Tentamen Formele Methoden voor Software Engineering (192135201)

19 april 2012, 8:45–12:15 uur.

- Vermeld je studierichting op het tentamen
- Geef aan of je de huiswerkopgaven gemaakt hebt, en in welke groep/bij welke werkcollegeleider.
- Naast de sheets van de colleges mag je de FSP Quick Reference Card en de JML Cheat Sheet gebruiken.
- Het cijfer voor dit tentamen is gelijk aan het het behaalde aantal punten gedeeld door 10.

---

1. *(25 punten)* Beschouw de volgende FSP-specificatie:

```
// Ideale winkel
IAPC1 = (vraag -> koop -> IAPC1).

// Winkel waar niet alles op voorraad is
ZAAK2 = ( vraag -> ( koop -> ZAAK2 | bestel -> BESTEL) ),
  BESTEL = (wacht -> BESTEL | lever -> koop -> ZAAK2).

||IAPC2 = ZAAK2 \ {bestel, wacht, lever}.

// Winkel met parallel bestelproces
ZAAK3 = ( vraag -> ( koop -> ZAAK3 | bestel -> ZAAK3 )
        | lever -> koop -> ZAAK3
        ).
BESTEL3 = ( bestel -> lever -> BESTEL3 ).

||IAPC3 = (ZAAK3 || BESTEL3) \ {bestel, lever}.
```

   (a) *(8 punten)* Teken de transitiesystemen van `IAPC1`, `IAPC2` en `IAPC3`.

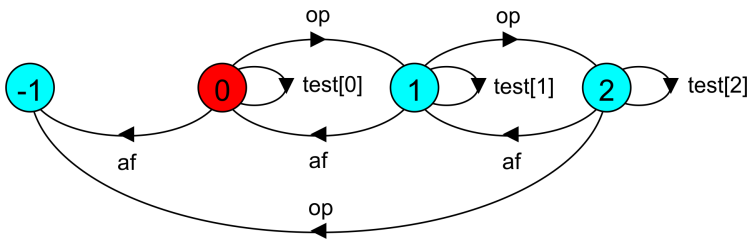
   (b) *(7 punten)* Welke van de drie bovenstaande processen zijn bisimilair? Geef de relatie tussen toestanden aan bij bisimilaire processen, of leg anders uit waarom dit niet kan.

   (c) *(5 punten)* Voldoet `IAPC3` aan de standaard-*progress*-eigenschap? Let uit waarom, of waarom niet.

   (d) *(5 punten)* Formuleer een *safety property* die uitdrukt dat er uitsluitend afwisselend `vraag`- en `koop`-acties kunnen plaatsvinden, en teken het bijbehorende transitiesysteem. Welke van de bovenstaande processen voldoet niet aan deze eigenschap, en onder welke omstandigheden?

2. *(25 punten)*

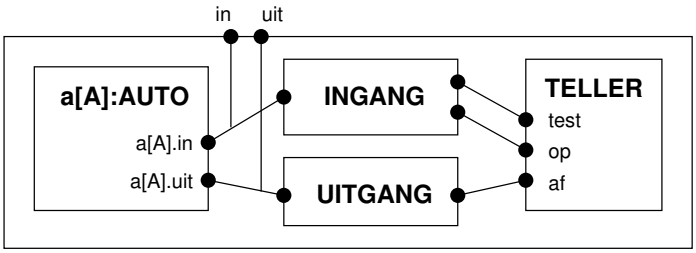
(a) (*8 punten*) Schrijf een process `TELLER` in FSP dat het volgende gedrag vertoont:



(b) (*10 punten*) Specificeer een parkeergarage met één ingang, één uitgang en een aantal auto's, waarbij:

- De ingang telkens test of de teller lager dan 2 staat, en zo ja, een auto toelaat en de teller ophoogt;
- De uitgang telkens een auto laat vertrekken en vervolgens de teller aflaagt;
- Een auto telkens de parkeergarage in- en uitrijdt.

Het systeem is schematisch in de volgende figuur weergegeven:



Geef FSP-definities voor `AUTO`, `INGANG`, en `UITGANG` en het samengestelde systeem.

(c) (*7 punten*) Is het samengestelde systeem veilig, in de zin dat er geen *error*-toestand bereikbaar is? Licht het antwoord toe.

3. *(30 punten)* Beschouw de volgende Java-interface:

```
/** Periode in de geschiedenis.                                    *
  * Een tijdperk wordt bepaald door een begin- en een eindjaar *
  * met minstens een jaar verschil. Jaartallen zijn positief.  */
interface Tijdperk {
    /** Verandert het beginjaartal van dit tijdperk. */
    void setBegin(int begin);

    /** Verandert het eindjaartal van dit tijdperk. */
    void setEind(int eind);

    /** Levert het beginjaartal van dit tijdperk op. */
    int getBegin();

    /** Levert het aantal jaren van dit tijdperk op. */
    int getDuur();

    /** Levert de combinatie van dit tijdperk en een
      * direct aansluitend ander tijdperk op. */
    Tijdperk combinatie(Tijdperk ander);
}
```

   (a) *(12 punten)* Stel een JML-contract op voor `Tijdperk`, waarin de beoogde werking zoveel mogelijk formeel gespecificeerd is. *Maak gebruik van modelvariabelen voor beginjaar en duur.*

   (b) *(8 punten)* Schrijf een implementatie voor `Tijdperk` *met velden voor het begin- en eindjaartal* en voorzien van alle additionele JML-specificaties die nodig zijn om te kunnen bewijzen dat de klasse correct is.

   (c) *(10 punten)* Bewijs de correctheid van de (zelf geïmplementeerde) methode `combinatie`.

4. *(20 punten)* Het maximum van een **`int`**`[]`-array is een waarde x in de array zodat geen van de (andere) waarden in de array groter is dan x.

   (a) (*3 punten*) Formuleer in predicaatlogica en in JML de eigenschap dat `x` het maximum van **`int`**`[] a` is (*zonder* de speciale JML-syntax voor `\max` te gebruiken).

   (b) (*3 punten*) Geef een contract voor de volgende Java-methode dat uitdrukt dat de methode het maximum van zijn argument oplevert:

```
int max(int[] a) {
    int result = a[0];
    int i = 1;
    while (i < a.length) {
        if (a[i] > result) {
            result = a[i];
        }
        i++;
    }
    return result;
}
```

   (c) (*14 punten*) Bewijs dat de methode correct is.