

Mise en œuvre d'un réseau P2P : JXTA

Harir Mekki Rachida

Université des Sciences et technologie Mohamed Boudiaf d'Oran

Faculté des Sciences

Département d'Informatique

mekki@univ-usto.dz

1. Introduction

La plupart des architectures fournissant des services sur le net sont de type client/serveur. Ce type d'architecture décharge le client au détriment du serveur sur lequel sont effectués la plupart des travaux pour la délivrance de services. L'inconvénient majeur de l'architecture client/serveur est le goulot d'étranglement serveur : plus il y a de clients connectés au serveur, plus l'efficacité du réseau est limitée par les caractéristiques physiques du serveur. De plus l'apparition des adresses IP dynamiques (les fournisseurs d'Internet attribuent à chaque connexion d'une machine sur le réseau une adresse IP différente) fait que le serveur ne peut être accédé qu'en connaissant son adresse IP. Aussi, le développement des réseaux sécurisés par un firewall qui filtre toute communication initialisée depuis l'extérieur du réseau. Tous ces inconvénients peuvent être contournés dans un réseau peer to peer. Dans un réseau peer to peer, tous les peers sont interconnectés et peuvent agir à la fois comme client et comme serveur. Ainsi, toute personne du réseau peut offrir des services et en consommer. Une recherche sur un service particulier nous indique les peers qui offrent celui-ci, il suffit alors de se connecter à celui de notre choix. Toute dépendance à un unique serveur est donc éliminée. Le résultat est donc un réseau hautement disponible sur lequel un service n'est pas confiné à un seul endroit.

JXTA (à prononcer juxta, en provenance du mot juxtapose) est une suite de protocoles Open Source développée par Sun. Les peers JXTA créent un réseau virtuel au dessus du réseau physique, cachant ainsi la complexité de celui-ci. Dans un réseau virtuel JXTA, chaque peer peut interagir avec tous autres, sans se soucier de son emplacement, du type de composant ou de l'environnement d'exécution et même s'il est situé derrière un firewall ou qu'il utilise une autre couche de transport réseau. La technologie JXTA fonctionne sur n'importe quel composant, qu'il s'agisse d'un PC, d'un téléphone cellulaire ...

2. L'architecture peer to peer

Tout réseau en peer to peer contient des éléments de base tels que les peers. Ces peers communiquent entre eux via un certain nombre de protocoles et peuvent se regrouper en peergroups.

Le peer est l'entité logique fondamentale d'un réseau peer to peer. Un peer peut fournir des services comme effectuer une tâche et communiquer le résultat directement ou indirectement. Un peer peut prendre plusieurs profils : peer simple (edge), peer de rendezvous (rendezvous peers) et peer de routage (relay peer).

Un peergroup est un ensemble de peers regroupés autour d'un intérêt commun. Ces peers peuvent restreindre l'accès à leurs services aux membres du groupe pour diverses raisons telles que sécurité, accès restreints aux données ou surveillance d'un ensemble de peers.

Dans tout réseau peer to peer, on trouve un ensemble de mécanismes pour la transmission des données sur le réseau, c'est la couche transport et comporte trois éléments fondamentaux :

- le Endpoint, le point de la source et la destination de toute donnée transmise sur le réseau ;
- Les pipes ou canaux de communication, ce sont des liens virtuels unidirectionnels entre deux peers , un d'entrée (inputpipe) et un de sortie (outputpipe), il est possible de créer des canaux bidirectionnels en créant deux canaux unidirectionnels entre deux peers ;
- Les messages sont des données mise en forme selon le protocole utilisé et transmis par le pipe.

3. Mise en œuvre de la communication peer to peer

Dans un réseau peer to peer, tout peer est capable de découvrir les peers et les services qui constituent le réseau.

La faisabilité de cette découverte s'effectue grâce aux Advertisements : une représentation structurée d'une entité (peer, peergroup, pipe, endpoint, service,...), toute entité du réseau P2P a un advertisement lui correspondant contenant les informations importantes la concernant (type, identificateur,...). Ainsi tout peer doit pouvoir :

- traverser les firewall et les équipements NAT (Network Address Translation)

La découverte des advertisement se fait de deux façons :

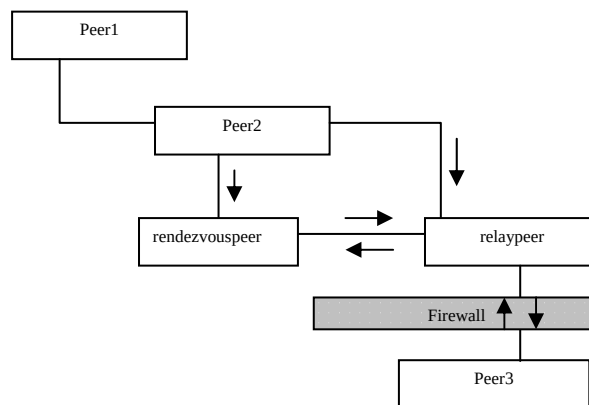


Figure 1: Un réseau P2P

-La découverte directe où tout peer doit être capable de découvrir d'autres peers sur le réseau local par diffusion en broadcast en en multicast, sans passer par un rendezvous peer.

-La découverte indirecte où le peer envoie une requête à un rendezvous peer qui localise les différents advertisements (de peers ou autres) par propagation à tous les peers qu'il connaît ou par stockage des advertisements (chaque peer peut stocker les advertisements qu'il connaît dans le but de réduire le trafic sur le réseau)

Le P2P a connu et connaît toujours un franc succès auprès du grand public grâce aux logiciels de

partage et de communication. Néanmoins, le Peer to peer se développe également auprès du monde

professionnel de par son utilisation notamment dans le calcul serveur, ce qui permet de décharger les serveurs. En revanche, le problème qui se pose avec son utilisation concerne le fait que les réseaux P2P n'ont pas été développés selon un standard. Conséquence, ces réseaux ne sont pas compatibles entre eux. C'est dans le but de résoudre ces problèmes que des plateformes dont JXTA ont vu le jour.

4. La plateforme JXTA

Basé sur des standards tels que TCP/IP, HTTP et XML, JXTA est indépendant de tout langage de programmation, de tout plateforme réseau et tout système d'exploitation. JXTA a été étudié pour permettre à des peers interconnectés de se découvrir, de communiquer facilement et s'offrir différents services.

JXTA fournit aux développeurs une structure solide permettant d'élaborer des applications peer to peer.

Pour répondre aux nombreux objectifs tels que les standards et l'interopérabilité, l'architecture JXTA est basée sur une suite de protocoles, indépendamment de tout langage, chacun apportant ses services spécifiques.

Plusieurs implémentations de JXTA existent, la plus avancée étant celle en Java (donc « multiplateforme ») qui utilise XML pour l'échange des données, ce qui fournit donc une grande interopérabilité.

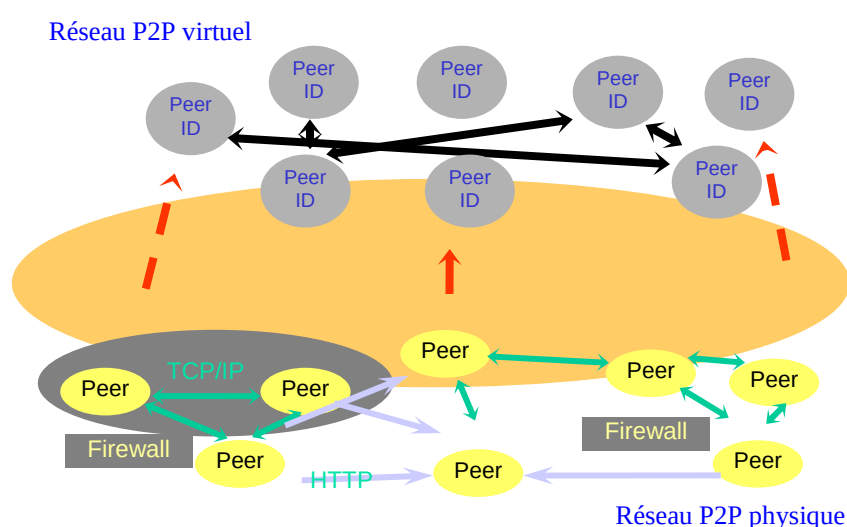


Figure 2: Un réseau JXTA au-dessus d'un réseau physique

Pour répondre aux nombreux objectifs tels que les standards et l'interopérabilité, l'architecture JXTA repose sur une suite de protocoles, indépendamment de tout langage, chacun apportant ses services spécifiques.

4.1 Manipulation des protocoles et des services

JXTA est une spécification d'un ensemble de 6 protocoles, qui permettent le partage, la communication et la collaboration entre les peers. Les 6 protocoles sont indépendants et on n'est pas tenu de les implémenter tous.

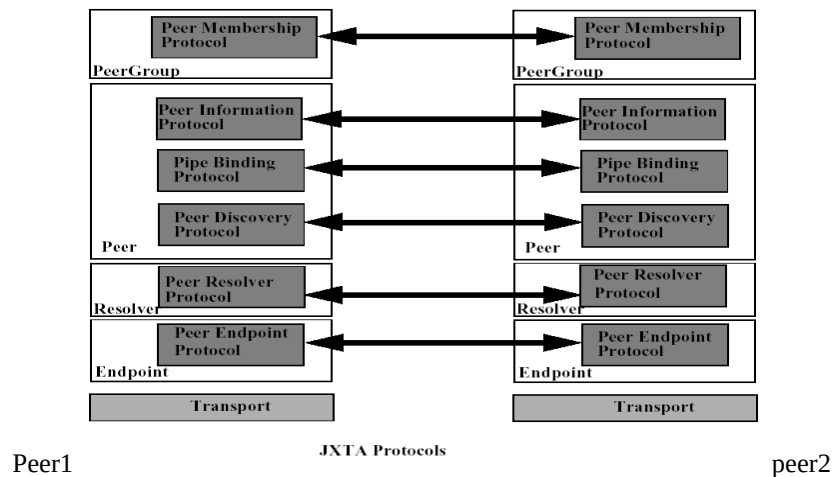


Figure 3: Pile de protocoles de la plateforme JXTA

Chaque protocole est mis en œuvre de chaque côté de la communication par un service, qui offre les primitives nécessaires à la réalisation de ce protocole. Ces protocoles sont basés sur l'échange de messages XML.

Comme dans toute pile de protocoles, il y a différents niveaux : un niveau transport et un niveau application.

Chaque protocole de la pile JXTA encapsule les protocoles de niveaux inférieurs et permettent l'abstraction physique de la machine lors de l'utilisation du protocole de niveau transport et l'abstraction du transport lors de l'utilisation des protocoles de niveaux applicatifs.

Chacun des protocoles de la figure 3 est mis en œuvre de chaque côté de la communication par les services associés. On a ainsi 6 services dans le noyau JXTA, à chacun de ces services correspond une interface fournie par JXTA.

- **Peer Endpoint Protocol (PEP)** : utilisé pour trouver une route entre deux peers , ce protocole s'appuie sur la couche transport du réseau sur lequel le peer se trouve. Le Peer Endpoint Protocole permet d'envoyer des messages entre deux peers sur des réseaux physiques différents. Deux interfaces **EndpointListner** et **Messenger**, fournies par JXTA, permettent la mise en œuvre du Endpoint Service.
- **Peer Resolver Protocol (PRP)** : utilisé pour envoyer une requête à un nombre indéterminé de peers et recevoir une réponse. Les messages utilisés dans le Peer Resolver Protocole sont des messages XML de deux types, le Resolver Query message (requêtes) implémenté par la classe JXTA **ResolverQuery** et le Resolver Response Message (réponses à des requêtes) implémenté par la classe **ResolverResponse**.
- **Peer Discovery Protocol (PDP)** : utilisé pour annoncer et découvrir du contenu. Ce protocole fonctionne comme une instance du Peer Resolver Protocol, il s'appuie sur deux types de messages : le **Discovery Query** Message au format XML contient le type d'advertisement à découvrir, et est implémenté par la classe **DiscoveryQuery**. Le discovery Response Message, au format XML, contient les advertisements demandés et est implémenté par la classe **DiscoveryResponse**.
- **Pipe Binding Protocol (PBP)** : permet de créer un lien entre deux peers souhaitant utiliser un pipe pour communiquer. Le Pipe Binding Query Message permet de rechercher un peer connecté à un pipe et le Pipe Binding Answer Message en réponse pour peer trouvé ou non.
- **Peer Information Protocol (PIP)** : utilisé pour obtenir des informations sur les peers distants. Le Peer Info Query Message, au format XML, doit contenir deux champs

<SourcePid> identification du peer qui génère la requête et <TargetPid> identification du peer dont on recherche des informations car, lorsqu'un peer envoie un Peer Info Query Message tous les peers en relation avec lui le reçoivent; Le Peer Info Response Message, message XML pour envoyer une réponse à ces requêtes.

- Rendezvous Protocol (RVP) : utilisé pour propager les messages dans le réseau via les rendezvous peers.

4.2 Architecture logiciel de JXTA

Pour permettre à ces protocoles de travailler ensemble et de former un système complet, une architecture doit être mise en place. JXTA est basé sur un modèle en couche, ce qui lui apporte de nombreux avantages.

Chaque couche apporte ainsi ses services, permettant aux couches supérieures de les utiliser.

- couche applications: calcul distribués P2P, skype, jeux en réseaux
- couche services : indexation, synchronisation, cohérence, sécurité, présence
- couche noyau : PeerGroup , Pipe, Monitoring, metering, security

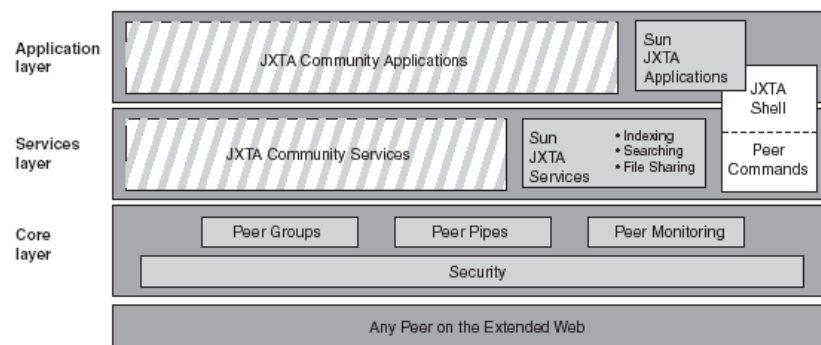


Figure 4: L'architecture en couche de JXTA

C'est dans le core layer que se situe le code implémentant les différents protocoles. Le services layer offre différents services qui utilisent les protocoles de la couche inférieure pour accomplir une tâche. L'application layer est la couche où viennent se placer les différentes applications, profitant des services de la couche précédente. Ces applications regroupent les peers et leurs offrent différentes fonctionnalités.

Le but de l'architecture en couche et des services est la réutilisation, évitant de réécrire le code à chaque développement.

4.3 Un exemple classique, le HelloWorld

L'application qui va suivre illustre la manière de lancer la plateforme JXTA. Elle va instancier cette plateforme puis afficher le nom et l'ID du peer ainsi que ceux du groupe. Cette instanciation s'effectue par l'intermédiaire de la méthode **newNetPeerGroup()** de l'objet statique PeerGroupFactory (net.jxta.peergroup). Celle-ci retourne un objet PeerGroup (net.jxta.peergroup) contenant les informations sur le groupe par défaut : le net peer group. Cet objet contient les implémentations de différents services de base de JXTA tels que le discovery, le membership ou le service de rendez vous.

Il contient également l'ID et le nom du groupe et ceux du peer sur laquelle la plateforme est exécutée.

Cette méthode signale toute erreur en lançant une `PeerGroupException` (`net.jxta.exception`). les informations souhaitées seront affichées par l'intermédiaire des méthodes correspondantes de l'objet `PeerGroup` : **`getPeerGroupName()`**, `getPeerName()`, **`getPeerGroupID()`** et **`getPeerID()`**.

La plateforme sera enfin arrêtée via la méthode **`stopApp()`**.

```
PeerGroup netPeerGroup;
try
{
    /* Création et démarrage du groupe par défaut de JXTA */
    netPeerGroup = PeerGroupFactory.newNetPeerGroup();
}
catch (PeerGroupException e)
{
    e.printStackTrace();
    System.exit(1);
}

System.out.println("Hello world "+netPeerGroup.getPeerGroupName());
System.out.println("Group ID = "+netPeerGroup.getPeerGroupID().toString());
System.out.println("Peer name = " +netPeerGroup.getPeerName());
System.out.println("Peer ID = " +netPeerGroup.getPeerID().toString());
netPeerGroup.stopApp();
```

L'exécution donnerait par exemple pour un peer nommé `peerA` :

```
Hello world NetPeerGroup
Group ID = urn:jxta:jxta-NetGroup
Peer name = peerA
Peer ID = urn:jxta:uuid-59616261646162614A78746150325033F3B
```

Tout peer, peer group, pipe ou autre ressource JXTA a un identificateur de référencement unique: un JXTA ID, c'est le système d'adressage JXTA, constitué d'une chaîne de caractères unique composée de trois parties :

- un identifiant de namespace = `jxta` ;
- une spécification de format = `urn` ;
- un ID= valeur unique

A tout peer crée, il s'en suit la création de peergroups par défaut, il s'agit de `WorldPeerGroup` et `NetPeerGroup`.

Après execution, on peut constater qu'un repertoire `.jxta` a été créé. Ce répertoire contient le fichier `PlatformConfig` et un répertoire de cache nommé `cm`, dans lequel sont sauvegardés les différents advertisements et d'autres fichiers d'index.

5. Conclusion

Si JXTA est encore peu connu, , il ne fait aucun doute qu'il s'agit d'une technologie très évoluée aux multiples avantages. Cette suite de protocoles permet d'élaborer une couche réseau solide sur laquelle peut reposer tout type d'application destiné à fonctionner sur Internet. C'est pourquoi, il serait sans surpris de voir JXTA intégré dans de nombreux projets futurs, commerciaux ou non.

Après un premier développement avec l'API Java de JXTA, une constatation s'impose : le modèle de cette technologie a été soigneusement étudié, offrant de nombreuses possibilités avec une grande facilité de développement.

JXTA a de plus l'énorme avantage d'être une solution Open Source, et donc en perpétuelle évolution ...

Références

- [1] Brendon J. Wilson. JXTA. Juin 2002. Site <http://www.brendonwilson.com/projects/jxta/index.shtml>
- [2] Heukmes. A. JXTA. Janvier 2007. Site <http://2dconcept.com/jxta.html>
- [3] JXTA Doc. Site <https://jxta.dev.java.net/>
- [4] JXTA : mise en œuvre de Java. Pierre Bozonnet. 2007
- [5] Sing Li. JXTA Peer-to-Peer Computing with java. Avril 2002