Projet SF

RAPPORT

Modèlisation et Vérification du comportement de véhicules automatiques sur un pont à voie unique

Réalisé par ***DOAN Cao Sang***

5 décembre 2015

Chapitre 1

Propriété

*L*e programme doit satisfaire les propriétés suivantes :

1. Il n'y a pas de collision (i.e. deux véhicules circulants en sens inverse) sur le pont.
2. Un véhicule qui arrive est certain de passer sur le pont à l'issue d'une durée bornée.

Chapitre 2

Explication

2.1 Question 1.1

2.1.1 Vaa

VerificationA est une place sortie, qui lie le composant *Vaa* avec le contrôlleur *CTRLP*, lorsque un véhicule A veut traverser le pont, d'abord il communique avec le contrôlleur *CTRLP* pour avoir son autorisation.

OKControlleurA est une place entrée qui lie entre *Vaa* et *CTRLP*, lorsque le *Vaa* peut traverser sur le pont, le *CTRLP* met son jeton pour que *Vaa* puisse passer.

OKPont est une place entrée qui lie le *P* avec le *Vaa*, le *P* marque cette place lors qu'il reçoit l'autorisation de contrôlleur *CTRLP*.

AuRevoirControlleurA est une place sortie, le *Vaa* marque cette place dès qu'il sort du *P*.

2.1.2 Vab

VerificationB est une place sortie, qui lie le composant *Vab* avec le contrôlleur *CTRLP*, lorsque un véhicule B veut traverser le pont, d'abord il communique avec le contrôlleur *CTRLP* pour avoir son autorisation.

OKControlleurB est une place entrée qui lie entre *Vab* et *CTRLP*, lorsque le *Vab* peut traverser sur le pont, le *CTRLP* met son jeton pour que *Vab* puisse passer.

OKPont est une place entrée qui lie le *P* avec le *Vab*, le *P* marque cette place lors qu'il reçoit l'autorisation de contrôlleur *CTRLP*.

AuRevoirControlleurB est une place sortie, le *Vab* marque cette place dès qu'il sort du *P*.

2.1.3 Pont

CapaciteControlleur est une place sortie, qui lie le composant *P* avec le contrôlleur *CTRLP* pour lui signaler sa capacité restante.

OKPont est une place sortie qui lie le *P* avec les 2 composants *Vaa* et *Vab*, le *P* marque cette place lorsque sa capacité reste suffisante.

2.1.4 CTRLP

VerificationA est une place entrée, qui lie le composant *Vaa* avec le contrôlleur *CTRLP*, lorsque un véhicule A veut traverser le pont, le *CTRLP* doit d'abord, recevoir sa demande.

VerificationB est une place entrée, qui lie le composant *Vab* avec le contrôlleur *CTRLP*, lorsque un véhicule B veut traverser le pont, le *CTRLP* doit d'abord, recevoir sa demande.

OKControlleurA est une place sortie qui lie entre *Vaa* et *CTRLP*, lorsque le *Vaa* peut traverser sur le pont, le *CTRLP* met son jeton pour que *Vaa* puisse passer.

OKControlleurB est une place sortie qui lie entre *Vab* et *CTRLP*, lorsque le *Vab* peut traverser sur le pont, le *CTRLP* met son jeton pour que *Vab* puisse passer.

CapaciteControlleur est une place entrée, qui lie le composant *P* avec le contrôlleur *CTRLP* pour savoir la capacité restante du *P*.

AuRevoirControlleurA est une place entrée, cette place est marqué par le *Vaa*.

AuRevoirControlleurB est une place entrée, cette place est marqué par le *Vab*.

2.2 Question 1.2

Cette composant contient 8 places, dont 4 ont pour le rôle d'interface, et 3 transitions.

La place **VehiculeA** modélise le véhicule A lors qu'il arrive devant le pont. S'il veut passer sur le pont, d'abord, il communique avec le contrôlleur par la transition *demanderEntrerA*, cette transition met un jeton dans l'interface **VerificationA** qui lie avec le contrôlleur *CTRLP* et passe à l'état **AttenteAutorisationA**. Lorsqu'il reçoit l'autorisation du contrôlleur *CTRLP* via l'interface **OKControlleurA** et celle du pont *P* via l'interface **OKPont**, il peut traverser le pont, en passant à l'état **DansPontA**. Ensuite, quand il sort, il marque la place d'interface **AuRevoirControlleurA** et aussi passe à l'état **FiniA** par la transition *sortPontA*.

La sécurité est garantie car, pour passer à l'état **AttenteAutorisationA** le véhicule doit communiquer avec le contrôlleur *CTRLP*. Pour entrer, il doit avoir l'autorisation du contrôlleur *CTRLP* et consommer le jeton dans la place d'interface **OKPont**, cette place modélise la consommation de la capacité du pont *P*. Enfin, il signale sa sortie du pont au contrôlleur *CTRLP* en mettant un jeton dans la place **AuRevoirControlleurA**.

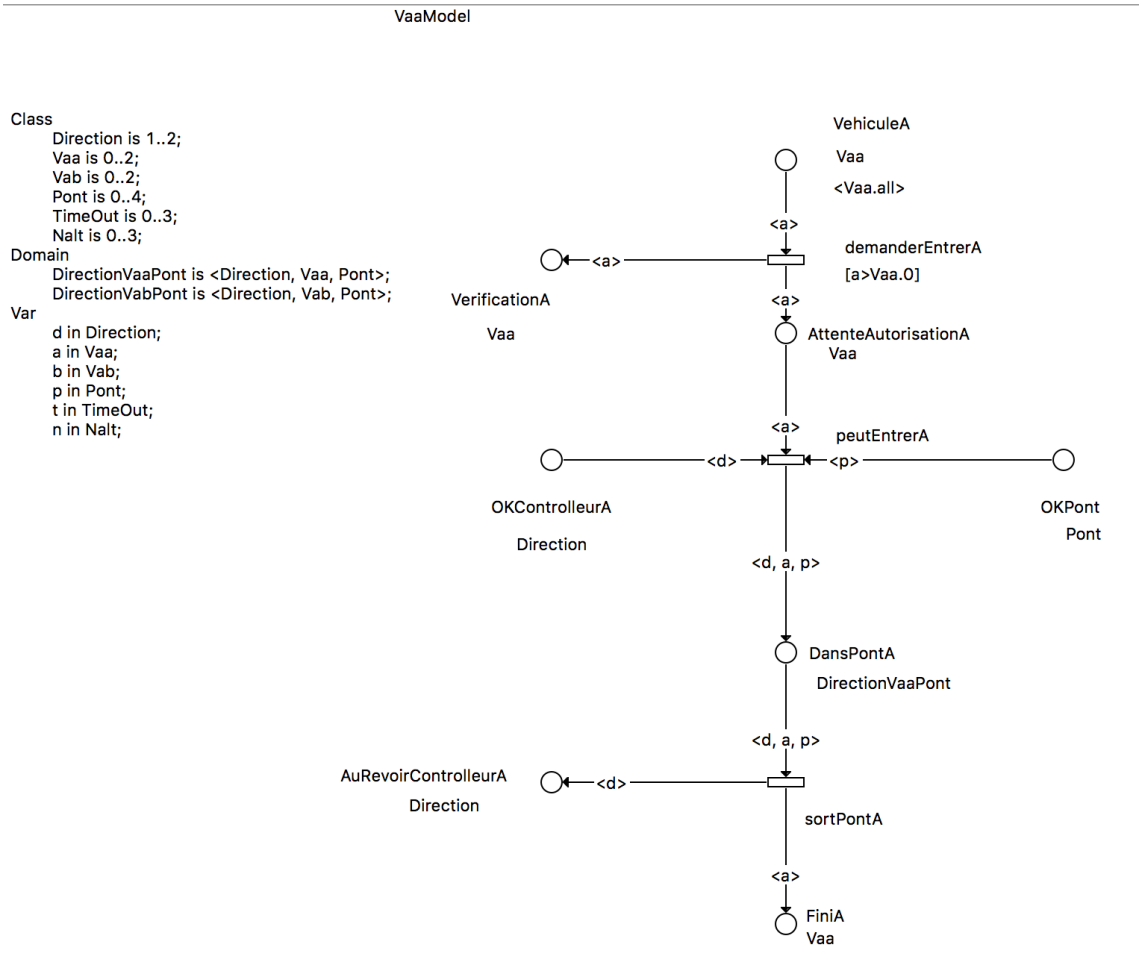

FIGURE 2.1 – Le composant modélise le véhicule automatisée A.

2.3 Question 1.3

Cette composant est identique avec celle de véhicule A, contient 8 places, dont 4 ont pour le rôle d'interface, et 3 transitions.

La place **VehiculeB** modélise le véhicule B lors qu'il arrive devant le pont. S'il veut passer sur le pont, d'abord, il communique avec le contrôlleur par la transition *demanderEntrerB*, cette transition met un jeton dans l'interface **VerificationB** qui lie avec le contrôlleur *CTRLP* et passe à l'état **AttenteAutorisationB**. Lorsqu'il reçoit l'autorisation du contrôlleur *CTRLP* via l'interface **OKControlleurB** et celle du pont *P* via l'interface **OKPont**, il peut traverser le pont, en passant à l'état **DansPontB**. Ensuite, quand il sort, il marque la place d'interface **AuRevoirControlleurB** et aussi passe à l'état **FiniB** par la transition *sortPontB*.

La sécurité est garantie car, pour passer à l'état **AttenteAutorisationB** le véhicule doit communiquer avec le contrôlleur *CTRLP*. Pour entrer, il doit avoir l'autorisation du contrôlleur *CTRLP* et consommer le jeton dans la place d'interface **OKPont**, cette place modélise la consommation de la capacité du pont *P*. Enfin, il signale sa sortie du pont au contrôlleur *CTRLP* en mettant un jeton dans la place **AuRevoirControlleurB**.

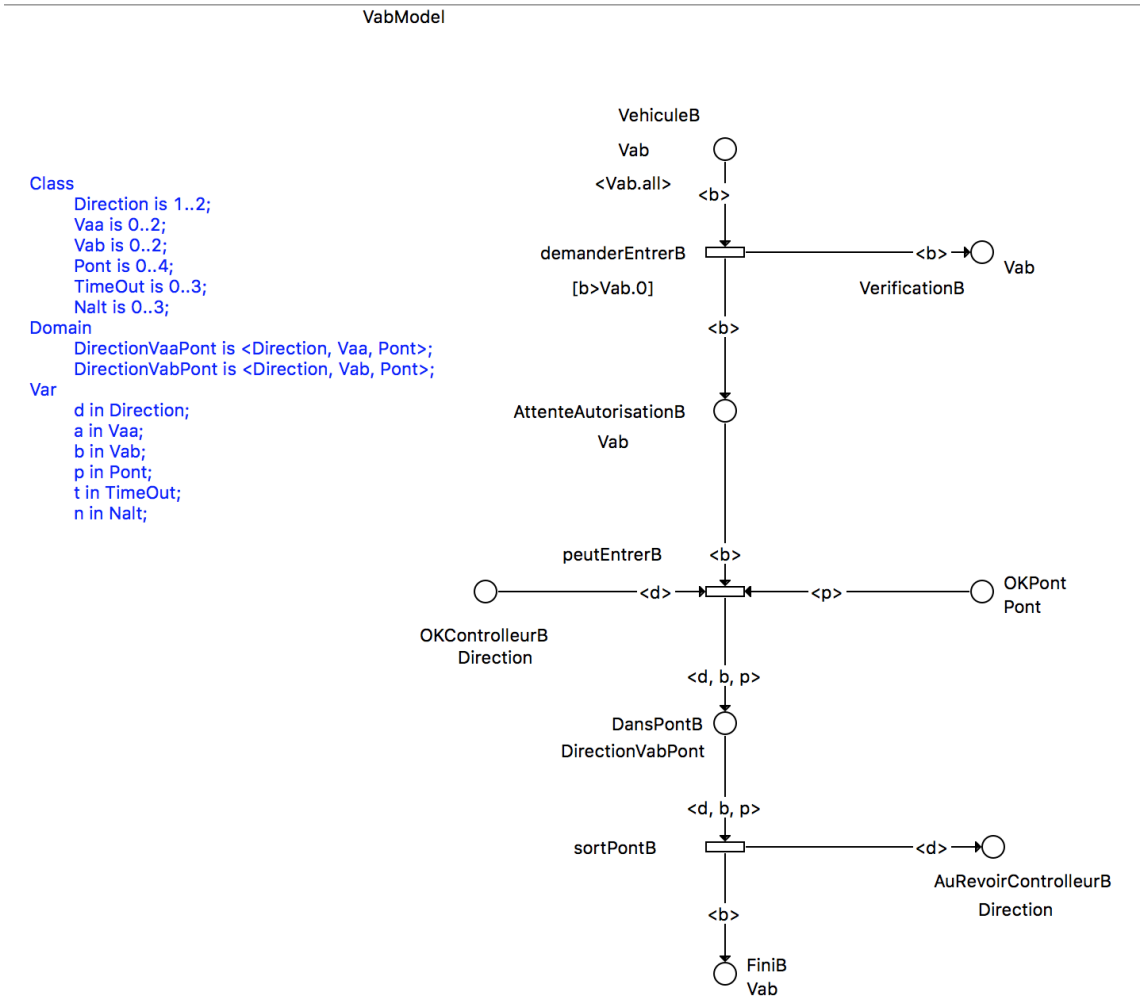

FIGURE 2.2 – Le composant modélise le véhicule automatisée B.

2.4 Question 1.4

Le comportement d'un pont est simple, il notifie sa capacité au contrôlleur *CTRLP* et diminue si consomé par un véhicule. Donc, il contient 3 places, dont 2 interfaces et une transition.

La transition *notifieCapacite* marque 2 interfaces **CapaciteControlleur** et **OKPont** par 2 jetons, 1 pour chaque.

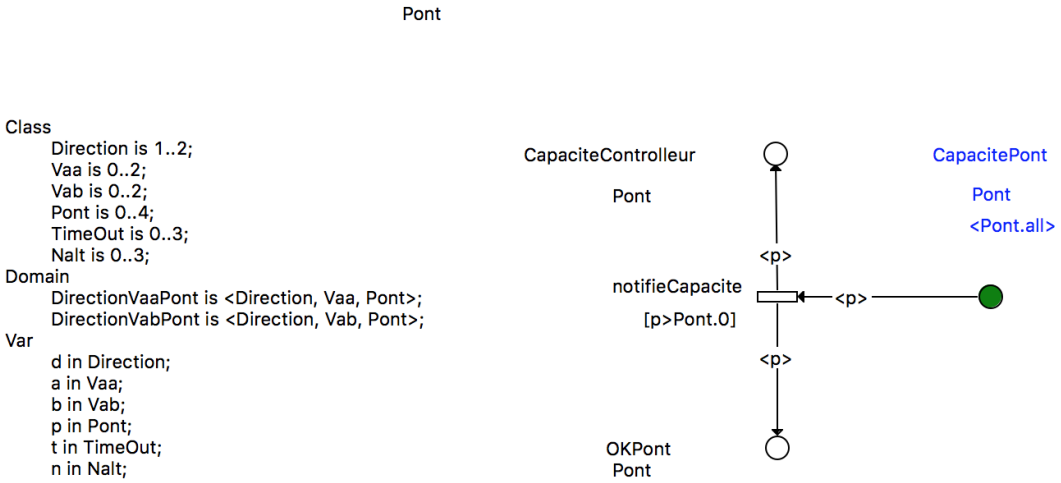

FIGURE 2.3 – Le composant modélise le pont.

2.5 Question 1.5

Ce composant contient 12 places dont 7 interfaces, et 6 transitions.

Lorsque le *CTRLP* reçoit une demande d'entrer d'une véhicule *Vaa* (idem, *Vab*), il vérifie le sens actuel (initilise à 1, direction de A vers B) est correspondant à celui de *Vaa*, le timeout restant, le nombre *Nalt* restant et la capacité *CapaP* restant du pont. Si ces conditions sont satisfaits, alors le contrôlleur *CTRLP* donne l'autorisation au véhicule en attente. Quand le véhicule sort du pont, le contrôlleur reçoit son signal.

Le timeout est modélisé par une place nommée *TimeOut* avec une seule transition *ecoule*. Le *TimeOut* décrémente une unité de temps automatique indépendamment avec le système. Lors de l'expiration de *TimeOut*, il réinitialise le *Nalt*, lui même, et change la direction actuelle. Idem pour le *Nalt*, si $Nalt = 0$, il réinitialise le *TimeOut*, *Nalt* et change la direction.

Quand la capacité du pont est à 0, alors aucun véhicule peut utiliser ce pont.

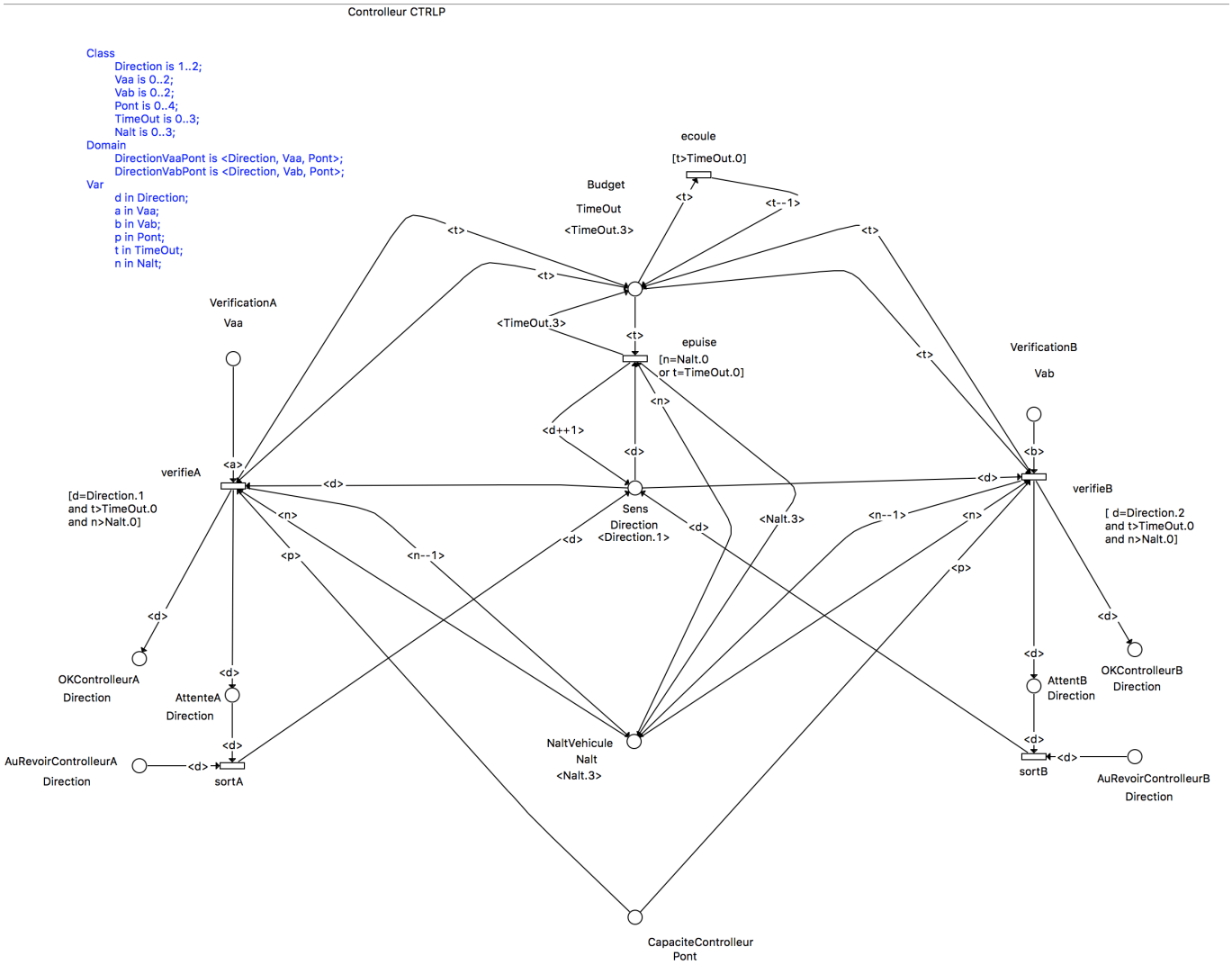

FIGURE 2.4 – Le composant modélise le pont.

2.6 Question 1.6

Cette assemblage contient 22 places, 13 transitions et 55 arcs.

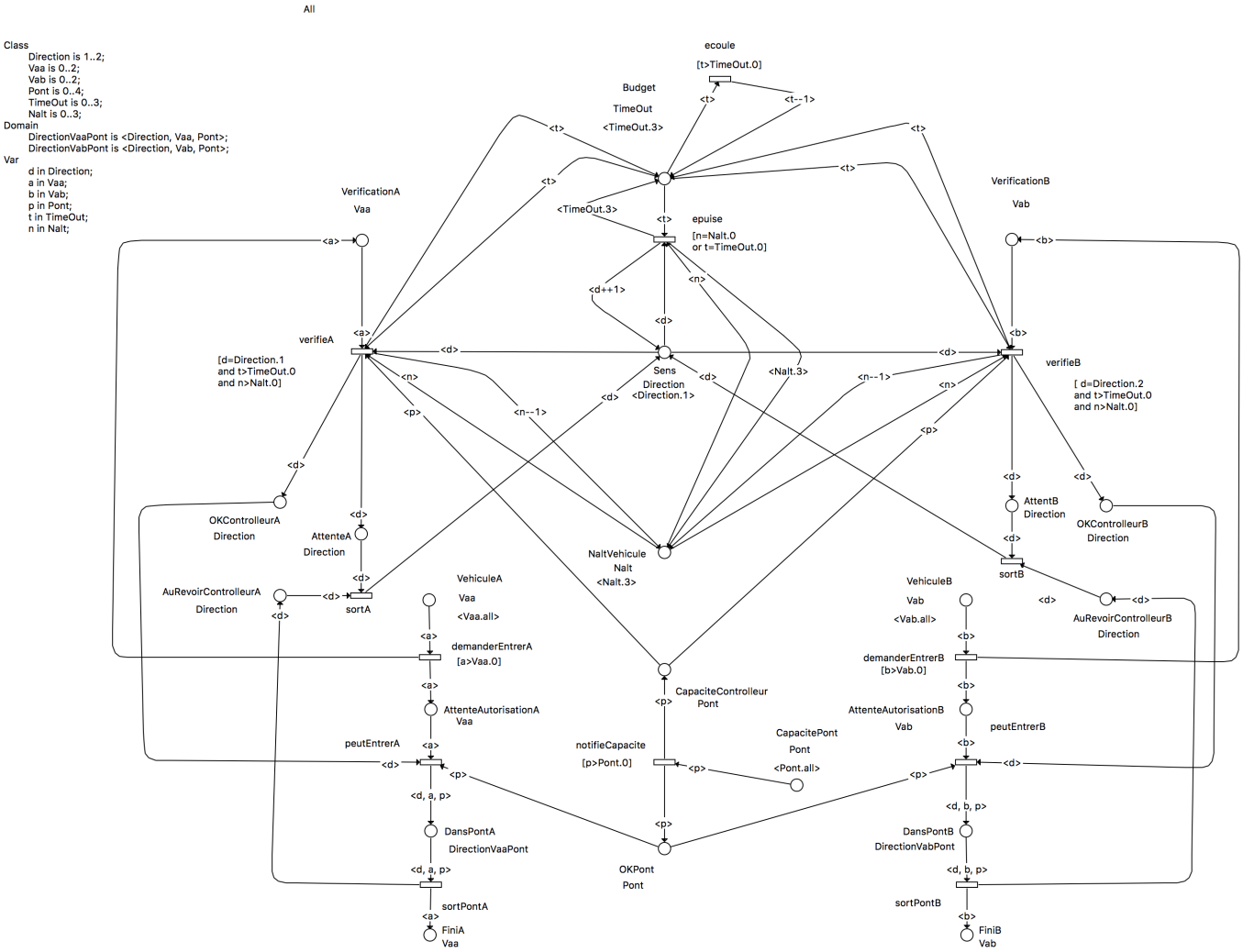

FIGURE 2.5 – Assemblage.

2.7 Question 1.7

Paramètres

Pour faciliter des vérifications car ma machine prend un temps énorme pour résoudre les formules. J'ai choisi :

— $N_{Vaa} = 2$
— $N_{Vab} = 2$
— $Capa_p = 4$
— $N_{Alt} = 3$

$Capa_p$ doit être supérieur au nombre total de véhicules, car sinon la propriété P2 ne peut être vérifiée.

Vérification par CosyVerif4PN

Pour vérifier la propriété P1, *"Il n'y a pas de collision (i.e. deux véhicules circulants en sens inverse) sur le pont."*, j'ai utilisé les formules suivantes :

1. *query node ((card(DansPontA) + card(DansPontB)) == 2)* (Fichier P1_1.txt)
2. *query node ((DansPontA == <.1.> and DansPontB == <.1.>)*
   or (DansPontA == <.1.> and DansPontB == <.2.>)) (Fichier P1_2.txt)
3. *query node ((card(DansPontA)==1) and (card(DansPontB)==1))* (Fichier P1_3.txt)

Pour vérifier la propriété P2, *"Un véhicule qui arrive est certain de passer sur le pont à l'issue d'une durée bornée."*, j'ai utilisé les formules suivantes :

1. *query verbose (AttenteAutorisationA==<.1.> and EG(DansPontA !=<.1.>))* (Fichier P2_1.txt)
2. *query verbose (implies(AttenteAutorisationA==<.1.>, AF(DansPontA==<.1.>))*
   and implies(AttenteAutorisationA==<.2.>, AF(DansPontA==<.2.>))
   and implies(AttenteAutorisationB==<.1.>, AF(DansPontB==<.1.>))
   and implies(AttenteAutorisationB==<.2.>, AF(DansPontB==<.2.>))) (Fichier P2_2.txt)

Quelques formules pour les "autres propriétés triviales attendues" :

1. *query node ((card(VehiculeA) + card(AttenteAutorisationA) + card(DansPontA) + card(FiniA)) != 3)*. Le nombre de véhicules *Vaa* reste constant. (Fichier P3_1.txt)
2. *query node (card(CapacitePont) > 5)*. La capacité du pont $Capa_p$ est constante.(Fichier P3_2.txt)
3. *query node (card(Budget) > 1)*. Il y a au plus 1 jeton dans la place **Budget** à la fois.(Fichier P3_3.txt)
4. *query node (card(Sens) > 1)*. Il y a au plus 1 jeton dans la place **Sens** à la fois.(Fichier P3_4.txt)
5. *query node (card(NaltVehicule) > 1)*. Il y a au plus 1 jeton dans la place **NaltVehicule** à la fois.(Fichier P3_5.txt)

2.8 Question 1.8