H P2P-Systeme

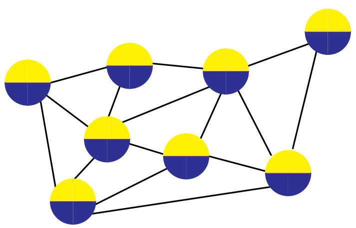
H.1 Überblick

- Einführung
- Napster ein hybrides P2P-System
- Gnutella ein unstrukturiertes P2P-System
- Strukturierte P2P-Systeme / Verteilte Hash-Tabellen
- JXTA eine Infrastruktur für P2P-Systeme

H.2 Einführung

1 Ausgangssituation

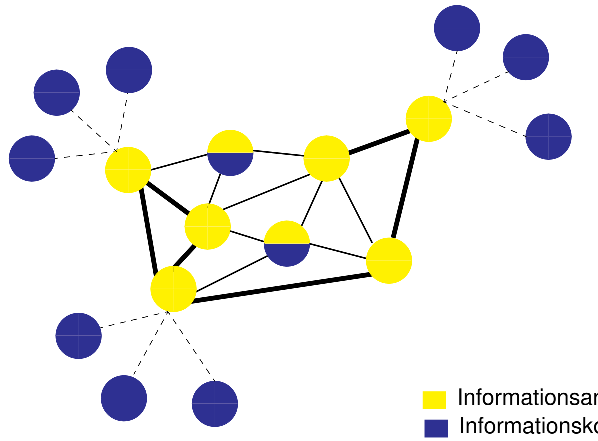
- Grundform des Internet (1969-1995)



Informationsanbieter
Informationskonsument

1 Ausgangssituation

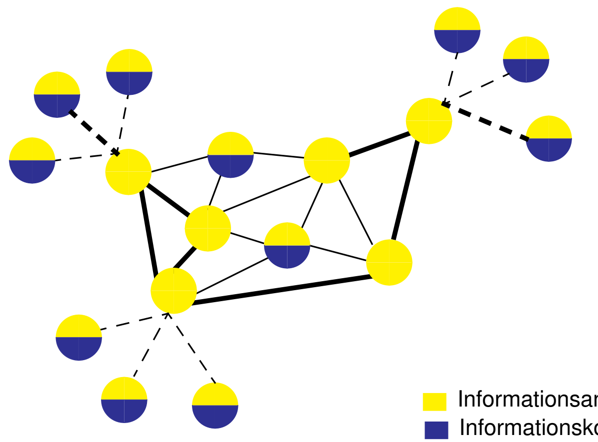
- WWW-dominiertes Internet (1995-1999)



Informationsanbieter
Informationskonsument

1 Ausgangssituation

- Peer-to-Peer Internet (ab 1999)



Informationsanbieter
Informationskonsument

1 Ausgangssituation

- Nachteile der Client-Server-Architektur
  - ◆ Verkehrsaufkommen
    - ➤ Konzentration beim Server, Unterlast in anderen Netzteilen
    - ➤ Asymmetrischer Verkehr
  - ◆ Skalierbarkeit
  - ◆ Ungenutzte Ressourcen in Clients
    - ➤ Speicherplatz, Rechenleistung, Informationen
  - ◆ Server oft Single Point of Failure
- Copyright

1 Was ist Peer-to-Peer?

- Ein System kann als Peer-to-Peer bezeichnet werden wenn,
  - ◆ eine gemeinsame Nutzung von Ressourcen statt findet
    - ➤ Jedes Teilsystem kann sowohl Informationsanbieter als auch Informationskonsument sein
  - ◆ jedes Teilsystem einen gewissen Grad an Autonomie besitzt
    - ➤ Keine zentrale Kontrolle oder Nutzung von zentralen Diensten
  - ◆ die einzelnen Teilsysteme nicht permanent mit dem Gesamtsystem verbunden sein müssen
    - ➤ Selbstorganisation des Systems
  - ◆ die einzelnen Teilsysteme keine permanente Netzwerkadresse benötigen
    - ➤ Von der Netzwerkschicht unabhängige Adressierung

1 Was ist Peer-to-Peer?

- Eine der vielen Definitionen (vgl. Tutorial KIVS 2003)

  Selbstorganisierendes System gleichberechtigter, autonomer Systeme ohne Nutzung zentraler Dienste auf der Basis eines unzuverlässigen Netzwerks

- Anforderungen an P2P-Systeme
  - ◆ Skaliert in einem globalen Rahmen
  - ◆ Einfaches publizieren von Informationen für eine beliebige Anzahl von Informationskonsumenten
  - ◆ Schnelles und effizientes Finden von Informationen
  - ◆ Teilnehmende Peers können zu beliebigen Zeitpunkten dem Gesamtsystem beitreten und es wieder verlassen

2 Klassifikation

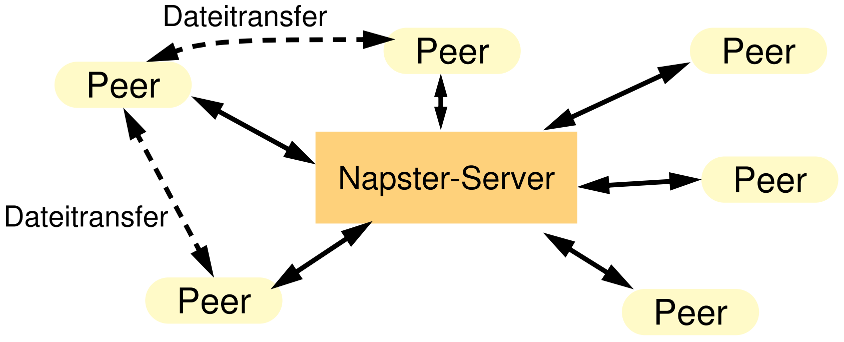
- **Client-Server:** WWW, Seti@Home
  - ◆ Klassische Rollenverteilung
  - ◆ Keine Interaktion zwischen Clients
- ***Hybride* P2P-Systeme:** Napster, ICQ, AIM
  - ◆ Gemeinsame Nutzung von verteilten Ressourcen
  - ◆ Interaktion zwischen Peers
  - ◆ Koordination durch zentrale Server
- ***Reine* P2P-Systeme:**
  - ◆ Vollständige dezentale Organisation und Nutzung der Ressourcen
  - ◆ Unstrukturierte P2P-Systeme (z. B. Gnutella)
  - ◆ Strukturierte P2P-Systeme (z. B. Systeme basierend auf verteilten Hash-Tabellen wie Chord)

3 Literatur

- Informationen über P2P-Systeme und Applikationen
  - Allgemeine Informationen über P2P-Systeme
    - http://openp2p.com
  - International Workshop on Peer-to-Peer Systems
    - http://www.cs.rice.edu/Conferences/IPTPS02
    - http://iptps03.cs.berkeley.edu/program.html
    - http://iptps04.cs.ucsd.edu/
  - Hauptseminar unseres Lehrstuhls
    - http://www4.informatik.uni-erlangen.de/Lehre/SS02/HS_DOOS
    - http://www4.informatik.uni-erlangen.de/Lehre/SS04/HS_P2P
  - Bücher
    - Peer-to- Peer: Harnessing the Benefits of Disruptive Technology, O'Reilly & Associates, 2001

H.3 Napster

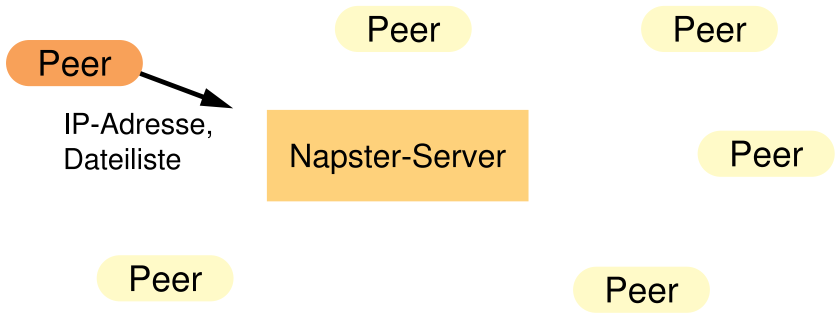
- Erstes populäres P2P-System (1999-2001)
  - Austausch von Musik-Dateien



  - Zentraler Verzeichnisserver
    - Verwaltet Adressen und Dateilisten der Peers

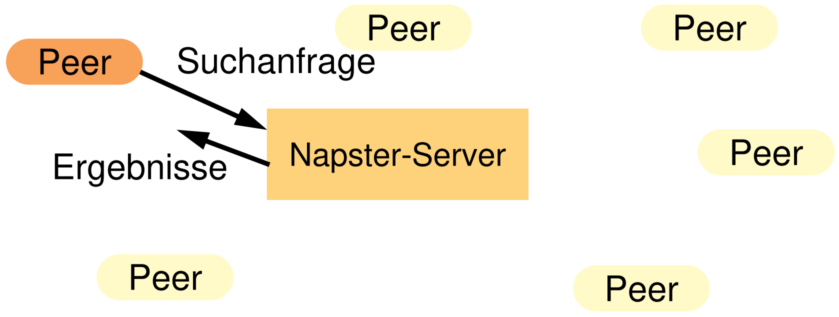
1 Protokoll - Verbindungsaufbau

- Verbindungsaufbau zum Verzeichnisserver
  - Peers geben eine Liste der gespeicherten Dateien und ihre IP-Adresse an den Server bekannt



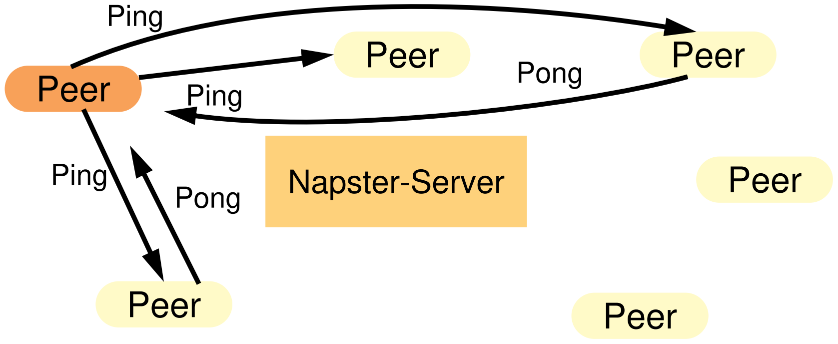
2 Protokoll - Dateisuche

- Anfrage an den Verzeichnisserver
  - Verzeichnisserver sucht im Datenbestand und liefert eine Liste von IP-Adressen



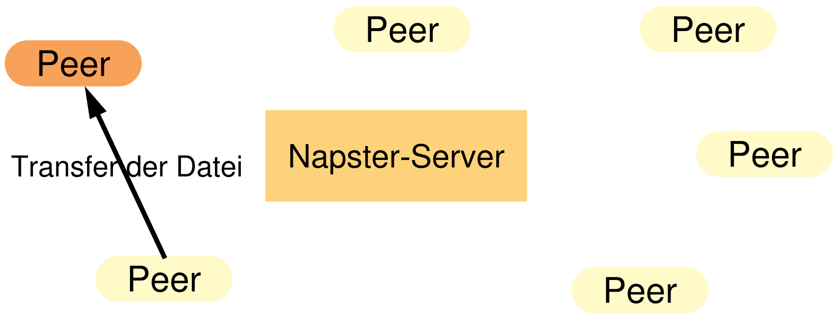
3 Protokoll - Dateitransfer

- Testen der potentiellen Zielrechner
  - Durch Ping-Nachrichten wird die Verbindungsqualität der Zielrechner getestet



3 Protokoll - Dateitransfer

- Übertragung der gesuchten Dateien
  - Benutzer baut direkte Verbindung zu dem Peer mit der besten Verbindung auf

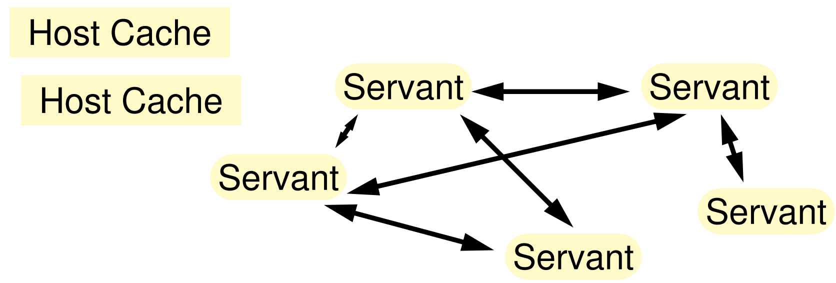


4 Zusammenfassung

- Eigenschaften
  - Skalierbarkeit ist eingeschränkt durch zentralen Verzeichnisserver
  - Verzeichnisserver ist Single Point of Failure
- Hybrides P2P-System
- Erweiterungen
  - Mehrere vernetzte Verzeichnisserver (OpenNap-Netzwerk)
  - Unterstützung für beliebige Dateiformate

H.4 Gnutella

- Ein Protokoll zur verteilten Suche nach Dateien
- Ursprünglich von Nullsoft als eine Art freies Napster entwickelt
- Literatur:
  - The Gnutella Protocol Specification v0.4



1 Grundlage

- Rahmenbedingungen:
  - Zur Teilnahme an einem Gnutella-Netzwerk benötigt ein Servant mindestens eine IP-Adresse eines bereits teilnehmenden Knotens
  - Nachrichten werden über TCP-Verbindungen als ASCII-Text ausgetaucht
  - In der Regel verfügt jeder Servant über mehrere ständige Verbindungen
  - Jeder Teilnehmer kann frei entscheiden welche Dateien dem Netzwerk zur Verfügung gestellt werden
  - Das Protokoll dient zur Suche von Dateien. Der eigentliche Transfer von Dateien wird über HTTP abgewickelt

2 Verbindungsaufbau

- Ermittlung der IP-Adresse eines aktiven Knotens zum Beispiel durch einen Host Cache
- Verbindungsaufbau erfolgt mittels eine *Connect*-Nachricht die durch den kontaktierten Knoten mit einer *Ok*-Nachricht beantwortet wird
- Im folgenden werden dann die eigentlichen Basisnachrichten über die permanente Verbindung ausgetauscht

3 Nachrichten - Erhaltung der Netzwerkstruktur

- *Ping* - wird verwendet um neue aktive Peers kennen zu lernen. Eine solche Nachricht kann zu einem beliebigen Zeitpunkt an andere Peers versendet werden.
- *Pong* - Anwort auf eine Ping-Nachricht die als Payload Informationen über einen aktiven Knoten enthält
  - IP-Adresse und Port des antwortenden Rechners
  - Anzahl der bereitgestellten Dateien
  - Größe der bereitgestellten Daten

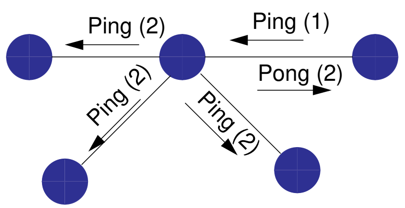
3 Nachrichten - Suche nach Dateien

- *Query* - Eine Suchanfrage nach Dateien
  - Gesuchte Zeichenkette
  - Mindestanforderungen an die Übertragungsgeschwindigkeit an den bereitstellenden Peer
- *QueryHit* - Antwort auf eine Suchanfrage falls ein Peer Dateien mit passenden Namen bereitstellt
  - Port und IP-Adresse des antwortenden Peers
  - Übertragungsrate mit der Dateien angeboten werden
  - Liste von Dateinamen, sowie einer eindeutigen ID pro Datei und ihrer jeweiligen Größe
  - Eine eindeutige ID die den bereitstellenden Peer indentifiziert. Diese wird durch eine Funktion über die IP-Adresse gebildet.

4 Dateitransfer

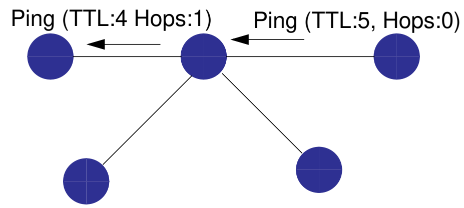
- Dateitransfer erfolgt nicht über das Gnutella-Netzwerk sondern via HTTP
  - Hierbei wird der Dateiname und die ID der zu übertragenden Datei über eine HTTP Get-Nachricht an den bereitstellenden Peer übermittelt welcher als Antwort die Datei liefert

5 Routing - Erhaltung der Netzwerkstruktur:



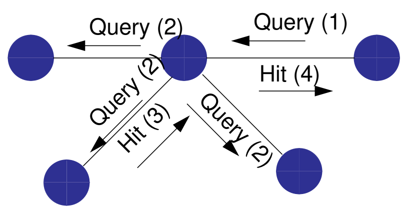
- Empfängt ein Peer eine Ping Nachricht, wird diese an alle verbundenen Peers weitergeleitet
- Jeder Peer der eine Ping Nachricht erhält kann mit einer Pong-Nachricht antworten. Diese wird dann zurück an den Initiator geroutet.
- Erhält ein Peer eine bereits beantwortete Ping Nachricht wird sie verworfen. Dies wird durch einen *Message-Cache* ermöglicht der die Nachrichten kurzzeitig zwischenspeichert.

5 Routing - Erhaltung der Netzwerkstruktur



- Jede Nachricht verfügt über die Felder Hops und TTL (Time To Live)
- Ein Peer der eine Nachricht weiterleitet inkrementiert das Feld Hops und dekrementiert das Feld TTL
- Sinkt das Feld TTL auf den Wert 0 wird die Nachricht nicht mehr weitergeleitet

5 Routing - Suche



- Empfängt ein Peer eine Query-Nachricht wird diese wie eine Ping-Nachricht an alle verbundenen Peers weitergeleitet
- Jeder Peer der eine Query-Nachricht erhält kann mit einer Query-Hit Nachricht antworten. Diese wird dann zurück an den Initiator geroutet.
- Erhält ein Peer eine bereits beantwortete Query-Nachricht wird sie verworfen. Jeder Peer der eine Nachricht weiterleitet inkrementiert das Feld Hops und dekrementiert das Feld TTL.

H.4 Zusammenfassung

- Unstrukturiertes P2P-System
- Skalierungsprobleme
  - ◆ Suchanfragen und Nachrichten zur Erhaltung der Netzwerkstruktur werden als Broadcast-Nachrichten versendet.
  - ◆ Die Konsequenz ist die Flutung des Netzwerkes und damit eine Überlastung von Knoten mit schmalbandigen Verbindungen.
  - ◆ Folge Abbruch von Verbindungen und Fragmentierung bzw. sogar Kollapse des Netzwerkes.
- Weitere Entwicklung: *Super Peers*
  - ◆ Einfache Peers haben nur Verbindungen zu einigen wenigen Super Peers
  - ◆ Einfache Peers teilen einem Super Peer ihre IP-Adresse und Dateiliste mit
  - ◆ Super Peers agieren als Proxy für einfache Peers
    - • Anfragen an einfache Peers können von Super Peers beantwortet werden
    - • Entlastung der einfachen Peers

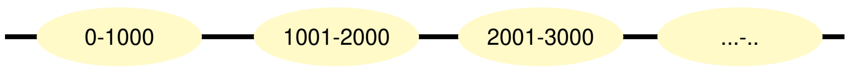
H.5 Verteilte Hash-Tabellen

1 Übersicht

- Grundidee verteilter Hash-Tabellen
  - ◆ Verteilung von Daten über alle Knoten nach einem Algorithmus der eine Topologie erzeugt (z.B. Ring)
  - ◆ Anforderung der Daten durch eine Anfrage beim verantwortlichen Knoten
- Zielsetzungen
  - ◆ Gleichmäßige Verteilung der Daten auf alle Knoten
  - ◆ Ständige Anpassung bei Ausfall, Beitritt und Austritt von Knoten
    - • Vergabe von Zuständigkeiten an neue Knoten
    - • Übernahme und Neuverteilung von Zuständigkeiten bei Austritt und Ausfall

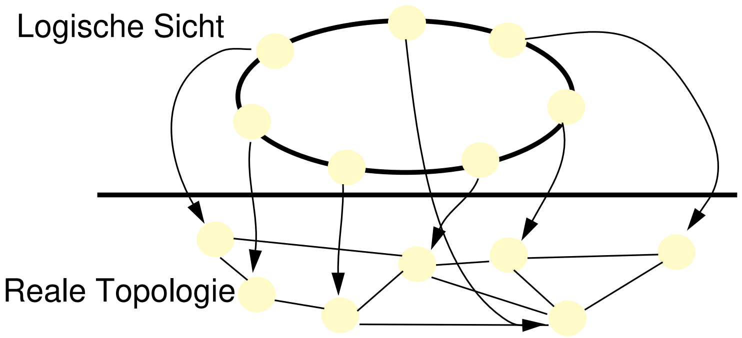
2 Prinzipielle Arbeitsweise

- Abbildung der Inhalte auf einen linearen Wertebereich durch eine Hashfunktion
  - ◆ Meist 0,... 2^{m-1} >> Anzahl der gespeicherten Objekten
- Verteilung des Schlüsselraums über alle Knoten durch Abbildung der Knoten Identifikatoren in den Wertebereich



2 Prinzipielle Arbeitsweise

- Jeder der Knoten ist mindestens für einen Teil des Schlüsselraumes zuständig
  - ◆ Oftmals sind auch mehrere Knoten für den gleichen Bereich verantwortlich
  - ◆ Die Zuständigkeit kann sich dynamisch ändern



2 Prinzipielle Arbeitsweise

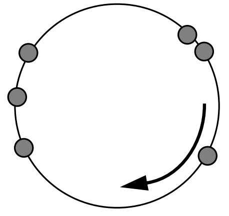
- Ablage von Daten
  - ◆ Erzeugung eines Schlüssel
  - ◆ Routing der Daten zu verantwortlichem Knoten
- Auffinden von Daten
  - ◆ Einstieg bei einem beliebigen Knoten und Suche nach Schlüssel
  - ◆ Routing zu gesuchten Daten

2 Prinzipielle Arbeitsweise

- Beitritt eines neuen Knotens
  - ◆ Zuteilung eines bestimmten Bereichs des Schlüsselraums
  - ◆ Initialisierung mit Routing-Informationen und Einbindung in die Routing-Strukturen
- Austritt eines Knotens
  - ◆ Aufteilung des Schlüsselraums auf benachbarte Knoten
  - ◆ Migration der Daten auf zuständige Knoten
- Ausfall eines Knotens
  - ◆ Nutzung redundanter Routing-Wege und Knoten
  - ◆ Erneute Zuweisung des Schlüsselraums

2 Chord

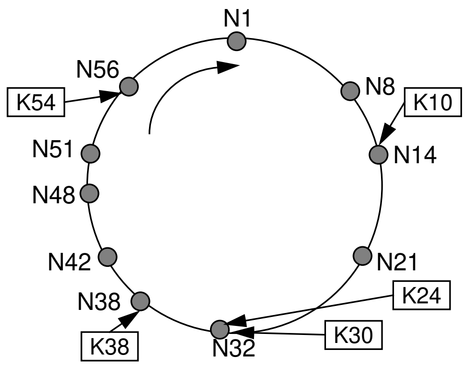
- Projektseite: http://www.pdos.lcs.mit.edu/chord
- Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan, *Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications*. IEEE Transactions on Networking
- Grundidee: Chord-Ring
  - ◆ Jeder Knoten besitzt eine eindeutige 160 Bit ID. Diese ist das Resultat einer Hashfunktion über die IP-Adresse des Knotens.
  - ◆ Ringförmige Anordnung der Knoten geordnet nach ihren IDs



2 Chord

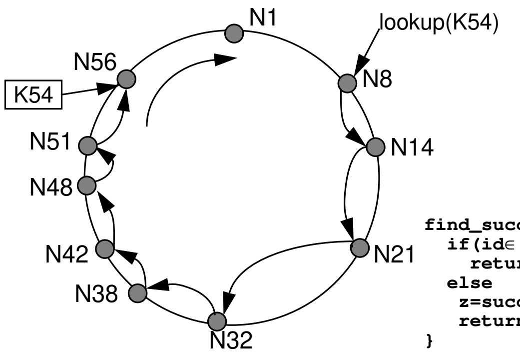
- Abbildung der Informationen auf die einzelnen Knoten:
  - ◆ Für jede Information wird eine eigene Hash-ID erzeugt
  - ◆ Die Information wird dem Knoten mit der unmittelbar im Ring folgenden ID übergeben

Beispiel: Anzahl der Knoten 10, 5 verwaltete IDs



2 Chord

- Naive Methode um Daten zu finden in einem Chord-Ring
  - ◆ Jeder Knoten kennt seinen unmittelbaren Nachfolger und leitet eine Anfrage solange weiter, bis der gesuchte Schlüssel zwischen der eigenen ID und der des unmittelbaren Nachfolgers liegt. Dieser Nachfolger verwaltet den gesuchten Schlüssel.



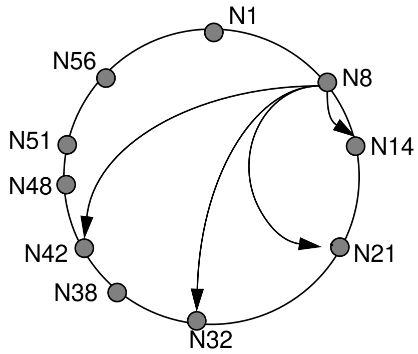

```
find_succesor(id) {
  if(id∈ (n,succesor])
    return successor
  else
   z=successor.find_successor(id)
   return z
}
```


2 Chord

- Für die skalierbare Lokalisierung von Daten verfügt jeder Knoten über zusätzliche Routing-Informationen:
  - ◆ Eine *Finger*-Tabelle, für die gilt, dass ein Finger in der Zeile k dem ersten Knoten im Ring nach der ID mit dem Wert
    $(n+2^{k-1}) \bmod 2^m$, $1 \le k \le m$ entspricht.
    Außer der ID des Knoten wird zusätzlich die IP-Adresse des Rechners verwaltet.
  - ◆ Die ID und die IP-Adresse des unmittelbaren Nachfolgers. Diese entspricht dem ersten Eintrag in der Finger-Tabelle
  - ◆ Zusätzlich wird noch die ID und IP-Adresse des unmittelbaren *Vorgängers* verwaltet

2 Chord

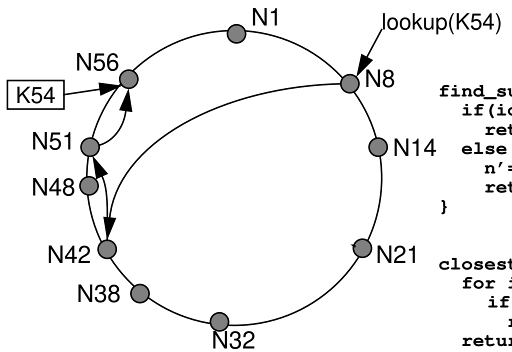
- Beispiel: Finger-Tabelle des Knoten N8 (m=6)Chord



ID	Verantworlicher Knoten
8+1	14
8+2	14
8+4	14
8+8	21
8+16	32
8+32	42

$(n+2^{k-1}) \bmod 2^m$, $1 \le k \le m$

- Suche nach einem Schlüssel mit Hilfe der Finger-Tabelle




```
find_succesor(id) {
  if(id ∈ (n,succesor])
    return successor
  else
    n'=closest_preceding_node(id)
    return n'.find_successor(id)
}

closest_preceding_node(id){
  for i = m downto 1
    if(finger[i] ∈ (n,id))
     return finger[i]
  return n
}
```

- Anfrage-Aufwand O(log n)

2 Chord

- Erzeugung eines neuen Chord-Rings:

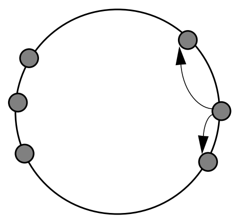
```
create(){
        predecessor= nil
        successor =n
}
```

- Beitritt eines neuen Knotens in den Chord-Ring:

```
join(n’){
        predecessor = nil
        sucessor = n’.find_successor(n)
}
```


2 Chord

- Um die Struktur des Chord-Rings auch bei einem oder mehreren neuen Knoten aufrecht halten zu können, ruft jedes Peer periodisch die Funktion **stabilize()** auf



```
stabilize(){
        x = successor.predecessor
        if( x ∈ (n,successor))
                successor = x
        successor.noftify(n)
}
notify(n’){
        if (predessor is nil or n’∈ (predecessor,n))
                predecessor = n’
}
check_predecessor(){
        if(predecessor has failed)
                predecessor = nil
}
```

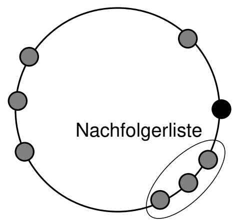

2 Chord

- Damit auch weiterhin ein schnelles Auffinden von Daten möglich ist, aktualisiert jeder Knoten periodisch seine Finger-Tabelle

```
fix_finger(){
        next = next + 1
        if (next > m){
                next = ⌊log(succesor − n)⌋ + 1
        }
        finger[next] = find_sucessor(n + 2^(next−1))
}
```


2 Chord

- Erweiterung der Routing-Informationen um eine Nachfolgerliste
  - Es kann nun der gleichzeitige Ausfall von *r-1* aufeinanderfolgender Knoten toleriert werden, wobei *r* der Länge der Nachfolgerliste entspricht.
  - Anpassung der **stabilize**-Funktion, damit die Nachfolgerliste mit der Liste des eigenen Nachfolgers abgeglichen wird. Ist dieser nicht erreichbar, wird der nächste Nachfolger in der Liste benachrichtigt.



2 Chord

- Erweiterung der Routing-Informationen um eine Nachfolgerliste (Fortsetzung)
  - Die Funktion `closest_preceding_node` wird so modifiziert, dass nicht nur in der Finger-Tabelle der Vorgänger gesucht wird, sondern zusätzlich in der Nachfolgerliste.
  - Sollte während des Aufrufs der Funktion `find_successor` ein Knoten ausfallen, kann dies durch den Ablauf eines Timers erkannt werden. In diesem Fall wird einfach der nächste, geeignete Knoten in der Nachfolgerliste und der Finger-Tabelle gesucht und aufgerufen.

2 Chord

- Berücksichtigung der realen Netzwerkstruktur
- Einfaches Verfahren:
  - Auswahl des nächsten befragten Knotens nicht nur auf Grund seiner Nähe zum gesuchten Schlüssel, sondern zusätzlich auf Grund der **Verbindungsqualität**
- Komplexeres Verfahren mit Erweiterung der Routing-Information:
  - Zu jedem Knoten der Finger-Tabelle wird zusätzlich eine Nachfolgerliste verwaltet
  - Nachrichten werden an Knoten mit der besten Verbindung aus der Liste weitergeleitet

3 Eigenschaften von verteilten Hash-Tabellen

- Schlüssel werden gleichmäßig auf alle Knoten verteilt
- Skalierbar mit Anzahl der Knoten (vgl. Literatur)
- Selbstorganisierend und damit keine manuelle Konfiguration nötig
- Unterstützung vieler verschiedener Anwendungen
  - Schlüssel haben keine semantische Bedeutung
  - Verwaltete Inhalte sind anwendungsunabhängig

H.6 JXTA

1 Überblick

- Initiative von Sun Microsystems mit der Zielsetzung eine generische Infrastruktur für P2P-Anwendungen zu entwickeln
- Der Name JXTA ist abgeleitet von dem englischen Verb *juxtapose:* nebeneinander stellen
- Projektseite
  - http://www.jxta.org

2 Überblick

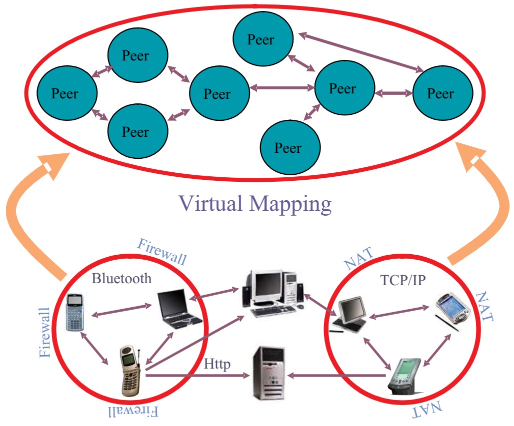
- Nachteile bisheriger Systeme
  - Meist proprietäre Protokolle
  - Protokolle in der Regel anwendungsspezifisch
  - Protokolle sind inkompatible zueinandner
  - Hoher Entwicklungsaufwand
- Ziele
  - Interoperabilität
    - Verbindung von File Sharing, Instant Messaging, ...
  - Plattformunabhängig von
    - Programmiersprache (Java, C, ...)
    - Betriebssystem (Unix, Windows, ...)
    - Netzwerk Protokoll (TCP/IP, Bluetooth, ...)
  - Universell verwendbar
    - Beliebige Geräte wie Desktops, PDAs, mobile Telefone, Sensoren, ...

3 Überblick

- JXTA
  - JXTA setzt sich aus einer Menge von Protokollen zusammen die eine Verbindung und Zusammenarbeit von Peers ermöglichen
  - JXTA bildet eine Middleware die eine einfache und schnelle Entwicklung von P2P-Anwendungen ermöglichen soll
  - JXTA wurde entwickelt um einem dezentralen P2P-Ansatz zu unterstützen. Jedes Peer kann sowohl Client als auch Server Funktionen übernehmen.
  - JXTA legt weder die Topologie des physikalischen, noch des virtuellen Netzwerkes fest

4 Überblick

- Realisierung verschiedener virtueller Netzwerktopologien auf Basis des gleichen physikalischen Netzwerkes



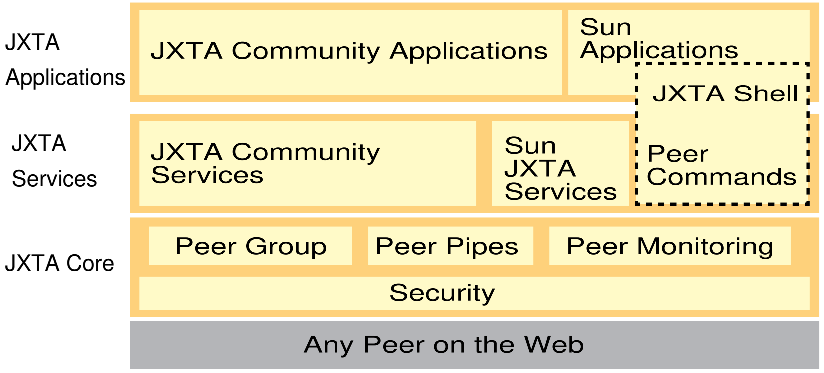
5 Literatur

- Project JXTA 2.0 Super-Peer Virtual Network (May 2003)
  - http://www.jxta.org/project/www/docs/JXTA2.0protocols1.pdf
- JXTA v2.0 Protocols Specification
  - http://spec.jxta.org/nonav/v1.0/docbook/JXTAProtocols.html
- Project JXTA: Java[tm] Programmers Guide
  - http://www.jxta.org/docs/JxtaProgGuide_v2.3.pdf
- Brendon J. Wilson: JXTA , New Riders 2002
  - http://www.brendonwilson.com/projects/jxta/pdf/JXTA.pdf

5 Architektur

- JXTA Core
  - ◆ Basis-Protokolle für den Betrieb von Peer-to-Peer Anwendungen
- JXTA Services
  - ◆ Dienste für Peer-to-Peer Applikationen
  - ◆ Beispiele: Suchdienst, Indizierung, File Sharing
- JXTA Applications
  - ◆ Anwendungen

5 Architektur

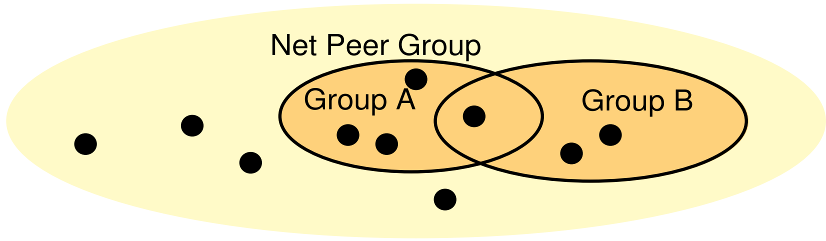


6 Grundkonzepte

- Peer - Ein oder mehrere Endgeräte welche die JXTA Core-Protokolle implementieren
  - ◆ Repräsentiert einen Teilnehmer des JXTA-Netzwerks
  - ◆ Ein Peer wird durch ein Advertisement beschrieben
  - ◆ Jedes Peer wird durch einen eindeutigen Identifikator referenziert
  - ◆ Jedes Peer gehört mindestens einer Peer Group an
- Spezielle Peer-Arten
  - ➤ Simple Peer - Minimales Peer (ohne Caching)
  - ➤ Edge Peer - Vorwiegend dienstnehmender Knoten
  - ➤ Rendezvous Peer - Bildet Treffpunkt für Peers
  - ➤ Router Peer / Relay Peer - Vermittler zwischen Knoten die auf Grund von Firewalls oder NAT nicht miteinander kommunizieren können

6 Grundkonzepte

- Peer Group - Eine Gruppe von Knoten mit gleichen Intressen
  - ◆ Peers einer Gruppe kooperieren in festgelegter Weise und besitzen ähnliche Dienste
  - ◆ Mitgliedschaft und Authentifizierung erfolgt durch die Peer Group, Schaffung eines sicheren Raums
  - ◆ Ein neues Mitglied einer Gruppe kann Anfragen nach Ressourcen, Diensten und Mitgliedern an die Gruppe stellen



6 Grundkonzepte

- JXTA ID
  - Identifiziert eine Entität eindeutig (Peer, Peer Group, Pipe, ...)
  - Referenzimplementierung stellt Universally Unique Identifier (UUID) bereit
  - Wird als Uniform Resource Name (URN) geschrieben
    - **urn:jxta:uuid-234324324592...AAA33FB5**
  - JXTA ID ist kanonisch und sollte als *opak* gesehen werden
  - Beispiel: Peer kann eindeutig identifiziert werden unabhängig von physikalischen Adressen, Netzwerkschnittstelle(n) oder Transportprotokolle
  - Trennung der Identifikation einer Ressource von ihrer Lokalisierung

6 Grundkonzepte

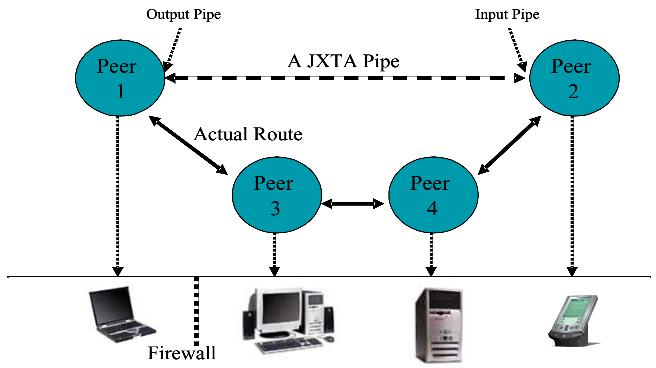
- Transportmechanismen
  - *Endpoint* - Start- und Endpunkt von Nachrichten aus Applikationssicht
    - Ergibt sich aus der Peer ID und allen Kontaktadressen
  - *Pipe* - Unidirektionale, asynchrone virtuelle Verbindung zwischen zwei oder mehr Endpoints
  - *Message* - Container für Daten die via Pipes zwischen Endpoints ausgetauscht werden
    - Message entspricht einem Datagramm
    - Jede Message verfügt über einen Umschlag (Sender und Empfänger Endpoint), einer variablen Anzahl von Protokoll spezifischen Headern sowie den Daten
    - Daten werden als eine geordnete Menge von (Name, Typ, Wert)-Tupel übertragen

6 Grundkonzepte

- Pipes
  - Virtuelle Verbindungen die zu verschiedenen Zeitpunkten mit unterschiedlichen Peer-Endpoints verbunden sein können
  - Jede Pipe kann durch eine JXTA ID eindeutig indentifiziert werden
  - Pipes sind unidirektional und besitzen mit *Output Pipe* eine Datenquelle und mit *Input Pipe* eine Datensenke
  - Es gibt zwei Arten von Pipes:
    - *Point-to-Point* Pipes für unidirektionale, asynchrone Verbindungen
      - Es gibt kein Acknowledgment oder Reply
      - Antworten werden über eine umgekehrt gerichtete Pipe versendet
    - *Propagate Pipes* für Multicast-Verbindungen

6 Grundkonzepte

- Pipes (Fortsetzung)



6 Grundkonzepte

- Advertisement
  - XML basierte Metadaten-Beschreibungen angebotener Netzwerk-Ressourcen, eines Peers oder einer Peer Group
  - Folgende Arten von Advertisements sind standardisiert
    - Peer, Peer Group,Pipe, Service, Rendezvous, Peer Endpoint
  - Advertisements können weitere Advertisements beinhalten
  - Advertisements können beliebig durch weitere Metadaten ergänzt werden
  - Rendezvous-Peers unterstützen die Suche und Verbreitung von Advertisements
  - Advertisments gelten nur zeitlich begrenzt !

6 Grundkonzepte

- Advertisement einer Peer Group

```
<?xml version="1.0" ?>
<!DOCTYPE jxta:PGA>

<jxta:PGA xmlns:jxta="http://jxta.org">

    <GID> urn:jxta:jxta-NetGroup</GID>

    <!-- Module Specification Id -->
    <MSID>urn:jxta:uuid-DEADBEEF2332342342...</MSID>

    <Name>NetPeerGroup</Name>

    <Desc>NetPeerGroup by default</Desc>

</jxta:PGA>
```


7 Protokolle

- Eine Suite von 6 Protokollen, die es ermöglichen sollen ad hoc ein Peer-to-Peer Netzwerk zu etablieren
- Die Suite kann in zwei Gruppen unterteilt werden
  - *Core Specification Protocols* - Minimales Protokollset für JXTA-Peers
  - *Standard Service Protocols* - Erweitertes Protokollset für bessere Interoperablität und mehr Funktionalitäten

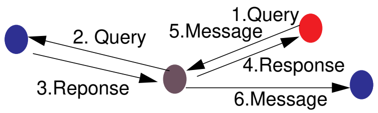
Standard Service Protocols	Peer Discovery Protocol	Pipe Binding Protocol	Peer Information Protocol	Rendezvous Protocol
Core Specification Protocols	Endpoint Routing Protocol	Peer Resolver Protocol		

8 Endpoint Routing Protocol (ERP)

- Ermöglicht es einem Peer dynamisch die Route zu einem anderen Peer über mehrere Hops hinweg zu ermitteln
- Verändert sich die Topologie des Netzwerks und eine Route bricht zusammen ermöglicht das Protokoll den Aufbau einer neuen Route
- Verwendungszwecke
  - Integration von Rechnern die via NAT adressiert werden oder durch eine Firewall geschützt sind
  - Umsetzung zwischen verschiedenen Transportprotokollen
  - Routing innerhalb von Ad Hoc Netzwerken (z. B. Bluetooth)

8 Endpoint Routing Protocol (ERP)

- Allgemeiner Ablauf
  - ◆ Soll eine Nachricht versendet wird zuerst ermittelt ob im lokalen Cache ein passendes Route-Advertisement voliegt
  - ◆ Ist dies nicht der Fall wird eine *Route Query Message* an Router/Relay Peers versendet
  - ◆ Verfügt ein Router-Peer über die gesuchte Route sendet er diese als eine Auflistung von *Hops* in Form einer *Route Response Message* an das anfragende Peer
  - ◆ Die Nachricht wir durch die Routinginformation ergänzt und kann nun entlang der Route an das Zielpeer versendet werden
  - ◆ Die Routinginformation wird entlang der Route aktualisiert und von den beteiligten Peers zwischengespeichert



9 Peer Resolver Protocol (PRP)

- Erlaubt es generische *Dienstanfragen* zu stellen
- Jede Anfrage besitzt einen spezifischen Namen der bei der Verarbeitung einer Anfrage dazu dient einen entsprechenden Handler zu identizifieren
  - ◆ Handler-Namen setzten sich in der Regel aus **Dienst-Name, Group-Id** und einer zusätzlichen Komponente zusammen
- Jede Dienstanfrage kann potenziell an alle Mitglieder einer Peer Group weitergeleitet werden

9 Peer Resolver Protocol (PRP)

- Allgemeiner Ablauf
  - ◆ Ein Peer versendet eine Anfrage als *Resolver Query Message* mit dem Rendevous Protocol an alle verbundenen Peers und Rendezvous Peers
  - ◆ Jedes Peer das die Nachricht empfängt überprüft ob es einen ensprechenden Handler betreibt und stellt die Nachricht zu.
  - ◆ Der Handler überprüft die Anfrage und sendet eine *Resolver Response Message* wenn er sie beantworten kann.
  - ◆ Unterbindet der Handler nicht explizit die weitere Zustellung (und handelt es sich um ein Rendezvous Peer) wird die Nachricht weitergeleitet.

10 Peer Rendezvous Protocol (RVP)

- Organisiert die Verteilung von Nachrichten in einer Peer Group
- Folgende Teilaufgaben werden durch das Protokoll adressiert
  - ◆ Regelt wie sich Peers bei Rendezvous Peers registrieren
  - ◆ Vernetzung der Rendezvous Peers
  - ◆ Verteilung von Nachrichten zwischen den Rendezvous Peers
- Wird von dem Peer Resolver Protocol und Pipe Binding Protocol genutzt um Nachrichten zu versenden, setzt auf dem Endpoint Routing Protocol auf

10 Peer Rendezvous Protocol (RVP)

- Verbindung von Peers und Rendezvous Peers
  - Es werden Leases genutzt um zeitbeschränkt die Nutzung eines Rendezvous Peers als Vermittler zu ermöglichen.
  - Ein Lease kann von beiden Seiten zu bliebigen Zeitpunkten aufgehoben werden.
- Allgemeiner Ablauf
  - Peer sendet eine *Lease Request Message* an einen Rendezvous Peer
  - Sollte der Rendezvous Peer noch freie Kapazitäten besitzen antwortet dieser mit einer *Lease Granted Message*
  - Wird der Lease nicht mehr benötigt so sendet der Peer eine *Lease Cancel Message*

10 Peer Rendezvous Protocol (RVP)

- Vernetzung der Rendezvous Peers
  - Zielsetzung ist das jedes Rendezvous Peer eine möglichst aktuelle und konsistente Sicht auf die Gruppe der Rendezvous Peers (*Peer View*) besitzt
- Allgemeiner Ablauf
  - Jedes Rendezvous Peer versendet *PeerView Messages* als *Probe* Nachricht an andere Rendezvous Peers. Eine *Probe PeerView Message* enthält ein *Rendezvous Advertisment* des anfragenden Peers
  - Die Probe Anfrage wird durch ein *Response PeerView Message* beantwortet. Diese kann weitere *Rendezvous Advertisments* beinhalten.

10 Peer Rendezvous Protocol (RVP)

- Kontrolle der Ausbreitung von Nachrichten innerhalb der *Peer View*
- Alle Nachrichten die innerhalb der *Peer View* vermittelt werden, sind in eine *RendezVousPropagat Message* eingebettet
- Die Parameter der *RendezVousPropagat Message* ermöglichen
  - Vermeidung von Schleifen und Dublikation
  - Kontrolle der Ausbreitung (TTL)

11 Peer Discovery Protocol (PDP)

- Veröffentlichen von Ressourcen eines Peers via Advertisement
- Suche nach Ressourcen (Peers, Peer Groups, Pipes ...)
- Allgemeiner Ablauf
  - Zur Suche nach einer Ressource wird eine *DiscoveryQuery Message* versendet
    - Angabe der Art der gesuchten Ressource
    - Maximale Anzahl der Antworten
    - Attribute zur Spezifikation der Suche
  - Sollte die Anfrage beantwortet werden können wird eine *DiscoveryResponse Message* versendet
    - Anzahl der Treffer und Antworten
- *DiscoveryResponse Message* dienen auch zum Veröffentlichen von Ressourcen

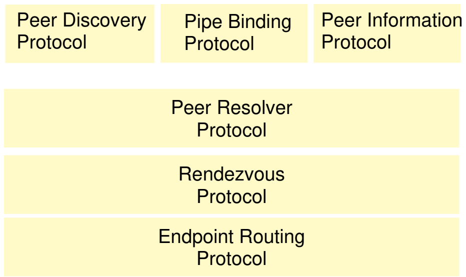
12 Peer Information Protocol (PIP)

- Dient zu Ermittlung von Statusinformationen
- Ablauf
  - Anfrage nach Statusinformationen durch eine *PeerInfoQuery Message* mit optionaler, zusätzlicher Anfrage
  - Anwort erfolgt durch eine *PeerInfoResponse Message*
    - Uptime, Verkehr, Last usw. sowie zusätzliche Informationen
- PIP verwendet PRP für die Vermittlung von Anfragen

13 Pipe Binding Protocol (PIB)

- Ermöglicht es virtuelle Verbindungen aufzubauen
- Auf der Empfangsseite können ein oder mehrere Peers verbunden werden
- Allgemeiner Ablauf
  - Es wird ein Pipe Advertisement veröffentlicht
  - Ein Peer bindet eine Input Pipe zur ID des Advertisements
  - Ein anderes Peer sucht eine entsprechende Pipe mittels einer *PipeResolver Message* mit dem Message Typ *Query*
  - Ein Peer welches die Input Pipe gebunden hat versendet eine *PipeResolver Message* mit dem Message Typ *Answer*
  - Anfragende Peer kann nun eine Output Pipe binden

14 Protokollstack

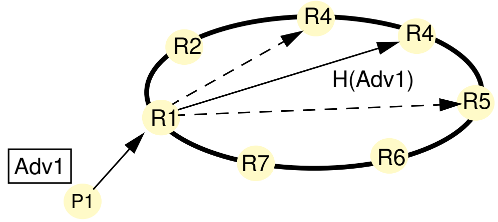


15 Bewertung des Protokollstack

- Geringer Overhead
- Geringe Anforderungen an das Transportsystem
  - unidirektionale, asynchrone Verbindungen
- Geringe Anforderungen an die Ausführungsumgebung des Peers
  - Je nach Anwendung und Peer muss nur ein Untermenge der Protokolle implementiert werden
- Geeignet für eine Vielzahl von P2P-Anwendungen
  - Dynamische Netzwerkstruktur und ständiger Wechsel von Peers

16 Rendezvous Netzwerk der Referenzimplementierung

- Veröffentlichung eines Advertisement

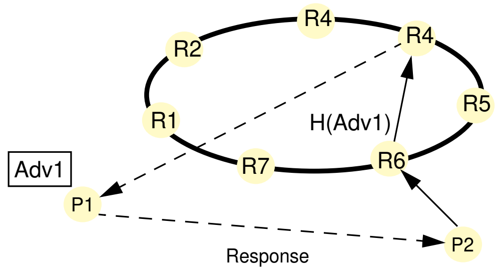


◆ Rendezvous Peers verwalten nur noch einen Advertisement-Index

◆ Für jedes Advertisement wir ein Hashwert bestimmt der festlegt welche Rendezvous Peers Informationen über das Advertisement speichern

16 Rendezvous Netzwerk der Referenzimplementierung

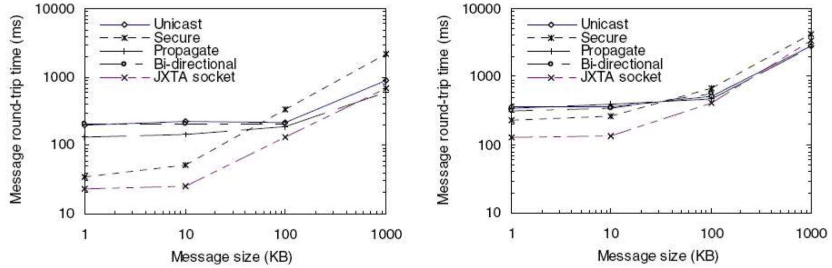
- Suche nach einem Advertisement



◆ Das zuständige Rendezvous Peer R4 leitet die Anfrage an P1 weiter.

17 Evaluierung

- Round-trip Messungen im LAN/WAN
- JXTA 2.2



18 Zusammenfassung

- JXTA versucht eine Standardumgebung für Peer-to-Peer Anwendungen zu etablieren
  ◆ Ist dies schon jetzt sinnvoll?
- JXTA wir in der Regel nicht immer die beste und effizienteste Lösung bieten können um eine Peer-to-Peer Anwendung zu entwickel