Unter vier Augen

Peer to Peer in Theorie und Praxis

Ekaterina Chtcherbina, Ludwig Mittermeier, Roy Oberhauser und Thomas Wieland

Trotz oder wegen RMI und CORBA funktionieren die meisten verteilten Systeme auch heute noch über einen zentralistischen Ansatz mit mehr oder weniger klar definierten Client- und Server-Rollen. Solange in einem vernetzten System die Konfiguration und damit die Rollenverteilung von Diensten und Anwendungen weitgehend invariant bleibt, ist dieses Vorgehen unproblematisch. Ist dies allerdings nicht der Fall, stoßen zentralistische Architekturen rasch an ihre Grenzen. Etwa dann, wenn das Internet und mobile Geräte ins Spiel kommen. Mit Peer-to-Peer-Technologien wie JXTA lassen sich beliebige Anwendungen, Teilnehmer und Dienste dezentral zu gemeinsamen virtuellen Welten zusammenschließen. Das häufig zitierte Beispiel Napster zeigt dabei aber nur die Spitze der Anwendungsmöglichkeiten.

Was also ist dran am Peer-to-Peer Computing? Der nachfolgende Artikel besteht aus zwei Teilen, die Sie sowohl in die theoretischen als auch in die praktischen Aspekte der Peer-to-Peer-Technologie einführen wollen. Ekaterina Chtcherbina und Thomas Wieland präsentieren die Grundlagen des Peer-to-Peer-Computing, während Ludwig Mittermeier und Roy Oberhauser anhand eines praktischen Beispiels illustrieren, wie sich die P2P-Technologie JXTA gezielt mit Web-Services kombinieren lässt.

Teil 1: Plattformen für Peer-to-Peer-Netze

Der Datenaustausch zwischen zwei gleichberechtigten Partnern ist zwar nichts Neues, erlebt aber derzeit unter dem Begriff „Peer-to-Peer“ (abgekürzt P2P) eine Renaissance. Nach dem Erfolg, den Systeme zum Tausch von Dateien wie Napster hatten, schmücken sich immer mehr Applikationen und Software-Projekte mit dem Etikett „P2P“, um sich als besonders innovativ herauszustellen. Dabei sind die einzelnen Programme jedoch sehr unterschiedlich in Ziel und Technologie, sodass sie alle vermutlich nur eine mehr oder weniger eng begrenzte Verbreitung erreichen werden. Erst eine standardisierte Plattform könnte eine kritische Masse anwachsen lassen, die jedoch zu einem anderen Verständnis von P2P führen könnte, als wir es heute noch haben.

In den letzten Jahren, seit sich im Sommer 2000 „Peer-to-Peer“ zum Modewort entwickelt hat, ist der Begriff zu oft mit der Tauschbörse Napster assoziiert worden. Dadurch ist der Eindruck entstanden, P2P sei gleichbedeutend mit Anarchie und dem Weg in die Illegalität. Dabei wird häufig vergessen, dass P2P-Verbindungen in vielen Fällen die ganz natürliche Art der Kommunikation sind.

Ursprünglich war das Internet selbst als P2P-System entworfen. Es sollte gleichberechtigte Teilnetze miteinander verbinden. Selbst nach einem Angriff mit Atombomben sollte die Kommunikation der Computersysteme in den USA noch auf irgendeinem Weg möglich sein. Jeder Rechner bot wie selbstverständlich auf der einen Seite selber Dienste, nutzte aber auf der anderen Seite auch die Dienste anderer.

Erst in den Neunzigerjahren, als das Internet zu einem Massenmedium wurde, verschoben sich die Aufgaben. Einem Heer von Client-Rechnern, die nur als Konsumenten von Inhalten darauf zugreifen, stehen heute verhältnismäßig wenige Server gegenüber, die um die Gunst der Nutzer buhlen und neben sinnvollen Inhalten immer mehr Werbung integrieren müssen, um sich zu finanzieren. Damit wurde das WWW zu einer Art „interaktivem Fernsehen“, bei dem es eine strenge Trennung zwischen Produzenten und Konsumenten gibt. P2P soll nun diese Rollenverteilung wieder aufheben?

Was ist P2P eigentlich?

Nüchtern gesehen wollen wir unter „Peer-to-Peer“ die bilaterale Kommunikation zwischen zwei oder mehr gleichberechtigten Partnern verstehen, die beide gleichwertige Fähigkeiten und Verantwortlichkeiten haben. Da es keine Rollenverteilung mehr wie im Server/Client-Modell gibt, macht es auch keinen Sinn, diese Begriffe weiter zu verwenden, nicht einmal als Kunstwort („Servent“ oder ähnliches). Betrachtet man nicht ein fest aufgebautes Netz wie etwa das Internet, sondern eine Ad-hoc-Umgebung, in der die Partner spontan miteinander in Verbindung treten, ohne auf eine zugrunde liegende Infrastruktur angewiesen zu sein, wollen wir von einem P2P-Netz zusätzlich noch verlangen, dass es adaptiv und selbstkonfigurierend ist. Das ist insofern gerechtfertigt, als dass ansonsten einer der Netzteilnehmer eine ausgezeichnete Rolle (eine Art Server) einnehmen müsste, die dem P2P-Gedanken widerspräche.

Diese Art der Kommunikation ist weder neu noch besonders ungewöhnlich – im Gegenteil: Sie entspricht der natürlichen zwischenmenschlichen Kommunikation. Jedes Telefongespräch ist eine Peer-to-Peer-Verbindung. Wenn wir also derzeit beginnen, unsere Telefone intelligenter und leistungsfähiger werden zu lassen, sodass sie nicht nur Sprache, sondern auch Daten übertragen können, liegt es nahe, diese Datenübertragung nicht über einen zentralen Server abzuwickeln, sondern direkt, eben im P2P-Modus.

Neue Funktechnologien wie Bluetooth legen genau eine solche Architektur nahe, sind sie doch auf die direkte Kommunikation zwischen zwei Geräten ausgelegt. Auf

diese Weise wird im Kleinen das realisiert, was das Internet in seinen Anfängen einmal war: ein Netz von gleichberechtigten Partnern.

Wenn man heute von P2P spricht, spielen derartige Ad-hoc-Szenarien noch keine große Rolle. Viel mehr Aufmerksamkeit erregen einige P2P-Projekte, besonders weil sie nicht nur technologisches Neuland betreten, sondern oft die traditionelle Rollenverteilung in der Informationsgesellschaft in Frage stellen. Dazu gehören:

- *Napster:* Die Tauschbörse Napster sorgte dafür, dass alle Nutzer Musikstücke im MP3-Format untereinander austauschen konnten – ohne Rücksicht auf deren Urheberrechte. Die Suche nach Stücken erfolgte dabei noch über einen zentralen Server, der Datenaustausch ging dann direkt von einem Rechner zum anderen.
- *Gnutella:* Seitdem Napster in seiner ursprünglichen Form eingestellt wurde, hat Gnutella einigen Aufschwung erlebt. Es ist ebenso ein System zum Austausch von Musikstücken, verzichtet aber auf jegliche zentralen Elemente. Die Suche erfolgt daher lawinenartig von einem Peer zu allen von dort aus unmittelbar erreichbaren usw., bis zu einer vorgegebenen Tiefe. Dieses Suchmuster spielt auch im Ad-hoc-Fall eine Rolle.
- *SETI@Home:* In diesem Projekt sollen die brachliegenden Rechenkapazitäten von Millionen Benutzern dazu verwendet werden, um in Signalen von Radioteleskopen Spuren extraterrestrischer Intelligenz zu bestimmen. Es ist nicht das erste, aber bei weitem bekannteste Projekt dieser Art. Auch andere haben sich zum Ziel gesetzt, ungenutzte Ressourcen anderer Peers für eigene Aufgaben zu verwenden.
- *Jabber:* Hierbei handelt es sich um ein Open-Source-Projekt für Instant Messaging, also die direkte – wenn auch schriftliche – Kommunikation zwischen zwei oder mehr Peers. Aufgrund der Nähe zur natürlichen Kommunikationsweise eignet sich P2P für diese Art von Anwendungen besonders.

Als Gemeinsamkeit der meisten Projekte bleibt also festzuhalten: Ziel beim P2P-Computing ist es meist, in einem Netz auf mehrere Peers verteilte Ressourcen (wie Dateien oder freie CPU-Zeiten) aufzufinden und gemeinsam zu nutzen. Der Zugriff auf diese Ressourcen erfolgt dabei nicht unberechtigt, sondern wird vom Benutzer des jeweiligen Peer autorisiert – zum Beispiel indem er seine Musikstücke in den Gnutella-Ausgangsordner kopiert.

Geht man noch einen Abstraktionsschritt weiter, stellt man fest, dass in einem P2P-Netz die Aspekte des Anbietens einer Information und der gezielten Suche nach einer passenden Information die größten Rollen spielen. Dieses Anbieten und Suchen kann bereits der eigentliche Zweck des Netzes sein, wie es bei Napster und Gnutella ja der Fall ist. Es kann allerdings auch nur Bestandteile einer größeren Applikation sein, die darauf ein intelligentes Informationsmanagement aufbaut.

Vorteile von P2P

Weshalb waren und sind Projekte dieser Art eigentlich so erfolgreich? Zum einen spielt dabei sicher der inhärente revolutionäre Geist eine gewichtige Rolle. Viele Nutzer sahen dadurch eine Möglichkeit, aus der Kommerzialisierung des Internet auszubrechen und ihr eigenes Netzwerk von Beziehungen aufzubauen – ganz wie im richtigen Leben. Zum anderen gibt es natürlich aber auch eine Reihe technischer Gründe, die für eine Verteilung von Inhalten über ein P2P-Netz sprechen.

Viele Server haben nur eine beschränkte Skalierbarkeit. Greifen zu viele Clients gleichzeitig darauf zu, kann es vorkommen, dass nicht mehr alle bedient werden können (s. Abb. 1, oben). Auch in der Netzumgebung des Servers kommt es dann zu einem erhöhten Datenverkehr, der eventuell nur schwer zu bewältigen ist. Das Problem gerade des Internet ist es, dass es je nach Erfolg eines Informationsangebots vorkommen kann, dass plötzlich Millionen von Clients auf einmal dieses Angebot anschauen möchten. Dieser Flaschenhals kann zwar durch den Einsatz von Caching-Mechanismen etwas gelockert werden, das prinzipielle Problem bleibt hingegen bestehen. Außerdem bedeutet eine Client/Server-Architektur auch immer eine Abhängigkeit von der Verfügbarkeit des Servers. Fällt dieser zentrale Punkt aus, ist das gesamte System lahmgelegt.

Peer-to-Peer-Netze dagegen übertragen die Informationen ohne die Vermittlung durch eine zentrale Stelle (s. Abb. 1, unten) und sind somit erheblich robuster. Der Ausfall eines Peers beeinträchtigt die Funktionsweise des Netzes im Allgemeinen nicht. Wird ein Peer nicht gefunden, versucht das System, auf einen anderen auszuweichen und die gewünschte Information von diesem zu beziehen. Schlimmstenfalls kann es vorkommen, dass einzelne Suchanfragen oder Zugriffsversuche erfolglos bleiben. Das mag zwar für den jeweiligen Anfragenden ärgerlich sein, beeinflusst die Stabilität und Verfügbarkeit des Gesamtsystems jedoch in keiner Weise. Die P2P-Technologie macht sich dabei also die Vorteile von Konzepten zunutze, die die Natur seit Urzeiten bei Schwarmvölkern wie Ameisen, Bienen oder Vögeln erfolgreich anwendet.

Die Dezentralisierung setzt sich oberhalb der rein technischen Ebene, aber auch auf der logischen Ebene fort. Der mögliche technische Ausfall einer zentralen Komponente wie eines Servers wäre nicht so bedrohlich, wenn zusammen mit dem Ausfall nicht auch der Verlust der dort residierenden Informationen drohen würde. Indem

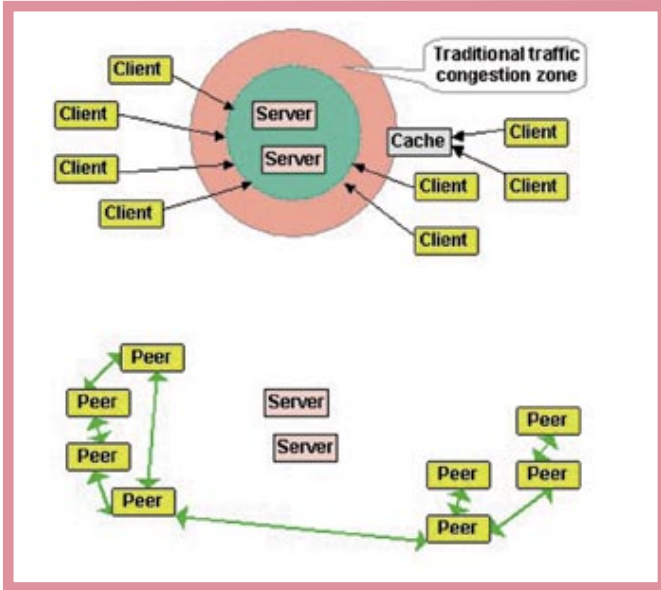

Abb. 1: Während sich im Client/Server-Modell aller Datenverkehr auf den Server konzentriert, tauschen im P2P-Netz die Peers ihre Daten direkt untereinander aus.

die Inhalte nicht an einer Stelle gespeichert werden, sondern in kleineren Dosen über ein Netz verteilt sind, reduziert sich die Gefahr, dass Informationen verloren gehen, ganz erheblich. Außerdem gibt es dadurch keinen Angriffspunkt mehr für Sabotage und Spionage, was in einigen Fällen durchaus von Bedeutung sein könnte.

Außerdem bedeutet P2P einen schonenderen Umgang mit Ressourcen. Da Daten nur untereinander ausgetauscht werden, ist die Netzlast deutlich geringer und die zur Verfügung stehende Bandbreite wird effektiver, nämlich symmetrischer genutzt. Damit ist diese Art der Kommunikation auch für schmalbandigere Verbindungen wie im Mobilfunkbereich interessant.

Betrachtet man ein Ad-hoc-Netz, das sich mehr oder weniger spontan aus einer Reihe von Teilnehmern bildet, beispielsweise aus mobilen Geräten, die sich drahtlos mit Kurzstreckenfunktechnologien wie WLAN oder Bluetooth verbinden, so ist für dieses das Peer-to-Peer-Paradigma vollkommen natürlich. Hierbei müssen wir stets von einem hoch dynamischen Netz mit laufend wechselnden Teilnehmern ausgehen, die auf keine zugrunde liegende Infrastruktur aufbauen können. Damit bildet P2P die Schlüsseltechnologie für die Selbstkonfiguration dynamischer verteilter Systeme.

Auf der Ebene der Applikationen haben sich im Bereich der Middleware verschiedene Ansätze wie DCOM, CORBA oder RMI etabliert. Diese sind jedoch meist für das Auffinden von Diensten auf einen Teilnehmer mit einer ausgezeichneten Rolle angewiesen, der als „Broker“ zwischen Dienstnutzer und Dienstanbieter vermittelt, so etwa der Naming-Service bei CORBA oder die Registry bei DCOM. In Umgebungen, wo keine Infrastruktur vorhanden ist bzw. keine vorausgesetzt werden kann, sind diese Technologien daher nur schlecht anwendbar. Wir werden später sehen, wie sich auch in diesem Fall nach dem P2P-Muster Dienste entdecken und nutzen lassen.

Strukturen von Peer-to-Peer-Netzen

Ziel der Middleware ist ja immer der wohl organisierte Austausch von Nachrichten. Zur Koordination der Beteiligten oder der Übertragung von Inhalten benötigt man Nachrichten in Client/Server-Architekturen genauso wie in Peer-to-Peer-Architekturen. Letztere bringen indessen für die Nachrichtenkommunikation zusätzliche Herausforderungen mit sich, die der Client/Server-Fall nicht kennt.

In P2P-Netzen, bei denen nicht alle Peers direkt miteinander verbunden sind, müssen Routing-Verfahren eingesetzt werden, um eine Nachricht über verschiedene Vermittlungs-Peers zum Empfänger zu transportieren. Diese Multi-Hop-Technik kommt besonders bei Ad-hoc-Netzen zum Einsatz, ist aber auch bei P2P-Verbindungen etwa im Internet zuweilen nötig, wenn die Peers sich nicht alle gegenseitig kennen. Was hier so kompliziert klingt, entspricht tatsächlich wieder nur dem alltäglichen Kommunikationsmuster; wenn ich dem Freund meiner Schwester etwas mitteilen möchte, trage ich diese Botschaft oft einfach meiner Schwester auf.

Außerdem muss man beachten, dass die Verbindung zwischen den Peers nicht notwendigerweise dauerhaft besteht, sondern lediglich vorübergehender Natur sein könnte. Daher muss man Caching-Mechanismen vorsehen, die Nachrichten so lange puffern, bis der Ziel-Peer wieder erreichbar ist bzw. bis der Ausgangs-Peer wieder eine Netzverbindung hat, um die Nachricht abzuschicken. Gegebenenfalls müssen die Nachrichten auch mit einer maximalen Lebenszeit versehen werden, nach der sie aus dem Puffer nicht mehr weitergeleitet werden sollen.

Die jeweils reinen Client/Server- und Peer-to-Peer-Konstellationen sind in der Praxis nur die beiden Enden des Spektrums. Dazwischen gibt es vielfältige Mischformen, die wir nach [Qua02] als „Hybrid-Systeme“ bezeichnen wollen.

- Bei Napster gibt es beispielsweise einen zentralen Server, der als Verzeichnis für die Angebote aller Teilnehmer dient. Das Suchen nach Musikstücken ist also ein reiner Client/Server-Vorgang. Die Übertragung der Dateien erfolgt dann direkt zwischen den Peers.
- Beim Instant-Messaging-System ICQ erfolgt die Kommunikation normalerweise direkt zwischen den Peers. Ein zentraler Server dient dabei im Hintergrund als Rückversicherung, wenn das Zustandekommen einer Verbindung scheitert.

Generell müssen von den vier Aspekten Funktionalität, Technologie, Daten und Präsentation nicht immer alle von den jeweiligen Peers übernommen werden, sondern können auch ganz oder zum Teil an einen Server delegiert werden. So kann beispielsweise die Datenhaltung zentralisiert werden oder ein zentrales Portal für die Peers geschaffen werden, die dann „nur“ noch Funktionalität bereitstellen (ähnlich dem SETI@Home-Ansatz, siehe auch [Qua02]).

Suche nach Informationen

Wie wir bereits gesehen haben, gehören Suchmechanismen zu den weiteren zentralen Funktionsmerkmalen, die ein P2P-Netz bieten muss. Dabei unterscheiden wir zwischen der lokalen Suche, also nur auf den Peers, die vom Suchenden aus unmittelbar erreichbar sind, und der großflächigen Suche, die dann auch weitere Peers mit einschließt. Für diese Art von Suche haben sich verschiedene Ansätze entwickelt, die entweder auf der Netztopologie direkt aufbauen oder nach der Übereinstimmung von Inhalten ausgerichtet sind.

Aufbauend auf der Netztopologie bedeutet, dass sich die Suche nach dem Grad der Dezentralisierung des Netzes richtet (s. Abb. 2). Gibt es eine zentrale Instanz, bei der die zu suchenden Informationen hinterlegt sind, kann und sollte diese auch für die Suche genutzt werden, denn sie ist sicher deutlich schneller. Im völlig dezentralen Fall, wenn kein solcher Server existiert, erfolgt die Suche von einem Peer zum nächsten bzw. zu allen, die von ihm aus direkt erreichbar sind. Diese Technik verfolgt beispielsweise Gnutella. Um nicht das Netz mit Suchanfragen zu überfluten, erhält jede

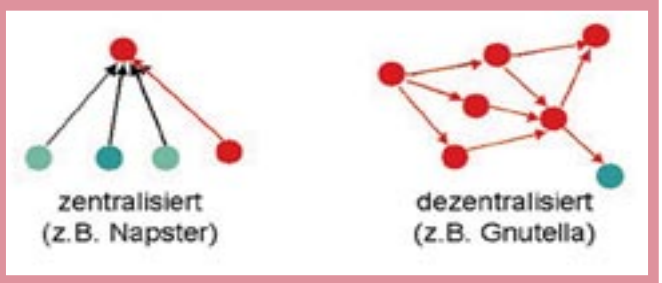

Abb. 2: Bei Suchverfahren, die sich an der Netztopologie orientieren, setzen zentrale Verfahren einen Server voraus, dezentrale durchsuchen dagegen alle Nachbarn bis zu einer vorgegebenen Tiefe.

dieser Anfragen eine maximale Tiefe, den Suchhorizont. Weitere Optimierungen können durch Reduktion dieser Tiefe durch Gewichtung der Verbindungen, etwa nach Zahl oder Art der Hops, der Round-trip-Zeiten oder Ähnlichem erfolgen.

Einen anderen Ansatz verfolgt die inhaltsbasierte Suche. Eine Möglichkeit dabei ist, einige Peers besonders auszuzeichnen und auf ihnen Hash-Tabellen für die Inhalte in ihrer Umgebung abzulegen. Bei einer Suche müssen dann nicht alle Peers, sondern nur die „Super-Peers", die über solche Tabellen verfügen, befragt werden. Bei nicht so umfangreichen Informationen kann auch gleich der gesamte Inhalt auf den Super-Peers abgelegt werden. Wie Sie merken, handelt es sich auch hierbei um ein hybrides Verfahren, das das Konzept der Client/Server-Welt auf eine dezentralisierte Umgebung abbildet. Der Nachteil dieser Hash-Tabellen ist, dass sich damit nur Inhalt mit wenigen Attributen sinnvoll erfassen und codieren lässt. Komplexere Informationen – wie sie in der Praxis meist vorkommen – lassen sich damit kaum verwalten.

Eine Alternative dazu ist, einen Peer nicht völlig zu durchsuchen, sondern darauf zu vertrauen, dass er die relevanten Informationen selbst deklariert. Bei diesem Suchverfahren nach dem Publisher-Subscriber-Entwurfsmuster bietet jeder Peer die Dienste, die er anderen zur Verfügung stellen will, auf einer Art „Werbetafel" feil. Ein Suchalgorithmus muss dann lediglich die „Werbetafeln" lesen und sich nicht mit dem Durchsuchen von Peers aufhalten.

Beide Verfahren lassen sich auch kombinieren (s. Abb. 3). Wenn alle Peers ihre Angebote auf solchen „Werbetafeln" beschreiben, können Super-Peers die Dienste, die Peers in ihrer Umgebung zur Verfügung stellen, abfragen und zwischenspeichern. Global gesehen geht die Suche dann dezentral von einem Super-Peer zum nächsten, wobei jedoch ein sehr viel dünneres P2P-Netz durchkämmt wird als das eigentliche. Lokal ist die Suche sogar zentralistisch, da der Super-Peer eine Server-Rolle für die anderen spielt. Ein derartiger Algorithmus ist beispielsweise in der P2P-Plattform JXTA enthalten. Zu dieser kommen wir aber gleich.

Die vielen verschiedenen Ansätze lassen es bereits vermuten: Die optimale Suchstrategie für P2P-Netze gibt es nicht. Wie bei anderen Suchproblemen auch, hängt sehr viel von der Applikation ab, also von der Art der Information, nach der gesucht werden soll.

P2P jenseits des Hypes

Wie bei jeder Begeisterungswelle fragt man sich auch beim Thema „Peer-to-Peer", was am Ende übrig bleiben wird. Insbesondere will der Entwickler wissen, wo er auf P2P setzen soll und wo besser nicht. Für ihn ergibt sich dabei folgendes Bild:

Im Internet und anderen stationären Netzen bedeutet P2P eine Verschiebung der Aufgaben von den Servern zurück zu den Clients. Das kann politisch-ideologisch motiviert sein, kann aber auch einfach aus der Erkenntnis erwachsen, dass heutige Clients meist so leistungsfähige Computer sind, dass ein Degradieren zur reinen Browsing-Station eine Verschwendung von Ressourcen wäre. In diesem Fall lassen sich P2P-Technologien also dann besonders gut einsetzen, wenn es darum geht, verteilte Ressourcen möglichst effektiv zu nutzen. Es mag sich einfach um ein verteiltes Caching für umfangreiche Downloads handeln, ein Auslagern von Rechenaufgaben oder Ähnliches. P2P kann auch für das gemeinsame Nutzen von Dateien verwendet werden.

In Ad-hoc-Netzen sieht die Situation etwas anders aus, insbesondere wenn diese

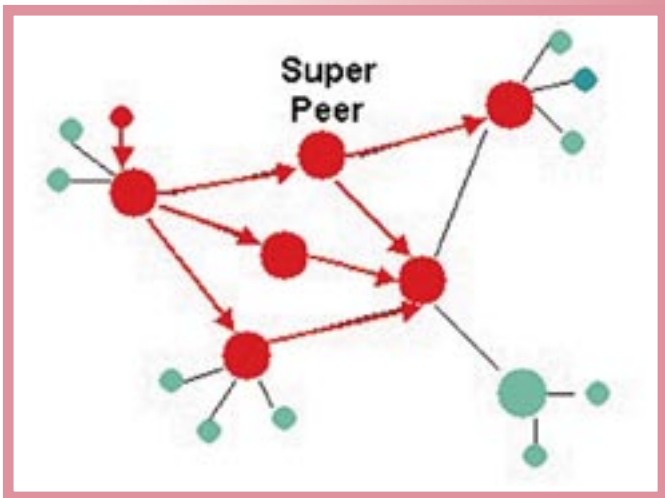

Abb. 3: Zur Suche können global dezentrale und lokal zentrale Konzepte mit Publish-Subscribe-Ansätzen kombiniert werden.

Netze drahtlos aufgebaut sind. Hier spiegelt das Peer-to-Peer-Konzept die natürliche Art der Kommunikation wider. Zentralistische Ansätze sind dabei aufgrund ihrer Abhängigkeit von einem bestimmten Punkt meist weniger robust und zuverlässig. Gerade für das Zusammenspiel von tragbaren Geräten eines Nutzers oder verschiedener Nutzer wird das P2P-Paradigma in Zukunft sicher große Bedeutung erlangen.

JXTA als Plattform

Der Erfolg hängt wie so oft jedoch von Standards ab. So lange jeder P2P-Ansatz ein eigenes Protokoll verwendet, ist die Interaktion mit anderen auf solche Peers beschränkt, die ebenfalls dieses Protokoll verstehen, was meist gleichbedeutend mit der Installation der jeweiligen Applikation ist. Leider fokussieren sich fast alle Projekte, die wir oben genannt haben, auf ein bestimmtes Produkt oder Protokoll; sie bieten keine allgemeine Lösung, sondern implementieren jeweils nur einen Teil des gesamten P2P-Konzepts, wie ein File Sharing System (Groove) oder ein Such-Protokoll (Gnutella).

Sinnvoller wäre es da, wenn alle Peers die „gleiche Sprache" verstehen könnten. Dafür ist eine standardisierte Plattform mit einem von allen akzeptierten Protokoll nötig, auf der die individuellen Anwendungen aufbauen. Zudem sind die P2P-Konzepte reichlich komplex, sodass nebeneinander konkurrierende Implementierungen keinen großen Nutzen bringt. Eine solche universelle Plattform könnte JXTA (gesprochen: „jaxta") werden. JXTA ist ein Open-Source-Projekt [JXTA], das vom Cheftechnologen Bill Joy von Sun Microsystems ins Leben gerufen wurde und nach wie vor intensiv von Sun angetrieben wird.

Anders als klassische Open-Source-Projekte handelt es sich bei JXTA primär nicht um ein reines Software-Projekt. JXTA steht vielmehr nur für die Spezifikation und die Architektur einer P2P-Plattform, worin die einzelnen Bestandteile, die Protokolle und deren Zusammenspiel festgelegt werden. Vorrangiges Ziel ist die Entwicklung einer Reihe offener Protokolle, die es allen zu einem Netz zusammengeschlossenen Geräten – vom Mobiltelefon und PDA bis zum Desktop-PC – erlauben, miteinander zu kommunizieren und Informationen als Peers auszutauschen.

Dieses Design ist abstrakt und programmiersprachenunabhängig, hat also trotz des „J“ im Namen und der Beteiligung von Sun zunächst nichts mit Java zu tun. Gleichzeitig mit den Konzepten entsteht jedoch auch eine Referenzimplementierung, die die Tragfähigkeit für die Praxis unter Beweis stellen soll und die in Java entwickelt wird. Mittlerweile sind aber auch schon Implementierungen, so genannte JXTA Bindings, in anderen Sprachen in Arbeit.

Die JXTA-Architektur

Die eigentliche JXTA-Plattform, die oft als „JXTA Core“ bezeichnet wird, bietet die wesentlichen Bausteine, die für Peer-to-Peer-Netze notwendig sind. Sie stellt den P2P-Anwendungen Komponenten mit Mechanismen wie Discovery, Transport sowie Etablierung von Peer Groups mit den zugehörigen Sicherheitsaspekten zur Verfügung.

Die Grundkonzepte im JXTA Core sind Peers, Peergruppen und Pipes (s. Abb. 4). Ein Peer ist dabei der Knoten im P2P-Netz, der seine Ressourcen für andere bereitstellt. Mit Peergruppe meint man eine Gruppe von Peers mit gegenseitigen Zusicherungen, innerhalb derer die Mitgliedschaft autorisiert und kontrolliert werden kann. Die Mitglieder der Peergruppe können die Dienste dieser Gruppe verwenden und Nachrichten austauschen. Alle Peers sind stets Mitglieder der so genannten NetPeerGroup – die NetPeerGroup ist die globale Gruppe (tatsächlich eindeutig im Universum), in der sich alle JXTA-Peers registrieren müssen.

Um eine JXTA-Anwendung zu starten, muss man einfach eine Instanz von NetPeerGroup erzeugen:

```
static PeerGroup netPeerGroup = null;

private void startJxta() {
 try {
  // create, and Start the default jxta NetPeerGroup
  netPeerGroup = PeerGroupFactory.newNetPeerGroup();

 } catch (PeerGroupException e) {
  // could not instanciate the group,
  // print the stack and exit
  System.out.println("fatal error: group creation failure");
  System.exit(1);
 }
```

Die Pipe ist die Abstraktion der Verbindung zwischen den Peers. Pipes sind unidirektionale, asynchrone Kommunikationskanäle, welche nicht auf festen Endpunkten und einem zentralen Schema für das Adressenmanagement basieren, sondern die folgenden Kriterien erfüllen:

- Der Transport-Endpunkt kann sich auf allen Knoten ändern (zum Beispiel von HTTP zu Bluetooth oder Wireless LAN),
- Pipes können auch mittels eingeschränkter Transportverfahren wie analog paging network implementiert werden.

Es gibt zwei verschiedene Arten von Pipes: Propagate Pipe und Unidirectional Pipe. Die Propagate Pipe wird für Broadcast-Nachrichten, also das breite Streuen ins Netz wie etwa bei Discovery-Nachrichten verwendet. Die unidirektionale Pipe ist für alle anderen Kommunikationsarten zwischen zwei Peers gedacht. Wir werden später sehen, welche Rolle jede der beiden Arten für das Auffinden von Peers und Ressourcen spielt.

JXTA legt nicht fest, welches Transportprotokoll (z. B. HTTP, Beep, TCP, TCL) verwendet werden muss. Basierend auf der Netztopologie und typischen Kommunikationsmustern kann der Entwickler selbst entscheiden, welche Protokolle er einsetzen möchte.

JXTA Services

Einer der wichtigsten JXTA Core Services ist der Discover-Dienst. Wenn ein Peer ans Netz geht, schickt er eine „Werbetafel“, die bei JXTA auf gut Englisch „Advertisement“ heißt, sowohl an alle Peers, die sich im selben lokalen Netz befinden wie er selbst, als auch an so genannte „Rendezvous-Peers“. Diese sind Peers, die den Rendezvous-Service unterstützen, um Anfragen auch über die Peergruppe hinaus zu verbreiten. Rendezvous-Peers ermöglichen damit die Kommunikation zwischen verschiedenen lokalen Netzen. Die versendeten Advertisements werden dabei im Cache der anderen Peers gespeichert.

Falls nun ein Peer einen anderen Peer oder andere Ressourcen auf einem solchen finden möchte, schickt dieser Peer eine Discovery-Nachricht ins Netz, welche das Advertisement aus dem Cache der empfangenden Peers liest. In Abbildung 5 kann man sehen, wie das Auffinden anderer Peers funktioniert. Wichtig ist, dass zuerst eine Propagate-Nachricht geschickt wird. Nachdem der Peer A die Antwort mit der entsprechenden ID der Input-Pipe von den anderen Peers und Rendezvous-Peers bekommen hat, wird eine Anfrage nach Advertisements an diese Input-Pipe geschickt.

Allgemein unterscheidet sich die Definition eines JXTA-Service kaum von der eines „Servers“ in einer Client/Server-Architektur. Ein JXTA-Dienst kann von einzelnen Peers oder von Peergruppen angeboten werden und von den anderen Peers, die zur selben Peergruppe gehören, verwendet werden. Solche Dienste umfassen beispielsweise Suche, Indexing oder File-Sharing.

Allgemein kann man eine konzeptionelle Parallelität von JXTA-Services zu XML-Web-Services feststellen (siehe Teil 2). Da JXTA jedoch für Ad-hoc-Umgebungen gedacht ist, tritt an die Stelle des zentralistischen UDDI-Ansatzes der oben beschriebene Discovery-Mechanismus.

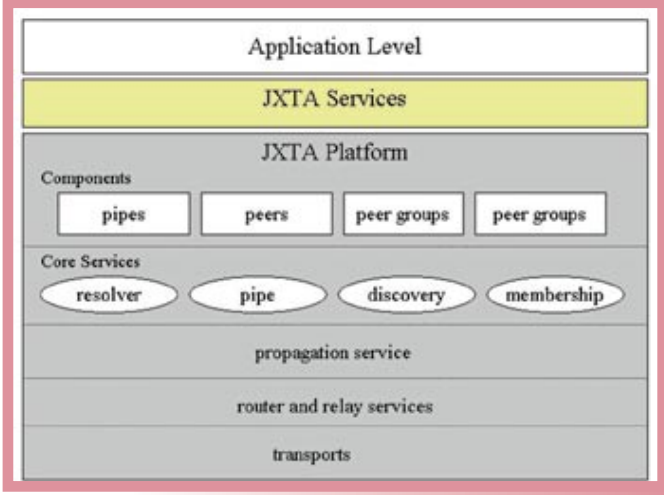

Abb. 4: JXTA sieht eine Drei-Schichten-Architektur vor.

Status von JXTA

Die JXTA-Protokolle sind die Basis für die Interoperabilität der JXTA-Anwendungen (s. Abb. 6). Mit JXTA kann man somit P2P-Anwendungen für verschiedene Umgebungen und Geräte entwickeln, wobei alle Geräte wie PDAs, Handhelds, Mobiltelefone, Desktop-PC usw. Bestandteile eines P2P-Netzes sind und mit einander kommunizieren können.

Unter den Implementierungen ist die Referenzversion für J2SE am weitesten fortgeschritten. Für die Entwicklung eigener Anwendungen oder auch die Mitarbeit an der Plattform kann sie jeder von [JXTA] herunterladen. Es gibt dort etwa jede Woche eine neue instabile („unstable") Version und ungefähr jeden Monat eine neue stabile Version. Mit dieser kann man heute P2P-Anwendungen implementieren, die die folgende Funktionalität umfassen:

- Discovery,
- Messaging,
- Sharing of Resources,
- Network Services,
- Ad-hoc Fähigkeit.

Dennoch hat das Projekt JXTA als Ganzes und die Referenzimplementierung im Besonderen noch nicht die Reife erreicht, als dass man sie für die Produktentwicklung guten Gewissens empfehlen könnte. Es stellt jedoch einen umfassenden und durchdachten Ansatz dar, der nach Behandlung der Kinderkrankheiten die oben beschriebenen Trümpfe durchaus auszuspielen in der Lage sein könnte.

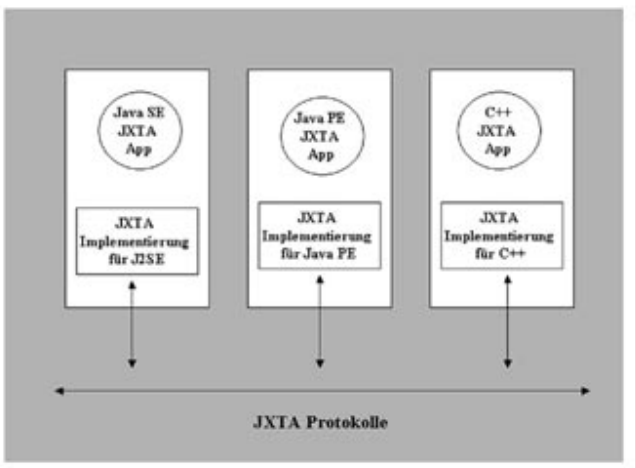

Abb. 6: Die JXTA-Protokolle ermöglichen die Interoperabilität verschiedener Implementierungen.

Ekaterina Chtcherbina ist Softwarearchitektin für mobile Anwendungen bei der Zentralabteilung „Corporate Technology" der Siemens AG. Sie interessiert sich besonders für Ad-hoc-Netze und das Zusammenspiel verteilter Dienste sowie für Multimedia-Kommunikation. E-Mail: ekaterina.chtcherbina@mchp.siemens.de.

Dr. Thomas Wieland leitet bei der Zentralabteilung „Corporate Technology" der Siemens AG die Vorfeldentwicklung auf dem Gebiet der Architekturen für die Vernetzung flexibler Dienste. Er ist außerdem Autor mehrerer Bücher zur Windows- und Linux-Programmierung. E-Mail: thomas.wieland@mchp.siemens.de.

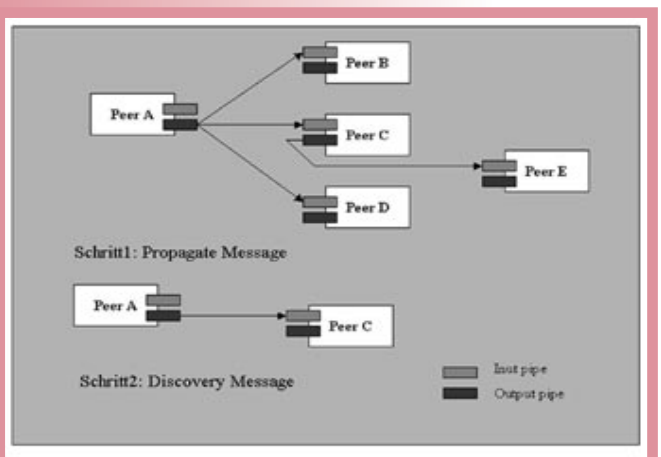

Abb. 5: Funktionsweise des Auffindens von Ressourcen in JXTA

Weiterführende Informationsquellen

http://www.openp2p.com
http://www.jxta.org