GKM+07] *Kalman Graffi, Aleksandra Kovacevic, Patrick Mukherjee, Michael Benz, Christof Leng, Dirk Bradler, Julian Schroeder-Bernhardi, Nicolas Liebau*; ***Peer-to-Peer Forschung - Überblick und Herausforderungen***. Information Technology (Methods and Applications of Informatics and Information Technology), July 2007, S.272-279.

Peer-to-Peer-Forschung – Überblick und Herausforderungen

Peer-to-Peer Research – Overview and Challenges

Kálmán Graffi, Aleksandra Kovačević, Patrick Mukherjee, Michael Benz, Christof Leng, Dirk Bradler, Julian Schröder-Bernhardi, Nicolas Liebau, Technische Universität Darmstadt

Zusammenfassung Peer-to-Peer ist ein fundamentales Designprinzip und stellt einen Paradigmenwechsel für die Kommunikation in Computernetzwerken dar. In diesem Beitrag wird zunächst definiert, welche Charakteristika Peer-to-Peer-Systeme ausmachen. Anhand von vier Qualitätsmerkmalen hat die Forschergruppe „QuaP2P" ihre Arbeitsbereiche untergliedert. Dieser Gliederung folgend wird anschließend der aktuelle Stand der Wissenschaft und Herausforderungen im Bereich Peer-to-Peer-Forschung zusammengefasst. Dies gibt dem Leser einen strukturieren Überblick, um sich mit den wesentlichen Arbeiten dieses hoch aktuellen Forschungsthemas auseinandersetzen zu können. ►►► **Summary** Peer-to-Peer is a fundamental design principle and marks a paradigm shift for communication in computer networks. In this paper first a definition for peer-to-peer systems is given based on their characteristics. Using four quality attributes the DFG Research Group "QuaP2P" subdivided its research fields. Subsequently, the state-of-the-art in peer-to-peer research and its challenges are summarized, following this classification. This will give a structured overview to the reader in order to provide him with the relevant related work when looking into this highly active research area.

KEYWORDS C.2.4 [Computer Systems Organization: Computer-Communication Networks: Distributed Systems] distributed applications, peer-to-peer, survey

1 Einleitung

Der Begriff Peer-to-Peer (P2P) wurde Mitte des Jahres 2000 geprägt. Heute sind Anwendungen, die auf dem P2P-Paradigma basieren, für über 50% des Verkehrs im Internet verantwortlich; manchmal liegt der Anteil sogar bei bis zu 80% [1]. Aufgrund der ständig anwachsenden Anzahl der Internetnutzer und deren verbesserten Anbindungsqualität ist ein weiteres Anwachsen des P2P-Verkehrs zu erwarten.

Der Begriff „Peers" bedeutet im Englischen Gleichgestellte. Er drückt den direkten Austausch von Diensten zwischen gleichgestellten Partnern aus. Dies stellt den Gegensatz zum bekannten Client/Server-Paradigma dar, bei welchem Clients Dienste von einem Server beziehen und der Server dies kontrolliert und steuert.

In der Wissenschaft existieren verschiedene Definitionen für ein P2P-System. Die Grundprinzipien beschreibt Oram [2]: „*Ein P2P-System* ist *ein sich selbst organisierendes System gleichberechtigter, autonomer Entitäten (Peers), das die gemeinsame Nutzung von verteilten Ressourcen in einer vernetzten Umgebung zum Ziel hat, wobei zentrale Dienste vermieden werden.*"

Kurz: Ein *System mit vollständig dezentraler Selbstorganisation und Ressourcennutzung.*

Über diese Grundprinzipien hinaus, kann ein P2P-System mittels neun Punkte charakterisiert werden, die man in zwei Eigenschaften untergliedern kann [3; 4], dezentrale Ressourcennutzung und dezentrale Selbstorganisation.

1.1 Dezentrale Ressourcennutzung:

1) Die relevanten Betriebsmittel (Bandbreite, Speicherplatz, Rechenkapazität) werden möglichst gleichmäßig verteilt genutzt und befinden sich an dem so genannten „Rand" der Rechnernetze, bei den „Peers".
2) Eine Menge von Peers nutzen untereinander die von ihnen

wiederum zur Verfügung gestellten Betriebsmittel.

3) Peers sind über das Internet miteinander verbunden und meist weltweit verteilt.
4) Ein Peer ist oftmals nicht stets unter derselben IP-Adresse zu identifizieren (variable Konnektivität).

1.2 Dezentrale Selbstorganisation:

5) Zur eigentlichen Nutzung der Betriebsmittel interagieren Peers direkt miteinander. Dies erfolgt im Allgemeinen ohne jegliche zentrale Steuerung.
6) Der Zugriff auf bzw. Transfer der gemeinsam benutzten Betriebsmittel erfolgt direkt zwischen den Peers.
7) Peers selbst besitzen sowohl Client- als auch Server-Funktionalität.
8) Peers sind untereinander gleichberechtigt. Sie besitzen individuell die vollständige Autonomie bezüglich ihrer jeweiligen Betriebsmittel.
9) Das Auffinden von Ressourcen sollte möglichst ohne jeglichen zentralen Dienst erfolgen.

Mit dieser Definition wird deutlich, dass P2P ein fundamentales Designprinzip ist und einen Paradigmenwechsel von *Koordination hin zu Kooperation*, von *Zentralität hin zu Dezentralität*, und von *Kontrolle hin zu Anreizen* widerspiegelt.

Ähnlich untergliedert sich auch die derzeitige Forschung in diesem Bereich. Im Folgenden werden die relevanten Arbeiten und Bestrebungen in den Bereichen Overlaynetzwerke, effiziente Abbildung von P2P Systemen auf das darunter liegende Netzwerk, Strategien für die Suche, grundlegende P2P-Dienste (wie Replikation, Konsistenz und Kohärenz), Sicherheit, sowie Applikationstypen basierend auf dem P2P-Paradigma vorgestellt.

2 P2P Overlays

P2P Forschung unterscheidet verschiedene Arten von P2P Overlaynetzen. Man kann sie klassifizieren nach dem Zusammenhang zwischen gespeicherten Informationen und Overlaynetztopologie (strukturiert oder unstrukturiert), Grad der Zentralisierung (rein dezentral, teilweise zentral oder hybrid) sowie der Struktur des Overlaynetzes (flach oder hierarchisch) [5].

Rein dezentral. Jeder Peer hat exakt die gleichen Aufgaben im Netzwerk. Es gibt viele Umsetzungen (Chord [6], Kademlia [7], Gnutella 0.4 [8], etc.).

Teilweise zentral. Einige Knoten haben eine wichtigere Rolle, man bezeichnet sie als *Supernodes*. Zu den Protokollen, die dieses Paradigma umsetzen, gehören Gnutella 0.6 [9], FastTrack [10], KaZaA [11], JXTA [12] und Globase.KOM [13].

Hybrid. Der Dateitausch ist wie in rein dezentralen Netzen organisiert, die Informationssuche (und die Indexierung) findet durch einen zentralen Server statt. Dies reduziert den Nachrichtenoverhead. Bekanntestes Beispiel ist Napster [14].

Unstrukturiert. Der Speicherort der Informationen in unstrukturierten Overlaynetzen ist vollständig von der Netztopologie entkoppelt. Sie können auf irgendeinem Peer liegen. Zur Suche schickt man jedem Nachbarn eine Anfrage (Flooding). Fortschrittlichere Protokolle benutzen beispielsweise *Random Walk* oder *Routing Indizes* als Suchstrategien. Beispiele hierfür sind Gnutella [8], BubbleStorm [15] und FreeHeaven [16].

Strukturiert. In strukturierten Overlaynetzen werden die Informationen in Peers gespeichert, die anhand ihrer Position in der streng eingehaltenen Overlaynetztopologie bestimmt werden. Oft werden hier statt der eigentlichen Daten nur die Indexinformationen, also der Ort der Daten, hinterlegt. Hierfür verwenden die meisten derzeitigen strukturierten Protokolle verteilte Hashtabellen (*DHT*). Vertreter sind die Protokolle Chord [6], Kademlia [7], CAN [17], Tapestry [18], Pastry [19] und Viceroy [20].

Hierarchisch. Hierachische Overlaynetze setzen sich aus verschiedenen Schichten von anderen Overlaynetztopologien zusammen. Ein Peer gehört mehreren Schichten an und hat somit auch verschiedene IDs. Ein Beispiel ist Omicron [21]. Dieses Protokoll gruppiert Peers, die de Bruijn-basierte Overlaynetze formen.

Flach. Fast alle genannten P2P Protokolle sind flach, sie bestehen nur aus einer Overlaynetzschicht.

Keines der zuvor genannten Overlaynetze ist einem anderem klar überlegen. Beispielsweise sind strukturierte Ansätze skalierbarer und Informationen können besser gefunden werden als in unstrukturierten, welche allerdings stark schwankende Peer Populationen besser unterstützen. Teilweise zentrale Overlaytopologien erleichtern das Auffinden von Diensten und reduzieren den Protokolloverhead. Jedoch setzen sie voraus, dass einige Peers die Kapazitäten haben, als Supernodes mehr Aufgaben zu erfüllen. Der Protokolloverhead in hybriden Overlaynetzen ist natürlich sehr gering, jedoch sind diese Netze ebenso offensichtlich nicht sehr ausfallsicher (*single-point-of-failure*).

3 Berücksichtigung unterer Netzwerkschichten

Zur Effizienzsteigerung von Overlaynetzwerken bietet sich die Einbeziehung von Eigenschaften unterliegender Netzwerkschichten an. Von besonderem Interesse sind die Topologie sowie die Bandbreitenverwaltung.

Ziel der Nutzung der Topologieinformationen ist dabei die Senkung der Latenz sowie des Verkehrsaufkommens im Underlay. Ein gängiger Ansatz in der Literatur ist es, die unterliegende Netzwerktopologie, bzw. geographische Position der Peers,

bei der Erstellung der Overlaytopologie zu berücksichtigen. Liu et al. [22] schlagen vor, dass Peers basierend auf geographischer Nähe ihre Nachbarn in der Overlaytopologie wählen. Als ausschlaggebend sehen Xu et al. in [23] die Latenz der Peers untereinander an. Sie schlagen vor, dass die Peers mittels Latenzzeiten zu Internet-Landmarken ihre Position bestimmen und anhand dieser Information sich mit nahen Peers verbinden. Eine Übersicht zur möglichen Nutzung von Topologiedaten in strukturierten P2P Overlays gibt [24]. Auch in [25] wird die in Latenz gemessene Nähe zwischen Peers als Kriterium verwendet, um Nachbarn im Overlay zu wählen. Tang et al. betrachten in [26] zusätzlich auch die Mehrfachverwendung von Verbindungen aus unterliegenden Schichten zwischen Overlay Knoten als Kostenmetrik. Weiter gehen noch Han et al. in [27] durch die Betrachtung der Zugehörigkeit der Peers zu einzelnen Internet-Service-Providern (ISP). Das ISP-übergreifende Verkehrsaufkommen gilt es zu minimieren, um sowohl die Latenz der Nachrichten als auch die Kosten für die ISPs zu senken. Der Latenzgewinn und die niedrigere Übertragungslast im Underlay motivieren die Berücksichtigung der Netzwerkinfrastruktur auch für Overlay Multicast Strategien. In [28] stellen Banerjee et al. eine skalierbare anwendungsschichtbasierte Multicastlösung vor, die das Wissen über die unterliegende Topologie nutzt, um den Nachrichtenaufwand zu minimieren und effiziente Multicast-Bäume zu erstellen. In [29] wird die Einführung einer Menge von Routingprimitiven für das Underlay vorgeschlagen, derer sich das Overlay bedienen soll, um anwendungsspezifisch optimiert zu routen.

Bandbreitenverwaltung in den Peers ermöglicht eine Kontrolle des Overlay- als auch Nutzungsverkehrsaufkommens sowie Unterstützung von Dienstgüte von verschiedenen (Overlay-)Nachrichtentypen. Chawathe et al. stellen in [30] Gia vor, eine Erweiterung von Gnutella [8], in der Peers die Anzahl der eingehenden Anfragen mittels periodisch ausgestellten Token regulieren. In [31] werden Scheduling und Active Queue Management Verfahren zur Regulierung des ausgehenden Datenverkehrs vorgestellt. Sie ermöglichen priorisierte Behandlung von unterschiedlichen Datenströmen gemäß Prioritäten für die Latenz und den Datenverlust.

4 Suche

Eine der wesentlichen Aufgaben eines Overlays ist die Lokalisierung von Ressourcen. Die Art der Ressourcen ist vielfältig. Es kann sich um Dateien zum Herunterladen, Benutzer zur Kommunikation, Texte oder andere Arten von Daten und Objekten handeln. Zentralisierte Systeme wie Napster [14] verfolgen für den Suchmechanismus eine Client-Server-Architektur. Vollkommen dezentralisierte P2P-Netzwerke benutzen hingegen auch für die Suche verteilte Algorithmen. Strukturierte und unstrukturierte Overlays verfolgen hierbei grundlegend unterschiedliche Ansätze.

Unstrukturierte Overlays. Der Vorreiter des unstrukturierten Ansatzes, Gnutella [8], benötigte in seiner ursprünglichen Form entweder $O(n)$ Nachrichten pro Suche oder konnte keinerlei Erfolgsgarantien geben. Der Suchaufwand macht den Einsatz von Gnutella in sehr großen Netzen wenig sinnvoll. Durch die Einführung von Supernode-Technologien wurde das Problem der effizienten Suche gemindert, aber nicht gelöst. Neuere Ansätze versuchen, die Lokalisierbarkeit von Daten bei geringem Nachrichtenaufkommen zu ermöglichen. Gia [30] setzt dabei auf 1-Hop-Replikation, Random Walks und selbstoptimierende Topologien. Sashar et al. [32] verwenden einen auf der Perkolationstheorie beruhenden zweistufigen Ansatz für die Suche und eine Daten-Replikation durch Random Walks. Der Ansatz von Ferreira et al. [33] basiert auf dem Geburtstagsparadoxon und repliziert sowohl Daten als auch Suchen mit Random Walks. Bubblestorm [15] verfolgt einen ähnlichen Ansatz, verwendet jedoch Flooding als Kommunikationsmethode. Im Gegensatz zu Gia erreichen die drei letzt genannten Systeme nicht nur das Auffinden eines Treffers, sondern nahezu aller Treffer auf eine Suche.

Strukturierte Overlays. Während bei unstrukturierten Systemen die Datenobjekte als Ganzes repliziert werden, erfolgt bei strukturierten Netzen eine Partitionierung des Suchraums nach Schlüsselwerten, um ein deterministisches Routing zu ermöglichen. So erreicht man zwar eine enorme Effizienz von $O(\log n)$ bei Schlüsselwertsuchen, muss jedoch alle weitergehenden Sucharten auf Basis dieser Indexstruktur implementieren. Bereichssuchen können z. B. mit Präfix- oder Intervall-Bäumen [34] oder mit raumfüllenden Kurven [35] realisiert werden.

Für Volltextsuche kann das Konzept inverser Indizes auf DHTs übertragen werden. Zur Performanzsteigerung wurden eine Vielzahl von Optimierungen vorgeschlagen: Wort-Tupel [36], Bloom-Filter Kompression [37] und Caching [37]. Weitere Vorschläge finden sich in [38].

Die Vollständigkeit, Geschwindigkeit, Lastverteilung und Bandbreitennutzung eines P2P-Suchsystems hängen in starkem Maße von den Umgebungsparametern ab. Yang et al. [39] verglichen die Volltextsuche in mehreren Systemen: Flooding über Supernodes, Random Walk auf unstrukturierten Topologien und DHTs mit inversen Indizes und Bloom-Filtern. DHTs benötigten durch den Index-Aufbau eine deutlich höhere Bandbreite, Random Walks waren bei der Antwortzeit wesentlich langsamer, der Supernodeansatz lag meist zwischen den anderen beiden Konzepten. Li et al. [38] untersuchten die

Leistung von DHT-Volltextsuchsystemen und kamen zu dem Schluss, dass bei den gewählten Parametern selbst mit allen Optimierungen nur die Leistung des ursprünglichen Gnutellas erreicht werden kann. Der Einfluss der Zipf-Verteilung von Dokumentensuchen [40] und ähnlicher statistischer Verteilungen bei Filesharing [41] ist ebenfalls ein Einflussfaktor z. B. wegen Hotspots, der bisher nur unzureichend untersucht ist.

5 Verlässlichkeit

In der Informatik wird Verlässlichkeit wie folgt definiert: *"the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers"* [42].

Replikation stellt ein adäquates Mittel dar, um die Verlässlichkeit von P2P-Systemen zu erhöhen [43]. Im Folgenden werden daher grundsätzliche Replikationsstrategien vorgestellt.

Es gibt zahlreiche Untersuchungen [44], wie oft Dienste/Informationen repliziert werden müssen, um diese effektiv zu finden und zu nutzen. Wir können davon ausgehen, dass die Popularität eines Dienstes oder einer Information einer Zipf-Verteilung folgt [45; 46]. Jedes Objekt i wird auf n_i Knoten gespeichert, die Anzahl aller zu speichernden Objekte ist N, m sei ein beliebiger aber fester Wert und die Anfragewahrscheinlichkeit für ein Objekt i lautet q_i. Davon ausgehend, gibt es typischerweise drei Ansätze:

- **Gleichmäßige Verteilung**: Alle Objekte werden auf der gleichen Anzahl an Knoten gespeichert, d. h. $n_i = \frac{N}{m}$
- **Proportionale Verteilung**: Die Anzahl der Replikas ist proportional zur Anfragewahrscheinlichkeit der jeweiligen Objekte i, d. h. $n_i \propto q_i$
- **Quadratwurzel Verteilung**: Die Anzahl der Replikas ist proportional zur Quadratwurzel der Anfragewahrscheinlichkeit der jeweiligen Objekte i, d. h. $n_i \propto \sqrt{q_i}$

Untersuchungen [44] zeigen, dass im direkten Vergleich der ersten drei Strategien, die Quadratwurzel Verteilung am besten abschneidet; es zeigt sich, dass es sich hierbei um einen theoretisch optimalen Wert handelt, wenn das Qualitätskriterium „Minimierung des Gesamt-Such-Datenverkehrs" lautet.

Die benötigte abzulegende Datenmenge lässt sich noch reduzieren, indem *erasure codes* [47] genutzt werden. Durch ihre Verwendung können Informationen in Fragmente aufgeteilt und wieder hergestellt werden, wenn nur ein genügend hoher Anteil der Fragmente zurückgeholt werden kann.

Ein *erasure code* spaltet eine Datei in eine Menge B mit b gleich großen Blöcken und transformiert diese Blöcke in eine neue Menge C mit c neuen Blöcken (die jeweils die gleiche Größe haben, wie zuvor ein „b-Block"). Optimale *erasure codes* können die Datei aus jeder möglichen Teilmenge C zurückgewinnen, die b Blöcke enthält. Da optimale Codes hohen Ressourcenaufwand erfordern, werden in der Praxis „fast optimale" Kodierungen verwendet, sie benötigen $(1+\varepsilon)\cdot b$ Blöcke, um die Datei zu rekonstruieren. Die Kosten, die auftreten, wenn die Replikas auf diese Weise verteilt werden, sind folglich $\frac{c}{b}$. Die Kosten, die entstehen, um ein Replika aus dem Netz wieder herzustellen sind $(1+\varepsilon)\cdot b$.

Wir können diese Technik verwenden, um beispielsweise die Verfügbarkeit der Daten unter steigendem Nachrichtenoverhead zu erhöhen [48].

Weitere Mechanismen, wie die in [49] beschriebenen, zeigen, wie die Überwachung der Platzierung der einzelnen Datei-Fragmente vermieden werden kann. Sie zeigt deutlich, wie die Verfügbarkeit von Daten verbessert werden kann, indem neue zufällige Fragmente einer Datei erzeugt werden und im Netzwerk abgelegt werden, anstatt alte wiederherzustellen, die verloren gegangen sind.

Das Projekt „PlanetP" [50] zeigt, dass die Verwendung des „erasure codes" in Kombination mit einer zufälligen Fragment-Platzierung (abhängig von einem lokalen Blick auf das Netzwerk) eine Verfügbarkeit von 99,9% erreichen kann, fast unabhängig vom Verhalten der einzelnen Knoten. Es zeigt sich, dass eine festgelegte Anzahl von Repliken zu kostspielig ist und ein kleiner Anteil an hochverfügbaren Knoten die erforderliche Anzahl an Replikas enorm verringert.

6 Sicherheit

Der Begriff Sicherheit im Bezug auf Peer-to-Peer Systeme gliedert sich in die Teilaspekte Datenintegrität, Vertraulichkeit, Authentizität, Verbindlichkeit und Anonymität. Aufgrund der inhärenten Dynamik und Dezentralität von verteilten Systemen lassen sich traditionelle Ansätze zur Lösung dieser Teilaspekte nicht ohne weiteres auf P2P Systeme übertragen [51].

Die Basis für fortgeschrittene Sicherheitsmechanismen (wie z. B. Zugriffs- und Nutzungskontrollen) bildet das Vertrauen zwischen einzelnen Teilnehmern [52]. In P2P Systemen ist das Aufbauen von Vertrauensbeziehungen allerdings ungleich komplizierter, da keine zentrale Anlaufstelle zur Überprüfung von Zertifikaten zur Verfügung steht. Eine Möglichkeit besteht darin, Reputationssysteme als Vertrauensbasis zu verwenden, wobei Informationen über das Verhalten von Nutzern gesammelt und analysiert werden [53; 54]. Die Ausgaben verschiedener Reputationssysteme unterscheiden sich stark. Während [55] lediglich Hinweise gibt, ob ein Peer vertrauenswürdig ist oder nicht, gibt es hingegen Systeme, die Peers den weiteren Zugriff verweigern, falls diese nicht kooperieren [56]. Es können auch Anreize gegeben werden, die den Benutzer zur Kooperation motivieren sollen [57]. Marti et al. geben in [58] eine mögliche Klassifizierung der vielen Reputationssysteme an.

Einen weiteren Schwerpunkt in der P2P Forschung bilden Nutzungs- und Zugriffskontrollen. Diese können die eben beschriebenen Mechanismen zur Vertrauensbildung als Grundlage nehmen. Die Authentifikation der einzelnen Teilnehmer erfolgt üblicherweise über Zertifikate [59]. Da man jedoch nicht davon ausgehen kann, dass zentrale vertrauenswürdige Instanzen zur Überprüfung der Zertifikate vorhanden sind, muss diese Aufgabe von den Peers übernommen werden. Dabei kommen quorum- und gruppenbasierte Ansätze zum Einsatz [60; 61]. Neben der Benutzeridentifikation muss gewährleistet sein, dass die systemweit festgelegten Richtlinien auch eingehalten werden. Hier bieten hardwarebasierte Sicherheitsmechanismen, wie die von der Trusted Computing Group [62], interessante Lösungsmöglichkeiten [63]. Dabei wird über den Prozess der *Remote Attestation* sichergestellt, dass sich die Clientplattform in einem vertrauenswürdigen Zustand befindet. Die Zusammensetzung von Benutzergruppen in P2P Systemen unterliegt einer hohen Dynamik. Daher werden effiziente Mechanismen zur Verwaltung von Gruppenzugehörigkeiten benötigt [64]. Steiner et al. bieten in [65] einen Ansatz, der das Diffie-Hellmann-Schlüsselaustausch-Protokoll von zwei auf eine beliebige Anzahl von Parteien erweitert.

7 Content Distribution

Die Grundform der P2P Content Distribution ist die Bereitstellung eines Systems für verteilte Speicherung, Veröffentlichung, Suche und Konsum digitaler Daten. Neben diesen grundsätzlichen Funktionen werden mittlerweile zusätzlich Sicherheitsfunktionen, Performanceoptimierungen, Ressourcenmanagement und Fairnessfunktionen implementiert [5]. P2P Content Distribution entstand aus der Idee des Dateitauschs. Die ersten Vertreter waren Napster, eDonkey und diverse Clients basierend auf dem Gnutella Protokoll. Der BitTorrent [66] Client Azureus und eMule sind die heutigen Spitzenreiter des P2P Filesharings und belegten im April 2007 Platz 1 und 2 aus über 100 000 Projekten auf der Alltime Download Liste der Open Source Entwicklungswebseite SourceForge. Mittlerweile wurde der reine Dateitausch ergänzt durch eine Reihe neuer Applikationen aus den Gebieten Kommunikation, verteiltes Speichern, anonymer Informationsaustausch und Streaming. Die bekannteste Applikation aus dem Kommunikationsbereich ist die VoIP Software Skype [67], welche mittlerweile über 500 Millionen mal heruntergeladen wurde [68] und von über 5 Millionen Benutzern gleichzeitig verwendet wird. Neben der Sprachübertragung kann P2P Technologie mit Erfolg in der Gruppenkommunikation eingesetzt werden. Groove Networks [69] bietet Projektplanung, Dateisynchronisation und virtuelle Meetings in einem kommerziellen Produkt.

Verteiltes Speichern innerhalb eines P2P Netzes erfolgt durch verschiedene Strategien, bekannt sind OceanStore [70] und Ivy [71].

Anonymer Informationsaustausch wird u. a. von FreeNet [72] und I2P [73] realisiert.

Ziel des P2P Streamings ist es, eine möglichst gleichmäßige Wiedergabe eines Medieninhaltes innerhalb einer gewissen Zeit zu gewährleisten.

Unabhängig von der Anwendungsdomäne müssen alle P2P Applikationen ihren individuellen Quality of Service (QoS) Anforderungen genügen. Diese sind konsistente Datenhaltung und klar verständliche Sprache im Bereich Kommunikation, möglichst gute Bild- und Tonqualität unter Beachtung der Puffergröße bei P2P Streaming, konsistenter und sicherer Datenzugriff bei akzeptabler Zugriffszeit im Bereich Distributed Storage, sowie ein Höchstmaß an Anonymität bei minimaler Einschränkung der Benutzerfreundlichkeit im Falle des anonymen Informationsaustauschs.

Je nach Zielsetzung werden unterschiedliche P2P Overlays verwendet, um optimale QoS Eigenschaften zu erreichen. Distributed Storage z. B. wird aufgrund gezielter Anfragen nach bestimmten Schlüsseln oft mittels einer DHT implementiert. Zum Beispiel nutzt OceanStore das DHT Overlay Tapestry [18]. Dies ermöglicht lokale Platzierung und schnelle Suche nach Objekten. Im Bereich Kommunikation zeigten sich Supernode Architekturen wie bei Skype skalierend und robust.

Neben der Auswahl eines geeigneten P2P Overlays spielt bei Filesharing Applikationen der Verteilungsalgorithmus für Dateien eine große Rolle. Auswertungen des Internet Datenverkehrs der Firma CacheLogic zeigten, dass der Hauptanteil der über P2P ausgetauschten Dateien mindestens 1 GB groß sind [74]. Die Übertragung von Dateien dieser Größe erfolgt in kleinen Stücken, den so genannten Chunks, welche je nach Implementierung bis zu 10 MB groß sind und die atomare Übertragungseinheit in P2P Netzwerken bilden. Da Teilnehmer in P2P Netzwerken eine geringere Verfügbarkeit als Server besitzen, ist die Wahl eines geeigneten Chunk Selection Algorithmus wichtig, um immer einen möglichst guten Verbreitungsgrad der Informationen aber auch eine zügige Auslieferung an Peers zu garantieren. Mögliche Kriterien zur Chunk Selection sind [75]:

- **Sequential**: Die Chunks werden nacheinander angefordert.
- **Rarest**: Der seltenste Chunk wird ausgewählt.
- **Random**: Es wird ein zufälliger Chunk gewählt.

Die Auswahl der *Random* und der *Rarest* Strategie ist effizienter als die sequenzielle Abarbeitung der Chunks. Die *Rarest* Strategie garantiert darüber hinaus eine lange Vorhaltung der vollständigen Verfügbarkeit der Datei.

Um die Akzeptanz für P2P Content Distribution sowohl bei den Entwicklern, als auch bei den End-

benutzern weiter zu erhöhen, sollten allgemeine P2P Frameworks entwickelt werden. Eine erste Implementierung basierend auf Supernode Technologie eines vielseitig verwendbaren P2P Frameworks ist im Rahmen des Projekts JXTA [12] entstanden.

8 Fazit

Peer-to-Peer ist ein fundamentales Designprinzip und stellt einen Paradigmenwechsel für die Kommunikation in Computernetzwerken dar. In diesem Beitrag haben wir die Definitionen sowie die relevanten Arbeiten und Bestrebungen im Bereich P2P vorgestellt. Für einen erfolgreichen und breiten Einsatz des P2P-Paradigmas in der Praxis ist es unerlässlich, dass die Qualität der P2P-Mechanismen in den hier vorgestellten Forschungsgebieten weiter verbessert wird. Ein Forschungsprojekt, das hier einen ganzheitlichen und systematischen Ansatz verfolgt, ist die DFG Forschergruppe *„Verbesserung der Qualität von Peer-to-Peer-Systemen durch die systematische Erforschung von Qualitätsmerkmalen und deren wechselseitigen Abhängigkeiten"* an der Technischen Universität Darmstadt [76]. In ihrem Rahmen entstand diese Arbeit.

Literatur

[1] G. Hasslinger, *Peer-to-Peer Systems and Applications*, ser. LNCS. Springer, 2005, no. 3485, pp. 369–382.

[2] A. Oram, Ed., *Peer-To-Peer: Harnessing the Power of Disruptive Technologies.* O'Reilly & Associates, 2001.

[3] R. Steinmetz, K. Wehrle, *Peer-to-Peer Systems and Applications*, ser. LNCS. Springer, 2005, vol. 3485, ch. What is Peer-to-Peer About?, pp. 9–16.

[4] R. Steinmetz, K. Wehrle, "Peer-to-Peer-Networking and -Computing", *Informatik Spektrum*, vol. 27, no. 1, pp. 51–54, Feb 2004.

[5] S. Androutsellis-Theotokis, D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies", *ACM Comput. Surv.*, vol. 36, no. 4, pp. 335–371, 2004.

[6] I. Stoica et al., "Chord: a Scalable Peer-to-Peer Lookup Service for Internet Applications", in *SIGCOMM*, 2001.

[7] P. Maymounkov, D. Mazieres, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric", http://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.ps.

[8] Clip2, "The gnutella protocol specification v0.4n", 2000, http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

[9] "RFC-Gnutella 0.6", http://rfc-gnutella.sourceforge.net/developer/testing.

[10] http://www.fasttrack.nu.

[11] http://www.kazaa.com.

[12] Sun Microsystems, "JXTA", http://www.jxta.org, 2004.

[13] A. Kovacevic, N. Liebau, R. Steinmetz, "Globase.KOM - A P2P Overlay for Fully Retrievable Location-based Search", *IEEE P2P*, 2007.

[14] http://www.napster.com.

[15] W. W. Terpstra, J. Kangasharju, C. Leng, A. P. Buchmann, "BubbleStorm: Resilient, Probabilistic, and Exhaustive Peer-to-Peer Search", SIGCOMM, 2007.

[16] R. Dingledine, M. J. Freedman, D. Molnar, "The Free Haven Project: Distributed Anonymous Storage Service", in *Workshop on DIAU*, 2000, pp. 67–95.

[17] S. Ratnasamy et al., "A Scalable Content-Addressable Network", in *SIGCOMM*, 2001.

[18] B. Zhao et al., "Tapestry: A Resilient Global-Scale Overlay for Service Deployment", *IEEE Journal on Selected Areas in Comm.*, 2003.

[19] A. Rowstron, P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems", in *ACM IFIP*, 2001.

[20] D. Malkhi, M. Naor, D. Ratajczak, "Viceroy: a Scalable and Dynamic Emulation of the Butterfly", in *ACM PODC*, 2002.

[21] V. Darlagiannis, "Overlay Network Mechanisms for Peer-to-Peer Systems", Ph.D. dissertation, TU Darmstadt, 2005.

[22] Y. Liu et al., "Location-Aware Topology Matching in P2P Systems," in *IEEE INFOCOM*, 2004.

[23] Z. Xu, C. Tang, Z. Zhang, "Building Topology-Aware Overlays Using Global Soft-State", in *IEEE ICDCS*, 2003.

[24] M. Castro et al., "Topology-Aware Routing in Structured Peer-to-Peer Overlay Networks", ser. LNCS, vol. 2584. Springer, 2003.

[25] S. Ratnasamy et al., "Topologically-Aware Overlay Construction and Server Selection", in *INFOCOM*, 2002.

[26] C. Tang et al., "Underlay-Aware Design of Overlay Topologies and Routing Algorithms", Michigan State University, US, Tech. Rep. MSU-CSE-04-9, 2004.

[27] J. Han, D. Watson, F. Jahanian, "Topology Aware Overlay Networks", in *IEEE INFOCOM*, 2005.

[28] S. Banerjee, B. Bhattacharjee, C. Kommareddy, "Scalable Application Layer Multicast", in *SIGCOMM*, 2002.

[29] A. Nakao, L. L. Peterson, A. C. Bavier, "A Routing Underlay for Overlay Networks", in *SIGCOMM*, 2003.

[30] Y. Chawathe et al., "Making Gnutella-like P2P systems scalable", in *SIGCOMM '03*.

[31] Graffi et al., "Overlay Bandwidth Management: Scheduling and Active Queue Management of Overlay Flows", in *IEEE LCN*, 2007.

[32] N. Sarshar et al., "Percolation Search in Power Law Networks: Making Unstructured Peer-to-Peer Networks Scalable", in *IEEE P2P*, 2004.

[33] R. A. Ferreira et al., "Search with Probabilistic Guarantees in Unstructured Peer-to-Peer Networks", in *IEEE P2P*, 2005.

[34] S. Ramabhadran et al., "Brief Announcement: Prefix Hash Tree", in *ACM PODC*, 2004.

[35] C. Schmidt, M. Parashar, "Enabling Flexible Queries with Guarantees in P2P Systems", *IEEE Internet Computing*, vol. 8, no. 3, 2004.

[36] O. D. Gnawali, "A Keyword Set Search System for Peer-to-Peer Networks", Master's thesis, Massachusetts Institute of Technology, 2002.

[37] P. Reynolds, A. Vahdat, "Efficient Peer-to-Peer Keyword Searching", in *Proc. of Int. Middleware Conf.*, 2003.

[38] J. Li et al., "On the Feasibility of Peer-to-Peer Web Indexing and Search", in *IEEE IPTPS*, 2003.

[39] Y. Yang et al., "Performance of Full Text Search in Structured and Unstructured

Peer-to-Peer Systems", in *INFOCOM '06*, 2006.
[40] L. Breslau et al., "Web Caching and Zipf-like Distributions: Evidence and Implications", in *INFOCOM*, 1999.
[41] K. P. Gummadi et al., "Measurement, Modeling and Analysis of a Peer-to-Peer File-Sharing Workload", in *ACM SOSP*, 2003.
[42] IFIP WG 10.4, "Dependable Computing and Fault Tolerance." [Online]. Available: http://www.dependability.org/.
[43] O. Heckmann et al., "Qualitätsmerkmale von Peer-to-Peer-Systemen", TU Darmstadt, Tech. Rep. KOM-TR-2006-03, 2006.
[44] Q. Lv et al., "Search and Replication in Unstructured Peer-to-Peer Networks", in *SIGMETRICS*, 2002.
[45] V. Almeida et al., "Characterizing Reference Locality in the WWW," in *ACM PDIS*, 1996.
[46] K. Sripanidkulchai, "The Popularity of Gnutella Queries and its Implications on Scalability", Cernegie Mellon University, Tech. Rep., 2001.
[47] H. Weatherspoon, J. Kubiatowicz, "Erasure Coding vs. Replication: A Quantitative Comparison", in *IPTPS*, 2002.
[48] W. K. Lin et al., "Erasure code replication revisited", in *Peer-to-Peer Computing*, 2004.
[49] F. M. Cuenca-Acuna, "A Probabilistic Approach to Building Large Scale Federated Systems", Ph.D. dissertation, Rutgers Univ., 2004.
[50] F. M. Cuenca-Acuna, R. P. Martin, T. D. Nguyen, "Autonomous Replication for High Availability in Unstructured P2P Systems", in *IEEE SRDS '03*, 2003.
[51] N. Daswani, H. Garcia-Molina, B. Yang, "Open Problems in Sata-Sharing Peer-to-Peer Systems", in *ICDT '03*, 2003.
[52] D. S. Wallach, "A Survey of Peer-to-Peer Security Issues", in *ISSS '02*, 2002.
[53] S. D. Kamvar, M. T. Schlosser, H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks", in *ACM WWW*, 2003.
[54] A. Singh et al., "Trustme: Anonymous Management of Trust Relationships in Decentralized P2P Systems", in *IEEE P2P*, 2003.
[55] A. Abdul-Rahman, S. Hailes, "A Distributed Tust Model", in *NSPW*, 1997.
[56] M. Srivatsa, L. Xiong, L. Liu, "Trustguard: Countering Vulnerabilities in Reputation Management for Decentralized Overlay Networks", in *ACM WWW '05*, 2005, pp. 422–431.
[57] Y. Tang, H. Wang, W. Dou, "Trust Based Incentive in P2P Network", in *IEEE CEC-EAST '04*, 2004.
[58] S. Marti, H. Garcia-Molina, "Taxonomy of Trust: Categorizing P2P Reputation Systems", *Comput. Networks*, vol. 50, no. 4, 2006.
[59] R. Sandhu, X. Zhang, "Peer-to-Peer Access Control Architecture using Trusted Computing Technology", in *ACM SACMAT '05*, 2005.
[60] N. Liebau et al., "A Secure Quorum Based Membership Mechanism for P2P Systems", in *AMCIS*, 2006.
[61] S. Gokhale, P. Dasgupta, "Distributed Authentication for Peer-to-Peer Networks", in *IEEE SAINT*, 2003.
[62] The Trusted Computing Group, http://www.trustedcomputinggroup.org.
[63] S. Balfe et al., "Trusted Computing: Providing Security for Peer-to-Peer Networks", in *IEEE P2P*, 2005.
[64] L. Pearlman et al., "A Community Authorization Service for Group Collaboration", in *IEEE POLICY*, 2002.
[65] M. Steiner, G. Tsudik, M. Waidner, "CLIQUES: A new Approach to Group Key Agreement", in *IEEE ICDCS*, 1998.
[66] B. Cohen, "Incentives Build Robustness in BitTorrent", in *P2PECON '03*, 2003.
[67] http://www.skype.com.
[68] Skype Limited, "Celebrating 500M Downloads", http://about.skype.com/2007/03/celebrating_500m_downloads_sky.html, Luxembourg.
[69] Groove Networks, "Groove Virtual Office", 2005, http://www.groove.net.
[70] J. Kubiatowicz et al., "Oceanstore: An Architecture for Global-Scale Persistent Storage", in *ACM ASPLOS '00*, 2000, pp. 190–201.
[71] A. Muthitacharoen et al., "Ivy: A Read/Write Peer-to-Peer File System", in *OSDI*, 2002.
[72] I. Clarke et al., "Freenet: A Distributed Anonymous Information Storage and Retrieval System", in *Workshop on DIAU*, 2000.
[73] I2P, http://www.i2p.net.
[74] D. Fergusson, "Trends and Statistics in Peer-to-Peer", in *Workshop on Tech. and Legal Aspects of P2P TV*, 2006.
[75] D. Kostic, "Maintaining High-Bandwidth under Dynamic Network Conditions", in *USENIX '05*, 2005.
[76] DFG Forschergruppe, "Verbesserung der Qualität von Peer-to-Peer-Systemen durch die systematische Erforschung von Qualitätsmerkmalen und deren wechselseitigen Abhängigkeiten (QuaP2P)", http://www.quap2p.tu-darmstadt.de, June 2006.

Autoren von links nach rechts und von oben nach unten: J. Schröder-Bernhardi, M. Benz, A. Kovacevic, K. Graffi N. Liebau, D. Bradler, P. Mukherjee, C. Leng

1 **Dipl.-Math., Dipl.-Inf. Kálmán Graffi** studied computer science and mathematics at TU Darmstadt. He graduated with both Dipl.-Inf. and Dipl.-Math. degrees in July 2006 with his thesis "A Security Framework for Organic Mesh Networks". Since June 2006 he is PhD student at Multimedia Communication Lab (KOM) at the Department of Electrical Engineering and Information Technology of TU Darmstadt. The focus of his research is efficiency management in P2P systems.
Address: Technische Universität Darmstadt, Merckstraße 25, 64283 Darmstadt, Deutschland, Tel.: +49-6151-164959,
E-Mail: graffi@kom.tu-darmstadt.de

2 **Dipl.-Ing. Aleksandra Kovačević** is a PhD student at Multimedia Communication Lab of TU Darmstadt. She graduated from the School of Electrical Engineering, University of Belgrade in 2004. Her research interests include peer-to-peer networks, particulary in location-based search, the investigation scalability and stability of overlay networks, as well as the simulation of large scale peer-to-peer networks.

Address: Technische Universität Darmstadt, Merckstraße 25, 64283 Darmstadt, Deutschland, Tel.: +49-6151-166886, E-Mail: sandra@kom.tu-darmstadt.de

3 **Dipl.-Inf. Patrick Mukherjee** is a PhD student at Real-Time Systems Lab at TU Darmstadt. He graduated in Computer Science at the Technical University of Berlin in October 2004. His research focuses on (semantic) correctness of versioned data objects distributed in peer-to-peer networks. He focuses in the building of peer-to-peer based platforms to support global project development.
Address: Technische Universität Darmstadt, Merckstraße 25, 64283 Darmstadt, Deutschland, Tel.: +49-6151-163776, E-Mail: patrick.mukherjee@es.tu-darmstadt.de

4 **Dipl.-Inf. Michael Benz** studied computer science at TU Darmstadt. He graduated with a Dipl. Inform. degree in December 2005 with his thesis "Cluster based rendering with distributed frame buffers". Currently he is a PhD student at the IT-Security Group at the Department of Computer Science of TU Darmstadt focusing on security issues in P2P systems.
Address: Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Deutschland, Tel.: +49-6151-164546, E-Mail: benz@sec.informatik.tu-darmstadt.de

5 **Dipl.-Inf. Christof Leng** studied computer science at TU Darmstadt where he graduated in September 2004. He has worked as a software developer and freelance consultant during his studies and afterwards. Since December 2005 he is a PhD student at the Databases and Distributed Systems Group (DVS) at the Department of Computer Science of TU Darmstadt. The current focus of his research is the reliable evaluation of complex queries in unstructured and structured P2P overlays.
Address: Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Deutschland, Tel.: +49-6151-167425, E-Mail: cleng@dvs1.informatik.tu-darmstadt.de

6 **Dipl. wirtsch.-Inf. Dirk Bradler** studied joint computer science and business economics at the TU Darmstadt. He graduated with a Dipl. Wirtsch.-Inf. degree in 2006 with his thesis "Deploying and Managing Distributed Services in a Peer-to-Peer Network". Currently he is a PhD student at the Telecooperation Group at the Department of Computer Science of TU Darmstadt. The focus of his research is to enrich the quality of P2P networks with the help of context information.
Address: Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Deutschland, Tel.: +49-6151-165245, E-Mail: bradler@tk.informatik.tu-darmstadt.de

7 **Dipl.-Inf. Julian Schröder-Bernhardi** studied computer science at TU Darmstadt where he graduated with a Dipl.-Inf. degree in 2006. Currently he is a PhD student at the Telecooperation Group at the Department of Computer Science of TU Darmstadt. The focus of his research is improving the dependability of peer-to-peer infrastructures and to provide metrics for the covered subareas availability, reliability, robustness and fault tolerance.
Address: Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Deutschland, Tel.: +49-6151-165245, E-Mail: jschroed@tk.informatik.tu-darmstadt.de

8 **Dipl.-Wirtsch.-Ing. Nicolas Liebau** studied joint electrical engineering and business economics at TU Darmstadt and received his diploma degree in 2002. From 2002 to 2005 he received a scholarship from the Deutsche Forschungsgemeinschaft for doing his PhD at the Multimedia Communications Lab (KOM) with Prof. Ralf Steinmetz. Since 2005 he is working as a researcher at the Multimedia Communications Lab (KOM). Since 2006 he is heading the Peer-to-Peer Networking group of KOM at TU Darmstadt.
Address: Technische Universität Darmstadt, Merckstraße 25, 64283 Darmstadt, Deutschland, Tel.: +49-6151-166240, E-Mail: liebau@kom.tu-darmstadt.de