Contrôle des processus des applications distribuées à partir de l'état des ressources: extension du DNS le système de nommage des ressources.

Amon ETTIEN - Marie Thérèse CHAMPARNAUD

Eric HORLAIT

Université Pierre et Marie Curie

Laboratoire MASI

4, place Jussieu

75252 PARIS Cedex 05

FRANCE

Résumé

Les performances des applications distribuées sont fonction des performances des ressources qu'elles consomment, principalement du système d'exploitation et du réseau. Les performances de ces ressources imposent des limites. La littérature présente un certain nombre de travaux d'optimisation portant sur l'architecture des ordinateurs, des protocoles et de leur implémentation, sur la régulation de charge, etc.

Ce papier propose un contrôle des processus liés aux applications distribuées à partir des informations d'état des ressources. Il explique l'intérêt de cette idée et propose à cet effet l'extension du système de nommage des ressources du réseau Internet pour gérer l'état des ressources.

Mots clés: Applications distribuées, serveur de noms, administration réseau, système d'exploitation, performance, processus, Open System Interconnection, ressource.

1 Introduction

Les applications distribuées impliquent deux ou plusieurs sites géographiquement parfois très éloignés. Parce qu'on ne tient pas compte jusqu'à présent de l'état du ou des site(s) distant(s), si la communication se solde par un échec du à l'état des composants impliqués, des ressources seront inutilement consommées et le coût de la communication gaspillé. Il est donc nécessaire de contrôler la durée de vie des processus des applications distribuées par l'état de tous les sites impliqués dans la communication. En clair, nous suggérons l'abandon de l'exécution des processus d'applications distribuées lorsque l'on détecte à un moment donné, que l'état d'une ressource quelconque provoquera ultérieurement l'échec de la communication. Dans ce papier nous présentons une extension du système de nommage des ressources du réseau appelé DNS (Domain Name System) pour qu'il fournisse (pour le moment) l'état des ordinateurs d'un réseau. L'objectif est d'utiliser l'état des ressources comme paramètres de contrôle dans l'exécution des processus d'applications distribuées et comme paramètre de gestion des processus dans le système d'exploitation.

La première partie présente les motivations de ce contrôle. La deuxième partie explique la nécessité de réduire le nombre de processus dans le système d'exploitation. La troisième décrit l'organisation et le fonctionnement du DNS. La quatrième présente les nouvelles fonctionnalités du DNS que nous voulons introduire afin de fournir les informations d'état concernant les ressources.

2 Motivations

Avec les applications distribuées, il arrive très souvent que des requêtes sur des sites distants se soldent par un échec du à plusieurs raisons: droits d'accès, paramètres incorrects, serveur surchargé ou désactivé, site ne

répondant pas pour cause de panne, liaison en panne, adresse erronée, etc. Tous les utilisateurs des applications distribuées, entre autre FTP (File Transfert Protocol), ont souvent reçu l'un quelconque des messages d'erreur suivants:

-"server not responding"

-"host unreachable"

...

Etant donné que l'on ne dispose pas d'informations d'état sur certains constituants du système distribué ou reparti, ou dont on en tient pas compte alors qu'elles sont ou peuvent être disponibles localement (dans une base de données d'un système d'administration de réseaux par exemple), on exécute des applications distribuées dont les correspondants (distants) ne pourront pas répondre par exemple pour cause de panne. Ceci amène: une surcharge inutile des machines hôtes, un gaspillage des ressources (bande passante du support de communication, cpu, mémoire, etc.), une augmentation inutile du coût d'exploitation, un allongement du temps de réponse des machines hôtes et donc une dégradation des performances de tous les processus.

Chaque fois qu'une application distribuée implique le "resolver" (mécanisme expliqué plus loin) et que le nom de la machine avec laquelle doit s'établir la communication n'est pas dans la zone ou dans le cache du serveur de noms (DNS) dont dépend la machine locale, une recherche va être effectuée pour faire la conversion de ce nom en adresse réseau. Internet est un réseau mondial, et une recherche peut impliquer, par exemple, des ordinateurs installés en France ou aux Etats-Unis. Cela occasionne dans certains cas, plusieurs messages. L'idée est de dire: si l'on va chercher le nom de la machine, quelque soit l'endroit où on le trouve, pourquoi ne pas s'enquérir de l'état de cette machine

(éventuellement de ses ressources) afin de savoir si, au moment précis où on obtient cette information, on peut communiquer ou non avec cette machine. Actuellement le "resolver" peut fournir entre autres, le nom ou l'adresse d'une machine, mais rien ne garantit que cette machine même sous tension, peut toujours recevoir de nouvelles communications, ou que le chemin pour l'atteindre existe.

Sous Unix, si l'information recherchée est fournie avant 20 ms au processus ayant fait l'appel au "resolver", ce processus continue son exécution, dans le cas contraire, il risque de s'endormir en mémoire et finalement sera transféré hors de la mémoire (swappé). Cela nécessite de sauvegarder tout son contexte et cette sauvegarde coûte cher du point de vue performance et consommation en ressources système.

L'architecture de la plate forme d'administration réseaux de l'OSI/ISO a défini entre autres, la gestion des performances et des configurations [ISO7498-4]. Seule la gestion des configurations prévoit l'usage des informations d'état; de plus les applications qui exploitent l'états des ressources sont très rares.

Sur un réseau local, l'administrateur système et/ou réseau peut informer chaque utilisateur qui se connecte, de l'état des ressources sous sa responsabilité soit par le courrier électronique, soit par **/etc./motd** (message of the day sous Unix) ou tout simplement de vive voix. Pour les applications distribuées sur de longues distances, il en va autrement.

3 Pourquoi réduire le nombre de processus dans le système d'exploitation peut améliorer le temps de réponse ?

Le système d'exploitation est au cœur de la gestion des informations: le traitement, le stockage et la transmission des données nécessitent des

ordinateurs performants. Des études récentes [Faller 92] ont montré que les performances d'un ordinateur se dégradent avec l'augmentation de sa charge (nombre moyen de processus en compétition pour l'utilisation du processeur). Par conséquent moins il y a de processus, plus le temps de réponse de la machine est petit. De toute façon, les performances du système d'exploitation conditionnent celles des processus.

Les applications distribuées sont de grosses applications dont l'exécution nécessite au niveau du système d'exploitation, plusieurs cycles cpu, beaucoup d'espace en mémoire, plusieurs accès disques, etc. Elles sont caractérisées par des contraintes de temps parfois très fortes.

Or les systèmes d'exploitation comme Unix ne garantissent pas qu'un processus particulier pourra être élu dans une limite de temps fixée. De plus le noyau étant non préemptif, un processus ayant des contraintes de temps s'exécutant en mode utilisateur ne pourra pas être élu tant qu'un autre processus s'exécute en mode noyau. Le système Unix n'impose pas de limite sur le temps d'exécution d'un processus, des processus peuvent donc demeurer pendant longtemps dans le système.

Le système d'exploitation est un système à files d'attentes, et les processus ont un comportement qui peut être capturé par des caractéristiques probabilistes. Il s'agit par exemple du temps d'attente d'un processus dans une file ou encore du nombre de processus dans une file ou dans le système, les instants successifs de naissance des processus, l'instant de réveil des processus endormis, etc.

Dans les systèmes d'exploitation, la capacité des files d'attente est bornée, ce qui permet à un processus d'attendre seulement dans le cas où il reste de la place dans la file appropriée. Cette politique fait rentrer dans le système tout processus tant qu'il y a de la place dans la table des processus. Le nombre de processus présents dans le système à un

moment donné est fini.

La tendance est l'accroissement des performances des ordinateurs grâce aux machines parallèles [Bjorkm. 93], aux processeurs ultra-rapide [Sites 92], etc. Notre proposition peut réduire la charge d'une machine quelconque par la réduction du nombre de processus. La conséquence immédiate est la réduction de la charge de la machine qui se traduit par la réduction du trafic sur le réseau étant donné que l'exécution d'une application distribuée génère nécessairement du trafic.

4 Organisation et fonctionnement du DNS ou Domain Name Server

Le DNS (Domain Name System) est un système distribué de recherche d'informations sur le réseau Internet. Il permet de retrouver entre autres le nom et l'adresse de n'importe quel ordinateur sur le réseau Internet. Le lecteur intéressé peut consulter [Mockap. 87a/b].

4.1 Conversion des noms en adresse IP (Internet Protocol)

Les communications entre les machines sur le réseau Internet utilisent des adresses IP. Sur les systèmes Unix, en général les conversions de noms d'ordinateurs en adresses IP se font à l'aide du fichier **/etc./hosts**. Le contenu du fichier **/etc./hosts** a le format suivant:

IP-number host-name nickname #comment

Sur un réseau local d'une centaine de machines, cette approche est efficace, mais quand le parc machines devient important, elle devient très lourde à gérer.

4. 2 Fonctionnement du DNS

A l'instar du service des noms de l'ISO [X.500], c'est un système de nommage unique pour toutes les ressources (services offerts, adresses de

machines, noms de réseaux, etc.) et repose sur un nommage hiérarchique (voir figure 1). Il comporte trois composants essentiels: l'espace des noms, le serveur des noms et le "resolver".

L'espace des noms repose sur deux notions:

- les zones: une zone correspond à un découpage hiérarchique de l'espace des noms. La zone racine (root) occupe le sommet de l'arbre.
- les "Resource Records" (RR) : un RR contient toutes les informations d'un nœud et de ses ressources sous le format:

<nom> <durée de vie> <classe> <type> <données>

Le serveur de noms maintient les RRs et répond aux requêtes des utilisateurs. Il est responsable d'une ou de plusieurs zones et gère une mémoire cache pour mémoriser les dernières informations obtenues.

Le "resolver" est une bibliothèque (fonction **gethostsbyname** par exemple). Quand cette fonction est incluse dans les programmes, elle fait appel au serveur de noms pour trouver l'information demandée.

L'espace des noms repose sur deux notions:

- les zones: une zone correspond à un découpage hiérarchique de l'espace des noms. La zone racine (root) occupe le sommet de l'arbre.
- les "Resource Records" (RR) : un RR contient toutes les informations d'un nœud et de ses ressources sous le format:

<nom> <durée de vie> <classe> <type> <données>

Le serveur de noms maintient les RRs et répond aux requêtes des utilisateurs. Il est responsable d'une ou de plusieurs zones et gère une mémoire cache pour mémoriser les dernières informations obtenues.

Le "resolver" est une bibliothèque (fonction **gethostsbyname** par exemple). Quand cette fonction est incluse dans les programmes, elle fait appel au serveur de noms pour trouver l'information demandée.

La réponse du serveur de noms peut être: 1) l'information demandée, 2) une erreur (comme par exemple le format d'adresse incorrect, la zone méconnue, etc.), 3) la référence à un autre serveur (requêtes itératives).

Le DNS implémente soit les requêtes récursives soit les requêtes itératives comme le montre la figure 1. Dans le cas récursif, lorsque le serveur de noms n'a pas l'objet demandé dans ses zones ou dans son cache, il transmet la requête à un autre serveur ainsi de suite jusqu'à l'obtention de l'information demandée. Les requêtes sont dites itératives lorsque l'adresse d'un autre serveur de nom est l'une des réponses possibles fournies au resolver. Ce dernier doit dans ces conditions interroger les serveurs de noms un par un jusqu'à l'obtention de l'adresse demandée.

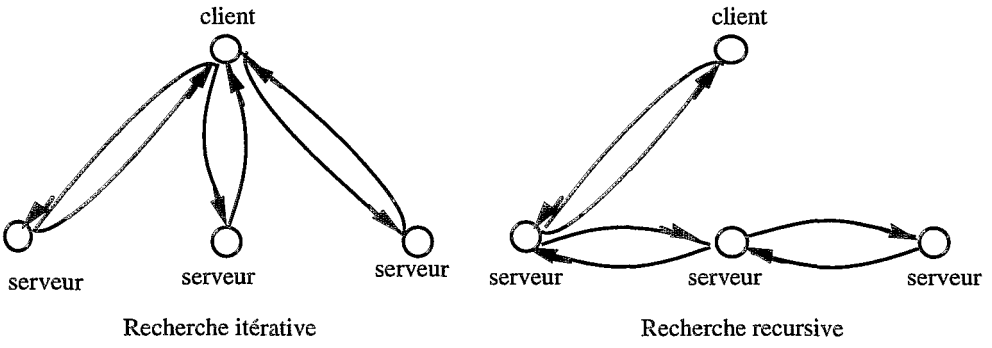

Figure 1: Différents types de recherches

Le DNS dans sa conception actuelle ne gère pas les états des ressources placées sous son autorité. Le contrôle des processus liés aux applications distribuées à partir des informations d'état requiert donc une organisation

conséquente des informations d'état et des règles de gestion cohérentes.

5 Nouvelles fonctionnalisés du DNS

5.1 Organisation des informations d'état

L'OSI (Open system Interconnection) définit des modèles d'états standard pour les objets gérés dans le cadre de l'administration des réseaux [ISO10164-2]. Un objet dans l'environnement OSI peut être dans l'un quelconque des états suivants:

- Disabled (indisponible): l'objet n'est pas en état de fonctionnement,
- Enabled (disponible): l'objet est prêt à être utilisé mais non en service,
- Active(en activité): l'objet est en service et peut accepter d'autres services,
- Busy (occupé): l'objet est en service et n'accepte plus de service,
- Locked (verrouillé): l'objet est verrouillé par les utilisateurs qui le détiennent,
- Unlocked (non verrouillé): l'objet peut être utilisé mais l'usage dépend de son état opérationnel(Active, Busy,etc.),
- Shutting Down (en arrêt): l'objet ne peut être utilisé car en arrêt.

Avec ces états, nous définissons deux ensembles d'états à partir des états standards précédents ce sont les ensembles *Recheable* et *Unrecheable* . L'ensemble *Recheable* (R) est défini par les états suivants: Enabled, Active ou Unlocked et l'ensemble *Unrecheable* (U) par les états Disabled, Busy, Locked et Shutting.

Un objet est dit R si son état appartient à l'ensemble *Recheable*, dans ces conditions, l'objet peut accepter des communications. De même, un objet

est U si son état appartient à l'ensemble *Unrecheable* où il ne peut accepter aucune communication. Ces deux ensembles se présentent comme suit:

Recheable = { Enabled, Active, Unlocked }

Unrecheable = { Disabled, Busy, Locked, Shutting }

Compte tenu de la structure hiérarchique des réseaux, la communication entre deux objets quelconque est fortement liée à l'état de tous les objets impliqués dans la communication (passerelles, ponts, routeurs, nœuds de rattachement, lignes de communication, etc.).

Père / Fils	R	U
R	R	U
U	U	U

Table 1: Règles de gestion des états des nœuds.

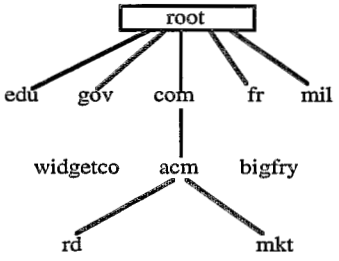

Figure 2: Organisation du DNS

Certaines règles de gestion sont à considérer au niveau du serveur de noms d'une zone. Entre autres, si un nœud père devient unrecheable, tous ses fils héritent de son état, c'est-à-dire tous les fils deviennent automatiquement unrecheable étant donné que si le père est injoignable, aucune communication ne peut être établie avec aucun des ses fils. La table 1 résume les différents cas possibles.

5. 2 Collecte des informations d'état

La gestion des états exige une vision (même approximative) sur l'ensemble des objets d'une zone. Cette gestion sera donc basée sur la vue à un moment donné qu'a le DNS sur les objets placés sous son autorité.

Les plates formes d'administration de réseaux utilisent des sondes pour déterminer la topologie du réseau sous leur autorité. Le principe consiste à envoyer un message d'identification des machines dans sa zone de visibilité à partir de leur adresse. Le DNS doit être doté d'un tel outil appelé "discovery" pour reproduire la topologie de base ou de référence de sa zone; cette topologie sera un élément clé dans la gestion des états des objets.

Le DNS envoie donc une sonde pour déterminer la topologie de la zone qu'il gère. Cette topologie doit être constamment mise à jour pour avoir une vue générale de sa zone et qui soit la plus récente possible et la plus proche de la réalité. A partir de cette topologie, il doit déterminer automatiquement la position des nœuds de sa zone les uns par rapport aux autres. La gestion des états, des noms/adresses ne se limite qu'aux objets qu'il voit. Ainsi par exemple, si un routeur à un moment donné n'est plus visible suite à une panne, il décide que tous les objets en aval de ce routeur sont dans l'état U. A l'initialisation comme après l'introduction d'un nouvel objet dans sa zone, il fait un sondage pour déterminer la topologie globale de référence; il garde en mémoire stable, cette topologie constamment mise-à-jour et qui renferme l'ensemble de tous les objets qu'il gère. Après un sondage, il compare la nouvelle vue avec la référence pour détecter les objets qui ne sont plus visibles. Tous les objets qu'il ne voit plus sont considérés comme étant dans l'état U; toutes les machines dans l'état "Busy" ou "Locked" qui sont vues, en réponse au sondage doivent par le biais d'un processus ayant les privilèges du "super utilisateur" (root sous Unix), indiquer leur état avec la date approximative où ils sortiront de cet état (voir la gestion des états). Cela permet d'éviter de s'en tenir au simple fait qu'une machine soit sous tension pour décider qu'elle peut accepter des communications. A la fin d'un sondage le DNS édite les RRs en se basant sur les règles de gestion des états .

5. 3 Gestion des états

Hypothèses:

H1: Un site pouvant abriter un ou plusieurs serveur(s) de type FTP ou autres, l'algorithme ne prend en compte que les états des machines, et suppose que le dysfonctionnement des serveurs est quasiment rare. Dans tous les cas l'état du ou des serveurs est subordonné à celui du site qui les abrite.

H2: La topologie du réseau est hiérarchique.

H3: Chaque nœud fils n'a qu'un seul père, par contre un père peut avoir plusieurs fils.

La gestion des états nécessite une démarche rigoureuse et cohérente. Le DNS s'appuie à cet effet sur la table 1. La politique est de réduire le risque, à partir du contenu des caches, de prendre des décisions contradictoires par rapport à la réalité sur l'état des ressources. Il faut donc analyser les états possibles des ressources afin de déterminer les décisions qui peuvent être prises en minimisant les risques d'erreur.

Pour l'état des ressources appartenant à l'ensemble R, il n'y a pas de raison évidente pour ne pas laisser les processus continuer leur exécution même si étant donné les distances (délais de propagation non nul, temps transfert des messages, etc.) et l'environnement non déterministe, l'état de la machine peut basculer dans l'ensemble U au moment précis où la communication lui parvient. De mêmes les ressources de l'ensemble U peuvent basculer à tout moment d'un état à un autre. Les états de U, cette fois n'ont pas le même degré de sévérité les uns comparés au autres.

On peut supposer qu'un objet "Disabled" ou "Shutting Down" prendra un certain temps pour appartenir à R, cela dépend de la cause de l'arrêt. La

seule façon d'éviter des décisions incompatibles avec l'état de l'objet est de donner une date (même approximative) à partir de laquelle il pourrait passer dans l'état R en tenant compte de la durée de l'intervention augmentée d'une certaine marge. Pour les objets dont l'état est soit "Busy" soit "Locked", en tenant compte des heuristiques ou statistiques sur la durée des tâches qu'ils ont en charge au moment précis, et qui justifient leur état, on peut déterminer la date éventuelle à partir de laquelle ils passeront dans l'état R.

Pour tous les objets de l'ensemble U, au niveau des RRs, le champ "durée de vie" doit être calculé en fonction de la date à partir de laquelle un objet est supposé appartenir à l'ensemble R. Il faut tenir compte des fuseaux horaires car la technique de l'horloge commune n'est pas encore maîtrisée sur un réseau à l'échelle mondiale. La seule manière de résoudre ce problème est d'imposer que chaque DNS qui reçoit un RR dont la durée de vie est exprimée dans un fuseau horaire autre que le sien, convertisse ce champ dans le fuseau horaire de sa zone, avant de le mettre dans son cache.

Pour des raisons de cohérence, le parcours de l'arbre doit être de père en fils. Avant d'éditer la ligne de champs d'un RR, le DNS doit déterminer si la l'objet associé est une feuille ou un nœud intermédiaire. Plusieurs cas peuvent se présenter:

- l'objet est une feuille isolée (non rattaché à aucun autre objet): le **<champ état>** de la ligne de champs du RR correspondant est édité en fonction des informations perçues,
- l'objet est un nœud intermédiaire (il est père), le **<champ état>** correspondant est édité selon l'information d'état qui résulte du calcul fait après un sondage; mais pour sa descendance, deux cas sont possibles en fonction de son état:

- le père est dans l'état U: toute sa descendance hérite récursive ment de cet état (en appliquant les règles contenues dans la table 1) étant donné qu'aucune communication ne peut transiter par lui, tous ses fils et autres descendants sont injoignables,

- le père est dans l'état R: tous ses fils reçoivent leur état tel que détecté après le sondage; les fils qui sont dans l'état U transmettent recursivement (jusqu'aux feuilles) leur état à leur descendance s'ils en ont une, par contre les "enfants" des fils (les petits fils du père) dans l'état R s'ils existent, reçoivent leur état respectif; puis selon que ces "enfants" sont U ou R, leurs descendances sont gérées à partir de la même démarche comme depuis leur ancêtre, jusqu'à ce qu'on atteigne les feuilles en se référant à la table 1.

5.4 Incidence sur l'implémentation des applications distribuées

Dans la perspective de la prise en compte de l'état des ressources, l'implémentation des applications distribuées et du "resolver" doit être revue.

Le "resolver" doit fournir l'adresse de l'ordinateur appelé que si l'état de ce dernier appartient à l'ensemble *Recheable* , sinon retourné un code d'erreur indiquant que la machine distante ne peut pas communiquer, plus éventuellement la date de reprise. Le comportement du programme qui implémente l'application distribuée doit être tel que: poursuivre son exécution si le "resolver" retourne l'adresse de l'ordinateur distant, sinon, terminer son exécution avec retour au niveau du shell avec un message d'information à l'utilisateur. La figure 3 illustre le comportement des

applications distribuées après l'intégration de la gestion des états au DNS.

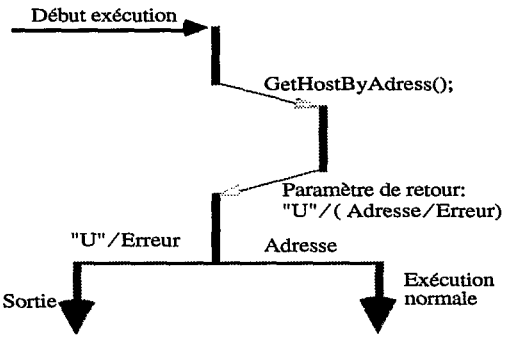

Figure 3: Comportement des applications distribuées suite à la modification du DNS

5. 6 Coût approximatif de la modification du DNS

Ce coût se reparti entre la découverte de la topologie et la gestion des états. La gestion des états va nécessiter un temps de calcul supplémentaire. En effet la conversion des temps, la gestion des filiations des nœuds demandent un temps de calcul supplémentaire.

La découverte de la topologie du réseau se fait par échange de messages entre le serveur et les nœuds. S'il y a "n" nœuds dans la zone du serveur, le coût du sondage en nombre de messages peut être:

- au moins (2*n) si le système implémente la répétition de messages suite à l'expiration d'un délai de garde; le serveur envoyant au moins "n" messages aux "n" nœuds, et recevant en retour "n" messages,
- au moins (n + 1) si le serveur fait un "broadcast" vers les "n" nœuds; le broadcast vers les sites et les "n" réponses qui en résultent dans le cas où tous les sites reçoivent le broadcast et qu'ils répondent tous. Sinon un ou plusieurs "multicast" (envoie

Références

[ISO7498-4] International Standards Organisation, Open system Interconnection - Basic Reference Model - Part 4: Management framework.

[Faller 92] Newton FALLER, "Measuring the Latency Time of Real-Time Unix-like Operating Systems" TR-92-037 June 1992.

[Mockap. 87a] P. MOCKPETRIS, "Domain Names - Concepts and Facilities Request For Comments 1034", 1987.

[Mockap. 87b] P. MOCKPETRIS, "Domain Names - Implementation and Specification Request For Comments 1035", 1987.

[X.500] International Standards Organisation,X.500: The Directory - Overview of Concepts, Models and Services.

[ISO10164-2] International Standards Organisation,System Management Overview - State Management Fonction IS 101164-2.

[Bjorkm. 93] M. BJORKMAN, P. GUNNINGBERG, "Locking Effets in Multiprocessor Implementation of Protocols", Proc, ACM SIGCOMM'93, San Francisco, September 1993.

[Sites 92] R. L. Sites, Alpha Architecture Reference Manual, Digital Press, Bedford, Mass

sélectif) seront nécessaires pour essayer de joindre de nouveau les sites n'ayant pas répondus d'abord au broadcast puis aux multicast avant l'expiration du délai de garde.

Le coût varie avec la période de rafraîchissement de la base. La deuxième solution est moins chère parce que dans tous les cas elle nécessite moins de messages que la première. Elle permet donc de faire une économie de bande passante.

6 Conclusion

L'algorithme réserve la consommation des ressources à des fins utiles, évitant ainsi le gaspillage des ressources. Il peut réduire le nombre de processus présents dans le système à un instant donné, et peut améliorer les performances du système d'exploitation et par voie de conséquence, diminuer le temps de réponse des programmes. Cependant, les systèmes distribués étant un environnement non déterministe, la vitesse de mise-à-jour (les informations changent-elles assez fréquemment ?), la cohérence des informations d'état et les différents délais de propagation peuvent occasionner des décisions contradictoires par rapport à l'état des ressources. Un programme peut être rejeté alors que l'information d'état qui a motivé la décision a changé sur le site distant. Ce peut être le cas d'un site qui relève d'une panne. Les sondages peuvent engendrer beaucoup de trafic en fonction du nombre des objets, la fréquence des sondages ne doit donc pas pénaliser les applications.

Etendue, notre proposition pourrait s'appliquer au nouveau système qui sera issu des travaux sur l'homogénéisation du système de nommage OSI (X.500) et de l'administration réseau.