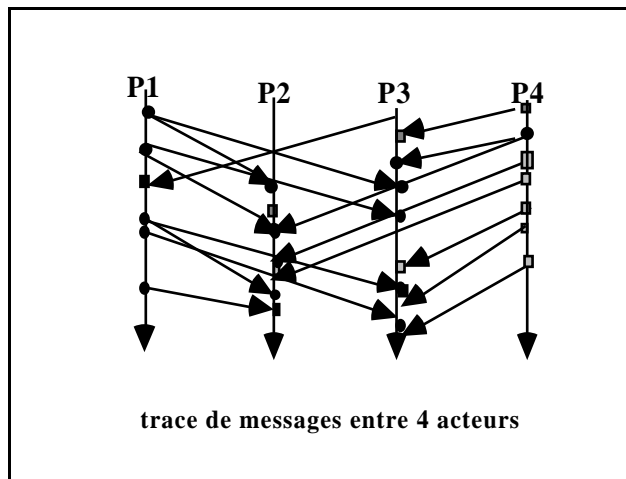
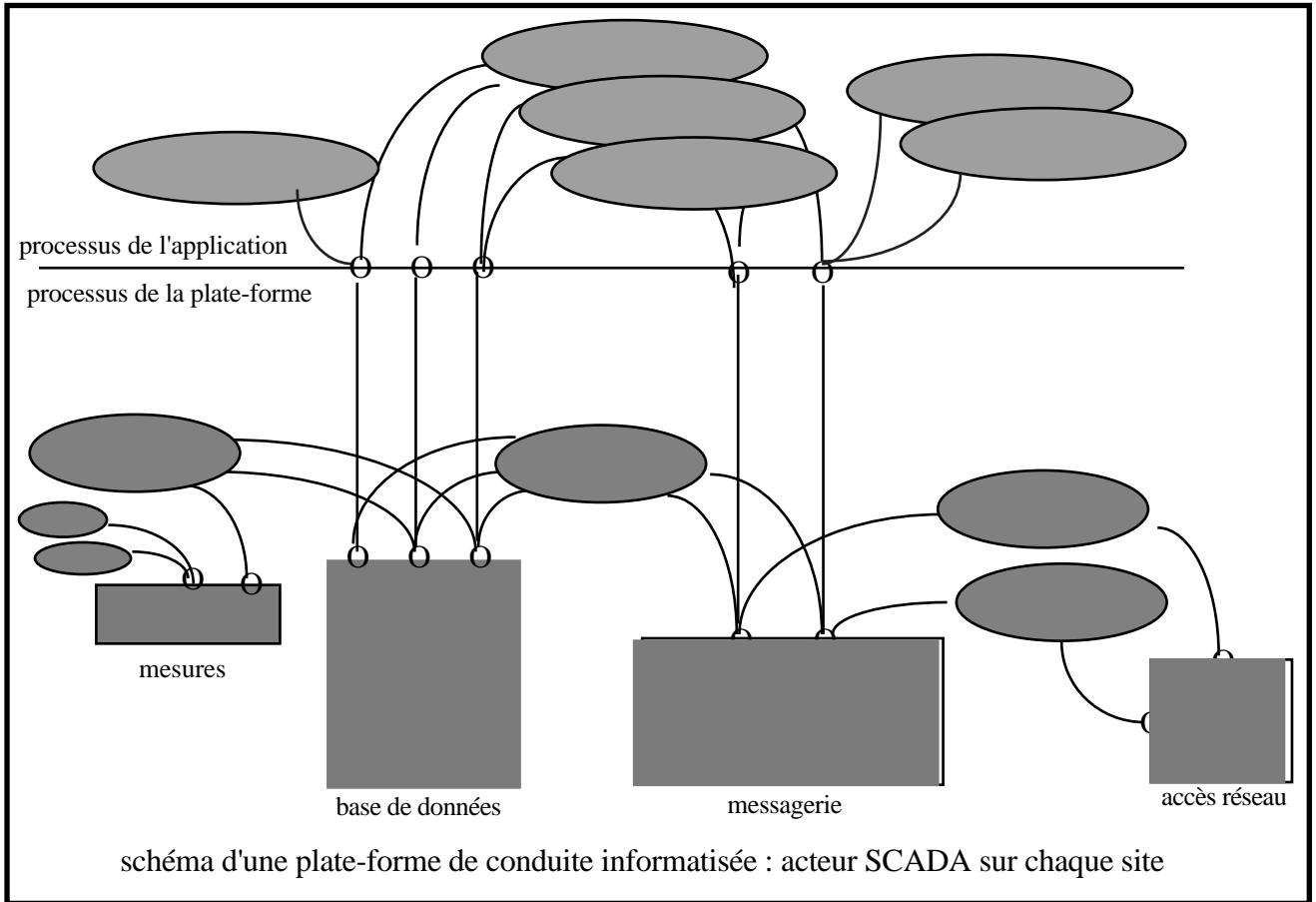


# Systèmes et Applications Répartis

**Ordres, état global, horloges, synchronisation, contrôle et reprise dans les systèmes répartis**

**C Kaiser**



## **ORDRES, ÉTAT GLOBAL, HORLOGES, SYNCHRONISATION, CONTRÔLE ET REPRISE DANS LES SYSTÈMES RÉPARTIS**

### **1. Besoin des applications réparties**

**Réel et modèles de communication élémentaire**

**Dépendance causale**

**Modèles de diffusion fiable et communication de groupe**

**Propriétés d'ordre dans les groupes**

### **2. Etat global d'un système réparti**

**Passé et coupures cohérentes**

**Détermination d'un état global cohérent**

### **3. Datation causale et horloges vectorielles**

**Horloges vectorielles et coupures cohérentes**

**Diffusion fiable avec ordre causal**

### **4. Ordre total par horloges logiques**

**Exclusion mutuelle répartie, avec horloge logique**

### **5. Pose de points de reprise répartis**

**Chemins en zigzag**

**Sauvegarde adaptative**

### **6. Étude de cas : calcul coopératif et objet répliqué sur 4 site**

### **7 Exercices avec solutions**

### **Bibliographie :**

**R.Balter, J.P.Banâtre, S.Krakowiak, éditeurs. *Construction des systèmes d'exploitation répartis*. Collection didactique INRIA 1991 (350 pages)**

**G.Coulouris, J.Dollimore, T.Kindberg. *Distributed Systems (2nd edition)*. Addison Wesley 1995 (601 pages)**

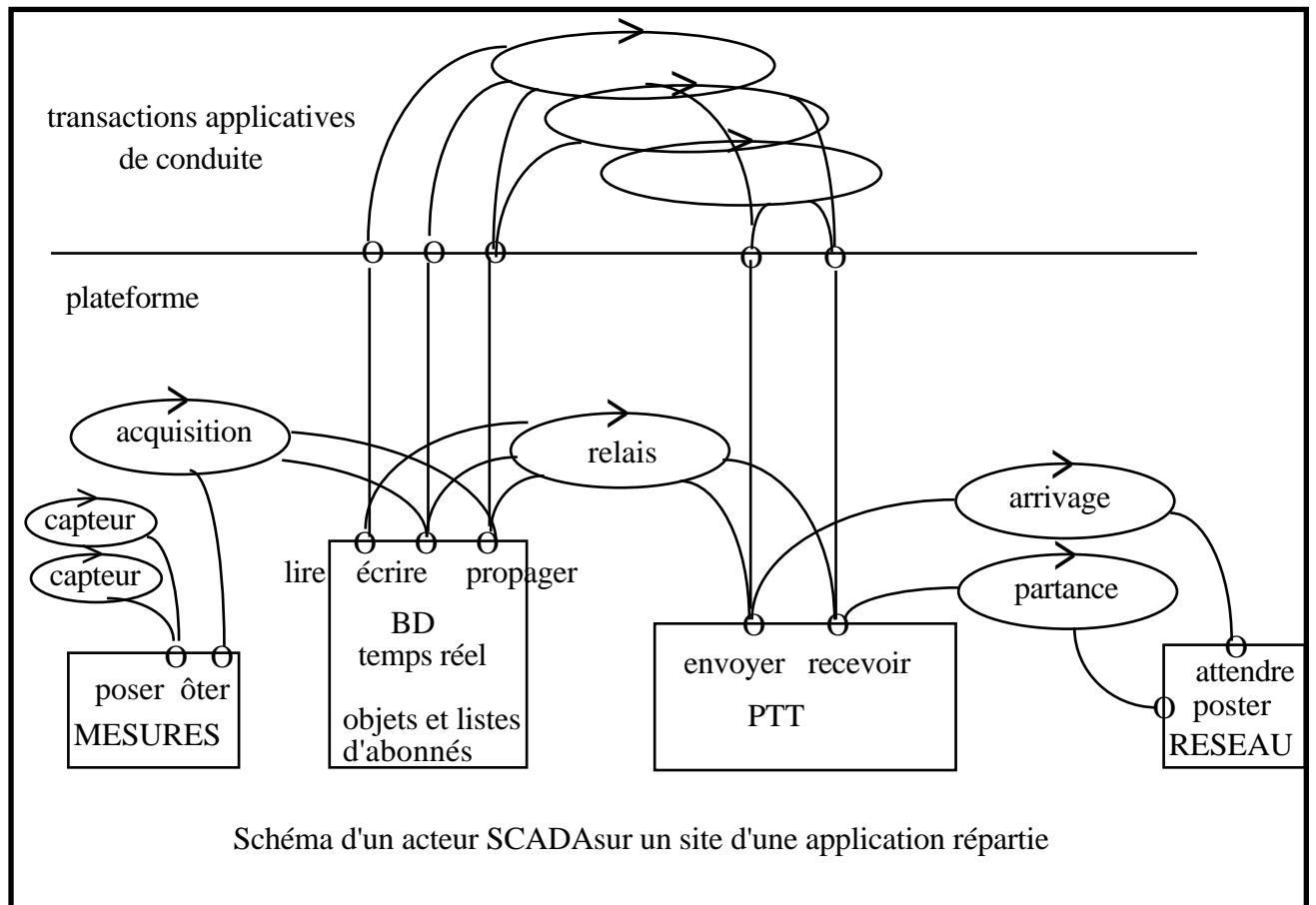
**S.Mullender. *Distributed Systems (2nd ed.)*. Addison Wesley 1994 (644 p.)**

**A.Tanenbaum. *Distributed Operating Systems*. Prentice Hall 1995 (614 p.)**

**J.Besancenot et al. *Les systèmes transactionnels*. Hermès 1997 (415 p.)**

**G.Blair, J.B.Stéfani. *Open Distributed Processing and Multimedia*. Addison Wesley 1998 (452 p.)**

## BESOINS DES APPLICATIONS REPARTIES



### APPLICATION REPARTIE = ENSEMBLE DE SITES

#### CHAQUE SITE SCADA COMPREND

##### UNE PLATE-FORME AVEC:

**des modules d'acquisition : captures concurrentes, acquisition synthétique**

**des bases de données temps réel : lecteurs rédacteurs en mémoire centrale**

**une messagerie : producteurs consommateurs**

**un module réseau : communication de messages intersites**

**des processus de service**

##### UNE COUCHE APPLICATIVE AVEC :

**des processus appelés transactions applicatives**

# **BESOINS DES APPLICATIONS REPARTIES**

## **REPARTITION SIMPLE (CLIENT SERVEUR)**

**accès local ou distant aux BD TR, chaque BD cohérente individuellement**  
**abonnement à BD primaire: copies secondaires pour lecture, sur autres sites**  
**transactions avec accès à une seule BD primaire à la fois**  
**messagerie entre les processus de divers sites**

## **REPARTITION PLUS COMPLEXE (APPLICATION COOPÉRATIVE)**

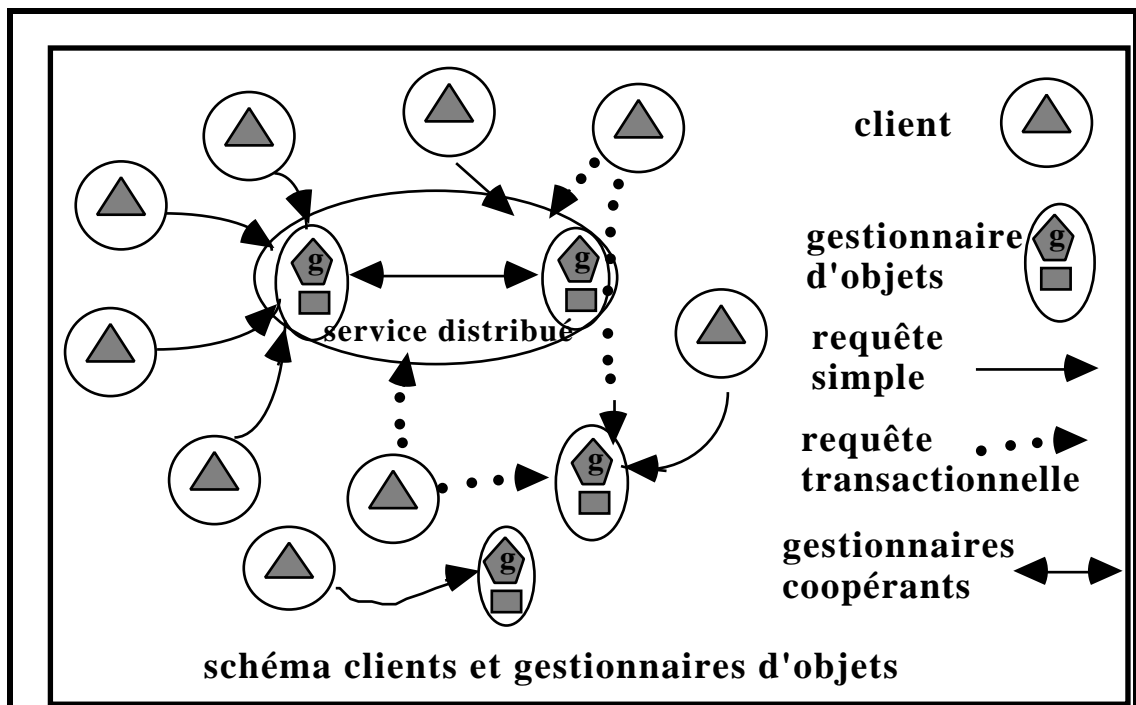
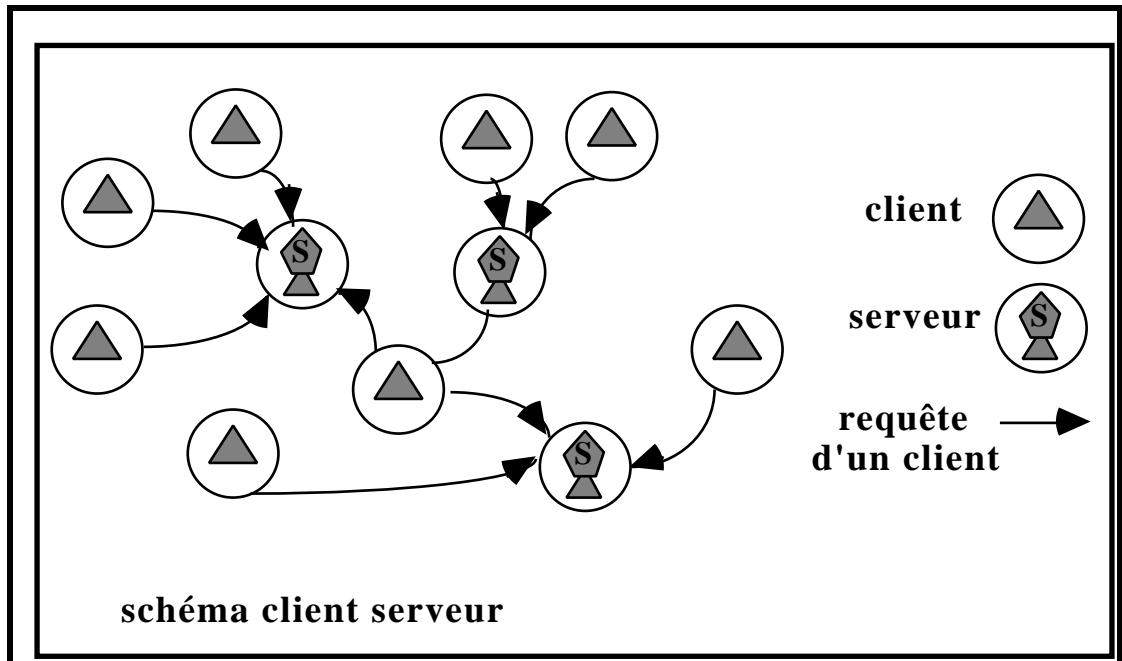
**Transactions avec accès emboîtés à plusieurs BD :**  
**problèmes de cohérence globale et interblocage**

**Copies multiples d'une même BD avec écritures sur chaque copie :**  
**cohérence faible ou forte (problème des caches multiples)**

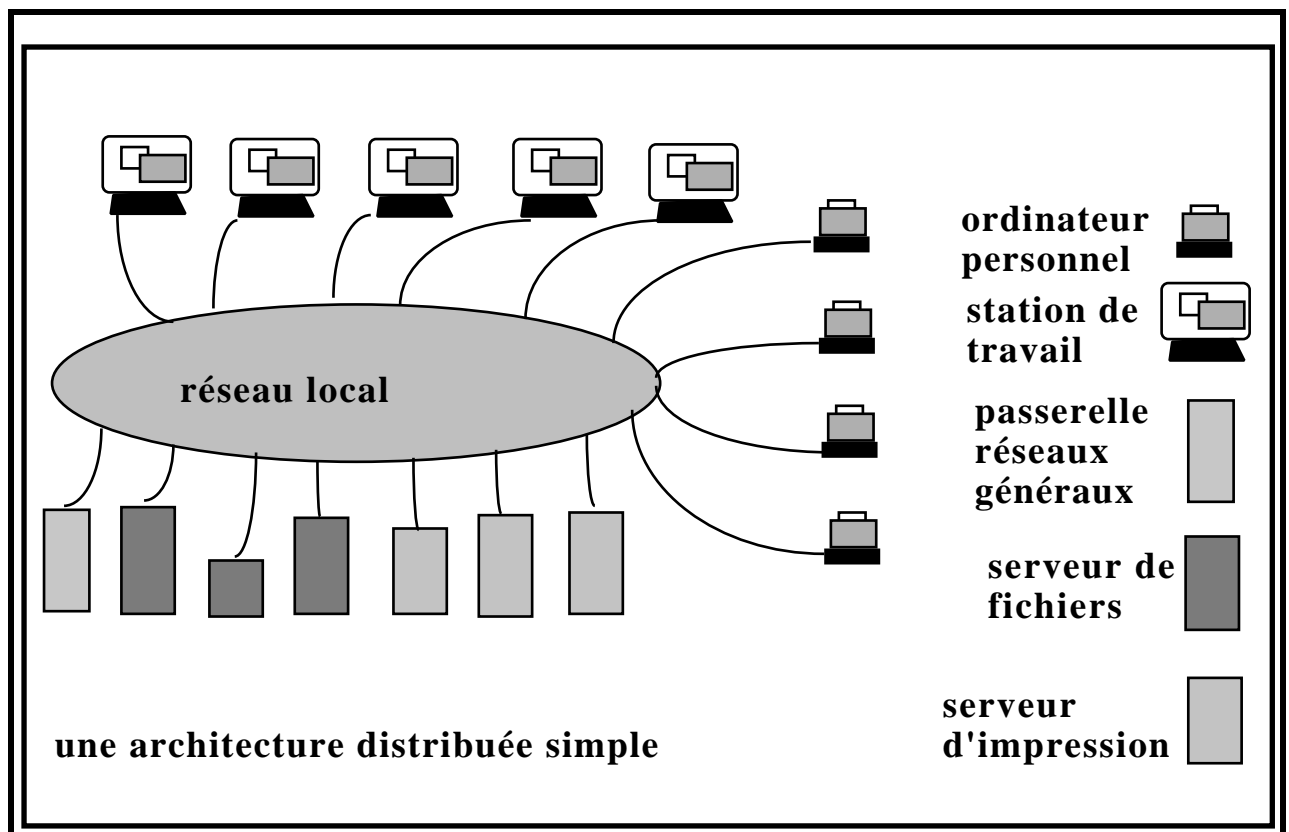
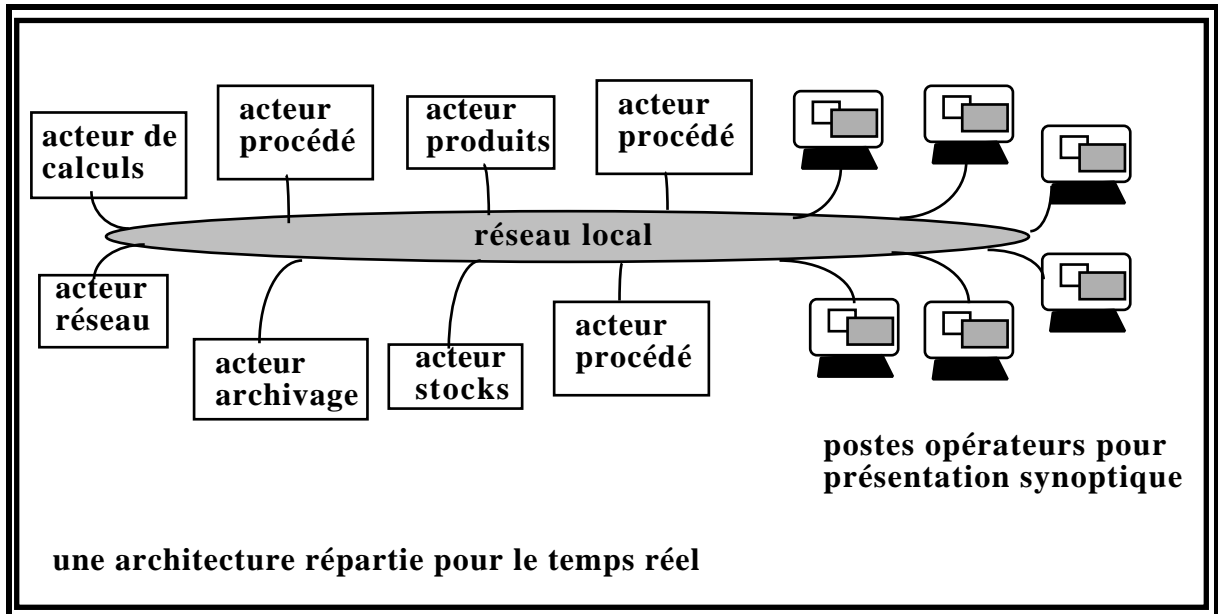
**Ensemble de transactions coopérantes. Problèmes de synchronisation :**  
**démarrage dans un état cohérent**  
**coordination par un site fixe, mais si absent (panne, maintenance) alors**  
**élection d'un nouveau coordinateur**  
**terminaison de la coopération**  
**mise au point répartie**  
**points de reprise cohérents**  
**diffusion d'information dans un groupe**

**Diffusion fiable et ordonnée à un groupe de processus sur des sites divers**  
**Mobilité des sites**

# BESOINS DES APPLICATIONS REPARTIES



## EXEMPLES D'ARCHITECTURE PHYSIQUE (ARCHITECTURE ORGANIQUE)



# **LE REEL DES COMMUNICATIONS ELEMENTAIRES**

## **POINT A POINT MODE MESSAGE**

**message d'un émetteur vers un récepteur sur un canal**

**perte de message, absence de récepteur (panne, maintenance,...)**

**contrôle d'erreur par acquit et délais de garde, mais duplication possible**

**numérotation des messages successifs**

**réception désordonnée d'une suite de messages**

**contrôle de flux pour asservir les vitesses de l'émetteur et du récepteur**

**(producteur-consommateur)**

**incertitude sur l'état du canal, de l'émetteur et du récepteur**

**durées de transfert variable, asynchrone (mais il existe des bus synchrones)**

## **POINT A POINT MODE CLIENT SERVEUR**

**l'émetteur est le client et le récepteur est le serveur**

**message requête d'un client vers un serveur sur un canal**

**message réponse du serveur au client**

**le client n'envoie pas d'autre requête avant d'avoir reçu la réponse**

**mêmes problèmes d'erreurs, d'incertitudes et de variabilité des délais**

## **DIFFUSION A UN GROUPE**

**Un émetteur et N récepteurs sur le canal**

**exemples : Ethernet, Token ring**

**perte de message pour R récepteurs avec  $0 \leq R \leq N$**

**d'une émission à l'autre perte sur des récepteurs différents**

**pas le même ordre sur tous les sites pour une suite de réceptions**

**pas de perception unique de l'ordre d'émission si émetteurs différents**

**durées de transfert variable**

**incertitudes sur l'état du canal, de l'émetteur et des récepteurs**

**incertitude sur la composition du groupe**

## LES SYSTÈMES RÉPARTIS

**Ensemble fini de sites interconnectés par un réseau de communication**

**Pas de mémoire commune**

**Pas d'horloge physique partagée par 2 processus ou plus**

### CHAQUE SITE

- **processeur, mémoire locale, mémoire stable (permanence des données en dépit de défaillances du processeur)**

### RÉSEAU DE COMMUNICATION

- **connexe : tout processus peut communiquer avec tous les autres**
- **communication et synchronisation entre processus par messages via le réseau de communication**

### DIFFICULTÉS CARACTÉRISTIQUES

#### OBSERVATIONS :

**deux observations faites sur deux sites distincts peuvent différer**

- **par l'ordre de perception des événements**
- **par leur date**

#### DÉCISIONS COHÉRENTES ENTRE PLUSIEURS SITES

- **visions différentes de l'état des ressources du système**
  - **pas d'état global de référence en temps réel au moment où il faut prendre des décisions**
  - **risque accru de défaillance (plus de composants)**
- => **la défaillance d'un élément n'est pas un événement rare**  
 => **importance des hypothèses sur les défaillances possibles**

**attention, à ne pas confondre avec les  
 SYSTÈMES PARALLÈLES CENTRALISÉS**

**encore appelés des multiprocesseurs à mémoire commune**

- **existence d'une mémoire commune (physique, réflexive, virtuelle)**
- **existence d'une horloge ou d'un rythme commun**

# LES MODELES DE LA COMMUNICATION ELEMENTAIRE

## MODÈLE DE COMMUNICATION SYNCHRONE FIABLE

### hypothèses les plus fortes

**Tout message arrive avant le délai  $d_{\max}$ , qu'il soit point à point ou diffusé.**

**Communication fiable : pas de perte de message, pas de panne de site**

**Réseau connexe : tout site peut communiquer avec tous les autres**

**parfois réseau isotrope : même délai  $d_{\max}$  pour tous les canaux**

**Les horloges des sites sont aussi synchronisées**

**(leur écart est borné par  $d_{h_{\max}}$ )**

**Les vitesses des processeurs sont supérieures à un minorant connu**

**Exemple : réseaux locaux avec heure reçue par radio sur chaque site,  
réseaux locaux industriels avec synchronisation d'horloge véhiculée par le  
réseau, réseaux locaux avec redondance des bus et des processeurs**

**• Mais cette hypothèse, valable avec une certaine probabilité, n'est pas toujours réaliste :**

**probabilité trop faible pour l'application,**

**probabilité variable dans le temps (cas des réseaux dont la charge instantanée est totalement imprévisible)**

**ELECTION EN COMMUNICATION SYNCHRONE FIABLE**

**Les sites  $S_1, S_2, \dots, S_i, \dots, S_N$  doivent élire un site coordonnateur**

**Chaque site  $S_i$  a un identificateur unique  $uid(i)$**

**et peut diffuser un message  $\langle \text{élection}, i, uid(i) \rangle$**

**Tous les sites démarrent l'élection en même temps à  $P$**

**(à dates fixes, par exemple, élection périodique)**

**Soit  $T = d_{\max} + dh_{\max}$ ,**

**Chaque site  $S_i$**

**(i) s'attend à recevoir un message d'élection à partir de  $P$**

**(ii) s'il n'a rien reçu au bout de  $T * uid(i)$  secondes, mesurées sur son horloge, il diffuse son message d'élection.**

**Résultat : le premier site qui diffuse son message est l' élu**

**Cet algorithme synchrone élit le site de plus petit  $uid$**

**commentaire : simple n'est-il pas? mais l'hypothèse synchrone est restrictive (pas de pannes, horloges communes) la "nature" est asynchrone.**

**ELECTION EN COMMUNICATION SYNCHRONE NON FIABLE**

**hypothèses : processeurs et canaux à silence sur défaillance**

**(pas de panne transitoire ou byzantine, mais silence après i, ii ou iii)**

**Chaque site  $S_i$**

**(i) à  $P$ , diffuse son  $uid(i)$  aux autres sites, lui compris**

**(ii) à  $P + T$ , il diffuse le  $\min(i)$  des  $uid(j)$  qu'il reçoit avant  $P + T$ ,**

**(iii) à  $P + 2*T$ , s'il note un consensus des  $\min(j)$  reçus (selon les cas, la majorité, ou l'unanimité des valeurs reçues), il élit la valeur de consensus (si l' élu est tombé en panne après sa phase i, il faut recommencer)**

## **MODÈLE DE COMMUNICATION ASYNCHRONE**

### **CHAQUE SITE**

- **processeur, mémoire locale, mémoire stable**  
(permanence des données en dépit de défaillances du processeur)

### **RÉSEAU DE COMMUNICATION**

- **connexe : tout processus peut communiquer avec tous les autres**
- **communication et synchronisation entre processus par messages via le réseau de communication**

### **PROPRIÉTÉS CARACTÉRISTIQUES**

- **Pas de mémoire commune**
- **Pas d'horloge physique partagée par 2 processus ou plus**
- **Absence de majorant connu sur le temps de transfert des messages**
- **Absence de minorant connu sur les vitesses des processeurs**

**Nota : appelé aussi modèle à délais non bornés**

**Exemple : cas de Internet et des réseaux longue distance (WAN)**

**PROBLÈME.** Un processus  $P_i$  ne peut pas savoir si un autre processus  $P_j$  est défaillant ou si la réponse qu'il attend de  $P_j$  est en préparation par  $P_j$  (processus très lent) ou encore en route (message très lent). En pratique, il est essentiel d'introduire une notion de temps (temps réel et non temps du processus  $P_i$ ) : combien de temps  $P_i$  doit-il attendre avant de suspecter  $P_j$  de défaillance

**suspecter une défaillance  $\neq$  détecter une défaillance**

## **MODELE DE COMMUNICATION ASYNCHRONE FIABLE**

**réseau connexe : tout site peut communiquer avec tous les autres**  
**communication fiable : pas de perte de message, pas de panne de site**  
**réaliste sur une courte durée**

### **PROPRIÉTÉ DE CAUSALITE ELEMENTAIRE**

**Par la nature physique de la communication, l'émission d'un message sur un site précède nécessairement la réception du message sur le site destinataire. Toute réception d'un message est causée par une émission antérieure. (il ne peut y avoir de réception spontanée de message)**

**Cette relation causale permet d'établir, dans un système réparti, une relation d'ordre partiel entre l'événement d'émission d'un message sur un site et l'événement de réception du message sur un autre site destinataire. cette relation se note (on lit précède) :**

$$\forall m, \text{EMISSION}(m) \rightarrow \text{RECEPTION}(m)$$

**(notée "happened before" par Lamport)**

**Plus exactement la relation est établie lorsque le message a été reçu**

$$\forall m, \text{RECEPTION}(m) \Rightarrow (\text{EMISSION}(m) \rightarrow \text{RECEPTION}(m))$$

## HYPOTHESES PROPRES A UN CANAL $C_{ij}$

(canal : liaison point à point entre un émetteur et un récepteur)

**DEFINITION :**  $m_1$  double  $m_2$  dans le canal  $C_{ij}$  si et seulement si

$$\text{EMISSION}_i(m_2) \rightarrow \text{EMISSION}_i(m_1)$$

$$\text{et } \text{RECEPTION}_j(m_1) \rightarrow \text{RECEPTION}_j(m_2)$$

### PROPRIÉTÉS D'ORDRE DES MESSAGES DANS LES CANAUX

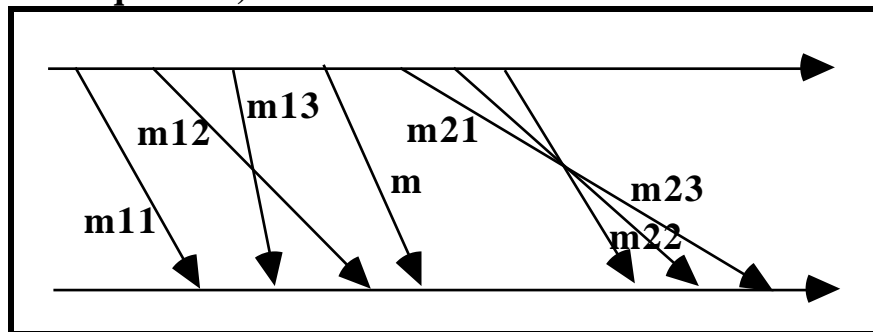
**1• un message marqueur  $m$  ne peut ni doubler ni être doublé sur  $C$**

$\forall m_1, \forall m_2,$

$$\text{EMISSION}(m_1) \rightarrow \text{EMISSION}(m) \Rightarrow \text{RECEPTION}(m_1) \rightarrow \text{RECEPTION}(m)$$

$$\text{EMISSION}(m) \rightarrow \text{EMISSION}(m_2) \Rightarrow \text{RECEPTION}(m) \rightarrow \text{RECEPTION}(m_2)$$

**2• un message ordinaire  $m$  n'impose pas de condition de réception, mais respecte celles des autres (il ne peut doubler les marqueurs et ne peut être doublé par les marqueurs ).**



**Tout marqueur  $m$  sépare les messages du canal en deux sous-ensembles**

$$\langle m = \{m_1 \mid \text{EMISSION}(m_1) \rightarrow \text{EMISSION}(m)\} \}$$

$$\text{et } m \text{ est un marqueur} \Rightarrow \text{RECEPTION}(m_1) \rightarrow \text{RECEPTION}(m)$$

$$\rangle m = \{m_2 \mid \text{EMISSION}(m) \rightarrow \text{EMISSION}(m_2)\} \}$$

$$\text{et } m \text{ est un marqueur} \Rightarrow \text{RECEPTION}(m) \rightarrow \text{RECEPTION}(m_2)$$

### TYPES DE COMPORTEMENT D'UN CANAL

**1• le moins contraint : tous les messages sont ordinaires**

**2• le plus contraint : tous les messages sont des marqueurs (canal FIFO)**

## RELATION DE PRÉCEDENCE ENTRE DES ÉVÉNEMENTS RÉPARTIS

**événement : instruction exécutée par un processeur**

(précédence : "happened before", L.Lamport, CACM 21,7, 1978)

a) A "précède" A' si A et A' sont des événements qui ont été générés dans cet ordre sur le même site S (ordre d'exécution local) :  $A \rightarrow A'$

b) A "précède" A' si A est l'événement d'émission d'un message M par le site P et que A' est l'événement de réception du message M sur Q :  $A \rightarrow A'$  (ordre causal pour chaque message)

La relation de précédence dans un système réparti est la fermeture transitive des deux relations précédentes.

si  $A \rightarrow B$  et  $B \rightarrow C$  alors  $A \rightarrow C$

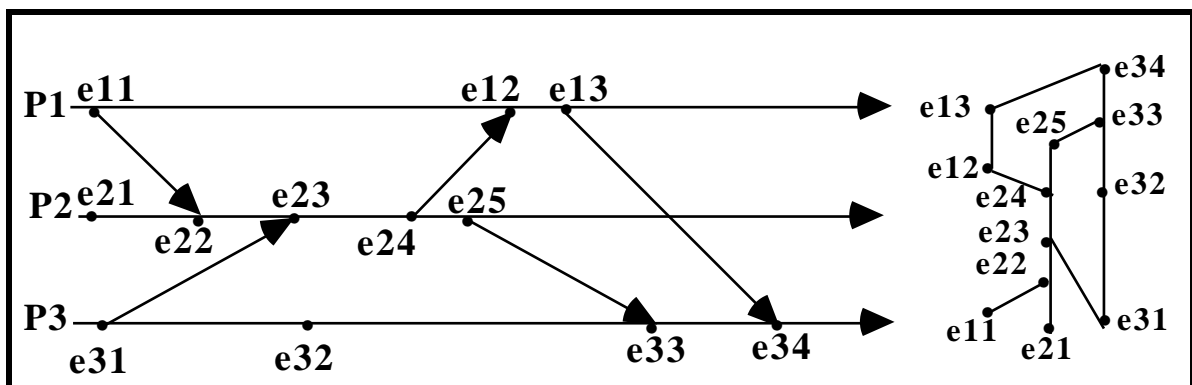
La précédence est un ordre partiel entre les événements du système réparti

La causalité entre événements implique la précédence entre eux :

$A \text{ cause } B \Rightarrow A \rightarrow B$

Une précédence entre événements exprime une causalité potentielle :

(si  $A \rightarrow B$ , A peut avoir influencé B).

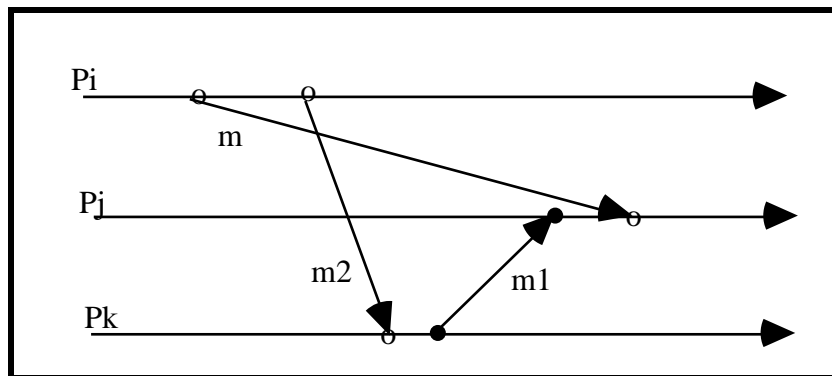


# DÉPENDANCE CAUSALE ENTRE MESSAGES

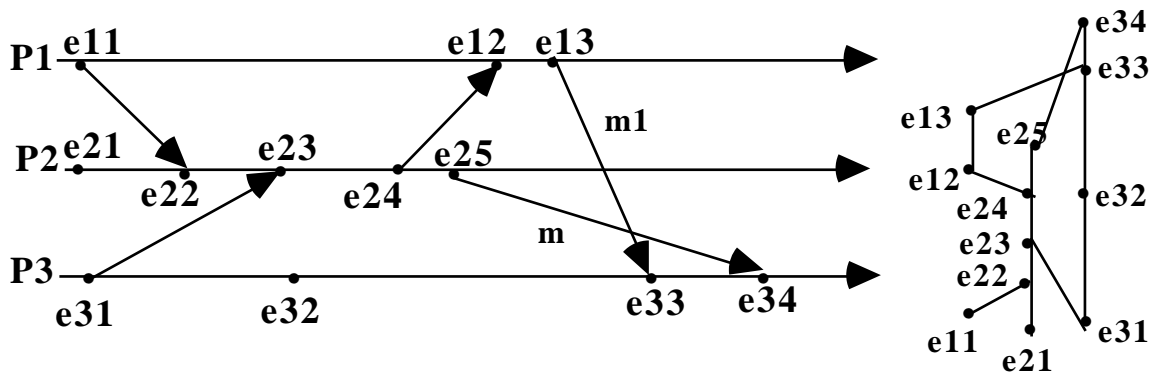
Les messages respectent la dépendance causale si et seulement si :

$\forall P_i, \forall P_j, \forall P_k, \forall m$  émis sur  $C_{ij}, \forall m_1$  émis sur  $C_{kj}$ ,

$EMISSION_i(m) \rightarrow EMISSION_k(m_1) \Rightarrow RECEPTION_j(m) \rightarrow RECEPTION_j(m_1)$



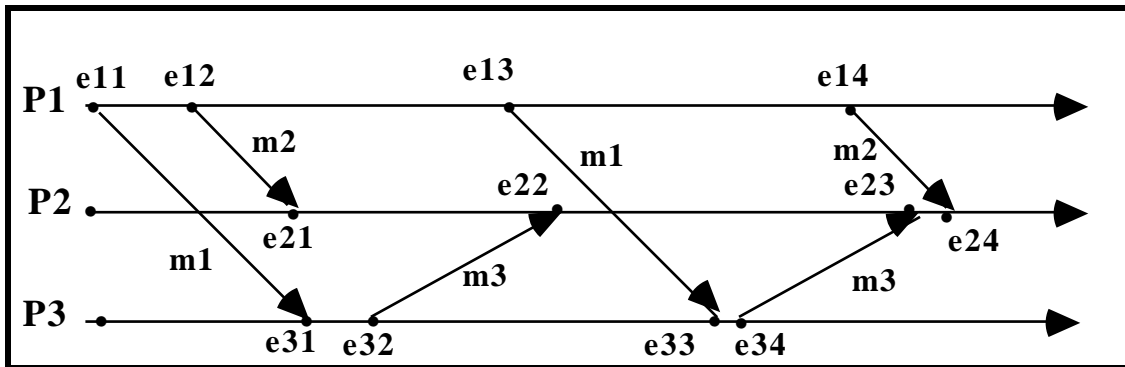
Premier exemple : la réception sur Pj ne respecte pas la dépendance causale



Deuxième exemple : la réception respecte la dépendance causale car les émissions e13 et e25 ne sont pas en relation de précédence

# DÉPENDANCE CAUSALE ENTRE MESSAGES

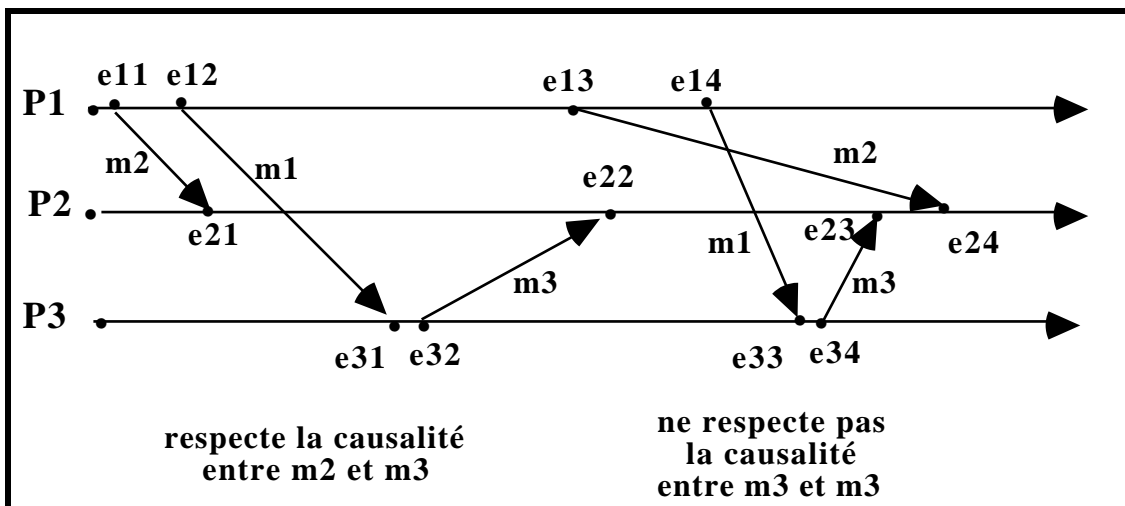
## Un problème de logique



pas de dépendance causale entre  $EMISSION_1(m2) \rightarrow EMISSION_3(m3)$   
 Donc il n'y a pas de dépendance causale entre m2 et m3

m1 : "Je vais demander à P2 de te rencontrer. Appelle le de ma part"

-----



$EMISSION_1(m2) \rightarrow EMISSION_3(m3) \Rightarrow RECEPTION_2(m2) \rightarrow RECEPTION_2(m3)$   
 Donc il y a dépendance causale entre m2 et m3

m1 : "J'ai demandé à P2 de te rencontrer. Fixe lui une date et un lieu"

## **MODELES DE DIFFUSION FIABLE ET COMMUNICATION DE GROUPE**

**Un message émis doit être reçu par n destinataires.**

### **MODELES DE COMMUNICATION**

**Communication inclusive ou non (l'émetteur reçoit le même message - le sien enrichi par le réseau- que les récepteurs)**

**Communication interne ou externe (l'émetteur, client du groupe, n'appartient pas au groupe)**

### **CLASSIFICATION SELON LES EMETTEURS ET LES RECEPTEURS (classification OSI)**

#### **MODE CENTRALISE ("multicast")**

**Un seul émetteur (toujours le même) et n récepteurs.**

#### **MODE CENTRALISE A CENTRE MOBILE**

**L'émetteur est unique par périodes.**

#### **MODE MULTI-CENTRE**

**N processus émetteurs peuvent à tout instant effectuer une diffusion vers P récepteurs.**

#### **MODE DECENTRALISE OU MODE CONVERSATION**

**Un ensemble de N sites peuvent être à tout instant émetteurs et sont tous destinataires des messages.**

# PROPRIETES D'ORDRE DANS LES GROUPES

## DIFFUSION RESPECTANT L'ORDRE LOCAL

**Pour deux diffusions successives du même processus, les messages sont délivrés dans le même ordre sur chaque site distant**

## DIFFUSION RESPECTANT L'ORDRE CAUSAL

**diffusion + causalité (Birman et Joseph 1987)**

**Relation de dépendance causale entre les messages,  
généralisée à la diffusion**

**Toute suite de diffusions de messages en relation de causalité implique la délivrance des messages sur tous les sites destinataires dans la même relation de causalité**

$\forall P_i, \forall P_k, \forall m$

**DIFFUSION<sub>i(m)</sub> précède causalement DIFFUSION<sub>k(m1)</sub>**

**$\Rightarrow$  RECEPTION<sub>j(m)</sub> précède RECEPTION<sub>j(m1)</sub> pour tout  $P_j$ .**

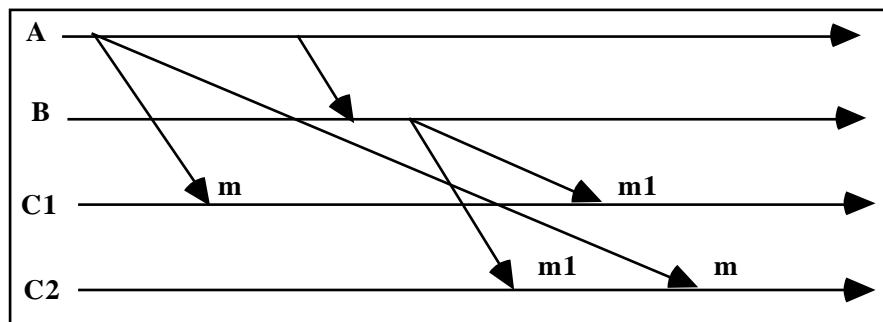
## EXEMPLES D'UTILISATION DE LA DIFFUSION CAUSALE

### Exemple 1:

A diffuse un courrier électronique m à C1 et C2 qui contient: "je demande à B de vous diffuser du travail par courrier électronique".

Pour respecter l'ordre causal, C1 et C2 ne doivent pas recevoir le courrier m1 de B avant celui m de A.

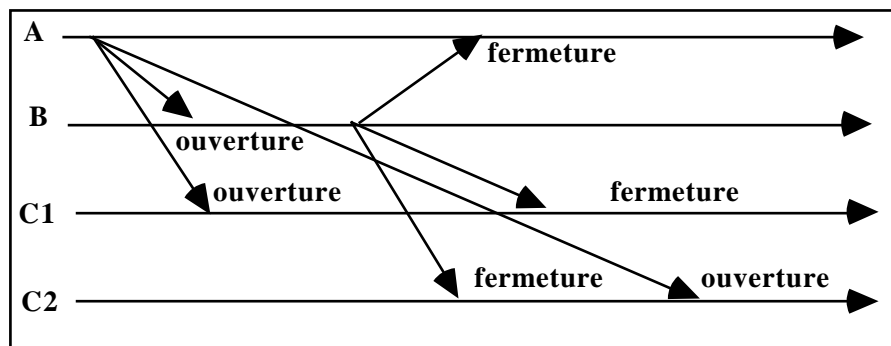
$$\text{émission}_A(m) \rightarrow \text{émission}_B(m1) \Rightarrow \text{réception}_C(m) \rightarrow \text{réception}_C(m1)$$



### Exemple 2:

A envoie à C1 et C2 une commande d'ouverture de vanne, en en rendant compte à B. Plus tard B envoie à C1 et C2 l'ordre de fermeture de la vanne, en en rendant compte à A.

Respect de la causalité : le message de A (ouverture) doit être enregistré avant celui de B (fermeture).



## DIFFUSION RESPECTANT UN ORDRE TOTAL SIMPLE

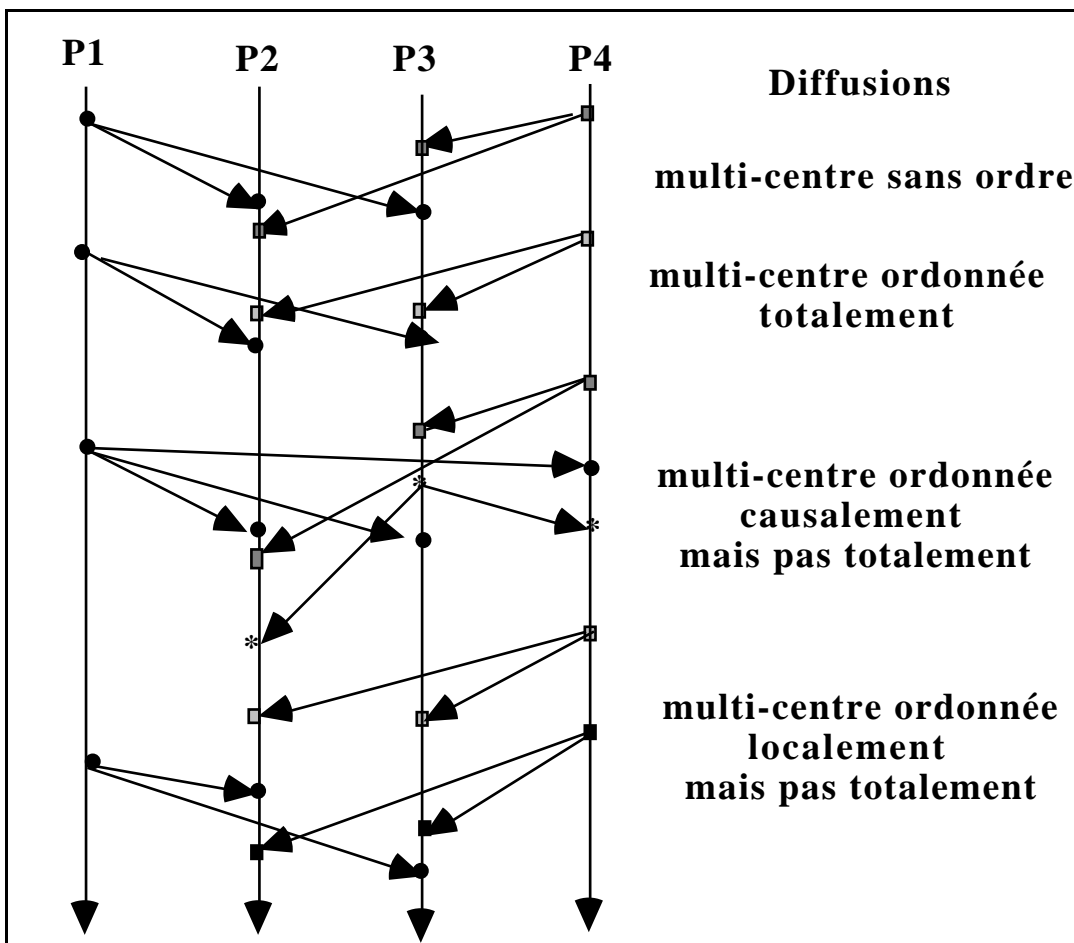
**Si plusieurs diffusions ont lieu concurremment de différents processus vers le même groupe de diffusion, alors tous les messages sont délivrés aux applications réceptrices dans le même ordre sur tous les récepteurs.**

**Exemple d'utilisation : copies multiples.**

**Le fait que toutes les opérations de modifications d'un ensemble de données en copies multiples soient effectuées dans le même ordre sur toutes les copies suffit à assurer le maintien de la cohérence (faible) des copies.**

**DIFFUSION RESPECTANT L'ORDRE TOTAL CAUSAL**

**L'ordre total respecte aussi la relation d'ordre causal entre messages**



## ETAT GLOBAL D'UN SYSTEME REPARTI

- **ETAT LOCAL  $EL_i$  D'UN SITE  $S_i$**   
état initial et séquence d'événements locaux sur  $S_i$
- **ETAT LOCAL  $EC_{ij}$  D'UN CANAL  $C_{ij}$**   
ensemble des messages en transit sur le canal  $C_{ij}$   
émis par  $S_i$  et non encore reçus par  $S_j$
- **EVENEMENTS (ATOMIQUES) FAISANT EVOLUER LE SYSTEME**  
 $\{EL_i\}$  événement interne sur  $S_i$   $\{EL'_i\}$   
 $\{EL_i, EC_{ij}\}$  émission de  $m$  par  $S_i$  sur  $C_{ij}$   $\{EL'_i, EC'_{ij} = EC_{ij} \cup \{m\}\}$   
 $\{EL_i, EC_{ki}\}$  réception de  $m$  par  $S_i$  sur  $C_{ki}$   $\{EL'_i, EC'_{ki} = EC_{ij} - \{m\}\}$
- **ETAT GLOBAL  $S = \{ \text{pour tout } i, j, \cup EL_i, \cup EC_{ij} \}$**
- **DIFFICULTES**  
 $EL_i$  n'est immédiatement observable que sur  $S_i$   
 $EC_{ij}$  n'est jamais directement observable, ni sur  $S_i$ , ni sur  $S_j$
- **OBJECTIF : définir un état global déterminable localement par tout site**

**REMARQUE : toute définition d'état doit respecter la dépendance causale**

**nota : on dit indifféremment site ou processus  
(dans ce cas il y a un processus par site)**

## PASSÉ D'UN ÉVÉNEMENT

**Le passé (ou historique) d'un événement  $e$ , c'est par définition :**

$$\text{hist}(e) = e \cup \text{ensemble des événements } e' \text{ tels que } e' \rightarrow e$$

**Seul le passé de  $e$  peut avoir influencé  $e$**

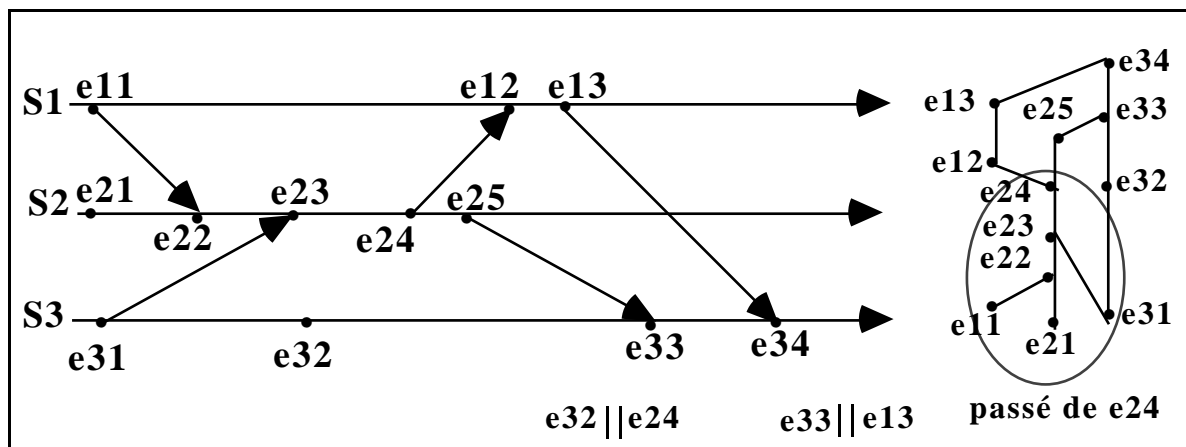
**Chaîne causale :  $e_0 \dots e_n$  tels que  $e_{i-1} \rightarrow e_i$  pour tout  $i \in (1..n)$**

**Événements concurrents (ou encore causalement indépendants)**

$$a \parallel b \Leftrightarrow \text{non } (a \rightarrow b) \text{ et non } (b \rightarrow a)$$

**aucun des 2 événements n'appartient au passé de l'autre**

**aucun des 2 événements ne peut influencer l'autre**



### Applications

**définition de la cohérence d'un état, d'une observation**

**mise au point répartie**

**mesure du parallélisme**

## COUPURES

soit  $E$  un ensemble d'événements constituant une application répartie

**Coupure** = sous-ensemble fini  $C$  de  $E$  tel que pour  $a, b \in E$

$a \in C$  et ( $b$  précède localement  $a$ )  $\Rightarrow$  ( $b \in C$ )

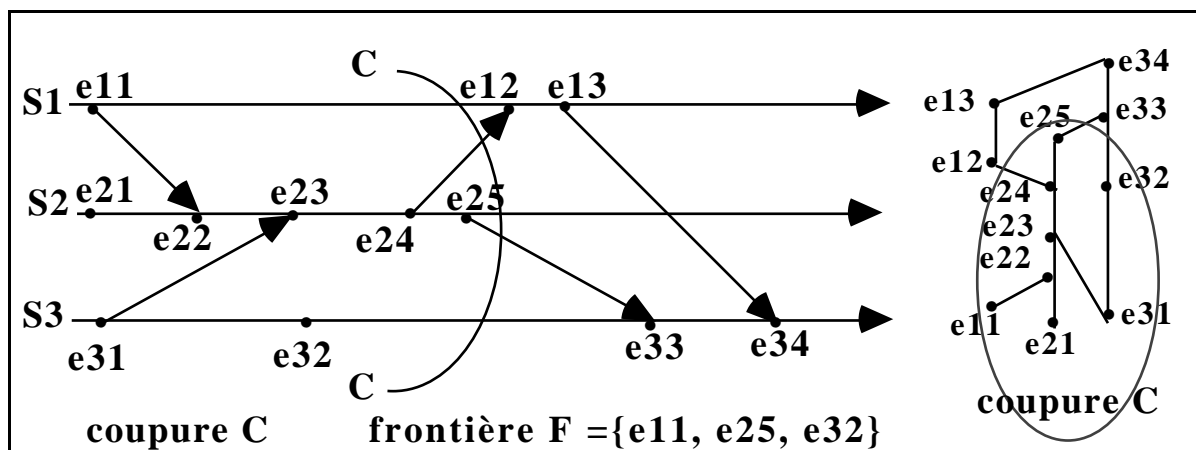
**Photographie instantanée d'un système** obtenue en prenant un événement par site et tous les événements du site qui le précède.

$C$  est un sous-ensemble de l'histoire de l'application qui contient toute l'histoire qui le précède : cela permet de définir un passé et un futur (par rapport à la coupure)

**Frontière  $F$  d'une coupure  $C$  :**

ensemble des événements les plus récents de la coupure, un par site

$a \in F \Leftrightarrow a \in C$  et il n'existe pas de  $b \in C$  tel que  $a \rightarrow b$



## COUPURES COHERENTES

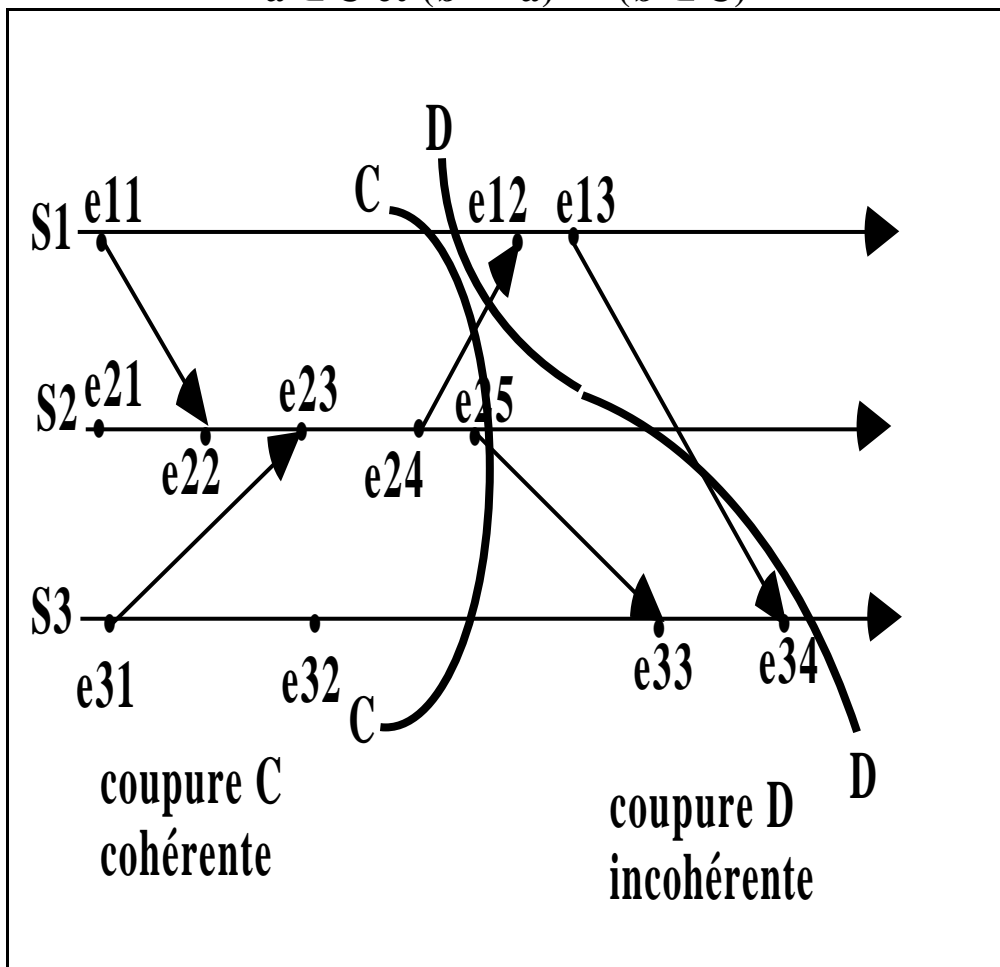
**Cohérence = respect de la causalité dans la coupure**

**une chaîne causale ne peut sortir et re-entrer dans la coupure**

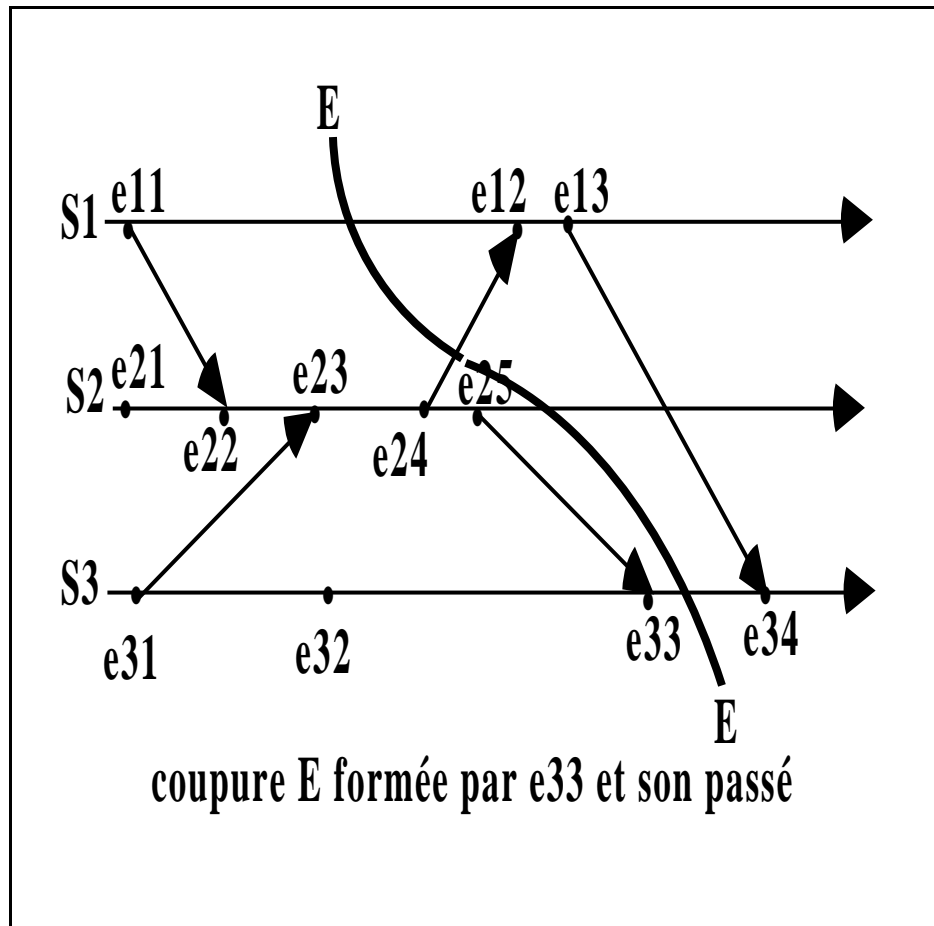
**un message ne peut pas venir du futur en franchissant la frontière**

**Coupure cohérente = coupure fermée par la relation de dépendance causale**

$$a \in C \text{ et } (b \rightarrow a) \Rightarrow (b \in C)$$



**Etat global cohérent = état associé à une coupure cohérente**



Exemple de coupure cohérente : le passé d'un événement

## ETAT GLOBAL COHÉRENT D'UN SYSTEME REPARTI

- soit  $EL_i$  état local du site  $S_i$  pour tout  $i$  (histoire locale de  $S_i$ )
- soit  $EC_{ij}$  état local du canal  $C_{ij}$  pour tout  $(i, j)$

état global  $S = \{ \text{pour tout } i, j, \cup EL_i, \cup EC_{ij} \}$

coupure associée =  $\{ \text{pour tout } i, \cup EL_i \}$

état global  $S$  cohérent si coupure associée cohérente

La frontière de la coupure associée définit un passé et un futur

**Il en résulte deux conditions nécessaires pour les messages  $m$  qui traversent une frontière de coupure**

**condition C1 :**

**si  $EMISSION_i(m) \in EL_i$  alors**

**soit  $RECEPTION_j(m) \in EL_j$ , soit  $m \in EC_{ij}$**

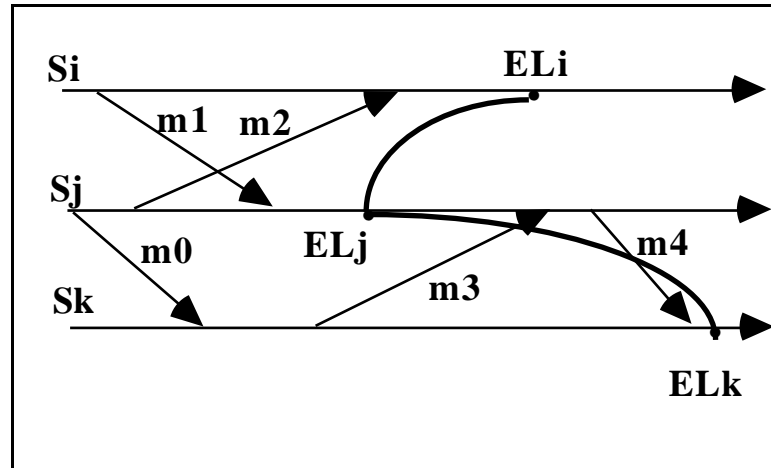
**tout message émis dans le passé est soit reçu dans le passé soit en transfert**

**condition C2 :**

**si  $EMISSION_i(m) \notin EL_i$  alors  $RECEPTION_j(m) \notin EL_j$**

**tout message émis dans le futur reste dans le futur**

## UNE EXECUTION REPARTIE



état global  $\{EL_i, EL_j, EL_k\}$

1•  $EMISSION_k(m_3) \in EL_k$  mais

comme  $RECEPTION_j(m_3) \notin EL_j$ ,

il faut que  $m_3 \in E_{Ckj}$

donc la condition C1 est vraie seulement si  $m_3 \in E_{Ckj}$

2•  $EMISSION_j(m_4) \notin EL_j$  mais  $RECEPTION_k(m_4) \in EL_k$

la condition C2 est fausse car

C2 : si  $EMISSION_j(m_4) \notin EL_j$  alors  $RECEPTION_k(m_4) \notin EL_k$

conclusion

$\{EL_i, EL_j, EL_k\}$  n'est pas un état global cohérent

# **DETERMINATION D'UN ETAT GLOBAL COHERENT**

**(Chandy - Lamport 1985)**

## **◆ Hypothèses**

- **le réseau de communication est connexe**
- **les canaux respectent l'ordre d'émission des messages (canaux FIFO)**
- **un site "élu" particulier lance la détermination d'état global**

## **◆ Principe**

- **association d'un message marqueur à chaque enregistrement d'état local d'un site pour que les autres sites puissent repérer les messages qui sont avant ou après cet enregistrement (les messages du passé et du futur)**
- **chaque site détermine son état local et celui de ses canaux en réception, puis envoie ces états au site "élu"**

## **◆ Propriétés**

- **l'algorithme se termine**
- **l'état global enregistré correspond à une coupure cohérente**
- **les états enregistrés des canaux sont corrects pour cette coupure**

## **SOLUTION DE CHANDY et LAMPORT (1985)**

### **◆ HYPOTHÈSE : CANAUX FIFO**

**tout message  $mk$  envoyé sur un canal FIFO sépare les messages du canal en deux sous-ensembles :  $<mk$  (avant  $mk$ ) et  $>mk$  (après  $mk$ ).**

### **◆ CONTRAINTES DE COHÉRENCE**

**• Quand  $S_i$  enregistre son état  $EL_i$ , toutes les émissions de messages faites par  $S_i$  avant  $EL_i$  sont captées dans  $EL_i$  et les réceptions de ces messages doivent être captées dans les  $EL_j$  et  $C_{ij}$  des sites  $S_j$  et celles-là seulement.**

### **◆ MISE EN OEUVRE**

**• Dès que  $S_i$  enregistre  $EL_i$ , il émet un message marqueur  $mk$  sur chaque canal  $C_{ij}$ .  $S_j$  doit enregistrer  $EL_j$  au plus tard à la réception de  $mk$  et  $S_j$  doit capter tous les messages  $<mk$ , soit dans  $EL_j$  soit dans  $C_{ji}$ . Les messages de  $>mk$  ne doivent pas être captés car ils viennent du futur de  $EL_i$  sur  $S_i$ .**

### **◆ ALGORITHME SUR CHAQUE SITE $S_i$**

**◆ Début ( $S_i$  "élu") ou première réception d'un marqueur  $mk$  (émis par  $S_j$ )**

**(1) enregistrer  $EL_i$**

**(2) enregistrer  $EC_{ji}$  comme vide car  $S_i$  a déjà reçu tous les messages  $<mk$  et ceux de  $>mk$  ne font pas partie de l'état global.**

**Au démarrage sur "élu", aucun  $EC_{ji}$  n'est enregistré.**

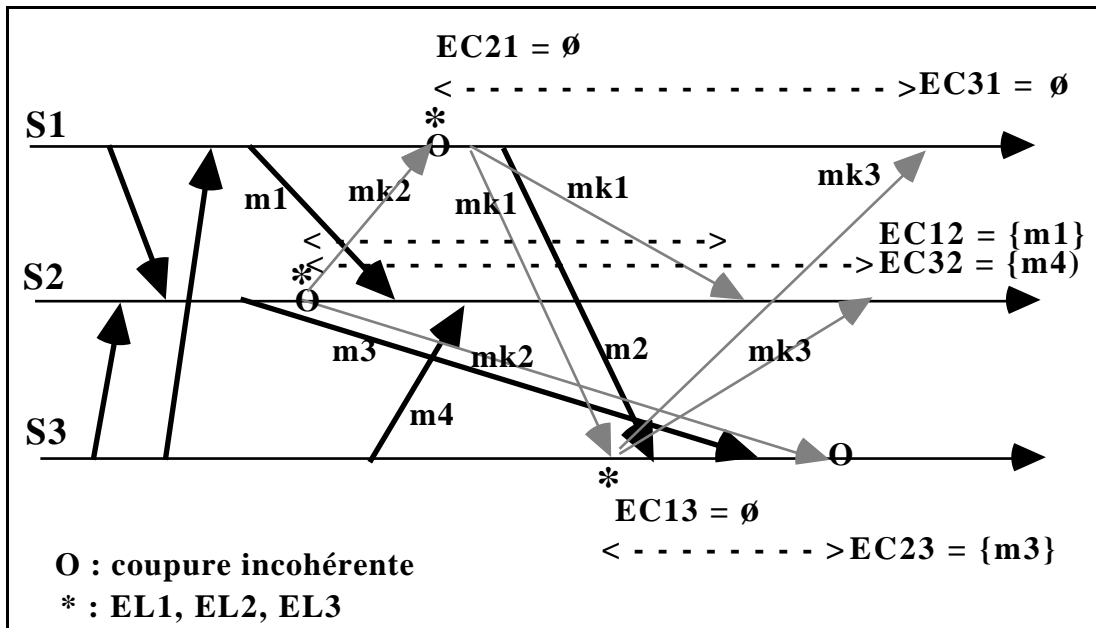
**(3) diffuser  $mk$  à tous ses voisins sur les canaux  $C_{ij}$**

**(1-2-3) doit être atomique**

**◆ Réceptions suivantes du marqueur  $mk$  (émis par  $S_j$ )**

**enregistrer  $EC_{ji}$  comme constitué des messages  $<mk$  reçus par  $S_i$  entre l'enregistrement de  $EL_i$  et l'arrivée de  $mk$  (envoyés par  $P_j$  avant  $EL_j$  mais pas reçus par  $P_i$  au moment de  $EL_i$ )**

# EXEMPLE DE DETERMINATION D'UN ETAT COHERENT

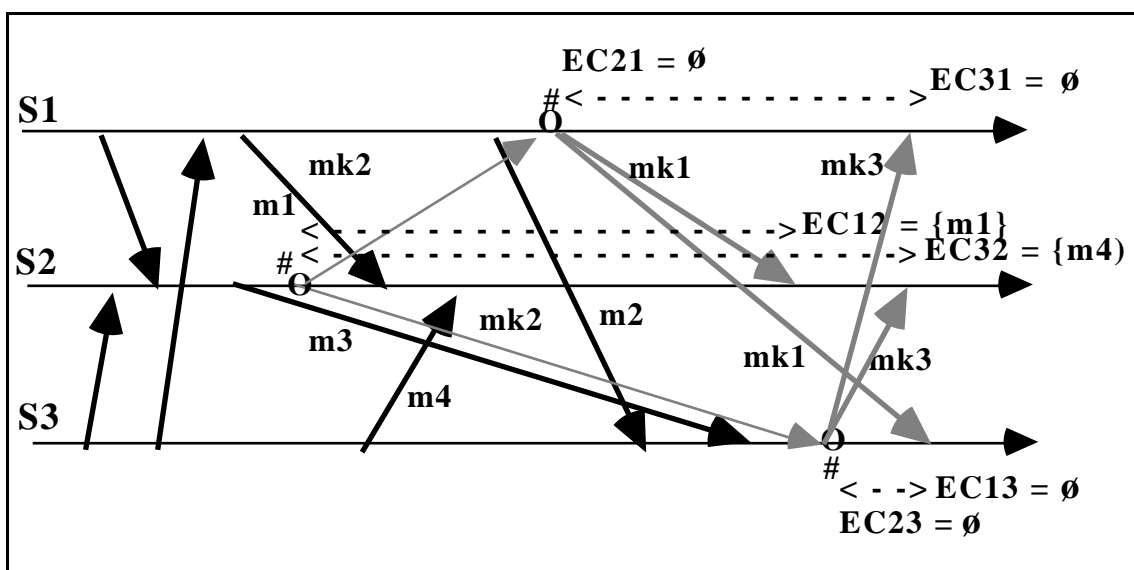


**S2 lance la détermination d'état global et diffuse mk2**

**Les 3 événements émission<sub>2</sub>(mk2), réception<sub>1</sub>(mk2) et réception<sub>3</sub>(mk2) déterminent une coupure incohérente.**

**Les marqueurs mk1 et mk3 permettent :**

- **de forcer la transitivité de la causalité et d'avoir une coupure cohérente.**
- **de noter dans les EC<sub>ij</sub> les messages qui traversent la coupure cohérente.**



**autre trace et autre état cohérent**

## **DETERMINER UN ETAT GLOBAL DANS UN SYSTEME REPARTI**

### **éléments de bibliographie**

#### **Solution pour un canal FIFO :**

**K.M. Chandy and L.Lamport, *Distributed Snapshots : Determining Global States of Distributed Systems*. ACM TOCS, Vol 3,1, (1985) pp. 63-75**

#### **solutions pour les canaux non FIFO**

**méthode cumulative (et rapide) de Lai et Yang : T.H.Lai and T.H.Yang, *On Distributed Snapshots*; Inf. Proc. Letters, Vol. 25, (1987), pp. 153-158**

**méthode non cumulative (mais lente) de Mattern : F.Mattern, *Virtual Time and Global States of Distributed Systems*. Proc. of Int. Workshop on Parallel and Dist. Systems, North Holland, 1988, pp. 215-226**

**solution utilisant des messages de contrôle : M.Ajuha, *Global Snapshots for Asynchronous Distributed Systems with non FIFO Channels*. Tec. Rep. #CS92-268, U. of Calif., San Diego (1992), 7 p.**

#### **solutions fondées sur l'ordre causal**

**méthode centralisée d'Acharya et Badrinath : A.Acharya and B.R.Badrinath, *Recording Distributed Snapshots Based on Causal Order of Message Delivery*. Inf. Proc. Letters, Vol. 44, (1992), pp.317-321**

**méthode répartie d'Alagar et Venkatesan : S.Alagar and S.Venkatesan, *An Optimal Algorithm for Distributed Snapshots with Causal Message Ordering*, Tec. Rep, U. of Texas, Dallas (1993), 7 p.**

#### **DOCUMENTATION UTILISEE**

**J.M.Helary, A.Mostefaoui, M.Raynal, *Déterminer un état global dans un système réparti*, RR. 2090, INRIA (1993), 21 p.**

## ENREGISTREMENT D'UNE TRACE

- **Objectifs :**

- **Reconstruire la séquence des messages échangés pour faire une analyse post mortem ou pour préparer la réexécution d'une situation à analyser.**

**Il faut donc conserver la dépendance causale entre les événements dépendants et indiquer les événements causalement indépendants.**

- **On date chaque événement  $e$  du système avec une méthode de datation causale (l'horloge causale). Soit  $D(e)$  la date ainsi fournie.**

**Elle permettra de reconstituer la trace si et seulement si :**

$$a \rightarrow b \Leftrightarrow D(a) < D(b)$$

$$a \parallel b \Leftrightarrow \text{non } (D(a) < D(b)) \text{ et non } (D(b) < D(a))$$

**C'est la condition forte des horloges car elle inclut :  $D(a) < D(b) \Rightarrow a \rightarrow b$**

**Rappel : condition pour que deux événements  $a$  et  $b$  soient concurrents, ou encore causalement indépendants :**

$$a \parallel b \Leftrightarrow \text{non } (a \rightarrow b) \text{ et non } (b \rightarrow a)$$

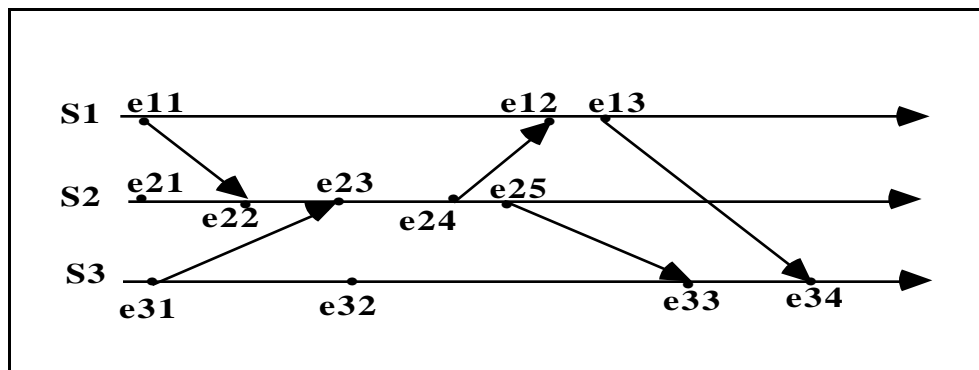
## UNE METHODE DE DATATION CAUSALE

### les historiques

**Rappel : passé d'un événement e**

- Le passé (ou historique) d'un événement e, c'est par définition :

$$\text{hist}(e) = e \cup \text{ensemble des événements } e' \text{ tels que } e' \rightarrow e$$



$$\text{hist}(e33) = \{e11 \ e25 \ e24 \ e23 \ e22 \ e21 \ e31 \ e32 \ e33\}$$

**Idee : utiliser le passé de e pour la datation car le passé permet de représenter la dépendance causale :**

$$a \rightarrow b \Leftrightarrow a \in \text{hist}(b)$$

$$a \parallel b \Leftrightarrow (a \notin \text{hist}(b)) \text{ et } (b \notin \text{hist}(a))$$

**Gros inconvénient : la taille de hist(e)**

**Remède : on observe que, pour définir hist(e), un événement par site suffit.**

# HORLOGES VECTORIELLES

(Fidge, Mattern 1988)

- **Projection de  $\text{hist}(e)$  sur  $S_i$  :**

$$\text{hist}_i(e) = \{ a \in \text{hist}(e) \mid a \in S_i \}$$

- **Propriété :  $e_{i,k} \in \text{hist}_i(e) \Rightarrow$  pour tout  $j < k : e_{i,j} \in \text{hist}_i(e)$**

Si on indice les événements de  $\text{hist}_i(e)$  et si un événement avec un indice  $k$  appartient à  $\text{hist}_i(e)$ , alors tous les événements d'indice inférieur à  $k$  font aussi partie de  $\text{hist}_i(e)$ .

- **Alors un seul entier suffit pour représenter  $\text{hist}_i(e)$ , c'est le nombre d'événements de  $\text{hist}_i(e)$ .**

Comme  $\text{hist}(e) = \cup \text{hist}_i(e)$ , on représente tout  $\text{hist}(e)$  avec un vecteur  $V(e)$

$$1 \leq i \leq n : V(e)[i] = k \text{ tel que } e_{i,k} \in \text{hist}_i(e) \text{ et } e_{i,k+1} \notin \text{hist}_i(e)$$

$$V(e)[i] = \text{nombre d'événements de } S_i \text{ "connus de } e"$$

(i. e. connus sur le site de  $e$  immédiatement après l'occurrence de  $e$ )  
(nombre d'événements de l'historique de  $e$  qui sont localisés sur  $S_i$ )

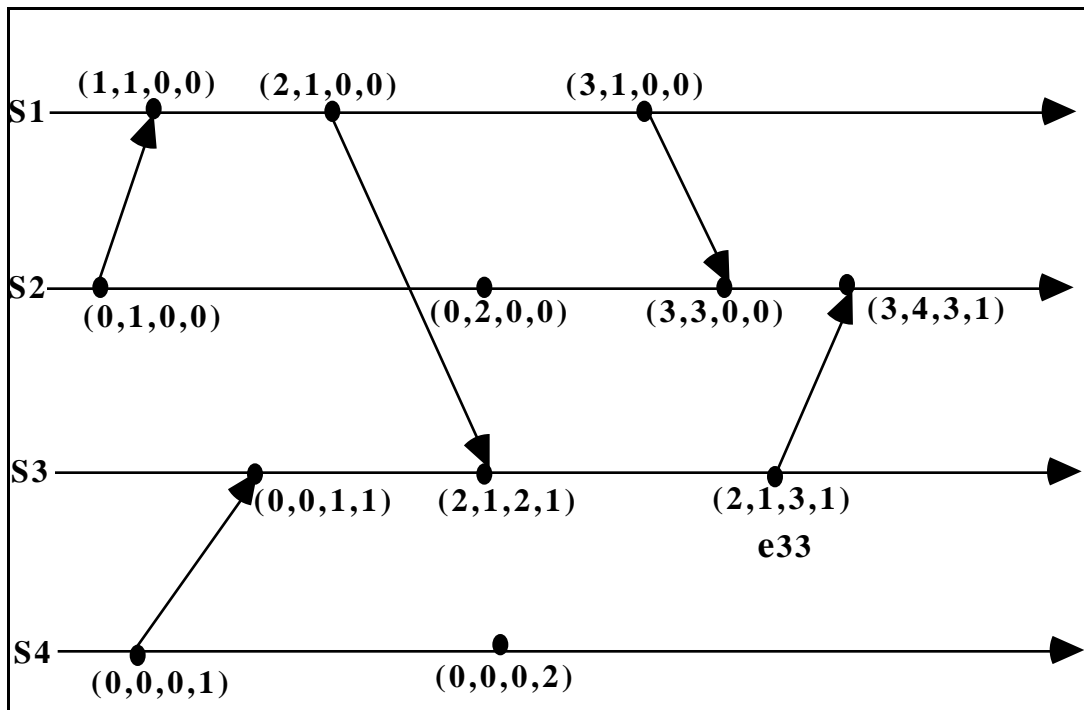
## REALISATION DES HORLOGES VECTORIELLES

On associe une horloge vectorielle  $V_i$  à chaque site  $S_i$

- Initialement  $V_i = (0, \dots, 0)$
- A chaque événement à dater local à  $S_i$ , on fait  $V_i[i] := V_i[i] + 1$
- Chaque message  $m$  porte une estampille  $V_m$  ( $V_m = V_i$  de l'émetteur)
- A la réception de  $(m, V_m)$  par un site  $S_i$ , on enrichit l'historique connu par  $S_i$  avec l'historique transporté par  $m$  :

$$V_i[i] := V_i[i] + 1$$

$$V_i[j] := \max(V_i[j], V_m[j]) \text{ pour tous } j = 1, \dots, n, j \neq i$$



- Autre façon de connaître l'"heure" de e33:

l'historique de e comprend

2 événements sur S1

1 événement sur S2

3 événements sur S3

1 seul événement sur S4

## PROPRIETES DES HORLOGES VECTORIELLES

- **Relation d'ordre partiel sur les horloges vectorielles :**

$V \leq V'$  défini par : quel que soit  $i$ ,  $V[i] \leq V'[i]$

$V < V'$  défini par  $V \leq V'$  et  $V \neq V'$

$V \parallel V'$  défini par  $\neg (V < V')$  et  $\neg (V' < V)$

Les horloges vectorielles représentent exactement la dépendance causale :  
pour tout  $a, b$

$$a \rightarrow b \Leftrightarrow V(a) < V(b)$$

$$a \parallel b \Leftrightarrow V(a) \parallel V(b)$$

- **Les horloges vectorielles sont "denses" :**

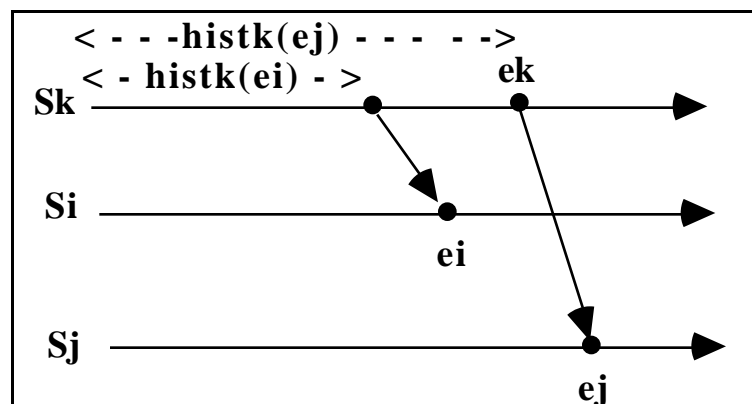
Soit  $e_i \in S_i$ ,  $e_j \in S_j$ ,

si  $V(e_i)[k] < V(e_j)[k]$ , pour  $k \neq j$ , alors il existe  $e_k$  sur  $S_k$  tel que

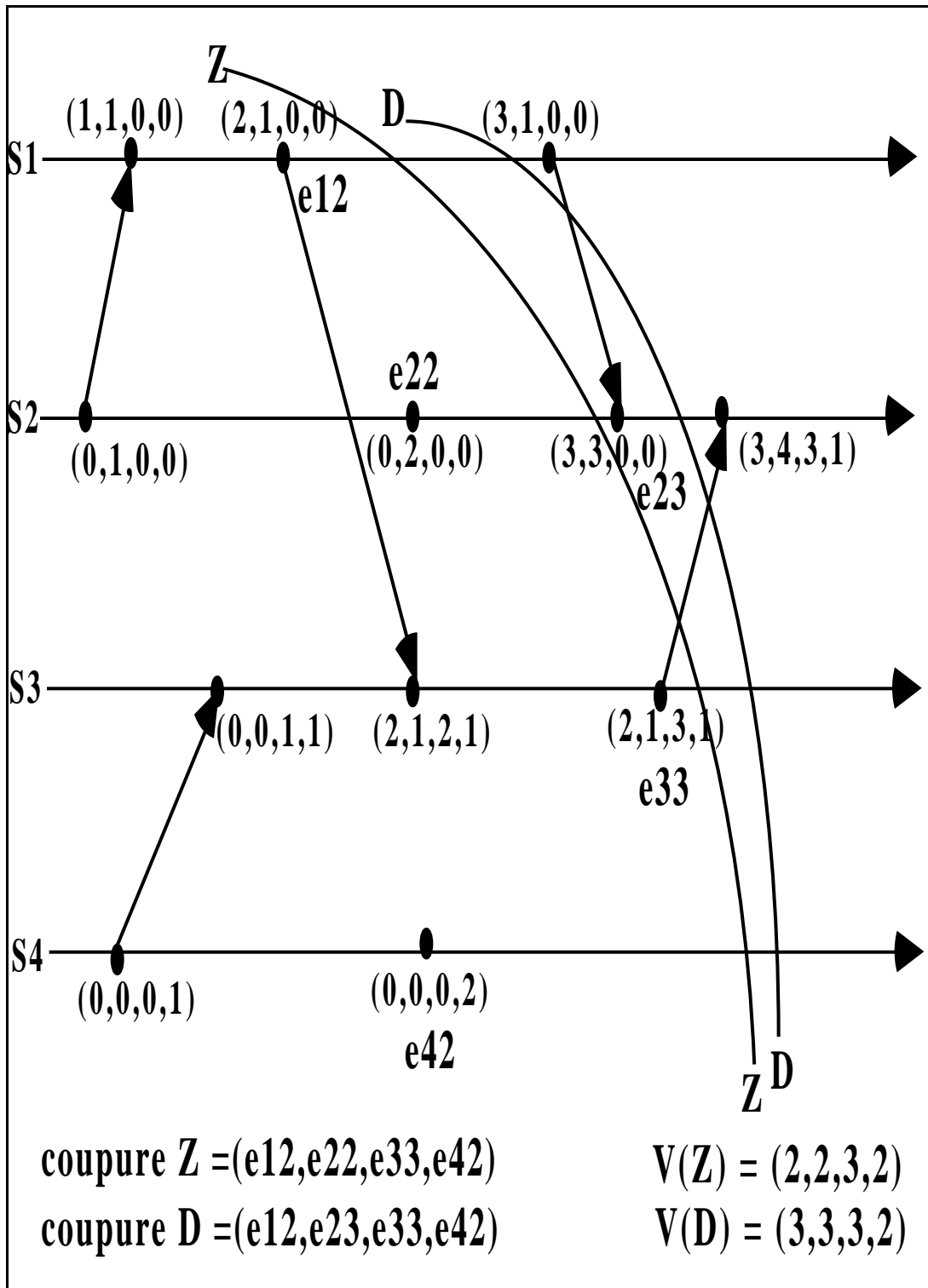
$$\neg (e_k \rightarrow e_i) \text{ et } (e_k \rightarrow e_j)$$

En effet  $V(e_i)[k] = |\text{hist}_k(e_i)| = \text{nombre d'événements de } \text{hist}(e_i) \text{ sur } S_k$

$V(e_j)[k] = |\text{hist}_k(e_j)| = \text{nombre d'événements de } \text{hist}(e_j) \text{ sur } S_k$



## HORLOGES VECTORIELLES ET COUPURES COHERENTES



- Date d'une coupure  $C = (c1, c2, \dots, cn)$

$$V(C) = \sup (V(c1), \dots, V(cn))$$

$$\text{soit pour tout } i, \quad V(C)[i] = \sup (V(c1)[i], \dots, V(cn)[i])$$

**La coupure est cohérente si et seulement si**

$$V(C) = (V(c1)[1], \dots, V(cn)[n])$$

$$V(Z) = (2, 2, 3, 2) = (V(e12)[1], V(e22)[2], V(e33)[3], V(42)[4])$$

$$V(D) = (3, 3, 3, 2)$$

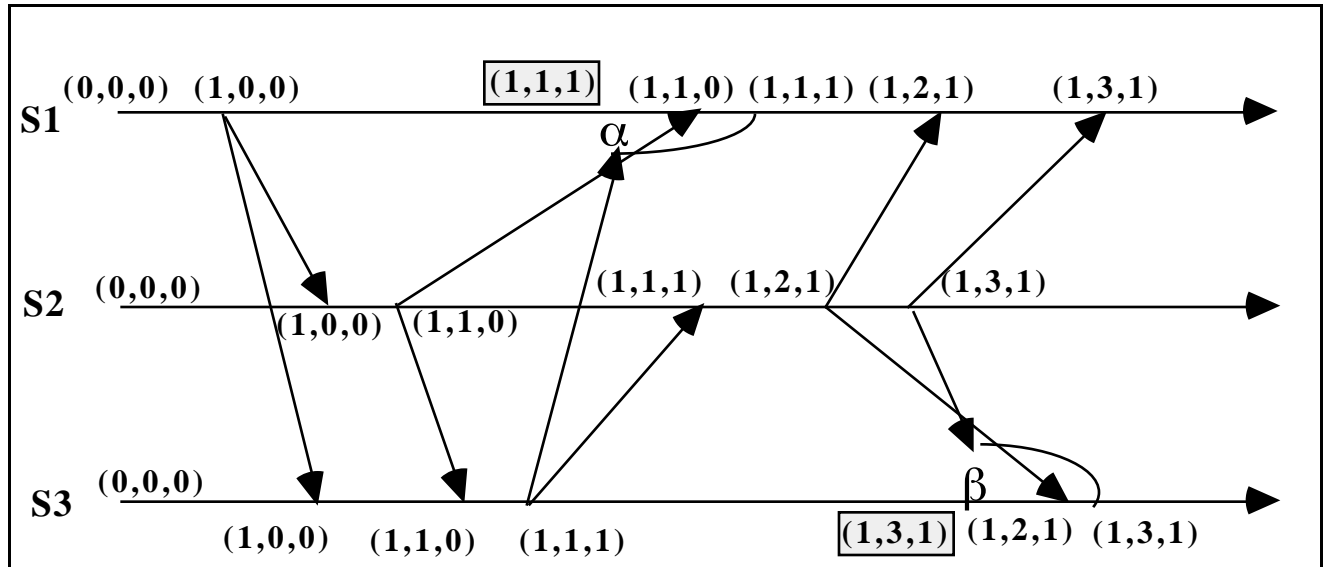
$$(V(e12)[1], V(e23)[2], V(e33)[3], V(42)[4]) = (2, 3, 3, 2)$$

- **Z est cohérente, D ne l'est pas**

## **DIFFUSION AVEC ORDRE CAUSAL**

**(Birman, Schiper, Stephenson 1990)**

- **Utilisation des horloges vectorielles pour garantir la causalité**
- **Si un message arrive trop tôt pour la causalité, on le fait attendre**
- **On ne compte que les diffusions :**
  - 1) **avant diffusion de m, le site Si exécute  $V_i[i] := V_i[i] + 1$**
  - 2) **estampille de m par Si :  $V_m = V_i$**
  - 3) **à la réception sur Sj de (m,  $V_m$ ) diffusé par Si, on attend que :**
    - a) **toutes les diffusions précédentes de Si soient arrivées sur Sj**  
soit  $V_j[i] = V_m[i] - 1$
    - b) **toutes les diffusions antérieures à m et reçues sur Si aient été aussi reçues par Sj**  
soit pour tous  $k \neq i, V_j[k] \geq V_m[k]$
  - 4) **après remise de m, on enregistre l'historique connu grâce à m**  
soit  $V_j := \max (V_j, V_m)$



en  $\alpha$  :  $V1 = (1,0,0)$  et  $Vm = (1,1,1)$  soit  $V1[3] = Vm[3] - 1$ ;  $V1[2] < Vm[2]$

en  $\beta$  :  $V3 = (1,1,1)$  et  $Vm = (1,3,1)$  soit  $V3[2] \neq Vm[2] - 1$ ;  $V3[1] \geq Vm[1]$

## LINEARISATION DE L'OBSERVATION D'UN SYSTEME

- Linéarisation de l'observation d'un système réparti S:

- "vue" en séquence des événements de S par un observateur interne
- "vue" en séquence des événements de S par un observateur externe

- C'est la définition d'un ordre total, soit  $\ll$ , sur les événements de S

- Linéarisation valide si elle est compatible avec la relation de précédence causale

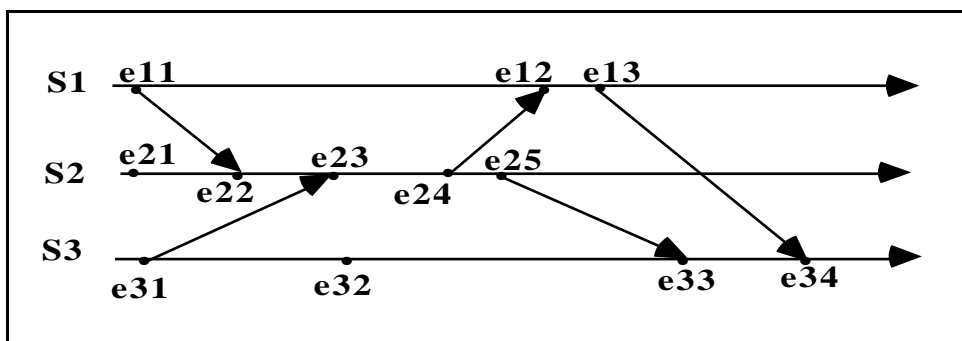
$$e, e' \in S : e \rightarrow e' \Rightarrow e \ll e'$$

- Exemple :

e11 e21 e31 e22 e23 e32 e24 e25 e12 e33 e13 e34 valide

e11 e21 e31 e22 e32 e23 e24 e25 e12 e33 e13 e34 valide (car e32 || e23)

e11 e21 e31 e22 e23 e32 e24 e25 e12 e33 e34 e13 invalide (car e13  $\rightarrow$  e34)



- On date chaque événement e du système avec une méthode de datation totale (l'horloge logique). Soit H(e) la date ainsi fournie qui est valide ssi:

$$e \rightarrow e' \Rightarrow H(e) < H(e')$$

Nota.

$$H(e) < H(e') \Rightarrow \text{non}(e' \rightarrow e)$$

C'est la condition faible des horloges car, ou bien  $e \rightarrow e'$ , ou bien  $e || e'$

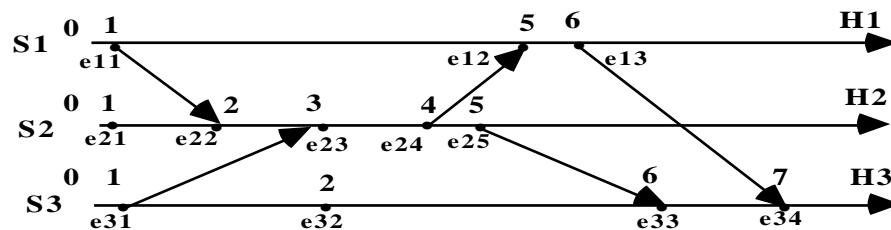
# REALISATION DES HORLOGES LOGIQUES

(Lamport 1978)

- **Objectif : réaliser une datation totale des événements**
  - respectant la dépendance causale
  - déterminable par consultation locale

Un compteur entier  $H_i$ , initialisé à 0, est maintenu sur chaque site  $S_i$ .

- A chaque événement  $e$ , localisé sur  $S_i$  :
  - $H_i := H_i + 1$
  - $e$  est daté par  $H_i$  :  $H(e) := H_i$
- Si  $e$  est l'émission d'un message  $m$ , celui-ci est estampillé par la date de son émission sur  $S_i$ , obtenue en lisant  $H_i$  :
  - $E(m) = H(\text{émission}(m)) := H_i$
- Si  $e$  est la réception d'un message  $m$  venant de  $S_j$ , l'estampille  $E(m)$  est utilisée par le compteur  $H_i$  (l'horloge logique locale sur  $S_i$ ) pour rattraper le retard sur  $H_j$ , s'il en a, avant de dater d'incrémenter  $H_i$  pour dater  $e$ .
  - $H_i := \max(H_i, E(m)) + 1$



- **Réalisation d'un ordre total ( $\ll$ ) :**
  - soit  $a$  sur  $S_i$ ,  $b$  sur  $S_j$
  - donc  $H(a) = H_i(a)$  et  $H(b) = H_j(b)$
  - $a \ll b \Leftrightarrow (H(a) < H(b)) \text{ ou } (H(a) = H(b) \text{ et } i < j)$
  - l'ordre total sur les sites est arbitraire (il doit être le même partout)

## EVALUATION DES HORLOGES LOGIQUES

### ◆ Hypothèses de validité

- le réseau de communication est connexe et fiable
- le temps de transmission des messages est borné supérieurement
- pas de communication cachée qui donneraient un autre ordre total  
(via des canaux comme le téléphone entre utilisateurs,  
ou comme une heure universelle fournie par satellite).

### ◆ Avantages

- première datation répartie introduite
- économique : datation par un seul nombre et non par un vecteur
- causalité des messages respectée par remise à l'heure du récepteur

### ◆ Utilisation importante des estampilles

- ordre total :  $e_{11} e_{21} e_{31} e_{22} e_{32} e_{23} e_{24} e_{12} e_{25} e_{13} e_{33} e_{34}$
- exclusion mutuelle répartie
- mise à jour de copies multiples
- diffusion fiable ordonnée totalement
- datation des transactions réparties pour gérer les verrous des BD
- datation des transactions réparties pour la prévention d'interblocage
- gestion cohérente de fichiers répartis
- génération répartie de noms uniques pour la désignation interne
- génération répartie de noms uniques pour l'authentification

### ◆ Limitation de la datation par estampilles

- la notion de dépendance causale est effacée artificiellement, car
- les estampilles ne sont pas denses : si  $H(e) < H(e')$ ,  
on ne peut pas savoir s'il existe  $e''$  tel que  $e \rightarrow e''$  ou  $e'' \rightarrow e'$

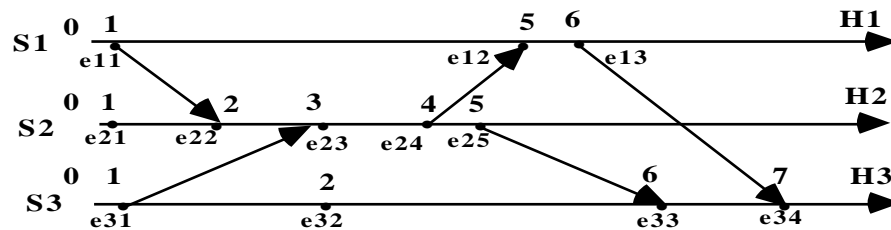
# LIMITATION DE LA DATATION PAR HEURE LOGIQUE

$$e \rightarrow e' \Rightarrow H(e) < H(e')$$

• c'est la condition faible des horloges car

(R)  $H(e) \leq H(e') \Rightarrow \text{non}(e' \rightarrow e)$        $[(a \Rightarrow b) \Leftrightarrow (\text{non } b \Rightarrow \text{non } a)]$

c'est à dire : ou bien  $e \rightarrow e'$ , ou bien  $e \parallel e'$



a) •  $e_{11} \rightarrow e_{33}$  se traduit en  $H_1(e_{11}) < H_3(e_{33})$        $e_{11} \ll e_{33}$

•  $e \parallel e'$  donne n'importe quoi :

$e_{32} \parallel e_{12}$  donne  $H_3(e_{32}) < H_1(e_{12})$        $e_{32} \ll e_{12}$

$e_{32} \parallel e_{22}$  donne  $H_3(e_{32}) = H_2(e_{22})$        $e_{32} \gg e_{22}$

$e_{32} \parallel e_{11}$  donne  $H_3(e_{32}) > H_1(e_{11})$        $e_{32} \gg e_{11}$

b) •  $H(e_{22}) < H(e_{33}) \Rightarrow \text{non}(e_{33} \rightarrow e_{22})$       ici  $e_{22} \rightarrow e_{33}$

•  $H(e_{12}) < H(e_{33}) \Rightarrow \text{non}(e_{33} \rightarrow e_{12})$       ici  $e_{32} \parallel e_{12}$

c) • seule certitude       $H(e) = H(e') \Rightarrow e \parallel e'$

$H(e) = H(e') \Leftrightarrow (H(e) \leq H(e')) \text{ et } (H(e) \geq H(e'))$

$(H(e) \leq H(e')) \text{ et } (H(e) \geq H(e')) \Rightarrow \text{non}(e' \rightarrow e) \text{ et } \text{non}(e \rightarrow e')$  [par (R)]  
 $\Rightarrow e \parallel e'$       CQFD

exemples  $H(e_{13}) = H(e_{33}) = 6$  . On a bien  $e_{13} \parallel e_{33}$

$H(e_{22}) = H(e_{32}) = 2$  . On a bien  $e_{22} \parallel e_{32}$

• Les estampilles ne sont pas denses : soit  $e$  et  $e'$  tels que  $H(e) < H(e')$ , on ne peut pas savoir s'il existe  $e''$  tel que  $e \rightarrow e''$  et/ou  $e'' \rightarrow e'$  dans l'exemple :  $H(e_{22}) = 2$ ;  $H(e_{32}) = 2$ ;  $H(e_{33}) = 6$

$H(e_{22}) < H(e_{33})$  et il existe  $e_{24}$  tel que  $e_{22} \rightarrow e_{24}$  et  $e_{24} \rightarrow e_{33}$

$H(e_{32}) < H(e_{33})$  et il n'existe pas  $e''$  tel que  $e \rightarrow e''$  et/ou  $e'' \rightarrow e'$

## EXCLUSION MUTUELLE REPARTIE (Lamport 1978)

### ◆ Hypothèses

- le nombre  $N$  des sites est connu
- les canaux sont FIFO
- ordre total réalisé par horloge logique

### ◆ Principe : connaissance mutuelle répartie acquise par chaque site"

### ◆ Demande d'entrée d'un site $S_i$ en section critique :

diffusion de  $(req, H(req), i)$  à tous les sites,  $S_i$  compris  
entrée en section critique quand  $S_i$  sait que:

- a) tous les sites ont reçu sa demande ou qu'ils en ont émis une aussi
- b) sa demande est la plus ancienne de toutes

### ◆ Réception par $S_i$ d'une demande d'entrée en section critique de $S_j$ : réponse systématique par $(acq, H(acq), S_i)$

### ◆ Libération de la section critique par $S_i$ :

diffusion de  $(lib, H(lib), S_i)$  à tous les sites,  $S_i$  compris

### ◆ Structure de données sur chaque site : tableau de $N$ messages, 1 par site

- initialement sur  $S_i$   $M_{ij} := (lib, 0, S_j)$  pour tout  $j$
- réception de  $M = (req, H(req), j)$  ou  $(lib, H(lib), S_j) \Rightarrow$   
 $M_{ij} := M$
- réception de  $M = (acq, H(acq), S_j) \Rightarrow$   
si  $M_{ij} \neq (req, H(req), j)$  alors  $M_{ij} := M$

si l'acquittement vient après une requête de  $S_j$ , on n'oublie pas celle-ci

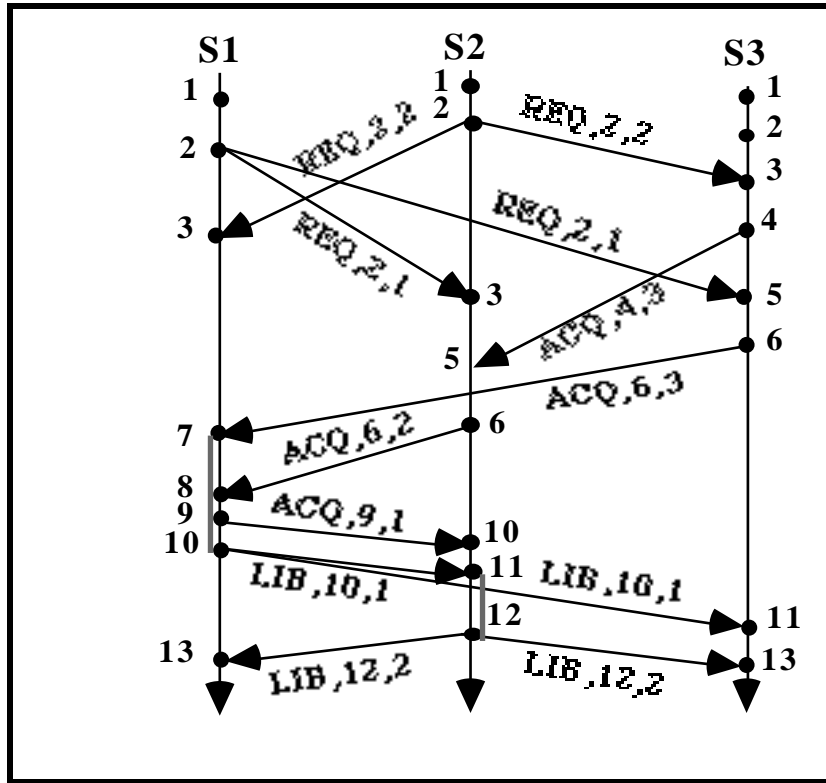
### ◆ Règle de décision pour chaque site $S_i$ :

- ◆◆  $S_i$  entre en section critique quand sa demande  $M_{ii} \ll M_{ij}$  pour tout  $j$   
En effet, comme les canaux sont FIFO,  
il n'y a plus de requête plus ancienne en chemin dans un canal

### ◆ Propriétés :

- $S_i$  est seul en section critique.
- Il n'y a pas de coalition car les entrées suivent l'ordre total
- Mais il faut  $3(n - 1)$  messages

## EXCLUSION MUTUELLE REPARTIE (Lamport 1978) EXEMPLE



|                              |         | H1 |     |          | H2 |     |          | H3 |
|------------------------------|---------|----|-----|----------|----|-----|----------|----|
| M11                          | LIB,0,1 | 0  | M21 | LIB,0,1  | 0  | M31 | LIB,0,1  | 0  |
| M12                          | LIB,0,2 | 0  | M22 | LIB,0,2  | 0  | M32 | LIB,0,2  | 0  |
| M13                          | LIB,0,3 | 0  | M23 | LIB,0,3  | 0  | M33 | LIB,0,3  | 0  |
|                              |         |    |     |          |    |     |          |    |
| M11                          | REQ,2,1 | 2  | M21 | REQ,2,1  | 3  | M31 | REQ,2,1  | 5  |
| M12                          | REQ,2,2 | 3  | M22 | REQ,2,2  | 2  | M32 | REQ,2,2  | 3  |
| M13                          | ACQ,6,3 | 7  | M23 | ACQ,4,3  | 5  | M33 | LIB,0,3  | 0  |
|                              |         |    |     |          |    |     |          |    |
| S1 entre en s.c.<br>à H1 = 7 |         |    | M21 | LIB,10,1 | 11 | M31 | LIB,10,1 | 11 |
|                              |         |    | M22 | REQ,2,2  | 2  | M32 | LIB,12,2 | 13 |
|                              |         |    | M23 | ACQ,4,3  | 5  | M33 | LIB,0,3  | 0  |

**S2 entre en SC à H2 = 11**

## **EXCLUSION MUTUELLE REPARTIE**

**(Ricart, Agrawala 1981)**

### **◆ Hypothèses**

- le nombre  $N$  des sites est connu
- les canaux sont quelconques (hypothèse FIFO non nécessaire)
- ordre total réalisé par horloge logique

### **◆ Principe : file d'attente répartie par morceau sur certains sites**

### **◆ Demande d'entrée d'un site $S_i$ en section critique :**

diffusion de  $(req, H(req), i)$  à tous les sites,  $S_i$  compris

entrée en section critique quand tous les sites ont donné leur accord

### **◆ Réception par $S_i$ d'une demande d'entrée en section critique de $S_j$ :**

- Si n'est ni en section critique ni candidat : accord  $(acc, H(acc), S_i)$
- Si est lui-même candidat :

accord  $(acc, H(acc), S_i)$  si demande de  $S_j$  antérieure

sinon mise en attente par  $S_i$  de la demande de  $S_j$

- Si est en section critique :

mise en attente par  $S_i$  de la demande de  $S_j$

### **◆ Libération de la section critique par $S_i$ :**

accord  $(acc, H(acc), S_i)$  à toutes les demandes mises en attente par  $S_i$

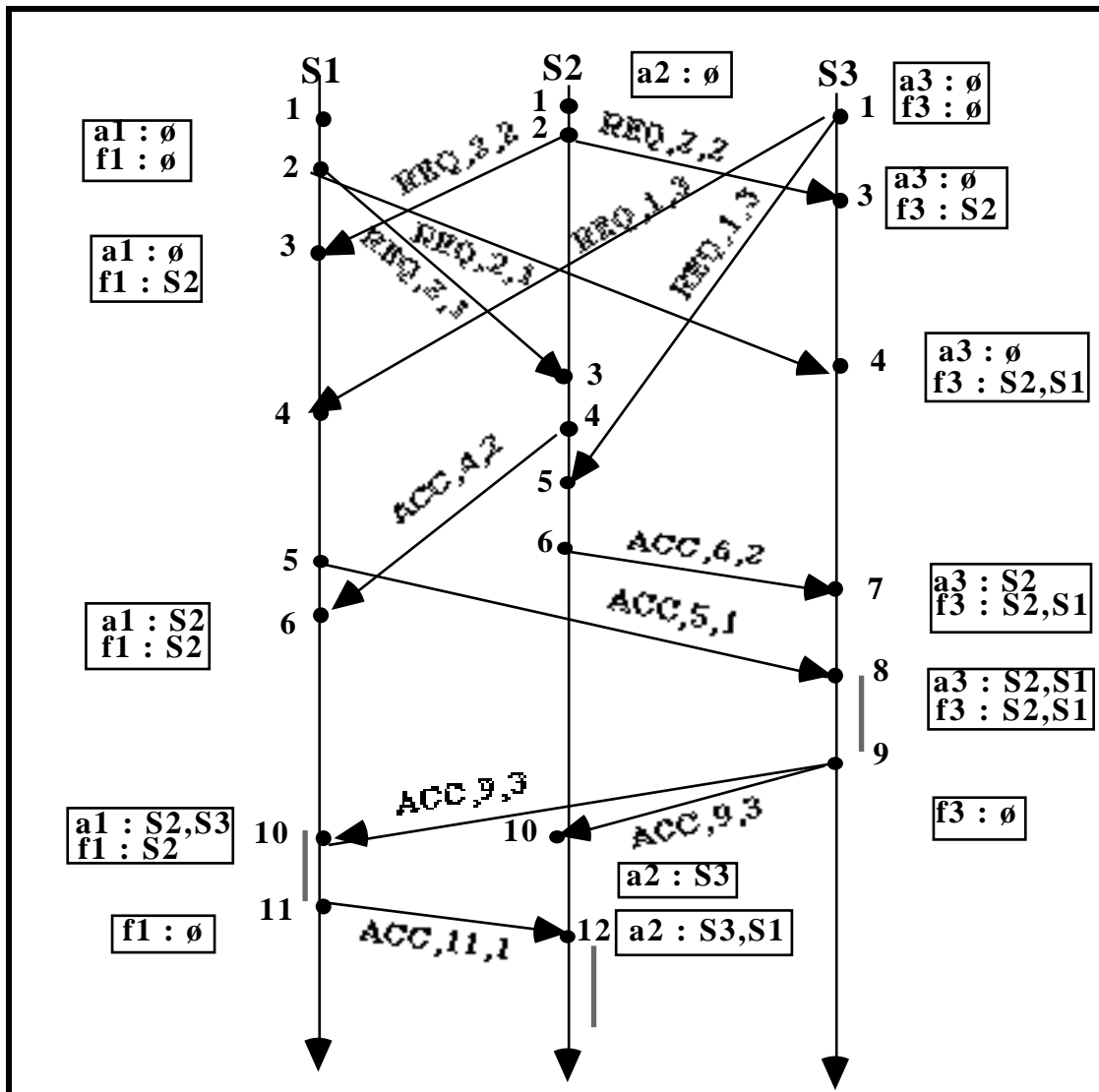
### **◆ Propriétés :**

- Si est seul en section critique.
- Il n'y a pas de coalition car les entrées en S.C. suivent l'ordre total (équité faible car les messages peuvent se doubler)
- Il faut  $2(n - 1)$  messages

## EXCLUSION MUTUELLE REPARTIE

(Ricart, Agrawala 1981)

### EXEMPLE



**légende :**     $a_i$  : ensemble des ACC reçus par le site  $i$   
                   $f_i$  : ensemble des sites mis en attente par  $S_i$

## **POSE DE POINTS DE REPRISE RÉPARTIS**

- \* **tolérance aux pannes et reprise arrière (ex. longs calculs scientifiques)**
- \* **validation en transactionnel (ex. cohérence et prévention d'interblocage)**

- **DEUX OPÉRATIONS A CONSIDÉRER**

| sauvegarde des information nécessaires à la reprise (points et messages)

| retour arrière à un état global cohérent et ré-exécution de l'application

- **TECHNIQUES DE POSE DES POINTS DE REPRISE**

**sauvegarde (pose) périodique indépendamment sur chaque site**

**effet domino**

**sauvegarde cohérente déclenchée périodiquement par un site initiateur  
(exemple : voir l'algorithme de Chandy, Lamport)**

**synchronisation explicite**

**coûteux en message**

**initiateur quelconque (tolérance aux pannes)**

**sauvegarde adaptative de Xu et Netzer (1993)**

**synchronisation implicite entre sites**

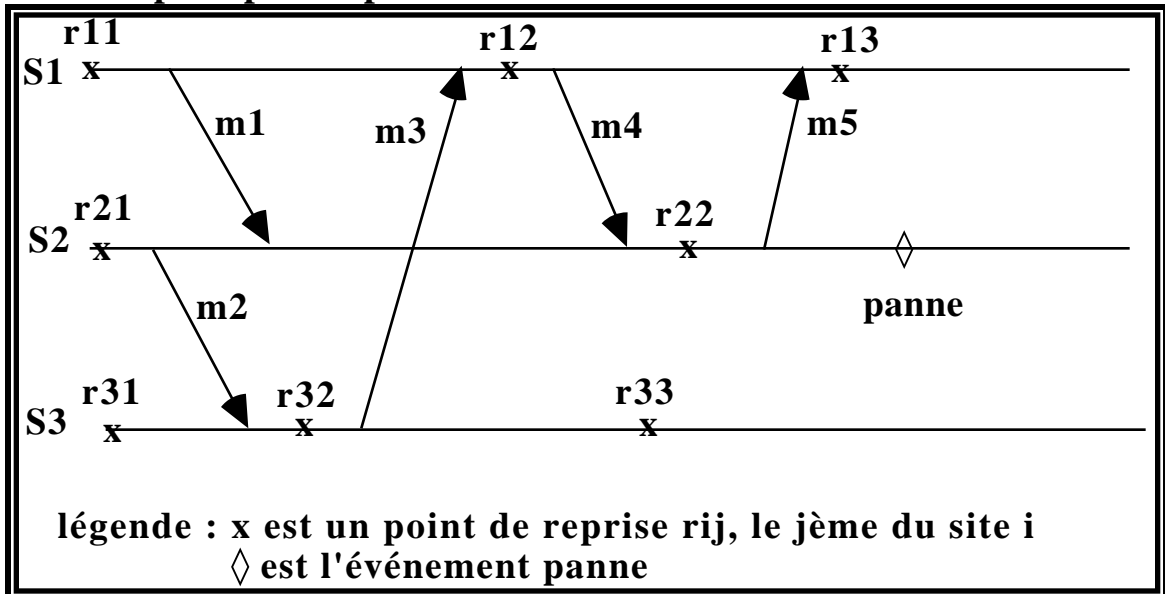
**points de contrôle forcés sur un site récepteur**

**Référence : [XU 93] J. XU, R. NETZER, Adaptive Independent Checkpointing for Reducing Rollback Propagation, 5th IEEE Symp. on Parallel and Distributed Processing, Dec. 93, Dallas TX, USA, 8 pages, 1993.**

### SAUVEGARDE INDÉPENDANTE SUR CHAQUE SITE

- effet domino : le retour arrière d'un site entraîne le retour des autres au delà du dernier point de reprise et cela parfois jusqu'au début du traitement

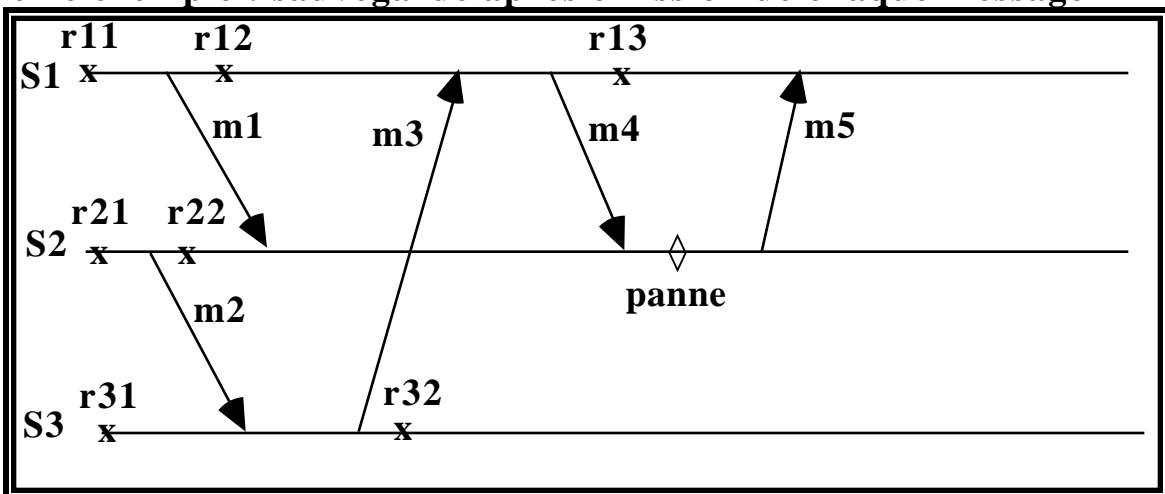
#### Premier exemple : points pris au hasard



Pour retrouver un point cohérent et repartir après réparation (ou avec un processeur en redondance passive sur S2), S2 remonte à r22, mais comme l'émission de m5 est perdue, il faut faire remonter S1 à r12 ; puis comme l'émission de m4 est perdue, il faut faire remonter S2 à r21 ; etc.

Aucune coupure r1a, r2b, r3c n'est cohérente sauf la coupure initiale r11, r21, r31.

#### Deuxième exemple : sauvegarde après émission de chaque message



On peut repartir avec r13, r22, r32 si on a noté l'émission de m1 et m4

## MODÉLISATION

- On note  $\rightarrow$  la dépendance causale
- On note  $r_{p,j}$  le jème point de reprise sur le site  $S_p$ ;  
Chaque  $S_p$  pose un point de reprise initial  $r_{p1}$  au début de l'exécution.  
Une ligne de reprise est un ensemble de points de reprise, un par site.  
Une ligne de reprise  $R$  est cohérente si elle forme un état global cohérent :  
tout message enregistré reçu sur un site a été enregistré émis sur un autre.  
enregistré $\Leftrightarrow$  fait partie du passé, de la coupure
- L'intervalle de reprise  $I_{p,i}$  est la suite d'événements entre  $r_{p,i}$  et  $r_{p,i+1}$   
(il contient  $r_{p,i}$  mais pas  $r_{p,i+1}$ )

### CHEMINS EN ZIGZAG

- Il y a un chemin en zigzag de  $r_{p,i}$  à  $r_{q,j}$  si et seulement si il existe des messages  $m_1, m_2, \dots, m_n$  ( $n \geq 1$ ) tels que
  - (1)  $m_1$  est envoyé par le site  $S_p$  après  $r_{p,i}$
  - (2) si  $m_k$  ( $1 \leq k < n$ ) est reçu par le site  $S_r$ , alors  $m_{k+1}$  est envoyé par  $S_r$  dans le même intervalle de reprise ou dans un intervalle postérieur (noter que  $m_{k+1}$  peut être envoyé avant ou après l'arrivée de  $m_k$ ),
  - (3)  $m_n$  est reçu par le site  $S_q$  avant  $r_{q,j}$

### CYCLE EN ZIGZAG

Le point de reprise  $r$  appartient à un cycle en zigzag si et seulement si il y a un chemin en zigzag de  $r$  à lui-même.

### CHEMIN CAUSAL ENTRE POINTS DE REPRISE

si  $r_{p,i} \rightarrow r_{q,j}$ , alors on a un chemin causal

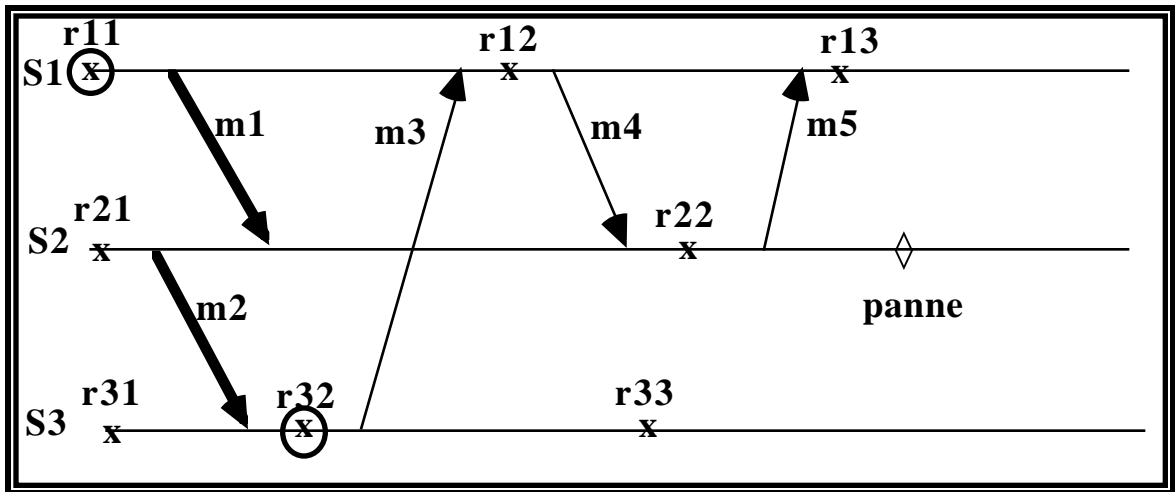
chemin causal  $\Rightarrow$  chemin en zigzag  
chemin en zigzag  $\not\Rightarrow$  chemin causal

### THÉORÈME

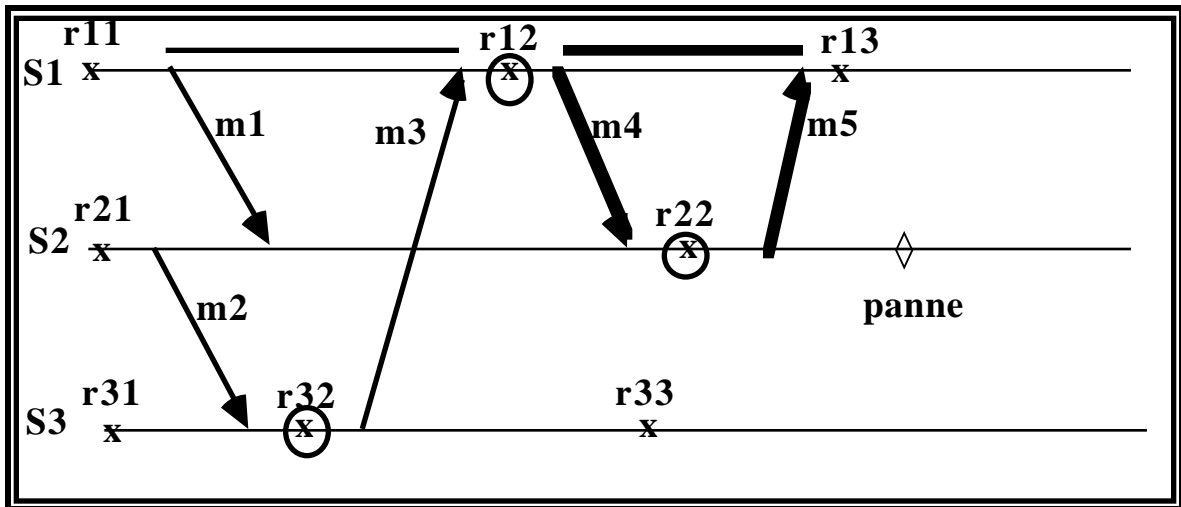
Un ensemble  $R$  de points de reprise sur différents sites peut être étendu faire partie d'une ligne de reprise cohérente si et seulement si aucun point de reprise de  $R$  n'appartient à un chemin en zigzag vers un autre point de reprise de  $R$  (ou vers lui-même, formant un cycle en zigzag).

**COROLLAIRE :** S'il n'y a pas de chemin en zigzag (ni cycle) dans  $R$  on peut toujours construire une ligne de reprise cohérente incluant  $R$

## EXEMPLES DE CHEMINS ET CYCLES EN ZIGZAG



chemin en zigzag de r11 à r32, chemin non causal



cycle en zigzag autour de r22 (messages m5 et m4)  
 cycle en zigzag autour de r32 (messages m3, m1 et m2)  
 cycle en zigzag autour de r12 (messages m4, m2 et m3)

## **PRINCIPE DE LA SAUVEGARDE ADAPTATIVE**

### **POINTS DE REPRISE UTILISABLE**

- **Un point de reprise  $r_{p,i}$  est utilisable s'il appartient à une ligne de reprise cohérente, c'est à dire s'il appartient à un état global cohérent.**

### **COROLLAIRE**

**Un point de reprise  $r_{p,i}$  est utilisable s'il n'y a pas de cycle en zigzag qui le contienne.**

### **PRINCIPE DE LA SAUVEGARDE ADAPTATIVE**

**Empêcher les cycles en zigzag de se former et pour cela :**

- **détecter si un message reçu sur un site achève de construire un cycle,**
- **si c'est le cas, forcer un point de reprise avant la réception du message.**

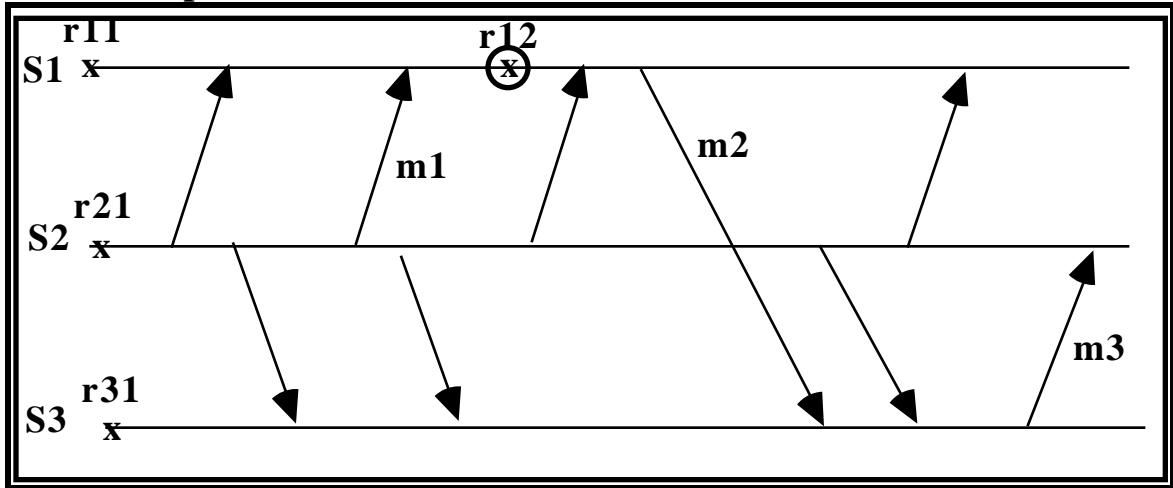
### **MÉTHODE APPROCHÉE**

**En fait on ne peut pas détecter si le cycle est en zigzag. Alors en approximation on détecte si le chemin est causal.**

**(à l'expérience, il y a très peu de chemins en zigzag non causaux)**

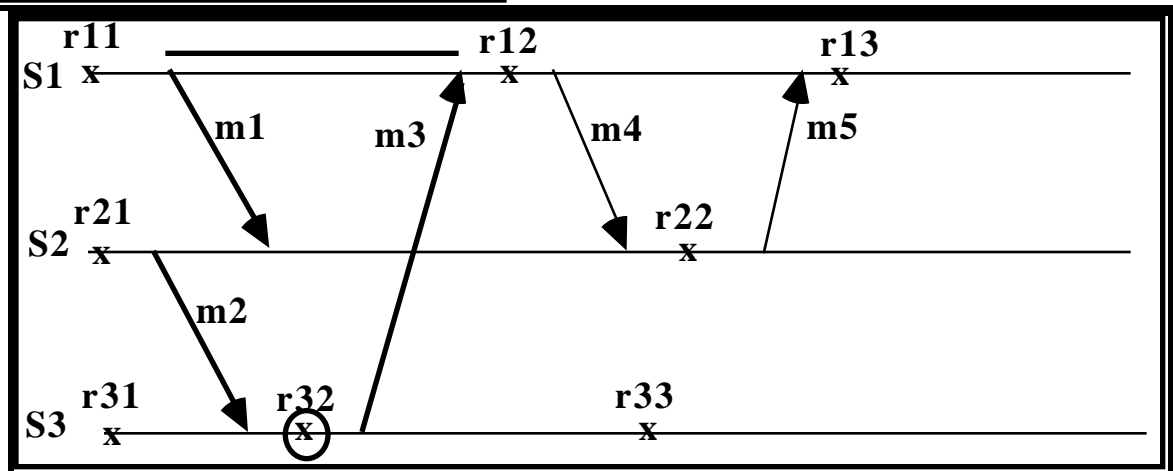
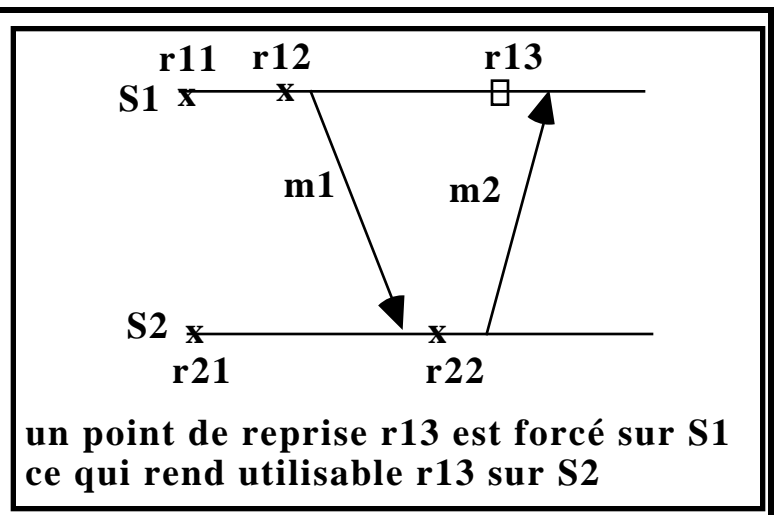
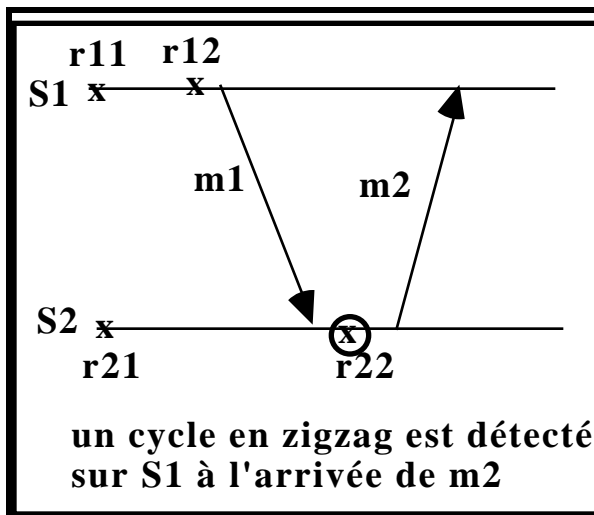
## DÉTECTION DES CYCLES EN ZIGZAG

- difficulté car parfois la connaissance du futur lointain est nécessaire



Le cycle en zigzag autour de r12 apparaît bien après r12 quand m3 arrive

### APPROXIMATION : DÉTECTER LES CHEMINS CAUSAUX



r32 : cycle en zigzag non détecté (il n'est pas causal)

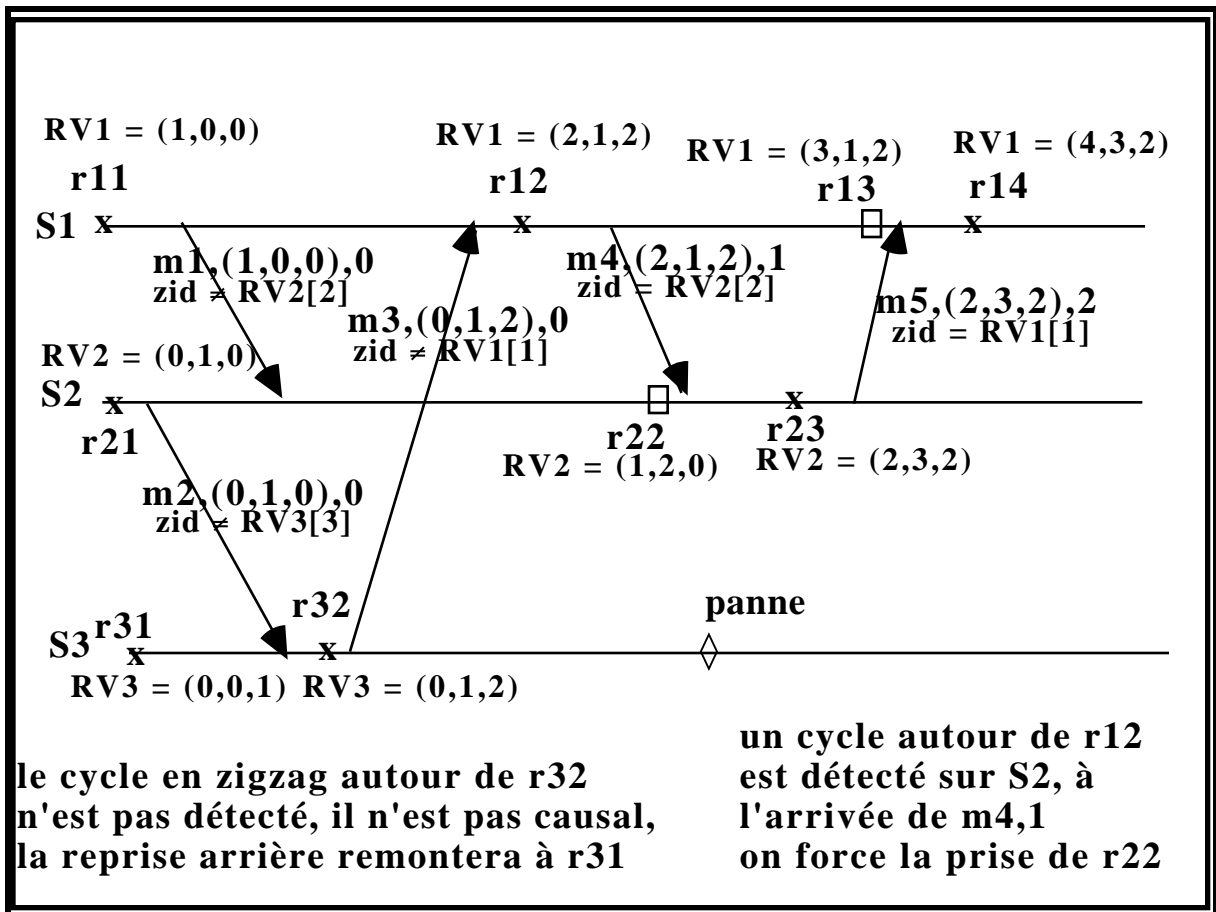
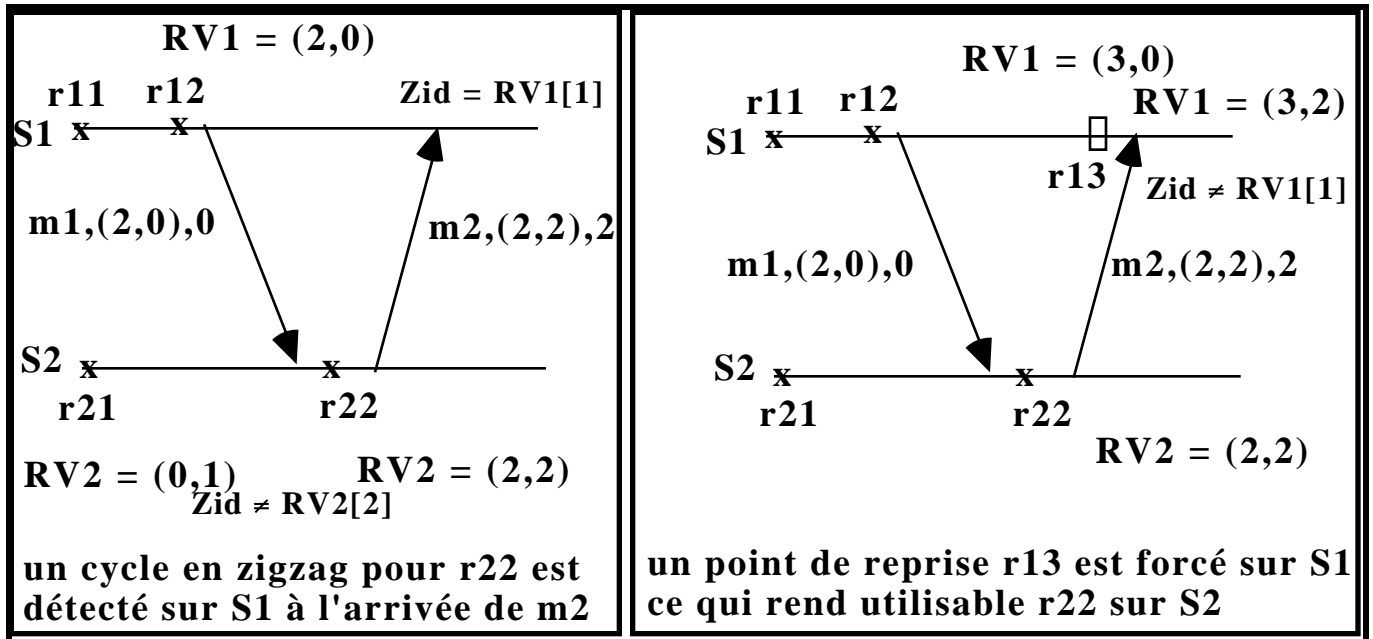
## ALGORITHME DE SAUVEGARDE ADAPTATIVE

- **Détecter les dépendances causales :**
  - On utilise les horloges vectorielles pour dater les points de reprise
  - On associe une horloge vectorielle  $V_i$  à chaque site  $S_i$
  - Initialement  $V_i = (0,0,\dots,0,1,0,\dots,0)$  nul partout sauf la  $i$ ème composante
  - A chaque point de reprise nouveau local à  $S_i$ , on fait  $V_i[i] := V_i[i] + 1$
  - Chaque message  $m$  porte une estampille  $V_m$  ( $V_m = V_i$  de l'émetteur)
  - A la réception de  $(m, V_m)$  par un site  $S_i$ , on enrichit l'historique connu par  $S_i$  avec l'historique transporté par  $m$  :
 
$$V_i[j] := \max(V_i[j], V_m[j]) \text{ pour tous } j = 1,\dots,n, j \neq i$$
  - $V_i[j]$  indique le passé de  $S_i$  situé sur  $S_j$  et tel qu'il est connu par  $S_i$   
 indique donc aussi le numéro du dernier point de reprise sur  $S_j$  qui est en dépendance causale avec le point de reprise courant sur  $S_i$ .
  - A la création d'un nouveau point de reprise  $r_{i,k}$  sur  $S_i$ , le site  $S_i$  sauvegarde dans  $RV_i$  les dépendances causales de  $r_{i,k}$ . Soit  $RV_i = V_i(r_{i,k})$ .
  - A l'envoi d'un message  $m$  de  $S_i$  vers  $S_j$ ,  $S_i$  surcharge le message  $m$  avec  $Zid = RV_i[j]$ , c'est à dire avec le numéro du dernier point de reprise sur  $S_j$  qui est sur un chemin causal avec  $r_{i,k}$
  - A la réception d'un message  $(m, V_m, Zid)$  par  $S_i$  et avant de traiter le message, l'algorithme regarde s'il existe un cycle causal entre le point de reprise courant sur  $S_i$ , daté  $RV_i[i]$  et le point de reprise sur  $S_j$  précédant l'émission. Il y a un cycle causal point de reprise sur  $S_j$  précédant l'émission si  $RV_i[i] = Zid$ .
- Puis  $V_i$  est mis à jour.

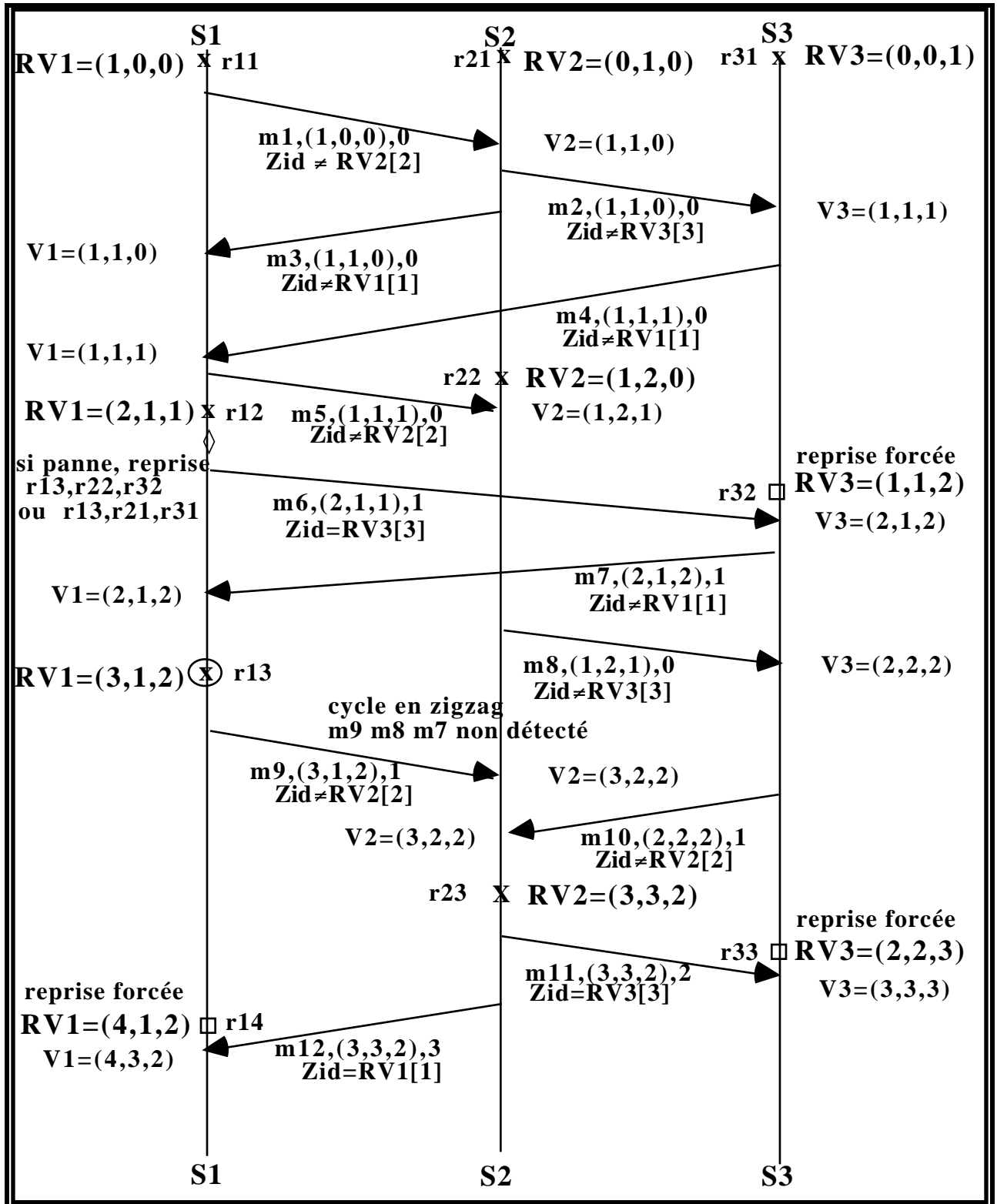
### ALGORITHME :

- 1 : si  $Zid = RV_i[i]$  alors                    forcer un nouveau point de reprise;  
    $V_i[i] := V_i[i] + 1$  ;  
    $RV_i = V_i$   
       finsi;
- 2 :  $V_i[j] := \max(V_i[j], V_m[j])$  pour tous  $j = 1,\dots,n, j \neq i$  ; -- historique

## EXEMPLES DE SAUVEGARDE ADAPTATIVE



# EXEMPLE DE SAUVEGARDE ADAPTATIVE



trois reprises sont forcées

sept reprises spontanées

# CONCLUSION SUR LA SAUVEGARDE ADAPTATIVE

## RÉSULTATS EXPÉRIMENTAUX :

- **Conditions d'expérimentation**

**6 programmes de tests (de 13 000 à 370 000 messages échangés au total)**

**Hypercube Intel iPSC/860 avec 16 noeuds utilisés pour les tests**

- **Réduction de la propagation arrière pour atteindre une ligne cohérente**

**Avec une sauvegarde périodique déclenchée indépendamment sur chaque site, il faut en moyenne faire un retour arrière de 3 à 4 points de reprise par site pour obtenir une ligne de reprise cohérente.**

**En ajoutant la sauvegarde adaptative, on réduit ce retour arrière à un peu moins d'un point de reprise par site.**

- **Surcoût pour la sauvegarde adaptative**

- **faible en gestion de l'horloge vectorielle et de tests sur les messages**

- **nombre de points de reprise additionnels : pose 4% de points de reprise de plus que la sauvegarde périodique indépendante**

## BILAN

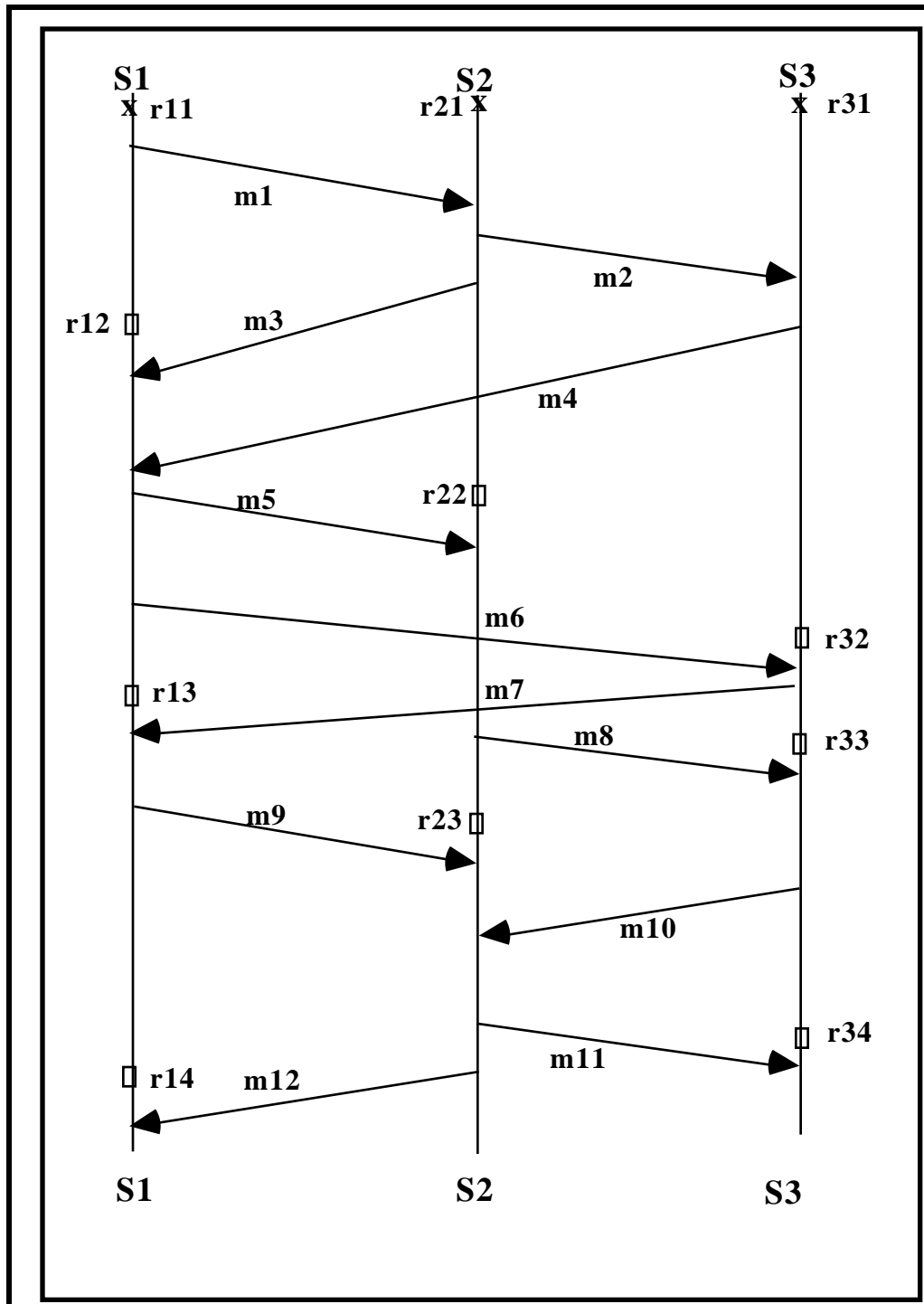
- **Condition nécessaire et suffisante pour qu'un ensemble de points de reprise soit cohérent : absence de chemins en zigzag entre ces points**

- **Méthode heuristique : on recherche les chemins causaux (leur absence est une condition nécessaire) au lieu de rechercher les chemins en zigzag**

- **Conclusion : gain obtenu par la réduction de l'effet domino**

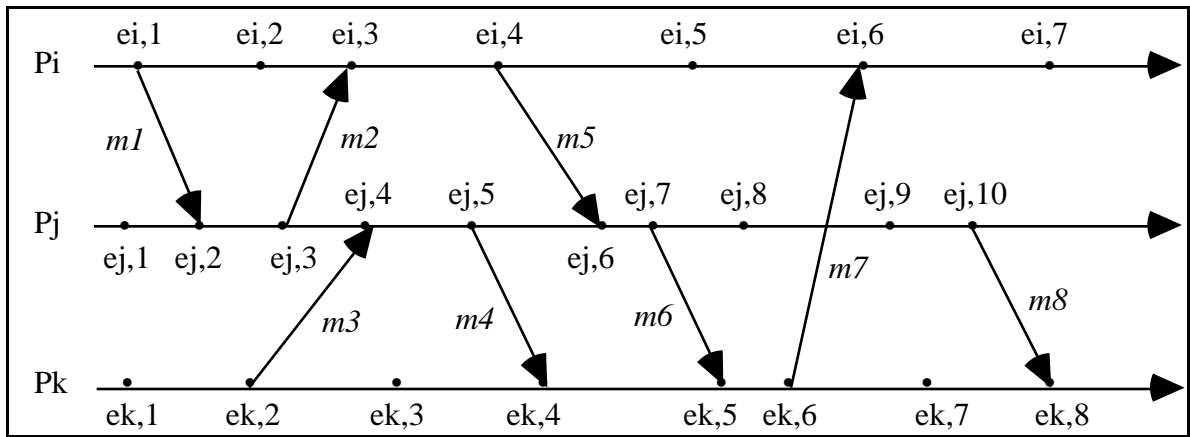
- **ne pas oublier qu'il faut en plus faire le retour arrière, avoir sauvegardé les messages qui franchissent la coupure et réexécuter l'application.**

**RETOUR SUR L'EXEMPLE AVEC UNE AUTRE MÉTHODE  
 ((RECEPTION)\*(ÉMISSION)\*SAUVEGARDE)  
 (heuristique de Russell 1980)**

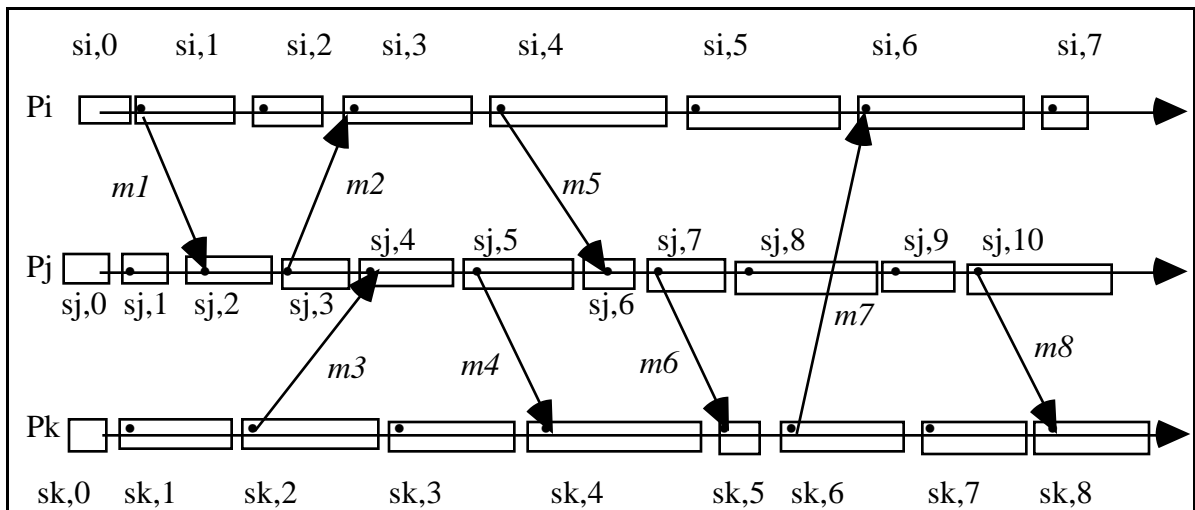


**toutes les onze sauvegardes sont forcées**

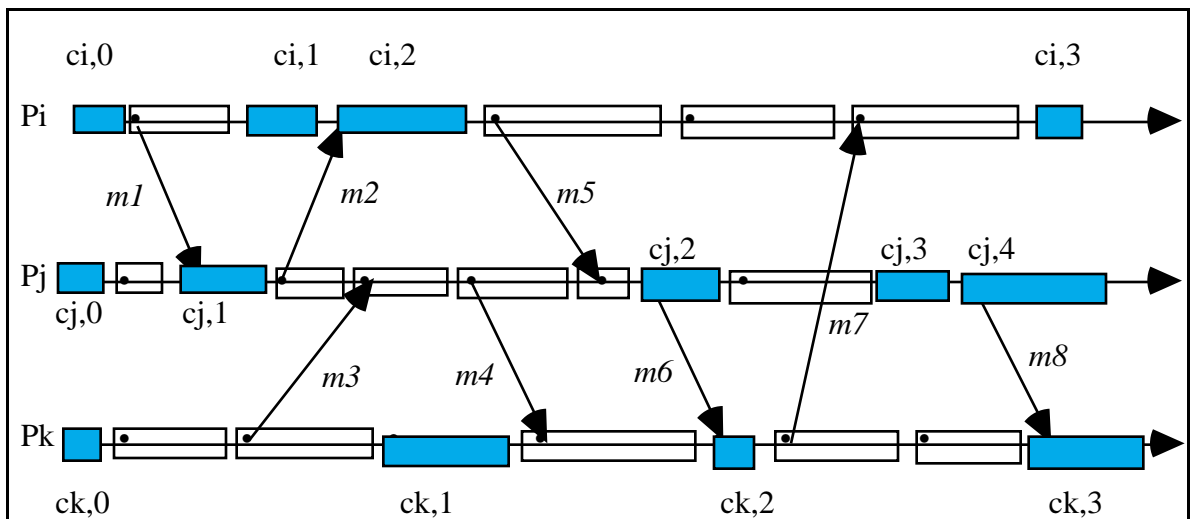
## AUTRES MODÉLISATIONS D'UNE EXÉCUTION RÉPARTIE



**1. Exemple d'exécution répartie modélisée par des événements**  
**événement = exécution d'instruction**



**2. Même exécution répartie modélisée par des états locaux**



**3. Même exécution répartie modélisée par des points de contrôle**

# ÉTUDE DE CAS

## Objet réparti et répliqué sur 4 sites S1, S2, S3, S4

(avec même valeur initiale 100 Keuros)

méthodes : ajouter, retrancher, multiplier, lire

### Application coopérative asynchrone

**S1** : traitement t1 avec la copie locale; envoi M1; délai variable ; traitement t2 avec la copie locale; envoi M2;

**S2** : traitement t4 avec la copie locale; envoi M4;

**S3** : traitement t3 avec la copie locale; envoi M3;

t1 : ajouter 20

t2 : retrancher 10

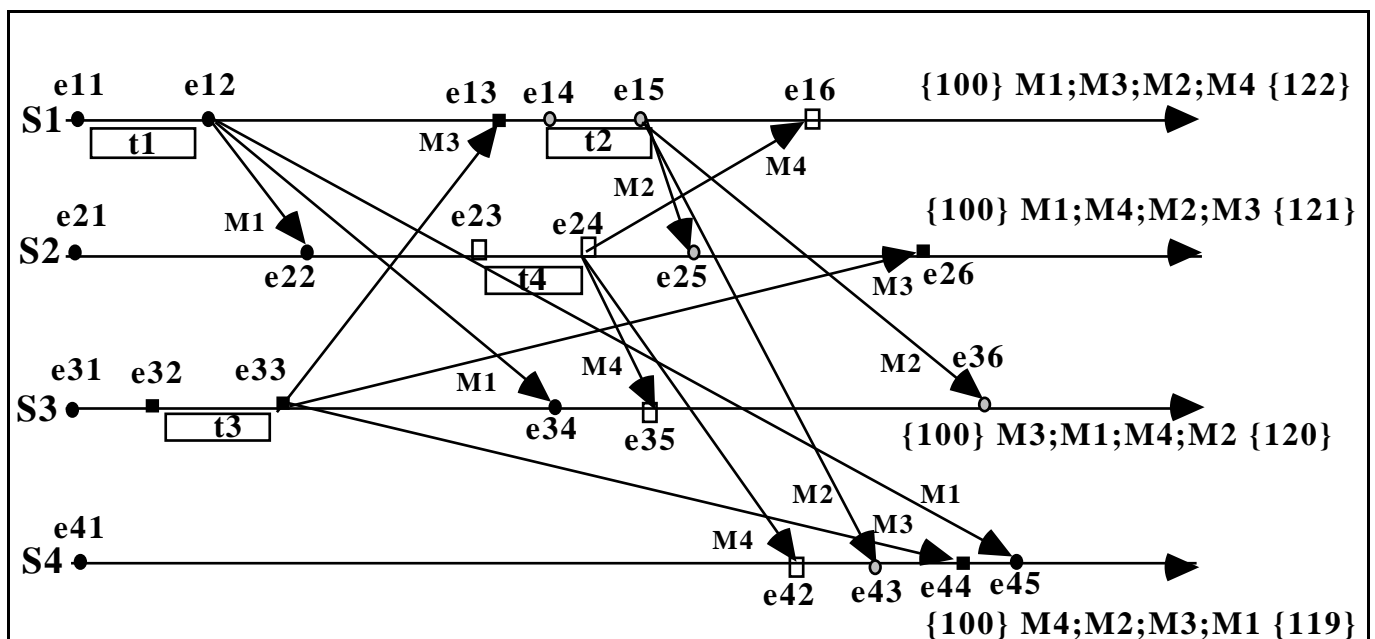
t3 : multiplier la valeur par 110%

t4 : lire et sauvegarder (ou imprimer ou visualiser) la valeur locale

M1, M2, M3, M4 : messages de mise à jour diffusés à toutes les copies.

Mi : exécuter ti sur votre copie locale

### DÉSORDRE NATUREL DES COMMUNICATIONS



Les sites visualisent 122, 120, 130, 100. Belle coopération!

Les répliques de l'objet valent 122, 121, 120, 119. Belle cohérence!

**ORDRE CAUSAL**

$\forall S_k, M_i$  émis par  $S_i$  sur  $C_{ik}$ ,  $M_j$  émis par  $S_j$  sur  $C_{jk}$ ,  
 $\text{émission}_i(M_i) \rightarrow \text{émission}_j(M_j) \Rightarrow \text{délivrance}_k(M_i) \rightarrow \text{délivrance}_k(M_j)$

**ORDRE LOCAL**

(c'est aussi un ordre causal)

$\forall S_k, M_i$  émis par  $S_i$  sur  $C_{ik}$ ,  $M_j$  émis par  $S_i$  sur  $C_{ik}$ ,  
 $\text{émission}_i(M_i) \rightarrow \text{émission}_i(M_j) \Rightarrow \text{délivrance}_k(M_i) \rightarrow \text{délivrance}_k(M_j)$

**ORDRE TOTAL**

$M_i$  diffusé sur  $S_i$  et émis sur  $C_{ik}$ ,  
 $M_j$  diffusé sur  $S_j$  et émis sur  $C_{jk}$ ,

$\Rightarrow$

$\forall S_k, \text{délivrance}_k(M_i) \rightarrow \text{délivrance}_k(M_j)$

ou exclusif

$\forall S_k, \text{délivrance}_k(M_j) \rightarrow \text{délivrance}_k(M_i)$

**ORDRE TOTAL CAUSAL**

$\alpha)$  la précedence causale est respectée, si elle existe

$\forall S_k, M_i$  émis par  $S_i$  sur  $C_{ik}$ ,  $M_j$  émis par  $S_j$  sur  $C_{jk}$ ,

$\text{émission}_i(M_i) \rightarrow \text{émission}_j(M_j) \Rightarrow \text{délivrance}_k(M_i) \rightarrow \text{délivrance}_k(M_j)$

$\beta)$  si elle n'existe pas, on a un ordre total simple

$M_i$  diffusé sur  $S_i$  et émis sur  $C_{ik}$ ,

$M_j$  diffusé sur  $S_j$  et émis sur  $C_{jk}$ ,

$\text{émission}_i(M_i) \parallel \text{émission}_j(M_j)$

$\Rightarrow$

$\forall S_k, \text{délivrance}_k(M_i) \rightarrow \text{délivrance}_k(M_j)$

ou exclusif

$\forall S_k, \text{délivrance}_k(M_j) \rightarrow \text{délivrance}_k(M_i)$

## RELATIONS ENTRE ÉVÉNEMENTS DE DIFFUSION

On a 6 relations entre les événements de diffusion des 4 messages

ordre local d'émission :

émission<sub>1</sub>(M1) → émission<sub>1</sub>(M2)

ordre causal d'émission

émission<sub>1</sub>(M1) → émission<sub>2</sub>(M4)

émission<sub>3</sub>(M3) → émission<sub>1</sub>(M2)

émission<sub>1</sub>(M1) || émission<sub>3</sub>(M3)

émission<sub>2</sub>(M4) || émission<sub>3</sub>(M3)

émission<sub>1</sub>(M2) || émission<sub>2</sub>(M4)

QU'EN EST-IL AVEC CE DÉSORDRE NATUREL DES RÉCEPTIONS?

{100}M1;M3;M2;M4{122}; S1 respecte l'ordre causal et l'ordre local

{100}M1;M4;M2;M3{121}; S2 respecte l'ordre local, et pas l'ordre causal

{100}M3;M1;M4;M2{120}; S3 respecte l'ordre causal et l'ordre local

{100}M4;M2;M3;M1{119}; S4 ne respecte ni l'ordre causal ni l'ordre local

Il n'y a pas d'ordre total

D'autres variantes de réception ne sont pas meilleures :

{100}M3;M2;M1{120} respecte l'ordre causal et pas l'ordre local

{100}M2;M1;M3{121} ne respecte ni l'ordre causal ni l'ordre local

### QUESTIONS :

Comment assurer le respect de l'ordre causal et de l'ordre local?

Comment obtenir un ordre total d'émission des messages?

Comment obtenir un ordre total d'émission et de délivrance des messages?

(synchronie de vue pour tous les sites)

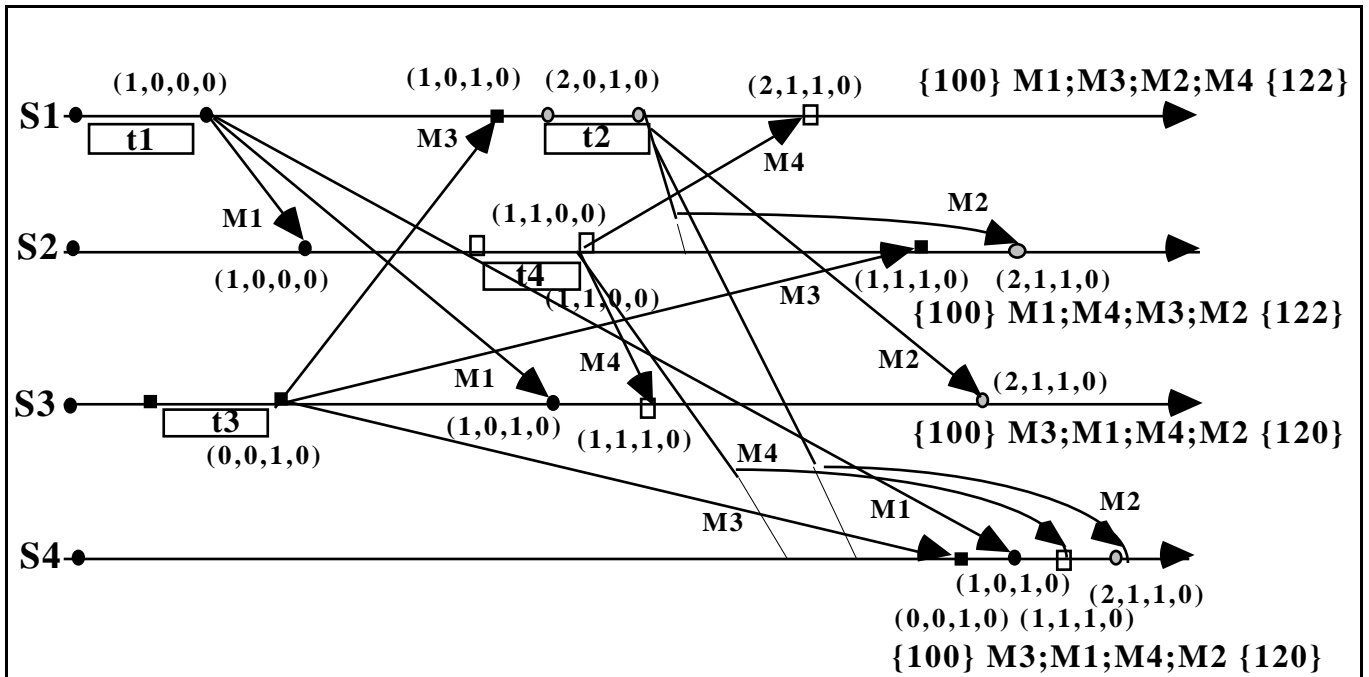
Comment obtenir une visualisation cohérente sans ordre total?

Comment prendre un ensemble cohérent de points de reprise

Comment installer la tolérance à une panne franche d'un processeur

## RESPECT DE L'ORDRE CAUSAL

**Comment assurer le respect de l'ordre causal et de l'ordre local?  
utilisation des horloges vectorielles**



Les sites visualisent 120, 122 ou 130.

Les répliques de l'objet valent 122 ou 120.

**Problème : Comment obtenir un ordre total d'émission des messages?**

**Réponse : un site séquenceur, ou un jeton circulant ou exclusion mutuelle**

**Comment obtenir un ordre total d'émission et de délivrance des messages?**

**numéroter les diffusions et diffuser le numéro dans le message**

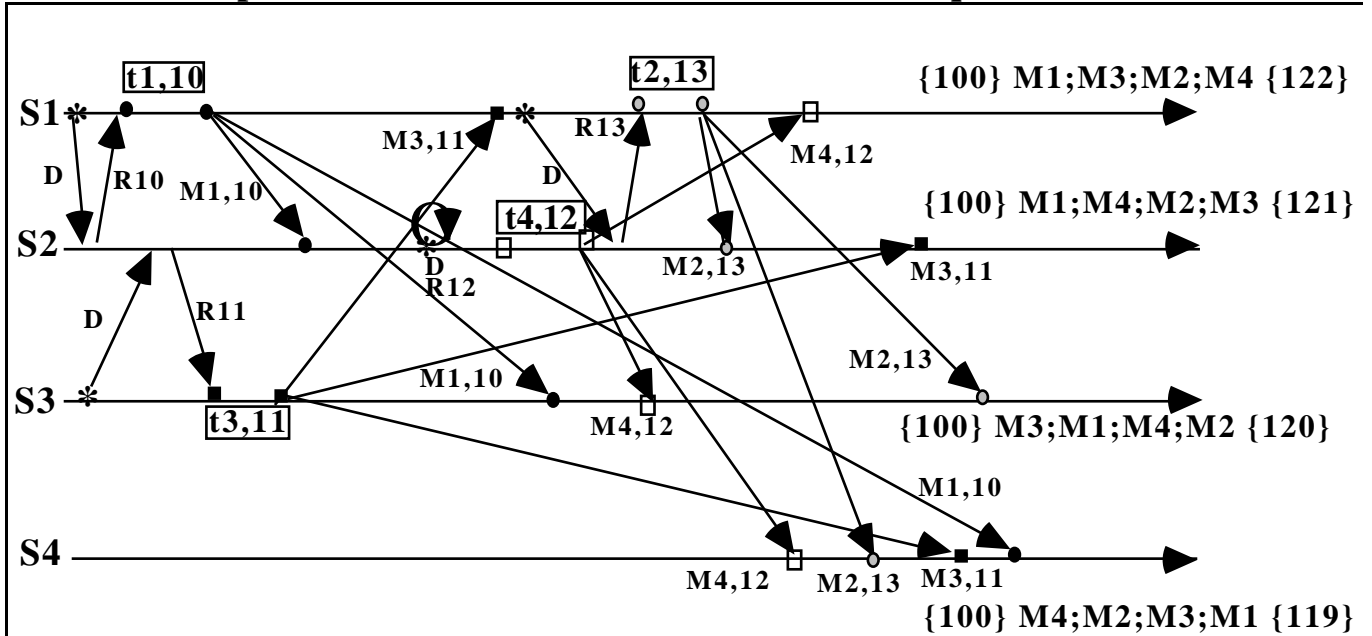
**autre solution avec objet primaire et objets secondaires**

# ORDRE TOTAL DE DIFFUSION DES MESSAGES AVEC UN SITE SÉQUENCEUR

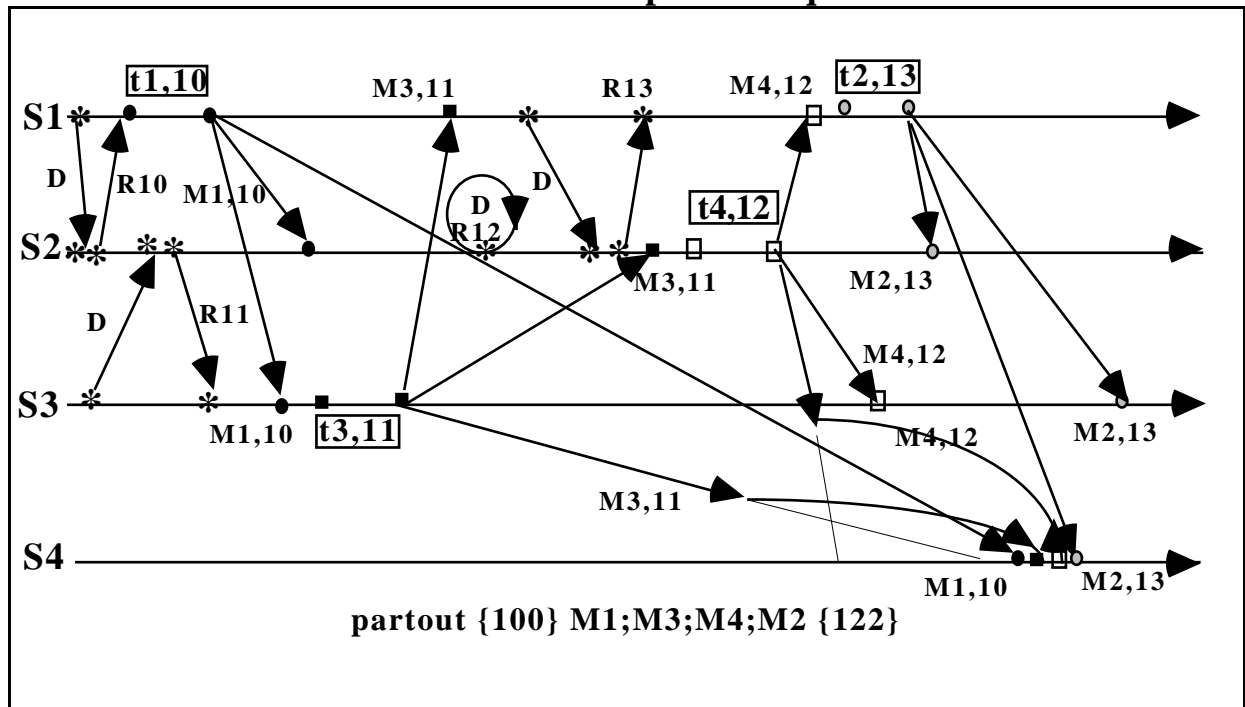
un site séquenceur fournit le numéro de diffusion

l'ordre total dépend de l'ordre de réception de la demande de numéro

Première étape : numéroter les diffusions avec un séquenceur sur S2



Cela ne suffit pas. Il faut réordonner les réceptions et les traitements selon l'ordre total fourni par le séquenceur

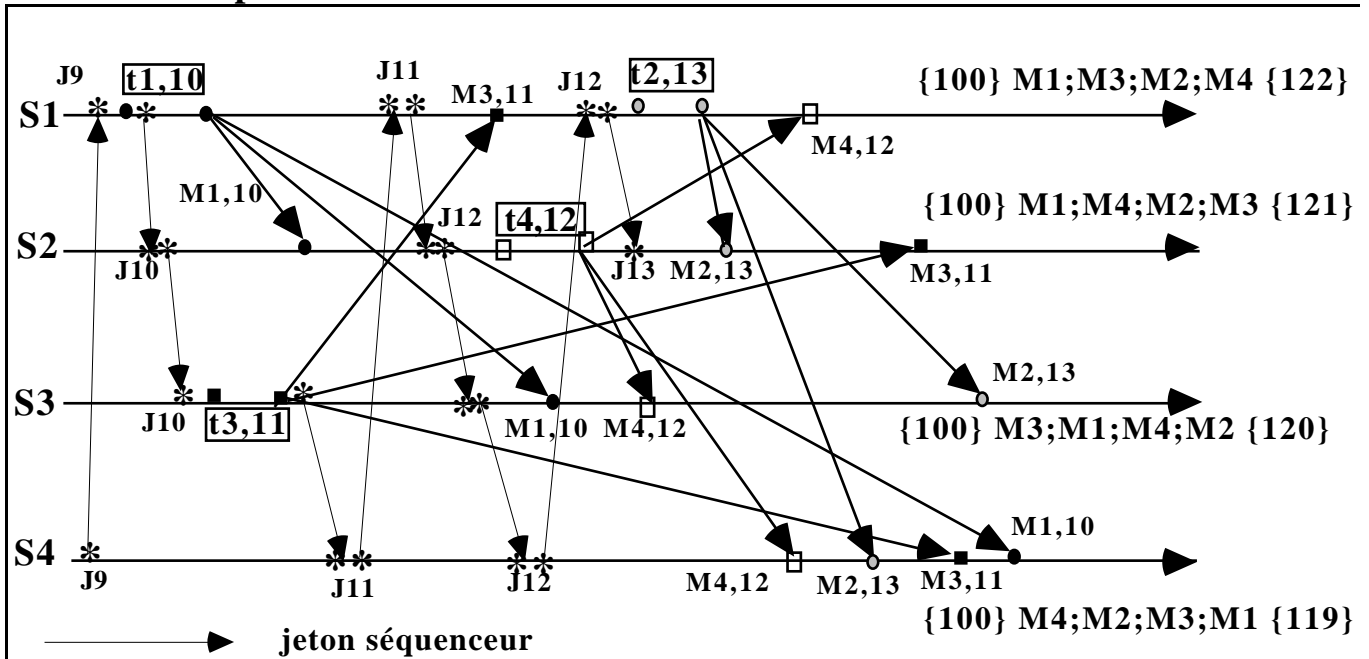


variante : le séquenceur reçoit le message à diffuser et fait la diffusion (voir plus loin la solution avec objet primaire et objets secondaires)

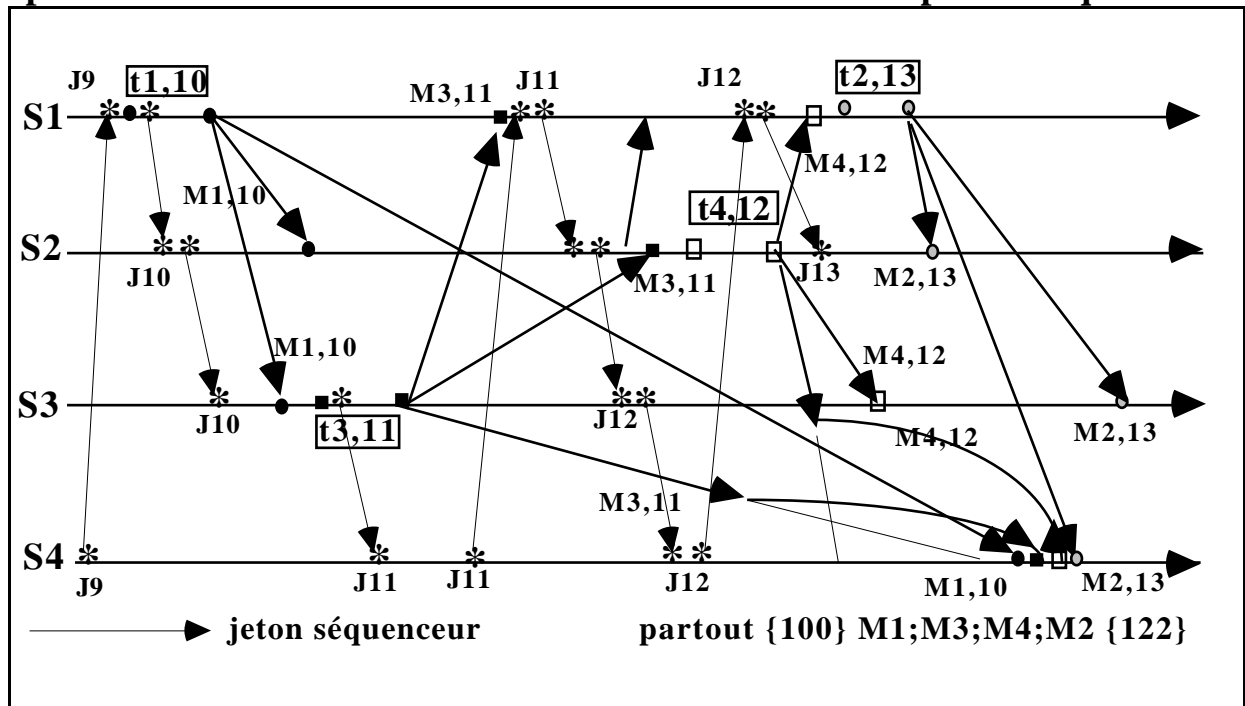
# ORDRE TOTAL DE DIFFUSION DES MESSAGES AVEC UN ANNEAU VIRTUEL ET UN JETON

le jeton contient un séquenceur qui fournit le numéro de diffusion  
l'ordre total dépend de l'ordre de réception du jeton

Première étape : numéroter les diffusions



Cela ne suffit pas (même si les canaux sont FIFO). Il faut réordonner les réceptions et les traitements selon l'ordre total fourni par le séquenceur

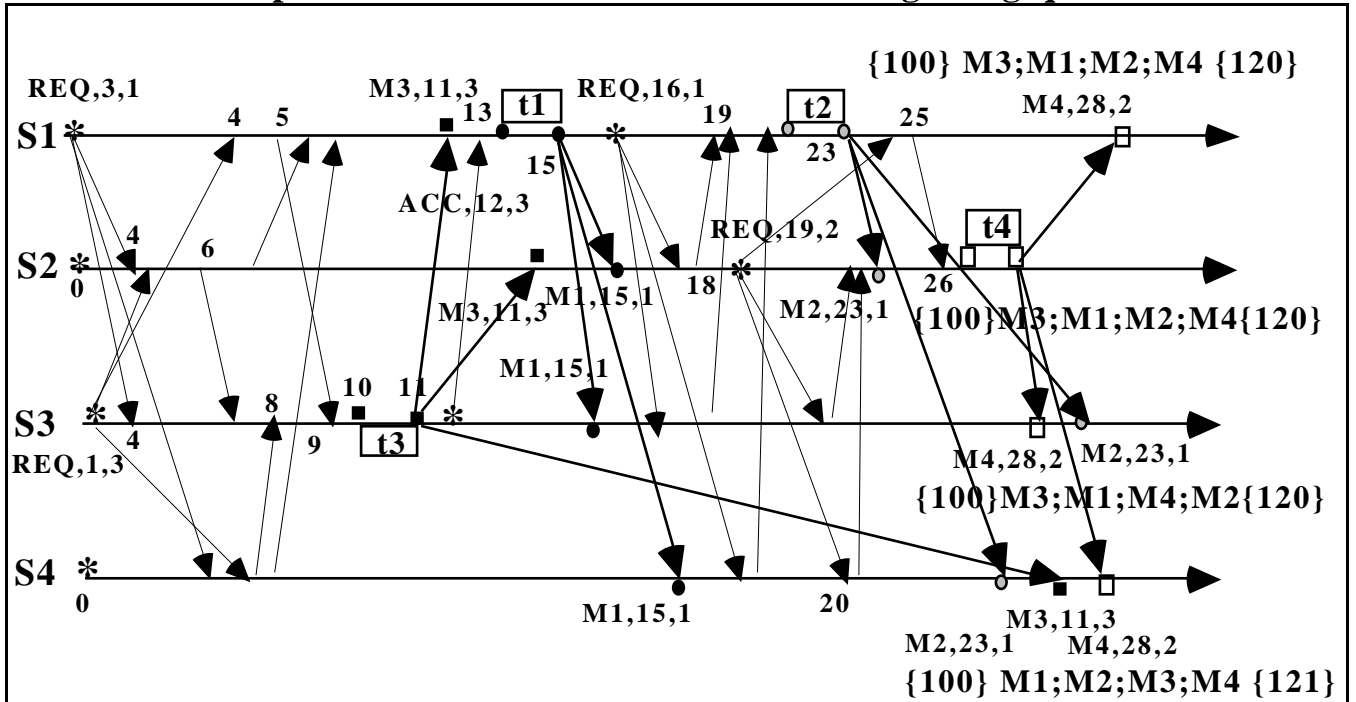


L'ordre total est causal. Peut-on avoir un ordre total non causal si les sites prennent des numéros en avance au passage du jeton ?

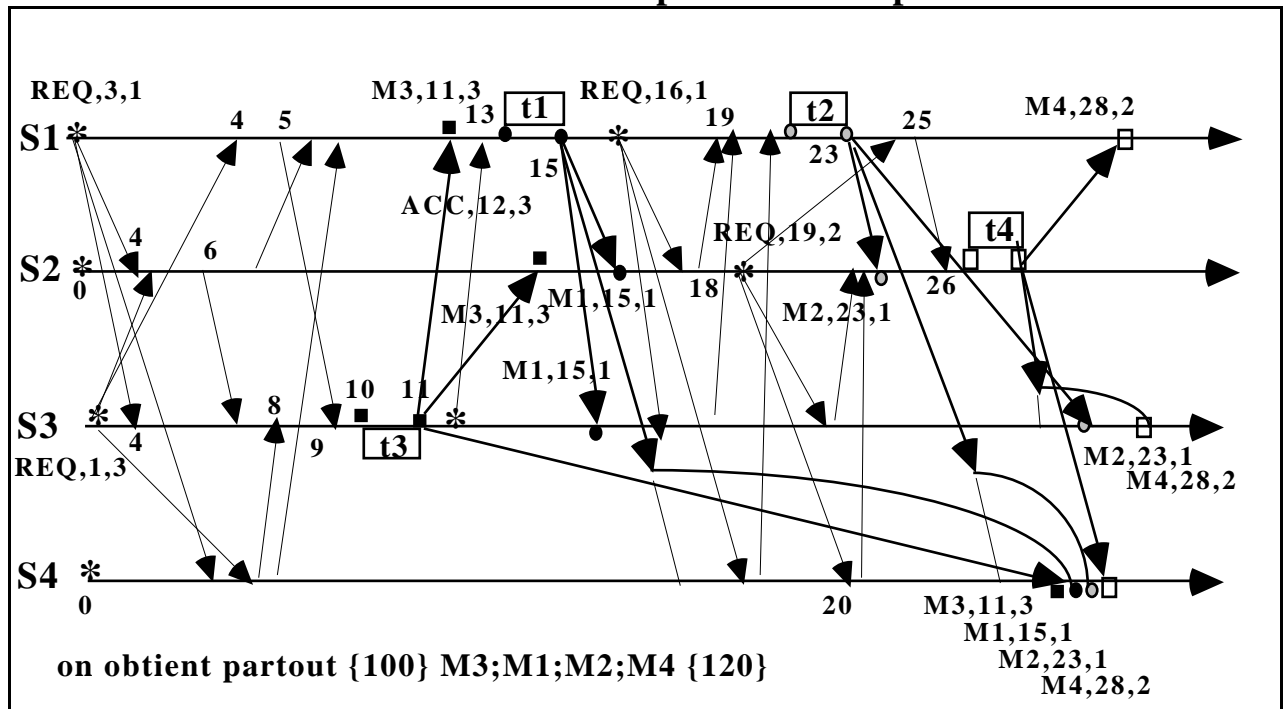
# ORDRE TOTAL DE DIFFUSION DES MESSAGES EXCLUSION MUTUELLE RÉPARTIE

on utilise Ricart-Agrawala

**l'ordre total dépend des valeurs initiales des horloges logiques**



**Cela ne suffit pas. Il faut réordonner les réceptions et les délivrer selon l'ordre total fourni par les estampilles**

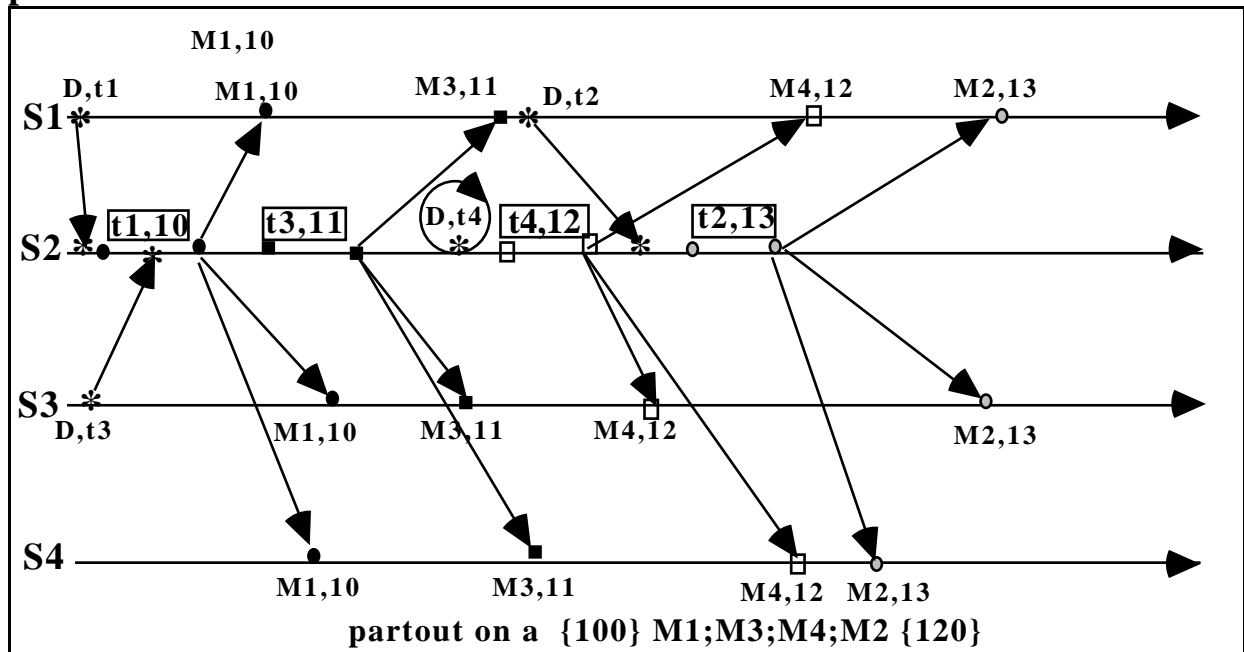


## ORDRE TOTAL DE DIFFUSION DES MESSAGES UN OBJET PRIMAIRE ET DES OBJETS SECONDAIRES

un site serveur primaire (est aussi séquenceur)

l'ordre total dépend de l'ordre de réception de la demande de service

Le primaire diffuse les traitements à faire



Si les canaux sont FIFO, on a l'ordre total

sinon, il faut numéroter les messages, comme ici

Permet des lectures concurrentes en cohérence faible des objets dupliqués

Problème en cas de panne du serveur primaire

Plus généralement :

Comment tolérer une panne franche de processeur

Comment prendre un ensemble cohérent de points de reprise

autres études de cas

1. étude de cas avec deux objets répartis A et B et des traitements coopératifs répartis. Problèmes de cohérence transactionnelle.

P1 : A - 20 si A reste positif;

P2 : B + 20;

P3 : A + 10;

P4 : consulter A et B

2. traitements transactionnels répartis avec 3 objets répartis

P1 : A - 20; B+20;

P2 : B-10; A+10

P3 : C := A+B

visualiser A, B, C