

# Contrôle de la brique ISAR

## Lundi 26 juin 2006, 10h15 à 11h45

### Version avec corrigé

Sans documents, dictionnaire autorisé pour les non francophones

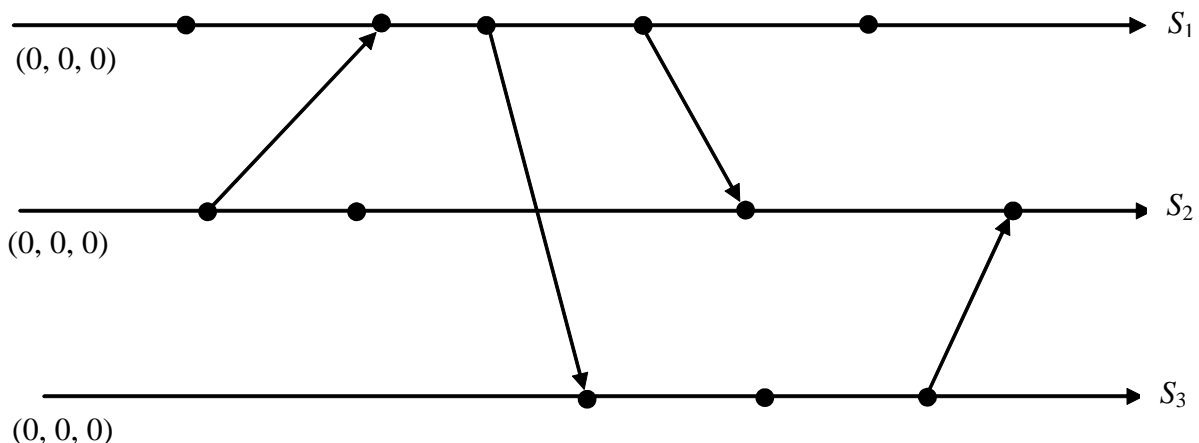
Le barème est indicatif

## I. Partie 1 : Algorithmique répartie (6 points)

N.B. répondre aux questions de cette partie sur une feuille séparée portant votre nom.

**Question 1 (1,5 points)** : Le dessin ci-dessous représente le déroulement du temps sur trois sites ; chaque ligne horizontale correspond à un site, le temps s'écoule de la gauche vers la droite. Chaque point noir correspond à un événement. Chaque flèche correspond à un message envoyé d'un site à l'autre. Les sites utilisent des horloges vectorielles. Indiquer à côté de chaque événement la date de celui-ci et à côté de chaque flèche l'estampille du message correspondant.

*Réponse*



**Question 2 (1,5 points)** : Donner la définition de ce qu'on appelle « état global d'un système réparti ».

*Réponse*

**Question 3 (3 points)** : Décrire un algorithme assurant l'exclusion mutuelle ; l'algorithme doit vérifier les propriétés suivantes :

- il ne doit pas consister à faire tourner sur un anneau un jeton qui tournerait même si aucun processus n'était demandeur de la ressource (c'est-à-dire ne devra pas être l'algorithme de Le Lann décrit dans le polycopié) ;
- le code de l'algorithme doit être le même sur tous les sites, sauf en ce qui concerne l'initialisation des variables ;
- l'algorithme doit éviter l'interblocage et la famine.

Tout algorithme vérifiant ces propriétés est autorisé, même s'il ne fait pas partie des algorithmes présentés dans le polycopié.

L'essentiel est de décrire le principe général de l'algorithme. On donnera aussi les variables nécessaires sur chaque site pour mettre en place l'algorithme, l'initialisation de ces variables et le pseudo-code de la programmation faite sur chacun des sites. On indiquera enfin, en fonction du nombre  $n$  de sites, le nombre de messages nécessaires pour l'utilisation de la ressource, ou bien un encadrement de ce nombre.

*Réponse*

## **II. Partie 2 : Systèmes répartis 1 (4 points)**

**N.B. répondre aux questions de cette partie sur une feuille séparée portant votre nom.**

**Question II.1 (2 points)** On souhaite construire un service d'alertes météorologiques à destination du grand public, basé sur la technologie JMS. Indiquez en nommant le mode de communication utilisé, les différents éléments permettant à plusieurs clients de recevoir ces alertes et les filtrer suivant des critères tels que la localisation, la date, un seuil de température.

*Réponse :* on s'attend sur cette question à retrouver une application pub/sub, et donc un topic, des messagelistener, des filtres à un premier niveau de Réponse (niveau général), à un niveau plus détaillé, on peut ajouter les différentes étapes permettant à un client de recevoir ces messages (objets connection, session, etc). Tout est dans le TP d'ISAR sur JMS, question 3.

**Question II.2 (2 points).** Indiquez en quoi les libertés d'implantation offertes par JMS permettraient le passage à l'échelle de ce type d'applications ?

*Réponse :* JMS ne normalise que l'API, pas le protocole. Donc, on s'attend ici à une discussion sur les topologies de MOM : la possibilité de faire de la communication de groupe, l'absence de synchro lourde, le cote asynchrone des messages, etc.

## **III. Partie 3 : Systèmes répartis 2 (6 points)**

**N.B. répondre aux questions de cette partie sur une feuille séparée portant votre nom.**

On dispose de deux applications A1 et A2 utilisées chacune par plusieurs utilisateurs. A1 se charge de déposer dans une boîte aux lettres qui fait partie de ses structures de données internes des événements tels que des demandes de rendez-vous. A2 dispose également d'une boîte aux lettres interne qu'elle lit pour intégrer les événements dans un agenda. On souhaite propager les événements des instances de l'application A1 vers les instances de l'application A2 en utilisant le service d'événements COS Event de CORBA.

**Question III.1 (1 point).**

Rappeler les grands principes et l'architecture du service COS Event de CORBA. On détaillera notamment les composants d'Administration.

*Réponse :* Un objet Admin Producer (resp Consumer) se charge de créer des objets Producer (resp Consumer) pour les mettre en contact avec le Channel. Chaque Producer (resp Consumer) se charge de se mettre en relation avec son proxy distant Consumer (resp Producer). Il existe deux modes Push et Pull pour les Producers et

*Consumers. Si un Producer est PullConsumer, son proxy est inverse ie ProxyPushProducer.*

**Question III.2 (1 point).** Indiquer dans quels modes (pour chacune des applications) il convient de définir les composants de COS Event qui seront intégrés à A1 et A2 afin que les architectures des applications définies plus haut ne soient pas modifiées. Justifier vos choix. Expliciter la manière dont se fera la propagation et notamment dérouler un scénario de notification d'événements d'une instance de A1 vers une instance de A2.

*Réponse :* Comme on ne peut modifier les applications il faut un Producer en Pull (pour extraire les événements de la boîte aux lettres de A1), et Consumer en Push (pour déposer les événements dans la boîte aux lettres de A2). Le ProxyPushConsumer pousse les événements de A1 vers le PullProducer qui les délivre auprès du Channel. Le PushConsumer en bout de Channel envoie les événements qu'il reçoit auprès du ProxyPullProducer qui les dépose dans la boîte aux lettres de A2. Le Channel a donc un rôle tout à fait actif qualifié dans la littérature d'Agent.

**Question III.3 (1 point).** Expliciter les différences entre l'appel de procédure à distance et l'appel de méthode à distance. Notamment on comparera deux appels successifs à une procédure et deux appels successifs à une méthode.

*Réponse :* le serveur est déterminé dynamiquement lors de chaque appel de méthode à partir de la référence, alors que seul le premier appel de procédure peut donner lieu à une détermination dynamique.

**Question III.4 (1 point).** Expliciter les différences entre l'asynchronisme des objets en CORBA et l'asynchronisme des objets en DSA (Annexe des Systèmes Répartis en Ada). Expliquer cette différence qui provient de l'absence de surcharge de méthode en CORBA.

*Réponse :* en Ada, la classe est déclarée asynchrone et donc toutes les méthodes pouvant être asynchrones le sont. Dans l'approche qui consisterait à rendre ou non une méthode asynchrone, en raison de la surcharge, la même méthode d'une classe et celle de sa classe dérivée pourraient avoir un asynchronisme différent ce qui rendrait l'implémentation de la souche difficile. Il faudrait interroger le serveur pour savoir ou non si une réponse est attendue.

**Question III.5 (1 point).** Rappeler le modèle de distribution original proposé par DSA. Donner des exemples de supports pour ce modèle de distribution.

*Réponse :* il s'agit des objets partagés (l'accès se fait directement au niveau des données et non par des méthodes). Les supports - de stockage - peuvent par exemple être de la mémoire partagée, des fichiers partagés, des bases de données ...

**Question III.6 (1 point).** Tromboline définit une application composée des objets CORBA Alice, Bob et Charlie qui interagissent les uns avec les autres par l'intermédiaire de méthodes sans valeur de retour, sans paramètre inout ou out, sans levée d'exceptions. Foulbazar se gausse de Tromboline et prétend qu'une telle architecture peut être optimisée en passant les méthodes en mode oneway. Tromboline s'insurge et met en évidence les risques de cette approche. Aider Tromboline en fournissant un exemple où effectivement cette optimisation induira une erreur de comportement.

**Réponse :**  $A \rightarrow B.M1 \rightarrow C.M1$  puis  $A \rightarrow C.M2$ . Si  $B.M1$ ,  $C.M1$  et  $C.M2$  sont oneway, l'exécution de  $C.M2$  peut survenir avant celle de  $C.M1$ . Donc le passage en oneway peut modifier l'ordre d'invocation des méthodes sur  $C$ .

#### IV. Partie 4 : Systèmes répartis 3 (6 points)

N.B. répondre aux questions de cette partie sur une feuille séparée portant votre nom.

##### Question IV.1 (3 points)

(a. 1 point) Rappeler l'architecture du système [SETI@HOME](#)

(b. 2 point) [SETI@HOME](#) est souvent qualifié de système pair-à-pair ? Donner un argument en faveur de cette qualification et un argument contre.

**Réponse :**

(a) Le "maître" coupe les données collectées par le télescope en sous-unités à analyser. Chaque participant se porte volontaire pour analyser une ou plusieurs sous-unités. Les résultats sont collectés et synthétisés par le maître.

(b) Tout participant se porte volontaire pour analyser les données donnant ainsi un peu de ses ressources, ce qui correspond à la logique pair-à-pair. En revanche le processus de découpe, de distribution, de collecte et de synthèse des résultats est centralisé ce qui met en défaut cette logique.

##### Question VI.2 (1 point)

A quel besoin répond le patron de conception intercepteur (interceptor) ?

**Réponse :** transformation d'un service en interposant une nouvelle couche de traitement et/ou en changeant la destination de l'appel.

##### Question VI.3 (1 point)

A quoi correspond la 3<sup>ème</sup> génération de GRID ?

**Réponse :** La 3<sup>ème</sup> génération de "GRID", matérialisée par le standard OGSA, introduit une approche orientée services (s'appuyant sur les services web), et permet la création, la maintenance et l'utilisation de services maintenus par des organisations virtuelles.

##### Question VI.5 (1 point)

L'exemple ci-dessous respecte-t-il la cohérence causale (même notations qu'en cours)

P1	W(x)1		
P2		R(x)1	W(x)2
P3			R(x)1    R(x)2
P4			R(x)2    R(x)1

**Réponse :** NON car les événements causalement dépendants ne sont vus dans un même ordre par tous les sites : les écritures  $W(x)1$  et  $W(x)2$  sont causalement dépendantes car  $P2$  a lu la valeur 1 en  $x$  avant d'écrire 2 en  $x$  ; cependant  $P3$  voit l'écriture de la

*valeur 1 dans x avant celle de la valeur 2 dans x et P4 voit les deux opérations d'écriture dans l'ordre inverse.*