COORDINATION PAR RENDEZ-VOUS MULTIPLE

SAIDOUNI Djamel Eddine

Université Constantine 2 - Abdelhamid Mehri
Faculté des Nouvelles Technologies de l'Information et de la Communication
Département d'Informatique Fondamentale et ses Applications

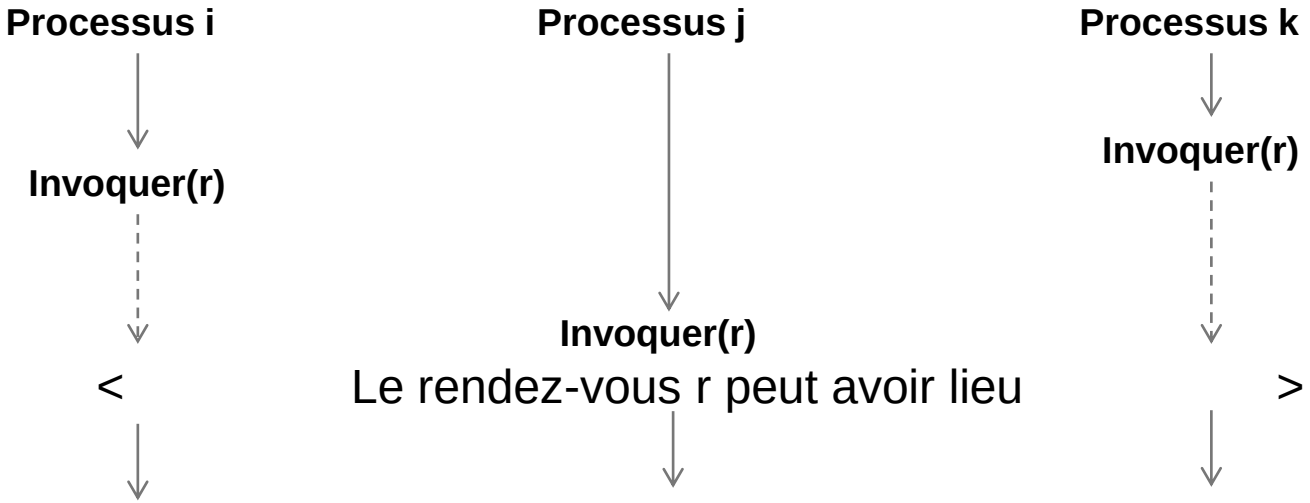
Laboratoire de Modélisation et d'Implémentation des Systèmes Complexes

Djamel.saidouni@univ-constantine2.dz
saidounid@hotmail.com
Tel: 0559082425

CONCEPT DU RENDEZ-VOUS

Définition: Un rendez-vous est caractérisé par la présence simultanée d'un certain nombre de processus à un point de contrôle commun qui a été défini au préalable.

Exemple: Considérons un rendez-vous r qui met en jeux les 3 processus i, j, k. Un ou plusieurs de ces processus peuvent invoquer le rendez-vous r; ils sont alors bloqués jusqu'à ce que r ait été invoqué par les 3 processus mis en jeu: Ceux-ci seront alors simultanément à leurs points de contrôle respectifs.



Les langages CSP et LOTOS offrent la programmation par rendez-vous

CONCEPT DU RENDEZ-VOUS
NON DÉTERMINISME

Afin d'avoir une plus grande puissance d'expression, il est intéressant d'offrir aux processus une structure de contrôle non-déterministe.

- ✓ Un processus doit pouvoir solliciter plusieurs rendez-vous et ne s'engager que dans l'un (choisi de façon non-déterministe) de ceux qui sont ou seront possibles.

Exemple: Soient 4 rendez-vous, chacun met en jeu, lorsqu'il est réalisé, les processus indiqués:

rendez-vous	r1={i,j,k}
rendez-vous	r2={i,j,m,p}
rendez-vous	r3={i,m,r,q}
rendez-vous	r4={j,k,l}

- ✓ Si le site i n'invoque que r1, il reste bloqué jusqu'à ce que les sites j et k invoquent également r1 et ce rendez-vous pourra alors avoir lieu.
- ✓ Par contre l'invocation par le site i de plusieurs rendez-vous, dans une commande non-déterministe, le bloque jusqu'à ce que l'un d'entre eux soit possible:

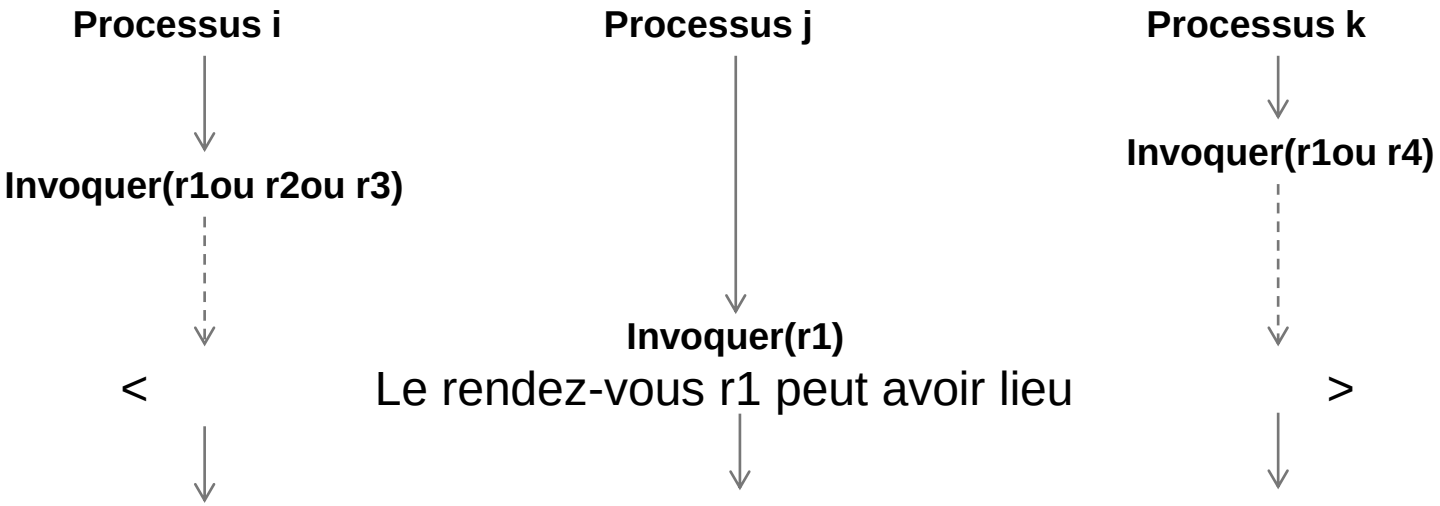
Exemple: invoquer (r1 ou r2 ou r3)

CONCEPT DU RENDEZ-VOUS
NON DÉTERMINISME (SUITE)

Le site i s'engagera dans le premier des 3 rendez-vous qui sera possible; si plusieurs le sont, il s'engagera dans un (quelconque) et un seul d'entre eux. Ces rendez-vous sont dits ***conflictuels.***

L'association d'un calcul à chaque rendez-vous permet de connaitre à postériori le rendez-vous réalisé:

invoquer r1 faire <calcul 1> fait
ou r2 faire <calcul 2> fait
ou r3 faire <calcul 3> fait
fin



PROBLÈME À RÉSOUDRE

- On s'intéresse au contrôle nécessaire à la mise en œuvre des rendez-vous.
- Un protocole réalisant ce contrôle doit assurer la coordination de sites intervenant dans le rendez-vous et prendre en compte le non-déterminisme potentiel de leurs invocations.
- Les propriétés attendues d'un tel protocole peuvent être décrites en utilisant la décomposition classique entre sûreté et vivacité.

Propriétés de sûreté:

- **(S1) Coordination synchrone**: Si à un instant donné un site i, qui a invoqué un rendez-vous r s'y engage, alors tous les autres sites qui interviennent dans le rendez-vous r, l'ont aussi invoqué et s'y engagent également.
- **(S2) Exclusion**: A un instant donné un site i ne peut être engagé que dans un seul rendez-vous.

Propriété de vivacité:

- Si des rendez-vous qui ont été invoqués sont possibles alors au moins l'un d'entre eux aura lieu.

PROBLÈME À RÉSOUDRE (SUITE)

- La propriété S1est appelée ***atomicité***; elle spécifie en effet un comportement du type tout ou rien.
- La propriété de sûreté met en évidence la difficulté que doivent résoudre les protocoles de mise en œuvre des rendez-vous: Garantir l'atomicité de chacun et une exclusion mutuelle entre ceux qui sont conflictuels.
- En ce qui concerne la vivacité, lorsque plusieurs rendez-vous sont possibles, dans le pire des cas un seul peut avoir lieu; ceci est la conséquence de l'invocation non-déterministe.

Intérêt: L'intérêt des rendez-vous c'est qu'ils constituent une primitive de synchronisation extrêmement puissante. Celle-ci peut être utilisée au niveau des groupes de processus, dans les systèmes d'exploitation répartis, pour synchroniser tout ou partie du groupe.
Dans langages de haut niveau, tel que LOTOS, offrent des primitives de synchronisation du type rendez-vous entre plusieurs processus. L'implémentation réelle de tels programmes nécessite l'utilisation des protocoles de mise en œuvre des rendez-vous distribués.

LA SPÉCIFICATION DE BAGRODIA

- A tout processus est associé un et un seul site.
- Les rendez-vous sont définis statiquement par l'ensemble des processus qu'ils mettent en jeu.
- On appelle RDV_i l'ensemble des rendez-vous où le site i intervient.

$$RDV_i = \{r \; tel \; que \; i \in r\}$$

- On dit que deux rendez-vous r_a et r_b sont conflictuels ssi $r_a \cap r_b \neq \emptyset$ (ils font intervenir au moins un site commun); tous les rendez-vous de RDV_i sont conflictuels.
- On associe à tout site i une variable $état_i$ qui prend les valeurs habituelles:
  - $dehors$: Lorsque le site i n'est pas intéressé par un rendez-vous.
  - $demandeur$: Lorsque le site i invoque un rendez-vou (de façon déterministe ou non).
  - $dedans$: Lorsque le site $i$se trouve engagé dans un rendez-vous r qu'il a invoqué.

Le but d'un protocole de mise en œuvre des rendez-vous est donc de gérer les transitions des variables $état_i$ de $demandeur$ à $dedans$; les autres transitions sont à l'initiative du site lui-même.

LA SPÉCIFICATION

- A tout site i est associé deux compteurs:
  - Var w_i et n_i : entiers monotone croissants initialisés à 0.
- Signification:
  - w_i = nombre de fois où le site i est passé dans l'état $demandeur$ (c'est-à-dire le nombre d'invocation de rendez-vous).
  - n_i = nombre de fois où le site i est passé dans l'état $dedans$ (c'est-à-dire le nombre d'engagement dans des rendez-vous).
- Initialement les sites sont dans l'état $dehors$.

La propriété d'exclusion (S2) qui stipule qu'un site ne s'engage que dans un rendez-vous à la fois s'écrit:

$$\forall \boldsymbol{i}: \boldsymbol{n_i \leq w_i \leq n_i + 1} \qquad \textbf{(S2')}$$

La propriété de coordination synchrone (S1), ou d'atomicité, qui stipule que tous les sites intervenant dans un rendez-vous doivent l'avoir invoqué pour qu'il puisse avoir lieu, s'écrit:

$$\forall \boldsymbol{r}: ((\forall \boldsymbol{j} \in \boldsymbol{r} : \boldsymbol{w_j = n_j + 1}) \Leftrightarrow \boldsymbol{r\ peut\ avoir\ lieu}) \qquad \textbf{(S1')}$$

LA SPÉCIFICATION (SUITE)

Remarquons que les compteurs w_i sont incrémentés par les processus eux-mêmes, trouver un protocole de mise en œuvre des rendez-vous revient à gérer les compteurs n_i de façon d'une part à ce que les relations (S1') et (S2') soient des invariants et d'autre part à ce que la vivacité soit assurée.

Mise en œuvre centralisée:

- Un site coordinateur gère les couples de compteurs (w_i, n_i)
  - Lorsqu'un site i passe dans l'état $demandeur$ il envoie un message de contrôle $prêt(i)$ au coordinateur et se bloque.
  - A la réception d'un message $prêt(i)$ par le coordinateur, ce dernier incrémente w_i et teste s'il existe un rendez-vous $r \in RDV_i$ pour lequel la condition apparaissant dans (S1') est vérifiée; si tel est le cas il envoie aux sites $\mathrm{k} \in r$un message $\mathrm{rdv} - \mathrm{ok}(r)$ pour leur signaler que le rendez-vous r a lieu, et met à jour les compteurs n_k.

Remarque: Avec un contrôleur centralisé, les relations (S1') et (S2') autorisent une mise en œuvre des compteurs modulo 2 (1 bit suffit).

UNE FAMILLE DE PROTOCOLES
PRINCIPE DES CONTRÔLEURS ET DU JETON

- Distribution de la solution centralisée précédente.
- Le coordinateur central étant mis en œuvre par les contrôleurs $CTL_1, CTL_2, \ldots, CTL_n$, placés sur différents sites.
- Un rendez-vous r peut alors être géré par un seul contrôleur (gestion centralisée) ou par plusieurs (gestion dupliquée). Ainsi si l'on choisit:
  - Un seul contrôleur qui gère tous les rendez-vous, on tombe sur la solution centralisée.
  - N contrôleurs chacun gérant tous les rendez-vous, on tombe sur une solution distribuée par duplication.
  - Un certain nombre de contrôleurs tel que chaque rendez-vous n'est géré que par un seul d'entre eux, on obtient une solution distribuée par partitionnement.
  - Toutes les solutions intermédiaires étant possibles, la seule contrainte étant que tout rendez-vous doit être géré par au poins un contrôleur.

UNE FAMILLE DE PROTOCOLES (SUITE)
PRINCIPE DES CONTRÔLEURS ET DU JETON

- La possibilité d'avoir des rendez-vous conflictuels rend nécessaire un mécanisme d'exclusion mutuelle entre les contrôleurs afin qu'un seul de ces rendez-vous soit activé.
- Un jeton d'exclusion va être utilisé à cette fin: On place les contrôleurs sur un anneau et seul le contrôleur possesseur du jeton peut prendre des décisions.

L'ALGORITHME
CONTEXTE ET MESSAGES

Processus: Chaque processus i est doté de:

- La variable $état_i$
- Des deux compteurs w_i et n_i
- La variable $ctl - par_i$: ensemble des identités des contrôleurs qui gèrent les rendez-vous auxquels le processus i peut participer. Ces rendez-vous constituent RDV_i.

$$ctl - par_i = \{j : CTL_j \text{ gère } r, \forall r \in RDV_i\}$$

L'ALGORITHME

CONTEXTE ET MESSAGES (SUITE)

Contrôleur:

- Chaque contrôleur CTL_j est associé un certain nombre de rendez-vous qu'il va gérer, cette association est le résultat d'un choix de distribution.
- On appelle $rdv - ctl_j$ cet ensemble de rendez-vous et $proc - ctl_j$ l'ensemble des processus intervenant dans les rendez-vous géré par CTL_j.
- Afin de pouvoir gérer les rendez-vous dont il a la charge, le contrôleur CTL_j est doté, pour chaque processus $i \in proc - ctl_j$, d'une variable locale $nbprêt_j(i)$. Il s'agit d'une image locale du compteur w_i; comme le processus envoie un message $prêt(i)$ à chaque incrémentation de w_i on a l'invariant:

$$\boldsymbol{nbprêt_j(i) \leq w_i}$$

L'ALGORITHME

CONTEXTE ET MESSAGES (SUITE)

Messages:

- Message $prêt$ envoyé par chaque processus qui invoque un rendez-vous aux contrôleurs gérant ses rendez-vous.
- Messages $rdv - ok(r)$ émis dans l'autre direction informant les processus que le rendez-vous r est activé.
- Le jeton est implémenté par un message véhiculant un ensemble de valeurs permettant au contrôleur qui le possède de décider, en conjonction avec ses variables locales, si des rendez-vous sont activables. L'invariant (S1') et le fait que les contrôleurs ont des images des w_i conduisent à placer dans le jeton des valeurs qui représentent les compteurs n_i, c'est-à-dire:

$$\boldsymbol{jeton}(\{\boldsymbol{nx_i}: \boldsymbol{1 \leq i \leq n}\})$$

L'ALGORITHME

CONTEXTE ET MESSAGES (SUITE)

Messages:

- Le compteur $nx(i)$ va être géré comme une image du compteur n_i. Une telle image va être à mise à jour anticipée ($nx(i)$ sera incrémentée avant n_i). on a l'invariant:

$$\forall \boldsymbol{i}: \boldsymbol{n_i} \leq \boldsymbol{nx(i)}$$

- Avec ces éléments l'invariant (S2') peut être réécrit comme suit:

$$\forall \boldsymbol{i}: \boldsymbol{n_i} \leq \boldsymbol{w_i} \leq \boldsymbol{n_i} + \boldsymbol{1}$$

$$\forall \boldsymbol{i}: \boldsymbol{n_i} \leq \boldsymbol{nx(i)} \leq \boldsymbol{w_i} \leq \boldsymbol{n_i} + \boldsymbol{1} \leq \boldsymbol{nx(i)} + \boldsymbol{1}$$

L'ALGORITHME

COMPORTEMENT DES SITES

Le comportement de tout site i est décrit par les deux énoncés suivants:

Lors de l'invocation de ($r1\ ou\ r2\ ou\ \ldots, ou\ rk$)

$$\%RDV_i = \{r1, r2, \ldots, rk\}$$
$$état_i = demandeur;$$
$$\forall\ CTL_j \in ctl - par_i : envoyer\ prêt(i)\ \text{à}\ CTL_j;$$
$$w_i + +;$$

Lors de la réception de $rdv - ok(r)$

$$état_i = dedans;$$
$$n_i + +;$$

L'ALGORITHME

COMPORTEMENT DES CONTRÔLEURS

Le comportement de tout contrôleur CTL_j est décrit par les deux énoncés correspondant aux deux réceptions de messages qu'il peut subir.

Lors de réception de $prêt(i)$

$\%\ i \in proc - ctl_j$

$nbprêt_j(i) + +$

Lors de la réception de $jeton(\{nx_i : 1 \leq i \leq n\})$

$\forall\ r\ \in rdv - ctl_j$

Faire **Si** $(\forall i \in r : nbprêt_j(i) = nx(i) + 1)$

Alors $\forall i \in r$ **Faire** {envoyer $rdv - ok(r)$ à i;

$nx(i) + +$;}

Finsi

Fait;

envoyer $jeton(\{nx_i : 1 \leq i \leq n\})$ **au contrôleur suivant**

MISE EN ŒUVRE DES COMPTEURS

Les compteurs utilisés satisfont les 3 invariants suivantes:

P1: $0 \leq nbprêt_j(i) \leq w_i$

P2: $n_i \leq nx(i) \leq n_i + 1$

P2: $nx(i) \leq w_i \leq nx(i) + 1$

Il est clair que les compteurs ne font que croitre, il y aurait un dépassement de capacité après un certain temps. Il faut trouver une solution pour les bornés.

Protocole de remise à zéro:

Soit $max + 1$ une valeur constituant un seuil à ne pas dépasser. Les relations P1, P2 et P3 permettent de conclure:

$$(nx(i) < max \Rightarrow (n_i < \max \; et \; w_i < max + 1 \; et \; nbprêt_j(i) < max + 1)$$

Il suffit donc de surveiller la croissance des compteurs $nx(i)$ contenus dans le jeton et d'exécuter un protocole de remise à zéro lorsqu'un tel compteur atteint $max - 1$.

PROTOCOLE DE REMISE À ZÉRO

Lancé par un contrôleur CTL_j possesseur du jeton, il met en jeu tous les processus et tous les autres contrôleurs CTL_k.

Hypothèse: Les canaux sont supposés FIFO (Un message raz ne doit pas doubler un message $prêt$ entre un processus et un contrôleur CTL_k pour que la variable $nbprêt_k(i)$ ait une valeur correcte).

Pour lancer la remise à zéro:

Condition: (CTL_j a le jeton) et ($\exists i$ tel que $nx(i) = max - 1$)

Faire 1) $1 \leq i \leq n$: $nx(i) = 0$; %raz du jeton

2) $1 \leq i \leq n$: $envoyer\ raz(CTL_j)$ à i;

3) attendre ack de chaque contrôleur CTL_k;

Fait $\%CTL_j$ peut utiliser ou transmettre le jeton

PROTOCOLE DE REMISE À ZÉRO (SUITE)

Lorsqu'un processus i **reçoit** $raz(CTL_j)$

$n_i = 0$; %raz des compteurs de i

Si $état_i = demandeur$ **Alors** $w_i = 1$ **Sinon** $w_i = 0$ Finsi

$\forall CTL_k \in ctl - par_i$: $envoyer\ raz(CTL_j, i, w_i)$ àCTL_k;

Lorsqu'un contrôleur CTL_k **reçoit** $raz(CTL_j, i, x)$

$nbprêt_k(i) = x$;

Si CTL_k a reçu un message $raz(\ ,\ ,\ )$ de chaque processus $i \in proc - ctl_k$

Alors envoyer ack à CTL_j

Finsi

Le contrôleur CTL_j informe tous les processus de la remise à zéro; chacun d'eux la réalise et en informe ensuite tous les contrôleurs qui gèrent ses rendez-vous, et ce n'est que lorsque tous les contrôleurs ont effectué la remise à zéro de tous leurs compteurs que le protocole est terminé. Le jeton ne peut pas être utilisé pour produire des rendez-vous durant cette exécution.