GLO-3004 : Spécification formelle et vérification de logiciels

Travail dirigé 3 –
Composition parallèle

1. Une montagne russe ne prend le départ que lorsque sa capacité maximale (M) de passagers est atteinte. Les passagers arrivent sur la plateforme, puis ils s'assoient. La plateforme ne doit pas nécessairement être pleine avant que les passagers s'assoient. Un passager qui est sur la plateforme peut attendre avant de s'asseoir. La plateforme ne peut pas accueillir plus de M passagers. Lorsque M passagers sont assis dans la voiture, on annonce le départ. À ce moment, la voiture roule, s'arrête, se vide de ses passagers et elle est alors prête à recevoir d'autres passagers. À partir du moment où le départ est annoncé, on peut faire (à tout moment) un arrêt d'urgence qui stoppe tout le circuit.

   Vous devez écrire le code `FSP` des processus `PLATEFORME` et `VOITURE`. Les actions de `PLATEFORME` sont `arrive`, `assit`, `depart`. Les actions de `VOITURE` sont `assit`, `depart`, `roule`, `arrete`, `vide`, `urgence`. Écrivez le code `FSP` de la `MONTAGNE_RUSSE` qui est la composition parallèle de `PLATEFORME` et `VOITURE`.

2. Prenez la définition du jardin botanique incluse dans le fichier joint avec le TD. Quelles sont les traces possibles parmi les suivantes ? Justifiez.

   (a) $est.ouvre, ouest.ouvre, est.arrive, ouest.arrive$

   (b) $ouvre, est.arrive, ouest.arrive, est.valeur.obtient, ouest.valeur.obtient$

   (c) $ouvre, est.arrive, ouest.arrive, est.valeur.obtient, est.valeur.lit.0,$
   $est.valeur.ecrit.1, ferme$

   (d) $ouvre, est.arrive, est.valeur.obtient, est.valeur.lit.0, est.valeur.ecrit.1, ferme$

   (e) $ouvre, est.arrive, est.valeur.obtient, est.valeur.lit.0,$
   $est.valeur.ecrit.1, est.valeur.rend, \ldots, est.arrive, est.valeur.obtient,$
   $est.valeur.lit.4, est.valeur.ecrit.5, est.valeur.rend$

3. Voici le code `FSP` d'un système de `N` écluses avec un bateau. Comprenez cette spécification dans les moindres détails avant de s'attaquer aux questions ! Cela vous permettra

de gagner du temps ! Vous pouvez vous inspirez de la figure `ecluse.png` fournie pour comprendre l'idée.

```
const N = 2 // Nombre d'ecluses

Ecluse       = EclusePleine,
EclusePleine = ( vider -> EcluseVide
                | porte[1].ouvrir -> porte[1].fermer -> EclusePleine ),
EcluseVide   = ( remplir -> EclusePleine
                | porte[2].ouvrir -> porte[2].fermer -> EcluseVide ).

// La valeur 0 correspond a la rive Ouest
// La valeur N+1 correspond a la rive Est
// La valeur 0 < i < N+1 correspond a la ieme ecluse
Bateau    = Bateau[0],
Bateau[0] = ( avancer.ouest -> Bateau[0]
            | porte[1].ouvrir
               ->( porte[1].fermer -> Bateau[0]
                  | avancer.est -> porte[1].fermer -> Bateau[1] ) ),
Bateau[i:1..N] = ( porte[i].ouvrir
                    -> ( porte[i].fermer -> Bateau[i]
                        | avancer.ouest -> porte[i].fermer -> Bateau[i-1] )
                  | porte[i+1].ouvrir
                    -> ( porte[i+1].fermer -> Bateau[i]
                       | avancer.est -> porte[i+1].fermer -> Bateau[i+1] ) ),
Bateau[N+1] = ( avancer.est -> Bateau[N+1]
              | porte[N+1].ouvrir
                -> ( porte[N+1].fermer -> Bateau[N+1]
                   | avancer.ouest -> porte[N+1].fermer -> Bateau[N] ) ).


||Traverse = forall[i:1..N]( eclu[i]:Ecluse )
             / { porte[i]/eclu[i].porte[1], porte[i+1]/eclu[i].porte[2] }.

||Trafic = ( Bateau || Traverse ).
```

(a) On veut cacher les actions d'ouverture et fermeture des portes. Dites comment modifier le code.

(b) On revient au code initial. De quelle façon modifieriez-vous le code pour représenter le fait que la porte `N` (et seulement cette porte) ne peut plus s'ouvrir ? Y a-t-il blocage ? Expliquez la cause s'il y a blocage, sinon, expliquez pourquoi il n'y a pas de blocage.

(c) On revient au code initial. Modifiez le code de telle sorte que :
— il contienne `B` bateaux ; attention, les écluses ne doivent pas être modifiées ;

— l'entrée dans la série d'écluses soit représentée par l'action `entrer` : i.e., à la porte 1, l'action `avancer.est` devient `entrer` et l'action `avancer.ouest` devient `sortir`. De même, à la porte `N+1`, l'action `avancer.est` devient `sortir` et l'action `avancer.ouest` devient `entrer`.

(d) On revient à la spécification du #3c. Ajoutez un processus `CONTROLEUR` qui s'assure qu'il n'y a jamais plus d'un bateau dans le système d'écluses.