

# Chapitre 10

## Annexes

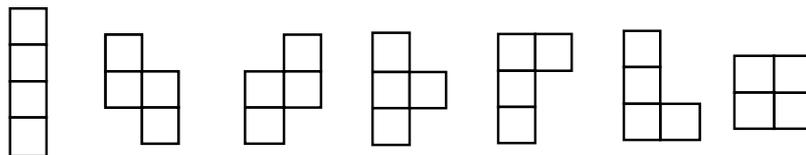
Ce chapitre reprend les corrigé d'exercices placés sur Internet. Cela permet pour l'instant la mise en page correcte sous L<sup>A</sup>T<sub>E</sub>X de ces exercices.

### 1 Le modèle Entités - Associations - Propriétés

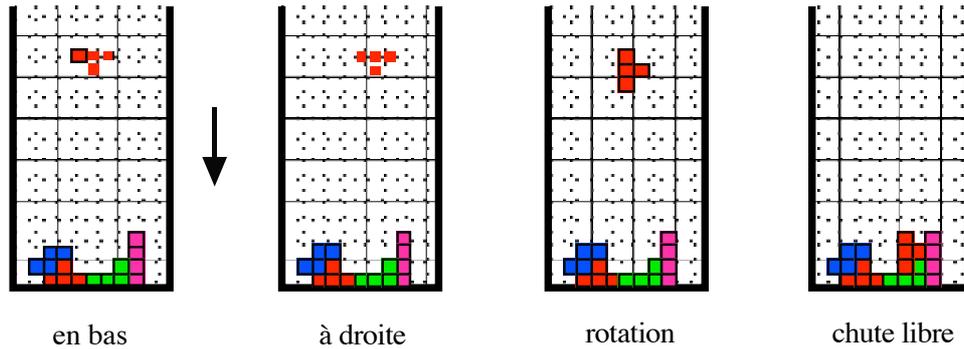
#### Exercice 1.1 (6.12)

*Nous souhaitons modéliser le fameux jeu Tetris pour construire un programme adaptable dans différents langages de programmation et pour différents environnements matériels. Le vocabulaire est propre aux auteurs et ne correspond peut-être pas au vocabulaire original du jeu. Le texte ci-après est donc une interprétation du jeu.*

*Le jeu est constitué de pièces et d'un cadre, appelé fosse, dans lequel tombent les pièces. Une pièce est un assemblage de  $n$  cubes. Dans une première version, le joueur manipulera des pièces de 4 cubes, appelées *tetraminos*. Il existe 7 assemblages de 4 cubes, différents, à une rotation près. Ces assemblages constituent les pièces de base du jeu :*



*Les pièces ont une orientation initiale. Lorsqu'elles tombent, le joueur peut modifier leur orientation ou leur position dans la fosse. La fosse a une largeur de 11 cubes et une hauteur de 20 cubes. Les pièces tombent une par une dans la fosse. Lorsqu'elles ont touché le fond de la fosse, les cubes sont séparés de la pièce et conservent leur position et une nouvelle pièce va tomber. Lorsque les cubes forment une ligne complète, cette ligne disparaît et les cubes situés au dessus descendent d'un cran. La pièce descend automatiquement d'un cran toutes les unités de temps. L'unité de temps est déterminée par la vitesse de descente, un paramètre du jeu. Le joueur peut modifier la position de la pièce au cours de la descente (rotation de 90° ou translation d'une case vers le bas, la gauche ou la droite).*



Les mouvements sont possibles dans la mesure où la pièce reste dans la fosse. La rotation se fait autour d'un axe imaginaire situé au milieu de la pièce. Quel que soit le mouvement, la translation périodique d'un cran vers le bas est appliquée.

Lorsqu'un certain nombre de lignes ont été supprimées, la vitesse est augmentée d'une unité. Le calcul des points se fait de la façon suivante : chaque pièce vaut  $Xp$  points, à chaque descente d'un cran, elle perd un point. Le nombre  $Xp$  de points d'une pièce varie en fonction de sa forme et donc de la difficulté à la placer. Plus la pièce descend, moins elle vaut de points lorsqu'elle touche les cubes empilés en bas de la fosse. Si la suppression d'une ligne correspond à la disparition de tous les cubes de la fosse, un bonus du quadruple des points d'une ligne supprimée est donné au joueur.

Les paramètres initiaux du jeu sont : la vitesse de descente appelé niveau de difficulté, la taille des cubes, la valeur des pièces, la valeur d'une ligne supprimée. Les paramètres modifiables au cours du jeu sont : le niveau de difficulté (uniquement incrémenté), la forme de la prochaine pièce à tomber, les statistiques des pièces tombées. Les dix meilleurs scores sont conservés sur fichier avec le nom des joueurs les ayant réalisés. Ce fichier pourra être détruit en dehors du jeu. Outre les actions sur la pièce courante, le joueur peut faire une pause dans le jeu, arrêter la partie, faire apparaître une image de travail pour cacher le jeu ou changer les options courantes du jeu. On pourra imaginer une extension du jeu à des familles de pièces de 5 cubes, ou à trois dimensions, de même qu'au calcul automatique des pièces possibles d'une famille de  $n$  cubes, et la forme initiale de la pièce qui doit tomber. On pourra aussi paramétrer les dimensions de la fosse avec le nombre de cubes. De même, on pourra construire un jeu de pièces quelconques et une fosse quelconque (en labyrinthe par exemple).

a) Ce texte contenant un certain nombre d'informations inutiles pour la représentation des données, sélectionner les informations en analysant le texte et constituer un dictionnaire des données.

b) Organiser les données sous la forme d'un modèle conceptuel des données.

c) Placer aussi précisément que possible les contraintes et cardinalités sur le modèle conceptuel des données.

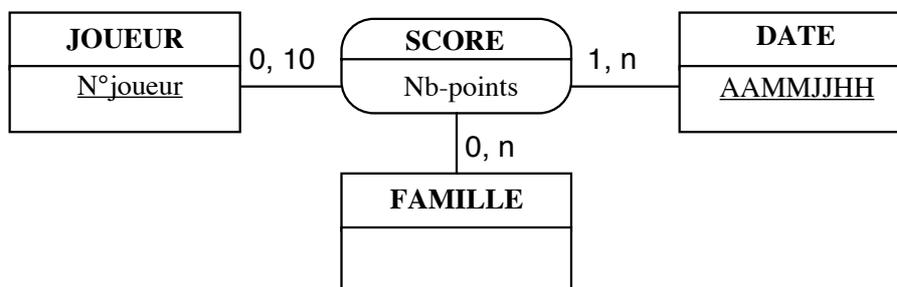
### Solution 1.1

a et b) Décrivons tout d'abord les éléments statiques, joueurs, fosses, modèles de pièces, ... Les joueurs seront décrits par leur Nom et Prénom; un  $N^o$  joueur servira à les identifier.

<b>JOUEUR</b>
N°joueur
Nom
Prénom

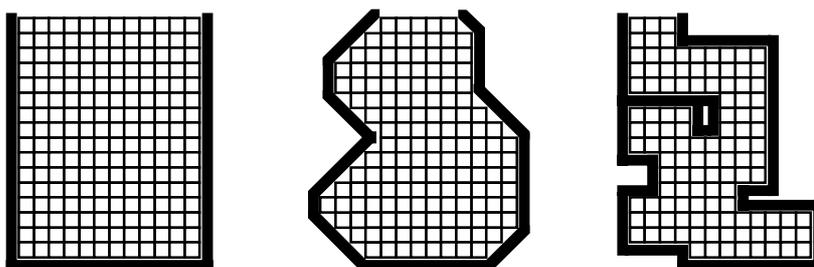
NB : aucune de ces informations ne figurant dans le texte, celui-ci ne faisant référence qu'à des joueurs, sans plus de précision. Nous avons donc fait appel à notre expérience.

Les dix meilleurs scores sont conservés. Un score est relatif au type de jeu (à la famille), car il évolue avec le nombre de cubes par pièces. Nous avons donc besoin de la structure ternaire suivante :

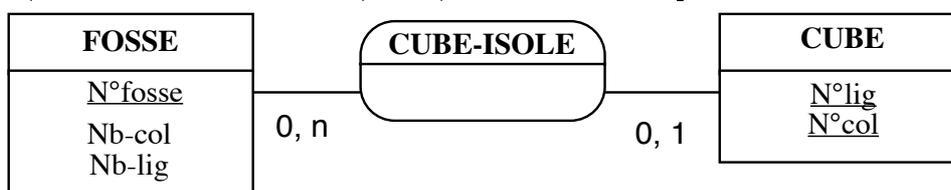


NB : nous supposons qu'en une heure un joueur n'intervient qu'une fois, ce qui signifie qu'il ne joue qu'une partie par heure ou bien qu'il n'obtient pas deux "top 10 scores" par heure. Nous compléterons, d'autre part, l'entité FAMILLE un peu plus loin dans la présentation de notre solution.

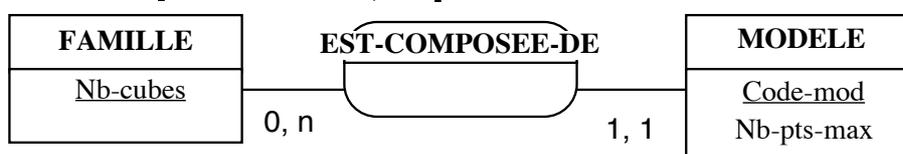
La fosse de jeu est composée de cases. Elles a des dimensions simples (un nb de cases/lignes et un nb de cases/colonnes) s'il s'agit d'une forme rectangulaire, des dimensions complexes s'il s'agit d'une forme non rectangulaire :



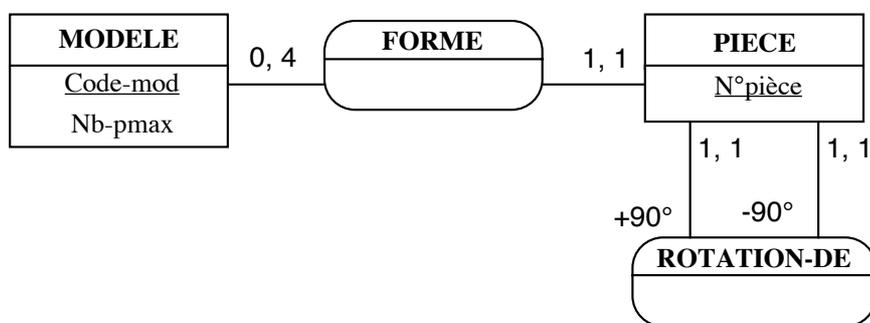
Nous ne développons ici que le cas des fosses simples, comme celles de gauche. La fosse est constituée d'un ensemble de cases, chaque case étant repérée par ses coordonnées ligne, colonne (la ligne 1 correspond au fond de la fosse, la colonne 1 à la colonne la plus à gauche). Ces coordonnées sont calculables. Elles ne seront donc pas représentées. Nous représenterons, par contre, les cubes isolés. Un cube, enfin, est une case occupée dans la fosse.



Les pièces du jeu appartiennent toutes à la même famille, celle des pièces à X cubes. Il peut y avoir plusieurs familles. Le choix de la famille, au début de la partie, entraîne la sélection d'un certain nombre de pièces dans les modèles associés à la famille. A chaque modèle est associé un nombre de points maximum, Nb-pts-max.



Un modèle de pièce (le texte parle d'assemblage et fournit des exemples) correspond à une forme, celle-ci pouvant être décrite par une petite matrice carrée ( $\text{Nb-cubes} \times \text{Nb-cubes}$ ), l'orientation initiale des pièces correspondant à celle de leur modèle. Il peut y avoir plusieurs pièces du même modèle. Un élément essentiel du jeu est la représentation des pièces (modèles, pièces-type, pièces effectives) et des opérations associées. La solution est de fournir quatre pièces-type par modèle, une pour chaque rotation de  $90^\circ$ . Certains modèles en nécessitent moins (le cube à une seule pièce, par exemple).



Les pièces effectives sont dans la fosse. Elles possèdent deux informations supplémentaires, position et nombre de points. Il est important de noter que l'on a pas besoin de disposer des positions absolues de chaque pièce jouée mais de son organisation (ses cubes, référencés par rapport à la matrice de base, son modèle) et de la position du coin supérieur gauche de ce modèle dans la fosse.

L'utilisateur peut aussi faire descendre doucement la pièce, la déplacer sur la gauche ou sur la droite, si la place est disponible. Il peut enfin la "lâcher" et la faire tomber au fond de la fosse (le nombre de points de la pièce est alors diminué d'une unité seulement). Tout ceci est toutefois calculable. Il est donc inutile de le mémoriser.

Le pas de descente (le cran) correspond à la vitesse de descente du texte. C'est le niveau de difficulté. Il est initialisé à une certaine valeur au début de la partie et est augmenté d'une unité toutes les N lignes supprimées (N est un des paramètres de la partie). Ces paramètres seront stockés dans une entité "à part".

Une réponse à cet exercice est le MCD de la figure 72.

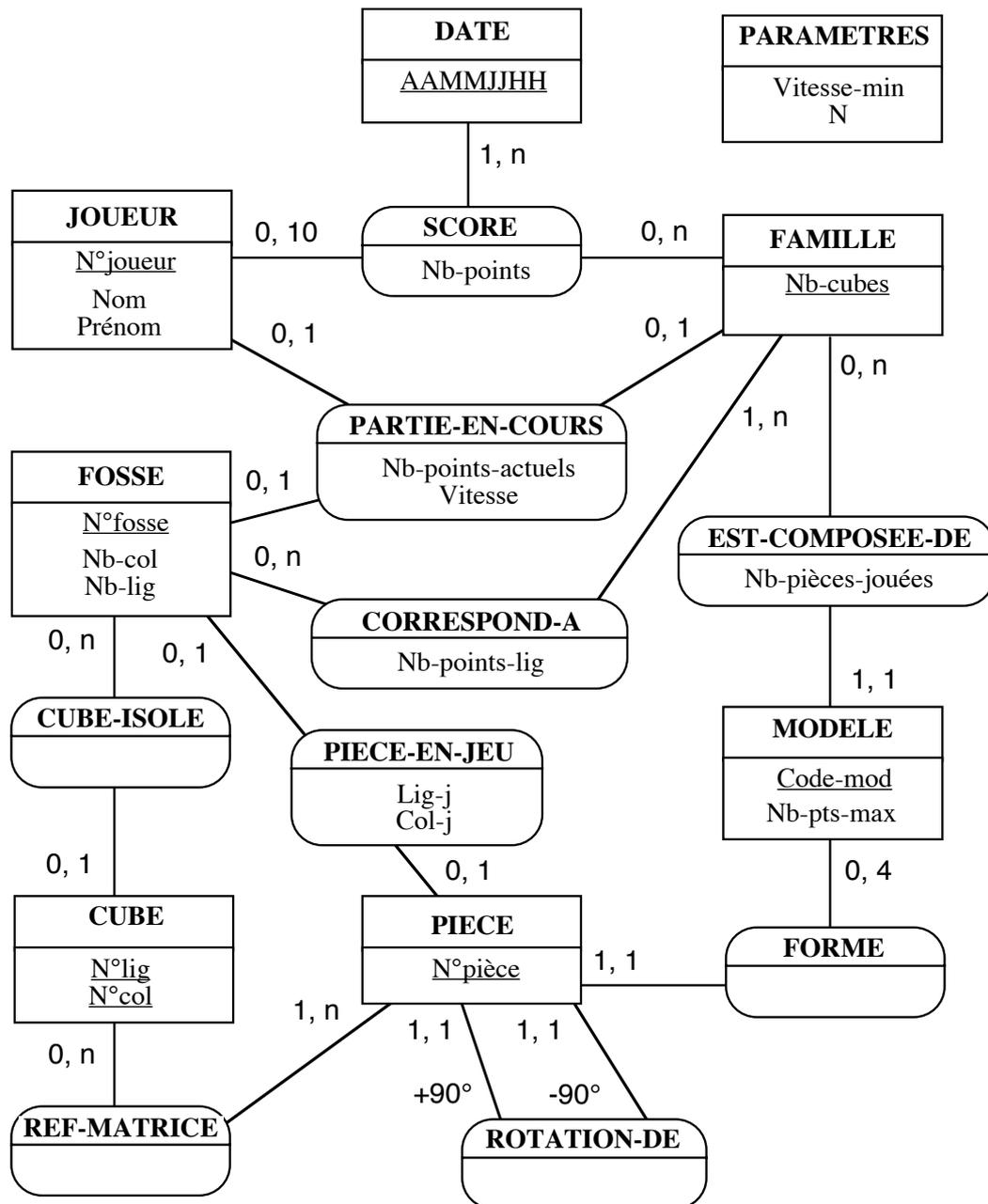
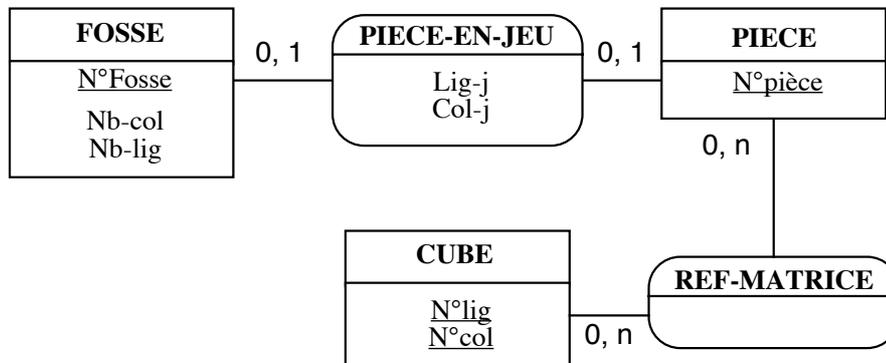


Figure 72 : MDC du cas Tetris

c) Pour placer précisément les contraintes, reprenons la description. On distingue la partie en cours (**PARTIE-EN-COURS**) des autres parties déjà jouées (**SCORE**). Il y a au plus une partie en cours et donc un seul joueur qui joue. On mémorise le nombre de points de la partie en cours. La partie en cours définit la famille des pièces utilisées et la fosse. A chaque famille correspond un ensemble de fosses acceptables (**CORRESPOND-A**). La fosse de la partie en cours doit être une fosse acceptable pour la famille (contrainte C1). Nous reviendrons sur le jeu ensuite.

Décrivons les pièces du jeu. Chaque famille propose des modèles de pièces (**EST-COMPOSEE-DE**). Les pièces qui tombent dans la fosse (une seule à la fois) sont des exemplaires de ces modèles de pièces, à une ou plusieurs rotations près. Comme il est difficile de calculer automatiquement la rotation d'une pièce quelconque, nous avons défini une forme de pièce par rotation, appelée simplement **PIECE**. Chaque modèle de pièce propose quatre formes (**FORME**) de pièces au maximum (une par rotation). On indique la forme obtenue par rotation à partir d'une pièce par une association en boucle **ROTATION-DE** (une loi de composition interne au sens mathématique).

Toute pièce est rotation d'une autre pièce du même modèle (contrainte C2). Chaque pièce est unique, elle est une forme d'un modèle de pièce d'une famille. Chaque pièce est formée de cubes, définis en position relative dans une matrice (REF-MATRICE) comme l'indiquent les schémas précédents de pièce. En ce sens, l'entité CUBE désigne une cellule dans la matrice de la pièce. Rappelons que les dimensions de la matrice d'une pièce sont définies par le nombre de cubes de la famille de la pièce (Nb-cubes x Nb-cubes) (contrainte C3).



La fosse est définie par des dimensions rectangulaires, nous ignorons les formes exotiques. La fosse de la partie en cours contient des cubes isolés et éventuellement une pièce en jeu. Les cubes isolés ont des coordonnées références conformes aux dimensions de la fosse (contrainte C4). Les coordonnées des cubes de la pièce en jeu sont conformes aux dimensions de la fosse (contrainte C5) et distinctes de celles des cubes isolés de la fosse. Il y a une seule fosse, une seule pièce et plusieurs cubes, ce qui explique les cardinalités (0, 1) des associations PIECE-EN-JEU et CUBE-ISOLE. Une seule fosse contient des cubes et une pièce, c'est la fosse de la partie en cours. Les autres ne contiennent ni cubes ni pièce (contrainte C6).

NB : attention, il y a surcharge entre la notion de cube d'une pièce et celle de cube isolé, tombé dans la fosse. Ces deux types de cubes sont sémantiquement totalement distincts. Nous avons utilisé une seule entité, car il y a les mêmes propriétés. On ne peut poser de contrainte sur ce point.

NB : la rotation d'une pièce dans le jeu ne modifie pas ses coordonnées (N°lig et N°col sont invariants) mais, implicitement, celles de ses cubes. La contrainte C5 est à re-vérifier après chaque rotation.

Certains éléments de la partie en cours sont mémorisés à différents endroits. Le nombre de points accordés lors de la suppression d'une ligne est supposé être fonction de la famille et de la fosse (CORRESPOND-A). Les statistiques concernent le nombre de pièces jouées de chaque modèle, nous les avons rangées dans l'association EST-COMPOSEE-DE. Nous mémorisons les vitesses min et max dans une entité PARAMETRES. Nous ne précisons pas le mode de calcul de la prochaine pièce à jouer dans la fosse (fonction aléatoire sur la famille des pièces? ?).

Les contraintes sont donc (forme textuelle) :

- C1 : la fosse de la partie en cours est l'une des fosses correspondant à la famille de la partie en cours.
- C2 : la rotation d'une pièce p est une pièce du même modèle.
- C3 : soit Nb cubes le nombre de cubes de la pièce, défini dans sa famille. La cardinalité exacte pour une pièce donnée dans l'association REF-MATRICE est Nb-cubes et chaque cube de la pièce a des coordonnées références comprises entre 1 et Nb-cubes (N°lig, N°col appartiennent à [1..4] pour un *tetraminos*).
- C4 : les cubes tombés dans la fosse ont des coordonnées conformes aux dimensions de la fosse.
- C5 : les coordonnées de la pièce en jeu sont conformes aux dimensions de la fosse. Les coordonnées proposées sont celles de la cellule en haut à droite de la matrice. On doit

vérifier que les cubes de la pièce (cellules occupées par la matrice), compte-tenu de la translation dans la fosse, sont conformes aux dimensions de la fosse et distincts des cubes isolés.

- C6 : une seule fosse contient des cubes et une pièce, c'est la fosse de la partie en cours. Les autres ne contiennent ni cubes, ni pièce.

Pour les modéliser, posons les entités et les associations sous forme de relations. Nous utiliserons les opérations de projection, de sélection et d'image relationnelle de l'exercice 3.4 de ce chapitre. Nous utiliserons les identifiants des entités et des relations, ainsi que des fonctions (notation pointée  $id.att$ ) pour atteindre les attributs non clés à partir des identifiants. Comme dans les autres exercices, nous confondrons entité et identifiant. Pour les associations, nous définissons les relations ou fonctions suivantes :

SCORE (JOUEUR, DATE, FAMILLE)	
PARTIE-EN-COURS (JOUEUR, FOSSE, FAMILLE)	
CORRESPOND-A (FOSSE, FAMILLE)	-c'est une relation binaire-
EST-COMPOSEE-DE (MODELE, FAMILLE)	-c'est une fonction totale-
FORME (PIECE, MODELE)	-c'est une fonction totale-
ROTATION-DE (PIECE, PIECE)	-fonction $+90^\circ$ -
PIECE-EN-JEU (PIECE, FOSSE)	-c'est une fonction partielle-
REF-MATRICE (PIECE, CUBE)	
CUBE-ISOLE (CUBE, FOSSE)	-c'est une fonction partielle-

Les contraintes sont traduites sous forme de prédicats :

$$\begin{aligned}
 C1 &: \forall(j, fo, fa) \in PARTIE-EN-COURS \bullet (fo, fa) \in CORRESPOND-A \\
 C2 &: \forall(p1, p2) \in ROTATION-DE \bullet FORME(p1) = FORME(p2) \\
 C3 &: \forall p \in PIECE, \exists nb = EST-COMPOSEE-DE(FORME(p)).Nb-cubes \bullet \\
 &\quad cardinal(REF-MATRICE(\!p)) = nb \wedge \\
 &\quad [\forall c \in REF-MATRICE(\!p) \bullet (c.N^o lig \leq nb) \wedge (c.N^o col \leq nb)] \\
 C4 &: \forall(c, fo) \in CUBE-ISOLE \bullet (c.N^o lig \leq fo.Nb-lig) \wedge (c.N^o col \leq fo.Nb-col) \\
 C5 &: \forall(p, fo) \in PIECE-EN-JEU, \exists li = (p, fo).Lig-j, \exists co = (p, fo).Col-j \bullet \\
 &\quad \forall c \in REF-MATRICE(\!p) \bullet \\
 &\quad (c.N^o lig + li \leq fo.Nb-lig) \wedge (c.N^o col + co \leq fo.Nb-col) \wedge \\
 &\quad // conforme à la fosse \\
 &\quad \neg[\exists(c1, fo) \in CUBE-ISOLE \bullet \\
 &\quad (c.N^o lig + li = c1.N^o lig) \wedge (c.N^o col + co = c1.N^o col)] \\
 &\quad // distincts des cubes de la fosse \\
 C6 &: \exists! fo \in FOSSE \bullet fo \in PROJECTION(PARTIE-EN-COURS, FOSSE) \wedge \\
 &\quad fo \in PROJECTION(PIECE-EN-JEU, FOSSE) \wedge \\
 &\quad fo \in PROJECTION(CUBE-ISOLE, FOSSE)
 \end{aligned}$$

Nous avons supposé, dans ce qui précède, que  $nb$ ,  $N^o col$  et  $N^o lig$  sont des entiers naturels positifs ou nuls. Nous avons volontairement "allégé" la notation et confondu entités et identifiants d'entités.

## 2 Les réseaux de PETRI

### Exercice 2.1 (8.11)

*NB : cet exercice est inspiré de [PX95], page 187.*

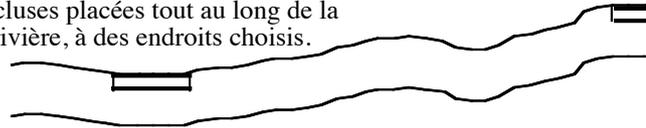
*Le texte a été revu de façon à rappeler aux étudiants le principe de fonctionnement des écluses et de leur éviter une perte de temps due au contexte. Il a également été rebâti de façon à mettre en évidence le besoin, dans certains cas, de RdP colorés. Un cours d'eau est rendu*

navigable par la présence d'une écluse. Des péniches désirent passer d'un bord à l'autre. Nous considérons que les bateaux vont de A en B conformément au schéma ci-dessous.

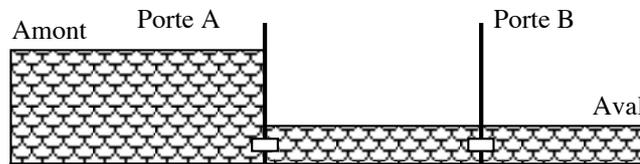
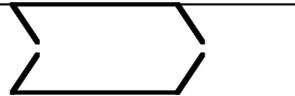


Les écluses permettent aux bateaux d'emprunter plus facilement les rivières en créant des paliers.

Ceci est possible grâce à un ensemble d'écluses placées tout au long de la rivière, à des endroits choisis.



L'écluse comporte deux séries de portes, manoeuvrées de la rive par un éclusier.



Chaque porte est dotée d'une vanne que l'on ouvre pour laisser l'eau entrer (porte A) ou sortir (porte B).

A l'arrivée d'un bateau, le niveau d'eau dans le sas est généralement égal à celui de B, mais il peut aussi que des mesures aient été prises afin de le régulariser par rapport à celui de A. Nous considérons donc que le niveau d'eau du sas est au départ celui de A ou de B. L'une des deux portes est fermée, nous ne savons pas laquelle. Il n'y a qu'un seul bateau.

Le principe de fonctionnement d'une écluse est le suivant : Un bateau situé en A, désirant parvenir en B, doit franchir le sas. Le franchissement du sas ne peut se faire que si les niveaux d'eau en A et dans le sas sont identiques. C'est l'ouverture des vannes en A qui permet la régularisation des niveaux. Lorsque le niveau du sas atteint celui en A (après ouverture des vannes), l'ouverture d'une porte en A permet le passage du bateau. Lorsque le bateau est entièrement dans le sas, la porte en A est refermée. Le bateau tente maintenant de passer en B. Pour cela, les niveaux du sas et en B sont régularisés de la même façon que précédemment : ouverture de vannes en B, puis, lorsque le niveau du sas aura atteint celui de B, ouverture d'une porte en B permettant le passage du bateau.

On supposera que, lors de l'arrivée du bateau, toutes les vannes sont fermées. D'autre part, il s'agit d'une écluse ne pouvant contenir qu'un seul bateau. Une porte, d'autre part, ne se ferme qu'après que sa vanne soit fermée.

a) Modéliser cette réalité à l'aide d'un réseau de PETRI, en décrivant le contenu des transitions. On supposera que le bateau trouve devant lui la porte A fermée.

b) Prendre en compte maintenant le fait que le bateau peut se trouver devant une porte A fermée ou une porte A ouverte (B est alors fermée).

c) Refaire l'exercice en positionnant le bateau en aval (il veut aller en amont).

d) Peut-on réunir les deux réseaux obtenus en réponse aux questions b et c en un seul ?

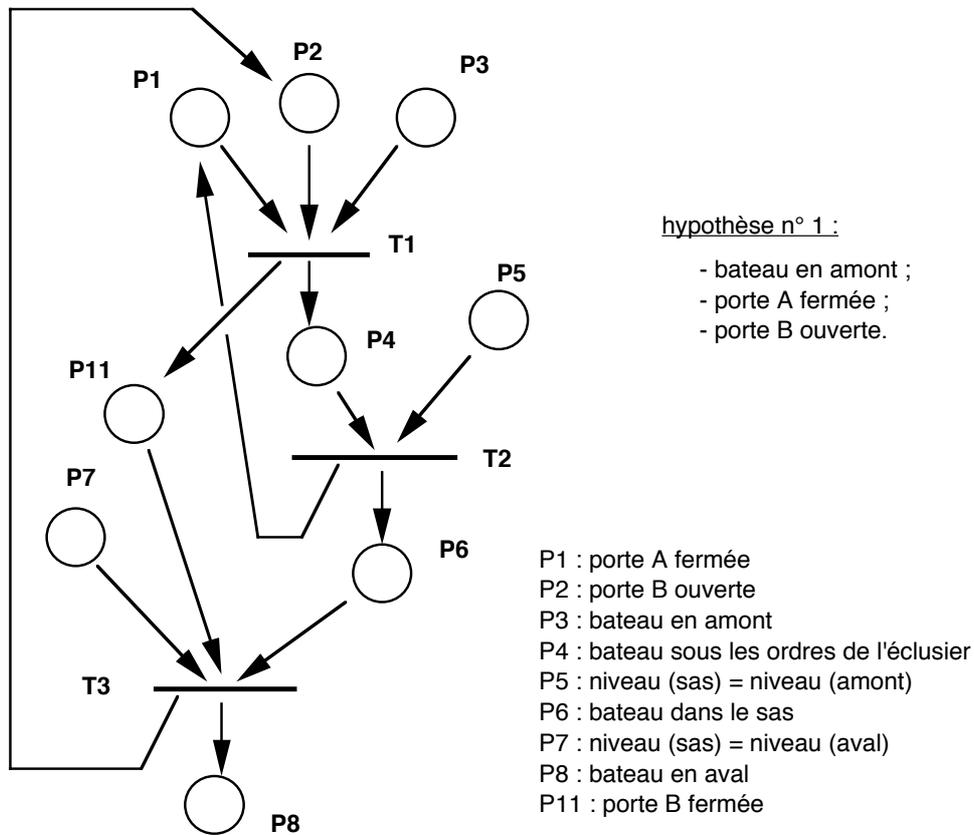
e) Utiliser les réseaux colorés pour résoudre le problème posé par la question précédente.

## Solution 2.1

Cet exercice a deux objectifs, un "public" et un caché. Le premier est de permettre aux étudiants de mettre en pratique sur un exemple simple (bien que dans un contexte qui n'est pas celui classique des systèmes d'information) les concepts vus en cours. Le second est d'amener doucement les étudiants à ressentir le besoin d'avoir recours à des extensions. Si nous développons "tranquillement" la solution hypothèse après hypothèse, inévitablement il arrive un moment où la question de rationalisation se pose. Les ensembles de transition sont voisins les uns des autres ; il est tentant de regrouper les deux schémas en un seul (sans perte d'information, bien sûr) ... et ceci n'est possible qu'avec des réseaux colorés. Ce sujet est un bon exercice pour ressentir les limites d'une modélisation à l'aide de réseaux de PETRI.

Nous choisissons le principe de modélisation suivant : les actions seront représentées par des transitions, les signaux (situations, événements) par des places.

a) Traitons, pour commencer, le cas suivant : le bateau se présente en amont, la porte A est fermée, la porte B est ouverte. Il y a trois transitions T1, T2, T3 que l'on peut visualiser sur le RdP suivant.



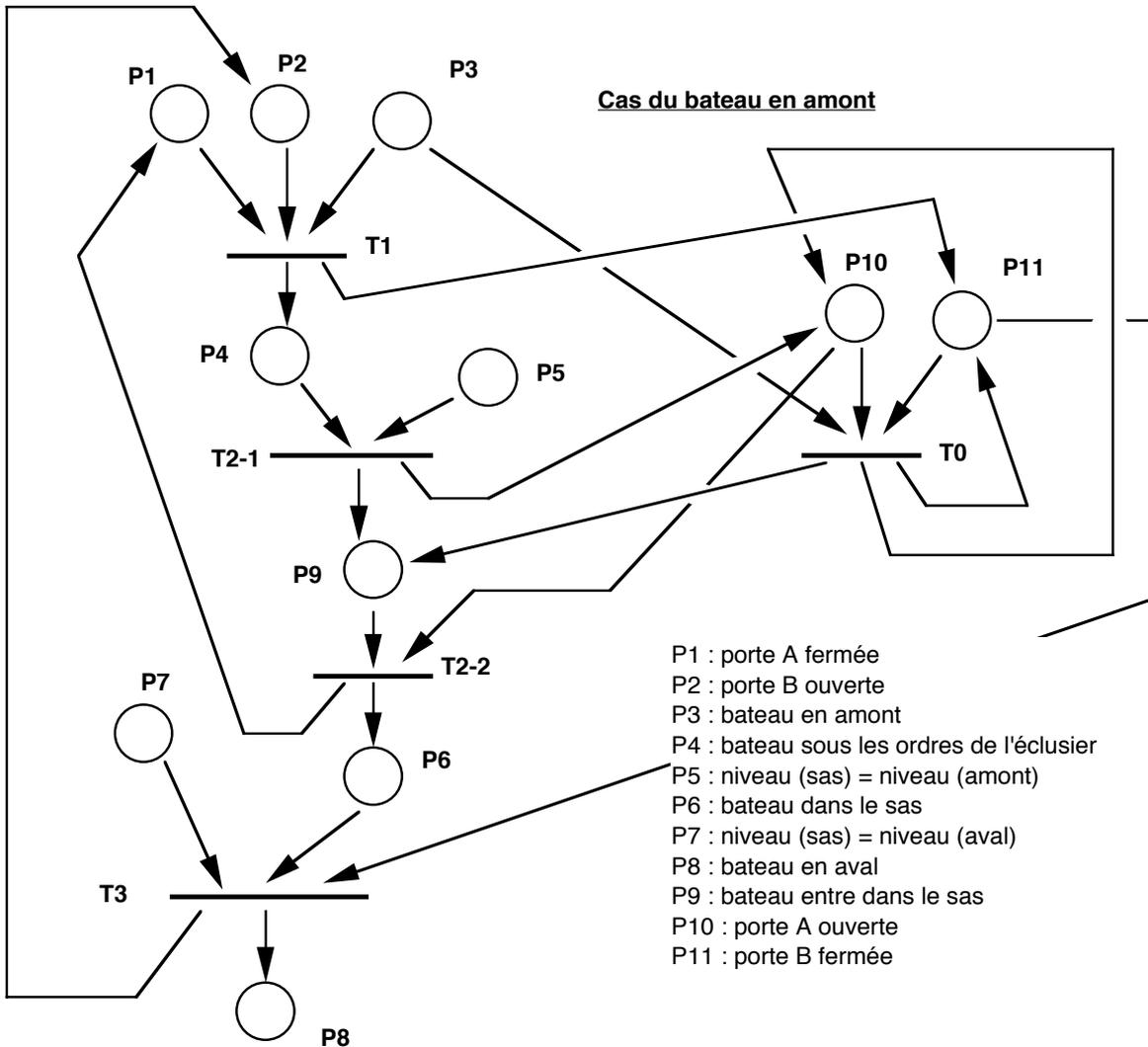
Nom de la transition    Contenu de la transition

T1	fermeture porte B
	ouverture vanne A
T2	fermeture vanne A
	ouverture porte A
	entrée du bateau dans le sas
	fermeture porte A
T3	ouverture vanne B
	fermeture vanne B
	sortie du bateau du sas

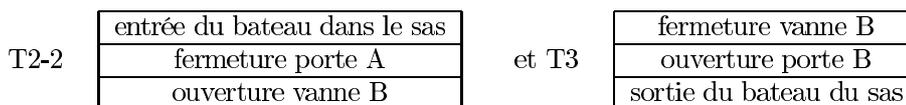
Les places P1 et P2, positionnées en entrée de la transition T1, servent à mettre en évidence l'hypothèse de départ. Nous aurions pu matérialiser le fait que l'entrée du bateau dans le sas

prend du temps en "cassant" en trois morceaux la transition T2 et en ajoutant une place permettant d'exprimer le fait que le bateau est tout entier entré dans ce sas. Nous ne l'avons pas fait par souci de simplification.

b) Envisageons maintenant le cas où le bateau se présente en amont, la porte A étant ouverte et la B fermée. Les transitions à exécuter sont au nombre de deux. Le premier paragraphe précédent. La première correspond à une partie de T2 (appelons-la T2-2), la seconde à T3. Nous pouvons donc réutiliser une partie du RdP précédent :

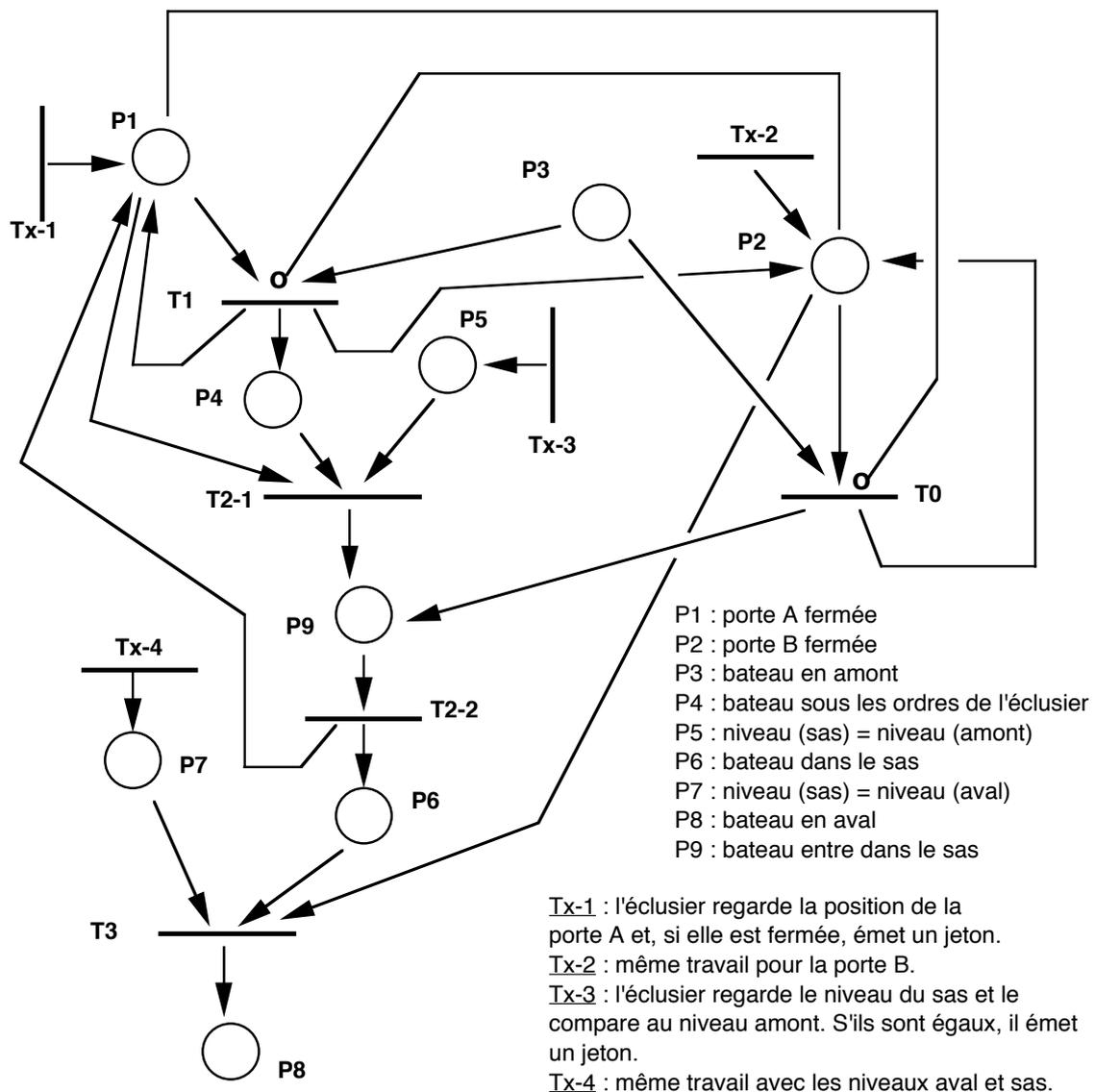


La transition T0 est vide. Il n'y a en effet rien à faire sur les portes et/ou les vannes. Le bateau peut entrer dans le sas. Cette transition est de fait un aiguillage.



Le traitement des portes (A ouverte, A fermée, B ouverte, B fermée) est dans ce RdP assez approximatif. Pour améliorer cette partie du graphe, nous allons utiliser une des deux extensions, celle appelée arcs inhibiteurs. Le RdP, après modifications, est le suivant :

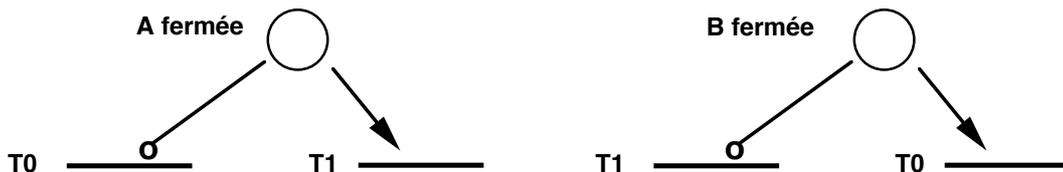
**Cas du bateau en amont**



Détaillons les modifications. Les places P1 et P10 sont "liées" par une loi simple : quand P1 contient un jeton, P10 ne peut en avoir, et réciproquement. Nous avons en effet l'équation<sup>1</sup> :

$$P1 \text{ ouverte} = \text{non} (P1 \text{ fermée})$$

A partir de ce constat, une seule place suffit pour P1 et P10. Nous tenons le même raisonnement pour P2 et P11. Ouvrir la porte consiste y prendre un jeton dans la place. Fermer la porte consiste y mettre un jeton dans la place.

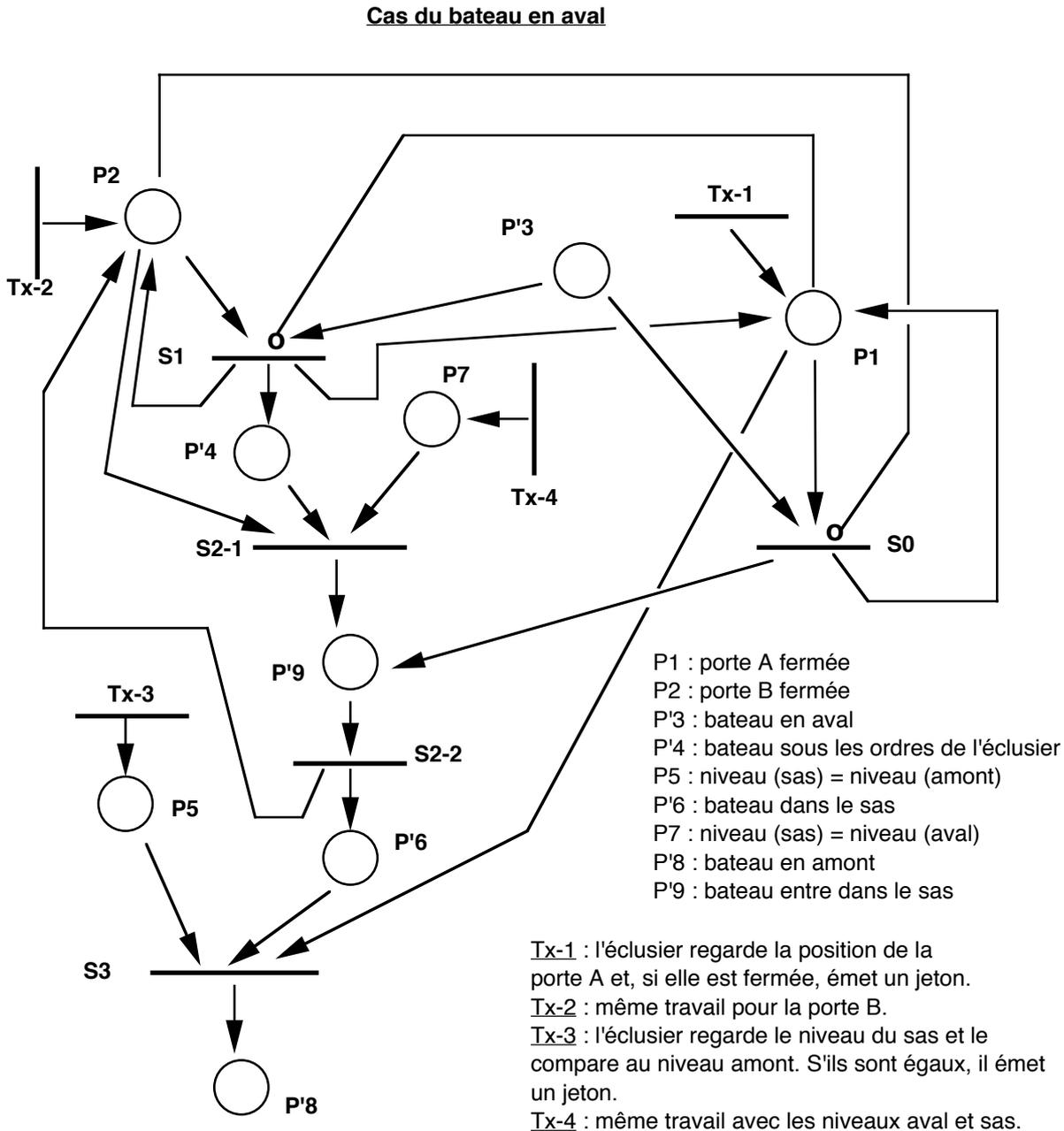


Nous avons ensuite matérialisé l'action de l'éclusier en faisant apparaître des transitions-source. Ce recours introduit dans le schéma des places dont le mode d'alimentation est inconnu,

<sup>1</sup>Nous excluons donc de notre modélisation le cas où les portes seraient entrouvertes, ni fermées, ni ouvertes.

aléatoire. Mais, à moins de modéliser les interactions entre l'éclusier et le système à un niveau supérieur, il n'y a pas d'autres solutions. Les transitions Tx-1 et Tx-2 servent au marquage initial des portes. Les transitions Tx-3 et Tx-4 sont des actions ponctuelles de l'éclusier pour la vérification des niveaux du sas. Nous ne les avons pas traitées de la même manière que les portes pour bien mettre en évidence la durée des opérations.

c) Après avoir traité des cas où le bateau est en amont, examinons les deux cas où le bateau vient de l'aval. Le RdP correspondant au bateau en aval est le suivant :



Les transitions à exécuter sont similaires aux précédentes, seul leur ordre d'exécution diffère. Le premier cas traité correspond à l'arrivée du bateau en aval, porte B fermée, porte A ouverte. Il y a trois transitions à déclencher :

Nom de la transition		Contenu de la transition
S1		fermeture porte A
		ouverture vanne B
S2	S2-1	fermeture vanne B
		ouverture porte B
	S2-2	entrée du bateau dans le sas
		fermeture porte B
S3		ouverture vanne A
		fermeture porte A
		sortie du bateau du sas

Le second est celui dans lequel la porte A est fermée, la porte B ouverte. Le bateau peut pénétrer directement dans le sas, ce qui se traduit par les transitions suivantes : S2-2 et S3.

entre ces deux transitions et une partie des précédentes.

d) Il est maintenant temps de prendre un peu de recul et de procéder à une comparaison des deux schémas obtenus. Il apparaît clairement que certaines des transitions  $S_i$  et  $T_i$  sont identiques (quoiqu'il ne s'agisse pas de celles auxquelles on pourrait penser). Pour mieux les mettre en évidence, nommons les ensembles d'actions :

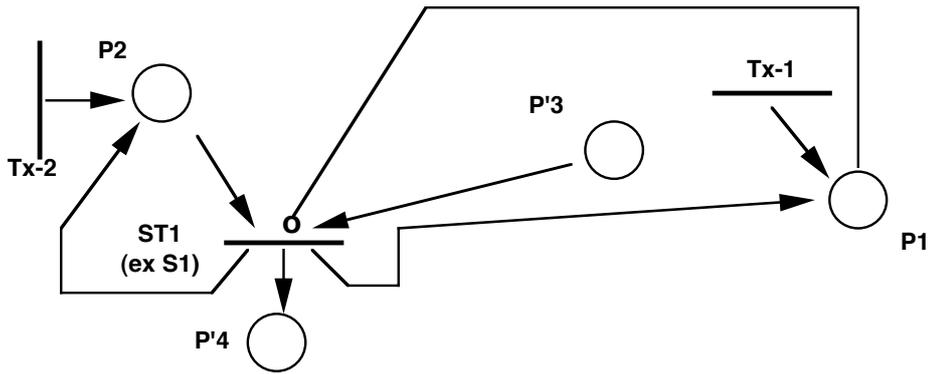
Nom de la transition	Contenu de la transition
ST1	fermeture porte A
	ouverture vanne B
ST2	fermeture vanne B
	ouverture porte B
ST3	entrée (de type 1) du bateau dans le sas
ST4	entrée (de type 2) du bateau dans le sas
ST5	fermeture porte B
	ouverture vanne A
ST6	fermeture vanne A
	ouverture porte A
ST7	sortie (de type 1) du bateau du sas
ST8	sortie (de type 2) du bateau du sas

Nous avons distingué deux types de mouvements du bateau, l'un se déroulant dans le sens du courant (type 1), l'autre dans le sens inverse (type 2). Compte-tenu de ce qui précède, nous pouvons définir les différentes transitions  $S_i$ ,  $T_j$  :

T1 = <ST5>	S1 = <ST1>
T2-1 = <ST6>	S2-1 = <ST2>
T2-2 = <ST3, ST1>	S2-2 = <ST4, ST5>
T2 = <ST6, ST3, ST1>	S2 = <ST2, ST4, ST5>
T3 = <ST2, ST7>	S3 = <ST6, ST8>

Il y a donc bien des actions communes aux deux situations envisagées (bateau en amont, bateau en aval). L'idée est donc de tenter de regrouper les deux schémas en un seul, avec pour transitions les ensembles d'actions  $ST_j$  évoqués plus haut ... et nous n'y arrivons pas ! Voyons pourquoi sur un extrait des schémas, concernant la transition ST1 :

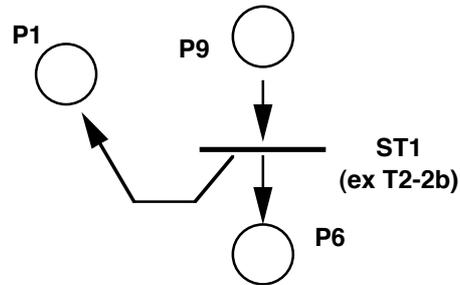
**Extrait du schéma aval**



Les conditions de déclenchement de la transition ST1 ne sont pas les mêmes quand le bateau se présente en amont et en aval. Il n'est donc pas possible de regrouper les deux transitions S1 et T2-2b (la logique sous-jacente dans un RdP étant, fondamentalement, le ET). Il nous faut employer autre chose.

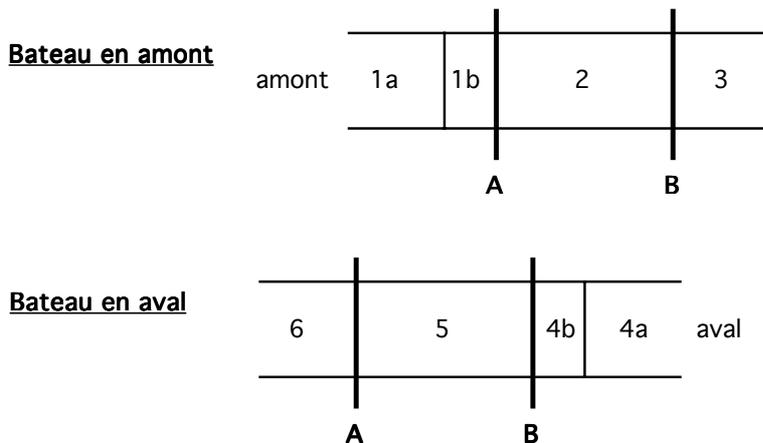
**Extrait du schéma amont**

- P1 : porte A fermée
  - P2 : porte B fermée
  - P'3 : bateau en aval
  - P'4 : bateau sous les ordres de l'éclusier
  - P6 : bateau dans le sas
  - P9 : bateau entre dans le sas
- Tx-1 : l'éclusier regarde la position de la porte A et, si elle est fermée, émet un jeton.  
Tx-2 : même travail pour la porte B.



En conclusion, on peut dire que le partage des places est possible mais le partage de transitions ne l'est pas.

e) Auparavant, nous allons partager l'écluse et ses environs en plusieurs zones et nous catégoriserons le bateau selon la zone dans laquelle il se trouve :



Les mouvements du bateau se résument alors à des changements de zone. Nous avons, en outre, besoin de préciser les différents états des deux portes. Compte-tenu des deux approches possibles du bateau, nous aurons recours aux valeurs suivantes :

Valeur	Signification	Valeur	Signification
<i>EO</i>	porte éloignée ouverte	<i>PO</i>	porte proche ouverte
<i>EF</i>	porte éloignée fermée	<i>PF</i>	porte proche fermée

Chacune des deux portes pourra prendre une de ces valeurs. Nous avons maintenant tout ce qu'il faut pour modéliser le fonctionnement de cette écluse. Nous allons employer des réseaux colorés, avec des transitions colorées, composées comme suit :

Transitions			Valeurs				
nouveau nom	contenu	ancien nom	Niv 1	Niv 2	Porte A	Porte B	Bateau
ST0		S0			EF	PO	4a
					EF	PO	4b
		T0			PO	EF	1a
					PO	EF	1b
ST1	Fermeture porte A Ouverture vanne B	S1			EO	PF	4a
					EF	PF	4b
		T2-2b			PO	EF	2
					PF	EF	2
ST2	Fermeture vanne B Ouverture porte B	S2-1		1	EF	PF	4b
					EF	PO	4b
		T3a		1	PF	EF	2
					PF	EO	2
ST3	Entrée bateau dans le sas (amont → sas)	T2-2a			PO	EF	1b
					PO	EF	2
ST4	Entrée bateau dans le sas (aval → sas)	S2-2a			EF	PO	4b
					EF	PO	5
ST5	Fermeture porte B Ouverture vanne A	S2-2b			EF	PO	5
					EF	PF	5
		T1			PF	EO	1a
					PF	EF	1b
ST6	Fermeture vanne A Ouverture porte A	S3a	1		EF	PF	5
					EO	PF	5
		T2-1	1		PF	EF	1b
					PO	EF	1b
ST7	Sortie bateau du sas (sas → aval)	T3b			PF	EO	2
					PF	EO	3
ST8	Sortie bateau du sas (sas → amont)	S3b			EO	PF	5
					EO	PF	6

Afin d'illustrer la manière de lire ce tableau, prenons un exemple. La transition ST6, dans sa version S3a, est mise à feu si la place Niv1 contient un jeton, si la porte A contient un jeton EF, la porte B un jeton PF et la place Bateau un jeton 5. Le résultat de la mise à feu est le suivant : la place Niv1 perd son jeton, la porte A "vaut" EO, la porte B et la place Bateau restent inchangées. Le réseau correspondant est présenté ci-après.

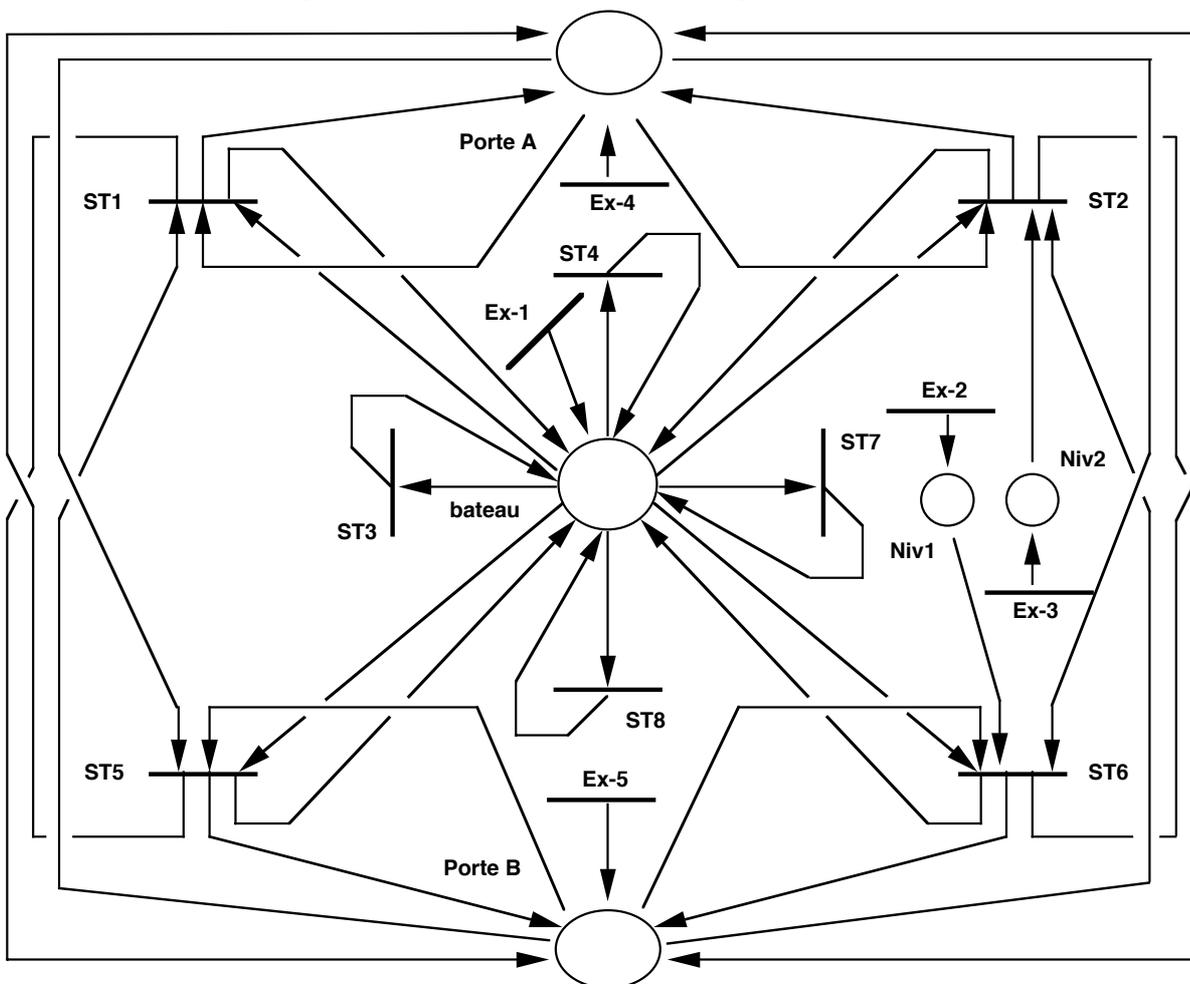
Il doit encore être affiné :

1. d'un point de vue structurel, les transitions ST3, ST4, ST7 et ST8 peuvent être unifiées en une seule qui regroupera toutes les transitions colorées ;
2. les deux places Niv1 et Niv2 peuvent, elles-aussi, être regroupées en une seule, avec des jetons colorés (prenant les valeurs 1 pour signifier que le niveau du sas est égal à celui de l'amont ou 2 si le niveau du sas est égal à celui de l'aval) ;
3. il faut ajouter la transition ST0. l'état des portes.

De même, il est possible de reprendre le rôle de l'éclusier, de le modéliser par un RdP et d'étudier l'interaction entre les deux réseaux. Actuellement, ce rôle est concrétisé dans le RdP grâce aux cinq transitions-source, nommées Ex-1, Ex-2, Ex-3, Ex-4 et Ex-5. Celles-ci ont la signification suivante :

Transition	Détail
Ex-1	Bateau en approche
Ex-2	niveau (sas) = niveau (amont)
Ex-3	niveau (sas) = niveau (aval)
Ex-4	Typage initial de la porte A
Ex-5	Typage initial de la porte B

Si le bateau se présente en amont, un jeton 1a est placé dans la place Bateau. S'il se présente en aval, c'est un jeton 4a qui est mis dans cette place. Si le niveau d'eau dans le sas est égal au niveau de la rivière en amont, un jeton est placé dans la place Niv1. Si le niveau d'eau dans le sas est égal à celui de la rivière en aval, ce jeton est placé dans la place Niv2.



On pourrait, enfin, et dans le cadre de la modélisation du rôle de cet éclusier, prendre en compte les cas "interdits", c'est-à-dire l'arrivée simultanée de deux bateaux, un en amont et un en aval, ou bien encore les configurations de portes "ouverte-ouverte" ou "fermée-fermée". Nous le ferons pas ici, faute de place et laissons au lecteur de le faire.

### Remarques finales

Avant de passer à un autre exercice, quelques remarques :

1. Dans la question a, l'état des portes après le passage du bateau est le même que celui avant son passage. Le système décrit est donc non déformant.
2. L'état initial des portes peut être quelconque dans la version avec arcs inhibiteurs, tandis que la version de départ n'autorise que certaines combinaisons. Ainsi par exemple, les deux portes ne peuvent être ouvertes -ou fermées- en même temps dans l'état initial.

3. Les places P1 et P2 doivent être 1-bornées, sinon le système risque d'être dans un état incohérent.
4. Les transitions Tx-1 et Tx-2 sont facultatives et peuvent poser des problèmes dans l'étude des propriétés. Elles sont en effet tirables à chaque instant. Un marquage initial quelconque (au plus un jeton dans chaque place) des places P1 et P2 résoud simplement, mais arbitrairement, le problème de la borne 1 et celui des transitions non souhaitées.
5. Les places P7 et P9 doivent être 1-bornées, sinon le système peut être dans un état incohérent. Il faut noter qu'on ne peut pas utiliser une seule place avec arc inhibiteur car le vidage, ou le remplissage, ne sont pas instantanés. En supposant les actions instantanées (ce qui est faux dans le RdP, car rien ne se passe entre temps), on peut traiter le problème du niveau dans le sas de la même manière que les portes, avec un marquage initial et des arcs inhibiteurs. Cela résoud le problème d'exclusivité, celui de la borne 1 et celui des transitions non souhaitées (car Tx-3 et Tx-4 sont tirables à chaque instant).

Une question subsidiaire pour finir : le réseau doit-il être k-borné ? Peut-on, par exemple, avoir plusieurs bateaux en amont en même temps ? Si la solution présentée ne le permet pas, modifier le RdP pour ce faire.

### 3 La modélisation des traitements avec Merise

#### Exercice 3.1 (8.7)

*La société LOCATION SA est une PME ayant pour objet la location de véhicules automobiles. A l'arrivée du client à l'accueil, l'hôtesse l'interroge puis lui propose un choix de véhicules disponibles. Le client choisit et fournit les éléments d'identification et on établit le contrat de location. S'il y a lieu, le client verse un chèque de caution, le montant de celle-ci étant inscrit sur le contrat. Les clés du véhicule sont remises, avec un exemplaire du contrat, au client. Ce dernier prend possession du véhicule auprès du responsable du parc véhicules. A cette occasion, un état du véhicule est rédigé par le responsable et signé par le client. Le kilométrage du véhicule est relevé et inscrit sur le document. Le fichier Véhicules est alors mis à jour. Au retour du client, un contrôle est effectué sur le véhicule (kilométrage effectué, niveau d'essence, état). Le contrat (deuxième exemplaire) est recherché, puis on procède au décompte définitif et on encaisse le montant de la location. La caution est rendue au client si le contrôle a donné satisfaction. Le client doit en principe effectuer le plein du réservoir avant de rendre le véhicule. Si ce n'est pas fait, le plein est fait par la société et facturé au client.*

- a) Dresser la liste des événements et des résultats.
- b) Donner la liste des opérations et les règles de synchronisation.
- c) Donner le MCT Merise correspondant à ce mode de fonctionnement.

#### Solution 3.1

Nous allons procéder, dans cet exercice, à une analyse de texte et en extraire, comme il est demandé, événements, résultats, procédures, ... Le texte est manifestement constitué de deux paragraphes, le premier traitant de ce qui se passe avant la location, le second du retour du véhicule. Nous allons donc présenter la solution en deux parties.

I) traitements ayant lieu avant l'emprunt du véhicule par le client :

- a) Les événements suivants peuvent être extraits du texte :
  - Phrase 1 : /
  - Phrase 2 : Arrivée du client, Choix proposé
  - Phrase 3 : Choix du client, Contrat établi
  - Phrase 4 : Chèque remis, Contrat avec montant
  - Phrase 5 : Clés remises, exemplaire contrat donné

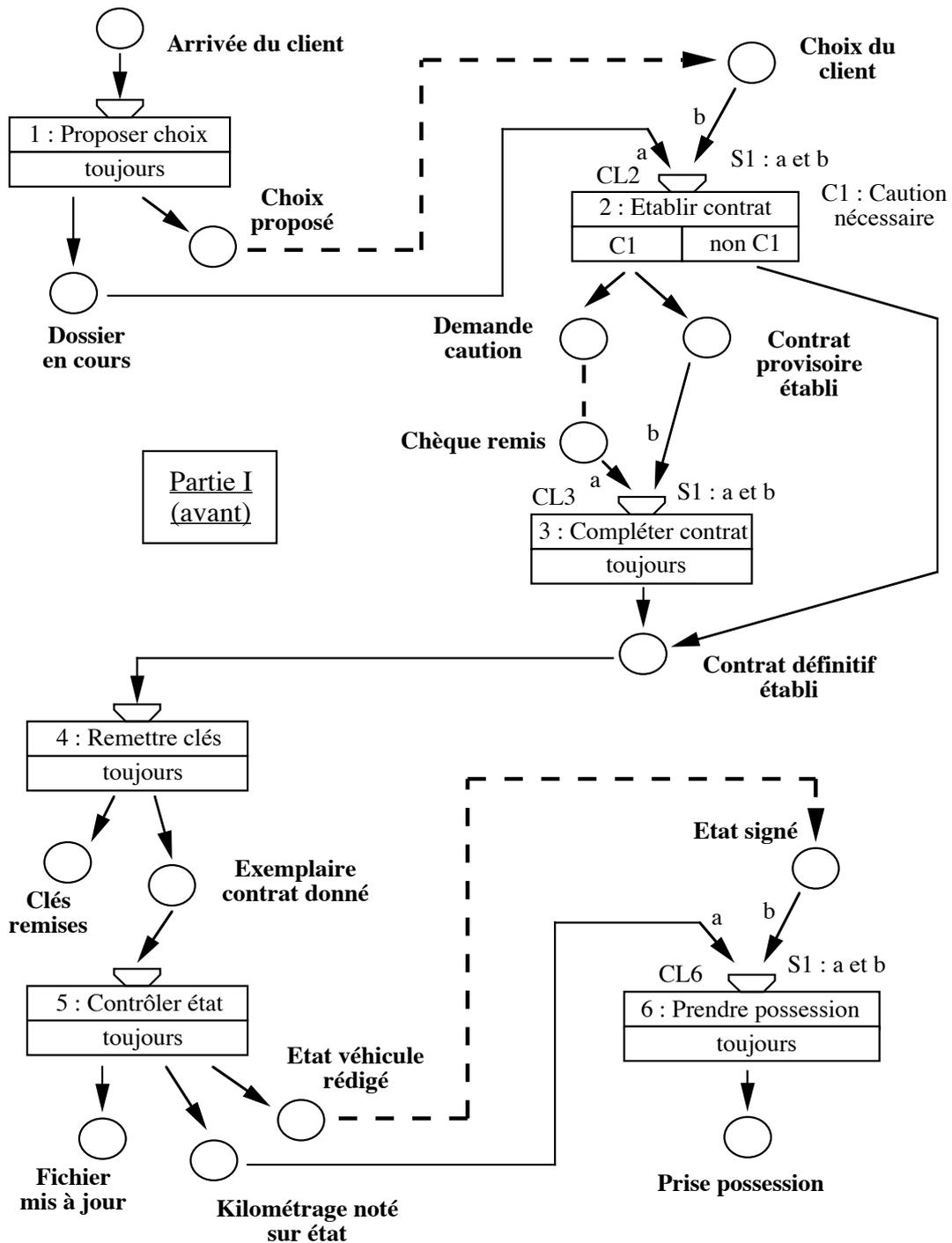
- Phrase 6 : Prise possession du véhicule
- Phrase 7 : Etat du véhicule rédigé et signé
- Phrase 8 : Kilométrage noté sur état
- Phrase 9 : Fichier mis à jour

Ce raisonnement, nous allons le compléter de deux façons : tout d'abord, nous allons ajouter des événements de liaison. Nous voulons, en effet, mieux maîtriser les interactions avec l'utilisateur. Chaque événement externe destiné au client sera systématiquement doublé d'un événement interne. Le traitement correspondant à la réponse ne sera mis en œuvre que s'il y a réponse (événement externe) et attente de réponse (événement interne). Nous allons ensuite, après avoir revu la structure du texte, reprendre les événements **Contrat établi** et **Contrat avec montant** et les remplacer par deux autres, **Contrat définitif établi** et **Contrat provisoire établi**. Ce dernier correspond mieux au cas où, la caution étant nécessaire, il faut en noter le montant sur le contrat avant de remettre celui-ci au client. Il y a donc bien deux contrats, un provisoire (sans montant) et un définitif. Pour permettre d'avertir le client de la nécessité de cette caution, nous ajoutons également un événement **Demande caution**. Ces retouches découlent de la présence d'une conditionnelle dans la phrase 4 ("S'il y a lieu, ...")

b) Les opérations suivantes ont été retenues : interroger le client et proposer un choix de véhicule, établir le contrat, compléter le contrat, remettre clés et donner exemplaire du contrat, prendre possession du véhicule, contrôler l'état du véhicule, noter le kilométrage, mettre à jour le fichier.

c) L'ordre dans lequel ces opérations doivent être exécutées n'est pas celui suggéré précédemment. Les trois dernières phrases, en effet, concernent des tâches à réaliser avant la prise de possession du véhicule. Ces opérations sont liées les unes aux autres par les liens de causalité exprimés dans le graphe ci-après :

Les trois conditions locales, **CL2**, **CL3** et **CL6**, sont identiques (**a.ref client = b.ref client**) et jouent le même rôle : garantir que le traitement opère bien sur des événements cohérents (le choix de Mr X et le dossier de ce même Mr X, pour l'opération 2; le chèque remis par Mme Y et le contrat provisoire de cette Mme Y, pour l'opération 3; ...).



II) traitements ayant lieu lors du retour du véhicule :

a) Il est plus difficile d'extraire des événements d'une analyse du texte. Celui-ci est, en effet, assez synthétique. Voici ceux que nous avons mis en évidence à la première lecture :

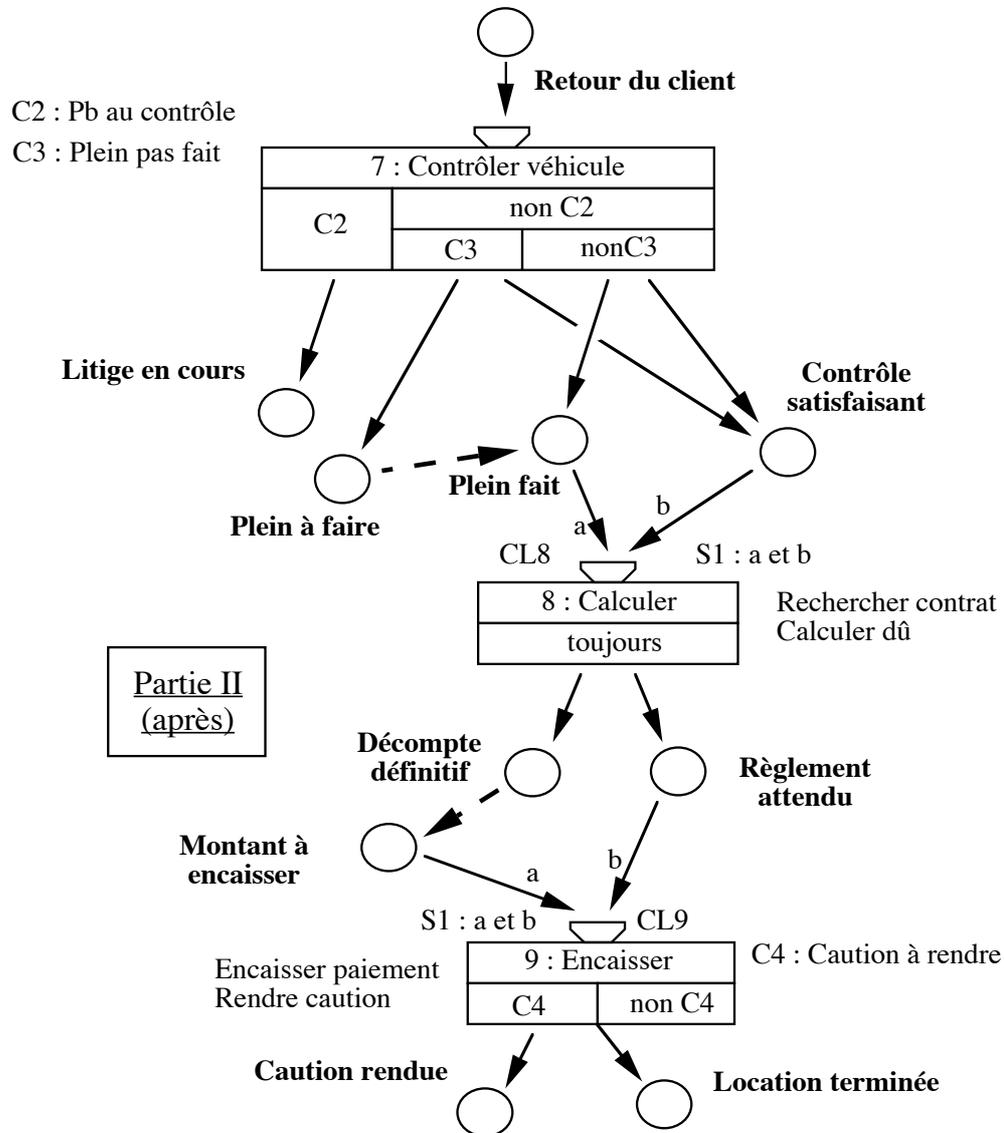
- Phrase 1 : Retour du client, Contrôle réalisé
- Phrase 2 : Décompte définitif, Montant à encaisser
- Phrase 3 : Caution rendue, Contrôle satisfaisant
- Phrase 4 : Plein à faire
- Phrase 5 : Plein fait

Une seconde lecture et l'appel à l'expérience permettent de modifier cette première liste. L'événement Contrôle réalisé est supprimé et devient une opération. Il est remplacé dans

la liste des événements par **Litige en cours**, qui fait le pendant de **Contrôle satisfaisant**. Un événement nouveau est ajouté, **Règlement attendu**, pour contrôler l'interaction avec le client.

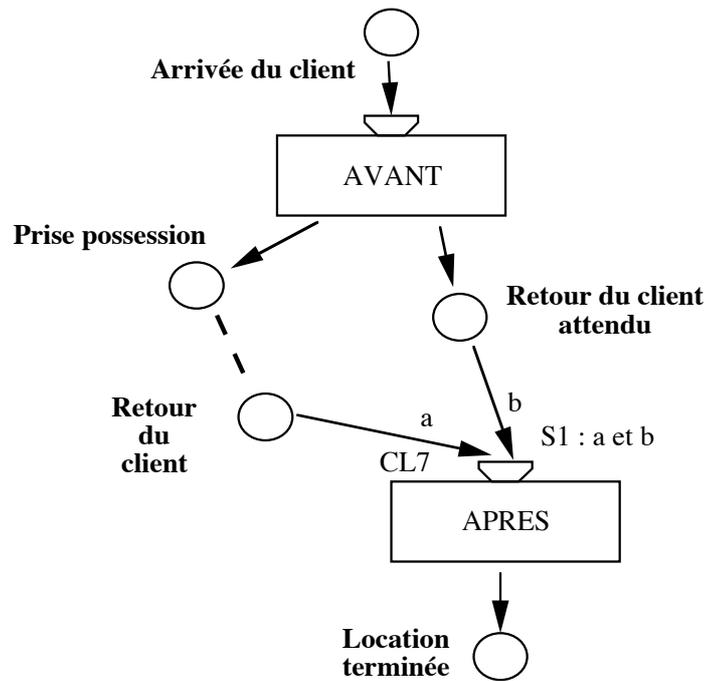
b) Les opérations suivantes peuvent être mises en évidence : contrôler le véhicule, rechercher le contrat et calculer ce qui est dû, encaisser le paiement et rendre la caution.

c)



La condition locale CL8 permet de garantir que les calculs portent sur des véhicules ayant satisfait au contrôle et ayant un plein fait. Quant à CL9, elle garantit que le montant à encaisser correspond bien au règlement attendu. Nous avons également utilisé dans ce MCT une notation particulière pour réduire le nombre d'arcs partant de l'opération 9. L'événement **Location terminée** survient que **C4** soit vraie ou que **C4** soit fausse. Cette notation (qui fait partie de Merise) est équivalente à celle qui consiste à placer deux flèches vers **Location terminée**, une qui part du "cas" **C4**, l'autre du "cas" **non C4**. Nous aurions pu faire la même chose pour les liaisons entre l'opération 7 et l'événement **Contrôle satisfaisant**.

III) La modélisation du fonctionnement de cette entreprise de location se fait à l'aide des deux MCTs, un pour ce qui est réalisé avant la location, un pour ce qui est réalisé après. La synchronisation entre les deux peut se faire par un événement interne, selon un schéma désormais classique :



La condition locale CL7 sert, ici, à ne traiter que des clients dont le retour est attendu.

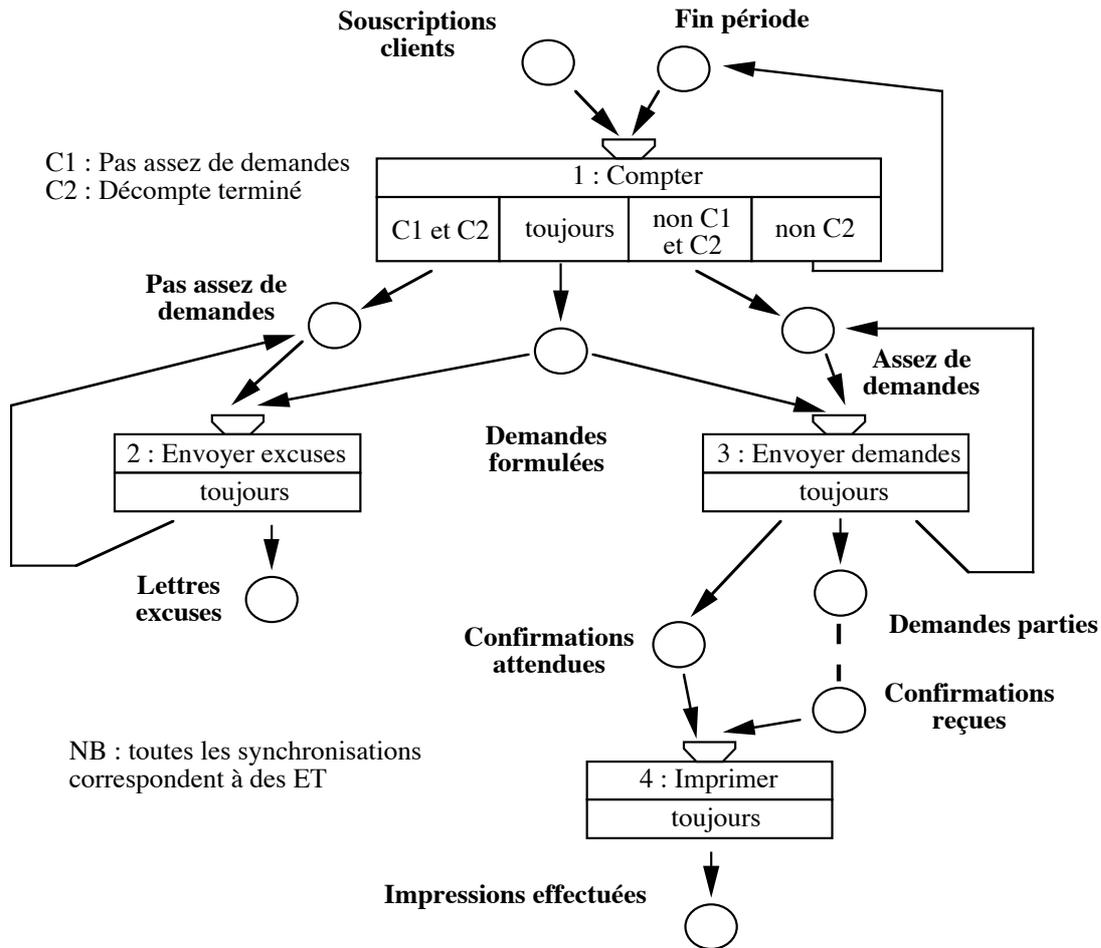
NB : il y a un petit problème dans le texte, qui ne décrit pas exactement ce qui se passe réellement lors d'une location de véhicule. Le contrôle effectué au retour prend usuellement en compte l'état du véhicule avant le départ. Dans l'opération 7, il conviendrait donc de rajouter Prendre état du véhicule avant, ce que l'on pourrait faire en déplaçant Rechercher contrat (actuellement dans l'opération 8) dans l'opération 7 et en émettant l'hypothèse que l'état du véhicule est dans le même dossier que le contrat.

### Exercice 3.2 (8.8)

Reprendre le sujet du CAPET Informatique 1979 (exercice 2.3, page 141) et modéliser le fonctionnement de cette société dans le formalisme des MCT.

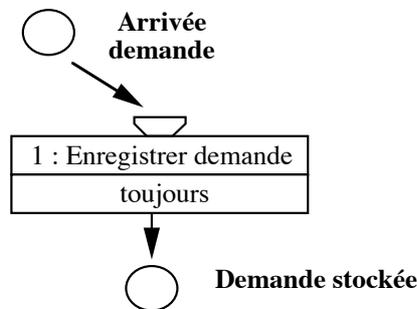
### Solution 3.2

La solution de cet exercice n'a pas été influencée par les modélisations précédentes du même énoncé, qu'il s'agisse des RdP (exercice 2.3 du chapitre 5) ou des DFD (exercice 1.3 du chapitre 7), mais obtenue directement à partir du texte. Nous proposons deux modélisations de ce problème. La première, inspirée de la "philosophie" RdP, peut être celle-ci :



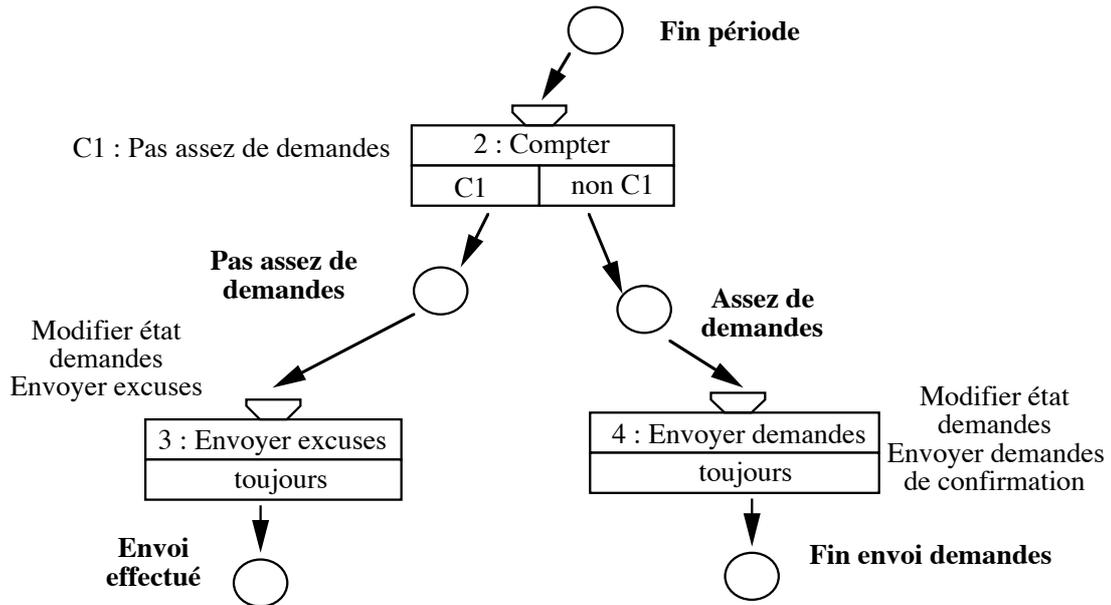
Dans cette modélisation, les demandes de souscription s'accumulent (la façon dont elles y arrivent ne nous intéresse pas) dans l'événement **Souscriptions clients**. Cet événement est, de fait, une place contenant des jetons-souscriptions. A la clôture de cette période, il faut compter le nombre de demandes reçues -ie. compter le nombre de jetons-, regarder s'il y en a assez et prévenir les clients. Les places en entrée et sortie d'une opération symbolisent le jeton d'itération : l'opération est réalisée pour chaque souscription client. Si les demandes sont confirmées (le texte ne précisant pas ce qui se passe si ces confirmations n'arrivent pas, nous n'avons pas envisagé cette hypothèse), l'impression peut avoir lieu. Il restera à vider les places d'itération **Assez de demandes** et **Pas assez de demandes** pour la prochaine souscription.

Nous pouvons voir les choses différemment, sous l'angle "manipulation d'une base de données". Dans cette seconde optique, à chaque réception d'une demande, elle est enregistrée :

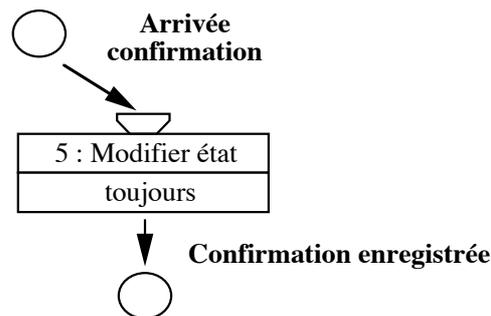


A la fin de la période, la base est parcourue et le nombre de demandes comptabilisé. Selon le résultat de ce comptage, une lettre d'excuses ou une demande de confirmation est envoyée

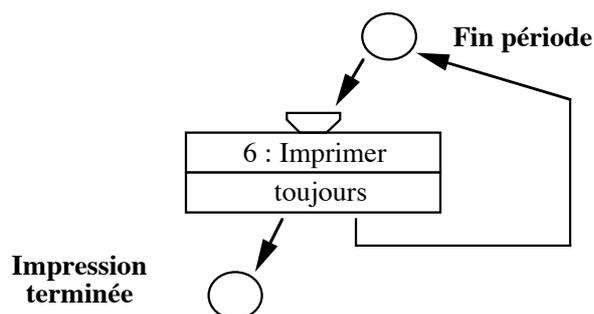
aux clients.



Chaque réponse est enregistrée :



L'impression se fait après une période d'attente des réponses :



Dans ce second travail de modélisation, les tâches sont plus fractionnées, la cohésion globale étant assurée par la base de données. Des préconditions doivent assurer la validité des opérations. Les demandes sont enregistrées au fur et à mesure de leur arrivée. A la clôture de la souscription (c'est un seul jeton), la base est relue et les demandes comptées. Selon les cas, les demandes sont marquées "acceptée" ou "annulée". L'impression se fait pour toutes les demandes confirmées, à la fin du temps d'attente des réponses. Il s'agit encore d'un seul jeton. On a abordé la description du point de vue utilisation de la base de données. L'algorithme suivant donne une idée du fonctionnement global :

Début

Tantque il y a des demandes de souscriptions faire

ENREGISTRER DEMANDE

finfaire

... Attendre fin de la période

COMPTER

Si pas assez de demandes alors

Pour chaque demande faire

MODIFIER ETAT DEMANDE ("annulée")

IMPRIMER EXCUSES

finfaire

sinon

Pour chaque demande faire

MODIFIER ETAT DEMANDE ("acceptée")

IMPRIMER DEMANDE DE CONFIRMATIONS

finfaire

Tantque il y a des confirmations faire

MODIFIER ETAT DEMANDE ("confirmée")

finfaire

... Attendre fin de la période

Pour chaque demande confirmée faire

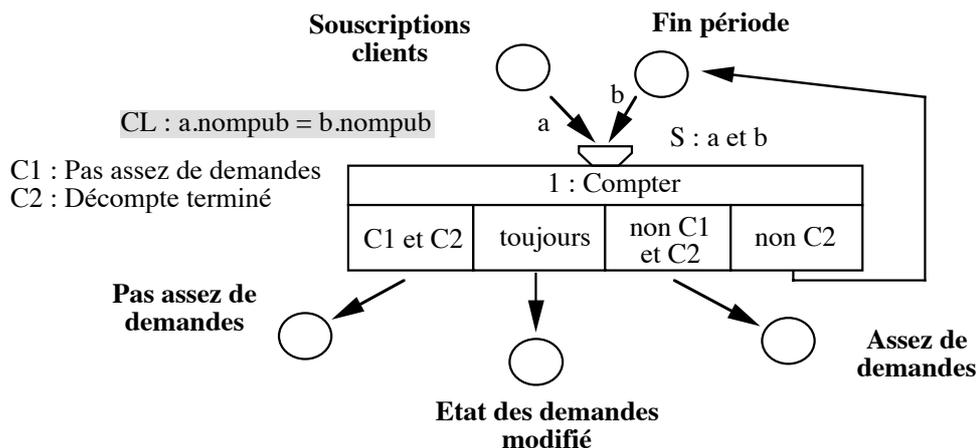
IMPRIMER

finfaire

finsi

Fin

Cet algorithme (comme nos MCTs du reste) ne fonctionne que pour UNE SEULE publication. Si nous devons en gérer plusieurs, il faudrait colorer les jetons, ce qui, avec un formalisme comme les MCT de Merise, nécessite le recours à un message (pour véhiculer le nom de la publication concernée) et l'ajout d'une condition locale (pour sélectionner les jetons correspondant à la "bonne" publication). Voici l'opération **Compter** du MCT de la première approche, "relookée" pour accepter plusieurs publications :

**Exercice 3.3 (8.13)**

Reprendre le MCT décrivant le fonctionnement de la société LOCATION SA (exercice 3.1) et exprimer le MOT correspondant.

**Solution 3.3**

Le texte fournit des renseignements sur les services réalisant les opérations. Il ne fournit aucune information temporelle. Nous avons dû émettre des hypothèses complémentaires concernant le "qui fait quoi ?" et supposer que l'opération 5 était l'œuvre du responsable du parc automobile (c'est lui, selon notre hypothèse, qui mémorise l'état du véhicule et qui met à jour le fichier) et que l'hôtesse assure l'opération 6.

Nous supposons également qu'au retour le contrôle du véhicule est assuré par le responsable du parc et que les autres opérations le sont par l'hôtesse. Nous supposons enfin que l'organisation souhaite traiter en priorité les factures de clients en règle. Nous devons donc dupliquer l'opération 8, 8a pour le traitement des clients en règle, 8b pour les autres.

par service nécessite de casser l'opération 4 en deux, 4a (**Prendre possession véhicule, Relever kilométrage** et Les MOTs résultant de cette ventilation-adaptation sont présentés en deux parties. La première met en évidence un seul acteur externe, le client, tandis que la seconde en introduit un autre, le pompiste qui effectue le plein du véhicule emprunté.

CL8 (nous en avons modifié le rôle en passant du MCT au MOT) assure maintenant que le plein qui a été fait correspond à un véhicule dont le contrôle était satisfaisant, mais dont le réservoir n'était pas plein.

