TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS — 2008

MID-YEAR

COMP 426

Formal Software Development

Time Allowed: 3 Hours

Instructions: Candidates should attempt **all SIX** questions.

This exam will be marked out of 100.

Foreign language translation dictionaries are allowed.

A summary of Z mathematical notation is provided at the end of this paper.

Question 1. Formal specification in Z [16 marks]

Below is an initial specification for a computerised Dating Agency which keeps a list of people seeking partners, and attempts to find potential partners on the basis of profiles which people provide. The state records the people on the "list" and their profiles; initially the list is empty. The system provides operations to add a new person, remove a person, and propose a match between two people based on a "compatibility" function f.

$[Person, Profile]$

$$f : Profile \times Profile \longrightarrow \mathbb{Z}$$

DatingAgency

$$list : \mathbb{P}\, Person$$
$$profile : Person \pfun Profile$$
$$\mathrm{dom}\, profile = list$$

InitDatingAgency

$$DatingAgency'$$
$$list' = \emptyset$$

RemPerson

$$\Delta DatingAgency$$
$$p? : Person$$
$$p? \in list$$
$$profile' = \{p?\} \rhd profile$$

AddPerson

$$\Delta DatingAgency$$
$$p? : Person$$
$$a? : Profile$$
$$p? \notin list$$
$$profile' = profile \oplus \{p? \mapsto a?\}$$

Match

$$\Xi DatingAgency$$
$$p!, q! : Person$$
$$\{p!, q!\} \subseteq list$$
$$f(profile(p!), profile(q!)) > 0$$

(a) [2 marks] Explain why *profile* is defined as a partial function. What would be the effect on the specification if *profile* was defined to be a total function?

(b) [4 marks] The above version of *Match* proposes a potential match by selecting an arbitrary pair of people from the list with a positive compatibility rating. Define a new version of *Match* which attempts to find the two people on the list with the best possible match. Explain your answer.

(c) [10 marks] Extend the system so that when a match is proposed, both people are recorded as being "matched" and thus not considered in future applications of the *Match* operation, and add a new *Reject* operation allowing either of the parties in a match to reject the match. Once a proposed match has been rejected, both parties should again be considered by *Match*, but *Match* should not propose a match that has previously been proposed and rejected. Show, and explain, any changes required to the state or to other operations in the system.

Question 2. Data refinement in Z [20 marks]

The following is an implementation of the initial Dating Agency system in Question 1 (note that iseq defines a set of injective sequences, i.e. sequences of unique elements):

*DatingAgency*1

$$names : \mathrm{iseq}\, Person$$
$$data : \mathrm{seq}\, Profile$$
$$\#data = \#names$$

*InitDatingAgency*1

$$DatingAgency1'$$
$$names' = \varnothing$$

*AddPerson*1

$$\Delta DatingAgency1$$
$$p? : Person$$
$$a? : Profile$$
$$p? \notin \mathrm{ran}\, names$$
$$names' = names \frown \langle p? \rangle$$
$$data' = data \frown \langle a? \rangle$$

*RemPerson*1

$$\Delta DatingAgency1$$
$$p? : Person$$
$$\exists n1, n2 : \mathrm{iseq}\, Person ; d1, d2 : \mathrm{seq}\, Profile \bullet$$
$$\#n1 = \#d1 \land$$
$$names = n1 \frown \langle p? \rangle \frown n2 \land$$
$$data = d1 \frown d2 \land$$
$$names' = n1 \frown n2 \land$$
$$data' = d1 \frown tail(d2)$$

*Match*1

$$\Xi DatingAgency1$$
$$p!, q! : Person$$
$$\exists i, j : \mathrm{dom}\, names \bullet$$
$$f(data(i), data(j)) > 0 \land$$
$$p! = names(i) \land q! = names(j)$$

(a) [8 marks] Give an abstraction relation showing the relationship between *DatingAgency* and *DatingAgency*1, and explain briefly how it is used to prove that *DatingAgency*1 is a data refinement of *DatingAgency*.

(b) [6 marks] Show how you would extend the state of *DatingAgency*1 to accomodate the change described in part **(c)** of Question **1**, and modify your abstraction relation from part **(a)** to reflect this change.

(c) [6 marks] Define a concrete version of *Reject* that operates on your extended state for *DatingAgency*1, and give a brief justification that it is a correct data refinement of your version of *Reject* from Question **1**.

Question 3. Object-Z and CSP [24 marks]

Consider a ticket machine for a public transport system that allows either single or return tickets to be dispensed to a number of destinations. The user can select the type of ticket (single or return) and a destination. If enough coins have been inserted, the machine returns a ticket.

(a) [8 marks]

Specify the ticket machine using an Object-Z class. The class should have operations for accepting several types of coins, selecting the type of ticket, selecting a destination, and dispensing a ticket. The order of the operations for selecting type and destination and inserting coins should not be restricted. Also make sure that a ticket is only dispensed when enough money has been inserted.

You can assume that *price* and *ticket* functions are given as follows. For a given destination and ticket type, *price* returns the correct fare and *ticket* returns a ticket

$$[TICKET, DESTINATION]$$

$$TYPE ::= single \mid return$$

$$price : DESTINATION \times TYPE \longrightarrow \mathbb{N}$$
$$ticket : DESTINATION \times TYPE \longrightarrow TICKET$$

(b) [4 marks] Calculate the preconditions of all operations.

(c) [4 marks] How are preconditions in Object-Z and Z interpreted? What influence does this have on refinement?

(d) [4 marks] Combine the above Object-Z class with a CSP process to obtain a ticket machine with the following behaviour: The machine requires the user to first select the destination and then select the ticket type. It then accepts coins. When the ticket price is reached, a ticket is given out.

(e) [4 marks] Combine the above Object-Z class with CSP processes to obtain a ticket machine with the following behaviour: The machine requires insertion of the coins first, followed by selection of ticket type, followed by selection of destination. If sufficient money has been given, a ticket is given out. Make sure the machine does not deadlock, that is, there is always an operation enabled.

Question 4. Semantics [12 marks]

(a) [2 marks] What are the main differences between operational and denotational semantics?

(b) [4 marks] Explain how the properties of a programming/specification language affect the kind of mathematical model used in defining denotational semantics. Illustrate your answer using suitable examples.

(c) [6 marks] We say that two programs, S and T are:

- operationally equivalent, written $S =_{op} R$, if for any initial state, S and T produce the same computation (i.e. perform the same sequence of atomic steps, and pass through the same sequence of states) when executed starting in any give initial state.
- Hoare equivalent, written $S =_H T$, if for any precondition P and postcondition R, either $P \{ S \} R$ and $P \{ T \} R$ both hold or neither of them holds.

(i) EITHER: Show that if S and T are operationally equivalent, then they are also Hoare-equivalent.
OR: Give a counter-example to show that this is not the case.

(ii) EITHER: Show that if S and T are Hoare equivalent, then they are also operationally equivalent.
OR: Give a counter-example to show that this is not the case.

Question 5. Weakest preconditions and refinement [16 marks]

(a) [6 marks] Define the following properties of a statement S, in terms of its weakest precondition semantics:

(i) monotonic (with respect to implication)

(ii) feasible (or strict)

(iii) terminating

(iv) disjunctive

(v) conjunctive

(vi) continuous

(b) [4 marks] Which of Dijkstra's healthiness conditions are **not** required in the refinement calculus? In each such case, explain why that property is not appropriate in a wide-spectrum language, and give an example of a construct in the refinement calculus which does not have that property.

(c) [3 marks] Define the weakest precondition for a specification statement, and show that specification statements are conjunctive.

(d) [3 marks] Define the weakest precondition for sequential composition, and show that sequential composition is monotonic with respect to refinement.

Question 6. Essay [12 marks]

Select a paper (or group of papers) you have read as part of COMP426. Give a brief summary of the paper(s), state the key ideas presented in the paper(s), illustrating them with examples as appropriate, and discuss the significance and/or limitations of the results presented.

(You will not get credit for repeating material used in answers to other questions.)

****