Système de surveillances des températures

MGL7160 Méthodes formelles et semi-formelles

Hiver 2001

1 Description du travail

1.1 Objectif

L'objectif de ce travail est de vous familiariser avec le développement de spécifications formelles pour un système réactif et concurrent. Ces spécifications devront être écrites dans la notation `LOTOS` et viseront à spécifier un petit système de surveillance des températures.

1.2 Ce qui vous est fourni

Une description informelle du système (rôle + entrées/sorties) est présentée à la section 2.

1.3 Ce que vous devez faire

Vous devez produire une spécification du système de surveillance des températures à l'aide de la notation `LOTOS`. Cette spécification doit utiliser et contrôler chacun des signaux et ports décrits à la section 2.

2 Description du système de surveillance des températures

2.1 Vue d'ensemble et rôle du système

On désire spécifier un système — appelons le ST — qui vise à surveiller la température d'un ensemble de sites de façon à s'assurer que la température, à chacun des endroits, ne dépasse jamais une température limite spécifiée. Si une telle situation survient, une alarme doit alors être déclenchée. À chacun des sites est associé un senseur permettant de mesurer la température ainsi qu'un thermostat permettant de fixer la température

désirée. À intervalles réguliers, les températures doivent être mesurées et vérifiées. De plus, une trace des informations sur les températures doit être conservée (produite par l'intermédiaire d'une imprimante).

2.2 Description détaillées des fonctions et des signaux et ports

Les signaux qui contrôlent directement le système ST

Les signaux qui suivent servent à contrôler l'activation du système de surveillance des températures (ST):

- `activerST`: Active le système ST. À partir du moment où ce signal est reçu, les températures seront vérifiées et une trace sera produite sur l'imprimante.
- `desactiverST`: Désactive le système ST. Les alarmes, si elles sont actives (en train de sonner), sont éteintes et toute activité de surveillance cesse, y compris la production de la trace.
- `eteindreAlarmes`: Permet à l'usager d'indiquer que toutes les alarmes actuellement allumées doivent être éteintes. Contrairement au signal précédent, l'activité de surveillance se poursuit.

Les ports pour l'interface avec les senseurs

- Des senseurs sont installés à divers endroits. La température courante à un site `id` peut être obtenue en effectuant une action de la forme suivante:

  ```
  lireValeurSenseur !id ?t: temperature.
  ```

  Chaque senseur reconnu par le système de surveillance est identifié par un numéro unique (`id`).

- À chaque site où un senseur est installé, il est possible de positionner un sélecteur (thermostat) de façon à indiquer une température qui ne doit pas être dépassée. Cette commande est de la forme suivante, où `id` identifie le senseur associé:

  ```
  specifierLimite !id !limite
  ```

  Initialement, la température limite par défaut sera de 20 C.

Le nombre de sites et senseurs à controler devra être indiqué par un argument de la spécification du système global.

Les ports pour l'interface avec les alarmes

- `alarmeTemperature`: Permet de contrôler l'alarme de température. Pour activer cette alarme, une action de la forme suivante doit être effectuée, action n'ayant aucun effet si l'alarme est déjà active:

  ```
  alarmeTemperature !allumer
  ```

  Quant à l'action suivante, elle permet d'éteindre un signal d'alarme (aucun effet si l'alarme n'est pas active):

  ```
  alarmeTemperature !eteindre
  ```

- `alarmeImprimante`: Semblable au port précédent, mais pour l'alarme associée au mauvais fonctionnement de l'imprimante. Voir le prochain paragraphe.

Les ports pour l'interface avec l'imprimante

- Un port `ecrireLog` permet d'effectuer des écritures sur l'imprimante. De façon régulière, le système de surveillance des températures doit lire le statut des différents senseurs et doit transmettre ces informations — numéro du senseur, heure et date courante, température mesurée, valeur booléenne indiquant si la température limite a été dépassée ou non — à l'imprimante.

- Pour assurer que les impressions sont faites correctement, après chaque impression le système doit s'assurer de recevoir une confirmation — `impressionEffectuee` — de la part de l'imprimante, confirmation indiquant que l'impression fut bien effectuée. Si après un certain délai cette confirmation n'est pas reçue, alors une alarme doit être déclenchée (`alarmeImprimante`).

Le port pour obtenir la date et l'heure

- Un port permet au système de déterminer la date et l'heure courante (interface avec une horloge externe) par une commande de la forme suivante:

  ```
  dateEtHeure ?dh: dateHeure
  ```

3 Suggestions et remarques

- Utilisez des paramètres (arguments de la spécification de haut niveau) pour indiquer les constantes spéciales (nombre de senseurs, température initiale, etc.).

- Bien que la notion de `CONCEPT` ne fasse pas partie du langage `LOTOS`, vous pouvez utiliser cette construction pour définir des concepts auxiliaires, par ex., conversion d'information en chaîne pour impression. Vous pouvez aussi, si nécessaire, utiliser tous les types disponibles en `Spec`.

- Il est possible d'avoir plusieurs instances d'un même processus en les distinguant à l'aide d'un argument supplémentaire qui représente un identifiant unique à chaque instance. Soit un processus défini (en partie) comme suit:

```
PROCESS Proc[act, ...]( np: numProc, ... ): NOEXIT :=
  act ?n: numProc ?x1: t1 ?x2: t2 [np = n]; ...
ENDPROC
```

  Ici, le premier argument du processus — entre "(...)" — représente le numéro d'identification unique du processus. La transition `act` reçoit elle aussi, comme premier argument, un numéro d'identification. Grâce au prédicat de sélection (`[np = n]`), seul le processus `Proc` ayant le numéro indiqué effectuera cette action, les autres n'effectuant aucune action. Ceci permet donc d'avoir plusieurs instances concurrentes d'un même processus, où une seule des instances effectue une action.

  Une façon différente, mais équivalente d'exprimer l'action avec garde plus haut serait aussi la suivante:

```
act !np ?x1: t1 ?x2: t2; ...
```

  Par exemple, soit l'expression suivante:

```
( Proc[act, ...](1, ...)
  |||
  Proc[act, ...](2, ...)
  |||
  Proc[act, ...](3, ...)
)
|[act]|
act !2 ...; act !3 ...;
```

  Alors, la première action `act` ne sera synchronisée avec et effectuée que par la deuxième instance du processus (`Proc[...](2, ...)`), alors que la deuxième action ne sera effectuée que par la troisième instance.