auditory, hands, gasp and voice. The HRA supports interrupts from external and internal events. The events result in the creation of new tasks. The modules are resources that have restriction on the concurrent number of tasks using them at the same time and for how long they can be occupied. The timing and resource constraints are described in the template definition (80).

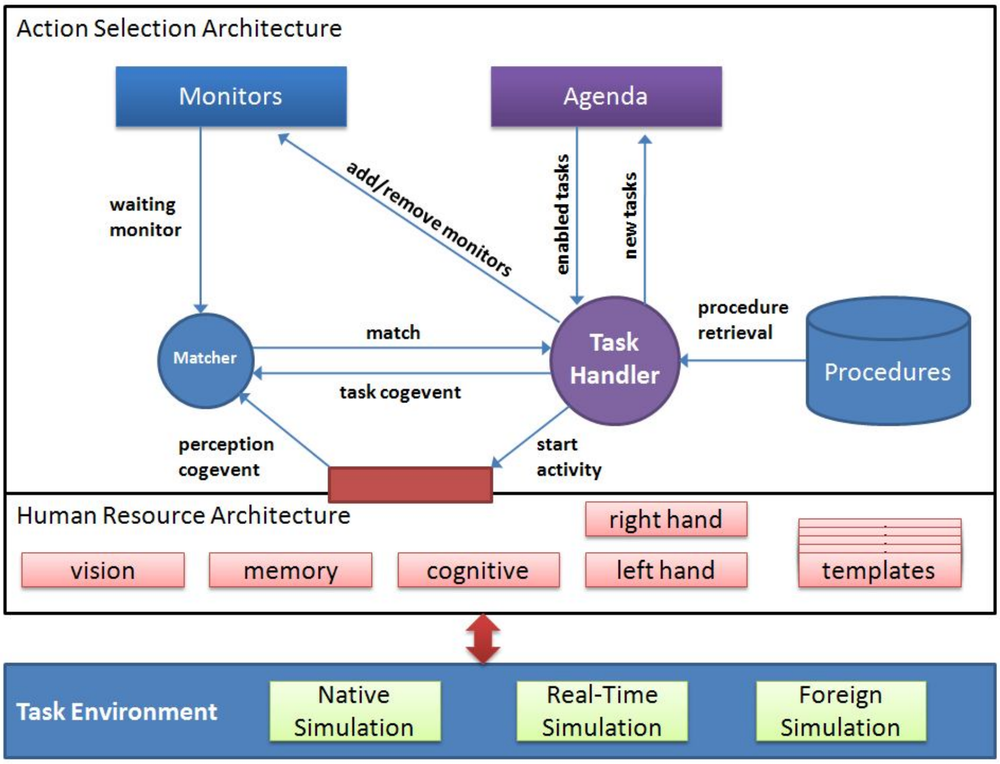
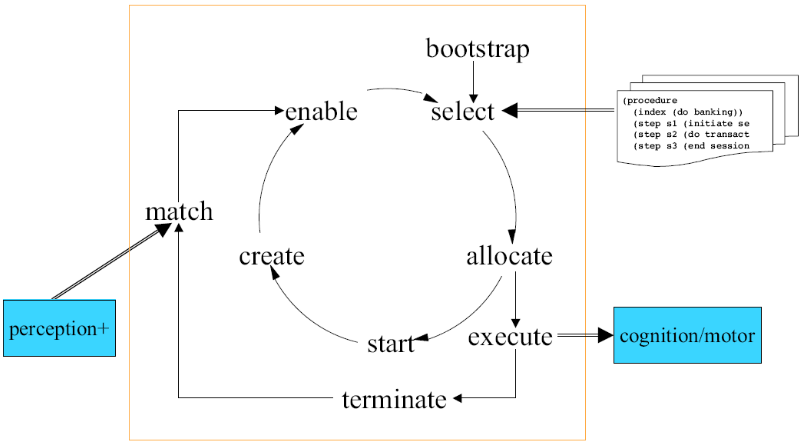


Figure 46 The APEX Cognitive Architecture, from (80)

The ASA cycle consists of five steps and a task monitor for pending tasks waiting for a matching condition (Figure 47). Tasks are stored in the agenda and when initiated a matching clause is used to index or identify a stored procedure for execution.



```
(procedure
  (index (do banking))
  (step s1 (initiate se
  (step s2 (do transact
  (step s3 (end session
```

Figure 47 The ASA Processing Cycle, from (80)

The procedure describes which resources are needed for performing the task. The allocation step allocates the specified resource and resolves conflicts that erupt due to limited resource supplies. Primitive tasks do not need to allocate resources and can be executed immediately. There exist some built-in primitive tasks that start or terminate another task. Each step in a procedure starts a new task and for every wait-for condition a monitor is created. Tasks from the external environment are called cog-events and represent changes in perception. Tasks are enabled when all monitors for that task are satisfied. A monitor is satisfied when it matches a cog-event (80).

There exist three kinds of applications supported by APEX: native, real-time and foreign simulations applications. The built-in library provides the most basic knowledge objects (vision , sound, etc.) for the architecture, but in most cases there would be a need for creating own modules for specialized tasks (81).

The APEX architecture has been used in projects that range from autonomous rotorcrafts to virtual airspace modeling and simulation (82).

4.6.1.2 HBR Properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix E: Properties of APEX. The pie diagram shows which model groups APEX concentrates on fulfilling in the architecture. APEX mainly focuses on perception, memory and reasoning (Figure 48). One should nevertheless keep in mind that CA's provide a framework for implementing any given model and that this model can contain HBR properties.

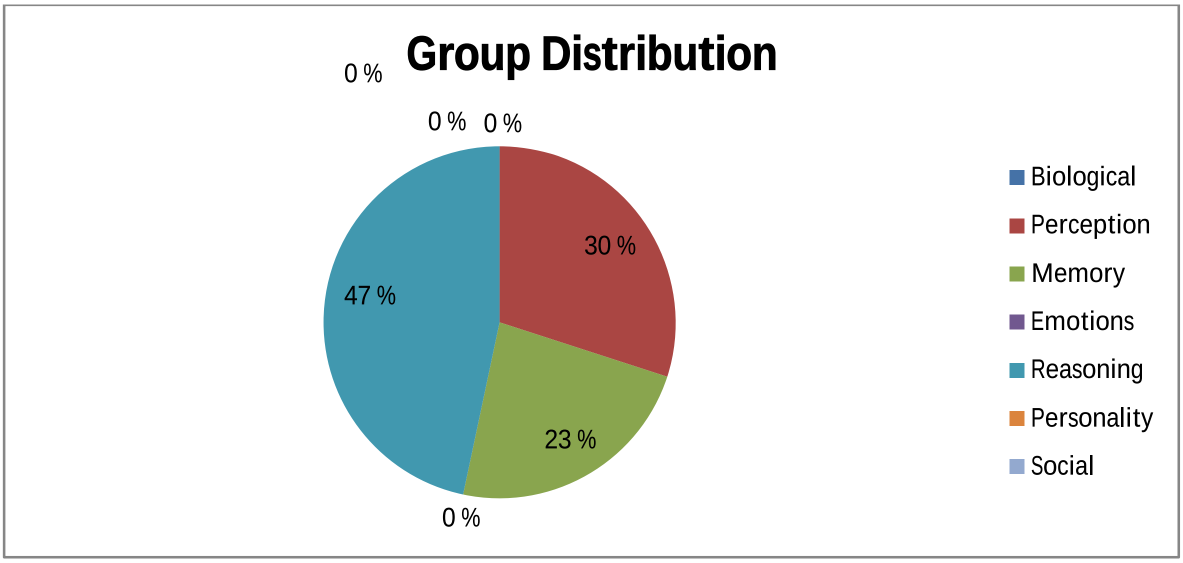


Figure 48 APEX Module Group Distribution

4.6.2 Atomic Components of Thought – Rationale (ACT-R)

4.6.2.1 Description

The ACT-R architecture consists of several modules that interact through a discrete event simulation system (Figure 49). Each module has a buffer which contains the information to be processed by the production rule interpreter. The meta-process maintains a simulation clock and an event queue that ensures timing and coordination of operations. It is possible to run multiple meta-processes at the same time, something which makes it possible to run different models in parallel. The event queue contains a sequence of events inserted by modules. The event states the time of execution, which module that made the request, in addition to specifying the action to be executed. The meta-process can be instructed to run in different modes, either in a simulated time frame or in real-time (83).

Knowledge is formatted as chunks and each buffer can store one chunk at the time. The architecture handles two types of knowledge, namely declarative and procedural knowledge. Declarative knowledge in the form of chunks and procedural knowledge in the form of production rules (84).

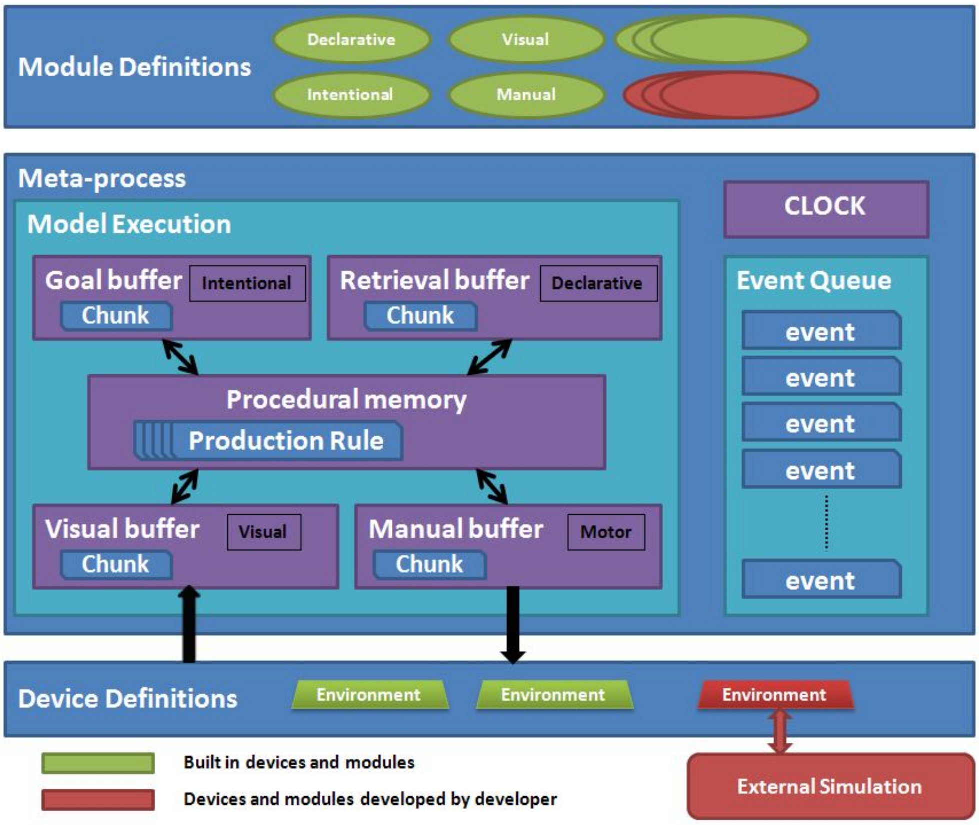


Figure 49 The ACT-R Cognitive Architecture, from (85)

The buffers are interfaces between the model execution system and the different modules. Four example modules are represented in the figure: visual (perceptive), manual (motor), intentional (goal) and declarative (memory storage) modules. Several other modules are built-in to the core system with a possibility to incorporate even more. It is the modules that incorporate the theories of the human mind. One therefore needs to distinguish between modules of theory and framework modules (software). The framework modules implements complementary functions which ease the implementation of models, while modules of theory adds cognitive functionality to the model execution system (Table 19). The device module adds functionality for interacting with the external world. The device functions as an interface that specifies the commands or operations available for collecting information or acting in a simulated environment (83).

Module Name	Description
VISION	A module to provide a model with a visual attention.
AUDIO	A module which gives the model an auditory system.
GOAL	The goal module creates new goals for the goal buffer.
NAMING-MODULE	Provides safe and repeatable new name generation.
PROCEDURAL	The procedural module handles production definition.
PRINTING-MODULE	Coordinates output of the model.
BUFFER-PARAMS	Module to hold and control the buffer parameters.
ENVIRONMENT	A module to handle the environment connection if opened.
RANDOM-MODULE	Provide a source of pseudorandom numbers.
SPEECH	A module to provide a model with the ability to speak.
PRODUCTION-COMPILATION	A module that compiles productions.
CENTRAL-PARAMETERS	A module that maintains parameters.
MOTOR	Module to provide a model with virtual hands.
IMAGINAL	A goal style buffer with a delay and an action buffer.
DECLARATIVE	Stores chunks from the buffers for retrieval.
DEVICE	The device interface for a model.
UTILITY	A module that computes production utilities.
BUFFER-TRACE	A buffer based tracing mechanism.

Table 19 Modules in ACT-R, from (83 s. p. 32)

The architecture uses several different equations for calculating memory retrieval and base-level learning. These equations are parameterized with values that reflect timing and probability constraints in human cognition (Table 20) (86; 87). The architecture use techniques that resemble the activation network of Maes (chapter 3.3.3.2).

Name	Description
Activation Equation	Describes the activation of a chunk. The activation of a chunk represents how information is accessed from LTM. The equation is a sum of base-level activation and associative activation.
Basel-level Learning Equation	The equation describes how the pattern of past occurrences of a chunk predicts the need to retrieve it.
Probability of Retrieval Equation	Chunks will only be retrieved if the activation value is above a certain threshold.
Latency of Retrieval Equation	Chunks that are activated will use a certain time period to be retrieved. This equation calculates the retrieval time.
Recognition Time Equation	Predicted recognition times of chunks which can be mapped to activation values.
Production Utility Equation	The utility function for production rule election. Multiple rules might apply for a given situation and the rule with the highest utility is chosen. The learning mechanism adjusts the cost and probabilities that underlie the utility function according to a Bayesian Network.
Production Choice Equation	The probability of choosing production i when there are n productions.

Table 20 ACT-R Equations

4.6.2.2 HBR Properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix F: Properties of ACT-R. The pie diagram shows which model groups ACT-R concentrates on fulfilling in the architecture. ACT-R mainly focuses on perception, memory and reasoning (Figure 50).

One should nevertheless keep in mind that CA's provide a framework for implementing any given model and that this model can contain HBR properties.

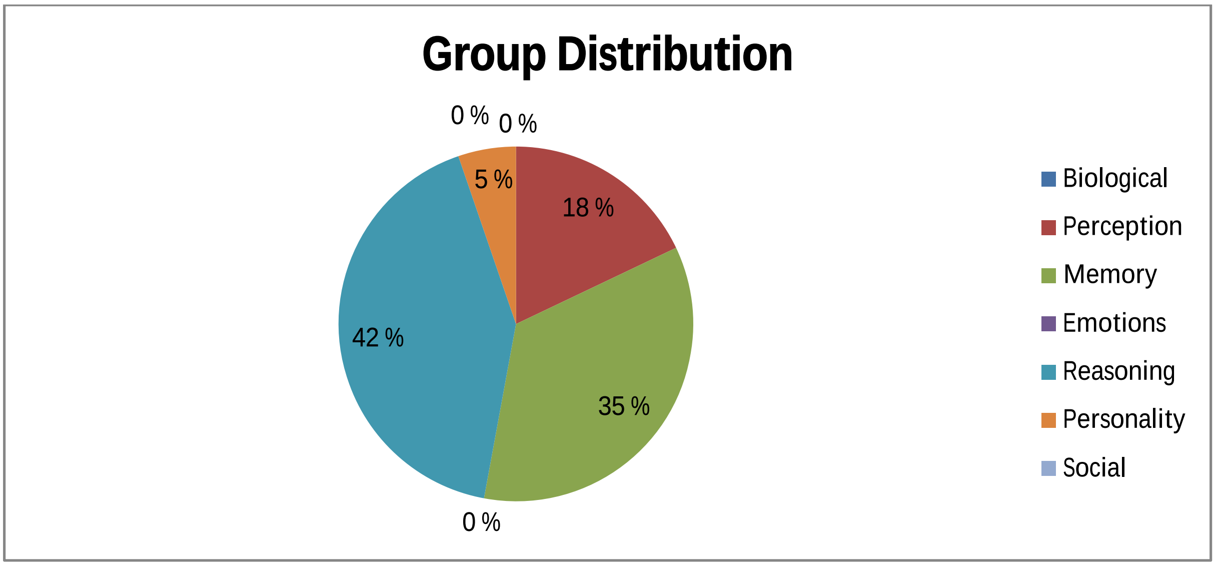


Figure 50 ACT-R Module Group Distribution

4.6.3 Business Redesign Agent-Based Holistic Modeling System (Brahms)

4.6.3.1 Description

The Brahms architecture was specifically designed to model work related environments. The theory incorporated into the architecture is threefold, namely socio-technical systems theory, activity theory and distributed cognition theory. Brahms models are described in the Brahms Language, which need to be compiled before it is run in the Brahms simulation engine. The agent and object behavior is presented as discrete events in the simulation engine (88). In contrary to other CAs the Brahms architecture constitutes a multi-agent system, which means that the model can operate in a computational distributed fashion (Distributed System).

The creation of Brahms models includes making sub-models for agents, objects, knowledge, activities, communication and geography (Figure 51). Agents can communicate with each other through the communication model and the social and work related belongings are stated in the group hierarchy of the agent model. Objects represent simulated world objects and geography state the location where agents and objects can be situated. An object represents a world fact and the belief of an agent does not need to coincide with the world fact. At the center of the architecture is the activity model. Instead of being task related the agents work in an activity based environment. There are four kinds of activities, namely primitive, composite, java and communication activities. The java activity is used when there is a need for connecting the Brahms simulation to external systems (88).

Agents in Brahms implement the BDI architecture (chapter 4.7) with a subsumption based twist (chapter 3.3.3.2). Basically this means that several activities can be active for the agent at the same time and that these activities can be related to each other in a hierarchy. For instance a student is going to college for 4 years and is taking a course in complexity theory for one semester. The two activities are active in parallel, but have different durations. The memory structure contains three types of memory, namely belief memory (BM), thought-frame rule memory (TRM) and work-frame rule memory (WRM). BM constitutes the agents beliefs about the world and itself and can be looked upon as declarative memory represented through FOL. Both TRM and WRM contain procedural rules for changing agent beliefs and constitute procedural memory. In addition work-frame production rules (work-frames) can also instantiate actions and create world facts through the execution of activities (88).

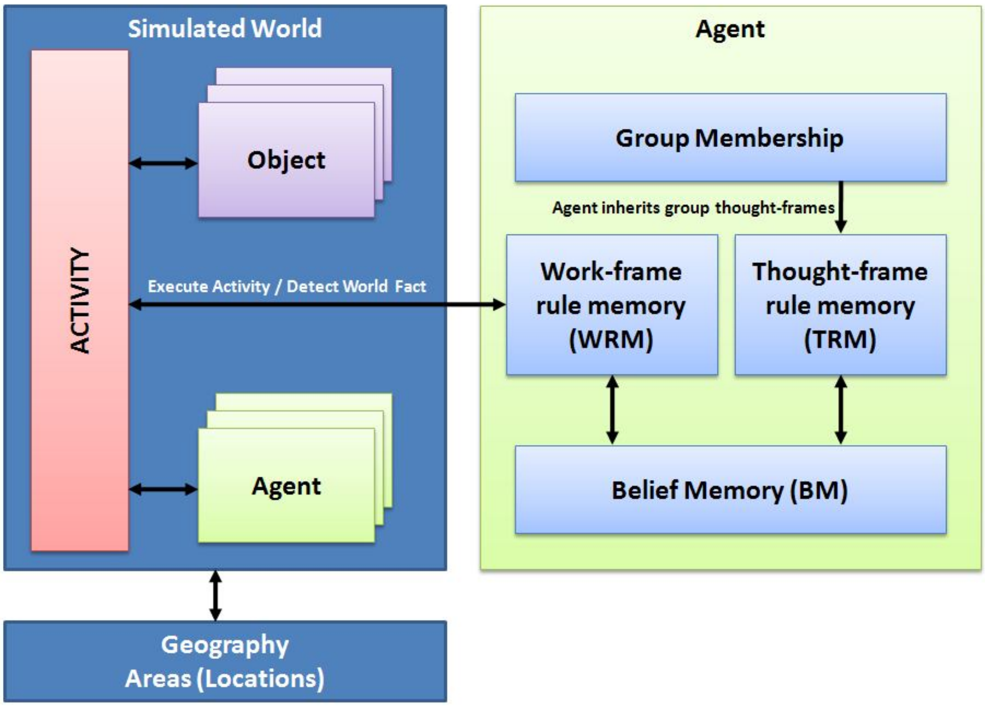

Simulated World
ACTIVITY
Object
Execute Activity / Detect World Fact
Agent
Geography
Areas (Locations)
Agent
Group Membership
Agent inherits group thought-frames
Work-frame rule memory (WRM)
Thought-frame rule memory (TRM)
Belief Memory (BM)

Figure 51 The Brahms Cognitive Architecture

The Brahms agent has a fixed rule memory, meaning that the productions rules cannot be changed by the agent during runtime. The BM stores both long-term and short-term beliefs, which means that the content is constantly being changed by rule firings from procedural memory. The belief is stored as long as the agent needs it but not longer (89). Each agent contains its own inference rule engine that schedules and executes events instantiated by the agent. Distributed activities are coordinated through the Brahms multi-agent event scheduler, which handles communication activities and creation or detection of world facts. Timing is controlled by an overall simulation clock (88).

4.6.3.2 HBR Properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix G: Properties of Brahms. The pie diagram shows which model groups Brahms concentrates on fulfilling in the architecture. Brahms mainly focuses on memory, reasoning and social behavior (Figure 52). One should nevertheless keep in mind that CA's provide a framework for implementing any given model and that this model can contain HBR properties.

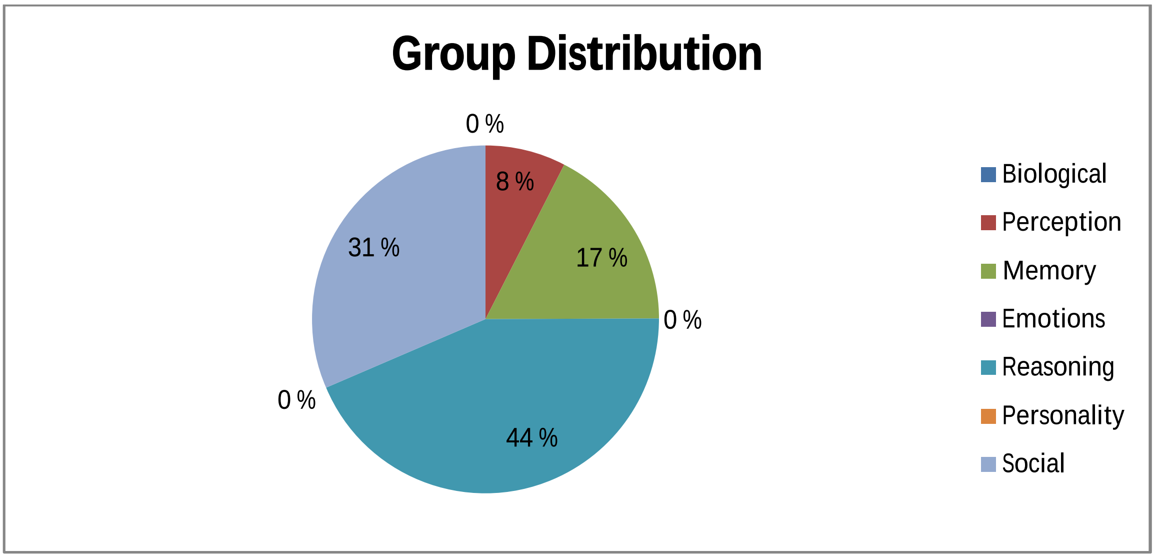


Figure 52 Brahms Module Group Distribution

4.6.4 Executive Process/Interactive Control (EPIC)

4.6.4.1 Description

The EPIC architecture is an event-driven architecture, where event objects are sent between different processors. A processor can even send events to itself. A global object called the Coordinator maintains an event queue, in which all processors can inject events. The event contains a pointer to the destination processor and a time stamp that tells when the event is supposed to arrive at its destination processor. The Coordinator arranges the events according to the time stamp and process them in that order. The simulation time is incremented according to the time stamps in the event queue. This decreases the simulation time considerably, since the time moves forward in discrete jumps instead of in real-time (90).

The processors receive the event and create new events as a response to the received event (Figure 53). The weakness being that it is not possible to change events once they have been injected to the event queue. The work around is to make each processor simulate time delays through self-scheduling, which means that a processor dispatch an event that is directed to itself that tells it to send the original event. The original event can now be canceled if the new event overrides the original event (90).

There are three sensory processors that receive stimuli from the task environment. These are the auditory, visual and tactile processors. Likewise there are two actuators or motor processors for manipulating the task environment, namely the vocal and manual motor processors. A third motor processor is used to generate eye movement, both involuntary and voluntary movements, which is the ocular motor processor. This motor processor does not affect the environment, but how the visual processor interprets visual stimuli (90).

The cognitive processor receives the inputs from the sensory processors and process the perceptual information according to stated productions rules. The result is the creation of command responses that are sent to the motor processors. The cognitive processor operates on a cycle-basis. One cycle consist of executing the rule actions that matched the conditions of the production rules in the production rule memory. The actions either update the production rule memory or send commands to the motor processors. The production system mechanism is called the Parsimonious Production System (PPS) production rule interpreter. Production memory contains the set of production rules implemented by the model, while LTM contains production rules that have not been activated during task execution. The production rule memory contains a subset of the rules in the production memory and constitutes a part of working memory. The matching is an operation that matches the contents of the production rule memory against the production memory. Production rule memory contains clauses that can be added or removed by sensory inputs (90).

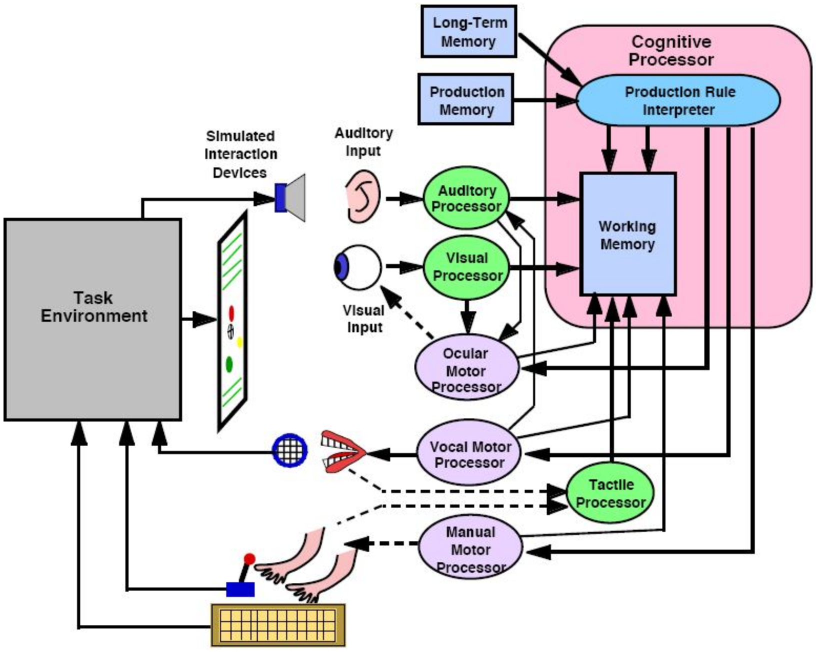


Figure 53 The EPIC Cognitive Architecture, from (90)

Every processor has a set of time parameters that determines the processing time. Parameters can be set to fixed or randomized values through use of statistic distributions. Simple parameters are independent single values, while complex parameters are dependent on another symbolic value. The cognitive processor has a cycle time parameter which is 50 milliseconds. The perception systems have numerous parameters that describe processing times for different visual, auditory and tactile sensations. For instance the time for an object to be visually registered when it is appearing or disappearing, the time for a sound to be registered and the source location identified or identifying the pitch of the sound (90).

The perception systems receive symbolic information from the task environment. EPIC does not translate visual images or sound to a representational format, preprocessing of perceptions must be done in the task environment. Input from the task environment is sent by calling sensory functions from the simulation device (task environment). The symbolic representations of visual or auditory sensations are inputs in these functions. Examples of auditory functions are create, destroy, set source, start and stop, while examples of visual functions are appear, disappear, location, size and property (90).

4.6.4.2 HBR Properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix H: Properties of EPIC. The pie diagram shows which model groups EPIC concentrates on fulfilling in the architecture. EPIC mainly focuses on perception, memory and reasoning (Figure 54). One should nevertheless keep in mind that CA's provide a framework for implementing any given model and that this model can contain HBR properties.

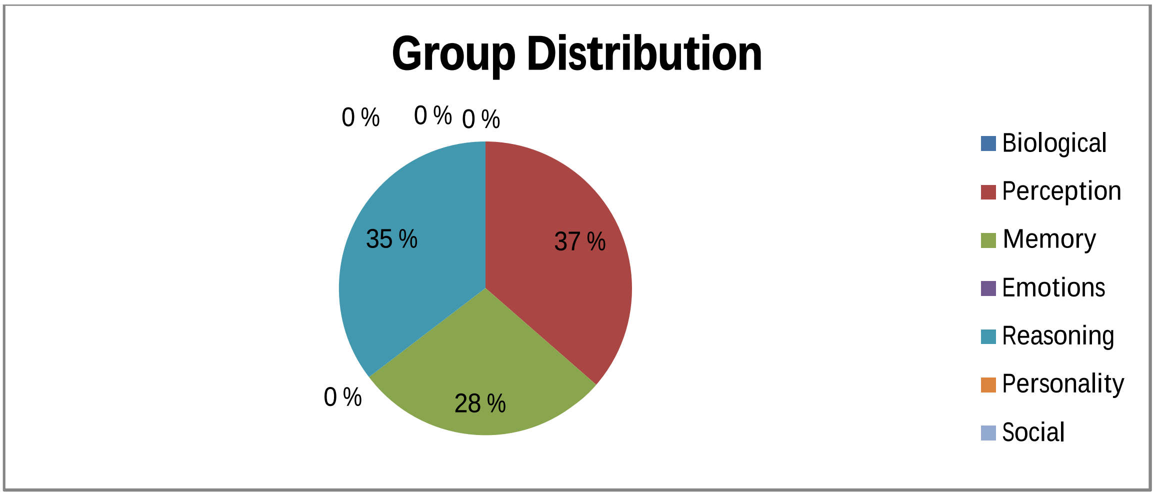


Figure 54 EPIC Module Group Distribution

4.6.5 Soar

4.6.5.1 Description

Soar is an architecture that bases its decisions on searching through problem spaces (chapter 3.2.7). A search through a problem space requires the knowledge of the desired end result through for instance a goal. The operations that one needs to perform to reach the goal are called operators in Soar. The search generates knowledge in the form of chunks that describe which operator to use when in a certain state, meaning that the architecture is capable of learning from experience (91).

The WM contains information that describes the current situation or state in the problem-solving cycle (Figure 55). The information elements stored in WM are called working memory elements (WME) and form statements or properties of an object. The object itself is not represented directly, but through an identifier used by WMEs. The set of WMEs with the same object identifier constitute the object attributes. The attributes point to a value that can be another object identifier, something which creates an interlinked network of objects. Objects that are not linked are removed from the WM by the Soar architecture. WMEs are created on the background of actions performed by productions, decision procedures or externally induced by input links and removed on the background of resolved impasses, decision procedures, production actions, invalid sensory data from input-link or when elements are not linked to a state (92 s. pp. 14).

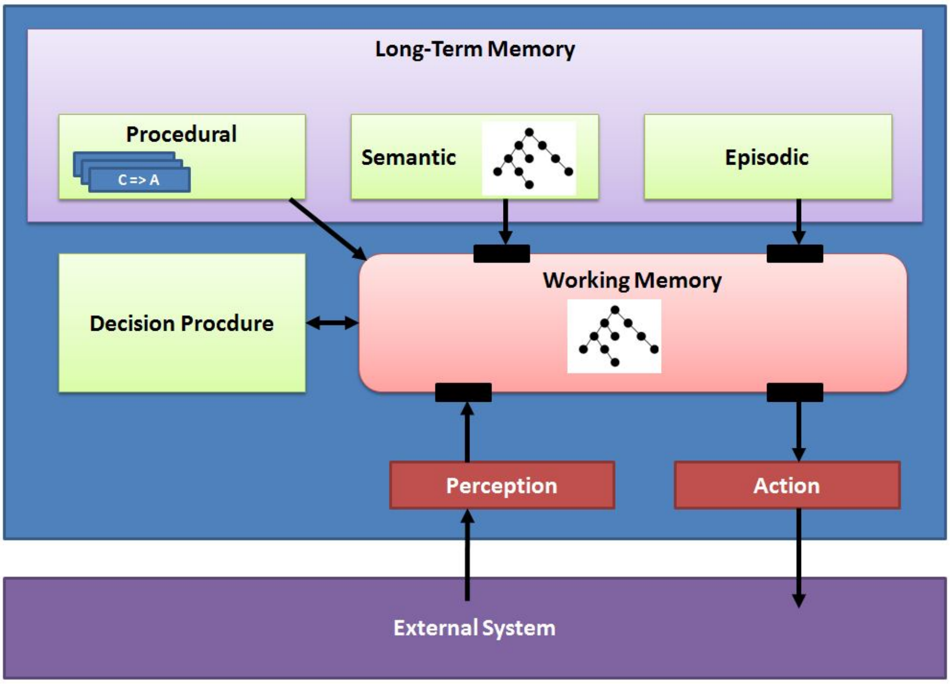


Figure 55 The Soar Cognitive Architecture, from (93)

Long term knowledge is formed in LTM through productions in the procedural memory, world facts in the semantic memory and experienced situations in the episodic memory (Figure 55). The production rule consists of a set of conditions (C) and a suggestion of an operator to choose (action selection - A). The production rule fires when the condition is fulfilled and as a result the architecture proposes an operator. It is the productions in the procedural memory that creates behavior, the semantic and episodic knowledge is needed when the architecture no longer is able to produce operator suggestions due to lack of knowledge.

The execution cycle of soar consists of five steps, namely input, proposal, decision, application and output (Figure 56). The input adds WMEs to the WM which again can result in the firing of production rules. The production rules that fired suggest one operator each. If there are no information in memory that tells Soar which operator to choose and impasse is created. Only one operator can be chosen at the time. If there on the other hand exits information on which operator to choose, then Soar will run that operator. An impasse is also created if there are no operators to choose from. An impasse is a mechanism that tries to resolve the situation of lacking knowledge, which means that Soar will try to search through the problem space for a solution. In the application step the chosen operator is run. The result either being the creation of new content in WM or producing a command output to be sent to the external environment.

An impasse can be said to have a recursive function for searching through the problem space (Figure 57). The mechanism will when the first impasse arise, create a selection state that tries one of the possible operators at random and then try to run it, thereby creating a new switch state (Switch-2). In the new switch state soar can continue to explore the problem space, if a new impasse is created, by creating a new selection state (Selection-2). The process continues until the goal state has been reached or Soar has run out of options. If Soar recognizes a state it has been in before the temporal switch state is dropped. When Soar is successful in finding the goal state it will create a chunk and store it in LTM. A chunk and a production rule is basically the same, but it is called a chunk if it is created by the system. When following the chain of created temporal switch and selection states back to the current state chunks are created and stored in LTM. Soar can now choose the next operator and will be able to do so until the goal is reached.

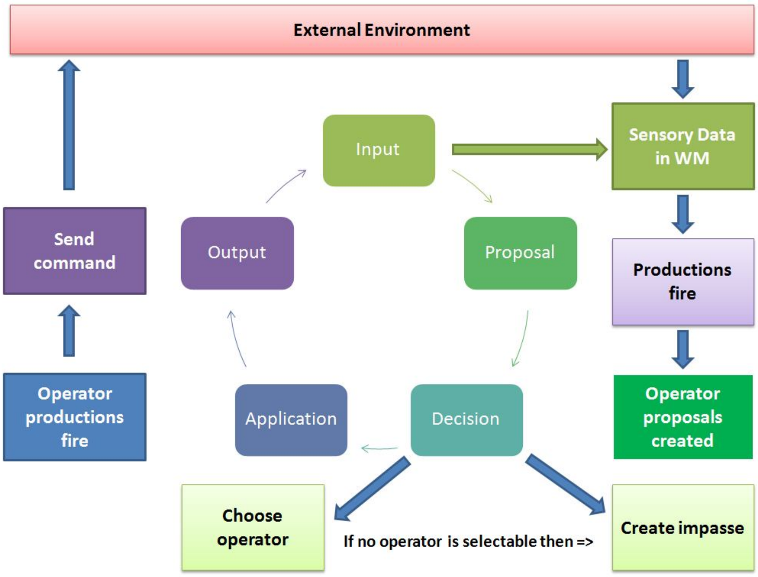


Figure 56 Soar Execution Cycle

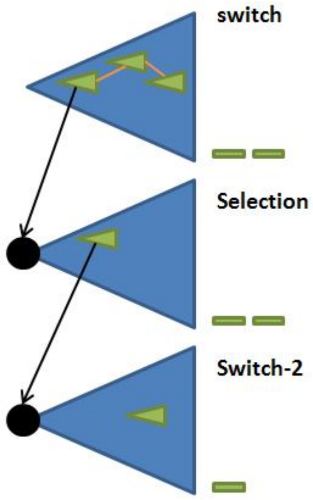


Figure 57 Soar Impasse Triggering

4.6.5.2 HBR Properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix I: Properties of Soar. The pie diagram shows which model groups Soar concentrates on fulfilling in the architecture. Soar mainly focuses on perception, memory and reasoning (Figure 58). One should nevertheless keep in mind that CA's provide a framework for implementing any given model and that this model can contain HBR properties.

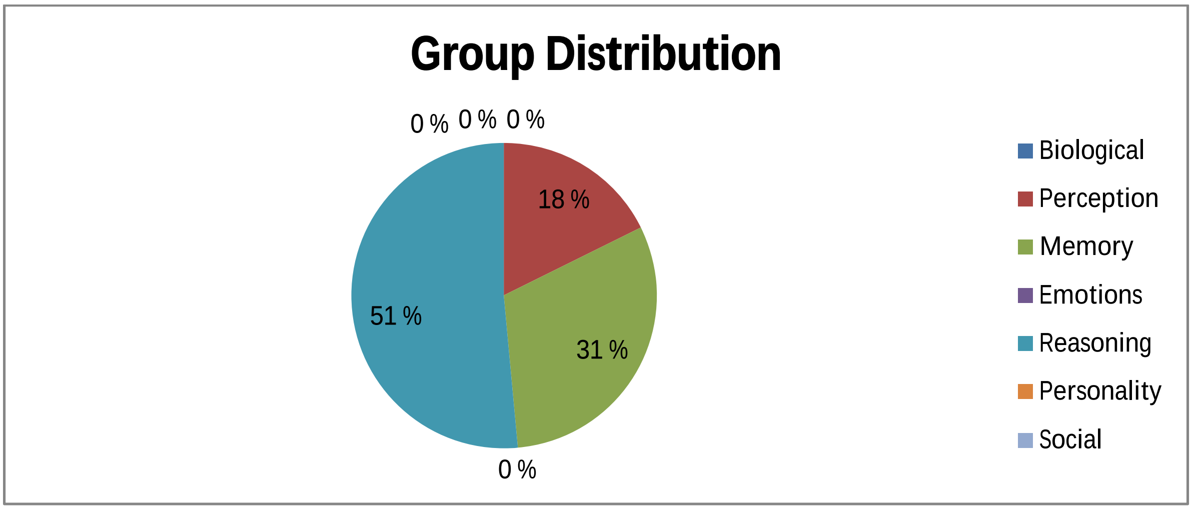


Figure 58 Soar Module Group Distribution

4.7 BDI Architecture

4.7.1 Description

BDI agents are considered to be practical reasoning agents (chapter 3.3.3.1) that contain a set of beliefs (B) that states what the agent knows about the world or think it knows about the world (current state). The desires (D) form the goals in the architecture by expressing wanted states of affairs (goal states). The intentions on the other hand are intended plans of actions that the agent will try to perform in its quest to reach the goal states (94; 95).

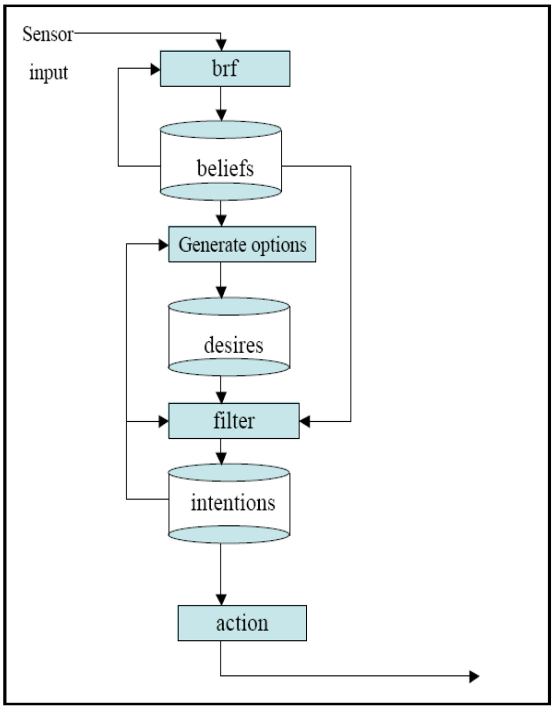


Figure 59 The General BDI Architecture, from (140)

The decision cycle of the BDI agent basically consists of five steps: observe, update, deliberate, reason and execute. The agent receives some perceptive information about the external environment through observation then it needs to update its internal representation of the world, which lead to a change in the current state of the agent. The agent contains a set of plans that function as options or paths to reach a desired end state. The agent therefore needs to prioritize a goal and choose a plan for reaching that goal. The reason for goal prioritization is that the agent can have more than one active goal, because it can only process one goal at the time. The goal has intentions attached to it through plans and the agent makes a commitment to follow one plan through evaluating the set of beliefs up against stated plan preconditions. Finally the agent chooses a plan for execution something which resembles the action of the agent or the behavior of the agent (94).

The BDI architecture does not constitute a cognitive architecture since it is not solely built upon principles of human cognition, but rather constitute an architecture that makes practical reasoning computational efficient (27 p. pp. 7). There are many implementations of the BDI architecture which are based on different programming languages and approaches to knowledge engineering. The significance of this is the subtle changes in the decision cycle process, even though the general principle stays the same (88).

4.7.2 HBR Properties

The evaluation is presented in the form of a sheet that evaluates the properties of HBR identified in chapter 4.3. The sheet is enclosed as Appendix J: Properties of BDI. The pie diagram shows which model groups BDI concentrates on fulfilling in the architecture. BDI mainly focuses on memory and reasoning (Figure 60). One should nevertheless keep in mind that BDI provide a framework for implementing any given model and that this model can contain HBR properties.

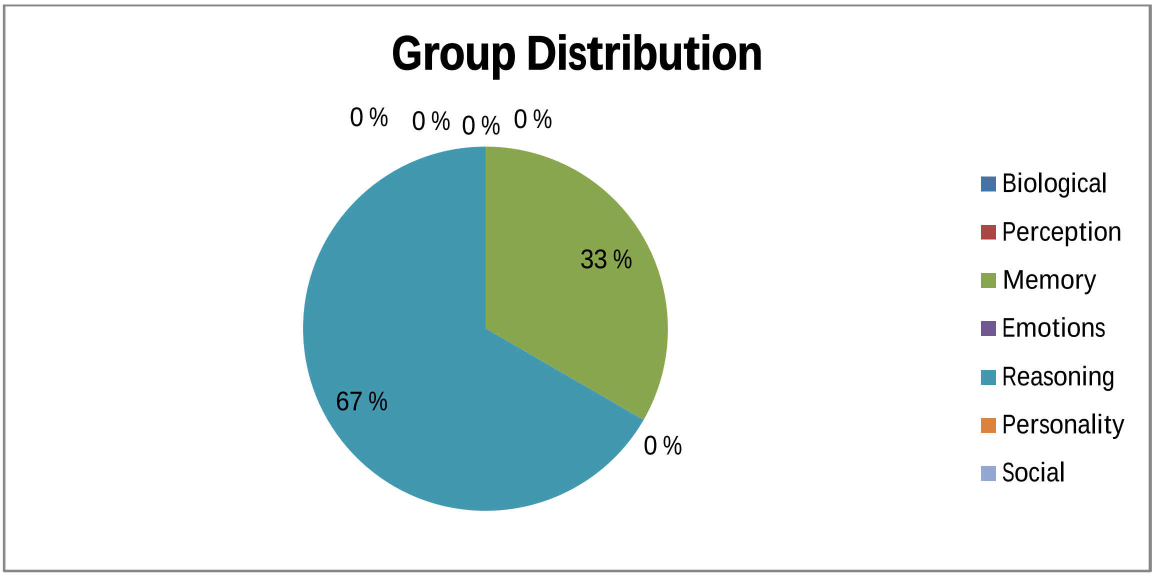


Figure 60 BDI Module Group Distribution

Chapter 5 Simulation Frameworks

The purpose of this chapter is to take a closer look at SEs and how one can interconnect these environments with agent frameworks or other simulation frameworks. Chapter 4 introduced several cognitive agent architectures and the BDI architecture. These architectures constitute agent frameworks and this chapter takes a closer look at their implementation and software constraints. The chapter also states the relationship between behavior model, agent architecture and agent platform.

5.1 Introduction

There exists many frameworks for conducting simulations and they can be classified into categories based on the area of application. Simulations are often used as tools for investigating system limitations and networking performance. These kinds of simulations model network nodes and radio channels etc. The scope of the master thesis does not cover simulation frameworks of this kind, but concentrates on simulations of synthetic environments and virtual worlds and how to incorporate intelligence into these simulations.

The field can be said to be divided into two distinct directions depending on the research objective. The academic community uses such simulations to investigate how the human mind is put together and how intelligent behavior arises from basic computational theories about cognition. On the other hand the commercial and gaming community concentrates on researching how to make modeling simple, affordable and rapid in the interest of speeding up development of intelligent gaming entities. This means that the academic community perceives the development of sane cognitive architectures just as important as being able to implement models in these architectures. The behavior that arises from modeling in the architecture should coincide with cognitive theories and collected research data. For instance it is a fact that human beings only have access to limited information about the world state and therefore make decisions that are considered to be not optimal but best-fit. The gaming community on the other hand wants to obtain behavior that is perceivably intelligent but not necessarily based on cognitive theories. This has lead to solutions where the computer agents cheat by knowing the complete world state. Believable intelligent behavior can be simulated through clever animation and deterministic behavior using finite-state machines (FSM). The game developer creates a finite set of deterministic behavioral functions that can be combined in different ways to create the illusion of intelligent behavior. In the recent years a handful games have been released that uses more sophisticated AI (96; 97).

The need for intelligent behavior in virtual world simulations is increasing because of increasing needs in both the civilian and military market for cheap affordable training environments and scenario analyses. Examples are oil drilling on the seabed of the North Sea or a military campaign in a desert environment. This has lead to commercialization of several architectures developed in the academic community like for instance COGNET/iGEN. The American military and government has several architectures at its disposal for developing simulations in synthetic environments, but most of them are not commercially available and are subject to limitations of distribution like for instance

IMPRINT or OneSAF. The gaming community has used software that eases the creation of intelligent behavior through using FSM editors like for instance Simbionic.

The concept of virtual worlds and SEs has lead to software that makes a distinction between the representation of the world and its physics and the representation of the inhabitants in this world. The virtual world creates a space with objects, in which inhabitants can operate. The rules of the object states what you can do with it. In addition the object has some properties that give information about shape, color and texture etc. Virtual worlds on the internet like Second Life are inhabited by avatars which are 3D representations of humans operating in the virtual space. In this case the avatar is controlled by a human being. Inhabitants that are controlled by an artificial intelligent entity are called agents. Since simulations trying to model natural systems or human systems produce events in an unpredictable order and magnitude, the environment is often riddled with uncertainties and is highly dynamic and complex. Simulating social and virtual environments is often accomplished by combining a multiple range of different agents and components (98).

Systems that provide the virtual world or space are called virtual environments and FFI uses several different virtual environments. Among these are the three simulation environments considered for this master thesis. These simulation environments have built in capabilities of creating AI, but in which none of them incorporates cognitive theories (chapter 5.2). This capability must be introduced by incorporating agents that base its reasoning on cognitive theories. The community around cognitive architectures is often sponsored by governmental and military research organizations and this has lead to the use of cognitive agents in SEs on research level.

Interconnecting virtual environments is hard because they might represent the world differently. Gaming worlds do not only create a world to operate in, they also create the necessary controls for a human operator to interact in the virtual environment. If one desires to connect different simulations together one need to create a mutual understanding of the world representation. If a ground team wants to cooperate with a squadron of aircrafts one need to interconnect the flight simulation and the ground simulation. The question is if this is done directly or through a third virtual environment. Nevertheless compatibility issues due to differences in the use of coordinates, terrain elevation data and damage models are common problems. A standard that tries to ease the compatibility issue is called the High-Level Architecture (chapter 5.3).

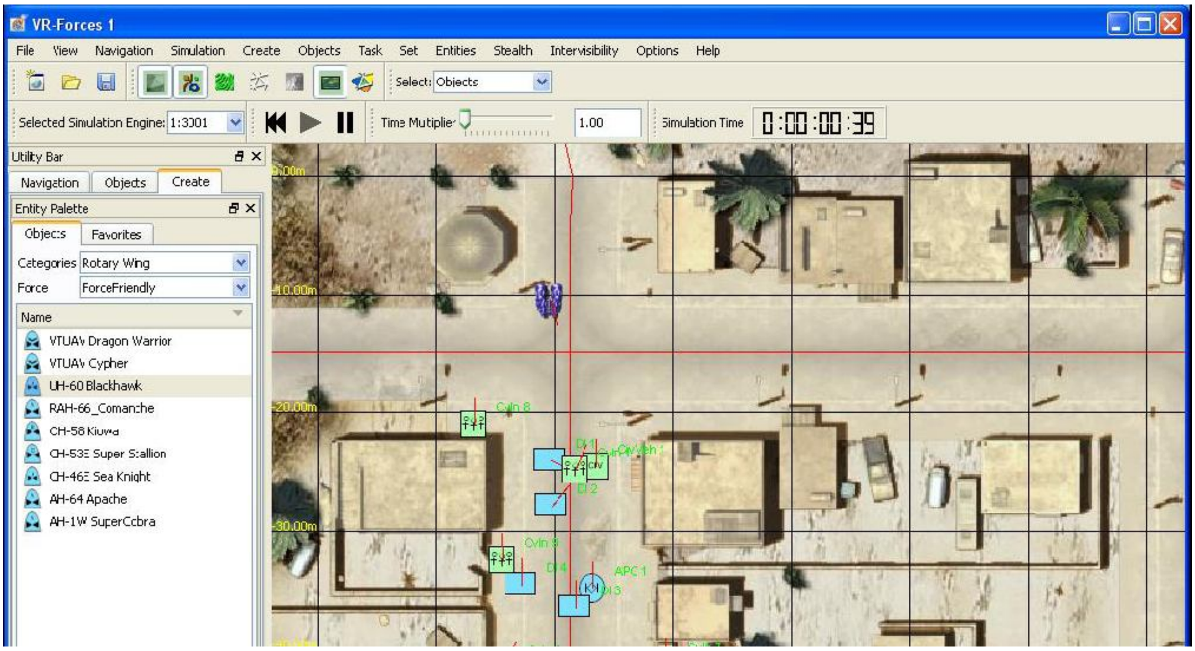
The cognitive architectures discussed in chapter 4.6 and the BDI architecture discussed in chapter 4.7 are implemented in software packages that provide support tools for developing models of human behavior. These tools are presented in this chapter in addition to the FSM editor Simbionic.

5.2 Virtual Environments

Components in a representation of the physical environment consist of objects that one might encounter in the real world. This includes different types of terrain and manmade structures like cities and bridges, which are considered to be static objects in the environment, while for instance weather and time of day are dynamic aspects of the environment. Most agents in military simulations operate in a physical virtual representation, meaning that they are represented by a physical object in the environment. This physical object could be a tank, aircraft or a person placed within the environment. FFI uses amongst others VR-Forces by MÄK Technologies, Unreal Tournament 2004 by Epic Games and Virtual Battle Space 2 by Bohemia Interactive as virtual environments. The artificial intelligence of agents in these systems is either based on plug-in modules or a type of scripting language.

5.2.1 VR-Forces

VR-Forces constitute a complete simulation toolkit for generating and executing battlefield scenarios. VR-Forces make it possible to customize exercises in a virtual environment using computer generated forces (CGF). It contains several databases supporting tactical, strategic and visual 2D or 3D views (Figure 61). An exercise planner can basically set up a scenario with a number of different units. The units can be individually tasked and can interact with the terrain and environment structures. The planning of a scenario is done through a graphical user interface which is connected to the VR-Forces simulation engine. New simulation components and functionality can be added through the VR-Forces C++ API, allowing customization of the application. Support for simulation interconnection is secured by being compliant with Distributed Interactive Simulation (DIS) and High Level Architecture (HLA) standards (chapter 5.3). Other software products from MÄK are the MÄK Stealth which is a 3D viewing tool and the MÄK Run-time Infrastructure (RTI) for interconnecting simulations (99; 100).



Select: Objects

Selected Simulation Engine: 1:3301

Time Multiplier 1.00

Simulation Time 0:00:00:39

Categories Rotary Wing

Force ForceFriendly

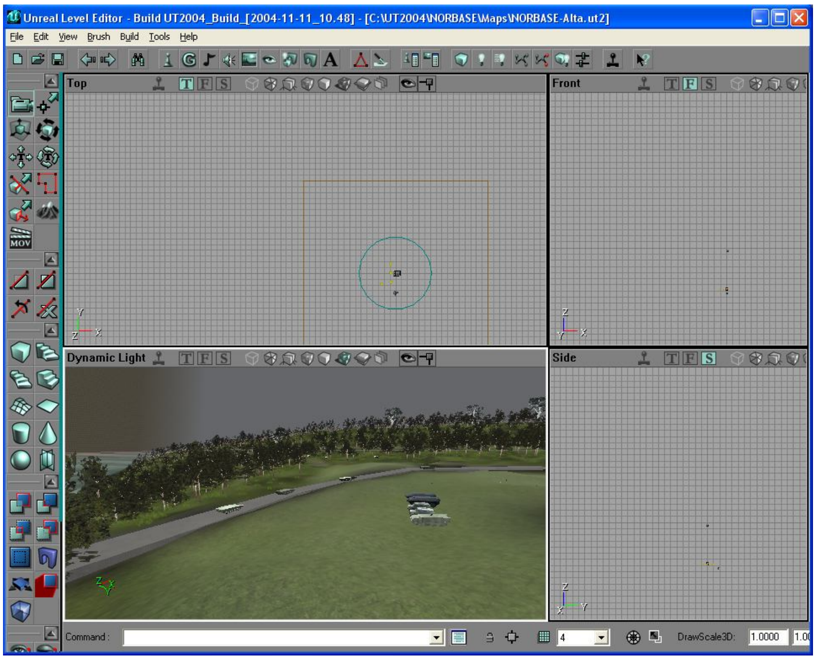
VTUAV Dragon Warrior
VTUAV Cypher
UH-60 Blackhawk
RAH-66_Comanche
CH-58 Kiowa
CH-53E Super Stallion
CH-46E Sea Knight
AH-64 Apache
AH-1W SuperCobra

Figure 61 VR-Forces 2D Editing

5.2.2 Unreal Tournament 2004

Unreal Tournament is a first-person shooter computer game by Epic Games and Digital Extremes. The game uses the Unreal Game Engine which has been the basis of many games (Figure 62). Unreal Tournament 2004 uses the Unreal Engine version 2.5, which basically takes care of the rendering of the 3D environment, together with a physics engine, collision detection (collision response), sound, scripting, animation, artificial intelligence, networking, streaming, memory management, threading, and scene graphics. By using the Unreal Editor it is possible to customize or modify the 3D environment, this includes terrain, material, mesh, animation and physics (Figure 63). Additionally you have editors for scripting, modification of the user interface and sound. To interconnect with other simulations UT needs an external simulation interface since HLA and DIS is not originally supported (101). The 3rd party software MÄK GameLink provides HLA for FFI's modified UT2004 environment called NORBASE.

Figure 62 Play Scene UT2004 NORBASE



Unreal Level Editor - Build UT2004_Build_[2004-11-11_10.48] - [C:\UT2004\NORBASE\Maps\NORBASE-Alta.ut2]

Figure 63 Unreal Editor NORBASE

5.2.3 Virtual Battle Space 2

Virtual Battle Space 2 is specialized commercial training simulation platform for different military training and experimentation scenarios. The software development is sponsored by the United States Marine Corps and United Kingdom ministry of Defense. The simulation kit is highly influenced by first-person shooter games, where different persons interact in a high resolution virtual battle space. The game engine is based on a first-person shooter game engine used for the game Armed Assault. The goal is to create realistic battle environments which emphasize high quality terrain and 3D resolution and has a built-in realistic physics engine for advanced damage models. The simulation kit simulates the real world in great detail, for instance the trees and grass move with the wind both produced by nature itself or by the rotor blades of a helicopter. The kit contains a real-time mission editor and a finite-state machine (FSM) editor for creating artificial intelligent movement. DIS/HLA functionality is provided by 3rd party software from Calytrix. An Application Scripting Interface (ASI) facilitates support for external AI. The simulation basically concentrated on a tactical level (102).

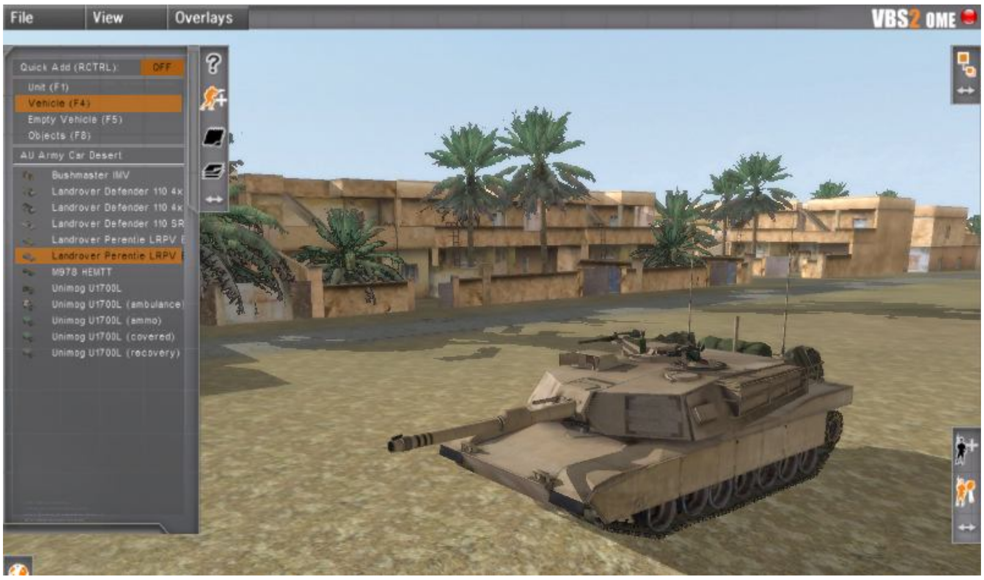
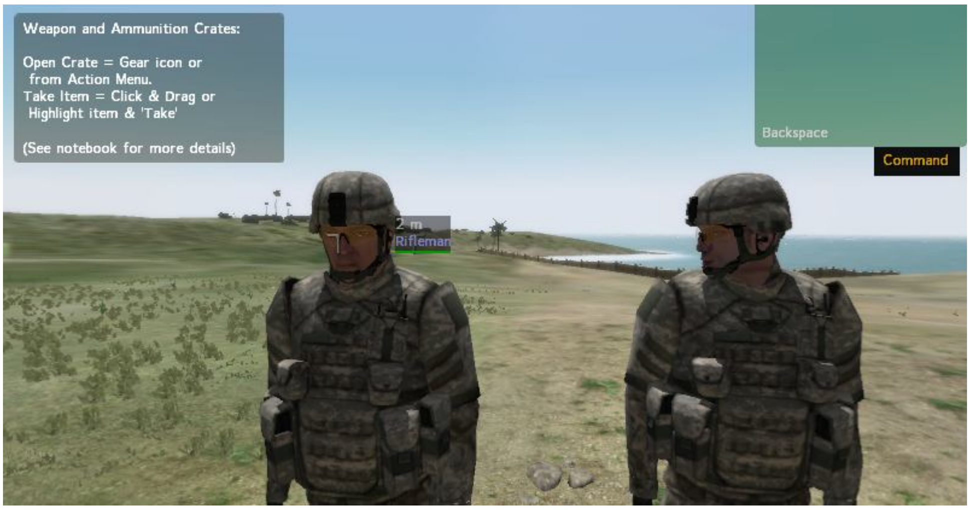


Figure 64 VBS2 Scene Editor



Weapon and Ammunition Crates:

Open Crate = Gear icon or
from Action Menu.
Take Item = Click & Drag or
Highlight item & 'Take'

(See notebook for more details)

Figure 65 VBS2 Gaming Scene

5.3 Interconnecting Simulations

Distributed Interactive Simulation (DIS) is an IEEE open standard and is both supported via commercial and open source implementations. The purpose of the standard is to support execution of real-time distributed simulations. The idea is to connect multiple computers together in a real-time simulation network. The High Level Architecture (HLA) is the successor of this standard and is documented in IEEE-standard 1516. HLA is a general purpose architecture which makes it possible for one simulation to communicate with another simulation. The architecture consists of three components, namely an interface specification, an Object Model Template (OMT) and HLA rules (103).

A simulation that is HLA compliant is referred to as a federate. A federation on the other hand is multiple federates connected via a HLA interface using a common Federation Object Model (FOM), meaning that a federation is a set of federates interacting. A collection of related data sent between simulations is referred to as an object and have attributes. Events sent between simulations are referred to as interactions and do also have parameters. A federate does not necessarily need to be a simulation, but can be an entity that needs access to the simulation data. One such example has already been mentioned, the MÄK Stealth which is a passive 3D viewer. In short the High Level Architecture constitutes a framework for interconnecting simulation frameworks (103).

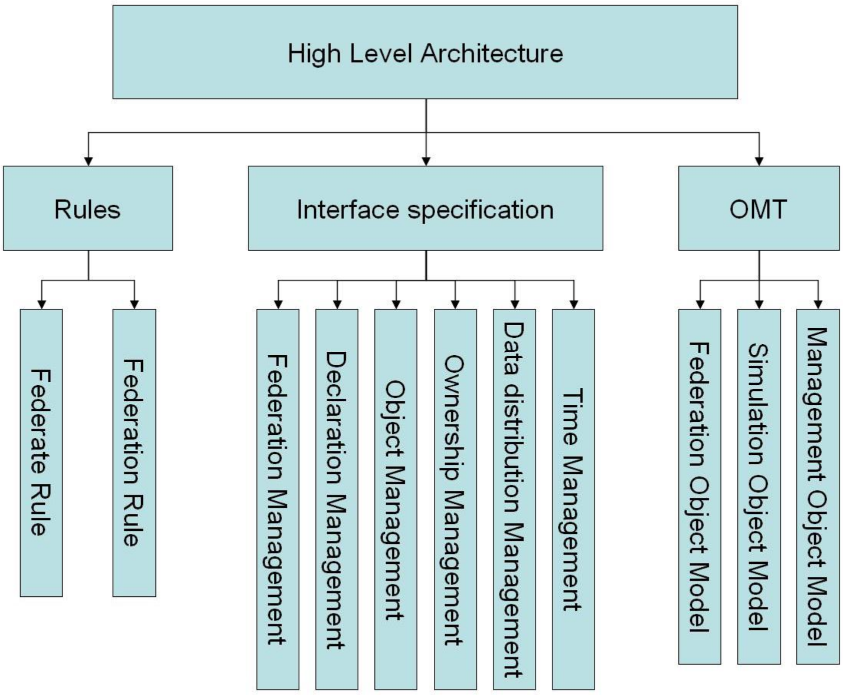


Figure 66 High Level Architecture (HLA), from (104)

Communication between simulations goes through an interface called the Run-Time Infrastructure (RTI) which is the implementation of the HLA interface specification. The RTI provides a programming library and an API. The OMT on the other hand specifies the format and structure of information sent between the simulations. The RTI software provides several services which amongst others are creation, control, modification and deletion of a federation execution. The RTI also manages multiple federates joining and leaving the federation execution, in addition to facilitating data exchange between federates (103).

The communication between the RTI and federates and within a federation is organized through two executive processes, the federation executive (FedExec) and the RTI executive (RTIExec). A third component is the RTI library which makes the HLA service methods available for federates. The methods are invoked by an interface called the RTI ambassador. Similarly the RTI can invoke methods in the federate using the federate ambassador, which is the interface that every federate must supply. There is one FedExec process for every running federation. The interface is visualized in Figure 67. The figure also shows the order of events when a user initializes a new federation within the RTI (105). RTI gateways are seldom used since they are not necessary in a normal configured network.

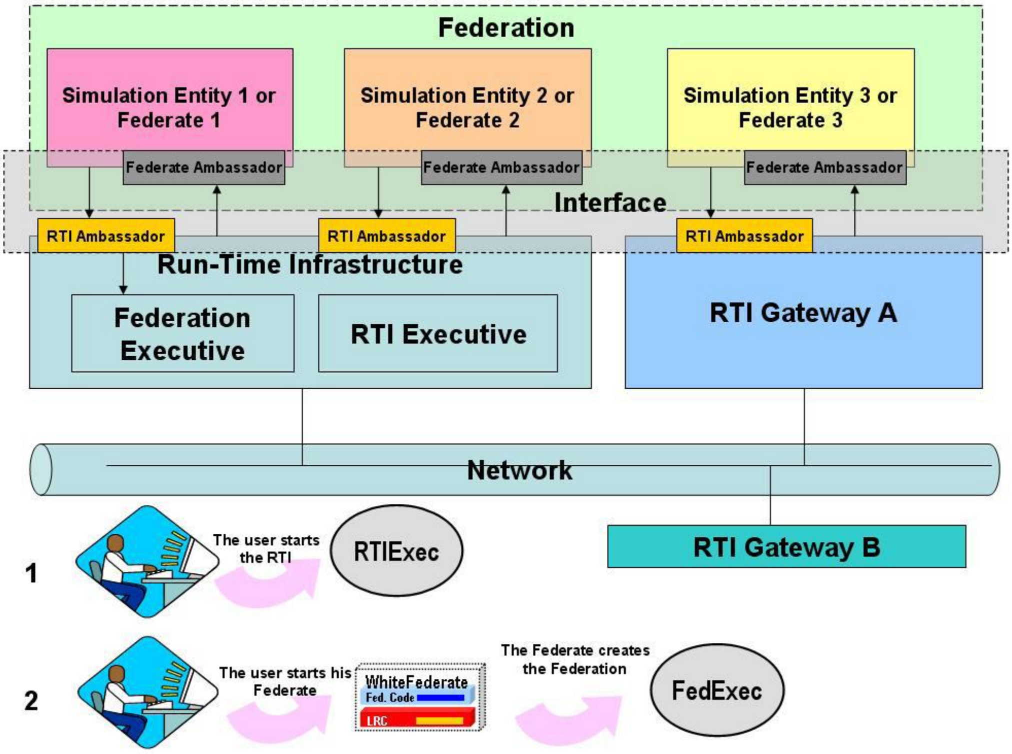


Figure 67 HLA Compatible Simulations, adapted from (105)

5.4 Agent Frameworks

There are many different types of frameworks for agent development and they are built upon different theories and approaches to artificial intelligence. An agent framework is a structure or platform that supports development of agents and is usually based around one or several architectural styles. Examples of such styles are the well known BDI architecture (94) and the subsumption architecture by Brooks (19).

The term agent framework covers agent development platforms, agent architectures and agent behavior models (Figure 68). An agent development platform is a structure that can support the life cycle of the agent and provide a communication interface for agent interaction. The agent behavior is in itself not implemented in any way. The agent development platform provides an API through which an agent can communicate with the platform (106). The agent architecture is the internal structure of the agent and constitutes a framework for creating behavior models (chapter 3.3.3). The behavior model provides the content of the architecture which in most cases is different forms of knowledge, for instance procedural and declarative. The notion of behavior is in some literature referred to as the result of the architecture and its content (93).

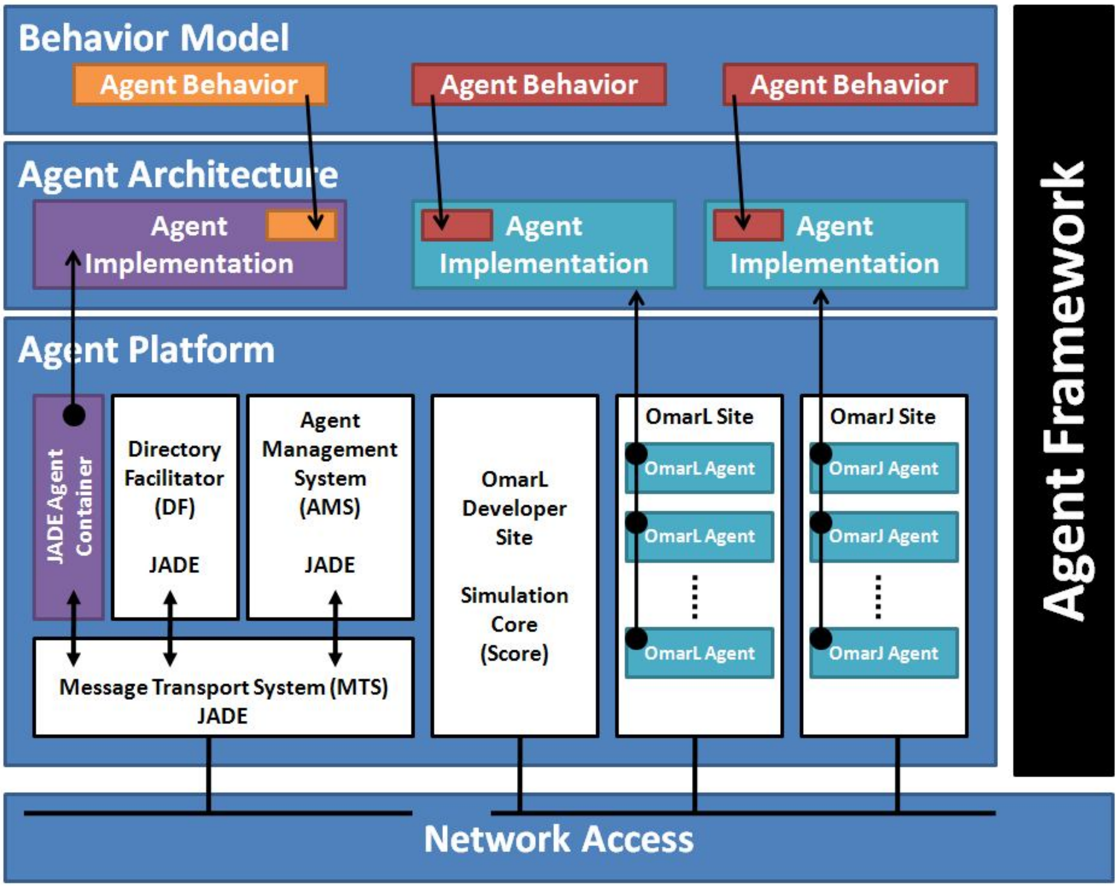


Figure 68 Agent Framework[7]

[7] The figures describing the agent platforms JADE and D-OMAR are structures fetched from respectively (106) and (64) the rest of the figure is synthesized from information available in chapters 3.3, 3.4, 4.6 and 4.7.

The Java Agent Development Environment (JADE) is one example of an agent platform. JADE is a platform for creating multi-agent systems. It contains a message transport system (MTS) that constitutes a network interface for developing distributed agent networks, an Agent Management System (AMS) that supervise agent access control to the MTS and a directory facilitator (DF) for creating distributed services (Figure 68). The agent is an instance running in the agent platform which means that the agent has a life cycle determined by the AMS. Each agent has a queue for sending and receiving messages. The instance of an agent constitutes a container for creating the agent's internal structure. Applying an agent architecture that facilitates reasoning involves implementing practical reasoning, deductive reasoning or cognitive agent architectures. The JADE agent platform is a middle-ware that complies with the specifications of the Foundation for Intelligent Physical Agents (FIPA) (106).

Another example of an agent development environment is D-OMAR. This development environment contains five components that have even more functionality than the relatively limited counterpart JADE. D-OMAR contains a procedural language component (the simulation core; Score), a time management component, an inter process communication component, an event recording component and an external communication component ensuring interconnectivity with other frameworks. D-OMAR is implemented both in LISP and Java, namely OmarL and OmarJ. The two implementations can operate together since they use the same signaling language and the same Score (Figure 68). The setup of D-OMAR in a network contains sites for OMAR agents (OmarL and OmarJ) and a developer site for the runtime environment (Score) (64).

5.5 Software for Developing Agent Behavior Models

In Chapter 4 several architectures for creating intelligent behavior was presented. This sub-chapter presents the software setup of the cognitive architectures presented in chapter 4.6 and 4.7. In addition three more architectures will be described, namely Simbionic, Jadex and JACK. The former is a commercial development tool for AI in simulations and games, while the two latter represent the BDI-architecture.

5.5.1 SimBionic

SimBionic is a commercial tool for creating AI in simulations and games (107). It provides an easy to use graphical user interface for constructing rule-based behavior through FSMs. The SimBionic application is divided in three components. The authoring tool which constitute the GUI, the runtime engine which execute the code created in the GUI and a simulation interface that connects the Simbionic runtime engine to the external simulation or game engine (Figure 69).

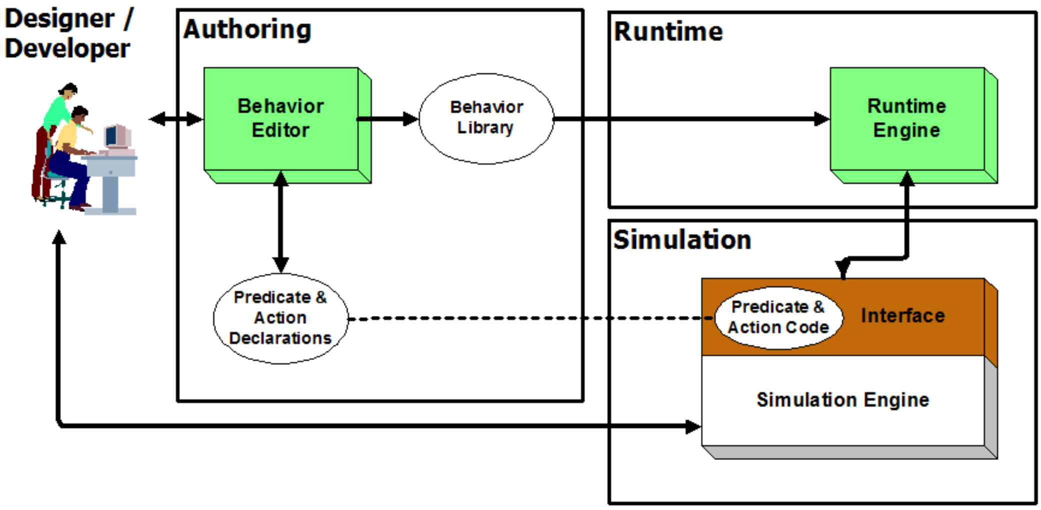


Figure 69 SimBionic Structure, from (107)

Through the editor one creates a behavior library that is available to the runtime engine. The interface to the simulation engine is available in two programming languages, namely C++ and Java (107). A feature-limited edition of SimBionic is available for free and this is the Simbionic Community Edition. Summary of software information is available in Table 21.

Software Information	
Name	Simbionic
Version	2.1.5
Date	30.12.2005
License	Commercial (Community Edition available for free)
Modeling Language	Visual Editor
Programming Language	C++ and Java
Support Tools	Visual editor, runtime engine and simulation interface

Table 21 Software Information Simbionic

5.5.2 Architecture for Procedure Execution (APEX)

Apex open source software is governed by NASA and used in several different NASA founded projects for producing intelligent behavior (chapter 4.6.1). Amongst others an Apex application has been developed that interconnects Apex agents with flight controls in the flight simulator X-plane. The software is freely available on the internet. The software packages does not include text editors for writing code, therefore one needs 3rd party software for writing the code. Interaction with external code is facilitated through writing a template or module that handles external connections. This has to be done in Lisp, while the behavior model is written in the procedure definition language (PDL) using the modeling technique known as CPM-GOMS (Cognitive, Perceptual and Motor – GOMS). Sherpa is a GUI for debugging and testing Apex applications (41). Summary of software information is available in Table 22.

Software Information	
Name	Apex 3.0
Version	3.0
Date	20.06.2006
License	NASA Open Source Software Agreement
Modeling Language	Procedure Description Language (PDL), technique CPM-GOMS
Programming Language	Lisp (Allegro Common Lisp)
Support Tools	• Sherpa v3-0-11-1 – creation, debugging, demonstrating and analyzing apex Applications. • Emacs is the recommended text editor. • Graphviz is needed for viewing diagrams in Sherpa

Table 22 Software Information Apex

5.5.3 Atomic Components of Thought – Rationale (ACT-R)

The ACT-R architecture has its foundations in the academic community and it still has a strong academic belonging (chapter 4.6.2). Even though the architecture has been used on an experimental level in both military and civilian applications the development tools consists of a package of different 3rd party open source software, meaning that one needs to download a compatible Lisp distribution and a text editor (83).

The ACT-R Environment Application provides facilities for debugging and code running. The code is written in Lisp using the ACT-R libraries. External access is facilitated by using a built-in device library or creating your own, but it is no secret that interconnecting ACT-R with an agent platform or simulation is far from trivial. Problem areas are communication via Lisp objects and time synchronization. Time is specifically difficult due to timing modes like real-time versus virtual time and event-based versus tick-based. The integration work of the model exceeds the time spent on creating the ACT-R model. At the moment there is work being done to provide an API for interfacing external applications with ACT-R. These results are a direct consequence of trials that the American army has run when they tried to interconnect Unreal Tournament and ACT-R. The project is known as the UT MOUT and is referred to as a worst case scenario for ACT-R integration (108). Summary of software information is available in Table 23.

Software Information	
Name	ACT-R 6
Version	1.3 [r617]
Date	11.06.2008
License	LGPL v2.1
Modeling Language	ACT-R model files written in Lisp
Programming Language	Lisp (Allegro Common Lisp)
Support Tools	• ACT-R Environment Application – debugging, demonstrating, analyzing and running ACT-R models. • Emacs is the recommended text editor. • Eclipse plug-in - jACT-R • Lisp application (ACL 6.2/7.0/8.0, MCL 5.0/5.1, etc) • A standalone version of the ACT-R Environment is available

Table 23 Software Information ACT-R

5.5.4 Business Redesign Agent-Based Holistic Modeling System (Brahms)

Brahms is the result of a master thesis written by Maarten Sierhuis and can be said to be academically founded. Nevertheless the software has been licensed to NASA by Agent iSolutions and is not commercially available. Brahms constitutes its own programming language with its own editor, compiler and virtual machine. The programming language is called the Brahms programming language and is specified in a language specification document. By downloading the Brahms Agent Environment one get all the tools that come with Brahms. To run the Brahms Composer and create models database software is needed, Brahms uses MySQL 4.1.x or MySQL 5.0.51 (43).

External access to Brahms agents and the Brahms environment is provided through java activities. These are specific activity types that are built with java code that gives access to the model internals. With this code it is possible to access the beliefs of an agent or an object etc. Summary of software information is available in Table 24.

Software Information	
Name	Brahms Agent Environment
Version	1.2.7
Date	29.05.2008
License	Software is licensed to NASA, freely available for research or non-commercial purposes.
Modeling Language	Brahms Language Specification v2.17
Programming Language	Brahms Language
Support Tools	• Brahms comes with a Brahms user interface called the Brahms Composer - debugging, demonstrating, analyzing and running Brahms models. • Brahms Compiler • Brahms Virtual Machine • Exists an Eclipse plug-in

Table 24 Software Information Brahms

5.5.5 Executive Process/Interactive Control (EPIC)

EPIC is academically founded cognitive architecture that emphasizes sensory processing. Production rules and parameters are set in files with the extension *.prs. The file defines the initial contents of the working memory and the production rules of the system according to an EPIC specific syntax. The file is a simple text file and you need a plain-text editor for the file creation. The parameter settings for the different components are preset by the system, but can be altered by defining them in the model files. The procedures follow the modeling technique called GOMS (90).

Collaboration with external environment is possible by the creation of devices. A device is a C++ class that inherits its methods from the device processor, which constitute an interface between the EPIC environment and a specific device. By overriding the virtual functions it is possible to take over the handling of device input events (109). Summary of software information is available in Table 25.

Software Information	
Name	EPIC Application
Version	2.0
Date	18.07.2004
License	Available for non-commercial use and research purposes only.
Modeling Language	Production rule statement files *.prs (C++)
Programming Language	Previous version in Lisp, last version in C++
Support Tools	EPIC application - debugging, demonstrating, analyzing and running EPIC models.

Table 25 Software Information EPIC

5.5.6 Soar

The Soar architecture is another academically founded cognitive architecture. Soar uses its own modeling language for describing the agent knowledge and productions rules. Multiple agents can be run in the Soar kernel at the same time. The agents need to have unique names within the kernel. Agents are running inside a wrapper class that determines the running mode of the agent. Examples of running modes are run forever and run for n phases. Soar agents can be run from files or through a command line interface. The command line interface also provides commands for information tracing, debugging, parameter configuring and I/O commands (92).

External accesses to the Soar kernel can be obtained through several different methods amongst them are gSKI, SGIO and SML. The gSKI (generic Soar Kernel Interface) is a tool for providing a mapping between external objects and objects in the Soar input and output links using C++ function calls. The SGIO (Soar General Input Output Library) is library that support mapping between external objects and objects in the Soar input and output links using C++ function calls or data transmission over sockets (deprecated). SML (Soar Markup Language) is a new interface to the gSKI using XML instead of C++ (110).

The possibilities of interconnecting Soar with external environments are extensive. The soar community supports API’s that interconnect programming languages like Java, Lua and C++, in addition to the scripting language TCL.

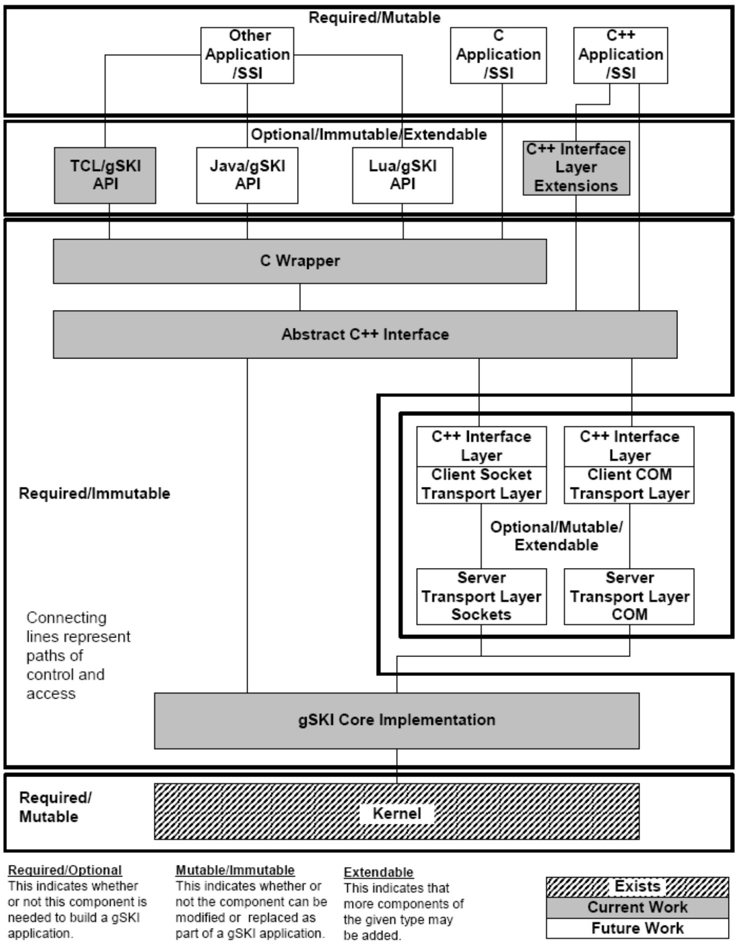

Required/Mutable
Other Application /SSI
C Application /SSI
C++ Application /SSI
Optional/Immutable/Extendable
TCL/gSKI API
Java/gSKI API
Lua/gSKI API
C++ Interface Layer Extensions
C Wrapper
Abstract C++ Interface
Required/Immutable
C++ Interface Layer Client Socket Transport Layer
C++ Interface Layer Client COM Transport Layer
Optional/Mutable/ Extendable
Server Transport Layer Sockets
Server Transport Layer COM
Connecting lines represent paths of control and access
gSKI Core Implementation
Required/ Mutable
Kernel
Required/Optional
This indicates whether or not this component is needed to build a gSKI application.
Mutable/Immutable
This indicates whether or not the component can be modified or replaced as part of a gSKI application.
Extendable
This indicates that more components of the given type may be added.
Exists
Current Work
Future Work

Figure 70 gSKI and Soar Systems, from gSKI summary (70)

The amount of software available to the Soar community is extensive due to that there are commercial organizations that produce Soar applications (111). This has lead to cooperative efforts between the academic community and the commercial community where software is released both ways. The commercial community release modeling tools and the academic community expand the functionality of the Soar kernel. The result is an active Soar community. Summary of software information is available in Table 26.

Software Information	
Name	Soar 8.6.3
Version	8.6.3
Date	31.10.2006
License	BSD
Modeling Language	Soar Programming Language
Programming Language	Soar Programming Language
Support Tools	• Java Soar Debugger. • Visual Soar for editing Soar code. • Eclipse plug-in for Soar programming. • SoarDoc for generating javadoc like soar documentation.

Table 26 Software Information Soar

5.5.7 BDI architectures

There are many different BDI implementations and their characteristics are different due to that they are implemented in different programming languages and that the modeling regime is conceptually different. One can group the different BDI architectures according to these differences in characteristics: Java-based agent languages, BDI languages and agent simulation languages.

	Java-based agent languages		**BDI languages**		**Agent simulation languages**	
	BDI-based	**Imperative**	**Goal-based**	**Subsumption**	**BDI-based**	**Imperative**
Brahms				X	X	
Jason			X			
AgentSpeak			X			
JADE		X				
JACK	X					
Jadex	X					
Swarm						X
Repast						X

Table 27 Comparison of Agent Oriented Languages, from (88)

The comparison is done according to how the architecture operates when reading the code. Procedural programming (imperative) is a programming paradigm that describes computation in terms of program state and statements that change that state, while a declarative programming paradigm describes beliefs and procedures that are triggered when certain states of the belief set is satisfied (preconditions) (88). This chapter will concentrate on introducing Jadex and JACK.

5.5.7.1 Jadex

Jadex is an open source academic implementation of the BDI architecture that uses the JADE agent platform for creating networked communication events. The agent description is defined in a XML file and the intentions or plans are defined in java classes. The XML file is read by the BDI reasoning engine where the goals, beliefs and events are instantiated. Using a XML file for defining agents is an advantage when creating mobile agents. This means that an agent can move from one node in a network to another (112).

The Jadex software package contains support tools for testing most aspects of the agent. The Jadex Control Center (JCC) contains several tools for investigating the contents of the agent during runtime, testing communication, tracing events and goals, testing classes and investigating dependencies between goals in an exploration graph. Jadex agents can communicate with JADE agents and there is a plug-in for managing JADE agents in the JCC (113).

Developing the knowledge model of Jadex agents is supported through Protegè which is an application for creating and editing ontologies and knowledge bases. The Jadex distribution contains a plug-in to Protegè called Beanynizer that generates java classes from the ontology created in Protegè (113). Summary of software information is available in Table 28.

Software Information	
Name	Jadex – BDI Agent System
Version	0.96
Date	15.06.2007
License	LGPL
Modeling Language	XML and Java
Programming Language	Java
Support Tools	• Jadex Control Center – test and debug agents. • Beanynizer – creation of knowledge ontologies in Protegè. • JADE Adapter – for conversation with JADE enabled agents. • Recommended text editor is Eclipse

Table 28 Software Information Jadex

5.5.7.2 JACK

JACK is a commercial BDI distribution that contains a development environment for BDI agents and BDI team agents and more. JACK supports the building of agent and team-based applications by using the graphical JACK development environment (JDE) (114).

The BDI engine is not a cognitive founded structure which is why there has been developed an extension to the JDE that adds cognitive functionality. The extension is called CoJACK and is based on several of the same principles as ACT-R. CoJACK introduces mechanisms to JACK that makes the decision process of the agents founded in cognitive theories. This includes focus of attention, memory retention and recall, as well as timing and errors. PMFs are included in the CoJACK moderator layer which provides the means of reconfiguring human factor parameters like fear, morale, leadership and fatigue etc. An extra feature of the CoJACK plug-in is the situation awareness (SA) model that provides a set of tools for creating moderated cognitive models. CoJACK uses the Endsley model of SA, where the concept of SA is divided into three discrete levels of sensation (chapter 2.4). CoJACK connects the beliefs in the agent model with the SA levels. When new beliefs are created a set of rules for tagging the beliefs with a compliant SA levels is applied (11).

Software Information	
Name	JACK
Version	5.3
Date	01.06.2008
License	Commercially licensed software
Modeling Language	JACK Agent Language
Programming Language	Java Based
Support Tools	• the Graphical Plan Editor and JDE browser • the Plan Tracing Tool and Design Tracing Tool • the Compiler Utility • the Design Tool

Table 29 Software Information JACK

Chapter 6 Case Studies

6.1 Introduction

This chapter presents three different case studies where components of AI are needed. The cases resemble research areas that FFI find interesting to investigate, being an important step towards creating a simulated training environment for military forces. The main purpose of presenting these cases is to shed light on different aspects of implementing behavior in a SE. The abstraction level or the resolution level of an implementation impacts the complexity of the behavior model. The chapter presents two cases that deal with crowds and these cases have different resolution levels. These case studies shed light on problems that arise on the background of increasing the resolution level. The last case involves human interaction with synthetic entities. The case study takes a closer look at problems that arise when human operators interact with synthetic entities. The case studies are:

- Population in a Virtual World (C1)
- Crowds and Terrorists in Urban Environments (C2)
- Virtual Team Mates for a Human in the Loop Leader (C3)

6.1.1 Analysis Approach

Every case is first described through a problem description and a problem analysis, which explains the problem domain of the case. Then the problem is analyzed on the background of the PEAS perspective (Performance, Environment, Actuators and Sensors) also known as the task environment (5 pp. 38-44).

The more complex decisions the agent has to make, the more complex the architecture of the agent needs to be, therefore on the basis of the analysis an agent architecture approach is presented. For more information on agent architectures see chapters 3.3, 3.4, 4.6 and 4.7.

6.1.2 Architectural Approach

For each case a conceptual architecture has been synthesized on the background of the information acquired during the PEAS analysis. The theoretical foundation for the analysis is to be found in Chapter 3. In addition information on crowd simulation and HITL simulations has been extracted from a number of articles on the subject. Due to the fact that the analysis use information from several sources which is tightly interlaced or combined it is hard to pin point a reference in every case. Book literature that has been used for supporting the analysis is (97). Articles that are relevant for crowd and group behavior are (32), (33), (38), (115), (116), (117), (118) and (119).

6.2 C1: Population in a Virtual World

6.2.1 Problem Description

Development of entities that can simulate a civilian population in such a manner, that it is perceived to have a close resemblance to the behavior of a real population, is still on the experimentation level. Training and rehearsing for missions that involves a civilian population means that you basically have to populate the virtual representation of a world location with civilian entities or agents. The problem consists of designing a simulation component, which can model or simulate the civilian population in any given area. This includes models for movement, cognition and damages. If possible the population should be generated from demographic maps, which contains information about population density amongst others. The use of or refinement of COTS or any other available tools for solving the problem is recommended. The goal is to train military leaders and troops to consider the impact an operation has on the local population and be able to review the consequences if they do not.

6.2.2 Problem Analysis

What is needed to implement a realistic civilian population? The individuals need to be interacting with the environment in some basic ways. Simple interaction includes patterns of movement and proper reactions to emerging crisis situations like someone shooting, a bomb going off or an eruption of a fire. The population also needs to interact with the arriving military force in some limited ways, for instance shooting a civilian or bombing a building should have impact on the relations with the local community. A mental model for the local population's attitudes and feelings toward the military force needs to exist both for evaluation and control purposes. This could be evaluation of the military personnel participating in the Computer Assisted Exercise (CAX), and control and review of events that happened during the exercise.

6.2.2.1 Performance Measures

How can the civilian population know where they are going and why? A goal gives a purpose to the movement of all civilian entities. Examples of this can be that most entities are assigned roles in their everyday life. This implies movement to and from home and school or work. *What should the movement algorithm consider?* In a city there can be different areas that are considered to be dangerous, which needs to be avoided. There can be a road block and another route needs to be chosen. General wandering should also be implemented. A movement model needs to cover basic movement kinetics and path-finding.

How should the entity or agent think or reason? The agent should reason about where to go on which hours of the day, where to stay for certain amounts of hours during the day and be assigned a home or place to stay overnight. The reasoning around the movement is confined to the roles added to the agent.

How should the local population show their discomfort or dislikes of the military force? Each agent can gather information about the local life through their limited scope of perception and when all information is put together it is possible to derive a statistical model for measuring the impact on the local population. The question is how much reasoning should be placed in the individual agent and what information should be sent to a potential centralized statistical reasoning agent.

How should environmental and individual damages be assessed and evaluated? Damages that are relevant for the simulation are those imposed by the military force and that affects the local population. The locals can be injured by actions taken by the military force, which imply the need for a state that describes the agent's health. Having a health measurement makes it possible to include hospital access and general health situation in the simulation. Damages on buildings or areas are important to estimate the level of hostility in the local population. Presumably there is often a relation between the hostility in an area and the amount of destruction of local buildings due to the fact that those buildings provided housing and work. Damages can also be seen as unwanted influences, for instance a blockade that stop food supplies or stop raw material to arrive at local industry building, reducing the number of jobs available. Basically the simulation should give feedback on the health status, unemployment status and other effects on social well being of the population.

6.2.2.2 Environment

The virtual environment can be versatile due to the fact that it can be rural or urban. The building areas could be divided into types, for instance residential, commercial or industrial buildings. The importance of the buildings, for the local community, varies with the type of building. High density of buildings imposes bigger risk for a military force, both on a tactical level and on the ability to enforce effective crowd control.

The environment is, from the agent's point of view, partially observable and there are some uncertainties in what the next world state would be, therefore the environment is considered to also be stochastic. The agents do not perform any explicit planning, but follow certain patterns of behavior therefore the environment can be said to be sequential (chapter 3.3.4.1), since all events that involved the agent in the past will determine the future actions. An agent can be influenced by other agents and players, also by changes in the world state. A decision to go to work might be altered if a crisis situation occurs, meaning that the environment in many aspects is dynamic, continuous and inhabited by multiple agents.

6.2.2.3 Actuators

The local population must be able to show its discontent or appreciation of the presence of military forces, either by protesting or being helpful. Actions taken by the military force need not be played out in the SE and a kind of textual or menu based communication panel could be introduced, which is directed toward the group agent. This panel would then communicate military imposed restrictions on the population via the group agent.

6.2.2.4 Sensors

The sensors are in this case dividable into a logical and physical part. The individual agent can perceive events happening in their proximity through simulation of human senses. These events triggers reactive behavior based on situations in the physical domain. The logical sensory inputs are in this case described by collecting each individual's impressions of the surroundings and perform statistical calculations. For instance declaring a state of martial law and introducing curfews will affect the local support, which again affects the amount of destruction and injuries the population can sustain. Levels of hostility will rise if one looses a family member or someone close. The amount of help the local population receives could be either conflict reducing, if given help is adequate, or conflict enhancing if it is too low. The logical part could be implemented in the form of a group agent or population agent.

6.2.3 Architecture

The PEAS analysis has revealed the need for access to terrain data and the implementation of character roles. Each agent contains a movement and damage model. The population agent contains models for group behavior (chapter 3.2.9, 3.4 and 4.3.6) and statistical analysis. The need for reactive behavior is highly limited. A conceptual architecture is presented in Figure 71.

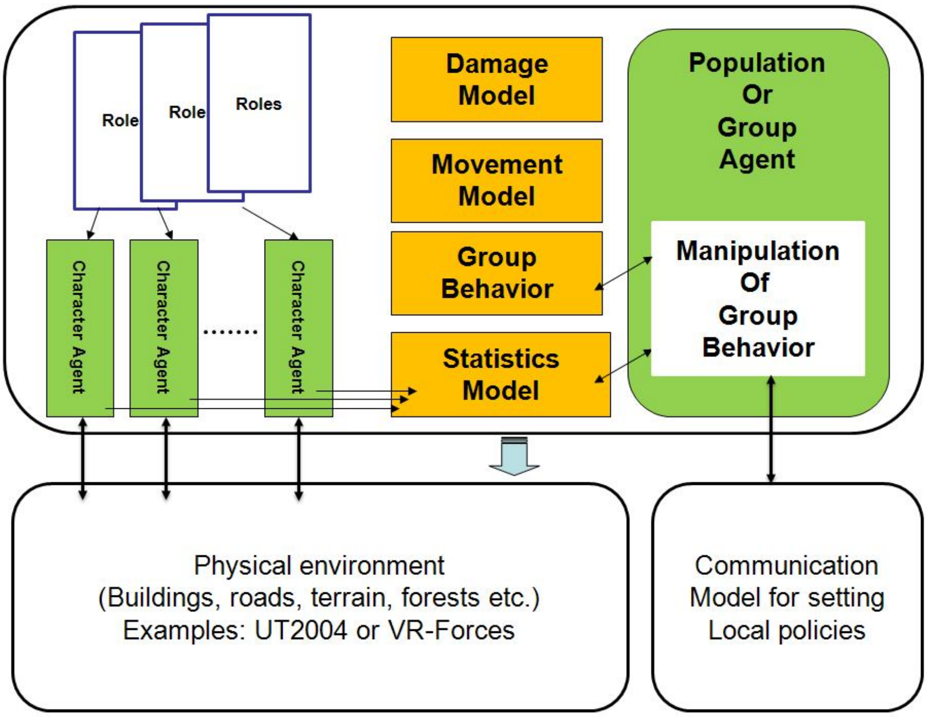

Role
Role
Roles
Character Agent
Character Agent
....
Character Agent
Damage Model
Movement Model
Group Behavior
Statistics Model
Population Or Group Agent
Manipulation Of Group Behavior
Physical environment (Buildings, roads, terrain, forests etc.) Examples: UT2004 or VR-Forces
Communication Model for setting Local policies

Figure 71 Synthetic Population

6.3 C2: Crowds and Terrorists in Urban Environments

6.3.1 Problem Description

To be able to train skills in crowd control you need a simulation component that can simulate a certain amount of people acting in a crowd. The crowd can be influenced by external events imposed on them. The influence can be imposed by a military force or some local pressure group. The intention is to interact with the crowd and be able to train the skills in crowd control and how to handle demonstrations. The use of or refinement of COTS or any other available tools for solving the problem is recommended. A relevant event to consider is the Meymaneh scenario in Afghanistan, where the Norwegian ISAF forces were attacked by demonstrators equipped with anti-tank missiles and grenades. The goal is to be able to train military troops in handling crowds and local influence groups and be able to review the consequences if they do not.

6.3.2 Problem Analysis

What do we need to consider when we want to create a virtual crowd? The crowd needs to respond on actions taken on it by some third party. There should be both a collective and individual mindset, which means that people might join the crowd or leave the crowd. The architecture must model measures that will disperse the crowd and that will lead to a crowd growing. The crowd should react in a manner of increasing or decreasing levels of aggressiveness (chapter 3.2.9, 3.4.1, 4.3.5, 4.3.6 and 4.5.2). This problem contains many of the same problems that are identified in C1 chapter 6.2.

6.3.2.1 Performance Measures

The agent should reason about basic reactive behavior, for instance not hitting anybody or an object while moving, walk in a sensible manner and react to changes in the environment. When certain criteria for a situation are fulfilled, escape behavior like fleeing or hiding can be triggered. C2 can be seen as an extension to C1.

How can the character agent know if it wants to join or leave a crowd? The agents need some way of evaluating the need for joining a crowd. The choice must be based on functions affecting the individuals and their relations to other group members, for instance a Bayesian value network, which means that the character's values and personality characteristics are weighted in a Bayesian network (32; 33). There can be some individual characteristics that lead to joining or starting a crowd. This could be individual emotions, for instance resentment toward the military force or some strong connections to other group members that are joining the crowd.

How will the agents or the group plan movement? There is a need for a simple basic individual movement model that takes care of wandering, avoiding collisions and alters speed rates. Group movement means that certain individuals lead the group or that the movement is based on some hidden consensus. This means that there is a need for a model for followers and leaders (32). When moving in a collective you need models that take care of flocking and alignment movements (38).

How will the crowd react? We have seen that there is a need for individual behavior models, but there is also a need for a collective behavioral model. When acting in a collective the behavior of the individuals alters itself, often the actions or expressions of the collective is more extreme than it would have been for the individual alone. Dependent on the actions taken by the military, the crowd will react in some way. A model for simulating different aggression levels is needed; this includes actions on each aggression level (119).

How does the crowd come into being? A local leader or a very upset local individual might start throwing rocks or show obscene gestures. This is hard to perceive in a simulated reality, therefore the creation of a crowd could also be based on some threshold values that is superseded. Protests are usually a result of some latent issues or problems imposed on the local population. The more people that are frustrated the easier it is creating unbearable situations for a coalition force.

6.3.2.2 Environment

The crowd needs to have a goal or destination of attack. This can be a military installation or a building of some sort, which means that there must be some objects that can be targeted. The targets are provided by the environment and the military force. The crowd reads its combined perception from member characters and therefore the group perception is a joint perception.

The agents do not have full control over the environment. Events are based on some probabilistic data and the world state, which means that the environment is considered to be stochastic. The goal could be derived from some local interest group or foreign influence. Future actions are dependent on former actions, and therefore the environment is sequential (chapter 3.3.4.1). The environment is both physical and logical. Physical in the sense that some events are based on physical events going on in the synthetic world or logical in the sense that agents functions as inputs for group knowledge. Other influences that impacts group behavior are the cultural and religious views. The environment is therefore considered to also be highly dynamic and continuous, containing multiple agents acting on different levels.

There is basically a need for modeling cultural values, religious values, economy, local resources and political fractions, in addition to leadership and inter/intra group relations.

6.3.2.3 Actuators

How can the crowd affect its surroundings? The crowd can affect its surroundings through weapons or objects it has picked up on the way. Action examples are setting cars on fire or houses on fire. The group is not a singular entity therefore it is acting through its members, which means that it can do whatever actions the members can do, but in a distributed way. For instance a leader can use its followers to attain some political goals. In this sense the followers are the leader's actuators. The actuators are both physical and logical. Physical in the sense of actions performed by the crowd and logical in the sense that emotions influence the actions of the group and it members, these values are not tangible and therefore considered to be logical.

6.3.2.4 Sensors

In which ways can the crowd perceive the situation? The crowd is a group of people consisting of individuals, which means that the sensory input is a joint input. The crowd could be able to differ between use of weapons against it that are lethal or non-lethal. It could be possible to do some limited reasoning with the group, through some textual interface, with the goal of finding out what is the problem or what can be done to help. Sensory input might be logical as well as physical. Logical inputs might be inputs from local radical groups or news on some events taking place in another country. In connection with the leader using the followers to achieve a goal, the followers also give information to the leader or group members. In this sense the followers or group members function as event input entities.

6.3.3 Architecture

The PEAS analysis revealed that C2 can be seen as an extension to C1. The consequence being that C2 is a simulation of a limited number of individuals in the population with higher reasoning capabilities. This implies social, mental and reactive models within the agent architecture that is influenced by personality and group memberships. Resources can be seen as objects that groups desire to control. A conceptual architecture is presented in Figure 72.

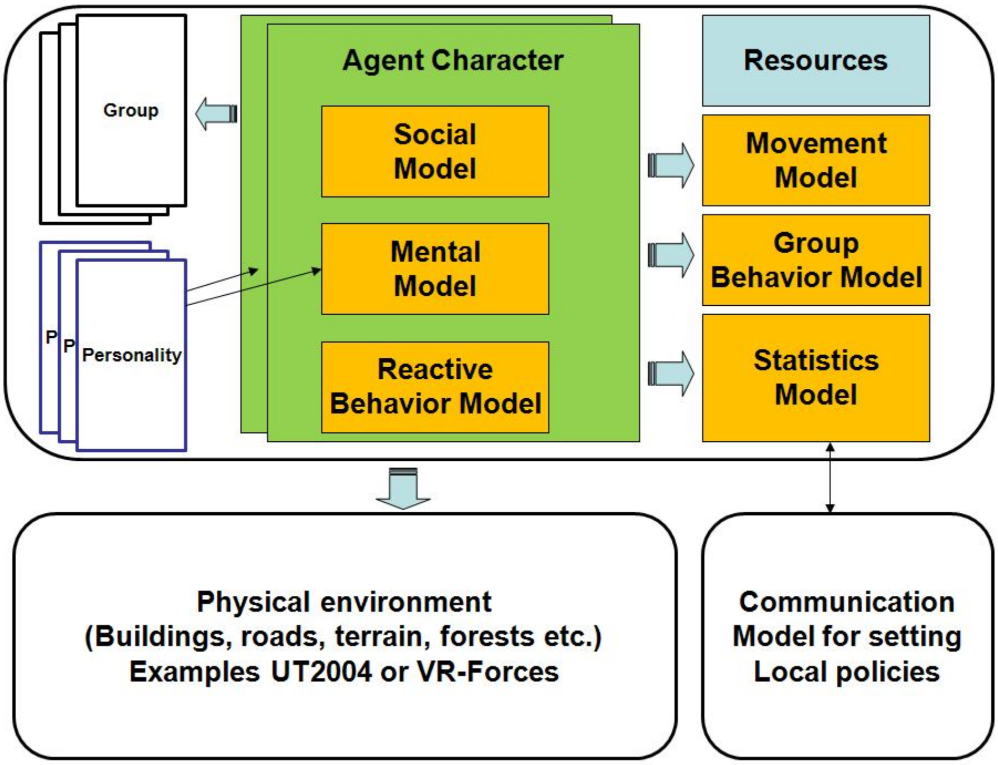


Figure 72 Crowd Control

6.4 C3: Virtual Team Mates for a Human in the Loop Leader

6.4.1 Problem Description

Educating and training team leaders is important due to the fact that it takes time and experience to develop an intuitive understanding of the battle space. Training future leaders in a virtual battle space with artificial team members has become even more important in the last decade, due to added complexity of the equipment and increased demands on speeding up the military decision process. The assignment is to identify a simulation framework that can simulate a number of team mates. Typical characteristics of team mates are the abilities to follow the leader in any given formation, be able to engage the enemy and perform some evasive and tactical maneuvers. The use of or refinement of COTS or any other available tools for solving the problem is recommended. The goal is to be able train squadron leaders in tactical maneuvering and increase their ability to synthesize a high degree of SA in a relatively short time frame, and be able to review the consequences if they do not.

6.4.2 Problem Analysis

How do we create a sensible virtual team mate? The team mate must be able to read some basic information from the state of the environment and inform the leader of the situation. A team’s role is to give the leader all the information needed in the interest of being able to take the correct actions on the background of the mission goals. The team should be able to interact both on visually perceived actions and receive orders through a simulated communication device.

6.4.2.1 Performance Measures

How can we evaluate the performance of the team mate? We need some movement modeling for basic maneuvering and for tactical and strategic maneuvering. There is a need for both tactical analysis and terrain analysis when exercising maneuvers. Included in these models is situation analysis (120; 121; 122). Terrain models for identifying tactical locations together with attack or defensive patterns developed on the basis of the terrain will enhance the team mate's decision making. There should be a communication model that simulated the real life leader to team communication as well as communication between team mates. A component that analyzes the actions of the team leader would ensure feedback on performance (123). The team mates need to be sensible, responsive and interpretable.

How should the team mates act and how independent should they be? The decisions made by the leader is based on his or her SA and if the team mate has better awareness the one closest to the situation should make the final call or at least be able to influence the decisions by informing the commander of what is going on (122). Military operations are conducted according to certain rules and procedures and the team mate should be aware of these guidelines and follow them. This could be field manuals, SOP (124) or ROE (125). To avoid misinterpretations of the situation a knowledge model that facilitates grounding (chapter 3.2.1) should be implemented.

What happens if the tank is damaged or disabled? The team mates can have different degrees of damage on their vehicles or aircrafts, which means that a team mate can slow the team down. There should be some way of making evacuation orders or repairing the vehicles.

6.4.2.2 Environment

The environment consists of the physical aspects of the terrain, including roads, buildings, farmland, airspace etc. Other factors that can be considered to be a part of the environment are the orders given by the team leader and the mission directives.

The agents do not have full control over the environment. Enemy activity and the movement of the leader can in most cases not be predetermined, which means that the environment is considered to be stochastic. Future actions are dependent on former actions, and therefore the environment is sequential (chapter 3.3.4.1). The environment is both physical and logical. Physical in the sense that some events are based on physical events going on in the synthetic world or logical in the sense that agents can receive information through communication acts. The environment is considered to be highly dynamic and continuous, containing multiple agents acting on different levels.

6.4.2.3 Actuators

The team mate basically affects the environment by exercising movement and by targeting enemies. Since the environment for a vehicle or aircraft also includes the other team members, the communication abilities must be looked upon as an actuator. Communication events sent between team mates influence the deliberation process of the agents, ultimately changing their behavior.

6.4.2.4 Sensors

Sensors are simulated through the synthetic environment. Most military vehicles or aircrafts have targeting systems and target detection systems. This can be either infrared sights or line of sight detections. The sensors available to the agent are dependent on the resolution level of the behavior model. There will be a great difference in available sensors if one chooses to model the personnel within the vehicle rather than the vehicle itself. Common SA is achieved by sensed confirmations of behavior patterns or by exchanging knowledge through communication devices.

6.4.3 Architecture

The PEAS analysis revealed that virtual team mates require a knowledge model that support common understanding of the situation. This includes a situation analysis that takes into account several different environmental variables. The knowledge model must also support the transformation of the leader’s intent into a format that is understandable for the artificial intelligent entities. This format should facilitate man-machine collaboration. A conceptual architecture is presented in Figure 73.

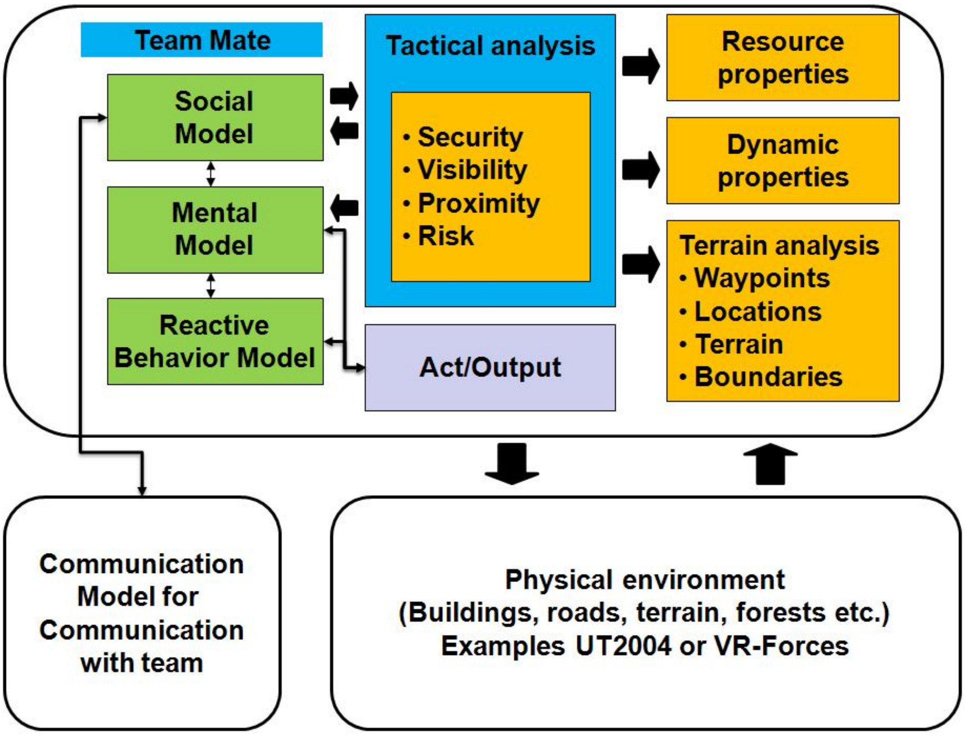


Figure 73 Virtual Team Mates

Chapter 7 Development of Demonstrator

The purpose of this chapter is to describe the development process of the demonstration software. Making a framework for controlling entities in a synthetic environment shed light on many of the problems that arise when trying to incorporate cognitive architectures or practical reasoning agents in to a simulation framework.

7.1 Demonstrator Framework

Chapter 6 describes three different cases where there is a need for artificial intelligent agents. The demonstrator is based on case C3. The operational scenario is set to be a Norwegian tank platoon in for instance Stridsvogneskadron 2 (Tank Company) which is a part of Panserbataljonen (Armored Battalion) at Setermoen, Troms.

On the background of the HBR evaluation and simulation framework evaluation in chapters 4.6, 4.7 and 5.5 I have chosen to implement a light version of C3 with the open source BDI architecture Jadex. The synthetic environments regarded as relevant for this demonstration are presented in chapter 5.2; ultimately we decided to use VR-Forces. The game controls for controlling the HITL contestant is fairly limited in VR-Forces, so the consensus became that one should try to use UT2004 for platoon leader control, but due to incompatible terrain databases the test scenarios where run in VR-Forces.

Software that has been used in this implementation is described in the tables below. Software provided by FFI is presented in Table 30 and other software used is presented in Table 31.

Software	Description	URL
Camtasia Studio version 5	The screen capture and video editing tool.	www.SoftwareCasa.com/Camtasia
Unreal Tournament 2004	Game control for platoon leader.	http://www.unrealtournament2003.com/
MÄK Game Link HLA version 1.3 or 1516 RPR FOM 2.0 draft 17	The HLA interface used by UT2004. The software package is no longer supported.	
MÄK VR-Forces HLA version 1.3 or 1516 RPR FOM 2.0 draft 17	The synthetic simulation environment.	http://www.mak.com/products/vrforces.php
MÄK RTI version 3.1/3.2 HLA version 1.3 or 1516 RPR FOM 2.0 draft 17	The run-time interface used to interconnect the different simulation entities.	http://www.mak.com/products/rti.php
MÄK Stealth 3D Viewer HLA version 1.3 or 1516 RPR FOM 2.0 draft 17	The 3D viewer. This viewer made the capturing of	http://www.mak.com/products/stealth.php

Table 30 FFI Provided Software

Software	Description	URL
Apollo for Eclipse	UML editor plug-in for eclipse.	http://www.gentleware.com/
Eclipse version 3.3.2	The programming environment.	http://www.eclipse.org/
Jadex version 0.96	BDI agent framework.	http://vsis-www.informatik.uni-hamburg.de/projects/jadex/links.php
Light Weight Java Game Library version 1.4	Mathematical methods for vectors.	http://www.lwjgl.org/
Protegè version 3.3.1	Development of ontology or KB.	http://protege.stanford.edu/
Microsoft Visio 2007	UML editor.	http://www.microsoft.com/
SUN JDK version 1.6.0-6	Java Development Kit.	http://java.sun.com/javase/

Table 31 Software Used in Demonstrator Development

The result of the demonstrator development process is available on the enclosed DVD. It contains the source code of the demonstrator framework and the videos displaying the test scenarios both in 2D and 3D view.

7.2 Defining User Requirements

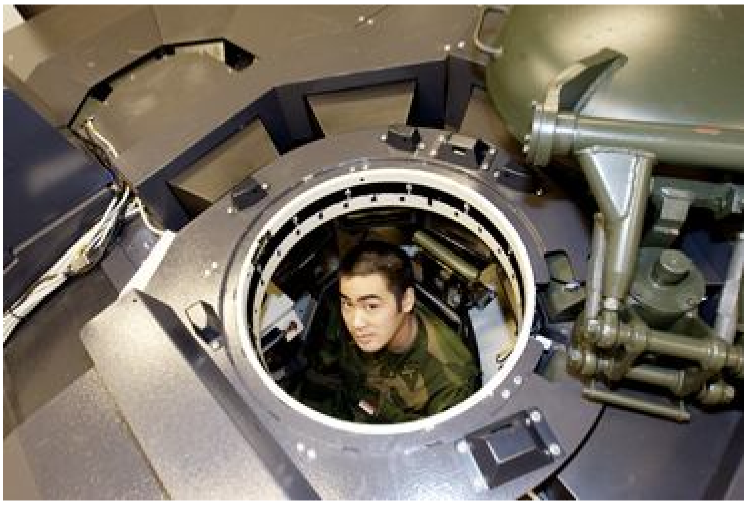


Figure 74 Simulation Cabin

Enabling tank platoon simulations requires information about how the platoon organizes its daily routines and how it operates in a war theatre. This information is available in military published field manuals that guide soldiers in different areas of operations. The US Army Field Manual for tank platoons (FM 17-15) is available on the internet and describes how the tank platoon fights: the procedures, the orders and tactical dispositions in different situations (126).

Even though the tactical foundations are equal in Norway and the US, there are some differences in general procedures for communication and planning, therefore a study trip to the TRADOC training facilities at Rena Camp was organized. Rena Camp houses the Combined Combat Simulator (CCS) for armored vehicles operated by the Norwegian Army (Figure 74). The simulator contains 8 simulation cabins, where 4 are CV9030N (Figure 75) and the last 4 are Leopard 1A5. Since the Norwegian Army is upgrading to Leopard 2A5 (Figure 76) so are the combat simulators.

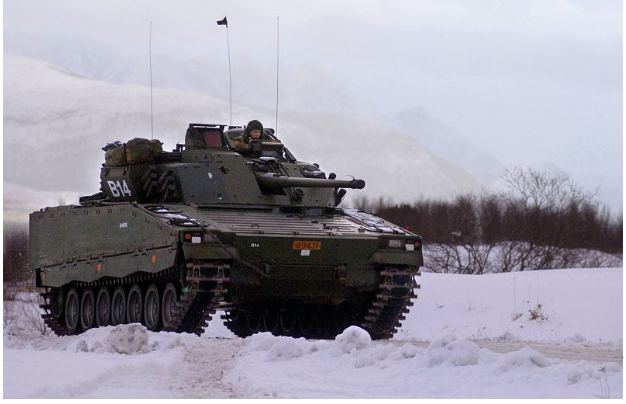


Figure 75 CV9030N

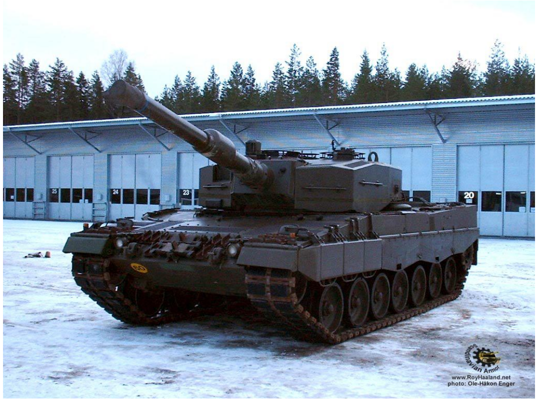


Figure 76 Leopard 2A5

The CV9030N is an Infantry Fighting Vehicle (IFV), which is a vehicle smaller and lighter than a tank and has enough space to contain an infantry squad. The vehicle provides cover from small firearms and rapid deployment of troops. The Leopard A25 is classified as a main battle tank and its main area of responsibility is to combat armored vehicles.

The training supervisor at Rena explained the use of the training battle simulator and how it can be configured to simulate up to half a mechanized battalion in attack. This configuration is achieved by letting every simulation cabin be the platoon leader and the rest of the platoon is simulated by artificial intelligent entities. The soldiers can train combat against each other or a computer generated artificial intelligent enemy where the maximum number of simultaneous enemies is 306.

The system is delivered by Kongsberg Defense & Aerospace and contains an AI component produced by Lockheed Martin. The original settings of the AI resulted in computer controlled units that operated too independently. The problem was grounded on the fact that the behavior of the AI controlled tanks was not concurrent with the commander's intent. The main reason being, that one was not able to inter-communicate the perceptions of the situation between man and machine. In the end most of the AI features ended up being disabled. The interface for giving orders and receiving feedback is basically a menu where it is possible to change formations, sectors of responsibility and pinpoint waypoints, amongst others.

The drawback with a graphical user interface is that it does not resemble the real life interaction. Communication between tanks mainly takes place cross radio channels. There are other possibilities like use of light and smoke, but the important information is exchanged on the different networks. Information can be sent digitally or directly through voice. It is normal to use three different voice channels, one for the platoon, one for the company and one for the battalion. The commands given over radio are short, formalized and with feedback. Using a GUI to simulate this rapid information exchange is futile, therefore a functionality that transforms the interaction from being click based to voice based is desired, both on orders from platoon leader and feedback from platoon members.

The Norwegian tank platoon operates on the bases of the Norwegian field manual for tank platoons called Stridsvogntroppen (127). The field manual explains the relations between the operations of the tank platoon and the joint operational doctrine, in addition to describing procedures for offensive and defensive tactical maneuvering. The effective tank platoon is the tank platoon that can maintain a unified understanding of the battle space. This means that the persons operating within the framework of the platoon know both the informal and formal procedures in the platoon and can utilize this common understanding into a tactical advantage. Recreating such an intuitive state of mind with a computational entity is far from trivial.

This short introduction to everyday life of a tank platoon gave insight to the most important elements to start with regarding the development of the demonstrator. The ultimate requirement is to recreate all procedures and tactical considerations described in the field manuals, but this task would probably take years to resolve, so to narrow the requirements down to something that is achievable within a short time frame two scenarios were developed. In both scenarios the idea is to use a GUI to give orders to the tanks in the platoon, which is and adaptation from the existing system at Rena but with even simpler requirements.

7.3 Demonstrator Architecture and Design

The agent architecture is developed on the background of the tank platoon, which normally operates within an armored company or battalion. The tank company (Stridsvogneskadron 2) consists of two tank platoons and a combat support platoon (Figure 77). Combat support makes sure that the company has access to easy repairs and resupply. The blue symbols follow the NATO standard for military symbols for land based systems (9).

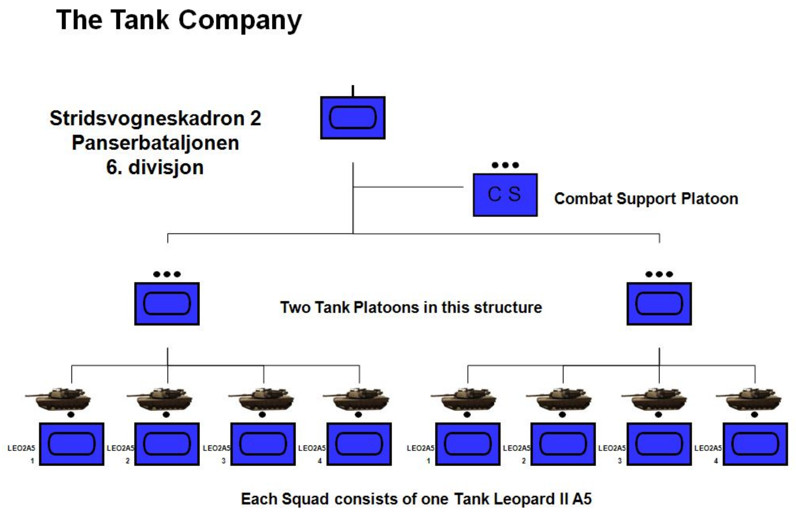


Figure 77 The Tank Company, from (126)

The tank platoon has 4 tanks and the command elements are organized in a hierarchical structure (Figure 78). Every tank has a crew of 4 persons were each person has his or her specific tasks or role to fulfill. The leader tank (1) is the command post of the commanding officer (CO), the tank next in command (2) is the command post of the executive officer (XO). Each tank also has a gunner, loader and a driver. The gunner is the next in command if the tank commander is killed and is responsible for targeting, the loader operate the machine gun and the main canon, while the driver controls the tank's movement in the terrain on the order of the tank commander (128). More information is available in the field manuals (126; 127).

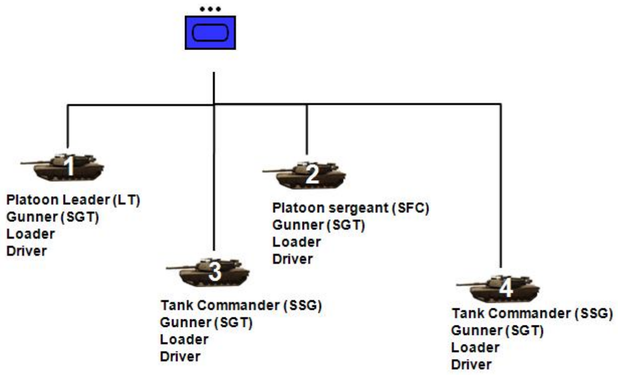
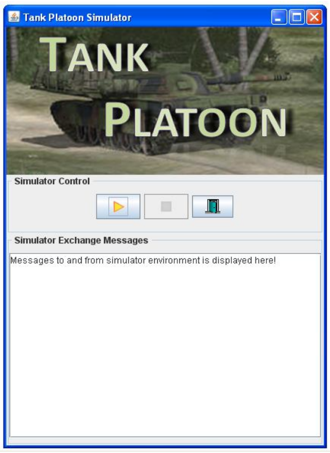


Figure 78 The Tank Platoon and its Crew, from (126)



Messages to and from simulator environment is displayed here!

Figure 79 Manager Frame

The platoon is implemented through agents, namely one agent per tank in the platoon (Figure 80). The leader agent has implemented methods that make it possible to communicate with the member agents and methods for communicating with the simulator manager. The leader agent has no deliberation on the information it receives, this is a task performed by the human operator. The member agents on the other hand are responsible for their own actions and should be able to deliberate on the orders or commands given by the platoon leader.

The abstraction level of the architecture is the tank and not the crew, since it is the visible behavior performed by the tank that is interesting.

The manager's role is to act as an interface between VR-Forces and the Jadex agent environment (Figure 79). In addition the manager can create the platoon agents, destroy the platoon agents or end the Jadex agent environment. The Java Radio is a Java library provided by FFI that provides an interface for interconnecting Java applications to VR-Forces through the MÄK RTI. The library provides functionality that makes it possible to send simulated radio messages to VR-Forces entities using HLA radio messages. A plug-in module for VR-Forces establishes the connection to the VR-Forces entity. The plug-in module was implemented by researcher Odd-Martin Staal supporting the development process of the demonstrator. The messages sent to VR-Forces are shown in the text area of the manager frame window. Messages are sent from VR-forces at different time intervals dependent on the movement of the simulated physical objects in VR-Forces.

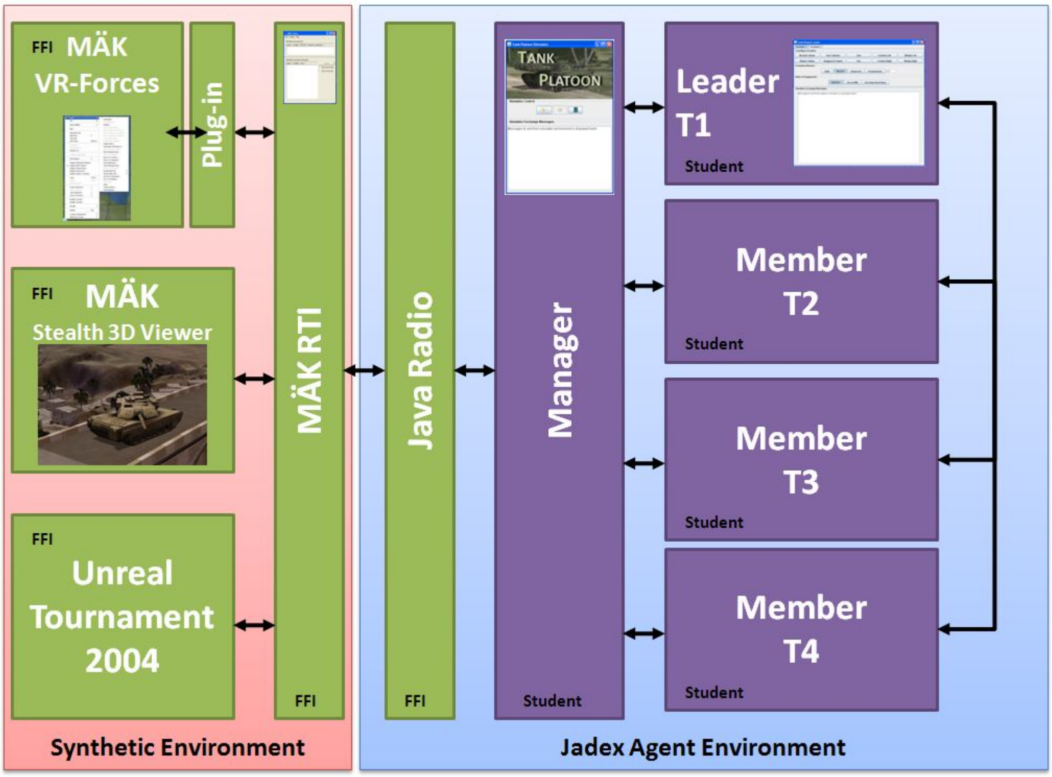
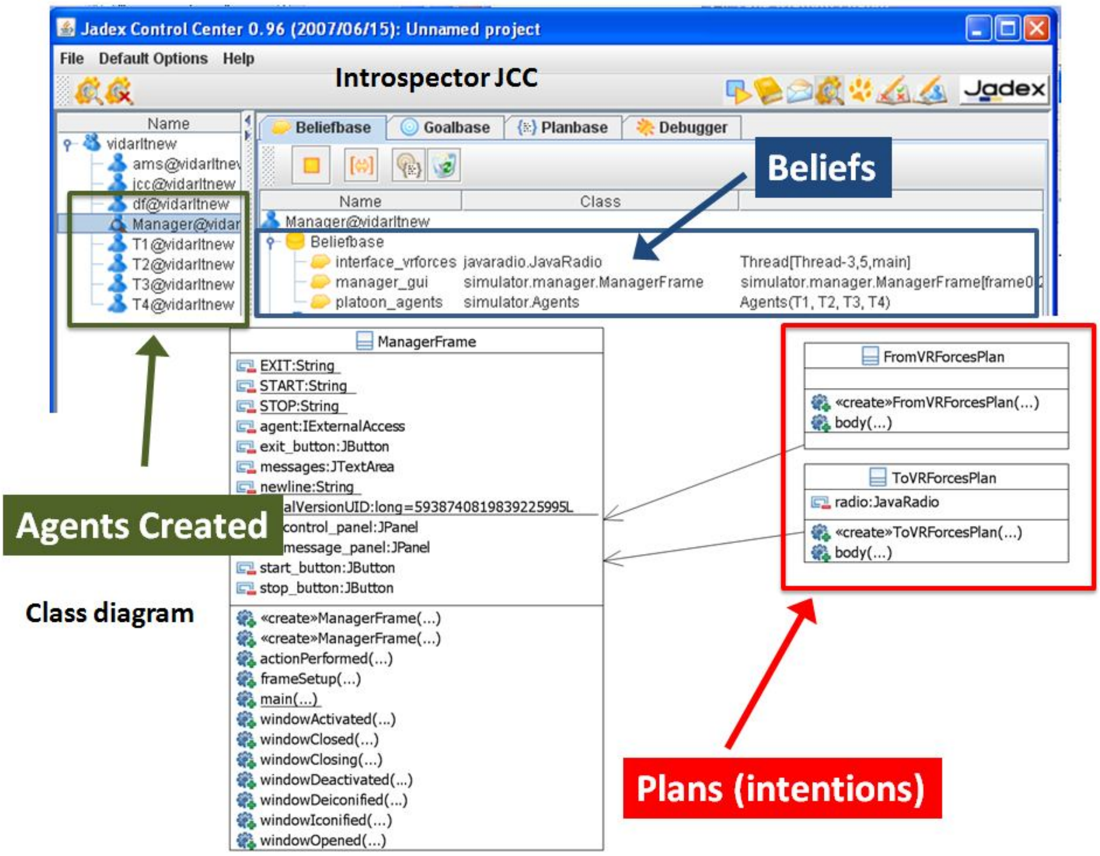


Figure 80 Demonstrator Architecture

The manager creates the platoon when the start button is pressed. The platoon creation is initiated by creating the agents T1 through T4. The numbering of each tank reflects the distribution of roles within the platoon (Figure 78). The organization of the communication leads to a divided communication domain. Each platoon agent maintains control over their simulated physical counterpart in the synthetic environment through the manager, which constitutes the external communication domain, while the internal communication domain consists of direct communication between the agents within the Jadex agent environment (Figure 80). It is possible to simulate radio transmission through VR-Forces, but this would have made the interfacing to VR-forces even more complicated. Instead exchange of messages between agents happens within the Jadex environment.

The manager is an agent that handles both the Java Radio interface and the GUI (Manager Frame) as beliefs (Figure 81). The platoon agents' identifiers are stored in the belief set of the manager so that it knows which platoon agent to forward messages to. It is important that the entities in VR-Forces follow the same naming conventions as the agents themselves. If not the forwarding mechanism would fail. When the manager agent receives messages either to or from VR-Forces the actions are performed by instantiating plans. The manager only contains two plans:

- Forwarding to VR-Forces (ToVRForcesPlan.class), which creates a string from the Java message object.
- Forwarding from VR-forces (FromVRForcesPlan.class), which also performs a message parsing function that translates a string into Java objects.



Name	Class	
Manager@vidarltnew		
Beliefbase		
interface_vrforces	javaradio.JavaRadio	Thread[Thread-3,5,main]
manager_gui	simulator.manager.ManagerFrame	simulator.manager.ManagerFrame[frame0
platoon_agents	simulator.Agents	Agents(T1, T2, T3, T4)

Figure 81 The Manager Agent

It is important to keep in mind that the Jadex agent is described through a XML file and that plans and beliefs are the only code written directly in Java. The manager agent uses a Jadex capability that enables it to create and destroy agents within the Jadex environment (jadex.planlib.AMS). The manager creates the platoon agents one by one with their initial beliefs and sends a setup message that contains the agent identifiers of all the platoon agents. These messages also triggers localized agent actions for belief setup (Figure 82). The destroy procedure is similar to the create procedure (without the setup message) and kills the agents in the Jadex Environment.

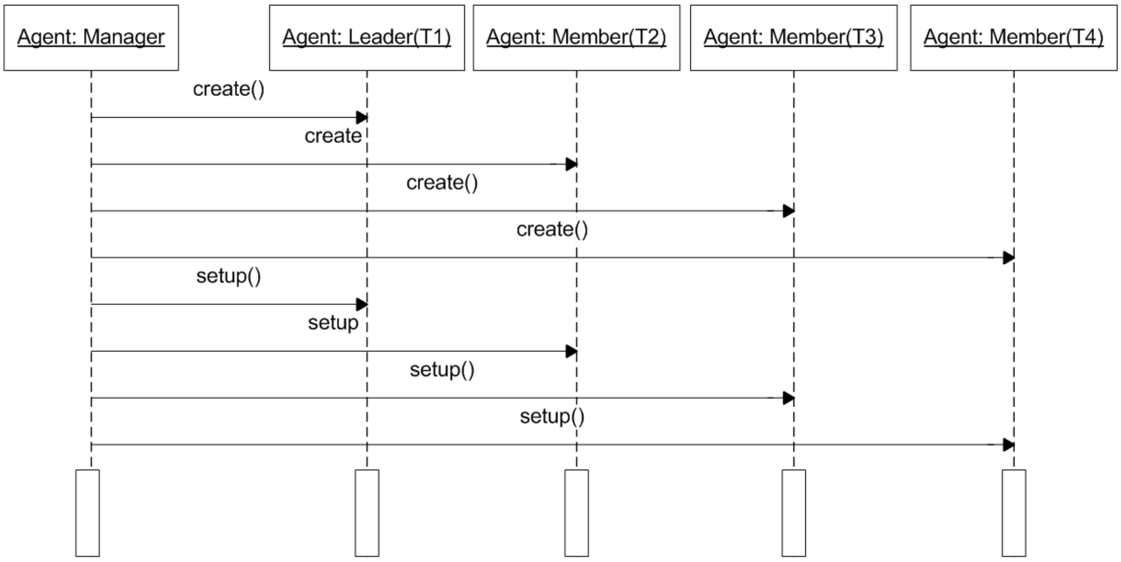


Figure 82 Sequence Diagram for Platoon Setup

The exchange of messages to and from VR-forces is immediately available after the setup procedure has been performed. A limited number of external messages have been implemented and are listed in Appendix K: Java Radio Message Configuration.

The external communication domain constitutes a simple forwarding mechanism for exchanging data information between VR-Forces and the agents. The knowledge connected to the information exchange form the external ontology of knowledge. Jade cannot automatically send Java objects between agents without a formal ontology. The ontology is created through the use of Protegè and Beanynizer. Using these tools it is possible to form a structure of Java objects that constitute the knowledge exchanged during Jadex message events. The Beanynizer plug-in generates the Java classes needed. One should be aware of the fact that Beanynizer files created with the ontology options *editable* and *fixed* are the only objects that one will be able to send through Jadex as message events without performing extensive modifications to Java objects that one desire to transmit (113).

Knowledge domains identified, other than the external communication domain, are the knowledge connected to the internal communication domain and the agent belief sets (Figure 83). The knowledge exchanged between the platoon agents needs its own ontology. The structure of knowledge within the agent itself also needs to be organized into a practical format. Finding a suitable organization of knowledge is imperative for creating efficient deliberation within AI entities and in Jadex especially for constructing message events.

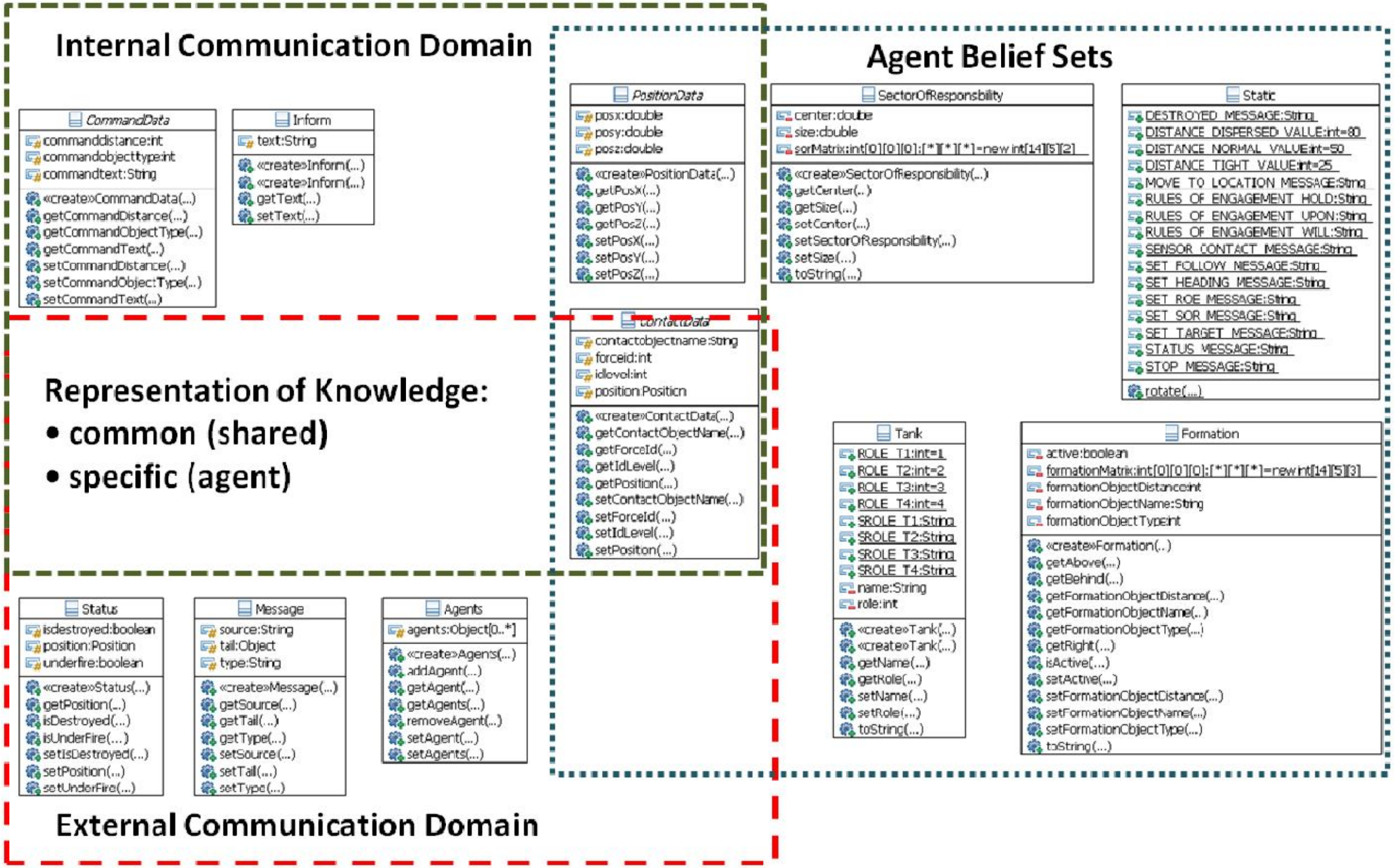


Figure 83 Domains of Knowledge

7.4 Demonstrator Scenario 1

Scenario 1 is a simple scenario where the tank platoon is going to move from a starting point to an end point (Figure 84). The platoon will not encounter any enemy contact during the movement. The leader tank should be able to change travelling formations, change the distance between the tanks and set the rules of engagement (ROE). When changing the traveling formation the sector of responsibility (SOR) also changes. The SOR constitutes the area that the tank is responsible for monitoring and scaning for enemy contacts.

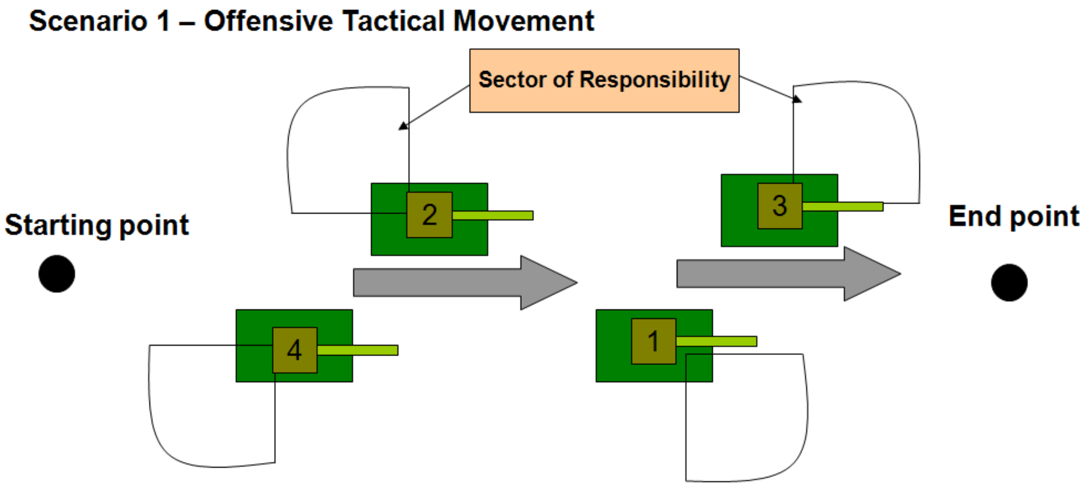


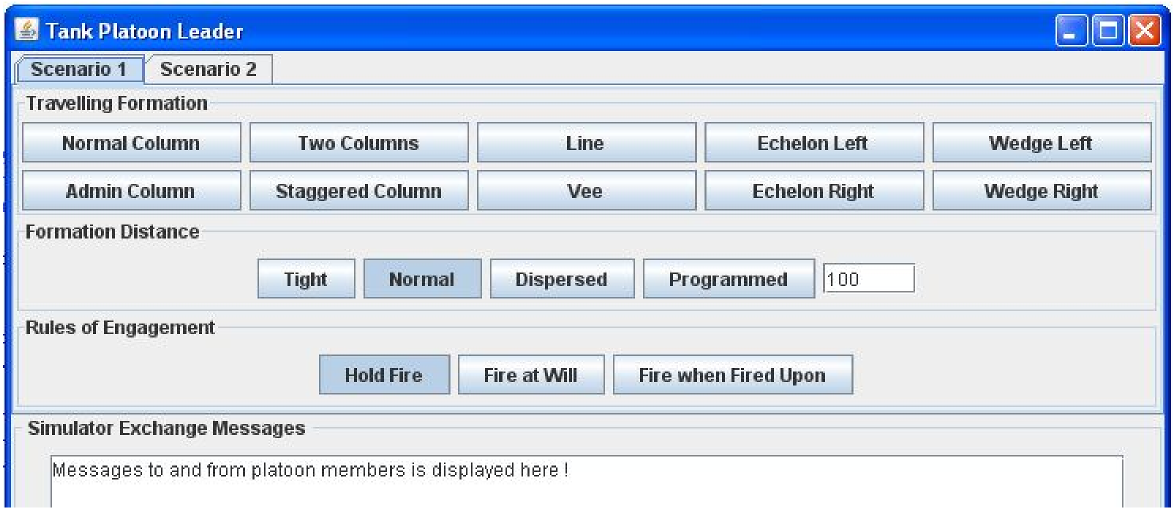
Figure 84 Scenario 1

The ROE is in reality a complex set of rules that regulates the amount of force a military unit can use in different situations. In this case the rules have been set to coincide with the simple settings in VR-Forces:

- Hold Fire
- Fire at Will
- Fire when Fired Upon

The traveling formations and the SOR is set according to fixed matrixes. A travelling formation is identified by the command type and the tank role (Formation.class). The matrix then returns a set of offset values that describes the relative position to the leader tank (T1). When the command is received by the member tank it calcualtes the offset values and send a message to VR-Forces that sets up a follow task with the given offset values. VR-Forces then executes the task and the unit starts to move into position. Directly after the transmission of the formation change, the agent sends the SOR message to VR-Forces, which is calculated based on formation type and tank role (SectorOfResponsibility.class).

Commands or orders sent from the leader tank to the member tanks are controlled through sending command messages (internal messaging). A command message constitutes a message event in Jadex, where the leader has an outbound message event and the members has an inbound message event. The order is instantiated by the human contestant through a menu (Figure 85). The menu is compartmentalized according to formation, distance and ROE.



Messages to and from platoon members is displayed here !

Figure 85 Leader Tank Command Menu Scenario 1

Every button in the menu initiates a command with a distinct number that identifies the command type. A command object is created and transmitted to the member agents (Figure 86). The member agents then returns an inform message to acknowledge the received command message. Before there can be any change in the movement of the simulated physical entity in VR-Forces the order must be transformed into a task operation that VR-Forces understands. A message java object is created with the necessary content (Appendix K: Java Radio Message Configuration) and is transmitted to the manager agent. The manager reformats the message into a string and sends it through the Java radio. If the VR-Forces application has been setup correctly and the scenario is running the simulated entity will perform the change in movement.

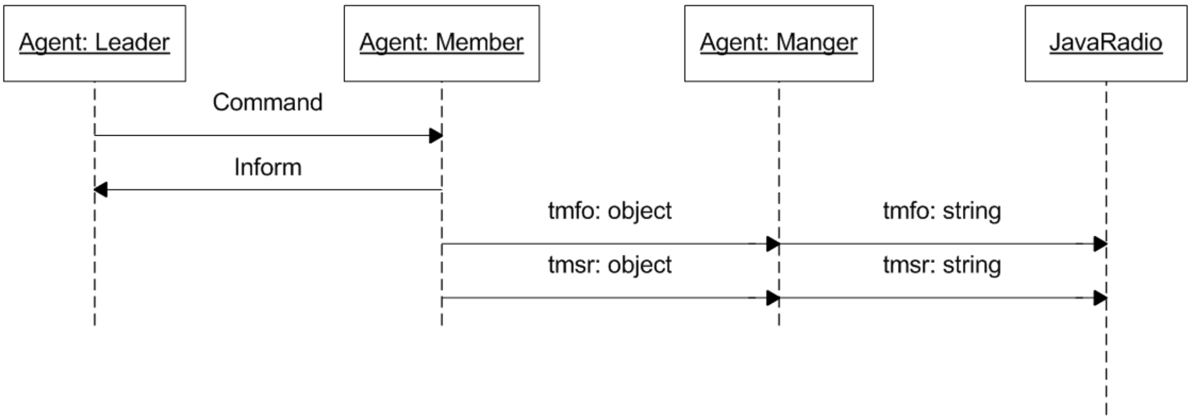


Figure 86 Sequence Diagram for Formation Commands

The process is equivalent in the case of a change in the formation distance or the ROE. A change in the distance results in a retransmission of the active formation with the new distance parameters, while a change of the ROE results in transmission of a string that identifies the new active ROE. The ROE is programmed to only be sent from the member agent if it is in reality different from the previously active ROE.

The different formation types are shown in Appendix L: Formation Types. Figure 87 shows how the formation matrix is configured. The matrix itself contains integers that will be multiplied with the formation distance (Formation.class). For more information on the use of formations in tactical operations take a look in the field manuals for the tank platoon (126; 127).

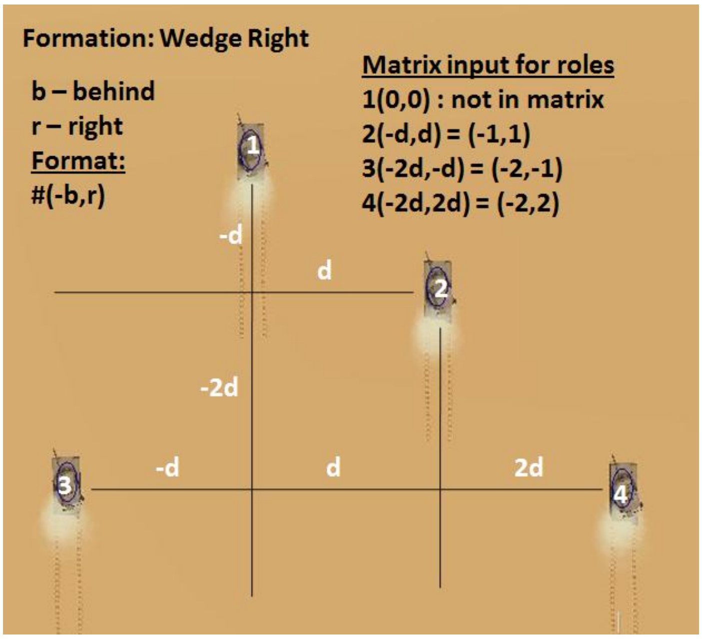


Figure 87 Formation Matrix Setup and Calculation

The scenario has been tested and recorded by using the Camtasia Studio software. There are 4 recordings that cover this test scenario (Figure 88):

- Scenario 1 – Route 1: VR-Forces 2D view
- Scenario 1 – Route 2: VR-Forces 2D view
- Scenario 1 – Route 1: Stealth 3D view
- Scenario 1 – Route 2: Stealth 3D view

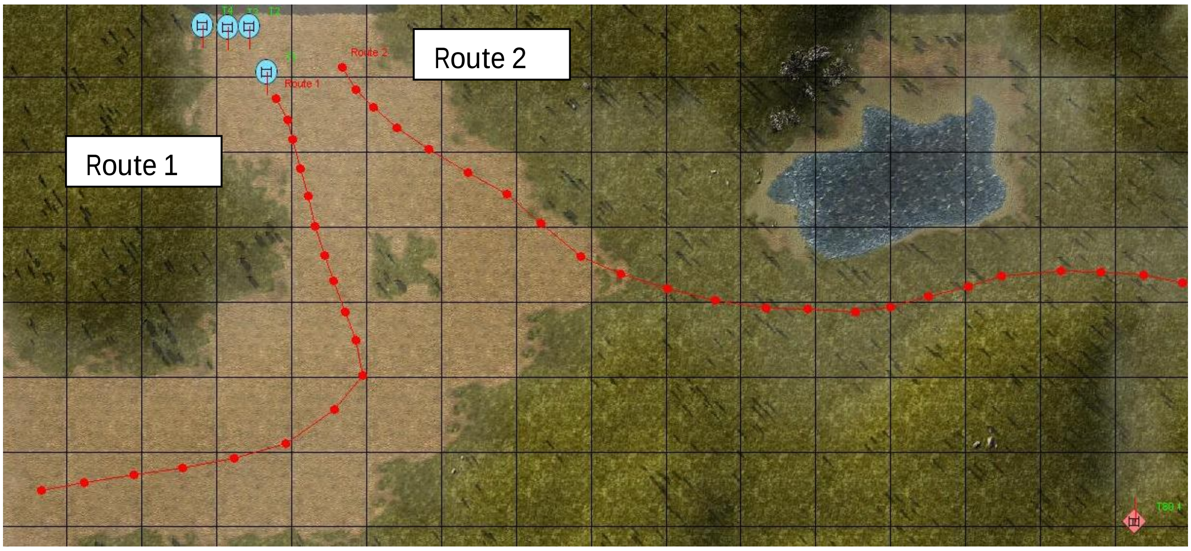


Figure 88 Scenario 1 Test Environment

7.5 Demonstrator Scenario 2

Scenario 2 is an extension to scenario 1, where the tank platoon now encounters an enemy while moving from the starting point to the end point (Figure 89). The orders from the platoon leader are extended to contact drills, retreat and stationary formations.

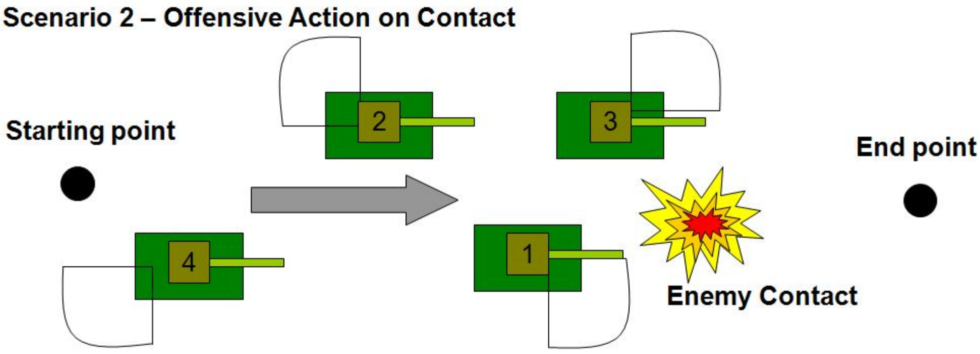
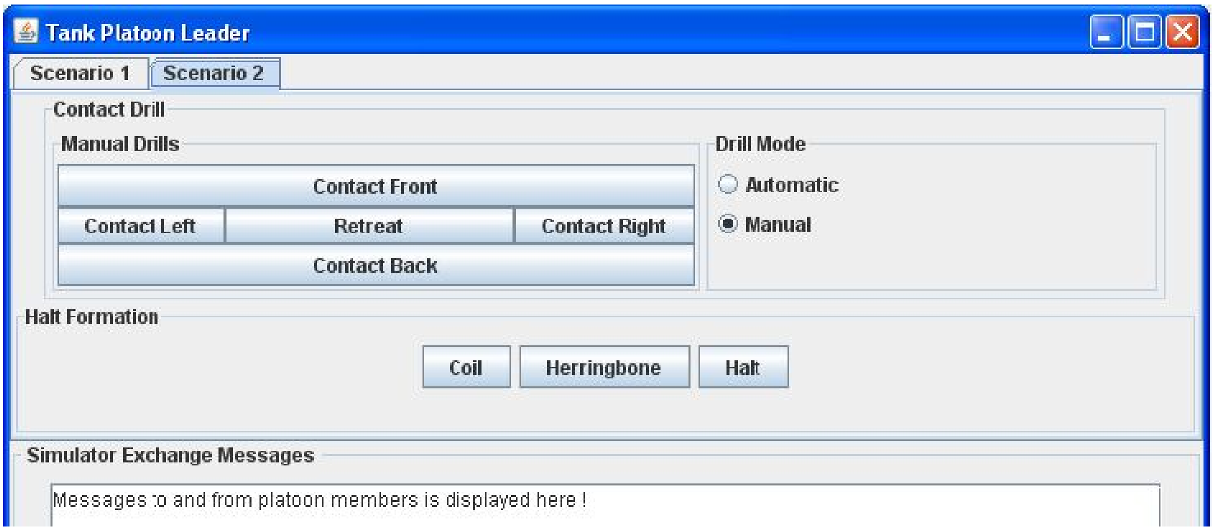


Figure 89 Scenario 2

Contact drills implemented are based on the demonstration at Rena Camp. The drills are simplified into five command buttons that state the direction of the contact and implicitly the maneuver to be performed by the platoon (Figure 90). The maneuver is basically to form a line in the direction stated by the order. The contact direction is seen in relation to the platoon's direction of movement. The retreat button is a simple order to move 100m backwards.

The drill mode states if the contact drill can be performed on the initiative of the platoon members or not. Manual means that it is up to the platoon leader to initiate contact drills, while automatic delegates the responsibility of initiating contact drills to all platoon members. Static or halt formations are basically formations that the platoon might perform when securing an area. The halt command basically tells the platoon to stop moving and release the platoon members from previously given orders. More information on static formations and combat drills is available in the field manual (126).

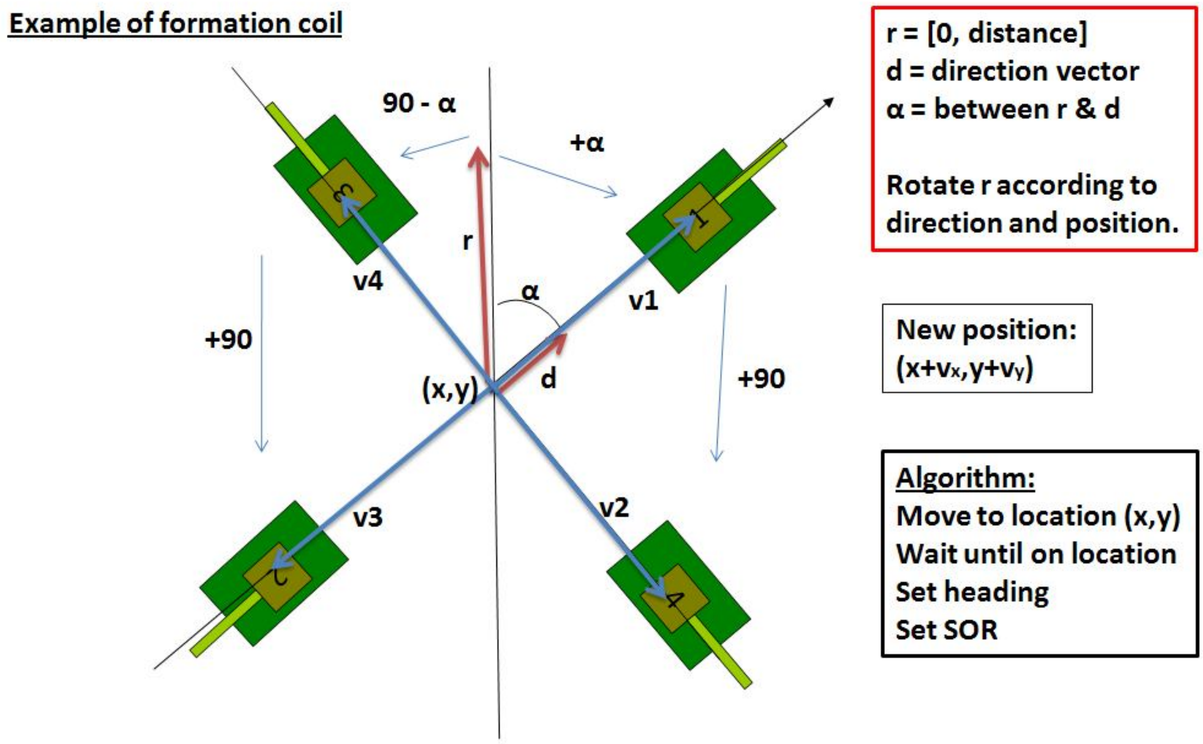


Messages to and from platoon members is displayed here !

Figure 90 Leader Tank Command Menu Scenario 2

The platoon agents does not have a complete internal picture of the external environment, which means that one of the tasks of solving the scenario 2 problem was to develop methods for calculating positional reference points that could be used to deduce the future location to move to. The solution chosen was to create a goal in the BDI structure that calculates a direction vector, which describes the direction of movement in the synthetic environment using database coordinates. This vector is the foundation for creating all the scenario 2 formations both the static and contact drill formations.

The vector is calculated on the background of the received status messages from VR-Forces. Every 10th meter of movement a status message is sent that updates the agent's positional data. For every update a direction vector is calculated using the previous and new position. Formations are created by creating a reference vector with a specific length and rotate it the wanted amount of degrees (Figure 91). The new vector determines the offset values from the last registered position of the leader tank. The leader tank sends its position when the order is given.



Algorithm:
Move to location (x,y)
Wait until on location
Set heading
Set SOR

Figure 91 Location Calculation for Coil

The use of vectors leads to a need for vector calculation methods. Instead of developing vector methods specifically for this demonstrator some methods were adopted from the Light Weight Java Gaming Library for 2D vector calculations. Nevertheless the vector rotate method was developed for specifically solving this task in the demonstrator (Static.class).

The active travelling formation when the contact drill order is given determines the configuration of the line that is going to be formed in the specified direction. Based on the position of the leader tank an offset is created for each tank role and the same algorithm as stated in Figure 91 is used to move the tank to the proper location and set the heading and SOR. The main difference being that the offset calculation is the sum of two vectors (Figure 92).

The movement of the leader tank is not automated therefore it is important that the person running the game control for the leader tank stays on top of the situation and not break formation. The time delay from an order is dispatched until its performed is short and only limited to the processing capacity of the processor.

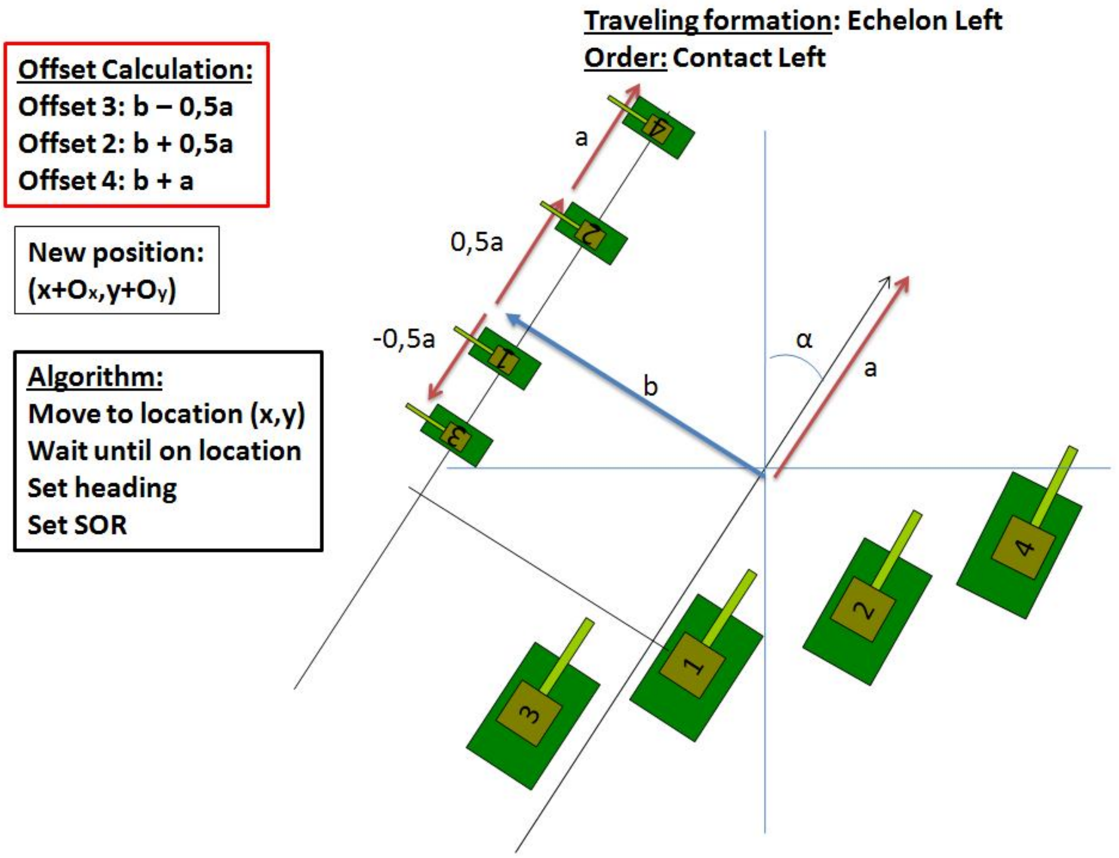


Figure 92 Location Calculation for Contact Drill

The scenario has been tested and recorded by using the Camtasia Studio software. There are 8 recordings that cover this scenario (Figure 93):

- Scenario 2 – Route 1 and 2: VR-Forces 2D view
- Scenario 2 – Route 3: VR-Forces 2D view
- Scenario 2 – Route 4: VR-Forces 2D view
- Scenario 2 – Route 5: VR-Forces 2D view
- Scenario 2 – Route 1 and 2: Stealth 3D view
- Scenario 2 – Route 3: Stealth 3D view
- Scenario 2 – Route 4: Stealth 3D view
- Scenario 2 – Route 5: Stealth 3D view

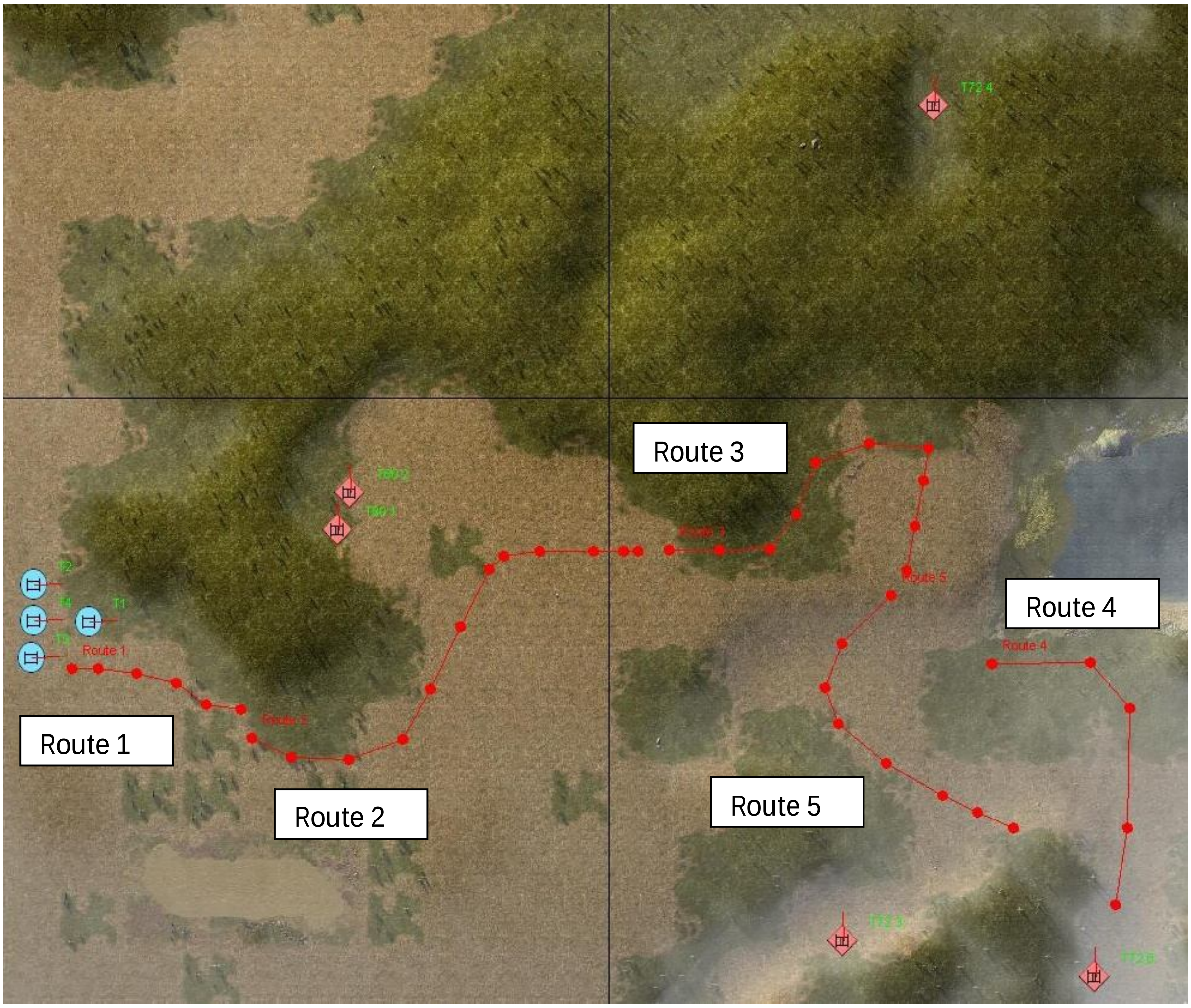
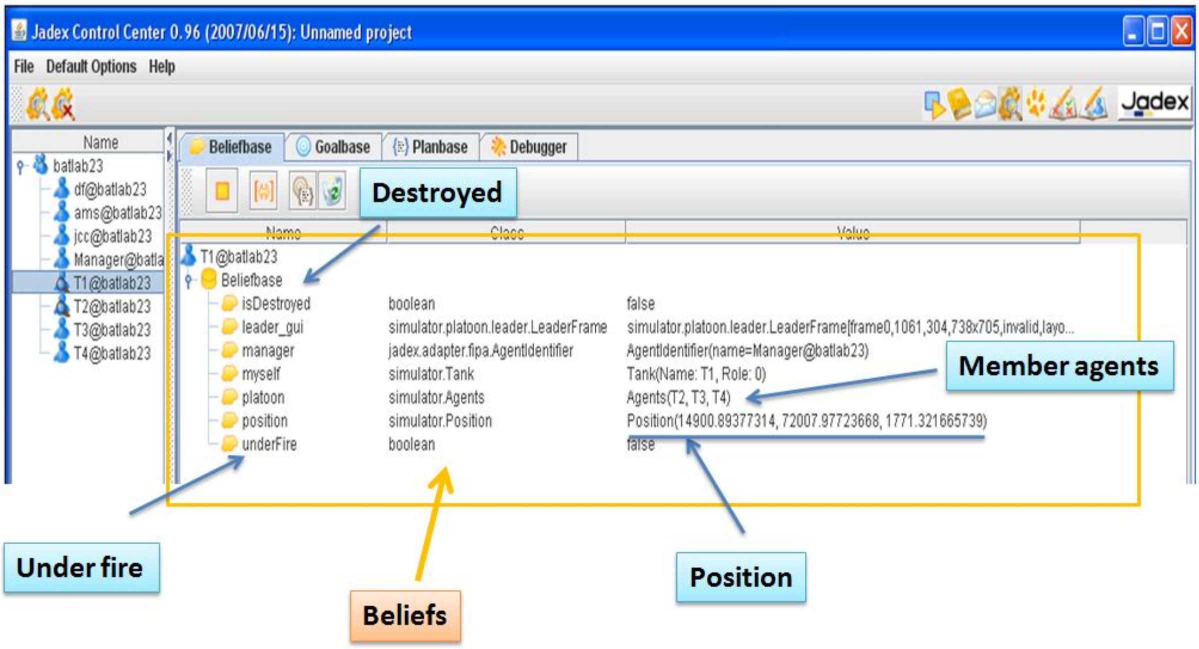


Figure 93 Scenario 2 Test Environment

7.6 Platoon Leader Beliefs, Goals and Plans

The amount of registered beliefs in the leader tank (T1) is fairly limited due to the fact that it is the HITL that is supposed to produce his or her own impressions of the situation. What does the leader agent need to know? It needs to know the agent identifiers of the platoon members in order to be able to send orders. In addition it needs to know its own position in order to issue orders that demand a reference point (scenario 2). Last but not least it needs to know if it is under fire so that the HITL is notified about possible threats and if it is destroyed the HITL should know that the game is over (Figure 94).



Name	Class	Value
T1@batlab23		
Beliefbase		
isDestroyed	boolean	false
leader_gui	simulator.platoon.leader.LeaderFrame	simulator.platoon.leader.LeaderFrame[frame0,1061,304,738x705,invalid,layo...
manager	jadex.adapter.fipa.AgentIdentifier	AgentIdentifier(name=Manager@batlab23)
myself	simulator.Tank	Tank(Name: T1, Role: 0)
platoon	simulator.Agents	Agents(T2, T3, T4)
position	simulator.Position	Position(14900.89377314, 72007.97723668, 1771.321665739)
underFire	boolean	false

Figure 94 The Leader Agent

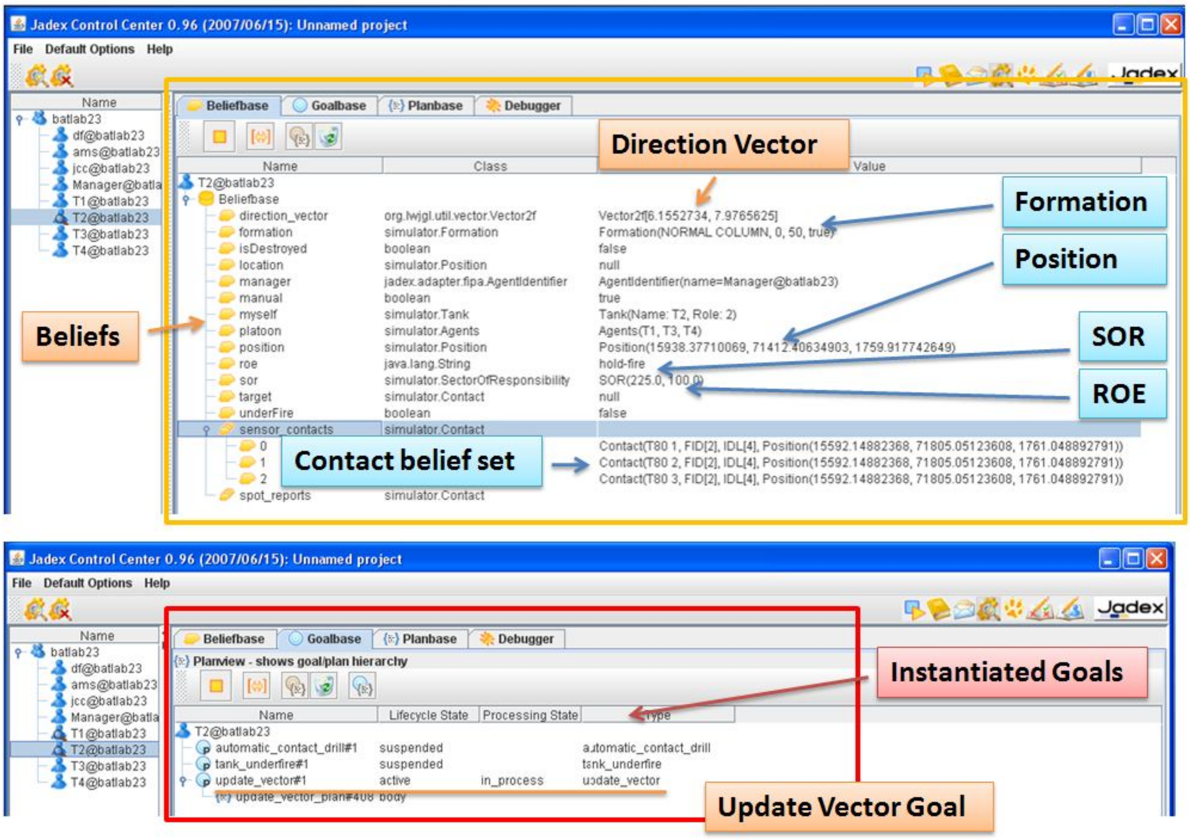
The sending of commands is taken care of by the Leader Frame GUI. The GUI has been presented in the previous chapters about the demonstrator scenarios. The leader agent only receives three different message events and each of them triggers the instantiation of a plan (Table 32). The leader agent has no goals due to the fact that goals are situated in the HITL contestant.

Plan	↔	Trigger Name	Trigger type	Description
SetupPlan	In	setup_ data	msg event	Receives the initial data for setting up the agent.
FromVRForcesPlan	In	from_ vrforces	msg event	Receives data from VR-Forces.
InformPlan	In	inform	msg event	Receives an inform string from member tanks.

Table 32 Platoon Leader Plans

7.7 Platoon Member Beliefs, Goals and Plans

In contrary to the leader the member tanks (T2, T3 and T4) have to keep a record of all their presumptions about the world in order to be able to deliberate on them. Each agent therefore contains a more extensive list of beliefs and goals (Figure 95). It contains the same beliefs as the leader agent and all the information it needs to calculate position in formations and move to locations. A list of sensor contacts that resembles the list of visual perceived contacts from VR-Forces and a list called spot-reports which are a list of reported sightings from other member agents. Based on these two lists the agent can assign a target to itself. In addition you have the direction vector, the ordered SOR and ROE.



T2@batlab23 – Beliefbase

Name	Class	Value
direction_vector	org.lwjgl.util.vector.Vector2f	Vector2f[6.1552734, 7.9765625]
formation	simulator.Formation	Formation(NORMAL COLUMN, 0, 50, true)
isDestroyed	boolean	false
location	simulator.Position	null
manager	jadex.adapter.fipa.AgentIdentifier	AgentIdentifier(name=Manager@batlab23)
manual	boolean	true
myself	simulator.Tank	Tank(Name: T2, Role: 2)
platoon	simulator.Agents	Agents(T1, T3, T4)
position	simulator.Position	Position(15938.37710069, 71412.40634903, 1759.917742649)
roe	java.lang.String	hold-fire
sor	simulator.SectorOfResponsibility	SOR(225.0, 100.0)
target	simulator.Contact	null
underFire	boolean	false
sensor_contacts	simulator.Contact	
0		Contact(T80 1, FID[2], IDL[4], Position(15592.14882368, 71805.05123608, 1761.048892791))
1		Contact(T80 2, FID[2], IDL[4], Position(15592.14882368, 71805.05123608, 1761.048892791))
2		Contact(T80 3, FID[2], IDL[4], Position(15592.14882368, 71805.05123608, 1761.048892791))
spot_reports	simulator.Contact	

Planview - shows goal/plan hierarchy

Name	Lifecycle State	Processing State	Type
T2@batlab23			
automatic_contact_drill#1	suspended		automatic_contact_drill
tank_underfire#1	suspended		tank_underfire
update_vector#1	active	in_process	update_vector
update_vector_plan#408	body		

Figure 95 The Member Agent

The leader agent did not have any goals. The member agent on the other hand has many goals. The reason for this is that goals function as repetitive cycles that try to maintain or achieve a specified state of affairs. For instance is the "update_vector" goal the model's intuitive sense of direction. The vector is created based on personalized information and therefore it is not known by the other agents, meaning that in some cases the algorithm for performing a contact drill to the left can end in chaos if not the perceived direction of movement is not common. The goals trigger a plan when certain conditions are fulfilled. Figure 95 shows for instance the suspended goal "tank_underfire" and it will stay suspended until the value of the belief "underFire" becomes true. The member agent instantiate plans through goals, belief changes and message events (Table 33).

Plan	↔	Trigger Name	Trigger type	Description
SetupPlan	In	setup_ data	msg event	Receives the initial data for setting up the agent.
CommandPlan	In	command	msg event	Receives a command from the leader agent.
LeaderPositionPlan	In	inform	msg event	Receives a position from the leader agent.
FromVRForcesPlan	In	from_ vrforces		Receives data from VR-Forces.
StaticFormationPlan	-	static_ formation	goal	The agent wants to achieve a new formation.
TravellingFormationPlan	-	travelling_ formation	goal	The agent wants to achieve a new formation.
SorPlan	-	sor	belief ch.	Alter the SOR if it has been changed.
RoePlan	-	roe	belief ch.	Alter the ROE if it has been changed.
MoveToLocationPlan	-	move	goal	The agent wants to move to a specific position.
UpdateVectorPlan	-	update_ vector	goal	The agent wants to update the direction vector.
ContactDrillPlan	-	contact_ drill	goal	The agent wants to perform a contact drill.
EnemyCheckPlan	-	enemy_ check	goal	The agent checks if some of the sensor contacts are hostile.
DestroyedPlan	-	tank_ isdestroyed	goal	The agent wants to perform its last duty in its life cycle.
UnderFirePlan	-	tank_ underfire	goal	The agent wants to perform some evasive actions.
AutoContactDrillPlan	-	automatic_ contact_ drill	goal	The agent wants to perform contact drill on its own initiative.
SpotReportPlan	-	spot_ report	msg event	The agent has received spot report from some of the other agents.

Table 33 Platoon Member Plans

Chapter 8 Discussion and Lessons Learned

This master thesis has just touched the surface of HBR. It has presented a small portion of the theoretical platform that constitutes the background of HBR and given an overview of agent and simulation frameworks. This chapter will comment on the agent architectures presented in Chapter 4 and their software implementation presented in Chapter 5. The case studies presented in Chapter 6 and the implementation of the demonstrator in Chapter 7, are concrete examples of practical use of HBR elements. Experience collected during the case analysis and the demonstrator development will be presented and discussed. Finally the chapter takes closer look at the most important lessons learned during the period of writing the master thesis.

8.1 Agent Frameworks

Defining HBR properties was the prerequisite for being able to categorize the different agent frameworks chosen for an in-depth evaluation (chapter 4.3). At the moment there are no specific taxonomies that define areas of HBR at least any that I could identify. The properties are loosely described in the literature and some of the expressions used have different meaning depending on which institution the paper originated from. Some of the reasons for this are that the research field of AI and then ultimately also HBR is rapidly evolving and new areas of research are discovered as we speak. The coordination between the research groups is in many ways insufficient, meaning that work on the area seems uncoordinated and hard to get an understanding of. The most referenced book on the area is Pew and Mavor (3), but this book is rapidly going out of date. The HBR properties in this master thesis are founded in research on AI and HBR and can be further developed into a taxonomy that defines HBR in a more concrete way. Recent projects have been focusing on comparing agent frameworks (39) and categorizing PMFs (29) this work will make it easier to understand and conduct research on HBR. Lack of standardization and coordinated research influence especially the capability to interconnect frameworks.

The evaluation of agent frameworks was founded on preliminary selection criteria for identifying agent frameworks eligible for scrutinization (chapter 4.4). Several agent frameworks where omitted due to that software and information was hard to come by or that they only were commercially available. An example of an architecture that has been used in many projects is the COGNET/iGEN framework. This framework has been used in the Agent-Based Modeling and Behavior Representation (AMBR) model comparison project and is considered to be implemented with the goal of easing behavior model development (39). The fact that some of the frameworks have been unobtainable constitutes a limitation of the agent framework survey performed. The choice of not including commercial frameworks was nevertheless a deliberate act to reduce the number of frameworks in the survey. Further investigations on these frameworks are encouraged.

There are initiatives that investigate alternatives to the rule-based approaches to HBR (chapter 4.1 and 4.2). In the future these frameworks will be more mature and eligible for use in greater scale than today. These are frameworks based on genetic algorithms and artificial neural networks (27).

The agent frameworks presented in Chapter 4 have different properties of HBR. The cognitive frameworks concentrate on covering aspects of memory, perception and reasoning. One of the reasons for this is that it is believed that this kind of structure can simulate all aspects of human cognition as long as the behavior model supports it. Frameworks that incorporate social and group behavior are few, so are the frameworks that incorporate biology, personality and emotions (chapter 4.5). The downside of having to create a behavioral model for every domain of the human mind is that it demands extended effort of development. Frameworks that already have incorporated such features ease the modeling process and ultimately the complexity of the behavior model.

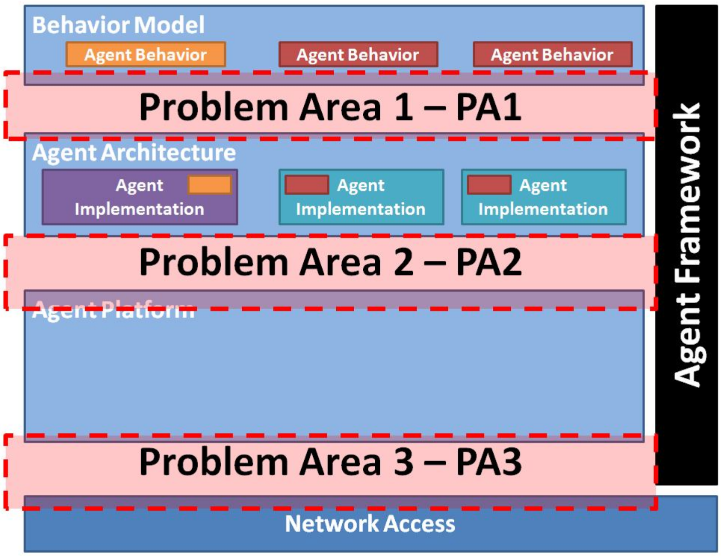


Figure 96 Problem Areas of Agent Frameworks

Figure 96 shows three main problem areas that need to be addressed in order to implement AI into military simulations. PA1 describes the area of behavior modeling within the agent architecture. Making specific behavior models that covers a certain area of operational psychology is one challenge, but the real challenge becomes evident when several behavior models need to interact in order to create new behavior. For instance if you implement a model of emotions, how should this model interact with the model of SA? The problem of interacting modules was presented in chapter 4.5.2 about PMFserv. To a certain degree most modelers need to think about how models interact.

PA2 identifies a major problem area of the cognitive architectures described in chapter 4.6, namely the problem of interacting with other agents and with the SE. The SE can be looked upon as an agent that the agent architecture needs to interact with. The agent platform has the responsibility to deliver networking capabilities to the agent. The agent platform can contain a HLA compliant framework or in the case of Jadex a FIPA compliant agent platform called JADE.

PA3 identifies the problem area of sending knowledge over a network. There are agent specific communication languages that take care of providing this service (chapter 3.4.2), for instance the FIPA ACL standard. In many cases there will be several communication domains to consider as in the case of the demonstrator. One domain of communication is the one that handles communication traffic between the agents, and the other domain that handles the communication traffic to and from the simulation environment. The number of communication domains depends on the structure of the simulated entities.

Creating behavior models within frameworks like ACT-R, EPIC and Soar are considered to be a challenging task. The main reason for this is that the support tools for creating behavior models are limited and that the architectures in themselves are considered to be complex environments to build models within. In many cases the frameworks lack an agent platform, which means that communication in principle is not supported (facilitating PA2 and PA3). The three academically founded frameworks have been used in many different projects, but in most cases the code developed in these projects are not available. This is especially true for military founded research projects. Soar has the most active user community compared to the other agent frameworks reviewed. In addition behavior models written in Soar are commercially available through Soar Technology (111). EASE (Elements of ACT-R, Soar and EPIC) is a framework that aims at combining the best elements of these three architectures, but the code is not published (39 p. 242).

In contrast to ACT-R, EPIC and Soar the agent frameworks Brahms, Jadex, and JACK have built in support for network communication, including explicit support for creating interfaces to external applications or simulations. The drawback of using BDI architectures is on the other hand that one looses the cognitive aspect in the simulation unless one builds a cognitive layer into the architecture (CoJACK).

All the agent frameworks reviewed in the master thesis uses a framework specific programming language or modeling language. The lack of standardization results in a number of different modeling languages which are incompatible. This is a huge disadvantage when it comes to porting a behavior model from one framework to another framework.

8.2 Development of Demonstrator

The use cases in Chapter 6 gave insight into problems concerning the resolution level of an implementation. The C1 and C2 uses cases are closely connected, but the resolution level of C2 is higher than C1. A higher resolution level leads to a more complex agent structure. Due to the perceived complexity level of implementing C1 and C2, the use case C3 was selected for implementation in a demonstrator (Chapter 7). Another factor was the fact that the availability of COTS on crowd simulations is limited. Implementing a population or a crowd would lead to a more complex agent structure due to the need of accessing larger portions of environmental data. The C3 use case is more scalable for testing purposes and the reasoning about terrain features could be handed over to the HITL contestant.

The most profound reason for choosing Jadex as the development framework was the support for communication and that the programming language needed to be a familiar one, in this case this was java. Learning and adapting to a new programming language would have consumed most of the development time. An open source framework has the advantage that the source code is available and can be changed, in contrary to JACK which is the commercial BDI equivalent. JACK has extensive functionality that goes far beyond the need for the demonstrator development. The simplicity of Jadex became its strongest advantage. The disadvantage is the lack of cognitive foundation.

The cognitive architectures are fairly complex environments to create models within, and in many of the projects that have incorporated them the time for creating the interface to the external environment consumed considerably amount of the total development time. In addition to that most cognitive architectures demand knowledge of Lisp or C++ for encoding the behavior model. I also got a recommendation from the NTNU professor Pinar Öztürk that using these frameworks would induce more problems than solutions within the short time frame. The conclusion was that for this limited demonstrator Jadex was the preferred choice.

The Jadex agent development framework contains an agent platform and BDI architecture. The problem areas that arose during the development process were:

- Which knowledge should the platoon members deliberate on?
- How should the deliberation process be modeled?
- How should the agents communicate with their counterpart in the virtual environment?
- How should the agents exchange knowledge between themselves?
- What should the ontology of the communication contents be?
- How should the agents communicate with the HITL leader?

These problems are different modes of the representation problem and the reasoning problem presented in chapter 3.3.4. The object is to create a platoon grounding process rooted in sensemaking (chapter 3.1). With other words the intention is to create a mutual understanding of the situation within the platoon, and create an interaction process where knowledge cannot be misinterpreted. The two scenarios that were developed laid the groundwork for identifying the knowledge content of the platoon member agents and the communication contents (Chapter 7).

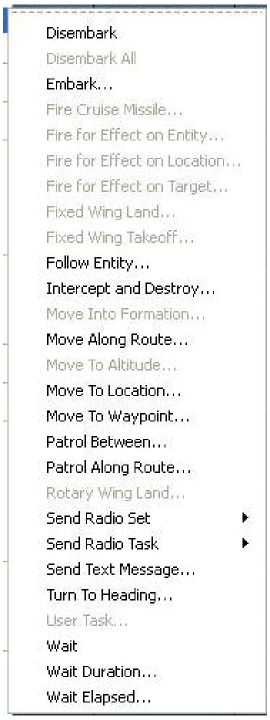


Figure 97 VR-Forces Tasks

During the testing of the scenarios several weaknesses of the implementation became evident. VR-Forces have a built-in low-level AI that takes care of movement and collision avoidance. The agent framework invokes tasks in VR-Forces that are predetermined by the task definition, which again defines the low-level AI (Figure 97). This has lead to patterns of movement that can be categorized unsuitable for the situation. This chapter will present two different occurrences were this is the case. The first case is a result of moving in formation and the second shows a formation where the leader is static and the platoon member are moving into formation.

In addition each agent calculates its own direction vector based on the positions sent to it from VR-Forces. The direction vector gives the direction of the movement, but it is an individual determined variable. Depending on the situation the direction vector of the agents can be different, which means that the platoon members have a different view upon the platoon's direction of movement. The direction vector is used to calculate positions in static formations and contact drills (chapter 7.5). When the command is given the platoon member calculates the position to move to, if the direction vector is not common to all the platoon members then the formation will be broken.

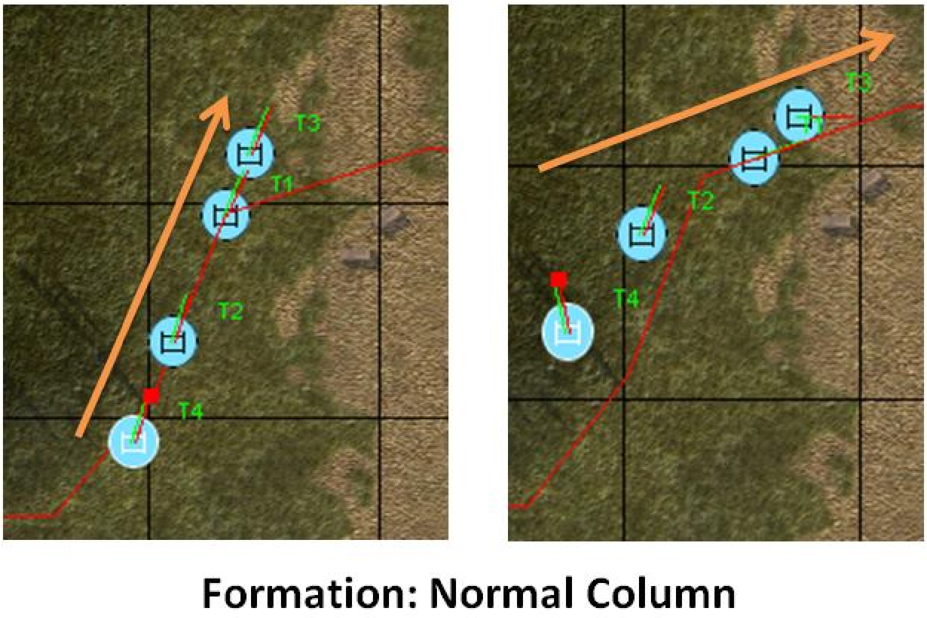


Figure 98 Occurrence 1

Figure 98 show how the formation normal column changes when the leader tank T1 changes it direction of movement. The route that the platoon is following is given by the red line. The Follow Entity task makes the platoon members keep their relative position to T1. The platoon member's movement pattern is considered to be irrational since they make a detour when T1 alters it movement direction. This case is applicable in many other similar cases where for instance a reduction in speed or change in direction at an earlier point in time would have made more sense. Incorporating a solution that reveals the movement intensions of T1 could make the movement patterns more sensible.

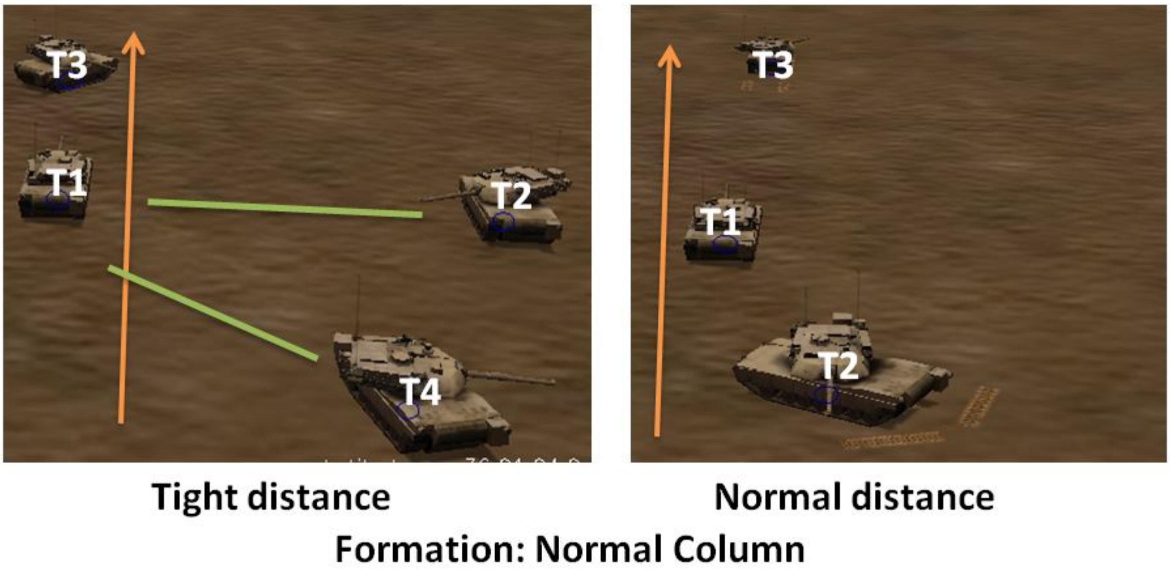


Figure 99 Occurrence 2

Figure 99 shows a case of distance adaptation from VR-Forces. The command to move into formation normal column has been given but the tanks T2 and T4 perceive the distance to the formation position to be so close that movement is unnecessary. When the distance is increased the tanks find movement sensible. Even after the distance change T2 is out of position being too close to T1 and not in line. To which degree the member tanks should adapt their movement is the question that needs to be answered here. In some cases the need to position the tank on the prescribed spot is of great importance in other cases it is not.

Both the above cases address the problem of coordinating an AI that is dived into levels. Even though the number of levels in this case was two the number of levels can vary. To make further adaptations one probably should consider making new tasks in VR-Forces that can utilize a more dynamic range of interactions, where the number of control parameters to be sent with the task invocation can be specialized for the high-level AI. The disadvantage would nevertheless be that such an interface would lead to incompatibility issues.

The original thought was to use UT2004 as the gaming console to control the leader tank T1. After trying out this simulation configuration it became clear that using UT2004 would lead to a less satisfying test scenario presentation. The reason for this is the well known problem of incompatible terrain databases. The member tanks were injected into UT2004 from VR-Forces and the leader tank was injected into VR-Forces from UT2004. With the result that tanks injected into UT2004, were moving above the ground or moving into the ground (Figure 100).

The MÄK Stealth 3D viewer has features that avoid these cases by using ground clamping. This is a feature that puts the object on the terrain surface as long as it is within a certain height-level.

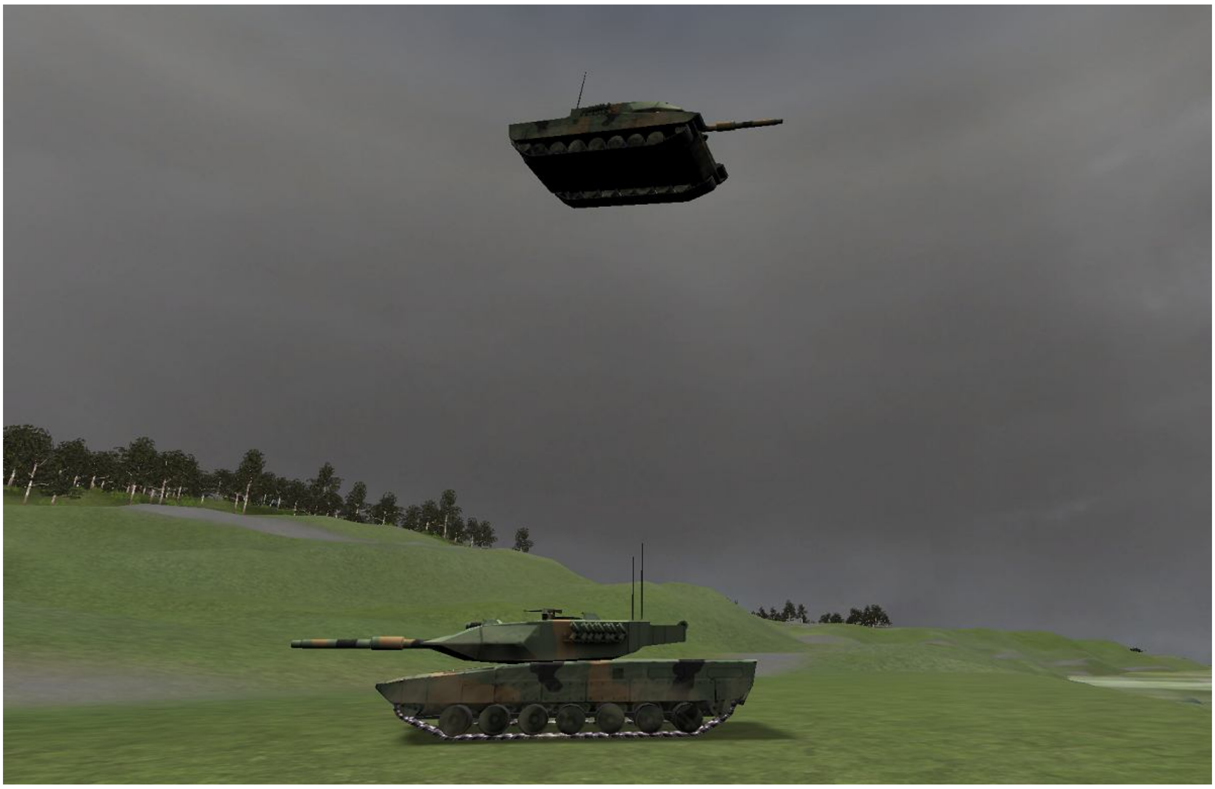


Figure 100 The Flying Tank

The main challenges of creating a tank platoon with a HITL leader is to create the interface between the human contestant and the synthetic entities, and incorporate tactical analysis based on knowledge of the battle space. Translating the intentions of the leader into an understandable format for an AI, demands an extensive knowledge ontology and an interaction schema that closely resembles the operation procedures in a real tank platoon. One of the desired requirements for the Combined Combat Simulator (chapter 7.2) was to communicate via voice, which is a challenge for the future.

The demonstrator has been tested for the basic commands given by the Leader Menu. Some planned features where nevertheless not implemented. This is the target selection algorithm and the actions on “underFire” and “isDestroyed”. Finishing the features that are visible in the test scenarios became the prioritized task when the time to test and write code ran out.

8.3 Lessons Learned

One reason for selecting this master thesis was the challenge of exploring an AI related research field and acquire further knowledge on topics that previously has been unknown to me. I originally wanted to spend more time on developing code, but the preliminary task of investigating the field of HBR and agent frameworks was more time consuming than expected. Effective modeling of human behavior acquires a certain degree of experience, something which is reflected in the demonstrator development process.

More than half of the development time was spent on developing different kinds of interfaces. For instance the graphical user interface, the communication interfaces and the information exchange with VR-Forces. I would have liked to receive more information from VR-Forces, but organizing the information into structures of knowledge is a time consuming job. A standardized interface for exchanging information with VR-Forces would have speeded up the process. Such an interface would have taken care of the processes of parsing messages in the right formats and to the right agents. I had two choices when I started the work on creating the agent structure. Either to incorporate the Java Radio into all agents or make an agent that implements a common interface that is available to all agents. The choice fell on the latter, because in my mind the manager constitutes such a standardized interface to VR-Forces.

I have found that I spent a disproportional amount of time on planning what to do when and where, when it is the assignment in itself that is the important task. The collecting of research literature became a quest of making a taxonomy that covered the field of AI and HBR. This turned out to be a learning process on how to not collect information. The goal or purpose of the collection must be stated before one starts collecting information otherwise one risks getting lost.

Chapter 9 Conclusion, Recommendations and Further Work

HBR and AI are research areas that are still believed to be in its infancy. Nevertheless a more coordinated effort between commercial and academic research environments would benefit the overall progress. The master thesis shows that most academically founded cognitive architectures are complex environments to model within, this is applicable to design, implementation, testing and verification of behavior models. Tools that can facilitate these processes would increase production of computational behavior models.

The different levels of AI in a distributed simulation are seldom synchronized and this is also the case of the demonstrator. Clearly defined interfaces between the different entities are required and if possible one should work towards a standardization of both communication and knowledge ontologies.

Some of the commercial alternatives to behavior modeling incorporate solutions that ease the modeling process at the sacrifice of cognitive features. Nevertheless how the behavior is created should be subordinate to what behavior is produced. If the resulting artificially induced behavior makes sense for the HITL contestants the job has been done.

9.1 Recommendations

The HBR properties found in this master thesis is a simplified pre-taxonomy overview of HBR identifiers. I suggest further development and extension of the HBR properties for later use in framework comparison matters. One can use the already available material provided by Barry G. Silverman (29).

When it comes to choice of agent framework most cognitive architectures seem to have a high threshold of use, which has a discouraging effect on novices. I propose further investigations of commercially available agent frameworks and recommend that one takes a closer look at JACK and CoJACK. In addition one should look at the support for creating crowd behavior.

There is a mismatch between the levels of AI between the demonstrator and VR-Forces. I propose that one investigates this problem area more closely. One can either try to develop new task types in VR-Forces or outsource larger portions of the low-level AI to the agent. Nevertheless one should investigate the possibilities provided by the AI plug-in B-HAVE for VR-Forces. Intelligent behavior on the battle field demands access to information provided by tactical situation analyses, including risk and terrain information.

Last but not least I propose that a pre-assignment for such master degrees is created for future projects. Sadly enough for me the assignment was not an option the autumn semester 2007. If it would have been available I could have spent more time on developing behavior models.

9.2 Further Work

The development process of scenario 2 was not entirely complete when the last test was run. The fire planning and target acquisition goals were not implemented. I suggest that if one desire to continue investigating the Jadex approaches, work on implementing fire planning and firing positions, namely chapter 6 and 7 in (127).

An agent approach to AI development is both scalable and easy to distribute. The JADE framework provides a directory facilitator that allows publication of services. An agent that provides tactical analysis services could be one idea if one desires to extend the platoon members with a larger knowledge base. The service approach also opens for parallel or independent development of a large range of military applications like mission planning and doctrine interpretations, SOPs or ROEs etc.

Further work on providing a common knowledge ontology would ease the development process enormously, on the background of that one has a common way of classifying knowledge both as beliefs and communication objects.

Bibliography

1. **Alberts, David S., Garstka, John J. and Stein, Frederick P.** *Network Centric Warfare - Developing and Leveraging Information Superiority.* Washington : DoD Command and Control Research Program, 2003.

2. **FFI.** *Forskningsplan del 1: 2008-2011.* s.l. : PDC Tangen AS, 2007.

3. **Pew, Richard W. and Mavor, Anne S.** *Modeling Human and Organizational Behavior - Application to Military Simulations.* Washington : National Academy Press, 1998. ISBN 0-309-06096-6.

4. **Mani, Devyani.** *Culture as a Key Element of Human Security.* Nagoya : United Nations Centre for Regional Development, 2004.

5. **Russel, Stuart and Norvig, Peter.** *Artificial Intelligence - A modern Approach.* 2nd. Upper Sadle River : Pearson Education, Inc., 2003.

6. *Building Interactive Virtual Humans for Training Environments.* **Kenny, Patrick, et al.** Marina Del Ray : Interservice/Industry Training, Simulation, and Education Conference, 2007.

7. *Norwegian Armed Forces Joint Operational Doctrine.* Oslo : Brødrene Fossum AS, 2007. 978-82-92566-02-2.

8. *Grader og Distinksjoner i Det Norske Forsvaret.* Oslo : Forsvarets Forum, 2002.

9. **Thibault, D. U.** Commented APP-6A - Military symbols for land based systems. s.l. : Defence Research and Development Canada, September 2005.

10. *NORCCIS at CWID 2008.* **Løkke, Jørgen.** Lillehammer : Norwegian Defence Logistics Organisation CIS Agency, 2008.

11. *Realistic Behaviour Variation in a BDI-based Cognitive Architecture.* **Evertsz, Rick, et al.** Melbourne, Australia : Strategic Independent Agents Alliance, 2008. Proceedings of SimTecT 2008. p. 6.

12. *Uncertainty, Utility, and Misunderstanding: A Decision-Theoretic Perspective on Grounding in Conversational Systems.* **Paek, Tim and Horvitz, Eric.** Cape Cod : AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems, 1999.

13. **Weick, Karl E.** *Sensemaking in Organizations.* Thousand Oaks : SAGE Publications Inc., 1995. 0-8039-7177-X.

14. **Eid, Jarle and Johnsen, Bjørn Helge.** *Operativ Psykologi.* 1st. Bergen : Fagbokforlaget Vigmostad & Bjørke AS, 2005.

15. **Sternberg, Robert J.** *Cognitive Psychology.* 3rd. Belmont : Thomson Wadsworth, 2003. 0-155-08535-2.

16. **Forsyth, Donelson R.** *Group Dynamics.* 4th. Belmont : Thomson Wadsworth, 2006.

17. *Social Identity Theory and the Organization.* **Ashforth, Blake E. and Mael, Fred.** 1, s.l. : The Academy of Management Review, 1989, Vol. 14.

18. **Wooldridge, Michael.** *An Introduction to Multiagent Systems.* Chichester : John Wiley & Sons, Ltd, 2002.

19. *A Robust Layered Control System for a Mobile Robot.* **Brooks, Rodney A.** 1, 1986, IEEE Journal of Robotics and Automation, Vol. 2, pp. 14-23.

20. *Situated Agents Can Have Goals.* **Maes, Pattie.** 1990, Robotics and Autonmous Systems, Vol. 6, pp. 49-70.

21. **Ferguson, Innes A.** *TouringMachines: Autonomous Agents with Attitudes.* Cambridge : Computer Laboratory, University of Cambridge, 1992. 250.

22. *Contentions Scheduling and the Control of Routine Activities.* **Cooper, Richard and Shallice, Tim.** 4, s.l. : Cognitive Neuropsychology, 2000, Vol. 17, pp. 297-338.

23. **Fischer, Klaus, Müller, Jörg P. and Pischel, Markus.** *Unifying Control in Layered Agent Architecture.* Kaiserslautern : German Research Center for Artificial Intelligence, 1994. TM-94-05.

24. **Løvlid, Rikke Amilde.** Lecture 9: Communication. *Lecture Powerpoint presentation in TDT4280.* Trondheim : NTNU, April 2008.

25. *Exploring the Constraints of Human Behavior Representation.* **Giordano, john C., Reynolds, Paul F. and Brogan, David C.** Charlottesville : Proceedings of the 2004 Winter Simulation Conference, 2004. 2004 ACM/IEEE.

26. **Silverman, Barry G., et al.** *A preliminary Investigation into Performance Moderator Functions for Synthetic Agents Development Efforts.* CIS Department, University of Pennsylvania. Philadelphia : Center for Human Modeling and Simulation, 2000.

27. **Morrison, John E.** *A Review of Computer-Based Human Behavior Representations and Their Relation to Military Simulations.* Alexandria, VA, USA : Institute for Defence Analyses (IDA), 2003. p. 152, Review.

28. **Ritter, Frank E., et al.** *Techniques for modelling human performance in synthetic environments: A supplementary review.* Wright-Patterson Air Force Base. s.l. : Human Systems Information Analysis Center, 2003.

29. **Silverman, Barry G., et al.** *Performance Moderator Functions for Human Behaviour Modeling in Military Simulations.* Systems Engineering Department & Institute for Research in Cognitive Sciences (IRCS), University of Pennsylvania. Philadelphia, PA, USA : Prepared for Human Behavior Program Defense Modeling and Simualtion Office, 2002.

30. **Silverman, Barry G., et al.** *Human Behavior Models for Agents in Simulator Games: Part I - Enabling Science with PMFserv.* Department of Electrical Engineering, University of Pennsylvania. Philadelphia : Ackoff Center for Advancement of System Approaches (ACASA), 2008. Review.

31. **Silverman, Barry G., et al.** *Toward A Human Behavior Models Antology for Synthetic Agent Development.* CIS Department, University of Pennsylvania. Philadelphia : Center for Human Modeling and Simulation, 2001.

32. **Silverman, Barry G., et al.** *Socio-Cultural Games for Training and Analysis.* Department of Electrical and Systems Engineering, University of Pennsylvania. Philadelphia : Asch Center for EthnoPolitical Conflict, 2006. Technical.

33. **Silverman, Barry G., et al.** *Athena's Prism - A Diplomatic Strategy Role Playing Simulation for Generating Ideas and Exploring Alternatives.* Department of Electrical and Systems Engineering, University of Pennsylvania. Alexandria, VA : Institute for Defense Analyses (IDA), 2005. Review.

34. *The Knowledge Level.* **Newell, Allen.** s.l. : Artificial Intelligence, 1982, Vol. 18, pp. 87-127.

35. *Exploring the Dynamics of the Appraisal-Emotion Relationship: A Constraint Satisfaction Model of the Appraisal Theory.* **Nerb, Josef.** 7, London : Cognition & Emotion, 2007, Vol. 21, pp. 1382-1413.

36. *The MicroPsi Agent Architecture.* **Bach, Joscha.** Berlin : In Proceedings of ICCM-5, 2003. ISBN 3-933463-15-7.

37. **Raja, Anita.** Metareasoning in Agent-Based Systems. *Department of Software and Information Systems.* [Online] University of North Carolina at Charlotte, July 26, 2008. [Cited: July 26, 2008.] http://www.sis.uncc.edu/~anraja/AAMAS07/MRABS.htm.

38. **Hudlicka, Eva, et al.** *Predicting Group Behavior from Profiles and Stereotypes.* Cambridge, MA, USA : Charles River Analytics, Inc, 2005. Technical Report R02661.

39. **Gluck, Kevin A. and Pew, Richard W.** *Modeling Human Behavior with Integrated Cognitive Architectures.* Mahwah : Lawrence Erlbaum Associates Publishers, 2005.

40. CNS Technology Website. *CELEST Technology Website.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://profusion.bu.edu/techlab/index.php.

41. APEX. *NASA - Intelligent Systems Division - Apex.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://ti.arc.nasa.gov/projects/apex/.

42. ACT-R. *ACT-R Welcome Page.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://act-r.psy.cmu.edu/.

43. Brahms. *Agent iSolutions Team.* [Online] May 29, 2008. [Cited: June 20, 2008.] http://www.agentisolutions.com/index.htm.

44. CHREST. *CHREST Home Page.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://people.brunel.ac.uk/~hssrrls/chrest/index.php.

45. Cognition and Affect. *Cognition and Affect, Architectures and Emotions.* [Online] March 29, 2008. [Cited: June 20, 2008.] http://www.cs.bham.ac.uk/research/projects/cogaff/.

46. iGEN. *The Cognitive Agent Software Toolkit.* [Online] January 18, 2005. [Cited: June 20, 2008.] http://www.cognitiveagent.com/.

47. *The Cognitive Foundry: A Flexible Platform for Intelligent Agent Modeling.* **Basilico, Justin, Benz, Zachary and Dixon, Kevin R.** Providence : Sandia National Laboratories, 2008. In proceeding 2008 BRIMS Conference.

48. CoJACK. *Autonomous Decision-Making Software.* [Online] February 29, 2008. [Cited: June 20, 2008.] http://www.agent-software.com/products/cojack/index.html.

49. 4CAPS. *Center for Cognitive Brain Imaging at Carnegie Mellom University.* [Online] August 16, 2006. [Cited: June 20, 2008.] http://www.ccbi.cmu.edu/4CAPS/index.html.

50. **Sun, Ron.** CLARION. *Cognitive Science Department at Carnegie Mellon University.* [Online] May 21, 2008. [Cited: June 2008, 2008.] http://www.cogsci.rpi.edu/~rsun/clarion.html.

51. **Richman, Howard B.** EPAM VI. *Pennsylvania HomeSchoolers.* [Online] September 24, 2005. [Cited: June 20, 2008.] http://pahomeschoolers.com/epam/.

52. **Kieras, David E.** EPIC. *Department of Electrical Engineering and Computer Science at University of Michigan.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.eecs.umich.edu/~kieras/epic.html.

53. —. GOMS. *Department of Electrical Engineering and Computer Science at University of Michigan.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.eecs.umich.edu/~kieras/goms.html.

54. IMPRINT. *U.S. Army Research Laboratory.* [Online] September 28, 2005. [Cited: June 20, 2008.] http://www.arl.army.mil/ARL-Directorates/HRED/imb/imprint/Imprint7.htm.

55. IPME. *Alion MA&D Operation.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.maad.com/index.pl/ipme.

56. JACK. *Autonomous Decision-Making Software.* [Online] February 29, 2008. [Cited: June 20, 2008.] http://www.aosgrp.com/products/jack/index.html.

57. Jadex. *Distributed Systems and Information Systems at Department of Informatics University of Hamburg.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://vsis-www.informatik.uni-hamburg.de/projects/jadex/links.php.

58. Jason. *Jason a Java-based interprester for an extended version of AgentSpeak.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://jason.sourceforge.net/JasonWebSite/Jason%20Home.php.

59. JESS. *JESS, the Rule Engine for the JavaTM.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://herzberg.ca.sandia.gov/.

60. MIDAS. *NASA - Human Systems Integration Division.* [Online] October 2005, 2005. [Cited: June 20, 2008.] http://human-factors.arc.nasa.gov/dev/www-midas/.

61. Micro Saint Sharp. *Alion MA&D Operation.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.maad.com/index.pl/micro_saint.

62. PDP++. *department of Psychology at University of Colorado.* [Online] August 21, 2007. [Cited: June 20, 2008.] http://psych.colorado.edu/~oreilly/PDP++/PDP++.html.

63. Emergent. *Department of Psychology at University of Colorado.* [Online] May 12, 2008. [Cited: June 20, 2008.] http://grey.colorado.edu/emergent/index.php/Main_Page.

64. D-OMAR. *D-OMAR - Distributed Operator Model Architecture.* [Online] October 30, 2006. [Cited: June 20, 2008.] http://omar.bbn.com/index.html.

65. **Silverman, Barry.** HBMR. *University of Pennsylvania - School of Engineering & Applied Science.* [Online] March 5, 2008. [Cited: June 20, 2008.] http://www.seas.upenn.edu/~barryg/HBMR.html.

66. **Aylett, Ruth.** The PSI model. *School of Mathematical and Computer Sciences at Heriot-Watt University.* [Online] January 20, 2007. [Cited: June 20, 2008.] http://www.macs.hw.ac.uk/~ruth/psi-refs.html.

67. microPsi. *The MicroPsi Project - building cognitive agents.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.micropsi.org/.

68. R-CAST. *PennState - Intelligent Agents Laboratory.* [Online] February 3, 2007. [Cited: June 20, 2008.] http://agentlab.psu.edu/.

69. SAMPLE. *Charles River Analytics.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.cra.com/publications/2005aykroyd1.asp.

70. Soar. *Sitemaker University of Michigan - Soar Home Page.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://sitemaker.umich.edu/soar/home.

71. *Approaches to Modeling the Effects of Fatigue on Cognitive Performance.* **Gunzelmann, Glenn and Gluck, Kevin A.** Providence : The Air Force Research Laboratory, 2008. In Proceedings of the 2008 BRIMS Conference.

72. **Van Dongen, Hans P. A. and Dinges, David F.** Circadian Rhythms in Fatigue, Alertness and Performance. [book auth.] M. H. Kryger, T. Roth and W. B. Dement. *Principles and Practice of Sleep Medicine.* 3rd Edition. Philadelphia : W. B. Saunders, 2000, pp. 391-399.

73. *A model to predict fatigue degraded performance.* **French, Jon.** Orlando : Proceedings of the 2002 IEEE 7th Conference, 2002. Human Factors and Power Plants. pp. 4-6 - 4-9.

74. *A physiological strain index to evaluate heat stress.* **Moran, Daniel S., Shitzer, Avraham and Pandolf, Kent B.** 1, 1998, Am J Physiol Regulatory Integrative Comp Physiol, Vol. 275, pp. 583-586.

75. **Hursh, Steven R.** Fatigue and Alertness Management using the SAFTE™model and FAST™. s.l. : Science Applications International Corporation, September 11, 2003.

76. **Chaiken, Scott R.** *A Verification and Analysis of the USAF/DOD Fatigue Model and Fatigue Management Technology.* Brooks : Air Force Research Laboratory, 2005. Technical.

77. **Lennè, Michael, et al.** *Review of recent research in Applied Experimental Psychology: Implications for Countermeasure Development in road Safety.* Melbourne : Monash University Accident research Centre, 2004. Review.

78. **Silverman, Barry G., et al.** *Human Behavior Models for Agents in Simulators and Games: Part II - Gamebot Engineering with PMFserv.* Department of Electrical and systems Engineering, University of Pennsylvania. Philadelphia : Ackoff Center for Advancement of Systems Approaches (ACASA), 2004. Review.

79. *Desiderdata for cognitive architectures.* **Sun, Ron.** 3, September 2004, Philosophical Psychology, Vol. 17.

80. **Dalal, Michael, et al.** An Architecture for Modeling Human Performance in Applied HCI Domains. Edinburgh, Scotland : NASA Ames Research Center, August 1, 2001.

81. *Apex Reference Manual 3.0 Beta .* Moffett Field : NASA Ames Research Center, 2005.

82. **Freed, Michael, et al.** Apex Project Overview. s.l. : NASA, 2005.

83. **Bothell, Dan.** ACT-R 6.0 Reference Manual. s.l. : ACT-R Research Group, June 11, 2008.

84. ACT-R 6.0 Tutorial: Unit 1. s.l. : ACT-R Research Group, June 24, 2008.

85. **Bothell, Dan.** Extending ACT-R. *Powerpoint Presentation.* s.l. : ACT-R Research Group, July 17, 2007.

86. *ACT - A Simple Theory of Complex Cognition.* **Anderson, John R.** 4, April 1996, American Psychologist, Vol. 51, pp. 355-365.

87. *An Integrated Theory of the Mind.* **Anderson, John R., et al.** 4, 2004, Psychological Review, Vol. 111, pp. 1036-1060.

88. *Brahms: a multi-agent modelling environment for simulating work processes and practices.* **Sierhuis, Maarten, Clancey, William J. and van Hoof, Ron J.J.** 3, 2007, Int. J. Simulation and Process Modelling, Vol. 3, pp. 134-152.

89. *On Short-Term and Long-Term Memory for Brahms Agents.* **Sierhuis, Maarten and van Hoof, Ron.** Stanford : AAAI Press, 2004. AAAI Spring Symposium 2004 Workshop on Interaction between Humans and Autonomous Systems over Extended Operations.

90. **Kieras, David E.** *EPIC Architecture Principles of Operation.* Ann Arbor : University of Michigan, 2004. Technical.

91. *Cognitive Theory, Soar.* **Lewis, Richard L.** Amsterdam : Pergamon (Elsevier Science), 2001, International Encylopedia of the Social and Behavioral Sciences.

92. **Laird, john E., Congdon, Clare Bates and Coulter, Karen J.** The Soar User's Manual Version 8.6.3. s.l. : University of Michigan, October 31, 2006.

93. **Lehman, Jill Fain, Laird, John and Rosenbloom, Paul.** *A Gentel Introduction to Soar, An Architecture for Human Cognition: 2006 Update.* s.l. : University of Michigan, 2006. Technical.

94. *Modeling Rational Agents within a BDI-Architecture.* **Rao, Anand s: and Georgeff, Michael P.** s.l. : Morgan Kaufmann publishers Inc., 1991. Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91). pp. 473-484.

95. *Plans and Resource-Bounded Practical Reasoning.* **Bratman, Michael E., Israel, David J. and Pollack, Martha E.** 1988, Computational Intelligence, Vol. 4, pp. 349-355.

96. *Behavior Modeling in Commercial Games.* **Diller, David E., et al.** s.l. : In proceedings of conference on Behaviour Representation in Modeling and Simualtion, May 2004, 2004. 04-BRIMS-079.

97. **Millington, Ian.** *Artificial Intelligence for Games.* 1st. San Francisco : Morgan Kaufmann Publishers, 2006.

98. **Hafnor, Hilde, Reitan, Bård and Hadzic, Dinko.** *Virkelig (sam)arbeid i en 3D virtuell verden?* s.l. : Forsvarets forskningsinstitutt/Norwegian Defence Research Establishment (FFI), 2007. FFI-rapport. ISBN 978-82-464-1284-9.

99. VT MÄK VR-Forces. *VT MÄK.* [Online] MÄK Technologies, June 13, 2008. [Cited: June 13, 2008.] http://www.mak.com/products/vrforces.php.

100. *VR-Forces User's Guide.* Cambridge : MÄK Technologies, 2006.

101. Beyond Unreal Tournament. *BeyondUnreal - Covering Unreal Tournament 2004, Unreal Tournament 3 & Gears of War.* [Online] June 13, 2008. [Cited: June 13, 2008.] http://www.beyondunreal.com/.

102. VBS2 VTK Virtual Training Kit. *Virtual Battle Space.* [Online] Bohemia Interactive, June 13, 2008. [Cited: June 13, 2008.] http://virtualbattlespace.vbs2.com/.

103. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules. New York : The Institute of Electrical and Electronics Engineers, Inc., September 21, 2000.

104. *The Introduction On High Level Architecture (HLA) and Run-time Infrastructure (RTI).* **Xiaoxia, shi and Qiuhai, Zhong.** Fukui : Fukui University, 2003. SIC Annual Conference.

105. **Wu, Chun-Hsien.** HLA Introduction. *Powerpoint Presentation.* March 18, 2003.

106. **Bellifememine, Fabio, et al.** JADE Programmer's Guide. s.l. : Telecom Italia S.p.A., June 18, 2007.

107. *SimBionic User Guide.* San Mateo : Stottler Henke Associates, Inc, 2005.

108. **Best, Bradley J. and Lebiere, Christian.** Cognitive Agents Interacting in Real and Virtual Worlds. [book auth.] Ron Sun. *Cognition and Multi-Agent Interaction: From Cognitve Modeling to Social Simulation.* New York : Cambridge University Press, 2006, pp. 186-218.

109. **Kieras, David E.** A Tutorial on Building Cognitive Models with the EPIC Architecture for Human Cognition and Performance. *Powerpoint Presentation.* s.l. : University of Michigan, 2004b.

110. **Pearson, Douglas.** Guide for Moving from SGIO to SML. s.l. : ThreePenny Software LLC, June 9, 2005.

111. Soar Technology. *Soar Technology, Inc.* [Online] June 20, 2008. [Cited: June 20, 2008.] http://www.soartech.com/home.php.

112. **Pokahr, Alexander and Braubach, Lars.** *Jadex User Guide.* Hamburg : University of Hamburg, 2007.

113. —. *Jadex Tool guide.* Hamburg : University of Hamburg, 2007.

114. JACK Development Environment Manual. Victoria : Agent Oriented Software Pty. Ltd., June 10, 2005.

115. *Crowd Simulation for Emergency Response Using BDI Agent-based on Virtual Reality.* **Shendarkar, Ameya, et al.** s.l. : In Proceeding of the 2006 Winter Simulation Conference, 2006.

116. *Joint Non-Kinetic Effects Model (JNEM) - A Six Month Success Story.* **Chamberlain, Robert and Metivier, Timothy.** s.l. : In Proceedings of Interservice/Industry Training, Simulation and Eduction Conference (I/ITSEC), 2006.

117. *Developing a Crowd Federate for Military Simulation.* **Petty, Mikel D., et al.** Arlington : In Proceedings of the Spring 2004 Simulation Interoperability Workshop (SIW). , 2004.

118. *A Cognitive Agent-based Approach to Varying Behaviours in Computer Generated Forces Systems to Model Scenarios like Coalitions.* **Fletcher, Martyn.** s.l. : In Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications, 2006.

119. *Designing Physical Layer Components in a Reconfigurable Crowd Federate.* **McKenzie, Frederic D., et al.** San Diego : Proceedings of the Spring 2005 Simulation Interoperability Workshop, 2005.

120. *A General Algebraic Structure for Situation Analysis.* **Maupin, Patrick and Jousselme, Anne-Laure.** Valcartier : In Proceeding on IEEE Conference on Information Fusion, 2005. ISBN 0-7803-9286-8.

121. *Interpreted Systems - A Possible Framework for Situation Analysis Applications.* **Hansen, Bjørn Jervell, Jousselme, Anne-Laure and Maupin, Patrick.** s.l. : FFI, 2006.

122. *Situation Awareness of Commanders: A Cognitive Model.* **Juarez-Espinosa, Octavio and Gonzales, Cleotide.** Providence : In Proceedings of 2004 BRIMS Conference, 2004.

123. *Using Cognitive Models for Computer Generated Forces and Human Tutoring.* **Livak, Tom, Heffernan, Neil and Moyer, Dale.** s.l. : In Proceeding of the 2004 BRIMS Conference, 2004.

124. *Battle Management Language Transformations.* **Bernard, Frederic, et al.** Neuilly-sur-Seine : In Proceeding Transforming Training and Experimentation through Modelling and Simulation, 2006.

125. *Modeling Rules of Engagement in Computer Generated Forces.* **Evertsz, Rick, et al.** Orlando, FL: U. of Central Florida : Behavior Representation in Modeling & Simulation (BRIMS) conference, 2007. Proceedings of the 16th Conference on Behavior Representation in Modeling and Simulation.

126. *Field Manual 17-15: Tank Platoon.* Washington : Headquarters, Department of the Army, 1996.

127. *Forsvarets Reglement 7-0-3: Stridsvogntroppen.* Rena : Hærens Transformasjons og doktrinekommando (TRADOK), 2008.

128. Panserbataljonen - Stridsvogneskadron 2. [Online] July 17, 2008. [Cited: July 17, 2008.] http://www.mil.no/haren/styrker/pbn/start/stridsesk2/.

129. **Svenson, Ola and Maule, A. John.** *Time Pressure and Stress in Human Judgement and Decision Making.* New York : Plenum Press, 1993.

130. **Lervik, Else and Havdal, Vegard B.** *Programmering i Java.* 2. Oslo : Stiftelsen TISIP og Gyldendal Norsk Forlag, 2003.

131. **Lewis, John and Loftus, William.** *Java Software Solutions - foundations of program design.* 3rd. s.l. : Addison-Wesley, 2004.

132. *AI Support for Building Cognitive Models.* **Amant, Robert St., McBride, Sean P. and Ritter, Frank E.** Menlo Park : AAAI Press, 2006. Proceedings of the Twenty-First National Conference on Artificial Intelligence. pp. 1663-1666.

133. *CoJack - Acheiving Principled Behaviour Variation in a Moderated Cognitive Architecture.* **Evertsz, Rick, et al.** Orlando, FL: U. of Central Florida : Behavior Representation in Modeling & Simulation (BRIMS) conference, 2008. Proceedings of the 17th Conference on Behavior Representation in Modeling and Simulation.

134. **Tolk, Andreas.** Human Behaviour Representation - Recent Development. RTO-ENP-017 *Paper presented at the RTO SAS Lecture Series on "Simulation of and for Military Decision Making.".* Hague, The Nederlands : NATO Research & Technology Organization - Studies, Analysis & Simulation, December 2002.

135. **Newell, Allen.** *Unified Theories of Cognition.* Cambridge : Harvard University Press, 1990.

136. COGENT. *COGENT online.* [Online] May 6, 2005. [Cited: June 20, 2008.] http://cogent.psyc.bbk.ac.uk/.

137. **Hollnagel, Erik.** CREAM. *Department of Computer and Information Science - University of Linköping.* [Online] May 31, 2006. [Cited: June 20, 2008.] http://www.ida.liu.se/~eriho/CREAM_M.htm.

138. **Braubach, Lars and Pokahr, Alexander.** *Jadex Tutorial.* Hamburg : University of Hamburg, 2007.

139. **Noy, Natalya F. and McGuinness, Deborah L.** *Ontology Development 101: A Guide to Creating Your First Ontology.* Stanford : Stanford Knowledge Systems Laboratory, 2001. Technical.

140. **Öztürk, Pinar.** Lecture 3: Intelligent agents - part 2: Deductive and Practical Reasoning Agents. *Lecture Powerpoint Presentation in TDT4280.* Trondheim : NTNU, July 2008.

Appendix A: Definitions

Appendix B: Frameworks for Modeling HBR Abbreviations

Model Name	Acronym/ Abbriviation
Adaptive Resonance Theory	ART
Architecture for Procedure Execution	APEX
Atomic Components of Thought - Rational	ACT-R
Business Redesign Agent-Based Holistic Modeling System	Brahms
Chunk Hierarchy and Retrieval Structures	CHREST
Cognition and Affect Project	CogAff
Cognition as a Network of Tasks	COGNET
Cognitive Agent Software Toolkit	iGEN
Cognitive Complexity Theory	CCT
Cognitive Environment Simulation	CES
Cognitive Foundry	CF
Cognitive JACK	CoJACK
Cognitive Objects within a Graphical Environment	COGENT
Cognitive Reliability and Error Analysis Method	CREAM
Cognitive Simulation Model	COSIMO
Concurrent Activation-Based Production System	CAPS
Concurrent Capacity-Constrained Activation-Based PS	4CAPS
Connectionist Learning with Adaptive Rule Induction ON-line	CLARION
Construction-Integration Theory	C-I Theory
Distributed Cognition	DCOG
Elementary Perceiver and Memoriser	EPAM
Executive Process /Interactvie Control	EPIC
Goals, Operators, Methods, and Selection rules	GOMS
GOMS Language Evaluation and Analysis	GLEAN
Human Operator Simulator	HOS
Improved Performance Research Integration Tool	IMPRINT
Integrated Performance Modelling Environment	IPME
JACK™ Intelligent Agents	JACK
Jadex	JADEX
Jason	Jason
Java Expert System Shell	JESS
Latent Sematic Analysis	LSA
Linked model of Comprehension-based Action planning Instruction	LICAI
Man-Machine Integrated Design and Analysis System	MIDAS
Micro Systems Analysis of Integrated Network of Tasks	Micro SAINT
Neural Network Simulation System	PDP++
Neural Network Simulation System	Emergent
Operator Model Architecture	OMAR
Performance Moderator Function Server	PMFserv
Procedural Reasoning System	PRS
PSI Theory	PSI
PSI Theory	Micro PSI
RPD enabled Collaborative Agents for Simulating Teamwork	R-CAST
Sim Agent Toolkit	SIM_AGENT
Simplified Model of Cognition	SMoC
Situation Awareness Model for Pilot-in-the-loop Evaluation	SAMPLE
State, Operator and Result	Soar

Appendix C: Frameworks for HBR and their Original Purpose

Acronym/ Abbriviation	Original Purpose
ART	It is a learning model using neural networks.
APEX	Model human performance in complex and dynamic environments.
ACT-R	Theory on human cognition and a framework for modeling task performance.
Brahms	Software framework for developing and simulating multi-agent models.
CHREST	An architecture that models human perception, learning, memory and problem solving.
CogAff	A multidisciplinary project for research on cognition and the enhancement of AI.
COGNET	A tool for facilitating cognitive task analysis and description of specific work domains.
iGEN	A tool for facilitating cognitive task analysis and description of specific work domains.
CCT	To provide the theoretical base for engineering human computer interactions.
CES	Clarification of cognitive demands on different problem-solving situations.
CF	Unified collection of tools for Cognitive Science and Technology applications.
CoJACK	Used for modeling the variation in human behavior in BDI architecture JACK.
COGENT	Modeling system for developing and exploring models of cognitive processes.
CREAM	Methodology for analyzing human performance.
COSIMO	Cognitive simulation model for decision making and behavior in accident management.
CAPS	A simple production system for modeling reading.
4CAPS	Networked collaborative modules that simulate brain cortical areas.
CLARION	Cognitive architecture for modeling human cognition and thought processes.
C-I Theory	Sentence comprehension and sentence processing.
DCOG	Modeling of expert behavior in complex work domains where there exist multiple methods.
EPAM	Psychological theory of learning and memory
EPIC	Cognitive architecture that accounts for human perceptual, cognitive and motor activity.
GOMS	A language to describe human computer interactions.
GLEAN	Software modeling environment for developing and analyzing simulations of HCI.
HOS	Provide a model of human capabilities and limitations to support design of HMS.
IMPRINT	Human Systems Integration (HSI) and Manpower and Personnel Integration tool.
IPME	Integrated environment of simulation and modeling tools for human performance.
JACK	Cross-platform environment for building, running and integrating commercial-grade MAS.
JADEX	Cross-platform environment for building, running and integrating open source-grade MAS
Jason	Java-based interpreter for an extended version of AgentSpeak.
JESS	Rule engine and scripting environment and a tool for building Expert Systems.
LSA	Automated corpus-based system for deriving meanings of individual words, sentences etc.
LICAI	To comprehend incomplete instructions from systems with GUIs.
MIDAS	Engineering aid during a system design process modeling human/automation interactions.
Micro SAINT	General purpose, discrete-event simulation software tool.
PDP++	Neural-network simulation system.
Emergent	Simulation environment for creating models of the brain and cognitive processes using ANN.
OMAR	Provides a suite of software tools to support the development of simulations using agents.
PMFserv	Unified behavior architecture interconnecting PMFs.
PRS	Architecture for real-time reasoning and system control.
PSI	A computational theory of psychology on cognitive, emotional and motivational processes.
Micro PSI	Describes the interaction of emotion, motivation and cognition of situated agents.
R-CAST	Tool for developing and studying high-level cognitive behaviors and team behaviors.
SIM_AGENT	Toolkit for the development of interacting agents in different environments.
SMoC	Perceptual cycle of decision-making processes.
SAMPLE	Model individual and crew SA in air combat, later on SA also in other environments.
Soar	General cognitive architecture for developing systems that exhibit intelligent behavior.

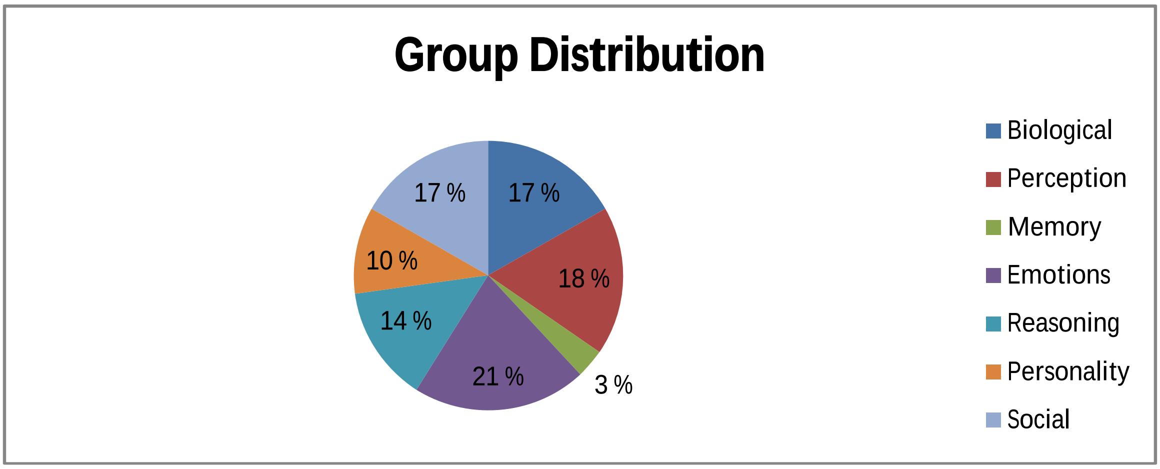
Appendix D: Properties of PMFserv

Architecture Property Sheet 1					
Group	**Biological**				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	Reserviors for sleep, energy etc.		1	
	Activity	Reserviors for sleep, energy etc.		1	
	Damage	Injury model		1	
	Fitness	None		0	
	Environment	Temperature		1	
Point Total				4	
Total Weight			**0,14**	**0,80**	**0,11**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	In perception object		1	
	Auditory	In perception object		1	
	Tacitile	In perception object		1	
	Olfanctory	In perception object		1	
	Gustatory	In perception object		1	
	Direct	Present		1	
	Constructive	None		0	
Point Total				6	
Total Weight			**0,14**	**0,86**	**0,12**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	None		0	
	STM	None		0	
	WM	None		0	
	Retrival PMF	None		0	
	Store PMF	None		0	
	Representation	Perception object		1	
Point Total				1	
Total Weight			**0,14**	**0,17**	**0,02**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	OCC model		1	
	PMF	OCC and GSP value trees		1	
Point Total				2	
Total Weight			**0,14**	**1,00**	**0,14**
Description					

Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Rule based		1	
	PS Representation	Gibbson affordance		1	
	DM Method	Augmented Decision Theory		1	
	Planning	None		0	
	Learning	None		0	
	Operation mode	Parallell		1	
Point Total				4	
Total Weight			**0,14**	**0,67**	**0,10**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	None		0	
	Values	GSP value trees		1	
Point Total				1	
Total Weight			**0,14**	**0,50**	**0,07**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	Role and social networking		1	
	Attraction	Clan alignment, inter personal		1	
	Satisfaction PMF	Agency vs Object		1	
	Conflict PMF	Mob-rules		1	
	Communication	None		0	
Point Total				4	
Total Weight			**0,14**	**0,8**	**0,11**
Description					

Visualisation of Evaluation



Appendix E: Properties of APEX

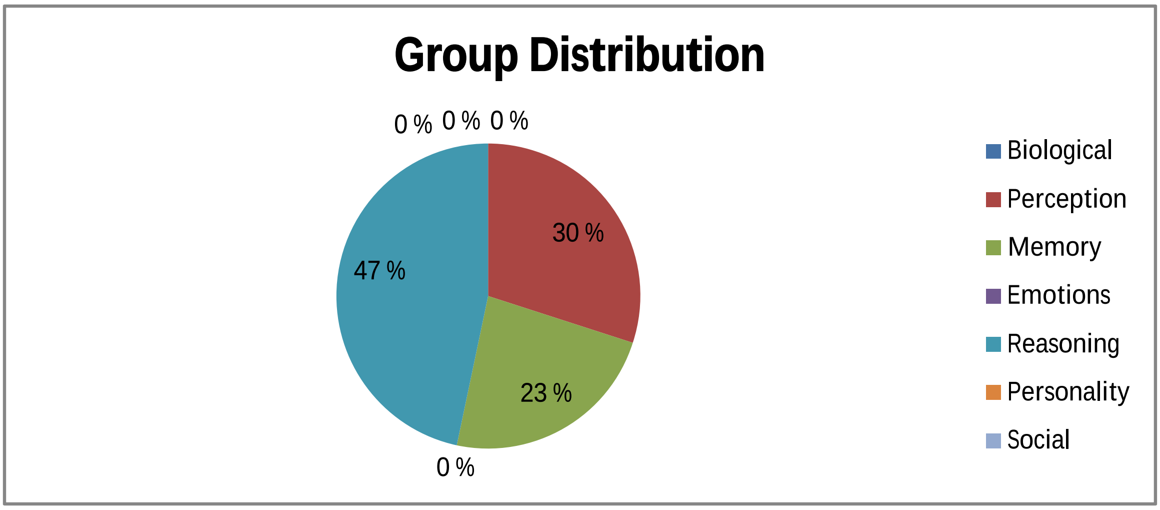
Architecture Property Sheet 1

Group	**Biological**				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	None		0	
	Activity	None		0	
	Damage	None		0	
	Fitness	None		0	
	Environment	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	In template object		1	
	Auditory	In template object		1	
	Tacitile	None		0	
	Olfanctory	None		0	
	Gustatory	None		0	
	Direct	Present		1	
	Constructive	None		0	
Point Total				3	
Total Weight			**0,14**	**0,43**	**0,06**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	Memory of storing beliefs		1	
	STM	None		0	
	WM	None		0	
	Retrival PMF	None		0	
	Store PMF	None		0	
	Representation	Visual, Physical Objects		1	
Point Total				2	
Total Weight			**0,14**	**0,33**	**0,05**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	None		0	
	PMF	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					

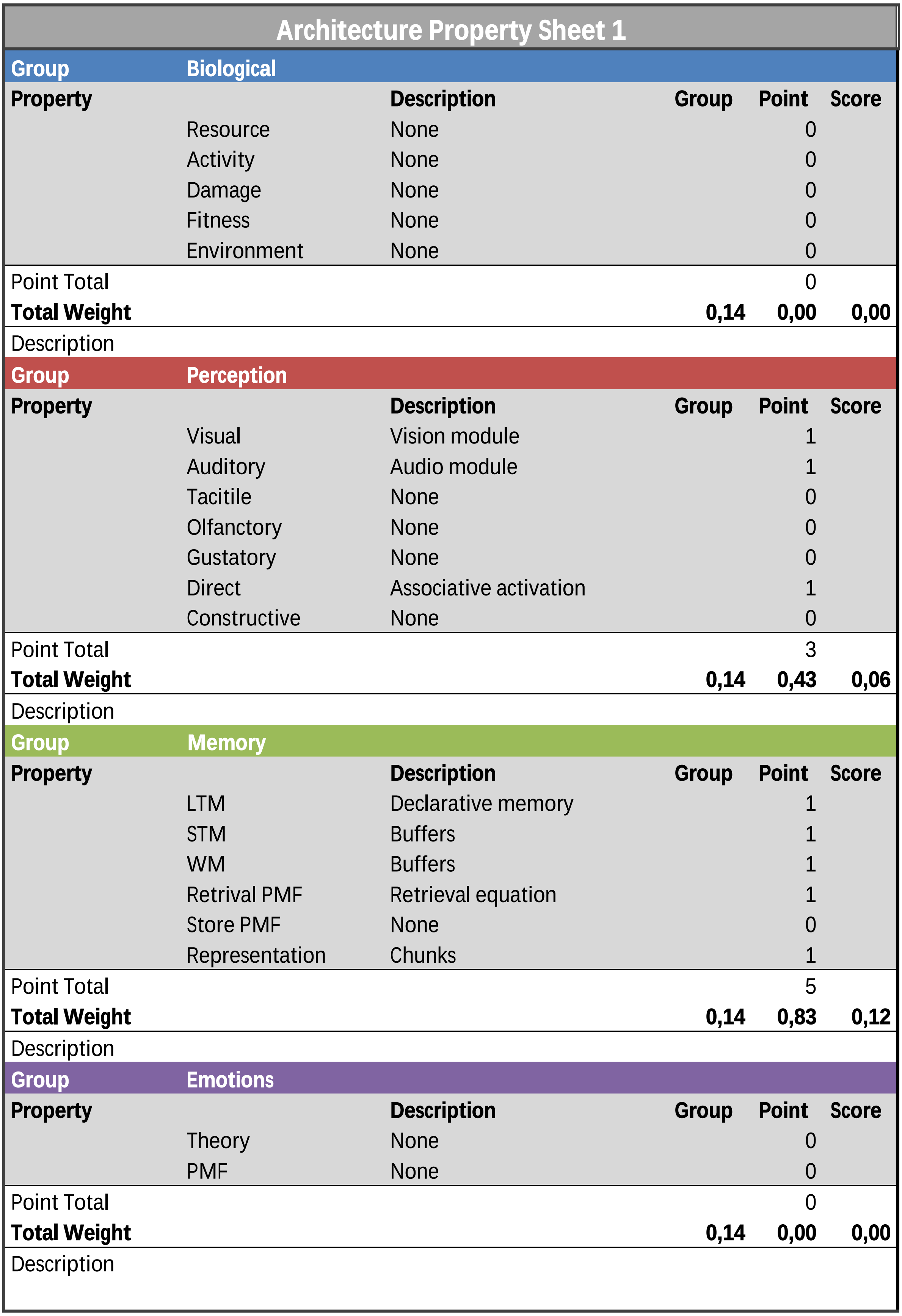
Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Rule-based action selection		1	
	PS Representation	Task Domain		1	
	DM Method	Conditional clauses		1	
	Planning	Implicit		1	
	Learning	None		0	
	Operation mode	Parallell		0	
Point Total				4	
Total Weight			**0,14**	**0,67**	**0,10**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	None		0	
	Values	Noen		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	None		0	
	Attraction	None		0	
	Satisfaction PMF	None		0	
	Conflict PMF	None		0	
	Communication	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					

Visualisation of Evaluation



Appendix F: Properties of ACT-R



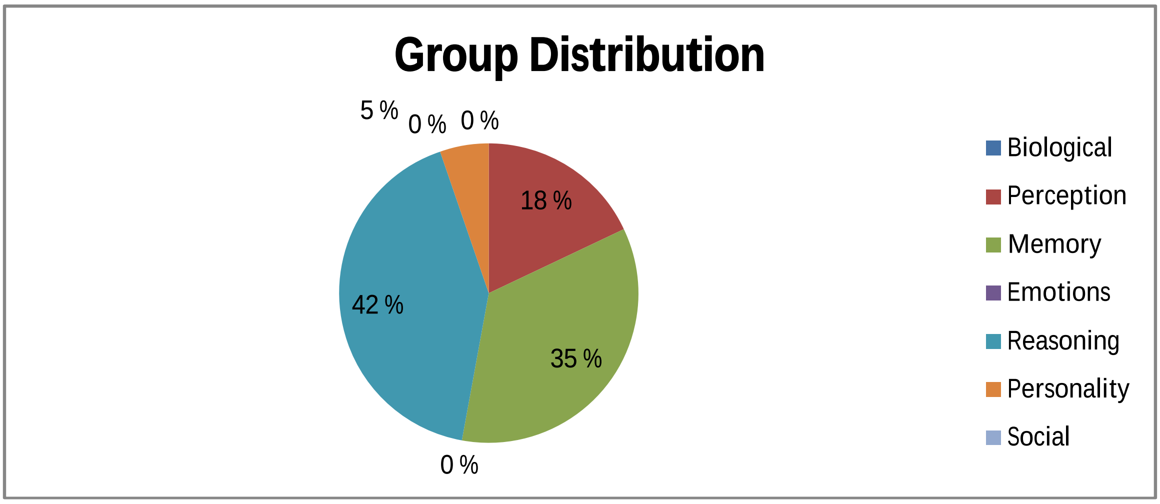
Architecture Property Sheet 1

Group	**Biological**				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	None		0	
	Activity	None		0	
	Damage	None		0	
	Fitness	None		0	
	Environment	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	Vision module		1	
	Auditory	Audio module		1	
	Tacitile	None		0	
	Olfanctory	None		0	
	Gustatory	None		0	
	Direct	Associative activation		1	
	Constructive	None		0	
Point Total				3	
Total Weight			**0,14**	**0,43**	**0,06**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	Declarative memory		1	
	STM	Buffers		1	
	WM	Buffers		1	
	Retrival PMF	Retrieval equation		1	
	Store PMF	None		0	
	Representation	Chunks		1	
Point Total				5	
Total Weight			**0,14**	**0,83**	**0,12**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	None		0	
	PMF	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					

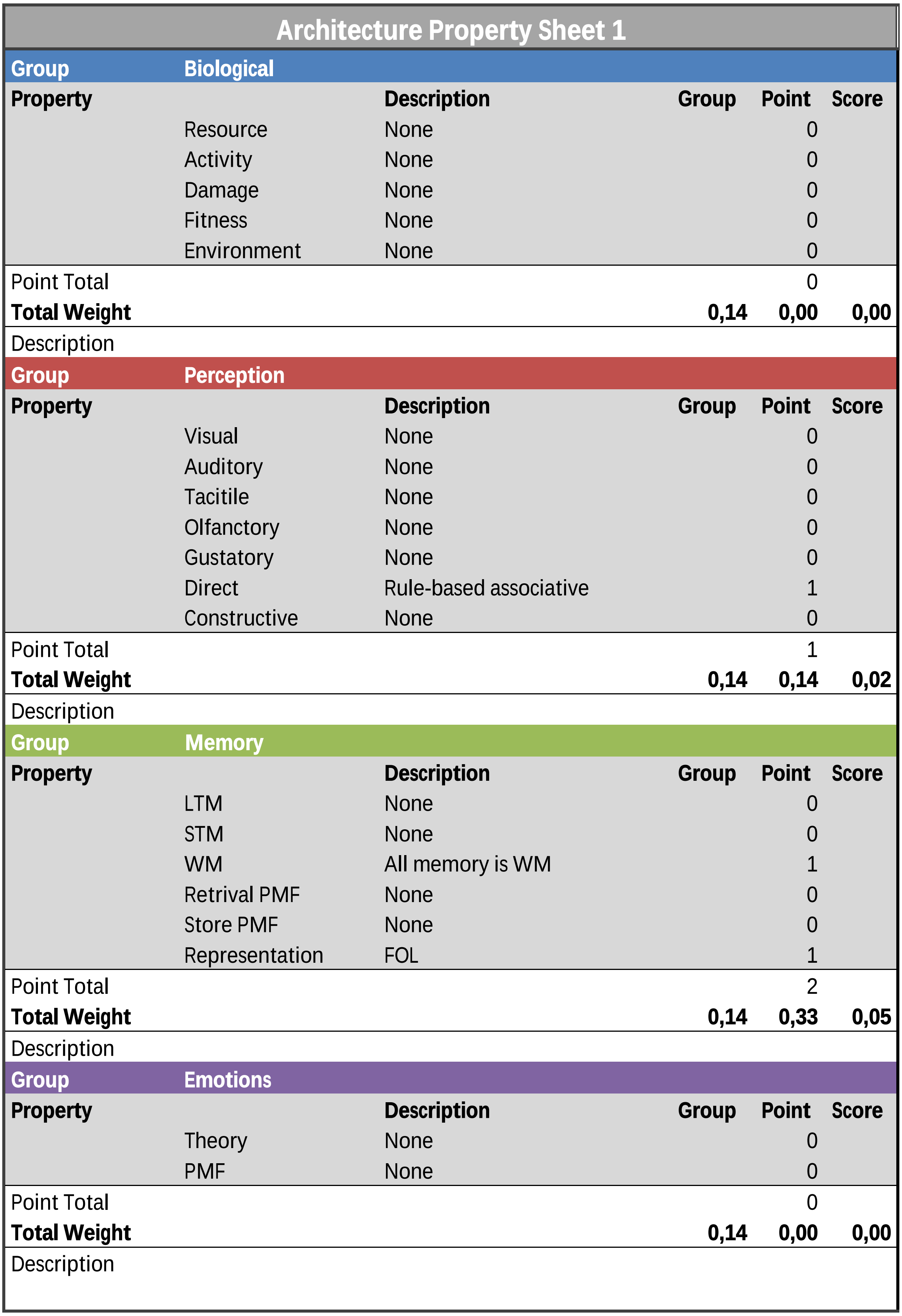
Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Rule-based		1	
	PS Representation	Goal chunks		1	
	DM Method	Production utility & Chunk activation		1	
	Planning	Implicit		1	
	Learning	Learning equation		1	
	Operation mode	Parallel		1	
Point Total				6	
Total Weight			**0,14**	**1,00**	**0,14**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	Parameter setting		0,25	
	Values	None		0	
Point Total				0,25	
Total Weight			**0,14**	**0,125**	**0,02**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	None		0	
	Attraction	None		0	
	Satisfaction PMF	None		0	
	Conflict PMF	None		0	
	Communication	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					

Visualisation of Evaluation



Appendix G: Properties of Brahms



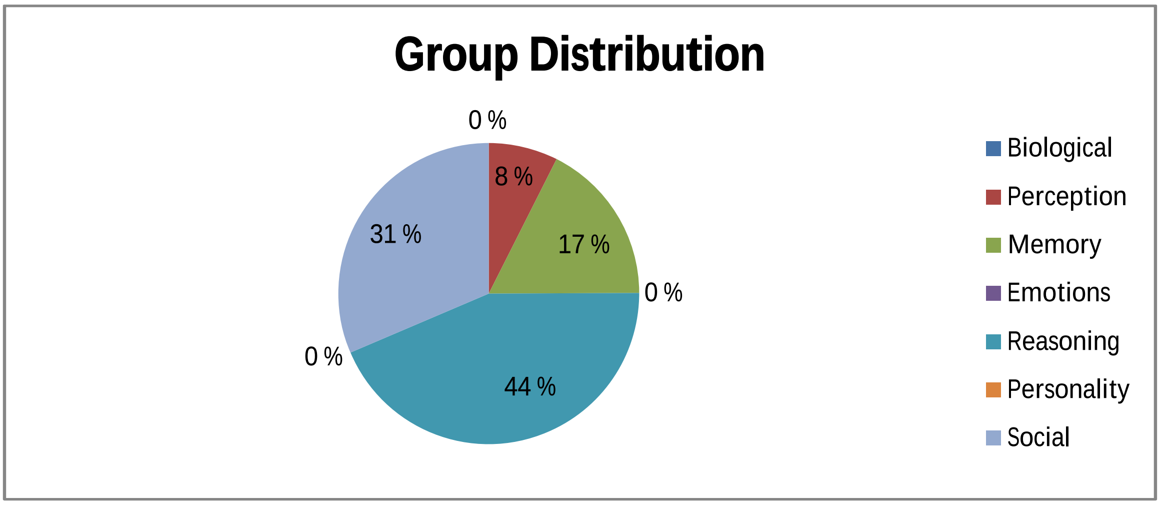
Architecture Property Sheet 1

Group	**Biological**				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	None		0	
	Activity	None		0	
	Damage	None		0	
	Fitness	None		0	
	Environment	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	None		0	
	Auditory	None		0	
	Tacitile	None		0	
	Olfanctory	None		0	
	Gustatory	None		0	
	Direct	Rule-based associative		1	
	Constructive	None		0	
Point Total				1	
Total Weight			**0,14**	**0,14**	**0,02**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	None		0	
	STM	None		0	
	WM	All memory is WM		1	
	Retrival PMF	None		0	
	Store PMF	None		0	
	Representation	FOL		1	
Point Total				2	
Total Weight			**0,14**	**0,33**	**0,05**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	None		0	
	PMF	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					

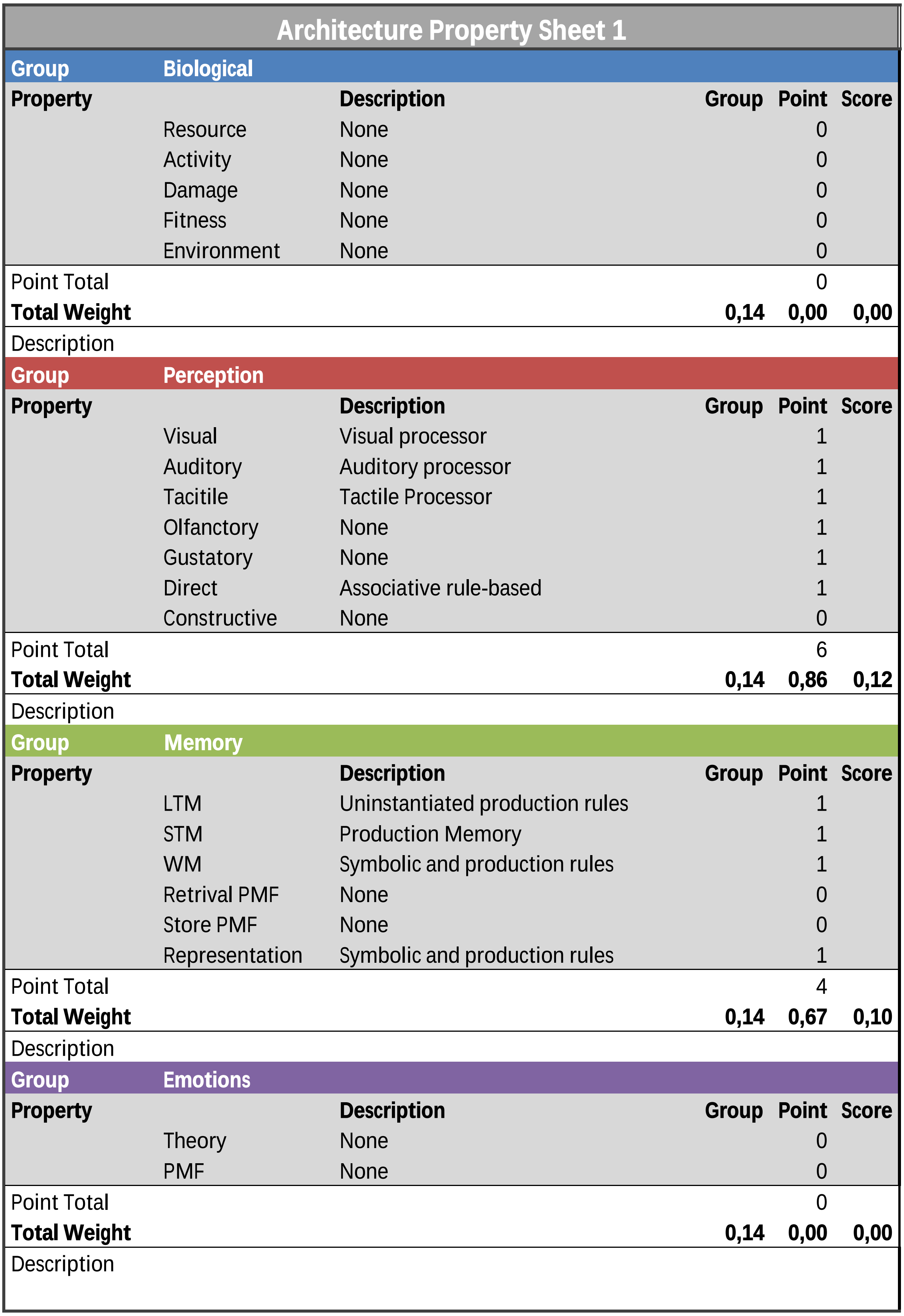
Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Rule-based forward chaining		1	
	PS Representation	FOL		1	
	DM Method	Propositional inference		1	
	Planning	Implicit Activity		1	
	Learning	None		0	
	Operation mode	Parallel		1	
Point Total				5	
Total Weight			**0,14**	**0,83**	**0,12**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	None		0	
	Values	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	Group hierarchy		1	
	Attraction	Group membership		1	
	Satisfaction PMF	None		0	
	Conflict PMF	None		0	
	Communication	Own model		1	
Point Total				3	
Total Weight			**0,14**	**0,6**	**0,09**
Description					

Visualisation of Evaluation



Appendix H: Properties of EPIC

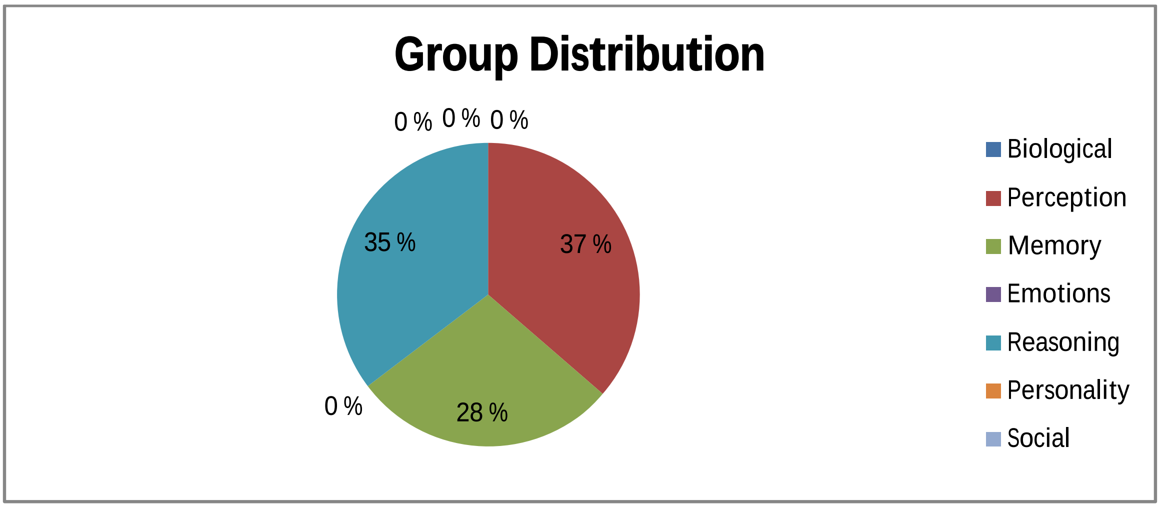


Architecture Property Sheet 1					
Group	**Biological**				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	None		0	
	Activity	None		0	
	Damage	None		0	
	Fitness	None		0	
	Environment	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	Visual processor		1	
	Auditory	Auditory processor		1	
	Tacitile	Tactile Processor		1	
	Olfanctory	None		1	
	Gustatory	None		1	
	Direct	Associative rule-based		1	
	Constructive	None		0	
Point Total				6	
Total Weight			**0,14**	**0,86**	**0,12**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	Uninstantiated production rules		1	
	STM	Production Memory		1	
	WM	Symbolic and production rules		1	
	Retrival PMF	None		0	
	Store PMF	None		0	
	Representation	Symbolic and production rules		1	
Point Total				4	
Total Weight			**0,14**	**0,67**	**0,10**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	None		0	
	PMF	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					

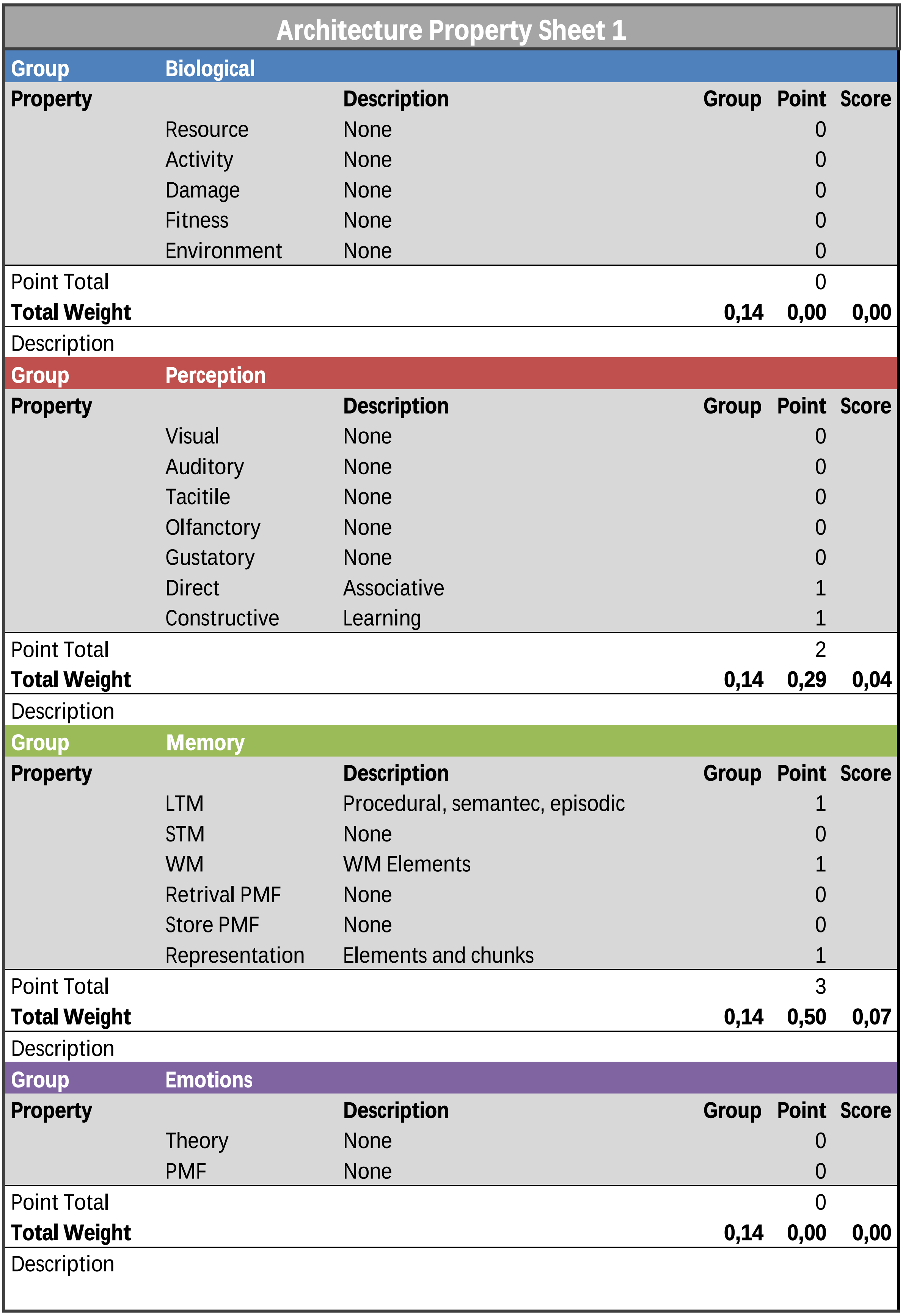
Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Rule-based		1	
	PS Representation	Production rules and events		1	
	DM Method	Conditional rule based		1	
	Planning	Implicit		1	
	Learning	None		0	
	Operation mode	Parallel		1	
Point Total				5	
Total Weight			**0,14**	**0,83**	**0,12**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	None		0	
	Values	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	None		0	
	Attraction	None		0	
	Satisfaction PMF	None		0	
	Conflict PMF	None		0	
	Communication	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					

Visualisation of Evaluation



Appendix I: Properties of Soar



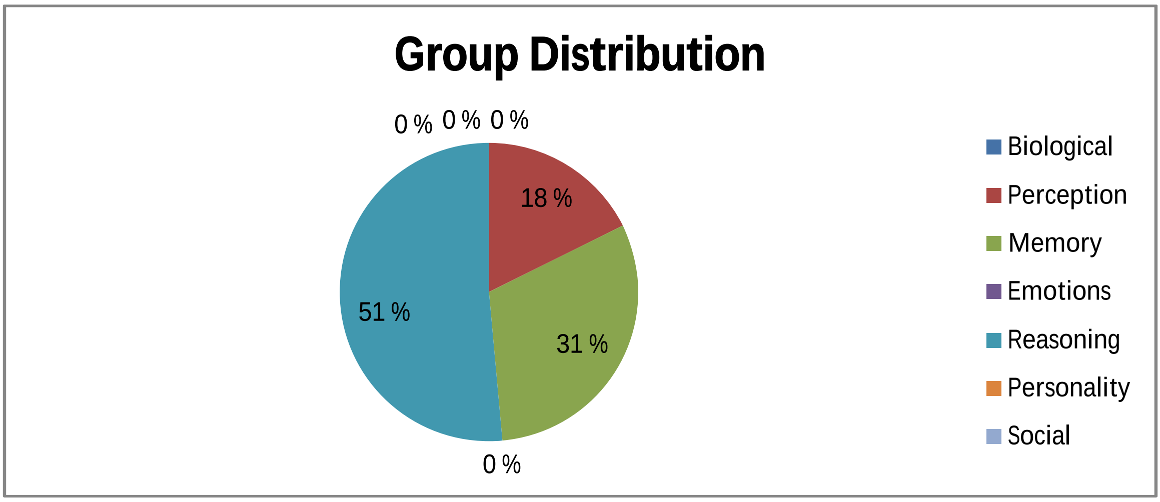
Architecture Property Sheet 1

Group	Biological				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	None		0	
	Activity	None		0	
	Damage	None		0	
	Fitness	None		0	
	Environment	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	None		0	
	Auditory	None		0	
	Tacitile	None		0	
	Olfanctory	None		0	
	Gustatory	None		0	
	Direct	Associative		1	
	Constructive	Learning		1	
Point Total				2	
Total Weight			**0,14**	**0,29**	**0,04**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	Procedural, semantec, episodic		1	
	STM	None		0	
	WM	WM Elements		1	
	Retrival PMF	None		0	
	Store PMF	None		0	
	Representation	Elements and chunks		1	
Point Total				3	
Total Weight			**0,14**	**0,50**	**0,07**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	None		0	
	PMF	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					

Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Forward Lookup Search		1	
	PS Representation	Goal state		1	
	DM Method	Rule-based		1	
	Planning	Implicit		1	
	Learning	Impasse		1	
	Operation mode	Serial		0	
Point Total				5	
Total Weight			**0,14**	**0,83**	**0,12**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	None		0	
	Values	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	None		0	
	Attraction	None		0	
	Satisfaction PMF	None		0	
	Conflict PMF	None		0	
	Communication	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					

Visualisation of Evaluation



Appendix J: Properties of BDI

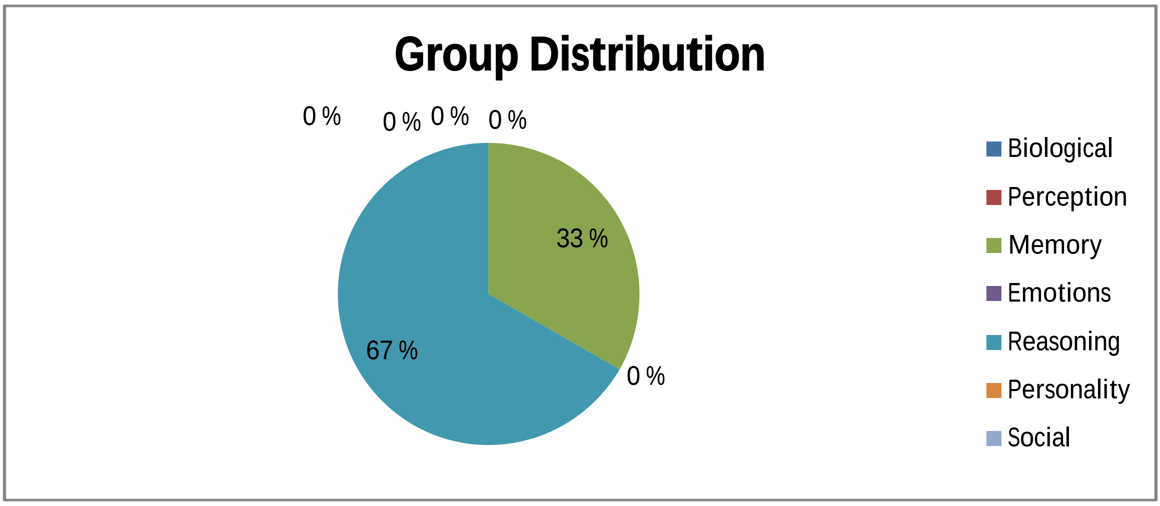
Architecture Property Sheet 1

Group	**Biological**				
Property		**Description**	**Group**	**Point**	**Score**
	Resource	None		0	
	Activity	None		0	
	Damage	None		0	
	Fitness	None		0	
	Environment	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Perception**				
Property		**Description**	**Group**	**Point**	**Score**
	Visual	None		0	
	Auditory	None		0	
	Tacitile	None		0	
	Olfanctory	None		0	
	Gustatory	None		0	
	Direct	None		0	
	Constructive	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					
Group	**Memory**				
Property		**Description**	**Group**	**Point**	**Score**
	LTM	None		0	
	STM	None		0	
	WM	The belief set		1	
	Retrival PMF	None		0	
	Store PMF	None		0	
	Representation	Implementation dependent		1	
Point Total				2	
Total Weight			**0,14**	**0,33**	**0,05**
Description					
Group	**Emotions**				
Property		**Description**	**Group**	**Point**	**Score**
	Theory	None		0	
	PMF	None		0	
Point Total				0	
Total Weight			**0,14**	**0,00**	**0,00**
Description					

Architecture Property Sheet 2

Group	**Reasoning**				
Property		**Description**	**Group**	**Point**	**Score**
	PS Method	Practical reasoning		1	
	PS Representation	Belief, Desires, Intentions		1	
	DM Method	Conditional		1	
	Planning	None		0	
	Learning	None		0	
	Operation mode	Parallel		1	
Point Total				4	
Total Weight			**0,14**	**0,67**	**0,10**
Description					
Group	**Personality**				
Property		**Description**	**Group**	**Point**	**Score**
	Intelligence	None		0	
	Values	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					
Group	**Social**				
Property		**Description**	**Group**	**Point**	**Score**
	Group	None		0	
	Attraction	None		0	
	Satisfaction PMF	None		0	
	Conflict PMF	None		0	
	Communication	None		0	
Point Total				0	
Total Weight			**0,14**	**0**	**0,00**
Description					

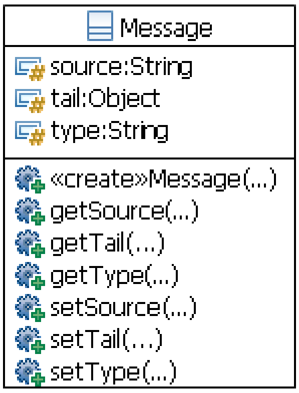
Visualisation of Evaluation

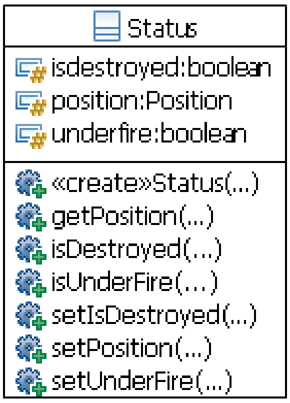


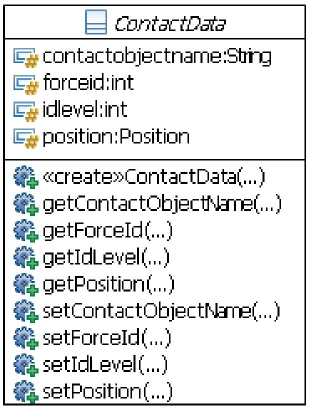
Appendix K: Java Radio Message Configuration

Direction	Message type	Source	Content
In	tmsc	senderObjectName	Sensor contacts
In	tmst	senderObjectName	Status of tank
In	tmde	senderObjectName	Destroyed Contact
Out	tmta	receiverObjectName	Set Target
Out	tmfo	receiverObjectName	Follow
Out	tmre	receiverObjectName	Set rules of engagement
Out	tmsr	receiverObjectName	Set sector of responsibility
Out	tmhe	receiverObjectName	Turn to heading
Out	tmmo	receiverObjectName	Move to Location
Out	tmsp	receiverObjectName	Wait

The messages are organized as message objects within the Jadex environment (see Message, Status and ContactData classes). The class has three attributes, namely source, tail and type. The source is equal to the sending or receiving tank, while the type is equal to the message type. The tail constitutes the content objects which usually are a string, status or contact objects. When received from VR-Forces or passed on to VR-Forces the messages are formatted as strings (see message format).







Message format

*** Messages from VR-Forces ***

--- Sensor contacts message ---

tmsc(senderObjectName){[contactObjectName1;idLevel1;forceId1;contact1PosX,contact1PosY,contact1PosZ][contactObjectName2;idLevel2;forceId2;contact2PosX,contact2PosY,contact2PosZ]...etc}
idLevel takes the values: 1-detected, 2-classified, 3-identified, 4-Full Knowledge
forceId takes the values: 0-other force (default), 1-friendly, 2-enemy
Position is VR-Forces database coordinates
Update condition: every second if there are contacts

--- Status message ---

tmst(senderObjectName){ownPosX,ownPosY,ownPosZ;underFire;isDestroyed}
Update condition: position > 10 meters from the last update, if the entity stops moving, or underFire or isDestroyed changes.

---Target Destroyed Message ---

tmde(senderObjectName){contactObjectName}
The contact object us the contact destroyed in VR-Forces

*** Messages to VR-Forces ***

--- Set Target ---

tmta(recipientObjectName){targetObjectName}
targetObjectName must match the VR-Forces object name

--- Follow Command ---

tmfo(recipientObjectName){targetObjectName;offsetX,offsetY,offsetZ}
targetObjectName must match the VR-Forces object name

--- set Rules Of Engagement ---

tmre(recipientObjectName){roeType}
roeType is a string, supported roeTypes are hold-fire, fire-at-will and fire-when-fired-upon

--- Set Sector of Responsibility ---

tmsr(recipientObjectName){center;size}
center is a double representation of the angle of the center of the section in degrees from north
size is the size in degrees of the sector

--- Set Heading ---

tmhe(recipientObjectName){heading}
heading is a double representing of the heading clockwise from north in degrees

--- Move to location command ---

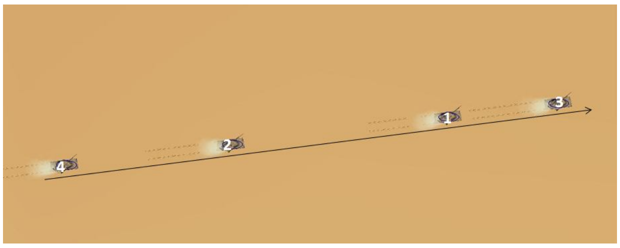
tmmo(recipientObjectName){coordX,coordY,coordZ}
coords are the VR-Forces internal (database) coordinates

--- Wait command ---

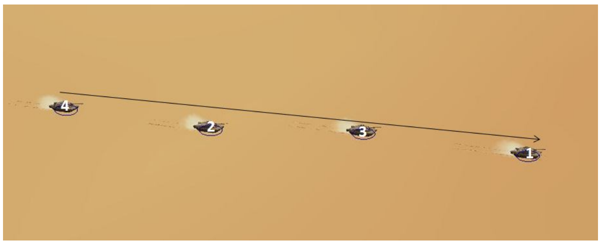
tmsp(recipientObjectName){}

Appendix L: Formation Types

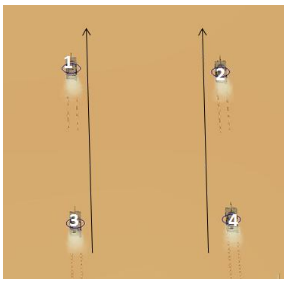
Normal Column



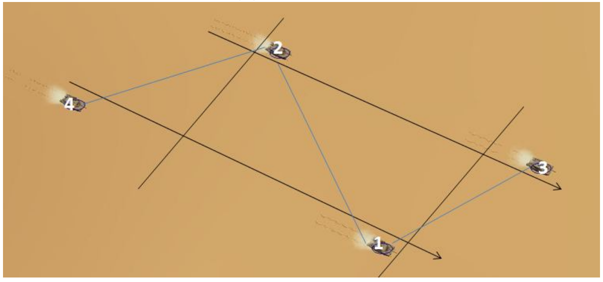
Administrative Column



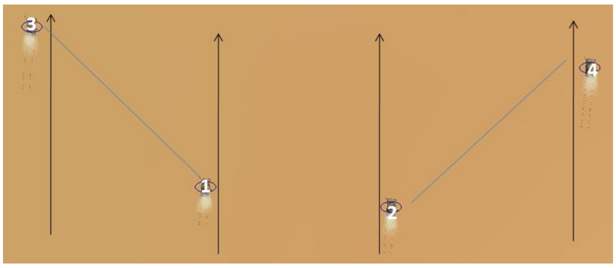
Two Columns



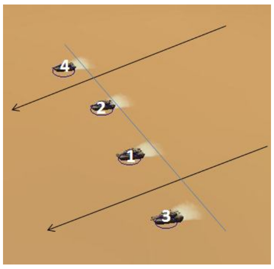
Staggered Column



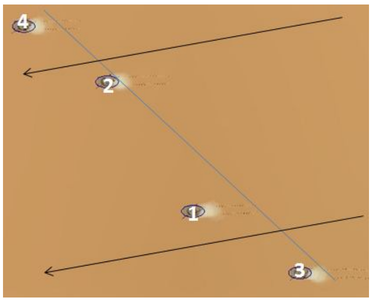
Vee



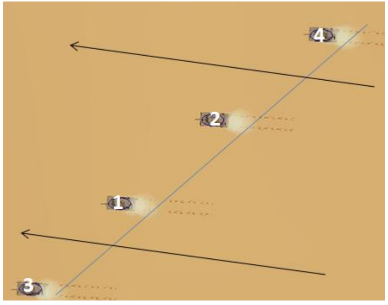
Line



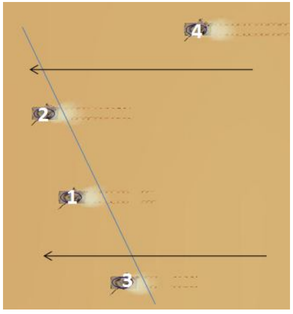
Echelon Left



Echelon Right



Wedge Left



Wedge Right