|  |  |  |
| --- | --- | --- |
| **Université Badji Mokhtar-Annaba**  Faculté des Sciences de l’Ingeniorat  Département d’Informatique |  | **جامعة باجي مختار – عنابـــــــــــــــة**  كلية علــــــــــــوم الهندســـــــــة  قسم الإعــــــــــــــلام الآلــــــــــــــي |

Année 2017-2018

Année 2008-2009

**Mémoire présenté en vue**

**de l’obtention du diplôme de Master**

|  |
| --- |
| **Étude comparative de construction d'une application distribuée avec RMI et JCSP net.)** |

**Domaine :** Mathématiques, Informatique et Applications aux Sciences

**Filière :** Informatique

**Spécialité :** Ingénierie des Logiciels Complexes

**Par : Melle/Mr Rahim soufiane**

**Jury d’évaluation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualité** | **Nom et Prénom** | **Grade** | **Université** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Remerciements

J’ai l’honneur de remercié le dieu qui m’a aider pour contenue mes études et qui m adonner les aptitudes pour faire les efforts, et je ne remercier pas Allah si je remercié pas mes enseignants et de plus prés Mr BenOuhiba mon encadreur, qui m’a aider beaucoup aux cours de ma formation et au cours de ma préparation de mémoire, et sans oublier touts l’équipe de ILC qui est une équipe très sérieux, et c’est sa qui ma encourager a contenue après avoir la licence

Et je remercier l’équipe de jure pour accepter à évaluer mon travaille.

Et sans oublier ma famille et mes amés qui ont encouragé pour contenue mes étude

Et sans oublier mes collèges qui j’ai passé des bons moments avec.

Dédicaces

Les plus brèves possibles

Table des Matières

A générer automatiquement

Table des Illustrations

A générer automatiquement

Table des Programmes

A générer automatiquement (Cette table est facultative)

Introduction

Contexte de recherche

De nous jour les systèmes distribués en une grande importance dans les solutions informatiques

Et la théorie est bien spécifier mais il reste l’aspect technique qui se traduit par l’utilisation des machine virtuelle et les middlewares et les plateformes sophistiquées, et tout sa pour faciliter l’implémentation des solutions,

Et pour ce la les grandes firmes de l’informatique comme Microsoft et ibm et d’autre en mis en point des technique d’échange des donne comme DCOM et dont de Microsoft et RMI de Sun microsystème et CORBA et d’autre,

Et en remarque l’aspect distribuée des applications qui est très rependue, et en pose la question

Problématique

Faces a ces avance technologique très dévers et très avantageuse l une par a port a l autre les concepteurs et les architectes des systèmes trouvent des difficultés tranche dans l’apport d’une solution par a port a l’autre,

Parmi les problèmes de comparaisons qui sont apparue en cite la comparaison entre RMI et JCSP.net qui sont des techniques de communication entre des applications distantes qui s’exécutent sur des machines distants

Motivations

C est pour ca que les comparaisons entre ses système trouve leur impotence, par ce que la comparaison c’est un retour d’expérience qui raccourcie le chemin face au problème et donne une vision très claire et rapide et économe les efforts et diminue le redéveloppement des applications

Objectifs

Dans ce travaille en vas comparer les deux méthodes de communication de point de vues conceptuelle et de point de vus implémentation et de tout autre point techniques qui vas nous attire pour prendre une vision claire sur l’apporte de chaque technique

Contenu du mémoire

Pour exprimes les aspectes de ce travailler je consacre en premier lieu un chapitre sur la présentation de RMI

En suit je présenterai un chapitre sur csp qui est une version local de jcspnet et ensuit je présenterai un JCSP net

Apres la deuxième partie c’est la partie pratique consiste à faire la conception d’une application distribuée intituler MiniIsante qui consiste à échanger les informations médical des patients entre les médecines et ca avec en prendre en considération les deux technique RMI et JCSP.net

Et après en vas parler sur l’aspect implémentation de cette application avec les deux technique RMI et JCSP.net

Introduction

Contexte de recherche

De nous jour les systèmes distribués en une grande importance dans les solutions informatiques

Et la théorie est bien spécifier mais il reste l’aspect technique qui se traduit par l’utilisation des machine virtuelle et les middlewares et les plateformes sophistiquées, et tout sa pour faciliter l’implémentation des solutions,

Et pour ce la les grandes firmes de l’informatique comme Microsoft et Sun microsystème et d’autre en mis en point des technique d’échange des donne comme DCOM de Microsoft et RMI de Sun microsystème et CORBA et d’autre,

Et en remarque l’aspect distribuée des applications qui est très rependue, et en pose la question

Problématique

Faces a ces avance technologique très dévers et très avantageuse l une par a port a l autre les concepteurs et les architectes des systèmes trouvent des difficultés tranche dans l’apport d’une solution par a port a l’autre,

Parmi les problèmes de comparaisons qui sont apparue en cite la comparaison entre RMI et JCSP.net qui sont des techniques de communication entre des applications distantes qui s’exécutent sur des machines distants

Motivations

C est pour ca que les comparaisons entre ses système trouve leur impotence, par ce que la comparaison c’est un retour d’expérience qui raccourcie le chemin face au problème et donne une vision très claire et rapide et économe les efforts et diminue le re - développement des applications

Objectifs

Dans ce travaille en vas comparer les deux méthodes de communication de point de vues conceptuelle et de point de vus implémentation et de tout autre point techniques qui vas nous attire pour prendre une vision claire sur l’apporte de chaque technique

Contenu du mémoire

Pour exprimes les aspectes de ce travailler je consacre en premier lieu un chapitre sur la présentation de RMI

En suit je présenterai un chapitre sur csp qui est une version local de jcspnet et ensuit je présenterai un JCSP net

Apres la deuxième partie c’est la partie pratique consiste à faire la conception d’une application distribuée intituler MiniIsante qui consiste à échanger les informations médical des patients entre les médecines et ca avec en prendre en considération les deux technique RMI et JCSP.net

Et après en vas parler sur l’aspect implémentation de cette application avec les deux technique RMI et JCSP.net

Chapitre 1   
Présentation de RMI

Introduction générale:

L’échange d’état entre machine dans un système distribué se fait par l’échange de message , c est pour sa que l’échange de message devient le problème centrale dans la concept des applications distribue, mais cette distribution de système a un bénéfice major c est le partage des ressource et faire semble que un ensemble de nœud comporte comme une seule nœud pour ne pas avoir un seul système centralise et la panne de cette nœud centrale faire tombe tout le système,

D’une autre part, cette distribution apporte des problèmes:

1. L’hétérogénéité :

une application distribuée est un couche applicatif logique qu situe au-dessus des autre couche y a compris la couche midellwer et la couche système d’exploitation et la couche matériel, et ses couches ont amenée a coopérer entre eux mais ses composantes ont leur spécification que rendre ce partage très difficile par exemple la couche réseaux est souvent défirent d un nœud a un autre et en a amenée a résoudre ce problème pour que les autre couche peuvent échanger des message, de-même pour la couche système d’exploitation Windows Lunix Unix Mac-OS.

De-même pour la couche midellwer en trouve .net RMI RPC CORBA et DCOM .. etc

De-même pour la couche application en trouve des langage de programmation déférant qui ont des spécification comme le langage JAVA et C#

1. Problème de la concurrence

Le partage de ressource pose un problème par ce que ces ressources souvent ne sont pas partageable en même temps par les composent il faut les ordonner, comme un ficher ou un table de base de donne ou un imprimante, de plus le partage de ressource dans un système reparte se fait par des messages et ses message pouvons être perdu se qui pose un autre problème dans la gestion de la concurrence par rapport au système centralisée.

1. Problème de sécurité :

La sécurité ses pose dans toutes les système informatique cependant dans un système distribue ce problème se produit de la distribution elle même, par ce que un message échangé entre deux machine peut être interrompue ou détruit et même modifiée de plus en a le problème d’ authentification pour identifié un client auprès d’un serveur il y a aussi l’attaque de service par l’essaie de saturer les capacités de services offerte par les composantes de système il y a aussi la sécurité de code mobile.

1. Problème de panne

une panne peut apparaître dans une couche inférieur par exemple la couche matériel provoque un panne dans les couches supérieur

1. L’absence d information globale:

le système distribué ne prend pas en considération une mémoire globale qui facilite la coordination entre ces composants, et le seul moyen de partager l‘état d un processus est l’envoi de message.

En effet , ce message peut faire de retard alors dans ce retard l’état de ce processus peut changée ce qui fait que ce message ne reflet pas l’état correct de ce processus, c est pour sa que la connaissance de l’état globale de système et une grande challenge dans le système distribuée

1. Le problème de vérification:

Ce problème existe dans tout les systèmes, particulièrement dans les système de multiple-processus.

Cependant ,l’équipe de test vas passer un temps considérable dans le test et validation de système.

et le coût de cette phase est de 60 % de coût globale de système, mais dans la phase de mise ne production souvent il apparue des bugs qui ne sent pas mise en considération dans la phase de teste alors l’équipe de d développement vas passe des nuits blanche a essayage de connaissance de la cause d l’erreur pour la corriger se qui rendre le problème de test et validation un très grande problème qui va être fatale dans les système distribuer comme par exemple le problème d’interblocage

Et pour résoudre ses problème il faute adapter les phase de développement pour mieux pailler a se problème soit dans la phase de développement ou de teste ou déploiement et la méthode la plus conviviale est la méthode qui palier le mieux a ses problèmes ces pour sa que la comparaison ente les diffèrent midellwer par exemple jeux un rôle très important

1.1 Introduction RMI

La nature dynamique de la Platform java été étendu par le technique RMI qui permet d’appel des méthodes d’un objet distant comme si il est dans la même machine virtuelle local, elle est etuliser en standard dans la platforme JAVA depuis la version 1.1,elle permettre au développeurs de se libérer des communication de la couche réseau par exemple TCP\IP et de concentrer sur la conception des services affère par les objets comme ils sont dans la même machine, de plus la machine virtuelle JAVA aide beaucoup les développeurs pour ne pas occuper de l’hétérogénéité des plateformes, ce qui donne une point forte au plateforme java.

RPC remot procedure call:

Et pour comprendre RMI vaut mieux parler de RPC (remot procedure call) qui est un mécanisme de haute niveaux pour échanger les messages entre les nœudq d’un système distribue:

Le RPC peut être présentée comme suite un machine peut appeler un procédure dans une autre machine en lui transmettre des paramètre et ce procédure peut renvoyer une résulta ou exécuter une action comme si ce procédure situe dans la même machine, en effet cela peu avoir des conséquence:

- un machine des deux peut tomber en pane entre le temps de l’appel et le retour de résultat,

- il y a un problème d’ adressage des paramètres a résoudre par ce que les paramètres sont dans un espace d’adressage différent de la zone d’adressage de la machine qui exécute le procédure.

Pour résoudre ce problème le processus qui émetteur la demande fiat appel a une souche locale qui est un représentant de procédure distante et cette souche faire la connexion et faire passe les paramètres a un autre souche distant qu représente l’émettre chez ce processus distant, alors l’appel de procédure ce fait simplement comme si elle est locale le la distribution soit transparent vis a vis de développeur , ce mécanisme est schématisé comme suivant :

Extraire les paramètres de message

Formalisme de message nom\_procedure (paramètres)

Application client

Application serveur

Client

Souche client

Souche serveur

Serveur

Temps

Extraire le résultat

Formalisme de message de résultat

Communication entre client et serveur RPC

Quelque remarques :

* Le passage des paramètres par référence se fait par un technique de copie restauration le client envoi une copier des paramètres au serveur et il reste bloquée jusque a se que le serveur traite le le résultat et il envoi une copier modifier au client et le client faite la restauration de celle si sur l’ancien valeur.
* Le passage des structures complexe comme les arbres par références peut ce fait en vas et viens par besoins.
* Les souche sont générer automatique et le programmeur cote client n a besoin que d une interface qui contient les entêtes des procédures a appeler et dans la partie serveur le programmeur doit implémenter cette interface
* Dans le cas de procédure qui na pas de valeur de retour le serveur envoi un notification au client pour le libérée

1.3 Architecture RMI :

L’appelle de méthode a distant n’est en réalité que un simple RPC assurée par le stub cote client et Skelton cote serveur, avec une capacité d’envoyer et de recevez les objets autre que les tableaux et type primitive comme est le cas dans RPC,

Cette technologie est représentée par l’architecture suivante :

client

servuer

skilton

stub

Couche transporte

Couche de referancement

Couche de referancement

Système RMI

Figure 1.1 Architecture RMI

1.2.1 Le stub : c’est l’objet représentant de l’objet distant dans la machine client ils reprisent l’objet distant a chaque appel de méthode il fait les tache suivantes:

Établer une connexion au prés de serveur

Envois l’appelle de méthode avec les paramètres (sérialisation)

Attendre le résultat

Dé sérialiser le résultat

1.2.2 Le Skelton : c est le représentant de l’objet appellent de client dans la machine serveur il fiat a chaque appel de méthode les taches suivants :

Il écoute les appelles des méthodes

Il fait la lecture des paramètres des méthodes si il ya des appels distant

Il fait appelle la méthode de l’objet référencée

Le sérialisation de résultat

Et envois le résultat au demandeur

Il est a note que le Skelton et le stub est généré automatiquement par le compilateur rmic le nom de la classe stub est XXX\_stub et le nom de la classe Skelton est xxx\_skel avec xxx est le nom de l’objet distant

1.2.3 La couche de refinancement : elle est responsable de référencer les objets distants pour donner les informations des objets à la couche transport,

Nb pour cette couche en peut utiliser la couche standard ou en peut par exemple utiliser le JNDI qui est un standard très interdisant pour laisser la configuration de serveur et client par un semple ficher de configuration sans soucier de gestion de cette couche

1.2.4 La couche transporte : cette couche effectuer les échanges des donnes entre les deux machines par un serveur socket cote serveur et socket cote client

En peut personnaliser cette couche par un protocole spécifique par un implémentation de socketfactory.

1.2 Mécanisme de téléchargement des définitions des classe distants :

Ce technique nous a permet de localiser et communique et charger la définition des objets distant, si objet distant sont héberger dans un composent qui s’appel registre ce registre envoi au client demandeur un stub qui est un proxy reprisent l’objet distant et a chaque appelle de méthode il envois les paramètres et récupère le résultat, il est a note que tout cette transparence vis-à-vis de programmeur est garante par un simple serveur web

Ce mécanisme de communication et représentée dans le chemin suivant :

Serveur Web

Client RMI

Serveur RMI

Registre RMI

Serveur Web

Mécanisme de communication RMI

Les ligne pontilles reprisentent le protocole de transfer les parametre t reseltat qui est assuree parle serveur web de client et serveur

Et les lignes continue reprisent le protocole RMI proprement dite

Techniquement ce mécanisme de transfert ulilise la sérialisation et deserialisation des objet pour permettre le transfert des paramètre et résultat et les exeptions et même le stube, avec le technique de chargement dynamique des definition des classe, le resultat c’est un platform destrebuee qui perrmetre l’appel des methode destant comme c’est c’est une appel locale.

Le mécanisme d’activation des Object distant :

RMI utilise un technique d’activation dynamique des objets qui ne sont pas référencée par des clients mais sont juste exposées aux clients, par se que les implémentations des ces objet peuvent attendre un million d’objet et ne peuvent pas être tous activé a tout moment dans se cas ils sont dans un état passif et le passage a l’état active et retardée jusqu’ a la premier appelle de méthode distant.

Gestionnaire de sécurité java

Le gestionnaire de sécurité java c’est un api java qui peut être utilisé dans le cas de tentative d’accès au poste par des machine distant et en peut utiliser avec la technologie RMI pour empêcher ou autorise quelque poste pour accéder au poste en distance, de plus sa nous invite de contrôler l’accès par un semple ficher de configuration qui l’administrateur peut éditer pour changer la stratégie.

Vous trouver un exemple dans l’annexe de l’utilisation de Security Manager qui en peut ne pas utilise.

Garbage cllector distribuée

Le mécanisme Garbage collector distribuée de JAVA utilise un technique de nombre de référence pour chaque objet distant ce nombre est incrémenté chaque Foix que il y a référencement par un client, ce client envoi un massage indique que l’objet distant est référencée et ce nombre est décrémentée chaque voix que cette référence est libérée par le client, dans ce cas la machine virtuel a le doris de récupérer cet objet si il n ya pas un référence local de ce objet

Multiplexage des communications

Ce technique est utilisé par RMI pour ne pas utiliser un socket serveur pour écouter les clients mai un semple socket, par exemple dans quelque système l’administrateur empêche les applette à ouvrir un porte pour l’écoute des clients, et sa pour le nécessité de sécurité, dans ce cas la seule moyen et de multiplexe le canal de communication de client pour envoyer et recevez les informations

Annexe : RMI en Pratique

Et pour comprendre ce mécanisme de communication en va présenter en vas utiliser un exemple de code basique qui reprisent l’appel d une méthode d un objet distant par un client :

Création de serveur RMI avec le registre :

Cette étape est très simple a partir de version 6 de java par ce que la définition de stub est transparent de point de vue de programmeur, elle est gérer par la Platform java,

Cependant le développeur n’a que faire la conception des objets à héberger et sa se fait par une conception d’une simple interface java qui représente les méthodes qui les clients peut appeler a distant cette classe a deux règle simple

1. Il faut qu’il étendre la clase Remot
2. Il faut ajouter throw remotExeption dans chaque méthode distante

Une interface qui représente une seul méthode a le squelette suivant :

public interface MonIterface extends Remote {

TypedeRetour NomMetode(TypeParametre t) throws RemoteException;

}

L’implémentation de cette classe par un objet, cette opération a aussi des règles :

1. Bien sur il faut implémenter toutes les méthodes de cette interface et,
2. Une classe distante doit ´étendre la classe java.rmi.server.UnicastRemoteObject [[1]](#footnote-0)(d’autres possibilités´es existent) [[2]](#footnote-1)
3. Définir un constricteur

Cette implémentation a la forme suivante

public class nomObjetDestant implements NomInterface {

public nomObjetDestant () {

super();

}

public TypedeRetour executeTask(TypeParametre t) {

// Traitement a faire

}

}

Âpres il faut créer un programme java qui démarre le service de ce objet :

Ce programme a la forme suivant

public class Lanceur {

public static void main(String[] args) {

if (System.getSecurityManager() == null) {

System.setSecurityManager(new SecurityManager());

}

try {

nomInterface nomObjet = new nomObjetDestant();

nomInterface stub =(nomInterface) UnicastRemoteObject.exportObject(nomObjet, 0);

Registry registry = LocateRegistry.getRegistry();

registry.rebind("nomSercvice", stub);

} catch (Exception e) {

System.err.println("ComputeEngine exception:");

e.printStackTrace();

}

}

}

Explication :

La première instruction

if (System.getSecurityManager() == null) {

System.setSecurityManager(new SecurityManager());

}

Il permet d’installer un manager de sécurité java par ce que en a affaire avec un téléchargement de définition des classe et sa doit être contrôler par des règles sur mesure, cette point va être présenté après dans la compilation et lancement de l’application.

Apres en a l’inspection suivante:

nomInterface nomObjet = new nomObjetDestant();

Cette instruction est une instruction standard il représente la créatine d’un référence local de l’objet serveur

Après en a :

nomInterface stub =

(nomInterface) UnicastRemoteObject.exportObject(nomObjet, 0);

Cette instruction transforme cette référence local en une référence qui pet être hibargé dans un registre RMI

Il est a note que ce 0 et un enter si il est 0 c a d que le porte de l’écoute de ce service, est définie soit par le système d’exploitation ou il est par défaut, et si le programmeur veut spécifier le port il doit changer le 0 par un nombre entier qui représente le porte² représente soit

Registry registry = LocateRegistry.getRegistry();

Cette instruction lance un registre RMI pour héberger l’objet destinent qui vient d’être créé,

Apres en l’instruction :

registry.rebind("nomSercvice", stub);

Cette instruction héberge l’objet dentant dan le registre

Et tout sa doit être envelopper par un try catche, par ce que ces instruction peuvent génère une exception par exemple échec d ouvrir un registre dans ou errer de hébergement de l’objet distant… etc.

Il faut préciser que  bind, unbind, ou rebind peuvent être appelé pour un registre qui se trouve dans la même machine de l’application serveur par contre l’instruction lookup que en vas utiliser dans le client pour récupérer une référence de l’objet distant peut être appelé pour une application distant et la raison de cette restriction et la sécurité

Après en vas créer un client qui vas demander les services de ce ou ses objets distant :

La création d’un client a la forme suivant :

public class ComputePi {

public static void main(String args[]) {

if (System.getSecurityManager() == null) {

System.setSecurityManager(new SecurityManager());

}

try {

Registry registry = LocateRegistry.getRegistry (args[0]);

nomInterface comp = (nomInterface) registry.lookup ("Compute");

TypeRetour valeur = comp.NomMethode();

System.out.println(valeur);

} catch (Exception e) {

System.err.println("ComputePi exception:");

e.printStackTrace();

}

}

}

C paragramme et très simple et basique il créé une référence au registre qui se trouve dans une machine qui est déterminée par l’argument 0 du programme et el interroge ce registre pour récupérer une référence de l’objet distant qui est représenter par un stub transparent dans les versions antérieur de la version 5 de java

Apres il doit invoquer une méthode de ce objet

Il est à noter qu’il faut créer une interface identique à l’interface de server pour récupérer la référence de l’objet distant

Une autre remarque très important que dans le cas d’utilisation des objets comme paramètre d un appel de la méthode distant ou que ses objet sont un type de retour dans une méthode distant il faut que ses type soient des type primitif ou des clase distant ou qualifier d’être serialisable

El est à noter que les constricteurs obligatoires ans l’implémentation de l’interface distant appel un contracture de stub qui est une tache transparent a partir de la version 6 de JDK ce stub contient un référence d l’objet distant et une méthode invok() qui a la tache d’appel de méthode distant[[3]](#footnote-2)

Dans le site de oracle vous trouvez un exemple très puisant ,Ce exemple et le serveur de calcule distribuée , il est très utile dans scénario, si en a une puissante machine qui peut exécuter un code arbitraire fourni par le cliente et transmettre le résultat au client, car il nous a permette de voire le cas de transmettre des informations de client ver le serveur (par les paramètres d’appel de méthode qui représente un classe contiens un méthode arbitraire a exécuter par le serveur) et de serveur au client (comme est le cas dans tout les application qui utilisent RMI c a d par le renvoi de définition de l’objet distant et le résultat après l’exécution).

Il est à noter que la gestion de concurrence est la tache de développeur et sa se faite par synchronaize et wait etc. et sa se peut introduire un inter-blocage dans le code par exemple

Note sur la sécurité dans le cas d’utilisation d un gestionnaire de sécurité dans le code de client ou de serveur il faut créer un fichier de configuration de sécurité, et voici un exemple de deux fichiers de sécurité qui donne tout les permissions à l’exécution dans un répertoire donnée :

Pour le serveur un ficher appeler server.policy

grant codeBase "file:/home/ann/src/" {

permission java.security.AllPermission;

};

Pour le client un fichier appeler client.polycie

grant codeBase "file:/home/jones/src/" {

permission java.security.AllPermission;

};

après il faut démarre le registre par cette commande :

start rmiregistry

Il est a note que la porte par défaut de registre et 1099 pour démarre le registre dans une autre porte vous devez utiliser la commande suivante par exemple:

start rmiregistry 2000

Exécution de serveur :

java -cp c:\home\ann\src;c:\home\ann\public\_html\classes\nomInteface.jar -Djava.rmi.server.codebase=file:/c:/home/ann/public\_html/classes/nominterface.jar - Djava.rmi.server.hostname=mycomputer.example.com

-Djava.security.policy=server.policy

PakageObjetDestant.nomObjetDestant

execution de client :

java -cp c:\home\jones\src;c:\home\jones\public\_html\classes\compute.jar -Djava.rmi.server.codebase=file:/c:/home/jones/public\_html/classes/

-Djava.security.policy=client.policy

PakageClient.nomClaseClient mycomputer.example.com 45

Le paramètre mycomputer.example.com désigne un nom de machine

Et le paramètre Djava.rmi.server.codebase=file:/c:/home/ann/public\_html/classes/nominterface.jar et Djava.rmi.server.codebase=file:/c:/home/ann/public\_html/classes/nominterface.jar reprisent le répertoire de client et le serveur

Pour plis d’explication sur le sujet de sécurité et de compilation et d’exécution consulter le site oracle dans la section RMI, sa ne défiance pas des base de compilation et exécution des applications java, il existe des exemples de code RMI dans le net qui n’utilisent pas le gestionnaire de sécurité dans ce cas le code va devenir plus facile avec la plate-forme eclipse.

1.4 Conclusion

Comme vous voyiez il n y a ni sockette ni porte malgré que en peut paramétrer les portes mais par défaut tout marche bien et le développeur va concentrer sur la conception de la couche métier et il a l’avantage de développer une application distribuée comme si elle est une seule application dans une seule machine virtuelle,

***CHAPITRE II JCSP.net:***

Définition :

CSP c’est un langage formel de modélisation des processus concurrent introduit en premier par [C. A. R. Hoare](https://fr.wikipedia.org/wiki/C._A._R._Hoare" \o ") en 1978 et depuis il a été améliorée et implémenté par plusieurs outil de modélisation et des bibliothèque de programmation et même leur syntaxe par l’ajout des types de donne et des formalisme comme le pi-calcule dans la dernière amélioration de occam-pi

Basée sur les processus et les canaux de communication, ce langage permettre de mettre les processus en concurrence ou en séquentielle ou alternative indéterministe, ce qui permet au développeurs de concentrer sur le le métier de chaque processus sans occuper de la partie technique de la communication comme si ces processus sont des objets de monde réel, ce qui n est pas permette dans la programmation séquentiel

Occam permette des programme de bénéficier de deux ponts très important

1- il permet de rendre le programme très lisible par cacher l’aspect technique de code source

2- il permet la vérification de la validité de programme avec la logique temporelle ce qui n’est pas permet dans les autre langages comme C++ ou Fortron ou pascal

Syntaxe CSP occam

Les processus dans occam son des objets qui exécutent une tache en permanente et n’acceptent la communication que par des canaux

Et ses processus peuvent être lancée séquentiellement ou parallèlement ou alternativement par un choix indéterminée

l’envoi des donnés sur un canal quelconque se fait par le mot clé ! sur le ce canaux

Et la réception se fait par le mot clé ? sur le canaux

Par exemple :

Mychannel? Ou Mychannel!

Et en a pas a soucier de l’implémentation de la communication

Et les canaux sont de type unidirectionnel ou bidirectionnelle de type un a plusieurs ou plusieurs a un

Il est a note que en peut mettre des processus encapsuler dans un autre processus par ce que la seul moyenne de communique un processus avec le monde extérieur et les canaux et alors le fête d ne pas exporter un canaux a l”Extérieur le rendre privée, avec se technique en peut exposer le canaux de processus conteneur et ne pas exposer le canaux des processus internes

JCSP

Introduction

L utilisation de la programmation séquentielle pour créé un système concurrent est très difficile par ce que le modèle implémente souvent ne reflet pas la réalité , en plus en a pas un moyen de vérifier les propriété de système, et avec les méthodes de teste traditionnel en peut être satisfaite de fonctionnement de système mais il reste un ou plusieurs cas que en a pas testée et qui soit généré chez le client et ,alors l ‘équipe de test vas passer des nuits blanches a provoquer le bug pour corriger, surtout dans un application muté thrides,

L’un des solution de ce problème et une méthode athématique qui base sur la vérification de modèle, ce modèle il faut qui il reflète la réalité ce qui n est pas facile a concevoir , s est pour sa que il est apparue des bibliothèque de programmation capable de mètre des threades en concurrence en suivant les concepts mathématique de langage CSP comme JCSP qui est un implémentation java de CSP aucam et JCSP.net qui est la version distribuée de JCSP qui suit ocame-PI.

Présentation de JCSP

##### Un processeur dans jcsp est un classe qui implémente interface CSProcess de

package jcsp.lang;

 public interface CSProcess

{

    public void run();

}

Et de même la base des canaux ce est deux type de canaux ChannelInput qui a un sul methode Read() :

interface ChannelInput {

public Object read ();

}

cet type de canaux est utilisé dans les autre type des canaux pour lire ,elle est bloquant jusque a se que l’autre bout de canaux a fait le lecture

Et il n a pas de tampon,

l’autre type de canaux et utilisee pour écrire a l’autre bout de canaux c est ChannelOutput il a une seul méthode write(Object o) qui est bloquant et n a pas de tampon

interface ChannelOutput {

public void write (Object obj);

}

Les autre type de canaux héritent de ses deux type basique en site :

One2OneChannel qui est bidirectionnel

Et One2AnyChannel un rédacteur et plusieurs lecteur

Et Any2OneChannel plusieurs rédacteur et un lecteur

Et Any2AnyChannel plusieurs rédacteur et plusieurs lecteur

Un exemple de processus en jcsp :

class SuccInt implements CSProcess {

private final ChannelInputInt in;

private final ChannelOutputInt out;

public SuccInt (ChannelInputInt in,

ChannelOutputInt out) {

this.in = in;

this.out = out;

}

public void run () {

while (true) {

int n = in.read ();

out.write (n + 1);

}

}

}

Comme en a dit par défaut les canaux n ont pas de tampons ,en effet il y a un classe interne qui occupe de bufeur

a savoir ChannelDataStore et bufer et InfiniteBuffer

- ChannelDataStore par défaut elle est de type ZiroBufer que signifier qui n a pas de tampon.

- la classe bufer qui a un tampons de type fifo

- Et InfiniteBuffer qu a un tampons de type fifi infini qui bloque le lecteur si il est vide et ne bloque pas les rédacteur jusque a il attendre le limite physique de la mémoire.

Il est a noter que les canaux sont synchronise et cette synchronisation se fait a laide de moniteur

Le parallélisme se fait avec un classe parallèle qui prendre en paramètre un tableaux des processus, cette classe exécute toutes si processus en parallèle dans des threads différente saufe la dernier qui est exécutée dans la même thread ,et cette thread parallèle se termine si toutes le thrides interne se termine

Le choix indéterminée ALTernative :

jusque ici en a parler que de l’exécution déterministe des choix ou commande c et a dire a l’arrive de message dans le canaux il doit être traité mais JCSP a un commande ALT qui peut traité un des message qui ont une garde vrai dans cette garde en peut combiner des teste binaire avec l’arrivage des entrées, et le choix d’une commande se fait par choix indéterministe.

CSTimer:

C’est un classe qui donne la possibilité de crier un timer pour qui il soit utilisée comme garde dans la classe Alternative

JCSP.net

L’ extension de JCSP par l’aspect distribué a donne naissance a JCSP.net, c est a dire que les canaux ne sont pas locaux mais sont des canaux destant qui ont des extrémité distant c est a dire que l’ émetteur et le récepteur ne sont pas dans la même machine.

avec une différence syntaxique que en plus de CSP de haward est ajoutée le Pi calcule qui reprisent dans le calcule des processus comme le lambda calcule dans informatique traditionnelle

et une autre différence sémantique que les objets passe dans un canaux locale par référence mais dans un canaux réseaux sont passée par copier

Processus

Processus

Net Chanel output

RX

TX

NET Chanel input

TX

RX

Architecture de canaux virtuel de JCSP.net

Un canaux en jcsp.net et deviser en client et serveur lien et ses deux liens sont déterminée par un adresse IP et porte et un numéro virtuel utilisé pour multiplexer et démultiplier les message de un même processus et les liens sont enregistré dans un serveur (Chanel name server CNS)

Ce diagramme présente ce mécanisme:

CNS

“name” Q 42

Q

Processus sever

P

Processus Client

2

1

3

1) le serveur demande au CNS d ‘ouvrir un canaux sur le numéro virtuel 42

2) et le client demande la connexion au canal 42

3) le client envoi au serveur les données sur le canal 42

Il est a noté que après la connexion de client et le serveur on a pas de soucis sur le porte et l’adresse IP et le numéro virtuel de l’autre bout de connexion, en est besoin que de nom de canal.

Les type de canaux réseau en plus des canaux locaux vue précédemment sont

One2netchannel une machine qui peut écrie sur le réseaux

Net2onechannel une machine qui peut lire sur le réseaux

Net2Anychannel plusieurs machine qui peuvent écrire sur le réseaux

Any2netCahnnel plusieurs machin qui peuvent lire sur le réseaux

Remarque en peut ne pas utiliser un CNS mai envoyer un objet NetChannelLocation dans un canaux existant ce objet représente les information de location qui peut être utiliser comme paramètre dans le connecteur de canaux, en parle ici de canaux anonyme.

Sa peut être utile dans le cas ou en peut communique entre processus sans que le canaux partagée de CNS soit libre, c’est comme un connexion local de type one2one ,en effet le CNS exige que une seule échangé soit possible et dans les connexion anonyme elle est prive et ce problème de synchronisation ne se pose pas.

En peut même envoyer un processus entier dans un canaux voici un exemple qui peut être utile dans le cas ou en veut exécuté un code dans une autre machine

while (running) {

Bond james = (Bond) in.read ();

james.plugin (a, b, c);

james.run ();

NetChannelLocation escapeRoute =

james.getNextLocation ();

One2NetChannel escape =

new One2NetChannel (escapeRoute);

running = james.getNuke ();

escape.write (james);

escape.disconnect ();

}

En effet ce code représente la méthode run() d’un processus distant qui accepte un processus et le exécute dans un autre thread ce qui est similaire au agent mobile dans les paradigme connue de code mobile

En peut aussi déplacer les extrémité des canaux mais sa change la topologie de réseau qui peut amenée a des situation inattendue soit disent interblocage

Tout ce qui est nécessaire est de communiquer le nom du canal CNS (ou NetChannelLocation) au nouveau processus d'écriture se que en appel migration de canal.

Explication de mécanisme de migration de canaux :

Q

P0

x

P1

R

y

Q

P0

x

P1

R

y

Avant

Après

Considérons un processus, x, sur le nœud Q, qui gère actuellement le canal "foo" enregistré par le CNS.

• Il veut transmettre cette responsabilité à un processus (volontaire), y, dans le nœud R, avec lequel il est en contact.

Les processus qui écrivent sur "foo" ne doivent pas être conscients de cette migration de canal.

Revenons à l'état initial ("foo" desservi par x sur le noeud Q).

D'abord, le processus x gèle le nom "foo" sur le CNS

Le CNS renvoie une clé de dégel pour traiter X

Le CNS ne résout plus "foo" pour les nouveaux auteurs et interdit également les nouveaux enregistrements du nom.

Le canal réseau est supprimé du processeur Q.

• Tous les messages en attente et futurs pour ce canal (42) sur Q sont renvoyés (NetChannelIndexException)

La méthode write () en P1 gère ce rebond par appel au CNS pour le nouvel emplacement de "foo".

• Cela ne réussira pas avant

le processus x (sur le nœud Q) passe sur le nom du canal ("foo") et la touche de déblocage du CNS

• et le récepteur (processus y sur R) déverrouille le nom "foo" (en utilisant la touche) et le réenregistre.

et le récepteur (processus y sur R) déverrouille le nom "foo" (en utilisant la touche) et le re-enregistre.

• La méthode write () en P1 entend maintenant depuis le CNS le nouvel emplacement de "foo"

et renvoie le message qui a été rebondi.

• Le (s) processus d'écriture à P1 (et ailleurs) sont

ignorant la migration

Résumé

JCSP.net permet la communication par canal virtuel entre les processus sur des machines séparées (JVM).

• Les canaux d'application / connexions entre les machines sont configurés (et supprimés) dynamiquement.

• Les canaux / connexions sont multiplexés sur les liens.

• Des liens peuvent être développés pour n'importe quel protocole de réseau et branché sur l'infrastructure JCSP.net.

• Aucune gestion centralisée - connexions per to per (amorçées à partir d'un serveur CNS de base).

• Les courtiers pour les services de correspondance définissables par l'utilisateur sont faciles à configurer en tant que serveurs d'applications ordinaires.

Les processus peuvent migrer entre processeurs (avec des classes chargées dynamiquement selon les besoins) - d'où les agents mobiles, les worker farmsl, le grid computation ...

• JCSP.net fournit exactement le même modèle de concurrence (CSP / occam) pour les systèmes en réseau que JCSP fournit dans chaque nœud physique de ce système.

• La logique du réseau est indépendante de la distribution physique (ou même si elle est distribuée).

• Accent mis sur la simplicité - à la fois dans la mise en place de réseaux applicatifs et dans le raisonnement à leur sujet.

Annexe A

Annexe B

***Résumé***

Le résumé ne doit pas excéder une douzaine de lignes. Il doit être écrit de sorte à permettre aux lecteurs de déterminer rapidement si le mémoire l’intéresse ou pas. Pour cela, il doit donner un bref aperçu sur le sujet, la problématique, la motivation qui pousse à la recherche de la solution ainsi que ce qui a été réalisé (aperçu de la solution proposée). Le résumé se termine par une note d’optimisme qui situe la solution par rapport à ce qui existe. Le résumé ne comporte pas de sous titre.

**Mots clés**. Plug-in, Eclipse, Composant logiciel,

**ملخص**

مع مرور الزمن

**كلمات مرشدة.** برمجيات، فصل الانشغالات،

1. « Java Remote Method Invocation ». [↑](#footnote-ref-0)
2. Voir l’exemple de site oracle le moteur de calcule qui n’etendre pas la classe java.rmi.server.UnicastRemoteObject [↑](#footnote-ref-1)
3. « Java Remote Method Invocation ». [↑](#footnote-ref-2)