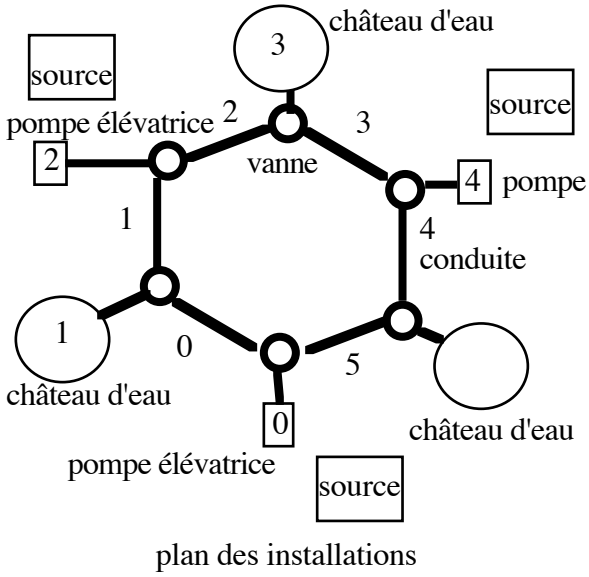
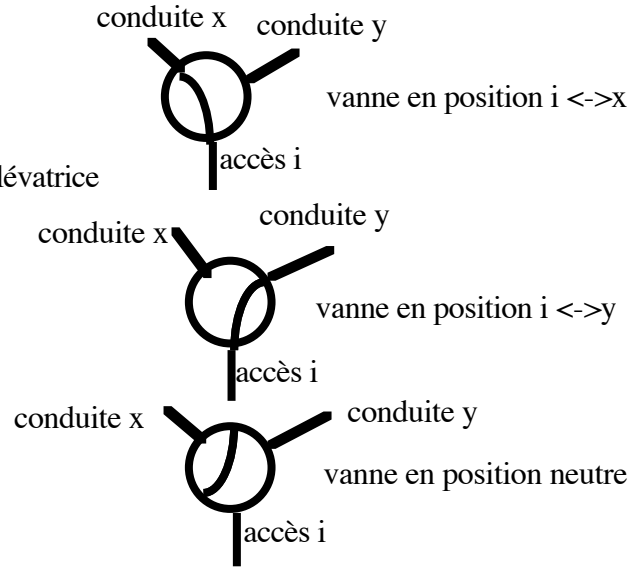
ED 11
Sémaphores privés

Dans une commune de grande étendue il y a 3 châteaux d'eau qui sont alimentés par 3 pompes élévatrices qui pompent l'eau de 3 points d'eau (sources ou réservoirs naturels). Pour éviter une pénurie d'eau en cas de tarissement d'un des points d'eau, le conseil municipal décide de construire des conduites pour pouvoir alimenter un château d'eau à partir de deux points d'eau. Mais par souci d'économie, il ne veut pas doubler les pompes élévatrices ; par contre installer les conduites n'est pas onéreux et au contraire apporte de l'emploi dans la commune. Le conseil municipal fait donc l'achat de 3 vannes et fait construire 6 conduites d'eau selon le plan ci-dessous.



plan des installations



Question1.

Le conseil municipal confie l'entretien des 6 conduites d'eau à 6 employés communaux. chacun d'eux est responsable d'une conduite et de son utilisation. Chaque employé peut donc actionner les deux vannes qui sont à chaque bout de la conduite, l'une vers un château d'eau, l'autre vers une pompe. Les vannes comportent trois positions (neutre, affectée à l'une ou l'autre des conduites), ce qui interdit à deux pompes d'alimenter le même château d'eau (il y aurait une surpression dangereuse), ou à une même pompe d'alimenter deux châteaux d'eau (il y aurait une dépression et arrêt de la pompe).

Montrer que si les actions des 6 employés communaux ne sont pas coordonnées, il peut y avoir interblocage.

Montrez que cet interblocage peut être évité par une gestion simple de l'emploi du temps de ces 6 employés.

Question 2.

Un membre informaticien du conseil municipal (ancien auditeur du CNAM, qui n'a pas tout oublié du cours système), décide d'étudier plusieurs règles d'action par ces employés sur les vannes et d'analyser leur efficacité par un programme de simulation .

Pour cela, il lance l'exécution de 6 tâches analogues. Les 6 tâches peuvent se dérouler en parallèle, il faut donc contrôler l'utilisation qu'ils font des vannes et compléter le comportement, ci-dessous, de chaque tâche par un prélude et un postlude réalisant l'évitement de l'interblocage :

Contexte commun

On suppose le type conduite prédéfini, c'est en fait le type ID_PROC

```
Typedef ID_PROC enum {0,1,2,3,4,5} ;

SEM SEMPRIV[ID_PROC] ; // Un sémaphore binaire associé à chaque tâche

SEM MUTEX ; // Exclusion mutuelle pour l'accès aux variables d'état

// déclaration des variables d'état à compléter

  TASK_CODE PROCESSUS  (ID_PROC  I, vanne X, vanne Y) {
  /* I : association d'un nom unique à la tâche */
  /* X = vanne à un bout de Z ; Y := vanne à l'autre bout de Z ;*/
  conduite  Z = I ;

  while true {
            prelude ;
            connecter X à la conduite Z ;
            connecter Y à la conduite Z ;
            simuler l'utilisation de la conduite pendant un délai aléatoire
  ;
            comptabiliser les durées d'attente et d'utilisation de Z ;
            déconnecter Y, déconnecter X ;
            postlude ;
            attendre un délai aléatoire;
             }

  }
```

Compléter le programme ci-dessus en introduisant des variables d'état et en appliquant la technique dite des sémaphores privés, avec un tableau de 6 sémaphores privés SEMPRIV[Z] où Z correspond au numéro de conduite. Ne pas oublier d'initialiser les sémaphores.

Question 3

On sait que cette méthode des sémaphores privés permet d'éviter l'interblocage, mais qu'elle ne prévient pas l'attente indéfinie d'une tâche par suite de coalition involontaire entre d'autres tâches.

Montrer un exemple de coalition possible.

Question 4

Pour éviter cette coalition, on donne à chaque conduite une priorité qui croît avec l'attente du tâche, et on ne connecte une vanne à une conduite que si la conduite voisine n'a pas une priorité plus forte.

Par exemple la priorité p[Z] d'une conduite Z vaut zéro quand la conduite est libre ou occupée. Que se passe-t-il quand la conduite Z est demandée ? Si les conduites Z-1 (modulo 6) et Z + 1 (modulo 6) ne sont pas occupées ou ne sont pas demandées avec des priorités plus grandes que p[Z], alors la conduite Z peut être attribuée et sa priorité p[Z] est remise à zéro. Si la conduite Z ne peut être attribuée, on en tient compte en augmentant de 1 la priorité p[Z] chaque fois que l'une des conduites Z-1 ou Z + 1 est libérée et lors de la demande de Z.

Reprendre le programme de la question 2 pour y introduire les priorités.

Question 5

Votre solution conduit-elle à interblocage ? Expliquer pourquoi.