# Blockseminar - Peer To Peer
# JXTA - Ein offenes Framework für den Aufbau von P2P-Netzen

LEHRSTUHL FÜR RECHNERNETZE UND TELEMATIK DER UNIVERSITÄT FREIBURG
Prof. Dr. Christian Schindelhauer

Betreuer: Arne Vater

Moritz Hartges
hartges@informatik.uni-freiburg.de

16. März 2007

**Zusammenfassung**

JXTA ist ein Projekt, das 2001 von Sun Microsystems gegründet wurde. Es ist unter Open Source-Lizenz veröffentlicht und wird zusammen mit der JXTA-Community weiterentwickelt. Das JXTA Framework ist der Versuch einen Standard für die Entwicklung von P2P-Netzwerken einzuführen. Den Entwicklern wird es ermöglicht, beim Programmieren von Anwendungen, die Kommunikationstechniken aus dem Framework zu verwenden. So entfällt die Hauptentwicklungsarbeit auf das Erstellen der eigentlichen Anwendung, ohne dass intensiv an den Strukturen für die Vernetzung der Peers und deren Kommunikation gearbeitet werden muss.
Durch einen einheitlichen Standard für P2P-Netze wird es ermöglicht, verschiedene Anwendungen auf dem gleichen Netz aufzubauen. Dadurch wird zum einen Interoperabilität gewährleistet und zum anderen wird Entwicklungsarbeit gespart.

# Inhaltsverzeichnis

# 1 Einleitung

JXTA ist ein Framework, das darauf ausgelegt ist ein Peer-to-Peer(P2P)-Netzwerk aufzubauen. Dabei stellt das Framework vorallem die verschiedenen Verbindungstechniken für die Kommunikation der Peers zur Verfügung. Wird eine P2P-Anwendung oder ein P2P-Programm erstellt, das auf JXTA basiert, so braucht sich der Programmierer nur noch wenige Gedanken über den Aufbau der Verbindung und Kommunikation der Peers zu machen. Dieses vor allem durch Sun Microsystems vorangetriebene Projekt ist der Versuch einen Standard für P2P-Applikationen zu entwickeln, um Interoperablität zwischen verschiedenen P2P-Programmen zu ermöglichen und die Programmierung solcher Programme zu vereinfachen.
2001 wurde das Projekt von SUN gegründet, geleitet wird es von Bill Joy und Mike Clary. Da JXTA unter Open-Source-Lizenz veröffentlicht ist, hat sich inzwischen eine große Comunity von Entwicklern gebildet die JXTA stetig weiterentwickeln. Treibende Kraft ist aber immer noch SUN. Der Name ist angelehnt an das englische Wort “juxtapose”, zu deutsch: Nebeneinanderstellung. Dies ist bezeichnend für die gleichwertige Hierarchie der Peers in einem P2P-Netzwerks. Laut [B+04] kann man es auch so verstehen, dass das JXTA-Projekt ein System neben der bewährten Client-Server-Architektur schaffen wollte, ohne diese zu verdrängen.
JXTA ist unabhängig vom verwendeten Netzwerk und Endgerät sowie von der verwendeten Plattform. Es lässt sich im Internet, LAN, WLAN aber auch beispielsweise mit Bluetooth-Verbindungen nutzen. Nicht nur klassische Computer mit den verschiedenen Betriebssystemen können einen Peer bilden, sondern auch PDAs, Handys oder Sensoren die vernetzt werden. Außerdem kann das Framework mit verschiedenen Programmiersprachen verwendet werden. Auf diese Vielfältigkeit wurde bei der Entstehung von JXTA großen Wert gelegt, weil nur so eine breite Anwendung möglich ist. Auf der Webseite des Projekts finden sich Bibliotheken für C, C++, C# und Java sowie Java ME.

# 2 Warum Peer-to-Peer?

Das klassische Internet basiert auf einer hierarchischen Verteilung der Aufgaben. Dabei beantwortet ein Server die Anfragen der einzelnen Clients, wie das Aufrufen einer Webseite. Seit einigen Jahren haben sich Programme mit eigenem P2P-Netz etabliert, die gewisse Aufgaben dezentral verteilen. So wird bei *Tauschbörsen* ein Netz zwischen allen Rechnern gesponnen, die mit dem entsprechenden Programm gerade im Internet sind. Dadurch wird eine sehr große Menge an Dateien und hohe Bandbreite zur Verfügung gestellt. Es entfällt ein zentraler Server der Unmengen von Anfragen bearbeiten muss, riesige Datenmengen zu speichern hat und somit hohe Kosten verursacht.
Diese, im grauen Bereich des Urheberrechts groß gewordene, Technik möchte man heute auch für kommerzielle und professionelle Programme nutzen, um Rechenleistung, Speicher, Transfervolumen und Verbindungskosten zu sparen und eine hohe Ausfallsicherheit zu ermöglichen. Bei zentralen Systemen genügt die (zeitweise) Überlastung des Servers um ein System lahm zu legen. Bei P2P-Netzen übernimmt jeder Peer sowohl Client- als auch Server-Tätigkeiten. Alle Anforderungen an das Netzwerk werden gleichmäßig auf die Peers verteilt. Fallen nun ein oder mehrere Peers aus, so werden deren Funktionen von den verbliebenen Peers übernommen. Durch diese Gleichverteilung der Aufgaben können P2P-Netze wesentlich Leistungsfähiger sein.

# 3 Konzepte

Vor JXTA gab es viele verschiedene P2P-Netzwerke. Da JXTA einen Standard schaffen möchte, definiert es abstrakt einen Satz von Protokollen und XML-Formaten. Auf diese Definitionen können P2P-Anwendungen zurückgreifen. Somit besteht eine fundierte Basis für die Entwicklung von P2P-Anwendungen. Gleichzeitig garantiert man durch die gleiche Basis eine hohe Interoperablität der verschiedenen Programme.
Im folgenden werden die Grundbestandteile eines P2P-Netzwerks erklärt und es wird auf ihre Funktion in JXTA eingegangen.

## 3.1 Peer

Als *Peer* bezeichnet das JXTA-Projekt jedes Gerät auf dem eine oder mehrere JXTA-Protokolle installiert sind. In [Geh04] wird darauf hingewiesen, dass dies nicht ganz richtig ist. Sind etwa zwei JXTA-Anwendungen auf einem Computer installiert, so stellen sie zwei Peers dar. [Geh04]:“Im Kern geht es bei dem Begriff des Gerätes um die Implementiereung der Protokolle und in diesem Sinne ist dann ein Programm ein Gerät.” Es können auch mehrere Computer ein Peer sein, um weniger Fehleranfällig zu sein oder große Lasten verteilen zu können. In [Wil02] wird ein Peer als “eine Einheit in einem P2P-Netz, die in der Lage ist nützliche Arbeit für das P2P-Netz zu verrichten und die Ergebnisse dieser Arbeit zu anderen Peers zu kommunizieren, direkt oder indirekt” definiert. Was nützliche Arbeit ist ist dabei nicht genauer spezifiziert.
Bei JXTA gibt es drei verschiedenen Arten von Peers, die verschiedene Rollen übernehmen. Die einfachste Art ist der *Simple Peer*. Er ist die Schnittstelle zum Benutzer und stellt eine Oberfläche bereit, die es ermöglicht *Services* in Anspruch zu nehmen und bereit zu stellen. Die bereitgestellten Services bestimmen dann auch, welche Arbeit in dem P2P-Netz erledigt wird.
Ein Problem von P2P-Netzen sind Netzwerkhürden, wie Firewalls, NAT[1] und Proxies. Da ein Simple Peer oft durch solche Hürden vom Internet abgeschnitten ist, übernimmt er in einem P2P-Netz die wenigste Verantwortung. So ist gewährleistet, dass durch das durch den Ausfall (die nicht Erreichbarkeit) des Peers das Netz nicht beeinträchtigt wird. Damit Peers hinter Netzwerkhürden trotzdem Verbindung zu anderen Peers aufnehmen können sind in JXTA wirkungsvolle Techniken zum Umgehen dieser Hürden implementiert.
Der *Rendezvous Peer* übernimmt die Aufgabe beim Auffinden von Ressourcen in einem Netzwerk zu helfen. Das Finden von Services und *Advertisements* nennt man *Discovery*. Der Rendezvous Peer sammelt und verwaltet Advertisements und stellt diese zur Verfügung. Dadurch wird das P2P-Netz entlastet, weil die Latenzzeiten verkürzt werden und so die Suche nach einem gewünschten Peer schneller von statten geht. Sinnvollerweise finden sich Rendezvous Peers im öffentlichen Internet um gut erreichbar zu sein.
Der dritte Peertyp ist der *Routing Peer*. Er befindet sich am Übergang zwischen Peers, die durch Netzwerkhürden abgeschirmt sind, und dem übrigen P2P-Netz. Der Routing Peer hält dabei Informationen vor, wie bestimmte Peers zu erreichen sind. Für nicht erreichbare Peers speichert der Routing Peer die Anfragen zwischen, so dass diese sie bei ihm abgefragt werden können. Damit ist der Routing Peer ein wirksames Mittel die Netzwerkhürden zu umgehen. Durch die auf den Routing Peers gespeicherten Routen wird die Netzlast auch weiter reduziert, weil nicht immer wieder neue Roten gesucht werden müssen, sondern bereits bekannte Routen bei einem Routing Peer angefragt werden können.

[1] Network Adress Translation

## 3.2 Peer Group

Eine *Peer Group* ist ein Zusammenschluss von Peers zu einer Gruppe. Damit ist die Peer Group das Mittel zur Organisation und Unterteilung eines P2P-Netzes. Dadurch kann Sicherheit, Redundanz und Lastausgleich geschaffen werden. Eine Gruppe bestimmt selbst, welche Services sie anbietet und somit, welche nützliche Arbeit sie verrichtet. Weiter bestimmt eine Gruppe, ob nur für Gruppenmitglieder oder auch für andere auf ihrer Ressourcen zugreifen können. Wichtig ist noch, das IDs der Peers nur gruppenweit eindeutig sein müssen (ähnlich der IPs in lokalen Netzwerken). Ein Peer kann gleichzeitig mehreren Gruppen angehören.

## 3.3 Services

Services sind allgemein die Dinge, weswegen sich Peers überhaupt vernetzen. Als Service wird alles bezeichnet was in einem P2P-Netz angeboten wird. Das kann alles sein was jemand als sinnvoll erachtet anzubieten oder in Anspruch zu nehmen, Dateien, Webservices, Rechnerressourcen, etc. Grundlegende Netzwerkfunktionen (z.B. Suche) sind allerdings schon in der JXTA *Core Services* (Abs. 4.1) implementiert und müssen nicht extra angeboten werden.

## 3.4 Advertisements

Advertisements sind XML-Dokumente, die alle Ressourcen eines Peers beschreiben. Die XML-Dateien müssen wohlgeformt sein, sonst gibt es keine Standardanforderungen, da die Beschreibungen oft sehr umfangreich und variabel sind. Durch den XML-Standard wird Plattformunabhängigkeit gewährleistet. In [jxt07] sind sieben verschiedene Advertisements beschrieben: Peer Advertisements, Peer Group Advertisements, Module Class Advertisements, Module Specification Advertisements, Module Implementation Advertisements, Pipe Advertisement und Rendezvous Advertisement. Die JXTA-Protokolle basieren dabei maßgeblich auf den ersten 5 Advertisements.
Advertisements werden mit einem Ablaufdatum veröffentlicht, die die Erreichbarkeit einer Ressource spezifizieren. Um das Ablaufen eines Advertisements zu verhindern, kann ein dieses erneut veröffentlicht werden.

## 3.5 Pipes und Endpoints

Pipes sind die eigentlichen Kommunikationskanäle eines P2P-Netzwerks. Dabei ist ein Pipe eine virtuelle, unidirektionale Verbindung von Peers. Am empfangenden Ende wird die Pipe als Input Pipe und am sendenden Ende als Output Pipt bezeichnet. Für bidirektionale Kanäle werden zwei entgegengesetzte Pipes eingerichtet, die aber nicht die gleich Route haben müssen. Über die Pipes werden sämtliche Daten, bei JXTA exemplarisch *Messages* genannt, ausgetauscht. Da bei JXTA alles als Message über Pipes versendet wird, ist das Vorhandensein von Endpoints garantiert. Pipes gehen immer von Endpoint des Startpeers zum Endpoint des Zielpeers. Dabei muss die Verbindung nicht direkt vom einen Peer zum anderen gehen, sondern kann sich über mehrere anderere Peers erstrecken. Meistens ist ein Endpoint eine Netzwerkschnittstelle, es kann aber auch z.B eine URL sein. Router Peers sind die Endpoints oft http-URLs, da http meistens auch aus abgesicherten Netzwerken heraus möglich ist.

# 4 Architektur von JXTA

Da man in einem P2P-Netz auf die zentralen Server verzichtet, die die Vermittlung und Kommunikation zwischen den verschiedenen Rechnern steuern, muss man dafür sorgen, dass die Peers sich selber in einem Netz organisieren um Aufgaben zu übernehmen und Leistungen anzufragen. Dabei müssen einige grundlegende Dinge beachtet werden.

- Aufinden von Peers und dezentralen Ressourcen im Netzwerk[2]
- Austauschen von Dokumenten mit beliebigen Peers
- Gruppenverwaltung von Peers und "Virtuellen Räumen"
- Sichere Komunikation von Peer zu Peer
- Authentifizierungsmechanismen
- Firewalls und NAT

All dieser Probleme nimmt sich JXTA an und möchte Standards schaffen. Das Framework übernimmt die Arbeit die P2P-Netzwerkstruktur zu gewährleisten, so dass die darauf aufbauenden Anwendungen nur die eigentliche Funktionalität implementieren müssen. Nur durch Standards ist es möglich konstant gute Qualität von Anwendungen zu gewährleisten, die auch prüfbar ist. JXTA ist in einem Schichtenmodell (Layers) organisiert (Abb. 1). Die Basis bildet das Netzwerkkommunikationsprotokoll (z.B. TCP/IP). Die darauf aufbauende Schicht (JXTA core) organisiert die sichere Kommunikation und es werden Funktionen für die Kommunikation zwischen den Peers breitgestellt. Außerdem werden in dieser Schicht kooperierende Peers in Peer Groups verwaltet. In den Höheren Schichten werden die Grundfunktionen zu Services zusammengefasst, die dann von Applicationen genutzt werden können.

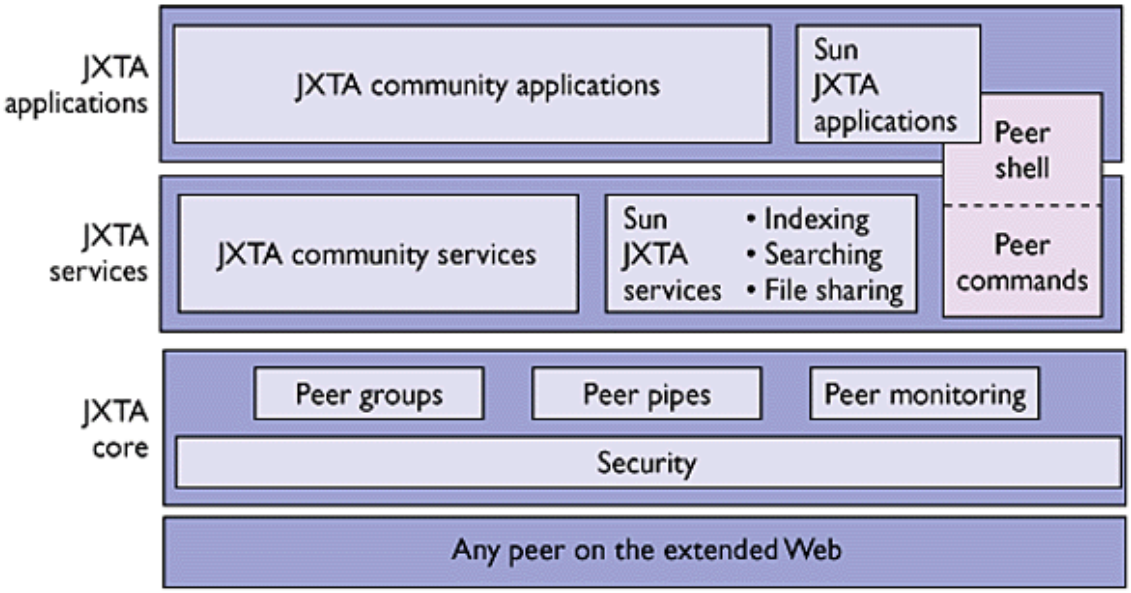


Abbildung 1: Die JXTA P2P-Softwarearchitektur, aus [Wik06]

[2] [Geh04]

## 4.1 JXTA Core Layer

Der Core Layer bildet die Schicht des virtuellen Netzwerks und stellt grundlegende Funktionen bereit, die für jede P2P-Anwendung gleich sind. Darin enthalten sind die Konzepte der Peers und Peer Groups und Funktionen wie der Discovery-Service, der das gegenseitige Finden der Peers ermöglicht. Weiter ist hier der Mechanismus implementiert, der die Kommunikation zwischen den Peers ermöglicht. Dies geschieht über Peer Pipes, unidirektionale Kanäle zwischen den Peers.
Im Core Layer werden die Methoden zur Verfügung gestellt, Gruppenmitgliedschaften abzulehnen oder anzunehmen. Die weitere Implementierung (z.B. Zutrittsbedingungen) wird in den einzelnen Anwendungen realisiert.
Das *Peer Monitoring* ermöglicht das Entwickeln von Managementfunktionen in einer Peer Group. Hier werden z.B. Bandbreiten- und Trafikbeschränkungen und Prioritäten gewisser Anfragen und Aktionen festgelegt.
Somit sind in der Core nur die essentiellen Services bereitgestellt. Diese Core Services bilden dann die Grundlage für die nächste Schicht.

## 4.2 JXTA Service Layer

Die Funkionen der zweiten Schicht werden von SUN und der JXTA Communitiy gemeinsam entwickelt und gepflegt. In dieser Service Schicht werden mit Hilfe der Core Services komplexere Services realisiert. Durch das Zusammenfassen der Fähigkeiten der Core werden Bausteine geschaffen, die als Basis für die Entwicklung von Anwendungen bereitstehen. Diese Bausteine sind vergleichbar mit klassischen Programmbibliotheken. So stellt die Schicht die JXTA-Suchfuntion, einfache File-Transfer-Services, Indexierung und Services für Grid-Computing zur Verfügung. Eine effizientere Suchfunktion wird meist in der Application Schicht implementiert.
Konkrete Anwendungen können die gleichen Services für verschiedene Ziele verwenden. So kann z.B. der Service für Grid-Computing verwendent werden um Primzahlen zu berechnen aber auch eine Anwendung ähnlich der Seti@Home[3] Idee findet hier das Management von Ressourcen.

## 4.3 JXTA Application Layer

In der dritten Schicht werden nun die eigentlichen Anwendungen zusammengefasst, die der Anwender zu sehen bekommt. Sie bauen auf die Fähigkeiten der Core und der Service Schicht auf. Es gibt bereits eine Reihe von Applikationen die von SUN und dem JXTA-Projekt entwickelt wurden, darunter P2P-Chat und Instant-Massaging, P2P-Groupware, Grid-Computing, Filesharing und P2P-Mailsysteme.
Eine Besonderheit zwischen der zweiten und der dritten Schicht ist die *Peer Shell*, die von SUN entwickelt wurde. Sie stellt dem Benutzer eine minimale konsolenbasierte Schnittstelle zur Verfügung, mit der man über die *Peer Commands* direkt auf JXTA-Services zugreifen kann. Die Grenze zwischen Service und Application Schicht ist nicht sehr scharf, da Applikationen auch andere Applikationen als Services verwenden können.

[3] `seti.alien.de`

# 5 Protokolle

In JXTA gibt es sechs verschiedene Protokolle, mit denen die Peers kommunizieren. Laut [jxt07] sind das Peer Resolver Protocol und das Endpoint Routing Protocol in der JXTA-Core angesiedelt, während das Peer Discovery Protocol, das Rendezvous Protocol, das Peer Information Protocol und das Pipe Binding Protocol zu den JXTA-Services gehören.
In folgenden Abschnitt werden die einzelnen Protokolle beschrieben.

## 5.1 Reer Resolver Protocol (PRP)

Dies ist das Protokoll, das sämtliche Messages der Peers im allgemeinen Format versendet. Es ist definiert, dass die Kommunikation per request / response Verfahren stattfindet, dazu beinhaltet das PRP zwei Message-Typen.

- Die Resolver Query Message ist zum senden einer Anfrage
- Die Resolver Response Message ist zum senden einer Antwort

Ein Peer sendet also eine Resolver Query Message an alle ihm bekannten Peers. Erhält ein normaler Peer die Anfrage, schaut er bei sich nach einem registrierten Resolver Handler, der überprüft ob die Anfrage gültig oder ungültig ist. Im ersten Fall wird eine passende Resolver Response Message an den Absender der Anfrage geschickt, ansonsten wird sie nicht weiter behandelt.
Erhält ein Rendezvous Peer die Anfrage verfährt er zunächst genauso wie ein normaler Peer, nach dem Senden der Antwort wird die Anfrage aber noch an alle dem Rendezvous Peer bekannten Peers weitergeleitet. Die antworten dann direkt dem Absender der Anfrage.

## 5.2 Endpoint Routing Protocol (ERP)

Das ERP wird gebraucht um auch Nachrichten an Peers zu versenden, die nicht direkt verbunden sind, weil sie in verschiedenen Netzwerken sind oder hinter Firewalls. Das ERP hat drei Message-Typen.

- Route Query Message, um die Peers festzustellen, über die eine Message an die gewünschte Endpoint Adresse versendet werden kann.
- Route Response Message, um auf eine Route Query Message zu antworten.
- Endpoint Router Message, die die Informationen enthält, um eine Message an ihr Ziel zu bringen.

Ist ein Peer nicht direkt mit dem Peer verbunden, an den eine Anfrage gesendet werden soll, sendet er eine Route Query Message an alle ihm bekannten Peers. Kennt ein Peer den Weg zu dem gewünschten Peer, sendet er eine Route Response Message an den ersten Peer zurück, die die Route zum Ziel-Peer enthält. Der erste Peer hängt die Endpoint Router Message, die die Routeninformationen enthält, an seine Nachricht an und sendet an den ersten in der Routingliste genannten Peer, der die Nachricht an den zweiten der Liste weitersendet, bis die Nachricht am Ziel angekommen ist.

## 5.3 Peer Discovery Protocol (PDP)

Das PDP legt dar, wie ein Peer andere Peers, Peer Groups, Services und Pipes finden kann. Es ist also der Mechanismus, der Informationen über das P2P-Netzwerk sammelt. Dazu sind zwei Message-Typen definiert.

- Die Discovery Query Message, um Advertisements zu finden.
- Die Discovery Response Message, um auf die Anfrage zu antworten.

Zum Finden anderer Peers sendet ein Peer eine Discovery Query Message an alle ihm bekannten Peers. Erhält ein normaler die Message, sucht er in seinem Cache nach passenden Advertisements und sendet sie mit der Discovery Response Message direkt an den anfragenden Peer.
Erhält ein Rendezvous Peer die Message, handelt er wie der normale Peer und gibt die Message zusätzlich an alle ihm bekannten Peers weiter. Diese können dann wiederum direkt dem anfragenden Peer antworten.

## 5.4 Rendezvous Protocol (RP)

Mit dem RP wird die Verbindung eins Peers mit einem Rendezvous Peer gemanagt. Um einen Redezvouz Peer benutzen zu können, muss sich ein Peer bei ihm ein Lease holen. Dazu werden drei Message-Typen benötigt:

- Die Lease Request Message stellt die Anfrage um Erteilung eines Lease
- Die Lease Garanted Message, um einen Lease zuteilen und die Dauer der Gültigkeit zu bestimmen
- Die Lease Cancel Message, mit der die Verbindung zu einem Rendezvous Peer abgebrochen wird

Damit ein Peer als Rendezvous Peer arbeiten kann, muss er seine Leistungsfähigkeit mittels des *Rendezvous Advertisements* offen legen.

Listing 1: Rendezvous Advertisement

```
<xs:element name="RdvAdvertisement" type="jxta:RdvAdvertisement"/>

<xs:complexType name="RdvAdvertisement">
    <xs:sequence>
        <xs:element name="RdvGroupId" type="jxta:JXTAID" />
        <xs:element name="RdvPeerId" type="jxta:JXTAID" />
        <xs:element name="RdvServiceName" type="xs:string" />
        <xs:element name="Name" type="xs:string" />
        <xs:element name="RdvRoute" type="jxta:RA" minOccurs="0" />
    </xs:sequence>
</xs:complexType>
```

Möchte sich ein Peer mit einem Rendezvous Peer verbinden, so sendet er eine Lease Request Message an den Rendezvous Peer. Dieser prüft nach Erhalt, ob er ein Lease erteilen will. Im positiven Fall sendet er eine Lease Garant Message an den anfragenden Peer.
Verwendet nun der Peer den Rendezvous Peer für eine Anfrage, wird von diesem zunächst geprüft, ob er dem Peer ein Lease zugeteilt hat. Ist das der Fall, bearbeitet er die Anfrage und

sendet sie an alle anderen Peers weiter, die ein Lease von ihm haben.
Wenn ein Peer einen bestimmten Rendezvous Peer nicht mehr verwenden möchte, kann er sich bei dem Rendezvous Peer mit der Lease Cancel Message abmelden und dieser löscht ihn aus der Liste seiner verbundenen Peers.

## 5.5 Peer Information Protocol (PIP)

Mit dem PIP kann man den Status eines entfernten Peers abfragen. Dieses optionale Protokoll erlaubt es einen anderen Peer zu beaufsichtigen, dies kann beim Filesharing vorteilhaft sein, um einen Peer effizienter nutzen zu können. Das PIP hat zwei Message-Typen:

- Die Peer Info Query Message, für die Anfrage
- Die Peer Info Response Message, für die Antwort

Bekommt ein Peer eine Peer Info Query Message, überprüft er zunächst, ob die Peer ID, an die die Message adressiert ist, mit der eigenen lokalen ID übereinstimmt. Ist das der Fall, antwortet er mit der Peer Info Response Message.

## 5.6 Pipe Binding Protocol (PBP)

Pipes sind die Kanäle über die Peers Daten zu anderen Peers senden und empfangen. Um Pipes benutzen zu können, müssen sie an einen physikalischen Endpoint angebunden werden. Der Ablauf dazu wird im PBP beschrieben. Es werden zwei Message-Typen gebraucht:

- Die Pipe Binding Query Message, um eine Anfrage an einen entfernten Peer zu senden.
- Die Pipe Binding Answer Message, um eine Antwort auf die Anfrage zu senden.

Wie in Abs. 3 erklärt, wird eine Pipe durch ein Pipe Advertisement beschrieben.
Um nun über Pipes Nachrichten verschicken zu können, muss ein Peer eine Input Pipe mit einem bestimmten Pipe Advertisement generieren. Anschließend kann er auf dieser Pipe auf Nachrichten warten. Möchte nun ein zweiter Peer etwas über diese Pipe versenden, muss er eine Output Pipe zu eben dem Pipe Advertisement generieren. Hierfür sendet er eine Pipe Binding Query Message an alle ihm bekannten Peers.
Erhält ein Peer eine Pipe Binding Query Message, überprüft er in seinem Pool von Pipes, ob er an eine passende Input Pipe gebunden ist. Ist das der Fall, sendet er eine Pipe Binding Answer Message mit seinem Peer Advertisement an den Absender. Dieser kann nun die Endpoint-Informationen aus dem Peer Advertisement herauslesen, eine Output Pipe generieren und Nachrichten versenden.
Für bidirektionale Kommunikation muss das Verfahren dann noch mal umgekehrt durchgeführt werden, so dass zwei Pipes in entgegengesetzte Richtung verlaufen.

# 6 Sicherheit

Die Frage der Sicherheit spielt natürlich gerade in P2P-Netzen eine große Rolle. Man möchte verhindern, dass vertrauliche Daten, die über das Netz versendet werden, von anderen mitgelesen werden können. Zum anderen möchte man Peers die einer Gruppe beitreten authentifizieren. Zu diesem Zweck sind in JXTA, die aus dem Internet bekannten, Verfahren zur sicheren Kommunikation enthalten.

## 6.1 Transport Layer Security (TLS)

In JXTA wurde der Standard der Internet Engineering Task Force Networking Group übernommen. TSL[4] ist für die gesicherte Übertragung von Daten, mit einem Protokoll wie TCP, zuständig. Da TSL direkt in der Transportschicht des OSI-Modells[5] auf dem Transportprotokoll aufbaut, ist auch die Implementierung in JXTA kein Problem.

## 6.2 Public Key Infrastructure

Dieses Verfahren kann zum einen für die Verschlüsselung von zu versendenden Daten, zum anderen zur Signatur von Nachrichten oder Peers, und somit zur Authetifizierung, genutzt werden. Hierbei macht man sich die asymmetrische Verschlüsselung zu Nutze. Schlüssel werden immer im Paar von einem *öffentlichen Schlüssel* und einem *privaten Schlüssel* erzeugt. Daten, die mit dem öffentlichen Schlüssel verschlüsselt werden, lassen sich nur mit dem privaten Schlüssel entschlüsseln. Wie der Name schon sagt, gibt man den öffentlichen Schlüssel weiter und behält den privaten Schlüssel für sich.
Möchte ein Peer A einem Peer B nun Daten zukommen lassen, die nur Peer B lesen kann, so verwendet er den öffentlichen Schlüssel von Peer B. Die verschlüsselten Daten erreichen dann Peer B, der sie mit seinem privaten Schlüssel entschlüsselt.
Möchte ein Peer seine Authentizität beweisen, signiert er mit seinem privaten Schlüssel eine Nachricht. Jetzt kann der Empfänger über den öffentlichen Schlüssel feststellen, dass die Nachricht sicher von dem Peer stammt, welcher die Nachricht verschickt hat.
Die Verwaltung der öffentlichen Schlüssel kann entweder über einem Dienstleister für *Certificate Authority* (CA) übernommen werden oder von einem (gruppen-)eigenen CA. Gerde in P2P-Netzen macht es Sinn, dass die Gruppen eigene CA bestimmen.
Möchte ein Peer in eine Gruppe aufgenommen werden, sendet er eine *Ceritficate Sign Request* an den ausgewiesenen Peer Group CA der Gruppe. Dieser Request gibt die Identität des Peers und seine gewünschten Rechte für die Gruppe an. Der Peer Group CA bestätigt den Peer, indem er ihm ein von ihm signiertes Zertifikat zurücksendet. Dieses Zertifikat verwendet der Peer für die Kommunikation innerhalb der Gruppe und weist sich auf diese Weise als Mitglied aus.

# 7 Anwendung und Ausblick

Das JXTA auch regen Einsatz findet, zeigen die verschiedenen Programme auf der Webseite des Projekts. Sun führt in seinem Artikel [Mic03] noch weitere Systeme auf, die in Verwendung sind.
Ein Verband von Tankstellenpächtern, die National Association of Convenience Stores (NACS), hat ein offenes Netzwerk gegründet, das verschiedene Sensoren vernetzt. Man hat dabei die Technik von JXTA eingesetzt um ein kosteneffizientes, dezentrales und ausfallsicheres Netzwerk zu schaffen. Geräte wie Füllstandsensoren, Benzinpumpen, Kassenterminals und Steuerungsanlagen verschiedener Hersteller lassen sich vernetzten.
Ein anderes Projekt, das Sun beschreibt, ist das Netzwerk der National Association of Realtors, ein Immobilienmaklerverband in den USA. Hier vernetzen die einzelnen Makler ihre Datenbanken, so dass sie aktuelle Angebote aus jeder Region des Landes abrufen können. Jeder Makler verwaltet dabei seine eigenen Datenbanken dezentral und stellt sie über das JXTA-Netzwerk

[4] für eine genaue Erklärung des TLS Verfahrens siehe: `http://de.wikipedia.org/wiki/Transport_Layer_Security`

[5] `http://de.wikipedia.org/wiki/OSI-Modell`

den Kollegen zur Verfügung. Das hat den Vorteil, dass kein Makler seine Daten aus der Hand geben muss und er nur veröffentlicht was er möchte. Des weiteren braucht man keine zentrale Administration um den Datenbestand aktuell zu halten, sondern die aktuellen Daten der Makler sind auch sofort für die anderen Teilnehmer des Netzwerks verfügbar.
Zwei Groupware-Lösungen, basierend auf JXTA sind in [Mic03] ebenfalls beschrieben:

- *Momentum* von InView Software ist ein System für den gemeinsamen Zugriff und die Bearbeitung von technischen Zeichnungen, Flussdiagrammen, Netzwerkplänen, Organigrammen und Ablaufschemata. Clients sind dabei für Linux, Windows und Solaris vorhanden.
- *IAM-Developing* von Internet Access Methods ist eine Lösung, die es mehreren Programmierern erlaubt, gemeinsam an dem gleichen Sourcecode zu arbeiten. Dabei kann ein Programmier in Echtzeit die Änderungen seiner Mitarbeiter nachverflogen.

Auf der Seite `http://www.jxta.org/servlets/ProjectList` findet man viele aktive Projekte für verschiedene Programme. Zum Beispiel:

- *gameplatform*, ein Ansatz, Multiplayer-Spiele über ein P2P-Netz zu spielen. Vorteil: Je mehr Mitspieler, desto mehr Hardwareressourcen im Netz und somit gute Skalierbarkeit.
- *go*, eine Turnierlösung für das bekannte Brettspiel Go.
- *jnd*, Filesharing mit einem Ansatz ähnlich wie bei Bittorrent.
- *juxtaprose*, ein selbstorganisierendes Diskussions-Netzwerk, dass gerade für das Finden und diskutieren über neue Services für JXTA genutzt werden soll.
- *tradingcenter*, ist ein kostenloses Auktionshaus.
- *trinytalk*, ist eine Voice over Peer-to-Peer Software.
- *venezia-gondola*, bietet die Möglichkeit über ein P2P-Netz einen Garagenflohmarkt, einen eigenen kleinen Onlinestore, zu erstellen.

All diese Programme zeigen, dass JXTA rege eingesetzt wird. Bisher ist noch kein anderes System so ausgereift um einen Standard für P2P zu schaffen. Somit sind die Chancen gut, dass sich JXTA durchsetzen wird. Sun betont noch einmal in [Mic03], dass es bei JXTA "nicht um den völligen Verzicht auf Server oder auf jede Form von Zentralisierung geht, sondern vielmehr darum, den Zugriff auf Ressourcen im gesamten Netzwerk zu ermöglichen oder zu vereinfachen."
Die große Community und die vielen offenen Projekte ermöglichen eine gute Weiterentwicklung und Anpassung an aktuelle Technologien und Ansprüche. So kann JXTA für viele neue Entwicklungen die Basis liefern. Auch für den mobilen Markt mit Kleingeräten wie Handy und PDA ist JXTA gerüstet und die verschieden Verbindungstechniken wie LAN und Bluetooth sind berücksichtigt, so dass sich JXTA nicht nur auf "normalen" Computern zuhause fühlt.
Wegen der großen Nachfrage nach P2P-Lösungen könnte JXTA in Zukunft immer wichtiger werden. Vorteile von JXTA sind dabei, dass verschiedene Programme in einem Netz vereint und so Ressourcen gespart werden können. Die Technologie ist unabhängig vom Betriebsystem, der verwendeten Programmiersprache und dem zugrunde liegenden Netzwerk einsetzbar. In der Spezifikation wird vieles offen gelassen, damit JXTA als Standard nicht limitierend wirkt.

Bleibt zu hoffen, dass Sun und die JXTA-Community das Framework im Interesse der Anwender weiterentwickeln und nicht eine Reihe von “illegalen Tauschbörsen” die Technologie in Verruf bringen. P2P-Netzwerke eine wichtige Form der Vernetzung, “Das zugehörige politische Äquivalent zu P2P-Netzwerken ist vielleicht die Demokratie, in der jeder Beteiligte gleiche Rechte und Pflichten hat.” [SM06]

# Literatur

[B+04] Robert Becker et al. Jxta: Einführung und Überblick. www.ag-nbi.de/lehre/04/S_P2PNET/Ausarbeitungen/JXTA.pdf, 2004.

[Bed03] Bruno Bedin. Jxta - eine p2p technik für java. www.it-academy.cc, 2003.

[Geh04] Nick Gehrke. *Peer-to-Peer Applikationen für eletonische Märkte*. DUV, 2004.

[jxt] jxta.org. `www.jxta.org`.

[jxt07] jxta.org. http://spec.jxta.org/nonav/v1.0/docbook/JXTAProtocols.html, 2007.

[Mic03] SUN Micosystems. Mehr als tauschbörsen. de.sun.com/homepage/feature/2003/jxta/#findoutmore, 2003.

[SM06] Christian Schindelhauer and Peter Mahlmann. *Peer-to-Peer-Netzwerke*. Springer, 2006.

[Wik06] Wikipedia. Jxta. de.wikipedia.org/wiki/JXTA, 2006.

[Wil02] Brendon J. Wilson. *JXTA*. New Riders, 2002.