

Le Projet SCEPTRE

2.2. Les Opérateurs T. R. de SCEPTRE

◆ Les primitives définies par SCEPTRE

CATEGORIE	OPERATION ELEMENTAIRE	PARAMETRES	FONCTION
GESTION DES TACHES	DEMARRER	TACHE	Lancer l'exécution de la tâche
	ARRETER	TACHE	Arrêter la tâche
	CONTINUER	TACHE	Continuer l'exécution de la tâche
	SE TERMINER		Terminer l'exécution de la tâche qui l'exécute
	CHANGER-PRIORITE	TACHE, nouvelle priorité	Donner à la tâche la nouvelle priorité
	ETAT	TACHE	Fournir l'état de la tâche
	PRIORITE	TACHE	Fournir la priorité de la tâche
	TACHE COURANTE		Fournir l'identité de la tâche qui l'exécute
	STATUT		Fournir le statut de l'opération élémentaire qui vient d'être exécutée par la tâche
SIGNALISATION	SIGNALER	EVENEMENT, TACHE	Signaler à la tâche que l'événement est arrivé
	ATTENDRE	Liste d'EVENEMENTS	Attendre que l'un (au moins) des événements de la liste soit arrivé
	ARRIVE	Liste d'EVENEMENTS	Prédicat qui est vrai si tous les événements de la liste sont arrivés
	EFFACER	Liste d'EVENEMENTS	Remettre les événements de la liste dans l'état non arrivé

Module SETR - Vincent Bombardier 11 janvier 2013

Le Projet SCEPTRE

2.2. Les Opérateurs T. R. de SCEPTRE

◆ Les primitives définies par SCEPTRE

CATEGORIE	OPERATION ELEMENTAIRE	PARAMETRES	FONCTION
COMMUNICATION ENTRE TACHES	ENVOYER	ELEMENT, FILE	Envoyer l'élément à la queue dans la file
	RETIRER	ELEMENT, FILE	Retirer le premier élément de la file
	VIDE	FILE	Prédicat qui est vrai si la file est vide
	PLEINE	FILE	Prédicat qui est vrai si la file est pleine
EXCLUSION MUTUELLE	ENTRER	REGION	Demander la propriété exclusive de la région
	SORTIR	REGION	Relâcher la propriété exclusive de la région

Module SETR - Vincent Bombardier 11 janvier 2013

Modèle CONSOMMATEUR - PRODUCTEUR

• Avec des Sémaphores:

$$cpt(s_1) = N$$

Prod

$P(s_1)$ $* (pt+i) = val$
<production> $i = (i+1) \% N$
 $V(s_2)$

(avec plusieurs producteurs)

$$cpt(s_2) = 0$$

Cons

$P(s_2)$ $val = * (pt+j)$
 $j++$
 $V(s_1)$

Prod

$P(s_1)$
 $P(mutex1)$
 $* (pt+i) = val$
 $i = (i+1) \% N$
 $V(mutex1)$
 $V(s_2)$

Cons

$P(s_2)$
 $P(mutex2)$
 $val = * (pt+j)$
 $j++$
 $V(mutex2)$
 $V(s_1)$

On suppose que la file d'attente est de taille non bornée

Prod

ENTRER (R)
 ENVOYER (X, F)
 SIGNALER (F-NON-VIDE) CONSO
 SORTIR (R)

Conso

ENTRER (R)
 si VIDE (F) alors
 EFFACER (F-NON-VIDE) SORTIR (R)
 ATTENDRE (F-NON-VIDE)
 si SORTIR (R) si
 RETIRER (Y, F)

Prendre

(Semaîtres)

Vendre

cpt -- ;
 si cpt <= 0 alors
 < mise en file d'attente
 fi

cpt ++ ;
 si cpt >= 0
 < réveille 1 processus de la file d'attente
 fi

i=1 $w(t=0) = C_1$
 $w(t=1) = C_1 = 1 \leq D_1$

i=2 $w(t=0) = C_2 = 2$
 $w(t=2) = C_1 \times 1 + C_2 = 3$
 $w(t=3) = C_1 \times \left\lceil \frac{3}{4} \right\rceil + C_2 = 3 \leq D_2$

i=3 $w(t=0) = C_3 = 2$
 $w(t=2) = C_1 \times \left\lceil \frac{2}{4} \right\rceil + C_2 \times \left\lceil \frac{2}{6} \right\rceil + C_3$
 $= C_1 + C_2 + C_3 = 5$
 $w(t=5) = C_1 \times \left\lceil \frac{5}{4} \right\rceil + C_2 \times \left\lceil \frac{5}{6} \right\rceil + C_3 = 6$
 $w(t=6) =$

Test de terminaison

↳ montre que sa respecte les contraintes de temps

Plus que le test d'acceptabilité n'était pas bon