Student name: ______________________________ Student number:

Faculteit der Exacte Wetenschappen
Exam Design of Multi-Agent Systems
Vrije Universiteit Amsterdam 22 October 2009

Exercise	1	2	3	4	**bonus**
points	25	30	20	15	10

Norm:
The tentamination mark **T** equals (the sum of the points scored for the exercises plus 10 bonus points) divided by 10.
The endmark **E** for the course Design of Multi-Agent Systems is calculated as follows: **E = (T + H + P) / 3**
Where : T = tentamination mark H = mark for the home work exercises P = mark for the small practicum

You find:

4 exercises
5 appendices (1A, 1B, 2, 3, 4)
You can answer in Dutch as well as English

Student name: ______________________________ Student number:

Exercise 1 (25 points):

Relevant Appendices: Appendix 1A and Appendix 1B.

This exercise consists of two parts. Motivate your answers.

Exercise 1a (15 points)

In chapter 1 of the syllabus a number of primitive agent concepts have been introduced (see Appendix 1B of the answer sheets). In Appendix 1A, you can find some information on an intelligent system to improve medicine usage.
Analyse this information according to the primitive agent concepts and fill out Appendix 1B (3 answer sheets) indicating which agent concepts are relevant for the medicine usage system. Remember to motivate your answers clearly.

Exercise 1b (10 points)

Would you call this system an agent? Motivate your answer.

Exercise 2 (30 points)

Relevant Appendix: Appendix 2.

This exercise concerns the building of an intelligent F16 opponent, and is explained in Appendix 2.

Exercise 2a (15 points)

Give a graphical representation of the information types that you would use in the intelligent F16 opponent agent.

Exercise 2b (15 points)

Use the information types you have identified above to specify rules of the knowledge base needed for the intelligent F16 opponent agent.

Student name: ______________________________ Student number:

Exercise 3 (20 points):

Relevant Appendix: Appendix 3.

This question is about information states and reasoning. Study the partial specification of Appendix 3. This is the public information state S of component wall_e:

```
S = [   observation_result(at_position(self, p0), pos),
        observation_result(at_position(self, p1), neg),
        observation_result(at_position(self, p2), neg),
        observation_result(at_position(plant, p1), pos),
        observation_result(at_position(engine_oil, p2), pos) ]
```

Exercise 3a (8 points)

Provide an information state S', that refines S and is also closed and consistent with respect to the knowledge base of component wall_e.

Exercise 3b (4 points)

Motivate that S' is a refinement of S.

Exercise 3c (4 points)

Motivate that S' is closed with respect to the knowledge base of component wall_e.

Exercise 3d (4 points)

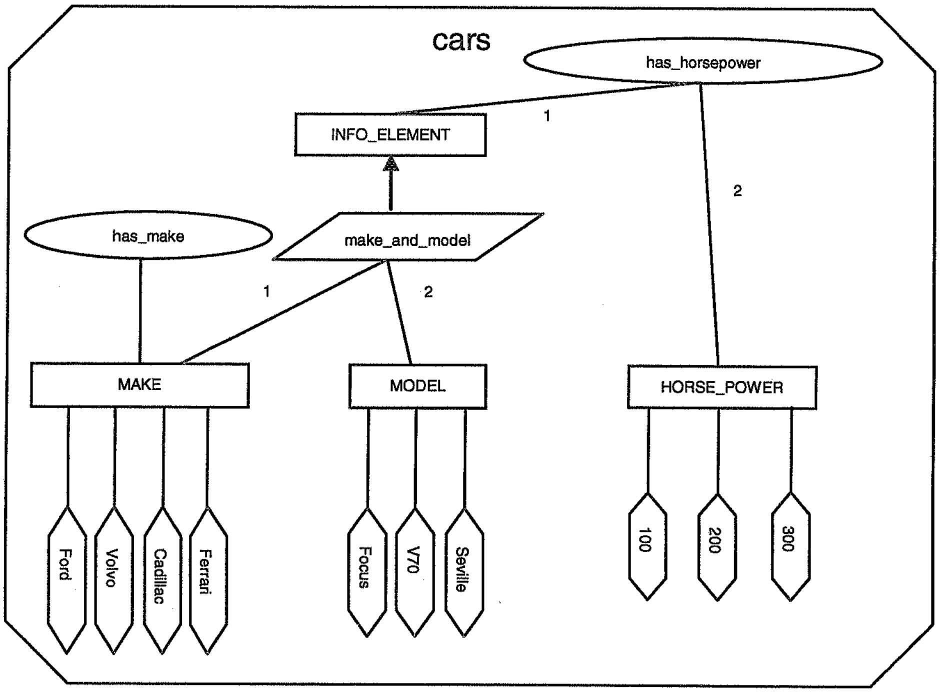
Motivate that S' is consistent with respect to the knowledge base of component wall_e.

Student name: ______________________________ Student number:

Exercise 4 (15 points):

Relevant Appendix: Appendix 4.

Consider the information type cars as shown in the Figure below. In Appendix 4 you can find a table consisting of a number of strings. Which of these strings are terms considering the information type cars? Which are atoms? And which are ground atoms? Which of the terms are well formed? Which of the atoms are well formed? Fill in your answer in the table in Appendix 4.



Student name: ______________________________ Student number:

Appendix 1A: Intelligent Medicine Box System

Assisting humans in timely usage of their medicine can be crucial for their health. When looking at HIV medicine, this medicine is very sensitive to irregular intake and needs to be taken at strict times in order for the patient to stay healthy. To assist humans in taking their medicine, the company called SimPill has decided to create an intelligent system that supports humans in taking their medicine on a regular basis. The system continuously monitors the medicine box of the patient to see whether medicine is taken out of the box. In case the system notices that no medicine has been taken at the appropriate time, the system can warn the user that he/she needs to take medicine. Furthermore, the system can also give warnings in case the patient tries to take medicine too early. If the system considers the patient insufficiently structured in its medicine usage (which is derived by looking at the history of medicine intake), the system contacts the doctor, and informs the doctor on the current medicine usage pattern of the patient. The doctor can give the system input on what strategy the system should follow to improve the patient's intake behavior.

Student name: ______________________________ Student number:

Appendix 1B: Answersheet (1 out of 3)

I. External primitive concepts	
A. Interaction with the world	
passive observations	
active observations	
performing actions	
B. Communication with other agents	
incoming	
outgoing	

Student name: ______________________________ Student number:

Appendix 1B Answersheet (2 out of 3)

II. Internal primitive concepts	
A. World Model	
B. Agent Models	
C. Self Model	
D. History	
E. Goals	
F. Plans	
G. Group Concepts	
Joint goals	
Joint plans	
Commitments	
Negotiation strategies	

Student name: ______________________________ Student number:

Appendix 1B Answersheet (3 out of 3)

III. Types of behaviour	
Autonomy	
Responsiveness	
Pro-activeness	
Social behaviour	
Own adaptation and learning	

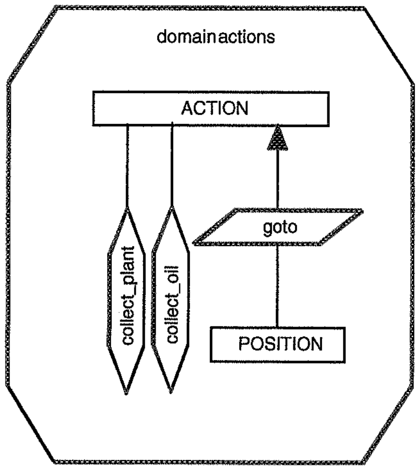
Student name: ______________________________ Student number:

Appendix 2 Intelligent F16 Opponent

The fighter pilots of any air force need to be well trained. However, the cost of letting such a pilot train in the air are very high. As a result, more and more often simulation environments are being used. Ideally, these simulation environments would contain enemies that act as one would expect of real world enemies: in an intelligent manner. Therefore, the air force decides to develop an intelligent agent that can act as an opponent within fighter pilot simulations, and can control a MIG, a well known Russian aircraft often used as an opponent in training simulations.

In order to control the plane, the agent can receive a number of observations based upon which it responds: it can see whether the F16 pilot has a radar lock on the MIG or not, and it can also observe the opposite, namely whether it has a radar lock on the F16 pilot or not. It can also be observed if the F16 pilot has fired a missile or not. Finally, the distance can be observed: is the F16 plane relatively close or relatively far away. Based upon this information, the agent can derive appropriate actions to undertake. In this case several actions are possible, namely: (1) fire a missile; (2) perform a fancy maneuver; (3) fire flares as a decoy, and (4) jump out of the plane.

Essentially the rules the agent uses are relatively simple. In case the other F16 pilot has a radar lock, has fired a missile, and is relatively close by, the agent jumps out of the plane. If the radar lock is observed, and also the fact that the missile has been fired, but the F16 pilot is relatively far away, the agent fires flares and performs a fancy maneuver. In case a radar lock is present, but no missile has been fired, the agent performs a fancy maneuver, and finally, if the agent itself has a radar lock upon the F16 pilot it always fires a missile.



In addition to these domain specific types, the generic types are shown in a textual format below.

information type truth_indication
  sorts SIGN ;
  objects pos,
    neg : SIGN;
end information type

information type observation_results
  sorts INFO_ELEMENT, SIGN ;
  relations observation_result: INFO_ELEMENT * SIGN;
end information type

information type observation_result_info
  information types truth_indication,
    observation_results,
    information_element_info;
end information type

information type actions_to_be_performed
  sorts ACTION ;
  relations to_be_performed: ACTION;
end information type

information type action_info
  information types actions_to_be_performed,
    domain_actions;
end information type

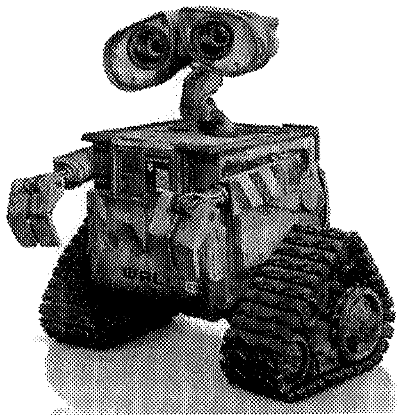
Student name: ______________________________ Student number:

Appendix 3 Wall-E

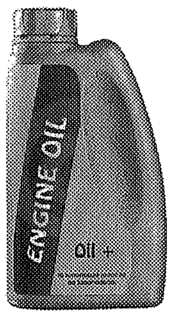
3.1 Problem Description

In this case the problem concerns Wall-E, a robot sent out by humans to clean up a planet, but also to discover life on the planet. Wall-E is in this case seen as an agent for which a simple case of behaviour is represented below.

p2



p0



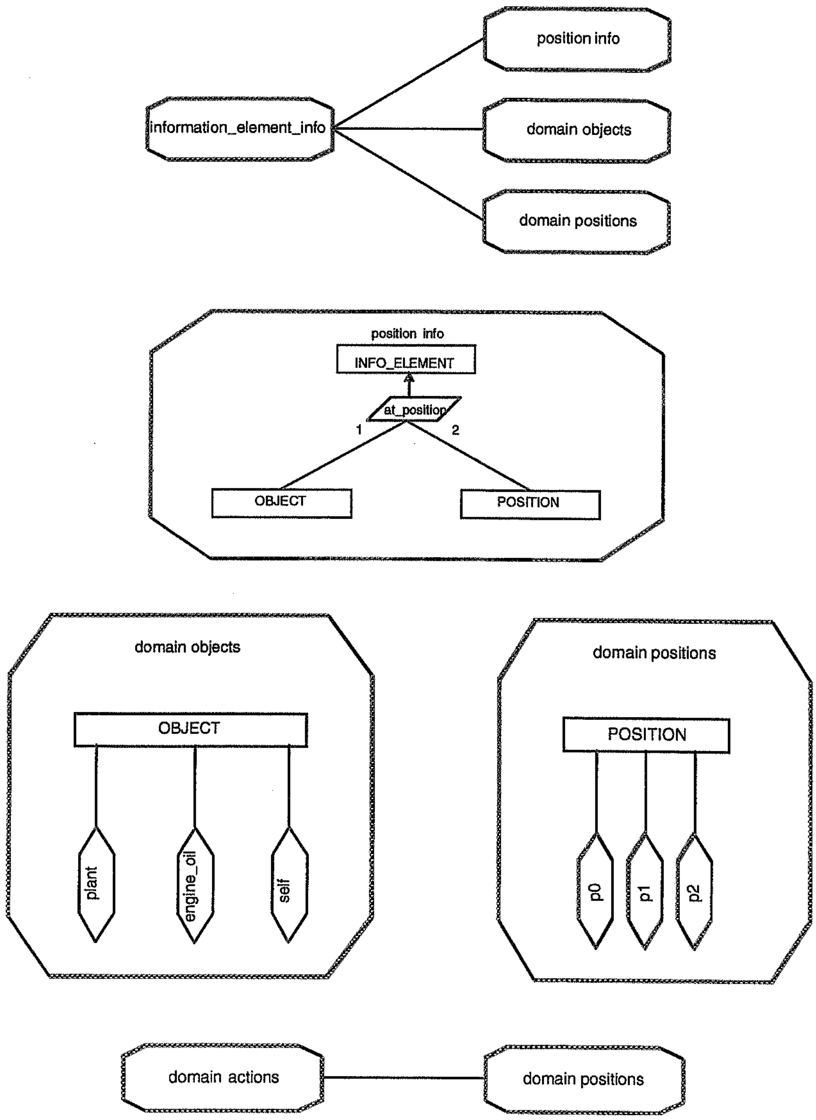
p1

Wall-E should always prioritize finding life above its own life. His own life is dependent on having sufficient engine oil. In the simple case we address here there are three positions in the world, namely p0, p1, and p2. Furthermore, three objects can be present, namely a plant, Wall-E itself, and engine oil. Each of these three objects can be placed at any of the positions. In the figure shown above, Wall-E is located at p0, the engine oil at p1, and a plant at p2. Note that this is merely an example of a configuration.

A specification of Wall-E expression the knowledge needed for the simple three locations world is expressed below.

3.2 Information types

The information types being used within the whole system are:



Student name: ________________________________ Student number:

3.3 Fragment of specification of the component

The component is primitive and is described shortly below.

The component wall_e

The interfaces are defined by:
  input interface: the information type observation_result_info;
  output interface: the information type action_info;

The contents of the knowledge base:

```
if      observation_result(at_position(plant, P:POSITION), pos)
  and   observation_result(at_position(self, P:POSITION), neg)
then    to_be_performed(goto(P:POSITION)) ;


if      observation_result(at_position(plant, p0), neg)
  and   observation_result(at_position(plant, p1), neg)
  and   observation_result(at_position(plant, p2), neg)
  and   observation_result(at_position(engine_oil, P:POSITION), pos)
  and   observation_result(at_position(self, P:POSITION), neg)
then    to_be_performed(goto(P:POSITION)) ;

if      observation_result(at_position(self, P:POSITION), pos)
  and   observation_result(at_position(plant, P:POSITION), pos)
then    to_be_performed(collect_plant) ;

if      observation_result(at_position(self, P:POSITION), pos)
  and   observation_result(at_position(engine_oil, P:POSITION), pos)
then    to_be_performed(collect_oil) ;
```

Student name: ______________________________ Student number:

Appendix 4: Answersheet for Exercise 4.

	term	atom	ground atom	well-formed
has_make				
make_and_model(Focus, H:HORSE_POWER)				
400				
has_horsepower(make_and_model(Ford, Focus), 100)				
H:HORSE_POWER				
has_make(make_and_model(Cadillac, Seville))				
has_make(M:MAKE)				
has_horsepower(Ford, 300)				
has_make(has_horsepower(I:INFO_ELEMENT), 300)				