Examen de la demi unité de valeur

SYSTÈMES ET APPLICATIONS RÉPARTIS

Durée 2 heures

Première session juin 2004

DOCUMENTS AUTORISES : TOUS

_____

Pour chaque question il est demandé une justification de la réponse proposée.

Certaines questions peuvent avoir des énoncés relativement longs. Cette longueur est uniquement due à une volonté de spécifier correctement les questions. Il n'y a pas de corrélation entre la longueur d'un texte et la difficulté d'une question.

Il est fortement recommandé de lire tous les énoncés avant de commencer à traiter les questions de façon à aborder le maximum de points.

Problème I
Horloges logiques et horloges physiques : causalité (12 points)

Nous nous intéressons tout d'abord au temps logique dans un système réparti avec n sites. Une caractérisation de la dépendance causale entre deux événements est donnée au moyen des horloges vectorielles de Fidge et Mattern (vecteurs Vi à n composantes qui résument l'histoire causale).

I.1) Rappelez la définition des horloges vectorielles de Fidge et Mattern et leur mode de fonctionnement ? (1 point)

I.2) Les horloges vectorielles permettent de définir une condition nécessaire et suffisante de causalité: Ei ? Ej $\Leftrightarrow$ Vi(Ei) < Vj(Ej) . Deux événements sont dans la relation de causalité (dans la relation arrive_avant ou en anglais happened_before) si et seulement si les horloges logiques associées sont dans la relation d'inégalité entre les vecteurs d'horloges associés aux deux événements. Expliquez pourquoi, dans le cas général, il n'est pas possible de caractériser la dépendance causale dans un système réparti à n sites avec une information vectorielle qui comporterait toujours moins de n composantes (énoncez des arguments ou mieux construisez un raisonnement) (2 points).

Remarque: on s'intéresse dans cette question au rôle de chaque composante dans le cas général. On ne considère pas certains cas particuliers ou la nature du problème à résoudre permet de simplifier la solution générale des horloges de Fidge et Mattern (voir question suivante). On ne considère pas non plus des techniques de compression, exploitant les redondances, qui permettent de diminuer l'encombrement des informations qui codent les n composantes entières d'un vecteur d'horloge dans une suite de messages.

I.3) On considère maintenant un système client serveur comportant *m* serveurs et *n* clients. Dans l'application considérée les clients sont totalement indépendants et ne communiquent jamais entre eux. Les seules communications existants dans ce système sont des communications entre un client et un serveur, ou entre deux serveurs. En tenant compte de cette restriction sur les communications, est-il possible d'améliorer la méthode des horloges vectorielles pour la rendre plus efficace (en nombre de composantes nécessaires dans les vecteurs d'horloge) tout en permettant toujours la caractérisation de la dépendance causale ? Que se passe t'il si les serveurs sont sans état (exemple de serveurs sans état, les serveurs HTTP ou NFS)? (2 points)

Chaque calculateur dans un système réparti dispose généralement d'une horloge physique (horloge temps réel), qui délivre sur requête de lecture, une valeur qui mesure le temps physique.

I.4) On suppose tout d'abord que les horloges temps réel d'un ensemble de calculateurs formant un système réparti peuvent être parfaitement synchronisées (c'est-à-dire que tous les calculateurs obtiennent au même instant le même résultat de lecture de l'horloge dont ils disposent). Quels avantages pourrait-on tirer dans la résolution des problèmes d'algorithmique répartie de la disposition d'une telle fonctionnalité. Citez des problèmes d'algorithmique répartie plus facilement résolus dans le cas ou les horloges physiques sont parfaitement synchronisées (2 points).

Pour vous aider à commencer votre réflexion voici une liste non limitative et sans ordre de problèmes d'algorithmique répartie (vous pouvez bien évidemment utiliser d'autres problèmes): diffusions totalement ou causalement ordonnées, points de reprise, gestions de cohérence mémoire partagée réparties, gestion de la concurrence dans les bases de données partiellement ou totalement répliquées, élection, traitement de l'interblocage, exclusion mutuelle en univers réparti, répartition de charge, ordonnancement réparti… Il ne s'agit pas de prendre au hasard dans la liste mais de justifier l'utilisation d'horloges physiques synchronisées dans la solution de chaque problème.

I.5) Le fait de construire des ensembles de machines ayant des horloges temps réel synchronisées est-il complètement impossible ou pouvez vous concevoir des solutions avec lesquelles on peut approcher de façon satisfaisante cette propriété de synchronisation ? (2 points)

Nous supposons maintenant que les horloges physiques de différents sites d'un système réparti sont synchronisées au moyen d'un protocole de synchronisation d'horloge par messages. On note HCi(t) le résultat de la lecture de l'horloge du site i à un instant t. Dans la notation précédente, t est un temps universel abstrait utilisé uniquement pour noter la date absolue d'un événement.

Un protocole de synchronisation d'horloges échange dans des messages les valeurs des horloges physiques de façon à ce que chaque horloge délivre la même heure. Compte tenu des choix protocolaires et des délais de transmission des messages ce protocole est plus ou moins bon. Nous supposons ici que le protocole de synchronisation d'horloges permet d'assurer en toutes circonstances une précision donnée d. C'est à dire que la différence des valeurs lues au même instant sur deux horloges physiques de deux sites différents i et j est toujours bornée par la valeur d (fonction du protocole utilisé et des caractéristiques du réseau). Quelque soient les sites i et j et un instant t on a :

$$|HCi(t) - HCj(t)| < d$$

Dans cette question, on suppose également que les communications sont à délais bornés. On peut définir une borne inférieure dmin et une borne supérieure dmax pour le délais de transmission d'un message entre deux sites quelconques du système.

I.6) À quelles conditions les horloges HCi, synchronisées par un protocole de synchronisation d'horloge ayant les caractéristiques précédentes, respectent-elles la causalité (3 points).

De manière plus précise soient deux événements différents Ei , Ej observés sur deux sites i et j qui sont dans la relation de causalité Ei ? Ej (Ei s'est produit avant Ej). Si l'on appelle t(Ei) et t(Ej) les dates absolues de ces deux événements, à quelles conditions les valeurs lues sur les horloges des deux sites sont elles dans la même relation d'ordre :

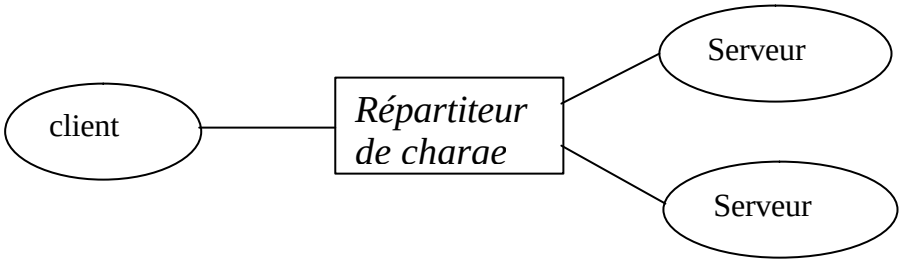
$$Ei \; ? \; Ej \Rightarrow HCi(t(Ei)) < HCj(t(Ej))$$

De la sorte, la datation physique est cohérente avec l'ordre causal.

Indication : il y a plusieurs conditions mais la plus importante à trouver relie de façon simple d et dmin.

Problème II: Répartition de charge pour des serveurs Corba (8 points)

Une application Internet client/serveur, développée en CORBA, pose des problèmes de performances. De façon à améliorer le service aux usagers on se propose de répliquer le serveur. La solution retenue prend pour objectif principal de ne pas modifier le comportement des clients. Pour cela on décide d'acheter une (ou plusieurs) nouvelles machines serveur et un répartiteur de charge. Un répartiteur de charge est un système informatique (calculateur et logiciel de répartition) auquel des clients s'adressent comme à un serveur et reçoivent les réponses comme d'un serveur. Le répartiteur, en fonction de son algorithme de sélection, s'adresse à l'un des serveurs effectif de l'application et lui demande d'effectuer le travail demandé par le client.



L'application CORBA, réalisée par les serveurs, est très simple. L'application offre des accès à des données identifiées par une clé qui est de type chaîne de caractère. On peut lire, écrire ou créer une donnée. Les valeurs associées à une clé sont également des chaînes de caractères. On gère deux exceptions : un signal d'exception lorsque une clé n'existe pas (cle_inexistante) et un signal d'exception lorsqu'une clé existe déjà (cle_deja_utilisee).

II.1) Donnez l'interface IDL CORBA de l'application réalisée par les serveurs (1 point).

De manière à préciser les hypothèses de fonctionnement de l'application, on rajoute au mode de fonctionnement décrit à la question précédente qu'un client peut faire des accès successifs formant une session de travail.

II.2) Proposez une seconde interface en IDL CORBA définissant le fonctionnement de la relation client/serveur dans le cas ou l'accès au serveur se fait dans le cadre de sessions de travail. Les sessions sont identifiées et chaque opération est effectuée dans le cadre d'une session délimitée dans le temps. Si une opération est effectuée dans une session inexistante, on doit lever une exception (2 points).

II.3) L'interface IDL CORBA du répartiteur de charge, visible par les clients, est la même que l'interface des serveurs. Lorsqu'un client souhaite accéder à un serveur de données, il s'adresse au répartiteur de charges. Celui-ci sert de point d'entrée vers les différents serveurs (il fait office de serveur proxy). Par contre, lorsqu'un nouveau serveur est ajouté au groupe des serveurs redondants, le répartiteur de charge doit être capable d'enregistrer ce serveur. De même il doit le retirer de sa liste lorsque celui-ci est interrompu (par exemple pour des opérations de maintenance). Comment sont gérés les noms dans l'architecture client/ répartiteur /serveur au niveau du service de noms CORBA (quels sont les noms connus des clients, du répartiteur, des serveurs) ? (2 points)

II.4) Donner l'interface IDL CORBA du répartiteur permettant de gérer l'enregistrement et le retrait des serveurs (interface CORBA du service de gestion de groupe des serveurs) ? (2 points)

II.5) Que préconisez-vous pour remédier au problème de sûreté de fonctionnement de cette architecture? (1 point)