# Conception des systèmes répartis

3ième Année Informatique et Mathématiques Appliquées
Durée : 2 heures
Documents autorisés : Précis et notes de cours

23 octobre 2003

**Remarque préliminaire** : Toutes les questions valent 2 points.

## Quelques problèmes de causalité entre flux multimedia

On considère une application échangeant des flux multimedia de type MPEGx par exemple. Un site A diffuse un flux vidéo vers 2 autres sites B et C. Après avoir reçu le début du flux émis par A, le site B émet un flux à son tour (flux audio par exemple) vers les sites A et C. Le chronogramme de la figure (1) donne une vision événementielle du déroulement de l'exécution répartie en ne conservant que les événements de début et fin de flux qu'il s'agisse d'un flux en émission ou d'un flux en réception.

**Questions**

1. Sur le chronogramme de la figure (1), les paires d'événements suivants sont-elles liées par la relation de causalité : $(A.d1, B.d2), (A.f1, B.f2)$, $(A.d2, B.f2)$, $(A.d2, C.f1)$ ? Justifiez vos réponses.
2. Montrer que la réception des flux 1 et 2 sur le site C présente une anomalie par rapport à la causalité en ce qui concerne leur début.
3. Un protocole ordonné pourrait-il corriger cette anomalie sur l'ordre de démarrage des flux sur le site de réception C ? Justifiez votre réponse.
4. De façon similaire, la figure 1 montre que les fins de flux sur les sites A et B se produisent dans un ordre différent : sur A, la fin de l'émission $A.f1$ du flux 1 précède la fin de la réception $A.f2$ du flux 2 et sur B, la fin de l'émission $B.f2$ du flux 2 précède la fin de la réception $B.f1$ du flux 1. Montrez que malgré cette inversion, aucune anomalie causale n'existe.
5. Dans le chronogramme étudié, la question précédente conduit à considérer que la fin de réception des flux 1 et 2 sur le site C peut se produire dans n'importe quel ordre. Modifier le chronogramme pour obtenir une situation où la réception des flux 1 et 2 devrait être forcément ordonnée comme dans le chronogramme fourni (dans lequel $C.f2 \prec C.f1$) pour ne pas provoquer d'anomalie causale.

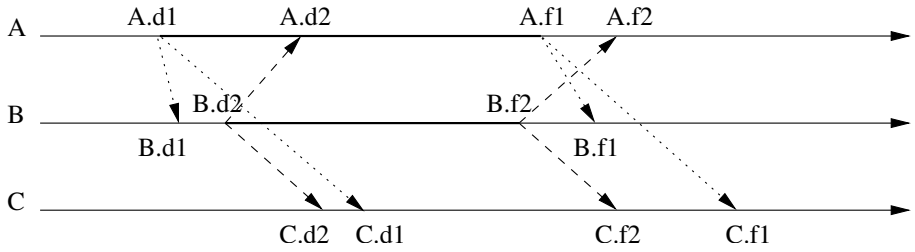


FIG. 1 – Vision événementielle des flux

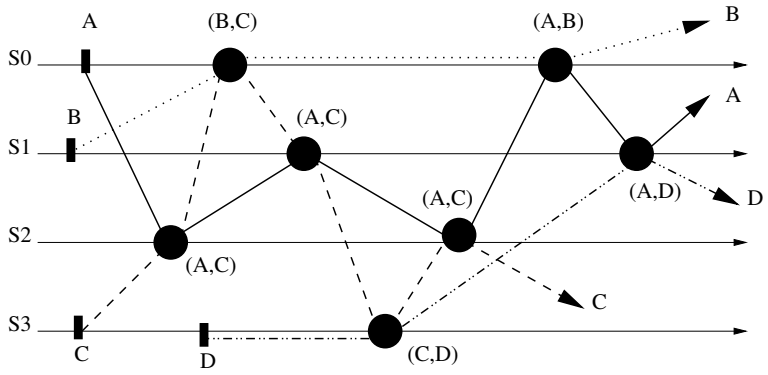


FIG. 2 – Rencontres d'agents mobiles sur des sites

# Rencontres entre agents mobiles

On considère une application répartie à base d'agents mobiles. Ces agents se rencontrent pour échanger des informations lorsqu'ils se trouvent simultanément sur le même site[1]. Une telle rencontre est considérée comme atomique et réalisée toujours entre une paire d'agents[2].

La figure (2) illustre le déplacement et les rencontres de tels agents désignés par $A, B, C, \ldots$

On se propose de tracer des informations sur les rencontres qui ont lieu durant l'exécution des agents. Pour cela, on peut envisager de collecter des informations soit sur les sites de rencontre, soit dans le contexte local de chaque agent. On s'intéresse plus particulièrement à la datation des rencontres.

## Trace sur les sites

On mémorise sur chaque site les dates des rencontres qui ont eu lieu. Le mécanisme de datation s'inspire du système de datation de Mattern. Une date est représentée par un vecteur de dimension $N$ où $N$ est le nombre de sites.

Chaque site possède une horloge vectorielle $H_i$ dotée d'une opération permettant de dater les rencontres et les agents vont transporter un vecteur courant unique représentant la date de leur dernière rencontre. On veut que la sémantique d'une date $T$ affectée à une rencontre sur un site $i_0$ soit la suivante :

$$\textbf{invariant} \quad (\forall i \neq i_0 \ : \ T[i] = Card(R_i)) \wedge T[i_0] = Card(R_{i_0}) + 1$$

dans lequel l'ensemble $R_i$ contient tous les événements de rencontre ayant eu lieu sur le site $i$ et qui précèdent causalement la rencontre datée $T$. À titre d'exemple, la date $T$ affectée à la rencontre $(A, D)$ sur le site $S_1$ devrait être le vecteur $T = (2, 2, 2, 1)$.

Initialement, l'horloge $H_i$ d'un site $S_i$ est initialisée à zéro : $\forall j \ :: \ H_i[j] = 0$. La classe Horloge suivante possède une seule opération Dater qui ne sert qu'à dater les événements de rencontre.

```
class Horloge {
   int cpt[]; int loc;   // "loc" localise le site de l'horloge
   // Datation d'une rencontre
   public int[] Dater( int T1[], int T2[] ) {  ... }
   public Horloge(int où, int N) {
      loc = où; cpt = new int[N] ;
      for (int i = 0; i<N; i++) cpt[i]=0 ;
      }
}
```

[1] On ne s'intéresse pas à leurs communications éventuelles à distance.

[2] Un agent peut par contre rencontrer successivement plusieurs agents avant de se déplacer vers un autre site.

La sémantique de Dater comporte deux paramètres d'entrée, en l'occurrence les deux vecteurs horloges transportés par les agents participant à la rencontre.

**Questions**

6. Donner sous forme d'un triplet de Hoare, la sémantique de Dater :

$$\{H_i.cpt = h\}\ H_i.\mathsf{Dater}(T1, T2)\{\ldots\}$$

   et programmer l'opération Dater dans la classe Horloge.

7. Décorer la figure avec les dates affectées aux rencontres en utilisant la feuille fournie en annexe.

## Trace dans les agents

Cette fois-ci, chaque agent a transporte une horloge vectorielle locale $H_a$ (horloges embarquées dans les agents). Chaque horloge locale est initialisée à zéro : $\forall a \in Agents\ :\ \forall i \in 1..M\ :\ H_a[i] = 0$. Une date est donc représentée par un vecteur de dimension $M$ où $M$ est le nombre d'agents[3]. Par contre, les sites n'ont plus d'horloges locales. On doit donc trouver une implantation différente de l'opération Dater. On suppose que chaque agent fournit en paramètre à l'opération Dater son indice et son vecteur horloge local. La classe Datation définit la nouvelle opération Dater :

```
class Datation {
   // Datation d'une rencontre
   static public  int[] Dater(int a, int Ha[], int b, int Hb[]) { ... }
}
```

On veut que la sémantique d'une date $T$ affectée à une rencontre entre un agent $a$ et un agent $b$ soit la suivante :

$$\textbf{invariant}\ \ (\forall k \neq a, b\ :\ T[k] = Card(R_k)) \wedge T[a] = Card(R_a) + 1 \wedge T[b] = Card(R_b) + 1$$

dans lequel un ensemble $R_x$ contient tous les événements de rencontre auxquels l'agent $x$ a participé et qui précèdent causalement la rencontre datée $T$. À titre d'exemple, la date $T$ affectée à la rencontre $(A, C)$ sur le site $S_1$ devrait être le vecteur $T = (2, 1, 3, 0)$.

**Questions**

8. En désignant, dans la postcondition, le vecteur date résultat par $T_r$, donner sous forme d'un triplet de Hoare, la sémantique de Dater :

$$\{H_a = h_a \wedge H_b = h_b\}\ Datation.\mathsf{Dater}(a, Ha, b, Hb)\{\ldots\}$$

   et programmer l'opération Dater dans la classe Datation.

9. Montrer que l'on a l'invariant :

$$\textbf{invariant}\ \forall T\ :\ (\sum_{x=1}^{x=M} T[x])/2 = nr_T + 1$$

   dans lequel le terme $nr_T$ est égal au nombre total de rencontres (quel que soit les participants) qui précèdent causalement la rencontre datée par $T$.

   Pour cela, montrer que si le précidat invariant est vrai avant l'exécution d'une opération Dater, il est encore vrai près l'exécution de l'opération.

10. Décorer la figure avec les dates affectées aux rencontres en utilisant la feuille fournie en annexe.

[3]On suppose que le nombre maximum d'agents est connu pour simplifier et conserver des vecteurs.

# Annexe

## Pour la réponse à la question 7

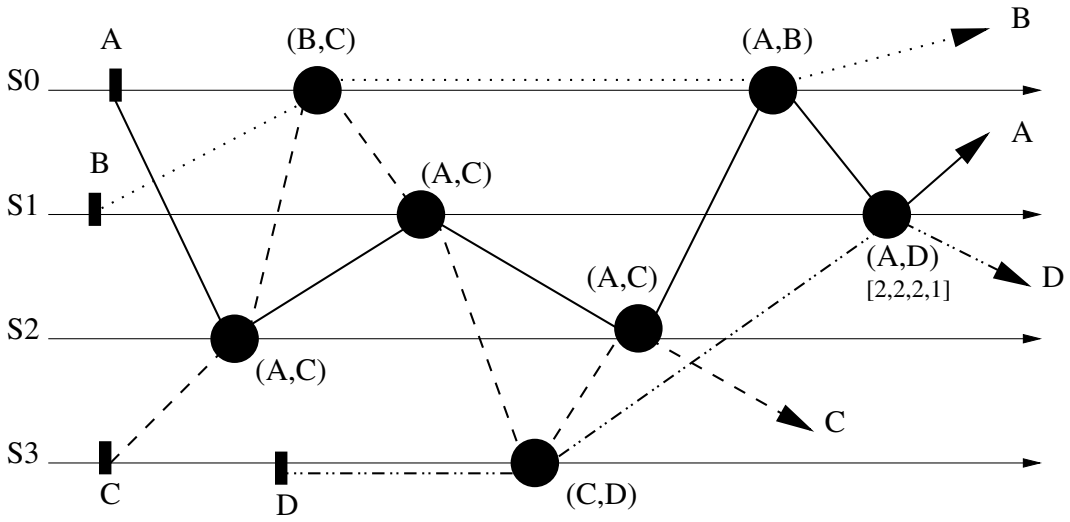


FIG. 3 – Datation des rencontres avec des horloges sur les sites

## Pour la réponse à la question 10

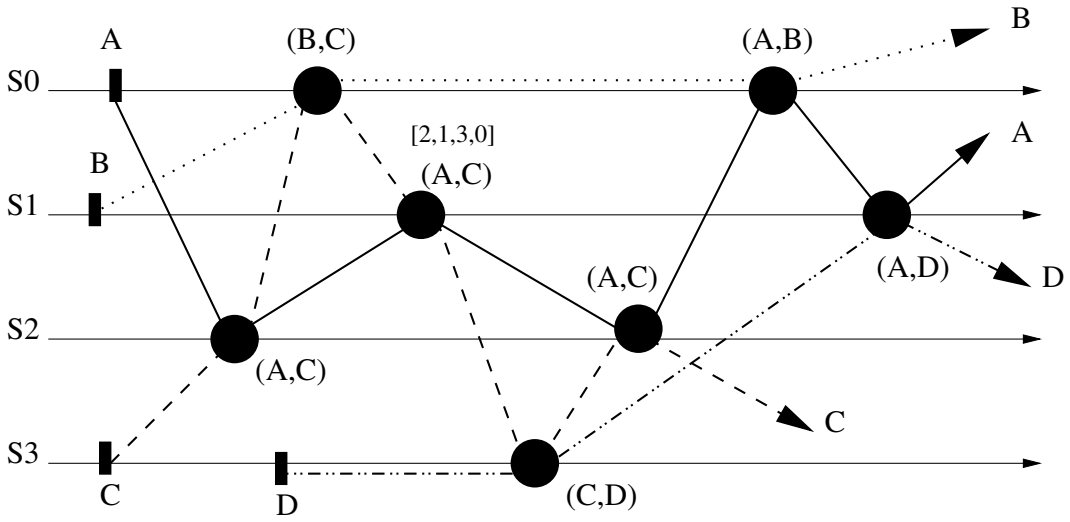


FIG. 4 – Datation des rencontres avec des horloges dans les agents