Especificação e Desenvolvimento Formal de 'Software'

Mestrado em Informática + Curso de Especialização em Informática
Ano Lectivo de 2000/01

Exame (Época normal) — 21 de Fevereiro de 2001
15h00
Sala do Mestrado

NB: *Esta prova consta de **5** questões de **4 valores** cada uma.*

PROVA COM CONSULTA (2)

Questão 1 Se um programador observar a sintaxe da linguagem VDM-SL poderá observar que se trata de mais uma linguagem de programação convencional, de sabor funcional. Indique que atributos da notação VDM-SL a convertem, na sua opinião, numa linguagem de *especificação formal*. Justifique sucinta e objectivamente.

Questão 2 A possibilidade de uma função dar resultado indefinido em VDM-SL conduz a uma lógica de funções parciais estendida com o valor lógico *indefinido* ($\star$). Neste contexto, complete a tabela de verdade do predicado $A \wedge \neg B$ que se segue:

A	B	$A \wedge \neg B$
$true$	$true$	
$true$	$false$	
$true$	$\star$	
$false$	$true$	
$false$	$false$	
$false$	$\star$	
$\star$	$true$	
$\star$	$false$	
$\star$	$\star$	

Questão 3 No contexto do problema do sistema de vigilância de uma fábrica química (`alarm.vdm`), dado em anexo, complete a seguinte especificação de uma função que indica quais os tipos de alarmes para os quais há peritos em serviço num determinado período de laboração:

```
QualiOfExperts: Period * Plant -> set of Qualification
QualiOfExperts(p,pl) == ......
pre ....
```

Questão 4 Atente na seguinte especificação do modelo de dados de um sistema de gestão de contas bancárias (muito!) simplificado:

```
types

BAMS = map AccId to Account;
Account :: H: set of AccHolder
           B: Amount;

AccId     = seq of char;
AccHolder = seq of char;
Amount = int;
```

Qual das seguintes duas alternativas para a especificação da operação de abertura de uma conta *nova*,

```
OpenAccount : BAMS * AccId * set of AccHolder * Amount -> BAMS
OpenAccount(bams,n,h,m) == bams munion { n |-> mk_Account(h,m) }
pre not n in set dom bams;
```

ou

```
OpenAccount : BAMS * AccId * set of AccHolder * Amount -> BAMS
OpenAccount(bams,n,h,m) == { n |-> mk_Account(h,m) } ++ bams ;
```

escolheria? Justifique a sua resposta com base na comparação do comportamento esperado de cada uma das alternativas apresentadas para a situação em que se tenta abrir uma conta que já existe.

Questão 5 Suponha que a especificação do problema da questão 3 se altera da forma seguinte:

```
Plant :: sch: Schedule
         alarms: set of Alarm
         experts : map ExpertId to Expert
inv pl ==  (dunion rng pl.sch) subset (dom pl.experts)

Schedule = map Period to set of ExpertId
inv sch = forall exs in set rng sch & exs <> {} ;
.
.
.
Expert :: quali: set of Qualification ;
```

1. Repare que o invariante sobre `Schedule` ficou simplificado. Explique por que é que a cláusula `forall...` deixou de ser necessária e qual o propósito do novo invariante sobre `Plant`.
2. Que alterações passam a ser necessárias (se é que algumas) nas funções `NumberOfExperts` e `ExpertIsOnDuty`? Justifique.

Anexo

```
--
-- Chemical plant model
-- Source: adapted from "Modelling Systems" by JNO
-- File: -rw-------    1 jno      jno           899 Jan 12 14:16 alarm.vdm
--

types

Plant :: sch    : Schedule
         alarms : set of Alarm ;

Schedule = map Period to set of Expert
inv sch == forall exs in set rng sch &
                  exs <> {} and
                  forall ex1,ex2 in set exs &
                         ex1.expertId = ex2.expertId => ex1 = ex2 ;

Alarm :: alarmtext : seq of char
         quali     : Qualification ;

Qualification = <Elec> | <Mech> | <Bio> | <Chem> ;

Expert :: expertId : ExpertId
          quali    : set of Qualification
inv ex == ex.quali <> {};

ExpertId = token ;
```

```
Period = token ;

functions

NumberOfExperts: Period * Plant -> nat
NumberOfExperts(per,pl) == card pl.sch(per)
pre per in set dom pl.sch ;

ExpertIsOnDuty: Expert * Plant -> set of Period
ExpertIsOnDuty(ex,pl) ==
{per | per in set dom pl.sch &
       ex in set pl.sch(per)} ;

ExpertToPage(al: Alarm, per: Period, pl: Plant) r: Expert
pre per in set dom pl.sch and
    al in set pl.alarms
post r in set pl.sch(per) and
     al.quali in set r.quali ;

-- end of alarm.vdm
```