The University of Melbourne

Semester 1 Assessment 2009

Department of Computer Science and Software Engineering
433-682 Software Agents

Reading Time	15 minutes.
Writing Time	Two hours.

This paper has 4 pages including this cover page.

Identical Examination Papers:	433-482 Software Agents.
Common Content Papers:	None.

Authorised Materials:
The exam is open book.
Text books and printed notes are allowed into the examination.
No calculators allowed.

Instructions to Invigilators:
Students must write their answers on the script books provided.

Instructions to Students:
This paper counts for 30% of your final grade. Your mark will be calculated on your best answers to the questions in this exam paper. You should attempt to answer *at least four* (4) questions from the available questions. Clearly mark which question you are answering in your script book. If you choose answer *more than four* questions, your mark will be calculated on your **best four answers** to questions you have attempted (we will mark all your questions). All questions are worth equal marks (7.5 marks each) totalling up to 30 marks.

Paper to be held by Baillieu Library: Yes.

Question 1 (7.5 Marks)

This question concerns the situation calculus. For the little MIndiGolog program listed below, list all the legal executions of this program using the situation calculus. For the purpose of listing all legal executions you can assume that *all* actions turn out to be possible with respect to the domain. You can assume the initial state is S_0.

```
proc(getOnFlight,
        (go(terminal1) || go(terminal2))
      : (buy(magazine) || buy(newspaper))
      : if(GateNo >= 90)
          then (go(gate(GateNo)): buy(coffee))
          else (buy(coffee): go(gate(GateNo)))
      : boardPlane
)
```

Question 2 (7.5 Marks)

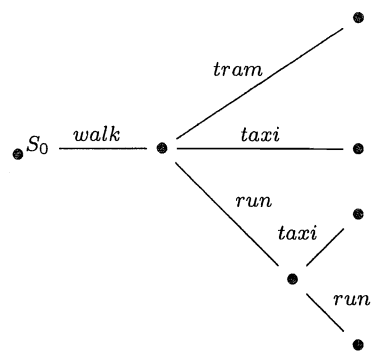

Figure 1: *Getting to the talk on time*: this Figure relates to Question 1. The Figure shows a possible worlds structure (states are denoted by black dots) including initial state S_0. The actions (denoted by edges) include *walk*, *tram*, *taxi* and *run*.

For the possible worlds structure shown in Figure 1, write down the corresponding Kripke model that incorporates the primitive proposition $\phi =$*"I will get to the talk on time"*.

Note that in this case, that the model will only concern *one* agent. Be sure to identify and define as much of the possible worlds structure as you can in terms of the Kripke model - including states (or possible worlds), binary relations (or equivalence relations) and the interpretation.

Question 3 (7.5 Marks)

This question concerns the muddy children puzzle, as covered in lectures. Draw a series of Kripke structures, where each structure corresponds to the instant immediately after the father asks *"Do you know if you have a muddy forehead?"*—one for each step—up until *all* the children know whether they have a muddy forehead.

Assume there are three children. Be sure to define what the vertices mean (what the labels mean) and what the edges in the model correspond to in the structures that you draw.

Question 4 (7.5 Marks)

Table 6.3
The viewpoint framework

Viewpoint models	Viewpoint aspect		
Abstraction layer	Interaction	Information	Behaviour
Conceptual domain modelling	Role models and organization models	Domain models	Goal models and motivational scenarios
Platform-independent computational design	Agent models and acquaintance models, interaction models	Knowledge models	Scenarios and behavior models
Platform-specific design and implementation	Agent interface and interaction specifications	Data models and service models	Agent behavior specifications

Figure 2: The viewpoint framework (Table 6.3, Sterling and Taveter, 2009) specifies viewpoint models for each aspect at three different abstraction layers.

Use examples from the **restaurant** domain to very briefly (using one sentence for each model) state a concrete instance (example) of each of the viewpoint models in Figure 2. Note that in all there are nine models that you will need to give the little **restaurant** examples for, including the

- Conceptual domain modelling:
  - Role models and organization models (interaction),
  - Domain models (information), and
  - Goal models and motivational scenarios (behavior).
- Platform-independent computational design:
  - Agent models and acquaintance models, interaction models (interaction),
  - Knowledge models (information), and
  - Scenarios and behavior models (behavior).
- Platform-specific design and implementation:
  - Agent interface and interaction specifications (interaction),
  - Data models and service models (information), and
  - Agent behavior specifications (behaviour).

Question 5 (7.5 Marks)

State, using either a table and/or bullet points the differences between the programming languages—Prolog, Golog, IndiGolog and MIndiGolog.

Be as specific as you can, stating using one or two sentences to how each language handles determinism, non-determinism, concurrency and the mechanism by which each language executes.

If you choose, you can utilise little examples from the **restaurant** domain covered in lectures and/or in your project(s). You are free to use examples covered in your project(s)—whether from the restaurant domain or not.

Question 6 (7.5 Marks)

This question concerns achieving concurrency and parallelism in the MIndiGolog programming language and requires you to explain, briefly, what is involved. State using either bullet points or a list of short sentences the steps involved in changing a MIndiGolog program that involves *only sequential execution* into a program that involves *parallel concurrent execution.*

You should be sure to **list** the kinds of things you need to take into consideration when programming for concurrency and, importantly, *what you need to watch out for.* For example, typically you have to make sure there are enough resources specified in the domain to enable concurrent execution in the first place (in the **restaurant** domain this might require specification of more than one waiter).

End of the Exam