The University of Melbourne

Semester 1 Assessment 2008

Department of Computer Science and Software Engineering
433-482 Software Agents

Reading Time 15 minutes.

Writing Time Two hours.

This paper has 4 pages including this cover page.

Identical Examination Papers: 433-682 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 five* (5) questions from the available questions. Clearly mark which question you are answering in your script book. If you choose answer *more than five* questions, your mark will be calculated on your *best five answers* to the questions you have attempted (we will mark all your questions).

One restriction is that you should answer no more than *two* of questions 1–3.

All questions are worth equal marks (6 marks each) totalling up to 30 marks.

Paper to be held by Baillieu Library: Yes.

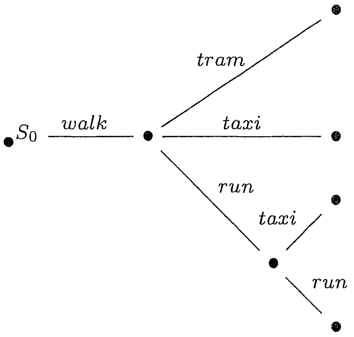

Figure 1: *Getting to the talk on time*: this Figure relates to Questions 1 and 2. The Figure shows a possible worlds structure (states are denoted by black dots) including initial state S_0. The actions (denoted by edges) include *walk*, *tram*, *taxi* and *run*.

Answer no more than TWO of questions 1–3

Question 1 (6 Marks)

This question concerns the situation calculus. For the possible worlds structure shown in Figure 1, list the situations that occur—using the situation calculus—for each state in the Figure.

You can assume the initial state is S_0 as shown in Figure 1.

Question 2 (6 Marks)

For the possible worlds structure shown in Figure 1, write down the corresponding Kripke model that incorporates the primitive proposition ϕ = *"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 (6 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 **(6 Marks)**

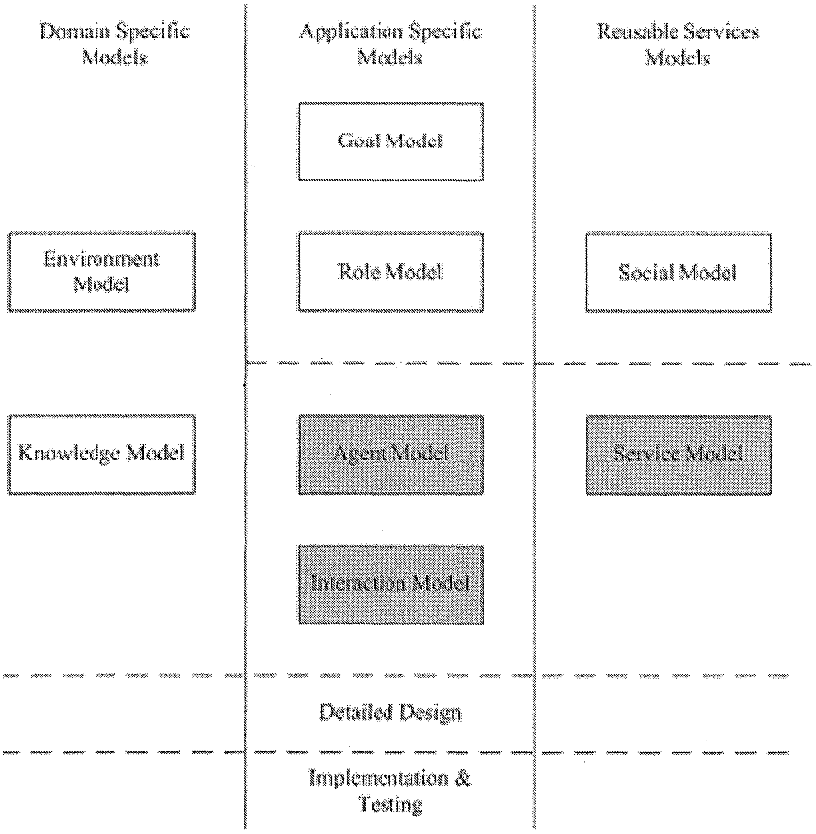

Figure 2: The ROADMAP Model (Juan, Sterling and Pearce 2002) specifies domain specific models (left), application specific models (middle) and reusable models (right).

Use examples from the cooking domain to very briefly (using one sentence for each model) state a concrete instance (example) of each of the domain-specific, application-specific and reusable services models of the ROADMAP model in Figure 2. Note that in all there are eight models that you will need to give the little cooking examples for, including the

- Domain specific models:
  - environment model,
  - knowledge model,
- Application specific models:
  - goal model,
  - role model,
  - agent model,
  - interaction model,
- Reussable services models:
  - social model, and
  - service model.

Question 5 (6 Marks)

State, using either a table and/or bullet points the differences between *two* languages—chosen from JASON, JADE, MIndiGolog and ConGolog programming languages. Be as specific as you can, stating using one or two sentences to how each language handles concurrency and the mechanism by which each language executes.

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

Question 6 (6 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 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 cooking this might require specification of more than one knife).