



Introduction aux systèmes répartis

Exercices pour le Module 1

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Introduction aux systèmes répartis

1 Systèmes répartis

2 Réseautique



Partage en réseau

Donnez 5 types de matériel et de données/logiciels qui peuvent être partagés en réseau?



Partage en réseau

Donnez 5 types de matériel et de données/logiciels qui peuvent être partagés en réseau?

Le partage d'imprimantes, de disques, et d'autres périphériques (numériseur, unité d'archivage...) est fréquent.

Les serveurs de calcul permettent de partager un CPU ou GPU rapide

Les données partagées peuvent être des pages Web, des fichiers, du vidéo, ou du code source. Par exemple, un serveur Web comme Apache rend des fichiers html disponibles à tous, ou un serveur comme Squid permet de conserver en mémoire, localement sur un serveur du réseau, les pages Web statiques fréquemment accédées agissant comme un cache partagé.



Synchronisation

Comment peut-on synchroniser les horloges entre les ordinateurs sur un réseau?



Synchronisation

Comment peut-on synchroniser les horloges entre les ordinateurs sur un réseau?

Il faut envoyer un message demandant l'heure et recevoir la réponse. L'imprécision est causée par les délais d'envoi. Si les délais sont symétriques, ils peuvent être calculés et leur effet compensé. Il est cependant impossible par un simple envoi de messages de savoir quelle est leur valeur relative.

Pour synchroniser plusieurs ordinateurs, il est préférable d'avoir une structure hiérarchique pour éviter que les imprécisions ne s'accumulent trop.



Initier une connexion

Comment peut-on établir une connexion réseau si on ne connaît pas les paramètres du réseau local, par exemple sur une tablette en entrant dans une gare?



Initier une connexion

Comment peut-on établir une connexion réseau si on ne connaît pas les paramètres du réseau local, par exemple sur une tablette en entrant dans une gare?

En supposant que réseau puisse s'établir physiquement, par infra-rouge ou micro-ondes (IRda ou IEEE 802.11), il faut un point de ralliement. Le périphérique peut envoyer un message à tous pour demander les informations de base du réseau, par exemple par DHCP/BOOTP qui fournit une adresse IP et plusieurs informations (réseau, masque, passerelle...). Le réseau local pourrait aussi envoyer périodiquement un message à tous contenant les informations de base ainsi qu'un URL pour trouver de l'information supplémentaire.



Interaction entre objets

Deux objets qui se trouvent sur des ordinateurs différents et qui sont implantés dans des langages différents doivent interagir, comment cela est-il possible?



Interaction entre objets

Deux objets qui se trouvent sur des ordinateurs différents et qui sont implantés dans des langages différents doivent interagir, comment cela est-il possible?

Le protocole TCP/IP fournit un service de communication indépendant de la plate-forme. Des bibliothèques pour accéder ce service et d'autres fonctions de base servent à isoler chaque programme de sa plate-forme. Une bibliothèque pour l'interaction entre les objets doit exister (e.g. CORBA) et doit aussi masquer les différences. Finalement, un format standard d'échange doit être établi pour toutes les données échangées (entiers, chaînes de caractères, structures...) de manière à tenir compte des différences imputables à la plate-forme et aux langages.



Appel de procédure à distance

Lors d'un appel ou de l'invocation d'une méthode à distance, comparé à une opération interne à un programme, quelles sont les sources possibles de problèmes?



Appel de procédure à distance

Le processus serveur peut être inopérant, le processus client peut défaillir, le réseau peut être inopérant, le système d'exploitation de l'un ou l'autre pourrait défaillir, un bris matériel à l'un ou à l'autre peut arriver. Il se peut aussi que tout fonctionne mais que le réseau soit extrêmement lent.

Lorsque le client est en panne, l'utilisateur peut souvent s'en rendre compte rapidement. Si le processus serveur n'existe plus, le client s'apercevra aussi rapidement d'un problème car la connexion sera refusée par le système d'exploitation du serveur. Si le processus serveur, l'ordinateur serveur, ou le réseau ne répond pas alors qu'il le devrait, il est impossible de savoir où se trouve le problème et il faut attendre longtemps avant d'être certain qu'il y a un problème et que ce n'est pas simplement un délai causé par un réseau surchargé. S'il n'y a pas de limite au temps que peut prendre un réseau surchargé, il faut attendre un temps infini avant d'être certain que le service est en panne et que ce n'est pas le réseau qui est lent!

Migration de ressources

Lorsque plusieurs serveurs sont utilisés, pourquoi voudrait-on migrer des ressources (e.g., fichiers sur disque, programmes en exécution...) entre ces ordinateurs? Comment suivre les ressources qui migrent?



Migration de ressources

Lorsque plusieurs serveurs sont utilisés, pourquoi voudrait-on migrer des ressources (e.g., fichiers sur disque, programmes en exécution...) entre ces ordinateurs? Comment suivre les ressources qui migrent?

Ceci peut être fait pour des raisons de proximité (minimiser le délai et la charge du réseau) ou pour équilibrer la charge de calcul, de stockage, ou de réseau.

Il serait possible de faire un appel à tous pour retrouver les ressources mais cela peut être inefficace. Maintenir un répertoire de localisation des ressources, ou conserver un pointeur vers la nouvelle localisation sur l'ancien site sont souvent beaucoup plus efficaces. On pourrait aussi avertir les clients du changement d'adresse mais ce n'est habituellement pas la meilleure solution car souvent les clients sont très nombreux et moins fiables.

Synchronisation de données réparties

Quels sont les besoins de synchronisation associés à un système réparti d'indexation et de recherche pour l'Internet?



Synchronisation de données réparties

Quels sont les besoins de synchronisation associés à un système réparti d'indexation et de recherche pour l'Internet?

Plusieurs ordinateurs peuvent bâtir morceau par morceau le nouvel index pendant que la copie actuelle est toujours en utilisation. Une fois que tous les morceaux sont fusionnés, le nouvel index peut être propagé partout et, au moment choisi, le remplacement se fait. Les nouvelles requêtes utilisent le nouvel index.



Code mobile

Donnez des exemples de code mobile?

Code mobile

Donnez des exemples de code mobile?

Il est utile d'avoir du code mobile lorsqu'un traitement est plus efficace si effectué au bon endroit. Par exemple, un applet permet d'exécuter une application directement dans le fureteur du client, sans occuper le CPU du serveur, et en évitant les délais associés au réseau.

De la même manière, il est utile d'avoir certains programmes d'analyse des données qui sont envoyés et exécutés par le serveur de base de données plutôt que de transmettre tout le contenu de la base de donnée par le réseau.



Tolérance aux défaillances

Donnez des exemples de défaillances logicielles et matérielles qui peuvent ou ne peuvent pas être tolérées grâce à la redondance?

Tolérance aux défaillances

Donnez des exemples de défaillances logicielles et matérielles qui peuvent ou ne peuvent pas être tolérées grâce à la redondance?

Les défaillances indépendantes peuvent être masquées ou tolérées dans un système redondant (disques RAID, processeurs TMR, mémoire ECC, réseau redondant...). D'autres défaillances sont reliées et ne peuvent donc pas nécessairement être masquées: erreur de programmation, catastrophe naturelle dans la région géographique où sont localisés les serveurs redondants, action de vol, sabotage, vandalisme concertée.



Introduction aux systèmes répartis

1 Systèmes répartis

2 Réseautique



Temps de service d'une requête

Un client envoie une requête de 200 octets et reçoit une réponse de 5000 octets. Le temps de traitement de la requête est de 2ms. En local, avec les processus client et serveur sur le même ordinateur, une latence de 1ms s'ajoute (e.g., changement de contexte), en plus du temps de transfert mémoire-mémoire pour la requête et la réponse à 500Mo/s. Par réseau, chaque paquet prend une latence de 5ms en plus du temps de transfert à 10Mbits/s. Les paquets ont une taille maximale sur ce réseau de 1000 octets. Si une connexion TCP doit être établie, une latence de 5ms s'ajoute pour initier la connexion au début de la requête. Dans ce contexte, quel est le temps total requis avec pour recevoir la réponse à une requête en local, par UDP et par TCP?



Temps de service d'une requête

Un client envoie une requête de 200 octets et reçoit une réponse de 5000 octets. En local nous avons la latence de 1ms, le transfert de 200 octets à 500Mo/s, le traitement de 2ms, la latence de 1ms, et le transfert de 5000 octets à 500MO/s.

$$1ms + 200/500000\text{octets/ms} + 2ms + 1ms + 5000/500000\text{octets/ms} = 4.0104ms$$

Par réseau avec UDP (sans connexion) on a la latence de 5ms, le transfert de 200 octets à 10Mbits/s, le traitement de 2ms et la réponse en 5 paquets de 1000 octets, chacun demandant une latence de 5ms et le transfert de 1000 octets à 10Mbits/s.

$$5ms + (200*8)/10000\text{bits/ms} + 2ms + 5(5 + (1000*8)/10000\text{bits/ms}) = 36.16ms$$

Par réseau avec TCP, il faut simplement ajouter 5ms pour initier la connexion. Nous ne tenons pas compte des accusés de réception prévus dans TCP qui pourront arriver plus tard mais qui ne retardent pas la réception de la réponse.

$$5ms + 5ms + (200*8)/10000\text{bits/ms} + 2ms + 5(5 + (1000*8)/10000\text{bits/ms}) = 41.16ms$$

Routeur

Quel est la tâche d'un routeur sur l'Internet? Quelles tables doit-il maintenir?

Routeur

Quel est la tâche d'un routeur sur l'Internet? Quelles tables doit-il maintenir?

Un routeur connecte plusieurs réseaux IP. Il reçoit des paquets de ces réseaux et pour chacun doit décider à quel réseau l'envoyer. Il doit avoir une table qui dit, pour chaque groupe d'adresse IP possible, vers quel réseau l'envoyer pour que le paquet atteigne sa destination par le meilleur chemin possible. Etant donné le très grand nombre de réseaux sur l'Internet, on ne peut nécessairement avoir une table complète à chaque routeur. A la place, des routes par défaut et des regroupements de réseaux existent et servent à minimiser les tables (e.g. route par défaut vers un routeur avec une table plus complète, groupe de réseaux qui commencent par XXXX pour lesquels les paquets doivent tous passer par un routeur en Asie de chemin connu).

Unicité des adresses

Comment sait-on que toutes les adresses IP sont uniques?



Unicité des adresses

Comment sait-on que toutes les adresses IP sont uniques?

Un centre s'occupe d'allouer les adresses IP et ne donne jamais la même adresse 2 fois. Si quelqu'un choisit une adresse au hasard sans passer par ce centre, cela peut causer des ennuis sérieux.

Avec ou sans connexion

Quel est le meilleur choix, TCP ou UDP, pour chacune des applications suivantes: Telnet, FTP, HTTP, RPC?



Avec ou sans connexion

Quel est le meilleur choix, TCP ou UDP, pour chacune des applications suivantes: Telnet, FTP, HTTP, RPC?

Telnet: la fiabilité est importante et la performance n'est pas un problème, TCP est plus simple à utiliser.

FTP: pour le transfert de très gros fichiers, les possibilités de fenêtres d'accusés de réception, retransmission, contrôle du débit... de TCP sont très utiles.

HTTP: pour de courtes réponses, UDP pourrait être intéressant.

Cependant, les transferts impliquent assez souvent de gros fichiers ou des requêtes multiples sur la même connexion et TCP est donc utilisé.

RPC: les messages sont généralement courts et un système de délai expiré est déjà implanté. UDP est donc un choix efficace dans ce contexte.

Localisation d'un service

Un serveur ouvre un port et lui assigne un nom (numéro).
Comment les clients peuvent-ils s'y connecter?

Localisation d'un service

Un serveur ouvre un port et lui assigne un nom (numéro).
Comment les clients peuvent-ils s'y connecter?

L'adresse d'un service est constituée de l'adresse IP et du numéro de port. On peut utiliser une adresse pré-établie (e.g. 132.207.99.1 pour la passerelle du réseau 132.207.99), un nom qui doit être communiqué (e.g. charles.polymtl.ca), ou un nom par convention (e.g. www.polymtl.ca, news.polymtl.ca).

Pour les numéros de ports, les numéros pré-établis sont souvent utilisés (e.g. 21 FTP, 22 SSH, 23 TELNET). Le service portmap existe aussi mais est principalement utilisé pour les Sun RPC.



Problèmes de sécurité

Discutez des principaux types de problèmes de sécurité sur l'Internet.

Problèmes de sécurité

Discutez des principaux types de problèmes de sécurité sur l'Internet.

Le premier type d'attaque est d'utiliser un processus sur le serveur pour y prendre pied. Ce processus peut être mal configuré (contrôle des accès), mal protégé (mot de passe par défaut), ou mal programmé (débordement de tampon). Un deuxième type d'attaque est de corrompre le réseau en espionnant ce qui s'y passe ou en falsifiant la provenance de messages. Un troisième type d'attaque est le déni de service en surchargeant un serveur ou un réseau de manière à le rendre inopérant.



File d'attente

Un service peut traiter $u=200$ requêtes par seconde. Il en reçoit présentement $l=100/s$ et on estime qu'il en recevra $l=180/s$ une prochaine journée. Quelle est N le nombre moyen de requêtes dans le système et R le temps de réponse moyen R pour ces deux cas?

File d'attente

Un service peut traiter $u=200$ requêtes par seconde. Il en reçoit présentement $I=100/s$ et on estime qu'il en recevra $I=180/s$ une prochaine journée. Quelle est N le nombre moyen de requêtes dans le système et R le temps de réponse moyen R pour ces deux cas?

Le taux d'utilisation U est de $I/u = 100/200 = 0.5$ pour le premier cas et de $180/200 = 0.9$ dans le second. Le nombre moyen de requêtes $N = 0.5 / (1 - 0.5) = 1$ pour le premier cas et $0.9 / (1 - 0.9) = 9$ dans le second cas. Le temps de réponse est de $R = N/I = 1 / 100/s = .01s$ dans le premier cas et $9 / 180/s = .05s$ dans le second cas.



Nombre de threads

Un système reçoit des requêtes qui demandent en moyenne 5ms de cœur de CPU, 8ms de disque et 20ms d'attente pour une requête imbriquée. Ces requêtes sont réparties uniformément entre les ressources et sont traitées par un ordinateur avec 8 coeurs de CPU et 16 disques. Combien de requêtes par seconde est-ce que ce système peut traiter au maximum? Quel est le nombre minimal de threads requis pour ce nombre de requêtes?

Nombre de threads

Un système reçoit des requêtes qui demandent en moyenne 5ms de cœur de CPU, 8ms de disque et 20ms d'attente pour une requête imbriquée. Ces requêtes sont réparties uniformément entre les ressources et sont traitées par un ordinateur avec 8 coeurs de CPU et 16 disques. Combien de requêtes par seconde est-ce que ce système peut traiter au maximum? Quel est le nombre minimal de threads requis pour ce nombre de requêtes?

Les 8 coeurs de CPU peuvent servir $8 \times 1000\text{ms/s} / 5\text{ms/r} = 1600\text{r/s}$.

Les 16 disques peuvent servir $16 \times 1000\text{ms/s} / 8\text{ms/r} = 2000\text{r/s}$. Le facteur limitant est donc le CPU dans ce cas-ci. Puisque chaque requête prend au total $5\text{ms} + 8\text{ms} + 20\text{ms} = 33\text{ms}$. Nous avons donc $1600\text{r/s} \times .033\text{s/r} = 52.8$ requêtes en même temps dans le système et donc 53 threads.





Architecture des clients pour l'infonuagique

Exercices pour le Module 2

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Services gratuits

Certains sites offrent gratuitement certains services comme la conversion entre différents formats graphiques ou la comparaison de fichiers PDF et la production d'un PDF où les différences sont mises en évidence. Comment cela peut-il s'expliquer (e.g. avantage pécunier direct ou indirect, philanthropie) ?



Services gratuits

Certains sites offrent gratuitement certains services comme la conversion entre différents formats graphiques ou la comparaison de fichiers PDF et la production d'un PDF où les différences sont mises en évidence. Comment cela peut-il s'expliquer (e.g. avantage pécunier direct ou indirect, philanthropie) ?

Un site peut offrir un service gratuit car il trouve un avantage pécunier direct ou indirect à le faire. L'exemple des chaînes de télévision qui se payaient à l'aide des publicités existe depuis longtemps.

Un site peut lui aussi tirer des revenus des publicités affichées lors de l'accès au site. Il peut aussi tirer profit de la banque de données constituée de tous les documents qui lui sont soumis; souvent les conditions d'utilisation que personne ne lit permettent au site d'utiliser les documents. Le site pourrait aussi contenir des pages malicieuses qui essaient de corrompre les clients qui s'y connectent.

Finalement, il est possible que celui qui offre le service (gouvernement, université, chercheur, individu...) le fasse comme un service à la collectivité, ou encore simplement pour la gloire qui peut y être associée.

Choix du type de client

Une grande entreprise planifie son organisation informatique et doit décider comment répartir la charge entre les clients et les serveurs. Quatre modèles de clients de puissance de calcul différente sont disponibles. Pour chaque modèle, le prix et la puissance sont comme suit : i) \$200 2000 MIPS, ii) \$400 80000 MIPS, iii) \$1000 100000 MIPS, iv) \$2000 175000 MIPS. Pour les serveurs, le coût est estimé à \$2000 pour 100000 MIPS car il y a un surcoût important pour toute l'infrastructure qui les entoure (espace, réseautique...).

Pour un client donné, les applications suivantes sont exécutées pour une certaine fraction des heures de travail et avec une puissance consommée associée : a) 1500 MIPS 15%, b) 50000 MIPS 25%, c) 90000 MIPS 10%, d) 150000 MIPS 10%. Un client ne travaille qu'avec une seule application à la fois. Lorsqu'une application peut s'exécuter sur le client, car il a une performance suffisante, cela ne coûte rien pour le serveur. Autrement, l'application s'exécutera sur un serveur et demandera une certaine puissance pour une certaine fraction du temps, ce à quoi un prix peut être associé. Lequel client sera-t-il le plus rentable d'acheter pour cette compagnie ?

Choix du type de client

Avec le client i, il faudra dépenser \$200 pour le client et $50000 \text{ MIPS} \times .25 + 90000 \text{ MIPS} \times .10 + 150000 \text{ MIPS} \times .10 = 36500 \text{ MIPS}$ pour exécuter les applications trop grosses, pour un coût total de $\$200 + 36500 \text{ MIPS} \times \$2000 / 100000 \text{ MIPS} = \930 .

Avec le client ii, il faudra dépenser \$400 pour le client et $90000 \text{ MIPS} \times .10 + 150000 \text{ MIPS} \times .10 = 24000 \text{ MIPS}$ pour exécuter les applications trop grosses, pour un coût total de $\$400 + 24000 \text{ MIPS} \times \$2000 / 100000 \text{ MIPS} = \880 .

Avec les clients iii et iv, il faudra dépenser \$1000 et \$2000 respectivement pour le client, et la solution sera nécessairement moins avantageuse que la ii qui ne coûte que \$880 (e.g. pour le client iii on a $150000 \text{ MIPS} \times .10 = 15000 \text{ MIPS}$ pour exécuter les applications trop grosses, pour un coût total de $\$1000 + 15000 \text{ MIPS} \times \$2000 / 100000 \text{ MIPS} = \1300). La solution ii est donc la plus avantageuse.

Magasiner les applications

Les magasins d'applications peuvent contenir des millions d'applications différentes. Dans la mesure où les applications seront exécutées sur les téléphones des individus, comment s'assure-t-on d'un bon niveau de sécurité? Est-ce que les applications ont été vérifiées par le magasin qui les offre?

Magasiner les applications

Il existe un grand nombre d'applications malicieuses qui espionnent les utilisateurs, ou tentent de dérober des informations personnelles (e.g. bancaires). Les sociétés comme Google ou Apple n'ont pas le code source des applications ni le temps requis pour les analyser en détail. Certains tests, dont la nature reste secrète, sont effectués afin de détecter certains cas d'applications malveillantes qui sont alors retirées du magasin. De plus, chaque application n'a accès qu'à un certain nombre de permissions qui doivent lui être octroyées explicitement par l'utilisateur. Toutefois, la granularité des permissions n'est pas assez fine et ne permet souvent pas de détecter les excès. Ainsi, une application peut avoir une raison valable pour justifier de demander le droit d'accéder à la caméra, au micro, au réseau et au GPS (e.g. reconnaître un oiseau en fonction de l'image, du son, de la localisation et de l'interrogation d'un serveur) mais en abuser pour espionner les moindres faits et gestes de l'utilisateur.

Autonomie versus ne rien installer localement

Comment peut-on bénéficier de la simplicité d'utilisation d'un client sans logiciels ni données installés localement, et en même temps de la liberté d'un ordinateur autonome lors de voyages en avion ou de séjours dans la nature sauvage sans Internet?

Autonomie versus ne rien installer localement

Comment peut-on bénéficier de la simplicité d'utilisation d'un client sans logiciels ni données installés localement, et en même temps de la liberté d'un ordinateur autonome lors de voyages en avion ou de séjours dans la nature sauvage sans Internet?

Un client qui ne requiert aucune installation pourrait toutefois noter les applications et fichiers souvent utilisés et en conserver une copie locale en cache afin de permettre un certain niveau d'autonomie.





Processus serveurs pour l'infonuagique

Exercices pour le Module 3

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Quand changer de noeud

La compagnie Netflix ne possède pas d'infrastructure informatique et effectue toutes ses opérations informatiques avec les services Amazon EC2. Une stratégie qu'elle utilise est de démarrer une instance, de mesurer sa performance et de la conserver si elle est au-dessus d'un certain seuil, ou au contraire de la terminer si la performance n'est pas acceptable. Combien d'instances un fournisseur de services infonuagiques devrait-il ordonner sur un même noeud physique : 4, 8, 10...? Est-ce que ce nombre doit plutôt dépendre de la mémoire ou du nombre de cœurs du noeud, du comportement des instances? Quelle est la conséquence de mettre trop ou trop peu d'instances sur un même noeud?

Quand changer de noeud

Avec trop d'instances, une ou des ressources (e.g., CPU, disque, réseau) seront saturées et deviendront un goulot d'étranglement. En conséquence, tout s'en trouvera considérablement ralenti inutilement, et les clients seront mécontents. Avec trop peu d'instances, aucune des ressources ne sera pleinement utilisée et il y aura de la capacité excédentaire inutilisée et donc gaspillée. L'idéal est de grouper sur un même nœud des instances complémentaires qui n'utilisent pas toutes les mêmes ressources en même temps. Ce peut être plusieurs instances avec un taux d'utilisation faible et des périodes de pointes différentes, une instance gourmande en CPU avec une autre limitée par les entrées-sorties... Il est possible en première approximation de prendre un chiffre comme 6 ou 8 pour le nombre d'instances, en fonction de la charge moyenne d'une instance, et de la mémoire et du nombre de cœurs disponibles sur un nœud. Sur un service comme Amazon, les clients choisissent un type d'instance, ce qui donne une bonne idée quant à la charge attendue par l'instance.

Empiler Kubernetes et OpenStack

Peut-on rouler Kubernetes par-dessus OpenStack? OpenStack par-dessus OpenStack? Qu'en est-il de OpenStack par-dessus Kubernetes et Kubernetes par-dessus Kubernetes? Expliquez pour chaque cas comment cela peut ou non se faire, et commentez sur l'opportunité et l'efficacité d'une telle configuration.

Empiler Kubernetes et OpenStack

Kubernetes par-dessus OpenStack est possiblement la configuration la plus courante. Le fournisseur infonuagique offre des machines virtuelles via OpenStack et le client utilise cela pour un déploiement Kubernetes. Il n'y a qu'un seul niveau de virtualisation, ce qui est supporté par matériel et la performance est adéquate. L'inverse est aussi fréquent. OpenStack est assez compliqué à déployer et une configuration au-dessus de Kubernetes permet de simplifier grandement son déploiement. Un fournisseur infonuagique pourrait facilement déployer OpenStack par-dessus Kubernetes pour offrir des machines virtuelles à ses clients. Encore là, il n'y a qu'un seul niveau de virtualisation et la performance est bonne. Le client peut encore ici rouler Kubernetes par-dessus OpenStack (qui est par-dessus Kubernetes) sans problème et avec une bonne performance.



Empiler Kubernetes et OpenStack

Rouler Kubernetes par-dessus Kubernetes est moins fréquent mais est possible avec une bonne performance. La difficulté est que certaines opérations pour configurer les conteneurs de Kubernetes requièrent les privilèges d'administrateur. Les conteneurs au premier niveau devront donc être des conteneurs avec des privilèges spéciaux. OpenStack par-dessus OpenStack, avec la virtualisation à chaque niveau, serait possible mais peu performant. Par contre, il est possible d'utiliser OpenStack en mode *bare metal* pour orchestrer le déploiement de OpenStack par-dessus des noeuds physiques commandés par des modules de gestion de noeud (e.g., IPMI ou AMT). Tout comme OpenStack par-dessus Kubernetes, ceci est un moyen d'automatiser le déploiement de OpenStack.



Migration de machines virtuelles

Une machine virtuelle dans KVM contient 4GiO de mémoire organisée en pages de 4KiO. On désire migrer cette machine virtuelle vers une autre machine physique localisée sur le même réseau local et connectée avec un réseau de 100Mb/s soit 2000 pages/seconde. Par contre, on veut minimiser le temps où la machine virtuelle est interrompue pour la migration et il faut donc copier toutes les pages sans suspendre la machine virtuelle en notant les pages qui sont modifiées après avoir été copiées. L'exécution de la machine virtuelle modifie ses pages en mémoire au rythme de 100 pages différentes par seconde. On veut que le temps de suspension, pour copier les dernières pages modifiées, soit inférieur à 1 seconde. Combien de temps durera la migration au total? Combien d'itérations de copies seront requises? Quel sera le temps d'interruption?

Migration de machines virtuelles

La machine virtuelle contient $1\text{Mi pages} = 1048576$, ce qui demande $1048576 \text{ pages} / 2000 \text{ pages/s} = 524.288\text{s}$. Pendant ce temps, 52428 pages pourraient être modifiées, ce qui demande 26.21s secondes au deuxième tour. Pendant ce temps, 2621 pages seraient modifiées, demandant 1.31 secondes pour le transfert du troisième tour. Rendu là, on suspend la machine virtuelle et il y aura 131 pages à transférer en .06 secondes pour finaliser le transfert avant de reprendre l'exécution de la machine virtuelle sur la nouvelle machine physique. Il faudra donc 4 tours de copies, un temps total de 551.86s mais une interruption de seulement .06s.





Communication dans les systèmes répartis

Exercices pour le Module 4

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Attaque concertée

Les bleus sont divisés en deux camps de part et d'autre des verts et communiquent par messager. S'ils attaquent ensemble, la victoire est à eux. Chacun n'attaquera que s'il a l'assurance d'un accord mutuel confirmé. Quel protocole peuvent-ils utiliser pour s'assurer de manière absolue que l'autre accepte d'attaquer avec eux?

Attaque concertée

Les bleus sont divisés en deux camps de part et d'autre des verts et communiquent par messager. S'ils attaquent ensemble, la victoire est à eux. Chacun n'attaquera que s'il a l'assurance d'un accord mutuel confirmé. Quel protocole peuvent-ils utiliser pour s'assurer de manière absolue que l'autre accepte d'attaquer avec eux?

Ce n'est pas possible!



Ordre des mises à jour

Donnez un exemple pour lequel l'ordre d'un message de groupe (multicast) de deux clients n'est pas important? Un ou c'est important?



Ordre des mises à jour

Donnez un exemple pour lequel l'ordre d'un message de groupe (multicast) de deux clients n'est pas important? Un ou c'est important?

L'ordre n'est pas important pour une requête où un seul ordinateur peut envoyer la réponse (ou lorsque la valeur demandée ne change pas et peut venir de n'importe quel ordinateur). Par contre, pour des mises à jour de serveurs répliqués, il est important que tous les serveurs voient les mises à jour dans le même ordre lorsqu'elles s'intersectent.



Tempête de messages

Un réseau local comporte 250 ordinateurs et un routeur qui offre le service DHCP. Si au démarrage (après une panne d'électricité) chaque ordinateur demande à chaque autre s'il offre le service DHCP, combien de messages seront-ils envoyés à cette fin? Comment cela se compare-t-il à l'utilisation de diffusion générale?

Tempête de messages

Un réseau local comporte 250 ordinateurs et un routeur qui offre le service DHCP. Si au démarrage (après une panne d'électricité) chaque ordinateur demande à chaque autre s'il offre le service DHCP, combien de messages seront-ils envoyés à cette fin? Comment cela se compare-t-il à l'utilisation de diffusion générale?

Si chaque ordinateur envoie un message à chaque adresse, il peut envoyer 251 messages s'il sait l'intervalle ou sinon pourrait essayer les 254 adresses possibles. Le nombre total de messages serait de l'ordre de n au carré, soit $250 * 254 = 63500$. Autrement, chaque ordinateur pourrait n'envoyer qu'un seul message en diffusion générale pour 250 messages. Dans tous les cas, il y aura aussi 250 réponses.



Mise à l'échelle des messages

Comparez les différentes garanties pour les messages de groupe (non fiable, fiable, atomique, totalement ordonné, causalement ordonné) en termes de mise à l'échelle et de nombre de messages, en supposant n membres et que peu de messages sont perdus?

Mise à l'échelle des messages

Comparez les différentes garanties pour les messages de groupe (non fiable, fiable, atomique, totalement ordonnancé, causalement ordonnancé) en termes de mise à l'échelle et de nombre de messages, en supposant n membres et que peu de messages sont perdus?

- **Non fiable:** un message en multi-diffusion par envoi de groupe.
- **Fiable** (localiser un répondant): un message en multi-diffusion et une réponse.
- **Atomique:** deux messages en multi-diffusion et $n - 1$ accusés de réception.
- **Ordonnancement total:** 2 messages pour le numéro de séquence plus un message en multi-diffusion; risque de surcharge du serveur de séquence pour la mise à l'échelle.
- **Ordonnancement partiel:** un message en multi-diffusion.



Service de vote

Un service de vote permet de voter (candidat et numéro d'électeur), et d'obtenir le résultat (nom du candidat et nombre de votes). Donnez le prototype de ces fonctions. Identifiez les paramètres d'entrée et de sortie.



Service de vote

Un service de vote permet de voter (candidat et numéro d'électeur), et d'obtenir le résultat (nom du candidat et nombre de votes). Donnez le prototype de ces fonctions. Identifiez les paramètres d'entrée et de sortie.

```
void Vote(char *nom, int identificateur);  
void Resultat(char **nom, int *nombre);
```

La première fonction a deux paramètres d'entrée alors que la seconde a deux paramètres de sortie.



Sémantique des appels à distance

Les électeurs veulent être certains que leur vote soit pris en considération. Quelle sémantique peut les satisfaire?



Sémantique des appels à distance

Les électeurs veulent être certains que leur vote soit pris en considération. Quelle sémantique peut les satisfaire?

La sémantique peut-être ne suffit pas. Par contre au moins une fois est acceptable dans la mesure où le numéro d'électeur sert déjà à filtrer les répétitions.



Réseau partiellement asynchrone

Un réseau avec paquets qui peuvent être perdus est utilisé pour obtenir une sémantique au moins une fois. L'implantation peut ou non utiliser une hypothèse de délai maximal. Quel est l'effet sur l'implantation?

Réseau partiellement asynchrone

Un réseau avec paquets qui peuvent être perdus est utilisé pour obtenir une sémantique au moins une fois. L'implantation peut ou non utiliser une hypothèse de délai maximal. Quel est l'effet sur l'implantation?

En retransmettant passé un délai maximal, ceci élimine en théorie la possibilité de recevoir deux réponses, une tardive et une pour la retransmission.



Création du code pour les proxy

Comment peut-on générer un proxy pour une méthode comme
Vote dans un langage comme C++ qui ne supporte pas la
réflexion?



Création du code pour les proxy

Comment peut-on générer un proxy pour une méthode comme Vote dans un langage comme C++ qui ne supporte pas la réflexion?

Le code pour le proxy est généré à partir du IDL. Le proxy contient une référence réseau vers l'objet réel et une méthode Vote qui envoie au processus de la référence réseau: le numéro d'interface, le numéro de méthode, l'identificateur de l'objet, le nom, et le numéro d'électeur. La procédure d'envoi identifie le type de l'ordinateur (gros/petit boutien) et retransmet le tout jusqu'à obtention d'un accusé de réception.



Performance d'un appel à distance

Un client prend 5ms pour calculer les arguments d'une requête RPC, le serveur prend 10ms pour traiter la requête. Chaque envoi ou réception de paquet demande 0.5ms au système d'exploitation et le temps d'envoi d'un message sur le réseau est de 3ms. L'encodage et le décodage des arguments prend 0.5ms par message. Le client et le serveur roulent chacun sur un seul cœur. Quel est le temps pour effectuer deux requêtes si le client est séquentiel? Si le client est multi-fils mais le serveur demeure séquentiel?



Performance d'un appel à distance

calculer args + encoder + envoi + transmission + réception + décodage
+ traitement + encoder + envoi + transmission + réception + décodage
 $= 5 + .5 + .5 + 3 + .5 + .5 + 10 + .5 + .5 + 3 + .5 + .5 = 25\text{ms}$, soit
50ms pour deux requêtes séquentielles.

En multi-fils, le client commence sa seconde requête après l'envoi à 6ms.
Sa seconde requête est envoyée après un autre 6ms, soit à 12ms. Le serveur reçoit la première requête à 9ms, la traite et la renvoie à 21ms.
La seconde requête est déjà là et peut être traitée à ce moment. La seconde réponse est envoyée à 33ms, et parvient au client à 36ms, pour un total de **37ms**.



CORBA sans constructeur

Pourquoi CORBA et les systèmes similaires n'offrent-ils pas de constructeurs



CORBA sans constructeur

Pourquoi CORBA et les systèmes similaires n'offrent-ils pas de constructeurs

CORBA offre une interface et la notion d'instance n'est donc pas automatique. Le cas par défaut est une instance unique qui se confond avec l'interface. Au besoin il peut y avoir une méthode d'une interface qui exporte des objets réseau associés à une même interface.



Taille des messages

Vous devez programmer un système d'appel de procédure à distance pour vérifier la disponibilité et le prix de certains items:

```
void inventaire(in string id, out unsigned long quantity, out unsigned long price); où id est un identifiant typiquement de 10 caractères, la quantité environ 12 et le prix est un entier (en centièmes de dollars) aux environs de $100 (10000 cents). Vous avez le choix entre CORBA et gRPC.
```

Quels sont les avantages de chacun. Si on ne tient pas compte des entêtes (e.g. HTTP2) mais seulement de l'encodage des arguments, CDR versus protobuf, lequel sera plus compact?



Taille des messages

Vous devez programmer un système d'appel de procédure à distance pour vérifier la disponibilité et le prix de certains items: `void inventaire(in string id, out unsigned long quantity, out unsigned long price)`; où `id` est un identifiant typiquement de **10 caractères**, la quantité environ **12** et le prix est un entier (en centièmes de dollars) aux environs de **\$100 (10000 cents)**. Vous avez le choix entre CORBA et gRPC. Quels sont les avantages de chacun. Si on ne tient pas compte des entêtes (e.g. HTTP2) mais seulement de l'encodage des arguments, CDR versus protobuf, lequel sera plus compact?

CORBA et gRPC sont relativement semblables. Les deux permettent d'interfacer facilement des programmes écrits en différents langages et ont été développés avec une attention particulière pour l'optimisation. CORBA est mieux établi et bien normalisé. gRPC est plus moderne et permet une plus grande flexibilité au niveau de la compatibilité (vers l'avant et vers l'arrière) car les champs sont auto-identifiés.

CORBA prendra pour son encodage, outre le code identifiant la fonction, la longueur de la chaîne `id` arrondie au multiple de 4 octets supérieur plus 4 octets pour stocker la longueur, soit 16 octets pour un identifiant de 10 caractères, 4 octets pour la quantité et 4 octets pour le prix, pour un total de **24 octets**.

De son côté, protobuf demandera 1 octet pour identifier le champ et le type combinés pour `id`, 1 octet pour la longueur, 10 octets pour l'identifiant, 1 octet pour le champ et le type combinés et 1 octet varint pour la quantité, 1 octet pour le type et champ combinés et 2 octets pour le prix (10000 cents), pour un total de **17 octets**.

gRPC offre donc un compromis très intéressant de flexibilité (champs et types auto-identifiés), de simplicité d'encodage et de compacité.



Communication par objets répartis

Exercices pour le Module 5

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Décompte de référence en réparti

Pour un système de ramasse-miettes réparti, les opérations AddRef et RemoveRef sont utilisées par les clients pour indiquer lorsqu'ils commencent ou cessent d'utiliser un objet réseau. Discutez-en les implications.



Décompte de référence en réparti

Pour un système de ramasse-miettes réparti, les opérations AddRef et RemoveRef sont utilisées par les clients pour indiquer lorsqu'ils commencent ou cessent d'utiliser un objet réseau. Discutez-en les implications.

- Si le message AddRef est perdu, l'objet pourrait être enlevé même si un client l'utilise toujours.
- Lorsqu'un message RemoveRef est perdu, un objet est maintenu même si aucun client ne l'utilise.
- Si le client termine abruptement sans faire de RemoveRef, l'effet est similaire.
- Il faut donc utiliser des accusés de réception et avoir une détection de clients qui terminent sans crier gare.

Fin implicite ou explicite

Les opérations AddRef et RemoveRef sont utilisées par les clients d'objets répartis, pour indiquer lorsqu'ils commencent ou cessent d'utiliser un objet réseau. Ceci permet le fonctionnement d'un ramasse-miettes réparti. Le Remoting en C# utilise plutôt des échéances sur les références aux objets répartis. Un client doit renouveler une référence avant son échéance, sinon le serveur considère que le client ne l'utilise plus. Quel est l'intérêt d'un tel système?



Fin implicite ou explicite

- Dans le premier cas, la fin de l'utilisation d'une référence doit être explicite. Si le client tombe en panne, le serveur croit faussement que la référence est toujours utilisée, ce qui peut causer à long terme des fuites de mémoire qui s'accumulent.
- Dans le second cas, la fin d'utilisation d'une référence est implicite. Si le client tombe en panne, la référence sera libérée correctement, ce qui est beaucoup mieux. Il faut toutefois s'assurer de renouveler la référence avant son échéance pour continuer à l'utiliser.





Services de fichiers

Exercices pour le Module 6

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Index décentralisé

Dans les premiers systèmes poste-à-poste comme Napster, la mise à l'échelle souffrait de l'existence d'un index centralisé. Proposez d'autres solutions d'indexation qui peuvent se mettre à l'échelle.



Index décentralisé

Dans les premiers systèmes poste-à-poste comme Napster, la mise à l'échelle souffrait de l'existence d'un index centralisé. Proposez d'autres solutions d'indexation qui peuvent se mettre à l'échelle.

L'infrastructure d'indexation peut être mise à l'échelle en divisant l'index entre un grand nombre d'ordinateurs. On peut aussi miser sur l'indexation par Google qui fonctionne de cette manière. Il est aussi possible de répliquer l'information d'index entre de nombreux nœuds qui participent au réseau poste-à-poste en supposant que l'index n'est pas trop grand et que les mises à jour ne sont pas trop fréquentes. Par exemple, les nœuds adjacents peuvent se partager l'index pour éviter d'avoir chacun à le stocker et maintenir au complet.



Participants egoïstes

Les réseaux poste-à-poste dépendent en partie de la bonne collaboration de plusieurs utilisateurs/serveurs. Quelles peuvent être les conséquences de participants égoïstes ou malicieux, donnez des exemples?



Participants egoïstes

Les réseaux poste-à-poste dépendent en partie de la bonne collaboration de plusieurs utilisateurs/serveurs. Quelles peuvent être les conséquences de participants égoïstes ou malicieux, donnez des exemples?

L'utilisateur idéal accepte de servir en échange de se faire servir et offre du contenu valide. Un utilisateur égocentriste peut ne vouloir que recevoir et ne rien envoyer. La plupart des systèmes ajustent la priorité d'un client pour recevoir en fonction de ce qu'il a envoyé. Il serait possible de fausser ces chiffres. Un client malicieux pourrait vouloir nuire aux recherches de fichiers en offrant des fichiers semblables mais au contenu vide, corrompu, ou possiblement avec un virus. Dans certains cas, un client malicieux pourrait aussi offrir des fichiers protégés par les droits d'auteur et noter les clients qui en prennent une copie.

Conserver l'anonymat

Certains réseaux poste-à-poste offrent un service qui veut garantir l'anonymat. Est-ce possible?



Conserver l'anonymat

Certains réseaux poste-à-poste offrent un service qui veut garantir l'anonymat. Est-ce possible?

Si plusieurs nœuds intermédiaires sont utilisés pour établir une connexion et que les nœuds intermédiaires ne divulguent pas leur information, ceci est possible en théorie. Si un bon nombre de ces nœuds intermédiaires sont compromis (e.g. un sur deux), il peut être possible de déduire l'information manquante. De plus, si celui qui veut percer l'anonymat a accès au réseau en plusieurs points, il peut suivre le cheminement des paquets et même en envoyer lui-même comme sonde. Une technique pour empêcher en partie cela serait d'envoyer chaque paquet reçu à plus d'une adresse, de sorte qu'il serait plus difficile de distinguer le vrai destinataire en épantant le cheminement des paquets.



BitTorrent versus Napster

Vous désirez rendre disponible un fichier de 1000MiO à 100 clients externes. Chaque client a une bande passante de 1MiO/s dans les deux directions. On vous propose deux méthodes, un serveur de type Napster ou un autre de type BitTorrent. Supposez que vous avez un serveur (Napster ou BitTorrent) et initialement un client contenant le fichier à distribuer (votre ordinateur). Les échanges de contrôle avec les serveurs et même entre les clients sont négligeables. Combien de temps cela prendrait-il environ avec chacune des méthodes, si le transfert est fait efficacement, sachant que BitTorrent fonctionnera avec des morceaux de fichiers d'une longueur de 1MiO.



BitTorrent versus Napster

La copie à 1MiO/s prendra 1000s. Avec Napster il faut y aller par étapes: 1 client prend une copie, ensuite 2, puis 4, 8, 16, 32, 64 (total 127). Ceci prend donc 7 tours ou 7000s. Avec BitTorrent, le transfert de chaque bloc prend 1s. Après un 1er intervalle de 100s, chaque client peut avoir obtenu un morceau de 1MiO différent du client initial; ils peuvent aussi avoir commencé à propager ces morceaux, mais ceci n'est pas tenu en compte dans cet estimé conservateur. Ensuite, pendant chacun des 9 intervalles de 100s qui suivront, 2ème au 10ème, chaque client peut envoyer son morceau, reçu à l'intervalle précédent, aux 99 autres clients, recevoir les 99 morceaux des autres clients et recevoir un nouveau morceau du client initial. Rendu au 11ème intervalle, il ne reste plus de nouveaux morceaux à recevoir du client initial et on termine de propager le nouveau morceau, reçu par chaque client du client initial pendant le 10ème intervalle. La propagation complète peut donc se faire en 1100s.

Serveur de fichiers sans état

Pourquoi n'y a-t-il pas de open/close dans le service de fichiers réparti présenté en exemple?



Serveur de fichiers sans état

Pourquoi n'y a-t-il pas de open/close dans le service de fichiers réparti présenté en exemple?

Ces fonctions impliquent un état conservé dans le serveur, ce qu'un service réparti veut éviter pour continuer à opérer de manière transparente même si le serveur se réinitialise intempestivement.



Identificateurs de fichiers uniques

Pourquoi avoir des identificateurs de fichiers uniques à travers le réseau? Comment les produire?



Identificateurs de fichiers uniques

Pourquoi avoir des identificateurs de fichiers uniques à travers le réseau? Comment les produire?

Cela permet de faire référence à un fichier sans avoir à y ajouter un contexte (serveur). Un tel identificateur unique peut être obtenu avec le numéro IP de la machine de création, concaténé à un numéro unique (compteur séquentiel) sur cet ordinateur.



Transparence incomplète

En quoi le comportement local peut-il différer du comportement avec NFS pour l'accès concurrent à un fichier (POSIX copie unique)?



Transparence incomplète

En quoi le comportement local peut-il différer du comportement avec NFS pour l'accès concurrent à un fichier (POSIX copie unique)?

Lorsqu'un programme modifie un fichier en local, les autres programmes voient immédiatement la modification. Par NFS, un programme sur un autre client pourrait utiliser une ancienne version en cache jusqu'à 3 secondes après l'écriture du premier programme.



Le client NFS

Quelle information le module client NFS doit-il conserver pour les processus qui l'utilisent?



Le client NFS

Quelle information le module client NFS doit-il conserver pour les processus qui l'utilisent?

Pour chaque fichier ouvert, le client NFS doit conserver le vfs inode, la position courante dans le fichier, et l'identificateur NFS du fichier.



Monter en dur avec NFS

Quelle est l'utilité de monter en dur par NFS?



Monter en dur avec NFS

Quelle est l'utilité de monter en dur par NFS?

Plusieurs programmes ne vérifient pas les erreurs sur les opérations comme read/write. Un éditeur de texte pourrait essayer de sauver un fichier et présumer avoir réussi alors que ce n'est pas le cas. Dans un tel contexte, il est mieux que l'opération bloque plutôt que de retourner un code d'erreur qui serait ignoré.



NFS automount

Comment le serveur automount permet-il d'augmenter la fiabilité et la performance de NFS?

NFS automount

Comment le serveur automount permet-il d'augmenter la fiabilité et la performance de NFS?

Il est possible de spécifier plusieurs serveurs pour un sous-arbre sous la responsabilité de automount. Il prendra le serveur qui répond le plus rapidement, ce qui contourne les serveurs défectueux ou trop lents. Ceci se limite normalement aux sections de l'arbre qui ne changent pas, sont accédées en lecture seulement, et sont donc répliquées facilement.



Messages perdus avec AFS

Comment AFS tient-il compte des messages de modifications qui seraient perdus?



Messages perdus avec AFS

Comment AFS tient-il compte des messages de modifications qui seraient perdus?

Lors d'une ouverture de fichier, si aucun message n'est arrivé du serveur depuis quelques minutes, le client se réenregistre pour recevoir les messages de modifications. Il ne sert à rien de revérifier alors que le fichier n'est pas utilisé.



AFS plus performant

Pourquoi AFS peut-il supporter plus de clients que NFS?



AFS plus performant

Pourquoi AFS peut-il supporter plus de clients que NFS?

Avec AFS, les messages de modifications évitent les très nombreuses demandes de revalidation de NFS. De plus, l'ouverture d'un fichier fait en même temps sa lecture du serveur, et la fermeture fait au besoin son écriture sur le serveur. Cela réduit le nombre total d'appels au serveur.





Service de répertoire de noms

Exercices pour le Module 7

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Le problème des alias

Quels sont les problèmes associés aux alias dans un service de noms?



Le problème des alias

Quels sont les problèmes associés aux alias dans un service de noms?

Les alias sont utiles lorsque plusieurs noms sont applicables à une ressource. Par contre, ils peuvent prêter à confusion puisque deux noms différents mènent à la même chose. Les alias au niveau des répertoires peuvent conduire à des cycles qui peuvent être des écueils pour les outils de navigation dans l'espace de noms.



Utilité d'une cache négative

Quel est le problème des noms inconnus dans un système par envoi à tous?



Utilité d'une cache négative

Quel est le problème des noms inconnus dans un système par envoi à tous?

Un système par envoi à tous permet à plusieurs serveurs de collaborer facilement, pour des fins de redondance ou de division de l'espace de noms. Cependant, lorsqu'un nom inconnu est demandé, aucun serveur ne répond et le client ne sait pas si sa requête s'est perdue ou si le nom est inconnu des serveurs. Certains systèmes incluent des entrées négatives, l'information qu'un certain nom n'existe pas.



Robustesse par les caches

Comment les caches peuvent-elles augmenter la disponibilité du service de noms?



Robustesse par les caches

Comment les caches peuvent-elles augmenter la disponibilité du service de noms?

L'idéal est d'obtenir la précision et la simplicité d'un système centralisé et la robustesse et performance d'un système réparti. Les caches permettent d'obtenir un excellent compromis. Les caches ne requièrent aucune configuration, elles ne font que mémoriser pour leur période de validité (TTL) les informations qu'elles voient passer; ce faisant elles peuvent servir une grande proportion des requêtes.



Le point final

Pourquoi ne pas mettre de point à la fin des noms d'ordinateurs
(e.g. www.polymtl.ca. qui est comme /ca/polymtl/www)?



Le point final

Pourquoi ne pas mettre de point à la fin des noms d'ordinateurs
(e.g. www.polymtl.ca. qui est comme /ca/polymtl/www)?

Les noms relatifs ne sont pas supportés dans les DNS, le seul cas particulier est le nom de la dernière composante seul. Dans ce cas, il n'y a pas d'ambiguïté puisqu'aucun nom absolu n'a moins de deux composantes.



Sauver un niveau

Pourquoi les serveurs à la racine (e.g. .com, .org, .edu...) contiennent-ils l'information pour deux niveaux?



Sauver un niveau

Pourquoi les serveurs à la racine (e.g. .com, .org, .edu...) contiennent-ils l'information pour deux niveaux?

Les noms au premier niveau sont très peu nombreux (.com, .edu...) et ne correspondent pas à des entités différentes. Ils peuvent donc être servis ensemble et inclure le second niveau. Un serveur séparé s'occupe cependant des noms correspondant à chaque pays (e.g. serveur pour .ca).

Itération et récursion

Quels sont les avantages respectifs des accès récursifs versus itératifs pour la résolution de noms?



Itération et récursion

Quels sont les avantages respectifs des accès récursifs versus itératifs pour la résolution de noms?

L'accès récursif est simple et permet au serveur de voir la réponse et de la conserver en cache. Par contre, s'il bloque pendant l'accès récursif, la performance en souffrira pour les accès concurrents. Lors de l'accès itératif, le client est référé à un autre serveur et le serveur peut facilement passer à la prochaine requête.



Réponses multiples

Quand un serveur de noms fournit-il plusieurs réponses à une requête?



Réponses multiples

Quand un serveur de noms fournit-il plusieurs réponses à une requête?

Un domaine peut avoir plusieurs serveurs de noms ou de courrier. Ils sont tous retournés lors d'une requête pour ces services.



Sécurité

Quelles mesures de sécurité doivent entourer un service de nom?



Sécurité

Quelles mesures de sécurité doivent entourer un service de nom?

La modification/ajout des entrée ne doit être possible que par les administrateurs, sauf pour certains champs (adresse, mot de passe) qui peuvent être réservés à l'utilisateur.

La délégation de portions de l'espace de nom doit être possible, et la consultation de certaines information peut être restreinte. Il faut bien sûr que l'usager puisse être certain de parler au serveur de nom et que le serveur de nom puisse aussi authentifier l'usager.





Services de temps et de coordination

Exercices pour le Module 8

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Services de temps et de coordination

① Services de temps

② Coordination



Ralentir le temps

Pourquoi est-il déconseillé de remonter dans le temps? Comment se défaire d'une avance de 4 secondes en 8 secondes?



Ralentir le temps

Pourquoi est-il déconseillé de remonter dans le temps? Comment se défaire d'une avance de 4 secondes en 8 secondes?

Plusieurs applications comme `make` se basent sur l'hypothèse que le temps avance toujours.

Soit S la bonne heure et S_1 l'heure de l'ordinateur.

- Au moment présent, $S = S_1 - 4$.
- Dans 8 secondes, on veut que $S + 8 = S_1 + c \times 8$.
- Donc, $S_1 - 4 + 8 = S_1 + 4 = S_1 + c \times 8$, donc $4 = c \times 8$, et $c = 0.5$.
- Il faut donc faire avancer le temps à la moitié de sa vitesse normale pendant les 8 prochaines secondes.

Ralentir le temps (2)

Comment se défaire d'une avance de 12 secondes en 16 secondes?



Ralentir le temps (2)

Comment se défaire d'une avance de 12 secondes en 16 secondes?

Soit H_{real} la bonne heure et H_{ordi} l'heure de l'ordinateur.

Soit c le facteur que l'on veut appliquer à chaque seconde qui passe pour pouvoir synchroniser l'heure de l'ordinateur H_{ordi} à l'heure réelle H_{real} .

- Au moment présent, $H_{\text{real}} = H_{\text{ordi}} - 12$.
- Dans 16 secondes, on veut que $H_{\text{real}} + 16 = H_{\text{ordi}} + c \times 16$.
- Donc, $H_{\text{ordi}} - 12 + 16 = H_{\text{ordi}} + 4 = H_{\text{ordi}} + c \times 16$, donc $4 = c \times 16$, et $c = \frac{4}{16} = \frac{1}{4} = 0.25$.
- Il faut donc faire avancer le temps au quart de sa vitesse normale pendant les 16 prochaines secondes (il se passera 1 seconde pour l'ordi en 4 seconde réelles, et ce pendant 16 secondes réelles).

Vous avez l'heure?

Trois messages pour connaître le temps sont envoyés et donnent: 10:54:23.674 en 22ms aller-retour, 10:54:25.450 en 25ms, et 10:54:28.342 en 20ms. Quel heure doit-on choisir pour ajuster l'horloge? Si on sait que le délai minimal de transmission est de 8ms dans chaque direction, que cela change-t-il?



Vous avez l'heure?

Trois messages pour connaître le temps sont envoyés et donnent: 10:54:23.674 en 22ms aller-retour, 10:54:25.450 en 25ms, et 10:54:28.342 en 20ms. Quel heure doit-on choisir pour ajuster l'horloge? Si on sait que le délai minimal de transmission est de 8ms dans chaque direction, que cela change-t-il?

On retient le message avec le plus petit temps d'aller-retour.

- Si le délai vers le serveur est 0, le temps reçu date de 20ms et l'heure devrait être de 10:54:28.362.
- Si le délai pour le retour est de 0, l'heure devrait être 10:54:28:342 telle que reçue.
- En supposant que le délai est symétrique, l'heure sera plutôt mise à $10:54:28.342 + 20\text{ms}/2 = 10:54:28.352$, avec une précision de +/- 10ms.

Si on sait que le délai est d'au moins 8ms dans chaque direction, cela enlève 8ms à l'imprécision qui devient +/- 2ms.

Vous avez l'heure? (2)

Trois messages pour connaître le temps sont envoyés et donnent:
12:08:45.134 en 27ms aller-retour, 12:08:46.144 en 23ms, et
12:08:48.546 en 25ms. Quel heure doit-on choisir pour ajuster
l'horloge? Si on sait que le délai minimal de transmission est de
10ms dans chaque direction, que cela change-t-il?



Vous avez l'heure? (2)

Trois messages pour connaître le temps sont envoyés et donnent: 12:08:45.134 en 27ms aller-retour, 12:08:46.144 en 23ms, et 12:08:48.546 en 25ms. Quel heure doit-on choisir pour ajuster l'horloge? Si on sait que le délai minimal de transmission est de 10ms dans chaque direction, que cela change-t-il?

On retient le message avec le plus petit temps d'aller-retour.

- Si le délai vers le serveur est 0, le temps reçu date de 23ms et l'heure devrait être de 12:08:46.167.
- Si le délai pour le retour est de 0, l'heure devrait être 12:08:46.144 telle que reçue.
- Si le délai est symétrique, l'heure sera mise à $12:08:46.144 + 23ms/2 = 12:08:46.1555$, avec une précision de +/- 11.5ms.

Si le délai est d'au moins 10ms dans chaque direction, cela enlève 10ms à l'imprécision qui devient +/- 1.5ms.

Calcul de décalage

Un serveur A échange des messages de temps avec un serveur B. B reçoit 16:34:13.430 de A à 16:34:23.480, et A reçoit 16:34:25.700 à 16:34:15.725. Quelles sont la différence et l'imprécision?



Calcul de décalage

Un serveur A échange des messages de temps avec un serveur B. B reçoit 16:34:13.430 de A à 16:34:23.480, et A reçoit 16:34:25.700 à 16:34:15.725. Quelles sont la différence et l'imprécision?

- $a = 23.480 - 13.430 = 10.05$
- $b = 25.7 - 15.725 = 9.975$
- Le décalage est donc de:

$$\begin{aligned}\frac{a+b}{2} \pm \frac{a-b}{2} &= \frac{10.05+9.975}{2} \pm \frac{10.05-9.975}{2} \\&= \frac{20.025}{2} \pm \frac{0.075}{2} \\&= 10.0125 \pm .0375 \\&\approx 10.013 \pm .038\end{aligned}$$



Le choix d'un serveur de temps

Quelles sont les considérations importantes lors du choix d'un serveur NTP?



Le choix d'un serveur de temps

Quelles sont les considérations importantes lors du choix d'un serveur NTP?

- Il faut un serveur précis, et à une courte distance sur le réseau pour minimiser l'imprécision apportée par le délai de transmission.
- Par contre, il faut aussi faire attention à ne pas être trop vulnérable. Si le serveur peut ne plus être disponible ou être compromis, il est bon d'avoir des moyens de vérification et de compensation: horloge locale, autres serveurs . . .



Ajuster l'horloge

Comment peut-on améliorer la précision du temps si à chaque resynchronisation l'ajustement est sensiblement le même?



Ajuster l'horloge

Comment peut-on améliorer la précision du temps si à chaque resynchronisation l'ajustement est sensiblement le même?

En mesurant le taux moyen d'ajustement requis (e.g. 2 secondes de retard par heure), il est possible d'ajouter un petit incrément au temps continuellement pour compenser un biais systématique dans l'horloge locale. Ceci peut augmenter sensiblement la précision résultante.



Vérifier un état réparti

Un ensemble de processus $p_0, p_1, p_2\dots$ contiennent chacun une variable v , ($v_0, v_1, v_2\dots$). Peut-on déterminer si à un moment donné ces variables sont toutes égales, si les processus sont sur un seul processeur? Sur plusieurs ordinateurs?

Vérifier un état réparti

Un ensemble de processus $p_0, p_1, p_2\dots$ contiennent chacun une variable v , ($v_0, v_1, v_2\dots$). Peut-on déterminer si à un moment donné ces variables sont toutes égales, si les processus sont sur un seul processeur? Sur plusieurs ordinateurs?

Sur un seul ordinateur, il est normalement possible d'avoir un verrou pour aller chercher le temps à chaque changement de la variable et de mémoriser les changements dans chaque processus. Il suffit alors de vérifier lorsque le temps progresse la valeur qu'avait chaque variable.

Dans un système réparti le tout est plus difficile. On pourrait avoir des changements avec notification synchrone à un processus de surveillance. Une notification asynchrone est beaucoup plus efficace mais il faudrait alors avoir des bornes sur les délais de transmission et possiblement sur les horloges internes pour avoir une réponse qui pourrait aller de oui, à peut-être, à non. D'autres combinaisons seraient possibles, par exemple avec l'envoi de messages entre les processus et des relations causales correspondantes.

Services de temps et de coordination

1 Services de temps

2 Coordination



Être ou ne pas être en panne

Peut-on construire un détecteur de pannes fiable à partir d'un canal de communication non fiable?

Être ou ne pas être en panne

Peut-on construire un détecteur de pannes fiable à partir d'un canal de communication non fiable?

- Pour avoir un détecteur fiable, il faut un système synchrone et, normalement, un canal fiable.
- Ainsi, si aucune réponse ne parvient après un délai donné, il y a nécessairement une panne.
- Si le canal n'est pas fiable, il faudrait avoir une garantie du genre au moins un message sur 100 est transmis correctement pour être capable d'avoir une certitude qu'une panne a lieu.



Serveur central mais ordre causal

Décrivez une situation où un serveur central d'exclusion mutuelle ne traite pas les requêtes en ordre causal.

Serveur central mais ordre causal

Décrivez une situation où un serveur central d'exclusion mutuelle ne traite pas les requêtes en ordre causal.

- Un ordinateur A envoie une requête R_A , puis un message à un ordinateur B .
- B envoie alors une requête R_B .
- Il pourrait arriver que les délais différents ou retransmissions fassent que R_B arrive avant R_A au serveur.



Chef intérimaire permanent

Dans l'algorithme *Bully*, un serveur de haute priorité qui revient d'une panne déclenche une nouvelle élection. Est ce nécessaire?



Chef intérimaire permanent

Dans l'algorithme *Bully*, un serveur de haute priorité qui revient d'une panne déclenche une nouvelle élection. Est ce nécessaire?

- Si le numéro de priorité reflète la qualité de l'ordinateur (performance), il se peut que cela soit souhaitable.
- Autrement, il y a un coût inutile à faire une élection et transférer la coordination à un nouveau serveur.
- L'algorithme peut facilement être modifié pour que le serveur qui revient vérifie si un serveur de plus haute ou de plus basse priorité est actif et le laisse continuer.



Exclusion mutuelle

Entre l'exclusion mutuelle par serveur central, celle en anneau et celle par envoi à tous, laquelle est préférable? Pourquoi?

Exclusion mutuelle

Entre l'exclusion mutuelle par serveur central, celle en anneau et celle par envoi à tous, laquelle est préférable? Pourquoi?

- L'exclusion mutuelle par serveur central ne demande que 2 messages pour acquérir un verrou et n'est pas sensible aux pannes des clients qui ne possèdent pas de verrou. C'est la plus intéressante.
- L'exclusion mutuelle par envoi à tous est très sensible à la panne de chaque client et demande $2(n - 1)$ messages pour acquérir un verrou.
- L'exclusion mutuelle en anneau n'offre pas les mêmes propriétés (premier arrivé, premier servi) que les méthodes précédentes, est sensible à la panne de chaque client, et demande des messages même en l'absence de demande de verrou.



Cohérence et réPLICATION pour les données réparties

Exercices pour le Module 9

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Cohérence vue des clients

Un système A utilise des serveurs répliqués, et les clients peuvent lire ou écrire sur chaque serveur. Sur le système B, les clients doivent écrire sur un serveur central mais peuvent lire de serveurs secondaires. Un serveur qui reçoit une écriture la propage aux autres serveurs. Les messages des clients sont numérotés et propagés dans l'ordre. Dans chaque cas, dites s'il s'agit de cohérence stricte, cohérence séquentielle ou cohérence causale.



Cohérence vue des clients

Un système A utilise des serveurs répliqués, et les clients peuvent lire ou écrire sur chaque serveur. Sur le système B, les clients doivent écrire sur un serveur central mais peuvent lire de serveurs secondaires. Un serveur qui reçoit une écriture la propage aux autres serveurs. Les messages des clients sont numérotés et propagés dans l'ordre. Dans chaque cas, dites s'il s'agit de cohérence stricte, cohérence séquentielle ou cohérence causale.

Le système A offre une cohérence causale. Les écritures d'un même client restent dans l'ordre mais les écritures de différents clients sur différents serveurs peuvent être propagées aux autres serveurs dans un ordre différent. Le système B offre une cohérence séquentielle, puisque tout est linéarisé à travers un serveur central.

Transactions concurrentes cohérentes

Un serveur gère plusieurs valeurs a_1, \dots, a_n . Ce serveur offre deux opérations à ses clients: $\text{value} = \text{Read}(i)$, $\text{Write}(i, \text{value})$. Les transactions T et U sont définies comme suit. Donnez deux sérialisations équivalentes de ces transactions?

T: $x=\text{Read}(j); y=\text{Read}(i); \text{Write}(j, 44); \text{Write}(i, 33);$

U: $x=\text{Read}(k); \text{Write}(i, 55); y=\text{Read}(j); \text{Write}(k, 66);$



Transactions concurrentes cohérentes

Si T procède entièrement avant U, le résultat final sera:

$$Tx = a_j^0; Ty = a_i^0; Ux = a_k^0; Uy = 44; a_i = 55; a_j = 44; a_k = 66;$$

L'entrelacement suivant des opérations produit le même résultat:

Transaction T	Transaction U
$x = \text{Read}(j)$	
$y = \text{Read}(i)$	$x = \text{Read}(k)$
$\text{Write}(j, 44)$	
$\text{Write}(i, 33)$	
	$\text{Write}(i, 55)$
	$y = \text{Read}(j)$
	$\text{Write}(k, 66)$

Si U procède avant T, le résultat final sera:

$$Tx = a_j^0; Ty = 55; Ux = a_k^0; Uy = a_j^0; a_i = 33; a_j = 44; a_k = 66;$$

L'entrelacement suivant des opérations produit le même résultat:

Transaction T	Transaction U
$x = \text{Read}(j)$	
	$x = \text{Read}(k)$
	$\text{Write}(i, 55)$
$y = \text{Read}(i)$	
	$y = \text{Read}(j)$
	$\text{Write}(k, 66)$
$\text{Write}(j, 44)$	
$\text{Write}(i, 33)$	

Les verrous

Montrez pourquoi lorsqu'une transaction a relâché un verrou elle ne doit plus en obtenir afin de permettre l'équivalence avec la sérialisation?



Les verrous

Montrez pourquoi lorsqu'une transaction a relâché un verrou elle ne doit plus en obtenir afin de permettre l'équivalence avec la sérialisation?

Soit deux transactions T et U:

- **T:** $x = \text{Read}(i); \text{Write}(j, 44);$
- **U:** $\text{Write}(i, 55); \text{Write}(j, 66);$

Avec T avant U, le résultat est:

- $x = a_i^0$
- $a_i = 55$
- $a_j = 66$

Si T commence avant U mais les verrous sont relâchés trop tôt, nous pourrions avoir:

Transaction T	Transaction U
Lock <i>i</i>	
$x = \text{Read}(i)$	
Unlock <i>i</i>	Lock <i>i</i>
	Write(<i>i</i> , 55)
	Lock <i>j</i>
	Write(<i>j</i> , 66)
	Commit
Lock <i>j</i>	Unlock <i>i, j</i>
Write(<i>j</i> , 44)	
Unlock <i>j</i>	
Commit	

Verrouillage en deux phases

Soit deux transactions T et U, lesquelles de ces sérialisations sont valides? Sont possibles avec un verrouillage en deux phases?

```
T: x = read(i); write(j,44);
U: write(i,55); write(j,66);
```

- a T x=Read(i); U write(i,55); T write(j,44); U write(j,66);
- b U write(i,55); U write(j,66); T x=Read(i); T write(j,44);
- c T x=Read(i); T write(j,44); U write(i,55); U write(j,66);
- d U write(i,55); T x=Read(i); U write(j,66); T write(j,44);



Verrouillage en deux phases

Soit deux transactions T et U, lesquelles de ces sérialisations sont valides? Sont possibles avec un verrouillage en deux phases?

```
T: x = read(i); write(j,44);
U: write(i,55); write(j,66);
```

- a** T x=Read(i); U write(i,55); T write(j,44); U write(j,66);
- b** U write(i,55); U write(j,66); T x=Read(i); T write(j,44);
- c** T x=Read(i); T write(j,44); U write(i,55); U write(j,66);
- d** U write(i,55); T x=Read(i); U write(j,66); T write(j,44);

Les 4 sérialisations proposées sont valides (*a* et *c* pour T avant U, *b* et *d* pour U avant T).

En *a* et *d*, le verrou sur *i* bloquerait la deuxième opération et ces sérialisations ne pourraient se produire.



Relâcher les verrous

Expliquez pourquoi il ne faut pas relâcher les verrous avant de se commettre, même après la fin des opérations.



Relâcher les verrous

Expliquez pourquoi il ne faut pas relâcher les verrous avant de se commettre, même après la fin des opérations.

Si une autre transaction utilise les valeurs modifiées et la première transaction échoue, l'autre transaction aura utilisé des valeurs incorrectes.
Il serait possible cependant de relâcher les verrous de lecture.



Validation avec contrôle optimiste

Une synchronisation optimiste est appliquée aux transactions T et U qui sont actives en même temps. Discutez ce qui arrive dans chacun des cas suivants:

```
T: x = read(i); write(j,44);
U: write(i,55); write(j,66);
```

- a T est prête en premier et la validation en reculant est utilisée?
- b En avançant?
- c U est prête en premier et la validation en reculant est utilisée?
- d En avançant?



Validation avec contrôle optimiste

a Reculant:

- Correct, Read(i) est vérifié avec les écritures de transactions concurrentes terminées (aucune).
- Pour U, aucune lecture n'est effectuée et il n'y a donc pas de conflit.

b Avançant:

- Write(j,44) est vérifié avec les lectures de transactions concurrentes actives (pas de lecture dans U).
- Lorsque U termine, les écritures ne posent pas problème car T est terminée.

c Reculant:

- Correct, U ne fait pas de lecture
- Read(i) est vérifié avec les écritures de transactions concurrentes terminées, un problème est détecté car la valeur a été modifiée par U; T doit être repris.

d Avançant:

- Write(i,55) est vérifié avec les lectures de transactions concurrentes actives, un problème est détecté car T veut lire i; la transaction U est annulée.
- Pour T, Write(j,44) est vérifié avec les lectures de transactions concurrentes actives, ce qui est correct.

Transactions imbriquées

Expliquez comment une transaction imbriquée s'assure que toutes les sous-transactions sont correctement commises ou annulées.



Transactions imbriquées

Expliquez comment une transaction imbriquée s'assure que toutes les sous-transactions sont correctement commises ou annulées.

Chaque sous-transaction maintient une liste des transactions prêtes à se commettre.

La transaction de premier niveau peut donc faire une recherche en profondeur des transactions prêtes pour la validation finale.



Journal de transactions

Les transactions T, U, et V sont définies comme suit. Décrivez le journal produit par cette application si un verrouillage en deux phases est utilisé et U acquiert i et j avant T? Décrivez comment ce journal peut être utilisé pour récupérer en cas de panne?

T: x = Read(i); Write(j,44);

U: Write(i,55); Write(j,66);

V: Write(k,77); Write(k,88);

Journal de transactions

Journal

```
P0: ...; P1: Write(i,55);
P2: Write(j,66);
P3: Prépare U(i:P1, j:P2), P0;
P4: Compléter U, P3;
P5: Write(j,44);
P6: Préparer T(j:P5), P4;
P7: Compléter T, P6;
P8: Write(k,88);
P9: Prépare V(k:P8), P7;
P10: Compléter V, P9;
```



Journal de transactions

Journal

```
P0: ...; P1: Write(i,55);
P2: Write(j,66);
P3: Prépare U(i:P1, j:P2), P0;
P4: Compléter U, P3;
P5: Write(j,44);
P6: Préparer T(j:P5), P4;
P7: Compléter T, P6;
P8: Write(k,88);
P9: Prépare V(k:P8), P7;
P10: Compléter V, P9;
```

- Pour récupérer, les variables a_1, \dots, a_n sont remplacées à leur valeur par défaut.
- Ensuite le journal est lu en reculant.
- Le récupérateur voit que V est complété et à P9 que V contient P8; il met donc a_k à 88.
- Il continue à reculer à P7, voit que T est complété et peut ensuite lire a_j à 44 en voyant P6 qui pointe à P5.
- La chaîne remonte ensuite à P4, où U est marqué complété; ceci fait reculer à P3 qui pointe vers une valeur de j qui est ignorée car la valeur la plus récente est déjà lue, et vers une valeur pour i ce qui permet de trouver que a_i devient 55.



Journal et validation en reculant

Les transactions T et U utilisent un contrôle optimiste de la concurrence basé sur la validation en reculant. Décrivez l'information écrite dans le journal si T débute en premier.

Transac. T	Transac. U
$x = \text{Read}(i)$	
	Write($i, 55$)
	Write($j, 66$)
Write($j, 44$)	
	Commit
Commit	



Journal et validation en reculant

Les transactions T et U utilisent un contrôle optimiste de la concurrence basé sur la validation en reculant. Décrivez l'information écrite dans le journal si T débute en premier.

Transac. T	Transac. U
$x = \text{Read}(i)$	
	Write($i, 55$)
	Write($j, 66$)
Write($j, 44$)	
	Commit
Commit	

Journal

P0: ...;
 P1: $i = 55$;
 P2: $j = 66$;
 P3: Prépare U($i:P1, j:P2$), P0;
 P4: Committre U, P3;
 P5: $j = 44$;
 P6: Prépare T($j:P5$), P4;
 P7: Committre T, P6;

- Lorsqu'une transaction est validée, le numéro de transaction apparaît dans le journal.
- U est validée car elle ne fait aucune lecture.
- Par contre, la transaction T est annulée car elle lit i qui est écrit par U.
- T est redémarrée et peut finalement compléter.



Verrous pour les transactions réparties

L'item a est répliqué sur A_x et A_y et l'item b sur B_m , et B_n .
Les transactions T et U sont définies ainsi:

T: Read(a); Write(b, 44);
U: Read(b); Write(a, 55);

- ① Montrez un ordonnancement de T et U, dans le contexte de verrous en deux phases pour les réplicats.
- ② Expliquez pourquoi les verrous ne suffisent pas à assurer la sérialisation.



Verrous pour les transactions réparties

Transac. T	Lock T	Transac. U	Lock U
$x = \text{Read}(A_x)$	lock A_x		
$\text{Write}(B_m, 44)$	lock B_m		
		$x = \text{Read}(B_m)$	wait B_m
$\text{Write}(B_n, 44)$	lock B_n		
Commit	unlock A_x, B_m, B_n	:	:
		$x = \text{Read}(B_m)$	lock B_m
		$\text{Write}(A_x, 55)$	lock A_x
		$\text{Write}(A_y, 55)$	lock A_y
		Commit	unlock B_m, A_x, A_y

- Supposons que B_m tombe en panne avant d'être verrouillé par U.
- U pourra procéder. La validation locale permet de vérifier si une copie qui n'a pas fonctionné n'a pas été recouvrée.
- Dans le cas de T, lorsque B_m est revenu et a déjà été lu, il découvre que le verrou détenu ne tient plus et la transaction doit être annulée.

Ordre local et global

Donnez un exemple d'ordonnancement entre deux transactions, T et U qui lisent et écrivent a et b, qui est cohérent sur chaque serveur mais incohérent globalement?

Ordre local et global

Donnez un exemple d'ordonnancement entre deux transactions, T et U qui lisent et écrivent a et b, qui est cohérent sur chaque serveur mais incohérent globalement?

Serveur X, T passe avant U, **correct localement**:

Transaction T	Transaction U
Read(a)	
Write(a)	
	Read(a)
	Write(a)

Serveur Y, U passe avant T, **correct localement**:

Transaction T	Transaction U
	Read(b)
	Write(b)
Read(b)	
Write(b)	

Globalement il y a un problème:

- la transaction U pourrait lire la valeur initiale de b, puis la valeur modifiée par T de a pour les combiner en une nouvelle valeur de b et de a.
- Autrement dit, T précède U qui précède T pour respecter l'ordre sur



Tolérance aux pannes

Exercices pour le Module 10

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Disponibilité

Trois ordinateurs offrent un service. Chaque ordinateur fait en moyenne une panne après 5 jours d'opération et cette panne prend 4 heures à réparer. Quelle est la disponibilité de ce système?



Disponibilité

Trois ordinateurs offrent un service. Chaque ordinateur fait en moyenne une panne après 5 jours d'opération et cette panne prend 4 heures à réparer. Quelle est la disponibilité de ce système?

La probabilité de ne pas être fonctionnel est de:

$$\frac{4\text{h}}{5 \times 24\text{h} + 4\text{h}} = 0.03$$

La probabilité d'avoir les trois ordinateurs simultanément en panne est de:

$$\binom{3}{3} \times (0.03)^3 \times (1 - 0.03)^0 = 1 \times (0.03)^3 \times 1 = 0.000027$$

La disponibilité est donc de $1 - 0.000027 = 0.999973$.



Serveurs DNS

Sur les ordinateurs Linux, le fichier `resolv.conf` permet de lister plusieurs serveurs de nom (DNS), qui seront interrogés dans l'ordre selon lequel ils apparaissent dans le fichier, jusqu'à ce qu'une réponse soit obtenue. Ceci offre donc une tolérance aux pannes de serveur DNS par redondance, lorsqu'une traduction de nom à adresse IP est requise. Si 5 serveurs sont listés dans le fichier de configuration, combien de serveurs en panne i) par omission ce mécanisme peut-il tolérer avant de cesser de fonctionner? Si les serveurs pouvaient fournir de mauvaises réponses selon deux scénarios, ii) aléatoires et iii) byzantines, comment pourriez-vous modifier ce mécanisme pour aussi tolérer des pannes de type ii)? De type iii)? Dans chaque cas, ii) et iii), combien de serveurs en panne, (arrêtés ou compromis), pourriez-vous tolérer tout en maintenant le service opérationnel?

Serveurs DNS

- Avec des pannes i) par omission, il suffit d'un seul serveur opérationnel pour fonctionner, soit jusqu'à 4 serveurs en panne qui peuvent être tolérés.
- Pour les pannes ii) aléatoires, il faudrait comparer les réponses et retenir celle qui revient le plus souvent. En supposant que les probabilités de deux mauvaises réponses aléatoires identiques sont minimes, on peut tolérer 3 pannes.
- Pour les pannes iii) byzantines, il faudrait comparer les réponses et retenir aussi celle qui apparaît le plus souvent. Dans un tel cas, jusqu'à 2 serveurs en panne pourraient être tolérés (3 bonnes réponses contre 2 mauvaises). En effet, avec 3 serveurs compromis, la mauvaise réponse pourrait l'emporter.



Disques en miroir

On vous demande de choisir entre deux configurations pour le prochain serveur de fichiers de votre entreprise. La première configuration consiste en un ordinateur avec 4 disques en miroir. Chaque disque contient l'ensemble des données et un seul disque suffit donc. Le second système est constitué de deux ordinateurs, qui sont deux serveurs redondants, chacun étant connecté à deux disques en miroir et un seul disque suffit donc pour un serveur. La probabilité qu'un ordinateur soit opérationnel (hormis les disques) est de 0.95. La probabilité qu'un disque soit opérationnel est de 0.85. Quelle est la probabilité que le service soit disponible, pour chacune des deux configurations?



Disques en miroir

- Première configuration:
 - Le système de disque est disponible sauf si les 4 sont indisponibles:
 $1 - (1 - .85)^4 = 0.99949375.$
 - Le service sera disponible si les disques et l'ordinateur le sont:
 $0.99949375 \times 0.95 = 0.949519063.$
- Seconde configuration:
 - Sur un serveur, les disques seront disponibles sauf si les 2 sont indisponibles: $1 - (1 - .85)^2 = 0.9775.$
 - Un serveur sera disponible si l'ordinateur et les disques le sont:
 $0.9775 \times 0.95 = 0.928625.$
 - Le service sera disponible sauf si les deux serveurs redondants sont indisponibles: $1 - (1 - 0.928625)^2 = 0.994905609.$
- La deuxième configuration est donc nettement mieux pour réduire le temps d'indisponibilité. Elle est toutefois légèrement plus coûteuse puisqu'elle utilise deux ordinateurs plutôt qu'un, tout en conservant le même nombre de disques.

Services de fichiers

Avec CODA, pourquoi les utilisateurs doivent-ils parfois intervenir manuellement?



Services de fichiers

Avec CODA, pourquoi les utilisateurs doivent-ils parfois intervenir manuellement?

- Le contrôle de concurrence est optimiste sur CODA et les opérations ne peuvent être annulées.
- Ainsi, si deux utilisateurs ont mis à jour en même temps leur copie d'un fichier, le second à transmettre la mise à jour au serveur doit décider comment réconcilier les deux versions, plutôt que d'écraser le travail de l'autre.



Paxos

On veut appliquer l'algorithme du consensus de Paxos à l'élection d'un serveur maître. Ce serveur maître doit être unique en tout temps afin d'assurer la cohérence du système (e.g. attribuer les sièges d'une salle de spectacle à des clients qui se connectent via un réseau externe). Si le serveur maître issu du consensus devient éventuellement non disponible, peut-on changer le consensus pour un nouveau maître? Comment peut-on distinguer entre un serveur maître inopérant et un serveur maître coupé du réseau? Comment empêcher un serveur maître coupé du réseau de continuer à se considérer maître (et vendre des sièges possiblement en double)? Donnez un exemple de deux propositions concurrentes où la seconde proposition commence alors que la première était presque terminée. Donnez un exemple de réseau de 5 serveurs participant au vote et partitionnés 2:1:2 et 2:3.

Paxos

Un consensus ne devrait normalement pas changer, le serveur maître peut cependant devoir changer au fil du temps. Pour concilier les deux, on peut avoir un consensus sur le serveur maître pour un intervalle donné. L'intervalle peut être dénoté par un rang (e.g. 14ème intervalle) ou par un temps de début. En cas de panne de ce serveur maître, un nouveau consensus peut s'établir pour un nouveau serveur maître pour le nouvel intervalle (e.g. 15ème intervalle) et tout le monde communique toujours avec le serveur maître du plus récent intervalle. Le serveur maître pourrait ne pas être en panne mais simplement coupé du réseau. On ne peut distinguer l'un de l'autre. Il peut alors être justifié d'élire un nouveau serveur maître si l'ancien n'est pas rejoignable. Si l'ancien serveur maître est en panne, il ne fait plus rien et ne cause pas de conflit.



Paxos

Si l'ancien serveur maître est simplement coupé du réseau et continue à se considérer comme maître, ce peut être problématique puisqu'il continue à se considérer en autorité pour effectuer des opérations critiques (attribution des sièges). Une solution que l'on retrouve souvent dans les configurations redondantes à deux serveurs (actif-passif) est que le nouveau serveur commande physiquement l'arrêt de l'alimentation électrique de l'ancien serveur pour prévenir ce genre de problème.

Paxos

Supposons 5 serveurs A, B, C, D, E. Au moment $n=1$, A se propose ($v=A$) comme serveur maître et obtient une promesse de A, B et C, ce qui constitue un quorum. Presqu'en même temps, au temps $n=2$, E se propose ($v=E$) comme serveur maître et obtient une promesse de D et E. Si A poursuit en demandant à D et E, il peut se faire ignorer ou se faire répondre qu'une proposition plus récente circule. En apprenant qu'une proposition plus récente circule, il peut abandonner. Autrement, il pensera que D et E sont en panne et, ayant déjà le quorum, A envoie un message demandant d'accepter $n=1$, $v=A$. Si ceci est effectué très rapidement, et que A reçoit la confirmation de A, B et C, la majorité est obtenue et le consensus atteint. Dans ce cas, E ne réussira pas à avoir un quorum de promesses et encore moins pour son acceptation.



Paxos

L'autre possibilité est que C reçoive la proposition de E avant la demande d'acceptation de A. Dans ce cas, il répond avec une promesse à la proposition de E, puisqu'elle est plus récente, et il ignorerá la demande d'acceptation de A. A ne pourra obtenir un quorum d'acceptation. De son côté, E recevra une promesse de C avec l'information que sa promesse la plus récente acceptée était pour $n=1, v=A$. Il reprendra la proposition à son compte et enverra une demande d'acceptation pour $n=2, v=A$ à tous et recevra la confirmation de C, D et E. Là encore, la majorité est obtenue et le consensus atteint.

Supposons les 5 mêmes serveurs A, B, C, D, E. Avec une partition du réseau 2:1:2, aucune partition ne regroupe une majorité de serveurs et il ne sera pas possible d'atteindre un consensus. Avec une partition 2:3, le consensus ne pourra être atteint que dans la partition de 3 serveurs, ce qui assure qu'un seul serveur maître sera élu, et non pas un serveur par partition.



Politique byzantine

Montrez qu'il est possible d'obtenir un consensus entre trois généraux byzantins dont un est fautif si les messages sont signés.



Politique byzantine

Montrez qu'il est possible d'obtenir un consensus entre trois généraux byzantins dont un est fautif si les messages sont signés.

Dans la mesure où les généraux qui agissent comme lieutenant peuvent vérifier la validité des ordres transmis, ils peuvent facilement s'entendre. Si le général en chef transmet des ordres contradictoires, les lieutenants peuvent s'en rendre compte. Si un lieutenant ment à propos des ordres reçus, il est tout de suite démasqué.





Informatique et développement durable

Exercices pour le Module 11

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

La notion de développement durable

Qu'est-ce que la notion de développement durable ajoute à l'économie traditionnelle?



La notion de développement durable

Qu'est-ce que la notion de développement durable ajoute à l'économie traditionnelle?

L'économie traditionnelle regarde les coûts monétaires associés à un projet et permet, par exemple, de calculer quel scénario serait le plus rentable de ce point de vue. Le développement durable répond aux besoins présents sans compromettre la capacité des générations futures à répondre à leurs propres besoins. Ceci ajoute à l'approche traditionnelle tout le calcul de l'impact sur l'environnement : la santé humaine, l'écologie, les changements climatiques et l'épuisement des ressources.



Les lois applicables

Quels sont les lois et codes applicables relatifs au développement durable pour un ingénieur au Québec?



Les lois applicables

Quels sont les lois et codes applicables relatifs au développement durable pour un ingénieur au Québec?

Le gouvernement fédéral et le gouvernement du Québec ont chacun une loi sur le développement durable. De plus, le code de déontologie de l'Ordre des Ingénieurs du Québec contient un paragraphe qui parle des obligations de l'ingénieur quant à préserver l'environnement et la santé. D'autres lois et principes du gouvernement du Québec touchent aussi au développement durable, comme le Bureau d'Audiences Publiques sur l'Environnement (BAPE), et les règlements sur la Responsabilité Elargie des Producteurs (REP).



Le rôle de l'ingénieur

Quel est le rôle de l'ingénieur en lien avec le développement durable?



Le rôle de l'ingénieur

Quel est le rôle de l'ingénieur en lien avec le développement durable?

L'ingénieur est un professionnel qui développe des projets. Il doit s'assurer de tenir compte du développement durable dans la conception, la réalisation, l'opération et le démantèlement de ses projets. Ceci est mandaté autant par les lois que par son code de déontologie. Ainsi, il s'assure de respecter ses obligations et de contribuer au développement harmonieux et durable de la société.



Le cycle de vie

Qu'est-ce que l'analyse du cycle de vie?



Le cycle de vie

Qu'est-ce que l'analyse du cycle de vie?

L'analyse du cycle de vie d'un projet ou d'un produit tient compte du cycle de vie complet, de la fabrication des intrants à la disposition finale des produits en fin de vie. Ceci permet d'évaluer et de comparer deux produits ou scénarios de projets en tenant compte de tous les aspects pertinents.



La dématérialisation

On parle de dématérialisation en raison de l'informatisation. Est-ce une bonne chose? Donnez des exemples de dématérialisation et discutez de leur impact sur le développement durable.



La dématérialisation

L'informatique permet de transmettre et de lire des documents sans avoir besoin d'imprimer sur papier, le fameux bureau sans papier! Ceci permet de réduire la consommation de papier (de la coupe de bois et la fabrication jusqu'à la mise aux rebuts). Ceci réduit aussi l'espace d'entreposage pour tous ces documents en papier. L'informatisation permet aussi des rencontres par vidéo-conférence plutôt que de voyager. Là encore, ceci peut économiser beaucoup de ressources (avion, auto, hotel). Le coût pour ces solutions de remplacement est l'utilisation d'équipement électronique (ordinateurs, serveurs, réseau de communication). Toutefois, dans la mesure où presque tous avaient déjà un ordinateur avant de commencer à éliminer le papier ou faire de la vidéo-conférence, le coût de cet équipement ne peut qu'en très petite partie être attribué à la dématérialisation. Conséquemment, ceci semble largement bénéfique.

Informatisation des procédés

En quoi est-ce que l'informatisation des procédés peut être bénéfique pour le développement durable?



Informatisation des procédés

En quoi est-ce que l'informatisation des procédés peut être bénéfique pour le développement durable?

Lorsqu'un procédé est commandé par ordinateur, il y a un coût additionnel en électronique mais aussi un bénéfice au niveau de l'efficacité. Un ventilateur à vitesse variable permet d'augmenter l'efficacité énergétique d'un système de ventilation et climatisation. De la même manière, un système intelligent d'épandage d'engrais permet d'obtenir de meilleurs rendements agricoles en utilisant moins d'engrais. Heureusement, la taille (et la quantité de matière première) et le coût des modules de commande diminue sans cesse, si bien que ces systèmes sont habituellement largement bénéfiques.



Optimisation et développement durable

Donnez quelques exemples où les logiciels d'optimisation ont un impact positif sur le développement durable?



Optimisation et développement durable

Donnez quelques exemples où les logiciels d'optimisation ont un impact positif sur le développement durable?

Les logiciels de conception, de simulation et d'optimisation de pièces pour les voitures et les avions permettent de produire des véhicules plus légers, plus aérodynamiques et plus efficaces, réduisant leur empreinte énergétique. Les logiciels d'optimisation du transport permettent de la même manière de transporter plus de personnes et de marchandise, plus rapidement, avec les mêmes routes et équipements, augmentant l'efficacité et réduisant les besoins en équipement et en énergie.



Les centres de données

Vous participez à la conception d'un nouveau centre de données. Quels sont en général les 2 intrants, en opération ou dans les autres phases, qui ont le plus grand impact sur les différents aspects évalués dans l'analyse du cycle de vie?



Les centres de données

Vous participez à la conception d'un nouveau centre de données. Quels sont en général les 2 intrants, en opération ou dans les autres phases, qui ont le plus grand impact sur les différents aspects évalués dans l'analyse du cycle de vie?

L'impact le plus important vient souvent de la consommation d'électricité, selon sa provenance. Le second impact le plus important vient de la fabrication des ordinateurs et autres appareils connexes (réseautique, alimentation de secours...).



Achats en ligne

Au moment d'acheter une nouvelle pièce pour un de vos vélos, vous hésitez entre la commander en ligne et l'acheter dans un commerce. Quelle solution est la plus intéressante d'un point de vue de développement durable?

Achats en ligne

Au moment d'acheter une nouvelle pièce pour un de vos vélos, vous hésitez entre la commander en ligne et l'acheter dans un commerce. Quelle solution est la plus intéressante d'un point de vue de développement durable?

Les données fournies ne sont pas suffisantes, la solution n'est pas évidente. Par exemple, si le commerce commande la pièce chez son fournisseur, il se peut que le coût en emballage et en transport soit le même que si vous l'aviez achetée en ligne. Il faut ensuite ajouter votre déplacement jusqu'au commerce. Dans ce cas, l'achat en ligne est probablement mieux. Par contre, si le commerce est près de votre domicile et il reçoit les pièces en grande quantité, sans beaucoup d'emballage additionnel (e.g. une boîte pour 50 unités ou des contenants réutilisables), l'achat local peut être plus intéressant.

Engins de recherche

Un membre de votre entourage prétend que chaque recherche sur le Web (e.g., sur Google) consomme beaucoup d'électricité et a un impact très négatif. Vérification faite, vous trouvez que chaque recherche consomme 0.3 watt-heure et qu'un utilisateur moyen consomme 180 watts-heure par mois. Est-ce réaliste? Est-ce un coût important? Est-ce qu'il peut y avoir globalement un impact positif à ces recherches?

Engins de recherche

Ceci implique qu'un utilisateur moyen fait environ 600 recherches par mois et dépense 2kWh sur une année, soit entre 10 et 20 cents en un an selon le coût de l'électricité. Si ces recherches remplacent un déplacement à la bibliothèque, permettent de choisir un trajet plus efficace pour un déplacement, ou aident à faire un achat plus avantageux (en tenant compte du cycle de vie), ce coût peut facilement être plus que compensé.



Conclusion et exemple

Exercices pour le Module 12

INF8480 Systèmes répartis et infonuagique

Michel Dagenais

École Polytechnique de Montréal
Département de génie informatique et génie logiciel

Google Build System

Un usager peut tester ses modifications récentes à un gros logiciel en téléchargeant l'ensemble du code source, le compilant et l'exécutant sur son poste de travail. Le Google Build System donne l'illusion de ce comportement à l'usager mais en fait la compilation est faite sur les serveurs. Quelles sont gains d'efficacité que permet le Google Build System?

Google Build System

Premièrement, ce système évite de copier les fichiers entre les serveurs et la station de l'utilisateur, tout se fait sur les serveurs. Deuxièmement, les fichiers sont partagés entre tous les développeurs de sorte qu'un fichier déjà compilé avec les bonnes options peut être fourni directement à un second utilisateur au lieu d'effectuer la même compilation une deuxième fois. Troisièmement, le fichier de sortie est vérifié avec la version antérieure, s'il n'y a pas de changement, il n'y a pas besoin de faire d'opérations supplémentaires; par exemple, si on modifie les commentaires dans un fichier de programme, le code compilé ne sera pas affecté et il n'y a pas besoin de refaire les tests ou l'édition de liens. Finalement, la compilation peut se faire en parallèle sur un grand nombre de serveurs, ce qui réduit les délais d'attente.



Google Wide Profiling

Le système Google Wide Profiling permet avec un surcoût très faible de mesurer et de comparer la performance de différents matériels et de différents modules et versions de logiciels.
Comment cela est-il possible?



Google Wide Profiling

Une approche par profilage permet d'ajuster la fréquence d'échantillonnage de manière à avoir un surcoût qui peut être aussi faible que voulu. Toutefois, si le nombre d'échantillons est très faible, la précision diminue. Dans le cas des systèmes Google, le nombre d'ordinateurs et de requêtes est gigantesque et il est donc possible de prendre l'information sur une très petite proportion des requêtes et néanmoins d'avoir beaucoup d'échantillons. Etant donné le très grand nombre de requêtes, on peut comparer la performance obtenue sur différents ordinateurs et sur différentes versions des logiciels. L'hypothèse sous-jacente est que toute différence sur la moyenne d'un grand nombre de requêtes sera causée par la différence de matériel ou de logiciel et non pas aux spécificités de chaque requête.

