Faculteit der Exacte Wetenschappen	Tentamen Ontwerp van Multi-agentsystemen
Vrije Universiteit Amsterdam	21 januari 2000

Opgave	**1**	**2**	**3**	**4**	**5**	**bonus**
Punten	20	15	20	20	15	10

Normering:
Het **tentamencijfer** T is gelijk aan (het totaal behaalde punten voor de tentamenopgaven plus 10 punten bonus) gedeeld door 10.

Het **eindcijfer** voor het hoorcollege Ontwerp van Multi-agentsystemen wordt als volgt berekend:

Eindcijfer = (T + H + P) / 3,

waarbij
T = tentamencijfer,
H = cijfer huiswerkopgaven,
P = cijfer voor 1 punts practisch werk.

Verzoek:
U wordt vriendelijk verzocht om voor Uw antwoorden gebruik te maken van de speciale **antwoordvellen**.

U treft aan:
5 opgaven
appendices
antwoordvellen

Opgave 1 (20 punten)

Read Appendix 1 and answer the following questions. The generic agent model of Chapter 6 consists of 7 components: own process control, world interaction management, agent interaction management, maintenance of world information, maintenance of agent information, cooperation management, and agent specific task.

Which of these components do you need and which do you not need to model Columbus? Motivate your answer and make explicit references to the text of Appendix 1.

Opgave 2 (15 punten)

The knowledge Columbus needs to determine his goals needs to be specified in one of the generic components. Let us call this component comp_c so as not to influence your answer to question 1. In Appendix 2 a partial specification is given of comp_c. Complete the knowledge base of the sub-component goal_determination of comp_c (10 points). Make sure that for every possible character of Columbus the knowledge base can be used to determine when to explore, and when to go back to resupply Columbus. The cautious explorer reckons that with the tanks half full he should be able to make it back to the last place where he resupplied. Explain every rule you add to the knowledge base in normal words (5 points).

Opgave 3 (20 punten)

Assume that at a certain moment in time the object level **output** information state of component goal_determination is S.

S = [current_goal(back_to_resupply), next_goal(explore), not current_goal(explore)].

Consider the three meta-information states M1, M2, and M3 of Appendix 3.

a) For each of the following pairs of information states denote whether or not they are level coherent: pair (S, M1), pair (S, M2), pair (S, M3).

b) For every pair that is not coherent, explain why that is so.

Opgave 4 (20 punten)

Suppose that before execution of the links reset_old_goals, and set_new_goals the input and output information states of component goal_determination are as given in table 1. First link reset_old_goals is executed, after that also link set_new_goals is executed. The specification of these links can be found in Appendix 2. Focus only on the atoms that can be represented using the information type goal_info.

a) Give the **input meta-**level information state of component goal_determination after execution of the link **reset_old_goals**.

b) Give the **input object** level information state of component goal_determination after execution of the link **reset_old_goals** (and after execution of the downward reflection).

c) Give the **input meta**-level information state of component goal_determination after execution of the link **set_new_goals**. Pay attention to the changes you made for 4a and 4b, remember that this link is executed after these changes took effect.

d) Give the **input object** level information state of component goal_determination after execution of the link **set_new_goals** (and after execution of the downward reflection). Pay attention to the changes you made for 4a, 4b, and 4c remember that this link is executed after these changes took effect.

object level input:	meta-level input:	object level output:
current_goal(explore): false	assumption(current_goal(explore), pos): unknown	next_goal(explore): true
current_goal(back_to_resupply): true	assumption(current_goal(explore), neg): true	current_goal(explore): false
	assumption(current_goal(back_to_resupply), pos): true	current_goal(back_to_resupply): true
	assumption(current_goal(back_to_resupply), neg): unknown	
	assumption(next_goal(explore), pos): unknown	
	assumption(next_goal(explore), neg): unknown	
	assumption(next_goal(back_to_resupply), pos): unknown	
	assumption(next_goal(back_to_resupply), neg): unknown	

Table 1 information states with respect to information type goal_info

Opgave 5 (15 punten)

Consider the domain of laying a table for four persons for dinner, see Figure 2. The laying of the table can be seen as a process that is to be controlled. In the design of a process controller for this domain, the design of the process control model described in Chapter 3 is to be reused. The domain specific information types have to be adapted.

a) Design the information type domain info for this domain. You can do this in one information type, but you are also allowed to make more levels of abstraction.

b) Design the information type domain actions for this domain. You can do this in one information type, but you are also allowed to make more levels of abstraction.

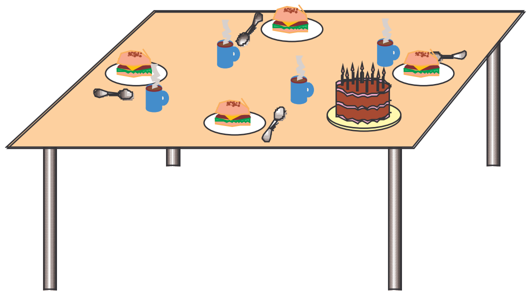

Figure 2

Appendix 1   Columbus: exploring space!

Only some 600 years ago the people that sailed the seven seas were often conftronted with the unknown. Large areas of Earth were still unmapped and maps that were available were often incomplete and not very accurate. Explorers were often considered heroes (only if they returned to tell their stories) or fools (those that did not come back). Experience learned that the major problem (next to plain disasters like shipwrecks) is that if you do some serious exploring then you tend to run out of supplies. With the start of the new millenium, your help is requested in designing the next generation of explorers.

In this new millenium a serious start will be made to explore space. Instead of sending people onto these very dangerous travels, the unmanned spacecraft will play a central role in this endeavour. The idea is to send out a mass of small spacecrafts that have the ability to observe their environment and are capable of resupplying themselves if they are able to find supplies in space. The first prototype called Columbus is almost ready; Columbus has everything it needs but for a reasoning component. That component will be responsible for Columbus' behaviour, it needs to be modelled according to the following requirements.

Columbus needs to be an agent (according to the weak notion of agenthood), for it will have to be totally self dependent (Earth will soon be too far away to be able to influence its behaviour). The main reason to send out Columbus is the need for space maps that contain lots of information about the planets, stars, asteroids, blackholes, and others the Columbus encounters during its travels. However, these travels take energy for the sensors and fuel for the engines that Columbus needs to steer itself and to get away from gravitation fields. Of course Columbus only gets a limited amount of supplies when it sets of on its voyage, further on more information on this point can be found.

The voyage Columbus is making is only useful to Earth if, every now and then, Columbus communicates to Earth a report containing the star map it has made so far. Earth is unable to reply to these messages. The map contains all information that Columbus picked up on its scanners and sensors and Columbus adds the space coordinates for that information (time stamp, current location, speed, direction, etc.). Columbus also has special sensors with which it can monitor its own state of supplies: **almost empty**, **half full**, **almost full**, **full**.

In the exploration of space the major problem that Columbus has to face is that it could run out of supplies. Hopefully, Columbus will discover new places to resupply itself during its explorations. The ship's sensors are fully capable of recognizing different materials that can be used as supplies. Of course Columbus needs to remember where these supplies are, and since it has to make a space map anyway, the information of where supplies can be found has to be added to the space map. While exploring space the supplies will slowly run out and there will be a point where Columbus has to decide whether to turn back to one of the places it knows it can resupply or whether to press on in the hope that a new supply will be discovered before the current supply runs out. So the question always is: does it continue to travel into the unknown (**explore**), or does it go back to a place where it knows it can resupply (**back to resupply**)? This decision process is not only based on Columbus's beliefs about its environment and its own status, but also on its own character, since Columbus can eiter be **reckless**, **cautious**, or a **coward**.

Important is that Columbus always has a goal!

You can assume, that initially, Columbus, has the goal to explore and not to go back to resupply, and it has one of the above mentioned characters. You can further assume that Columbus will resupply at every opportunity unless its tanks are full.

Appendix 2 A part of the specification for Columbus

2.1 top-level

```
information type agent_characteristics
   sorts           CHARACTERISTIC;
   relations    own_characteristic:   CHARACTERISTIC;
end information type

information type domain_agent_characteristics
   sorts           CHARACTERISTIC;
   objects         reckless, cautious, coward:  CHARACTERISTIC;
end information type

information type agent_characteristics_info
   information types agent_characteristics, domain_agent_characteristics;
end information type

information type goal_info
   sorts           GOAL
   objects         explore, back_to_resupply:       GOAL;
   relations    current_goal, next_goal:      GOAL;
end information type
```

2.2 component comp_c

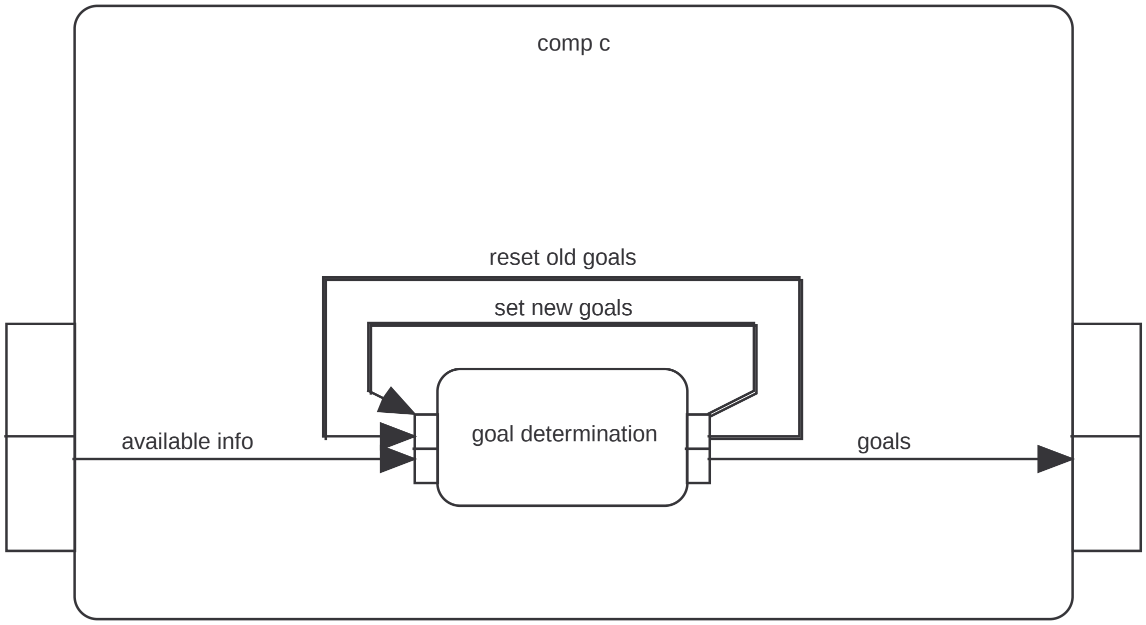

Figure 3 A partial view on comp_c

```
task control knowledge base of comp_c
    if      start
    then    next_component_state(goal_determination, awake)
      and       next_link_state(available_info, awake);

    if      evaluation(goal_determination, next_goal_determined, any, succeeded)
      and       not component_state(goal_determination, busy)
      and       previous_component_state(goal_determination, busy)
    then    next_link_sequence_state([ goals, reset_old_goals, set_new_goals], uptodate);
end task control kb

private link reset_old_goals: epistemic - assumption
    domain goal_determination
        level GD_meta_level
        information types epistemic_goal_info
        /* the standard epistemic information type with respect to information type goal_info */
    co-domain goal_determination
        level GD_meta_level
        information types assumption_goal_info
        /* the standard assumption information type with respect to information type goal_info */
    sort links identity
    term links identity
    atom links     (true(current_goal(G: GOAL)), assumption(current_goal(G: GOAL), neg)): <<true, true>>;
end link

private link set_new_goals: epistemic - assumption
    domain goal_determination
        level GD_meta_level
        information types epistemic_goal_info
        /* the standard epistemic information type with respect to information type goal_info */
    co-domain goal_determination
        level GD_meta_level
        information types assumption_goal_info
        /* the standard assumption information type with respect to information type goal_info */
sort links identity
term links identity
atom links     (true(next_goal(G: GOAL)), assumption(current_goal(G: GOAL), pos)): <<true, true>>;
end link
```

```
mediating link available_info: object - object
   domain comp_c
      level comp_c_object_level
      information types agent_characteristics_info, belief_info
   co-domain goal_determination
      level GD_object_level
      information types agent_characteristics_info, belief_info
sort links identity
term links identity
atom links identity
end link

mediating link goals: object - object
   domain goal_determination
      level GD_object_level
      information types goal_info
   co-domain comp_c
      level comp_c_object_level
      information types goal_info
sort links identity
term links identity
atom links
(next_goal(G: GOAL), current_goal(G: GOAL)): <<true,true>, <unknown,unknown>, <false, false>>;
end link
```

component goal_determination

```
input information types      agent_characteristics_info, belief_info, goal_info;
output information types     goal_info;
task information             evaluation criterion:  next_goal_determined
initial task information:    target(next_goal_determined, next_goal(G: GOAL), confirm);
knowledge base goal_determination_kb
   information types agent_characteristics_info, belief_info, goal_info;
   contents
   if    own_characteristic(reckless)
     and    belief(suppy_status(almost_empty), neg)
   then next_goal(explore);

   if    own_characteristic(cautious)
   /* you can use more rules if you want or need to */
   .....
```

```
    .....
    then  next_goal(explore);

    if      own_characteristic(coward)
      and        belief(suppy_status(almost_full), pos)
    then  next_goal(explore);

    if      own_characteristic(coward)
      and        belief(suppy_status(full), pos)
    then  next_goal(explore);

    if      own_characteristic(reckless)
      and        belief(suppy_status(almost_empty), pos)
    then  next_goal(back_to_resupply);

    if      own_characteristic(cautious)
    /* you can use more rules if you want or need to */
    .....
    .....
    then  next_goal(back_to_resupply);

    if      own_characteristic(coward)
      and        belief(suppy_status(half_full), pos)
    then  next_goal(back_to_resupply);

    if      own_characteristic(coward)
      and        belief(suppy_status(almost_empty), pos)
    then  next_goal(back_to_resupply);
end knowledge base

contents goal_determination_kb;
end component
```

Appendix 3

The three information states for question 3.

M1 = [

known(current_goal(explore)),	not true(current_goal(explore)),	false(current_goal(explore)),
known(current_goal(back_to_resupply)),	true(current_goal(back_to_resupply)),	not false(current_goal(back_to_resupply)),
known(next_goal(explore)),	true(next_goal(explore)),	not false(next_goal(explore)),
not known(next_goal(back_to_resupply)),	not true(next_goal(back_to_resupply)),	not false(next_goal(back_to_resupply))

]

M2 = [

known(current_goal(explore)),	not true(current_goal(explore)),	false(current_goal(explore)),
known(current_goal(back_to_resupply)),	true(current_goal(back_to_resupply)),	not false(current_goal(back_to_resupply)),
known(next_goal(explore)),	true(next_goal(explore)),	not false(next_goal(explore)),
known(next_goal(back_to_resupply)),	not true(next_goal(back_to_resupply)),	false(next_goal(back_to_resupply))

]

M3 = [

not known(current_goal(explore)),	not true(current_goal(explore)),	false(current_goal(explore)),
not known(current_goal(back_to_resupply)),	true(current_goal(back_to_resupply)),	not false(current_goal(back_to_resupply)),
not known(next_goal(explore)),	true(next_goal(explore)),	not false(next_goal(explore)),
known(next_goal(back_to_resupply))		

]

Antwoordvellen   Student: Lichting:

Antwoorden op de vragen 4a en 4b

meta level input after execution of reset_old_goals before downward reflection		**object level input** after execution of reset_old_goals and after downward reflection	
atom	truth-value	atom	truth-value
assumption(current_goal(explore), pos)			
assumption(current_goal(explore), neg)			
assumption(current_goal(back_to_resupply), pos)			
assumption(current_goal(back_to_resupply), neg)			
assumption(next_goal(explore), pos)			
assumption(next_goal(explore), neg)			
assumption(next_goal(back_to_resupply), pos)			
assumption(next_goal(back_to_resupply), neg)			

Tabel 4

Answersheets Student: Year:

Answers to questions 4c and 4d
(think of what you entered in Table 4!)

meta level input after execution of set_new_goals and before downward reflection		**object level input** after execution of set_new_goals and after downward reflection	
atom	truth-value	atom	truth-value
assumption(current_goal(explore), pos)			
assumption(current_goal(explore), neg)			
assumption(current_goal(back_to_resupply), pos)			
assumption(current_goal(back_to_resupply), neg)			
assumption(next_goal(explore), pos)			
assumption(next_goal(explore), neg)			
assumption(next_goal(back_to_resupply), pos)			
assumption(next_goal(back_to_resupply), neg)			

Table 5