ACCELERATING xP2P

Internet Week 2001

P2P技術の事例と今後の展開
技術事例編

梅田英和（スカイリー・ネットワークス、jnutella.org）
hidekazu.umeda@skyley.com

1. Gnutellaプロトコル
   - Gnutellaとはどんなソフトか？
   - 接続からファイルのダウンロードまで
   - 5種類のメッセージ（descriptor)
   - GTKT (Gnutella Toolkit)について
2. Jnutella P2P プロトコル
   - JPPPのコンセプト
   - アーキテクチャ概要
   - 実装とJPPP SDK

ACCELERATING xP2P

Gnutellaプロトコル

アジェンダ

1. Gnutellaとはどんなソフトか？
2. 接続からファイルのダウンロードまで
3. 5種類のメッセージ（descriptor)
4. GTKT (Gnutella Toolkit)について

ACCELERATING xP2P

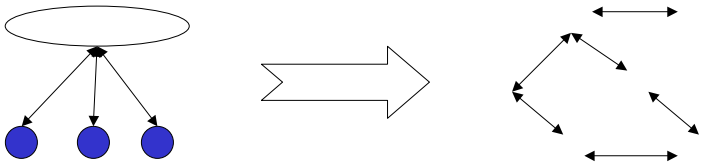
Gnutellaとはどんなソフトか．．？

1. サーバとクライアントを兼ね備えるソフトウェア

＝サーバント

2. サーバント同士が接続しあって中心のないネットワークを構成

3. 分散検索とファイル共有



ACCELERATING xP2P

接続からファイルのダウンロードまで

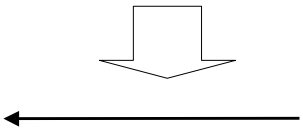
1. TCP/IPコネクションの確立
2. ネゴシエーション
3. Ping, Queryの送信（検索）
4. QueryHitの受信（検索結果）
5. ファイルのダウンロード（HTTP）

ACCELERATING xP2P

接続からファイルのダウンロードまで

2. ネゴシエーション

"GNUTELLA CONNECT/0.4"

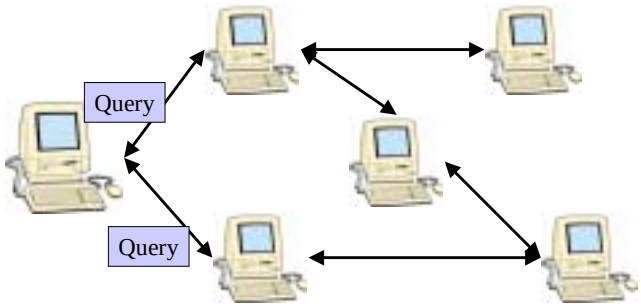


"GNUTELLA OK"

ACCELERATING xP2P

接続からファイルのダウンロードまで

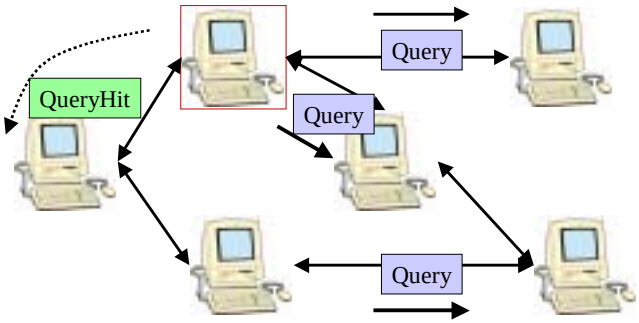
3. 検索の様子



ACCELERATING xP2P

接続からファイルのダウンロードまで

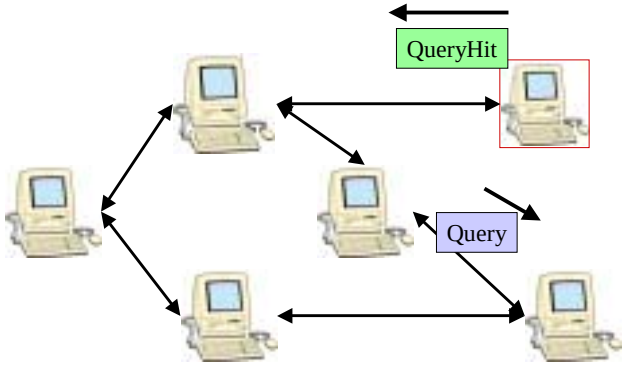
3. 検索の様子



ACCELERATING xP2P

接続からファイルのダウンロードまで

3. 検索の様子

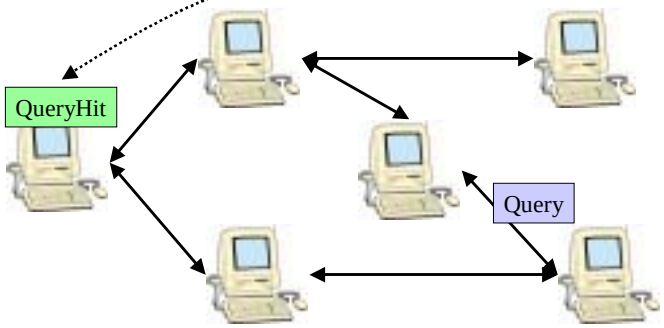


ACCELERATING xP2P

接続からファイルのダウンロードまで

4. 検索結果の受信

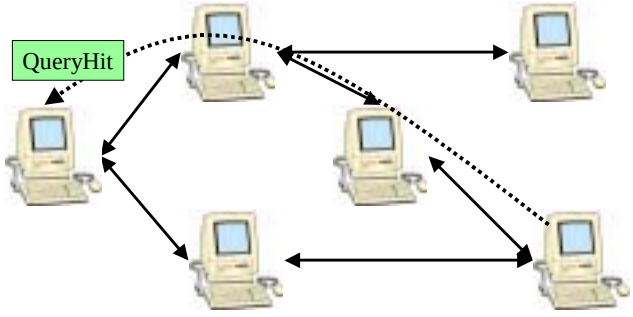
サーバントが検索結果をリレーして、
検索元まで送り返す



ACCELERATING xP2P

接続からファイルのダウンロードまで

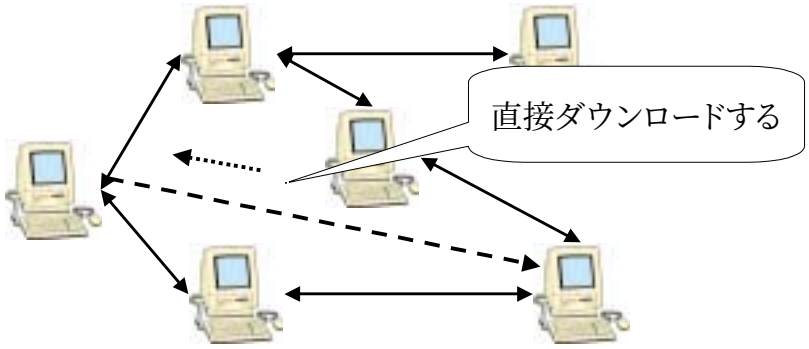
4. 検索結果の受信



ACCELERATING xP2P

接続からファイルのダウンロードまで

5. ファイルのダウンロード



ACCELERATING xP2P

接続からファイルのダウンロードまで

5. ファイルのダウンロード

HTTP GETメソッドで相手から直接ダウンロード
ファイルのリレー転送はしない

GET /get/<File Index>/<File Name>/ HTTP/1.0¥r¥n
Connection: Keep-Alive¥r¥n
Range: bytes=0-¥r¥n
User-Agent: Gnutella¥r¥n¥r¥n
¥r¥n

ACCELERATING xP2P

5種類のDescriptor

1. ネットワーク状態を調べる、検索をかける、検索結果を返すなどの機能を持つメッセージを、Descriptorと呼んでいる

2.Gnutella では、必要な時に適切なDescriptorを送受信することで検索とファイル交換機能を実現する。

3. Ping, Pong, Query, QueryHit, Push　の5つ

ACCELERATING xP2P

5種類のDescriptor

Type	Function	Routing
Ping	Pongの応答を促す	broadcast
Pong	公開しているファイル数、総バイト数 待ち受けポート番号などを返す	sendback
Query	ファイル検索をかける	broadcast
QueryHit	検索にヒットするファイルを持っている サーバントが検索結果を返す	sendback
Push	Firewallの内側にいる相手からファイルを 受け取る	sendback

ACCELERATING xP2P

5種類のDescriptor

Descriptor構造

・ヘッダ部とペイロード部に別れている

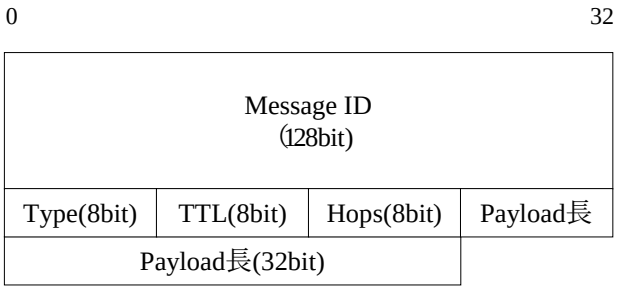
ヘッダ (23バイト)
ペイロード (タイプによってサイズが違う)

ACCELERATING xP2P

5種類のDescriptor

ヘッダ構造

・すべてのDescriptorに共通
・各Descriptor毎に固有のIDを持ち、受信データの重複やループ判定に使う。

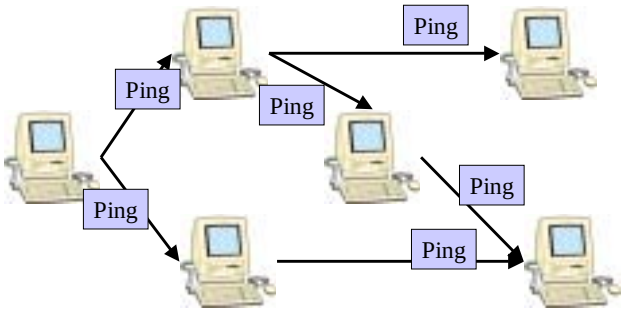


ACCELERATING xP2P

5種類のDescriptor

1. Ping

・TypeID = 0x0
・Pingはペイロードを持たない
・サーバントからPong応答をもらうために送信
・broadcastタイプ

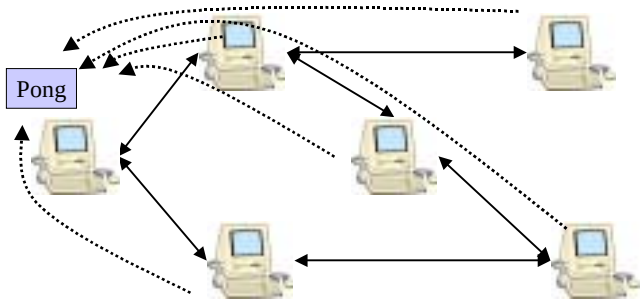


ACCELERATING xP2P

5種類のDescriptor

2. Pong

・TypeID = 0x1
・14バイト
・Pingへの応答として、公開しているファイル数、自分のIPアドレス、ポート番号などを返答
・send backタイプ

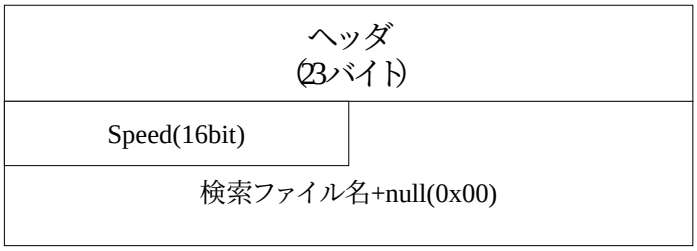


ACCELERATING xP2P

5種類のDescriptor

3. Query

・TypeID = 0x80
・ファイル検索を行なうためのメッセージ
・null終端させたファイル名で検索
・不定長
・broadcastタイプ



ACCELERATING xP2P

5種類のDescriptor

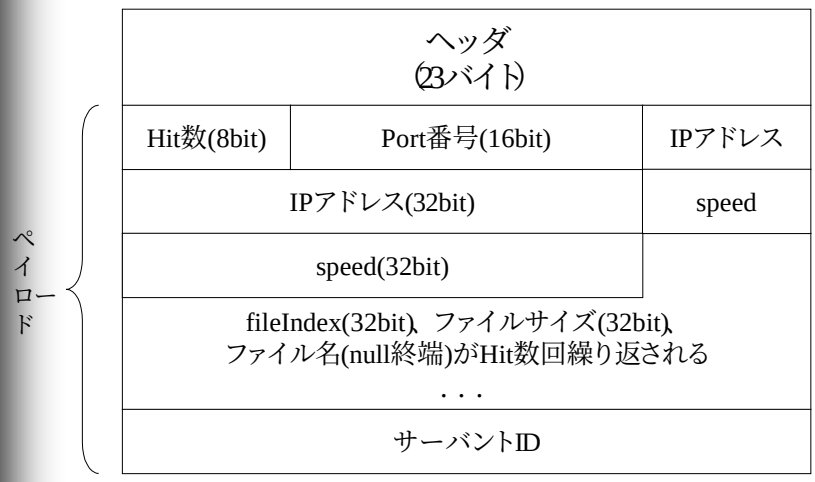
4. QueryHit

・TypeID = 0x81
・検索にヒットするファイルを持っている場合に返答
・ファイル名、通信速度、IPアドレスなど
・不定長、検索キーワードはnull終端
・send backタイプ

ACCELERATING xP2P

5種類のDescriptor

4. QueryHitの構造

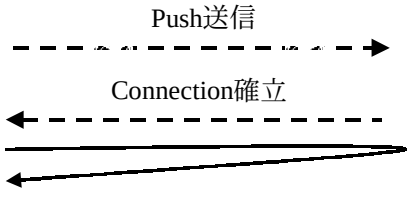


ACCELERATING xP2P

5種類のDescriptor

5. Push

・TypeID = 0x40
・26byte
・ファイルをダウンロードする際、相手側からファイルを送りこんでもらう（Firewall対策）
・send backタイプ

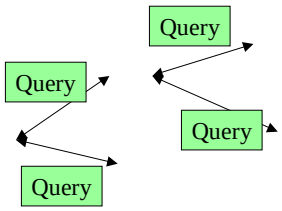


ACCELERATING xP2P

ルーティング規則

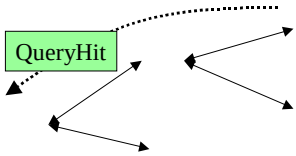
・Gnutellaのルーティング規則は基本的に２種類
・broadcastとsend back

broadcast



隣接する全てのサーバントにデータを転送

send back



メッセージが辿った経路を正確に逆行

ACCELERATING xP2P

ルーティング規則

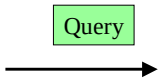
	Descriptor
Broadcast	Ping, Query
Send back	Pong, QueryHit, Push

PushはMessage ID ではなくServant IDを元に送り返されるので、やや特殊

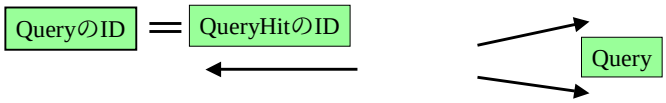
ACCELERATING xP2P

ルーティング規則

Broadcastタイプのメッセージが送られてきたら



DescriptorのMessage IDをコピーして、応答する



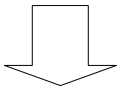
Message IDが同じということは、その2つのDescriptorが応答関係にあるということ。

ACCELERATING xP2P

GTKt（Gnutella Toolkit）

・Jnutella.orgで開発・保守しているGnutellaプロトコル・ライブラリ
・プロトコルとアプリケーション・GUIを分離

これまでは各サーバント作者が個別にプロトコルを実装

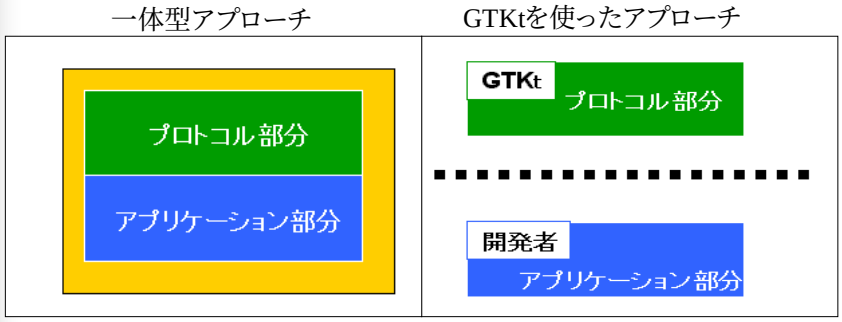


・サーバント間での非互換性
・プロトコルのバージョンアップに個別に対応しなくてはならない

ACCELERATING xP2P

GTKt（Gnutella Toolkit）

・プロトコル部分を集中的にメンテナンスすることでライブラリの信頼性が向上
・Gnutellaプロトコルを様々な用途に適用可能
・最新動向をフォロー
・モバイルへの移植も考慮



ACCELERATING xP2P

GTKt（Gnutella Toolkit）

ステータス

・ver0.14
・BearShare拡張QueryHit対応
・PongFollowup対応
・50KB
・各種サーバントとの互換テスト (Reflector, BearShare, ToadNode, LimeWire, Furi)
・Descriptorのルーティングをカスタマイズ可能

・Push、ファイルアップロード未実装

ACCELERATING xP2P

Gnutella最新動向

現行の問題点

・スケーラビリティ
pingに応答するだけでも10000超
ping, pongが帯域をひっ迫しつつある

・ダウンロードの成功率
NAT、Firewall環境にいるサーバントが多くincoming接続やpushが働かない

・多国語対応
検索キーワードの文字コードはASCIIしか考慮されていない。

ACCELERATING xP2P

Gnutella最新動向

・BearShare拡張QueryHit
QueryHitに特殊フィールドを追加（BearShare独自仕様）
互換性を犠牲にしてユーザビリティを優先
Busyフラグで相手の負荷状態を把握可能

・Ping, Pongキャッシュ、Multiplex送信
Gnutellaネットワークの60%がping-pong応答に費やされている現実から、ping, pongの応答・転送アルゴリズムの改良を提案

ACCELERATING xP2P

参考URL

・Jnutella.org
http://www.jnutella.org/

・GTKt
http://www.jnutella.org/gtk/gtkt013.zip

・GDF
http://groups.yahoo.com/group/the_gdf

ACCELERATING xP2P

Jnutella P2P プロトコル

アジェンダ

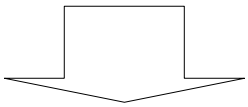
1. JPPPのコンセプト
2. アーキテクチャ概要
   - •コアとプロファイル
   - •マルチホップと通信モデル
   - •ルーティング
3. 実装とJPPP SDK

ACCELERATING xP2P

JPPPコンセプト

一言でいうと．

携帯電話でGnutellaを実現するにはどうしたら良いか、何が足りないか、というテーマの追求から始まったMog Projectの成果



モバイル・ワイヤレス端末のみで自律的に動作する、実用レベルのPure P2Pシステム

ACCELERATING xP2P

JPPPコンセプト

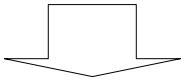
2つのポリシー

Fully Decentralized

Napsterと異なり、Central point（従来の言葉でいう「サーバ」）が存在しない

P2Pアプローチ

アプリ、サービス開発、ユーザにとってのメリット、ビジネスモデル、思想を含めた全体的な運動。



・コア部分をオープンソース化
・SDKとして開発環境と開発モデルを提供
・様々な商用バージョンが可能なように

ACCELERATING xP2P

アーキテクチャ概要

コアとプロファイル—問題意識として．．

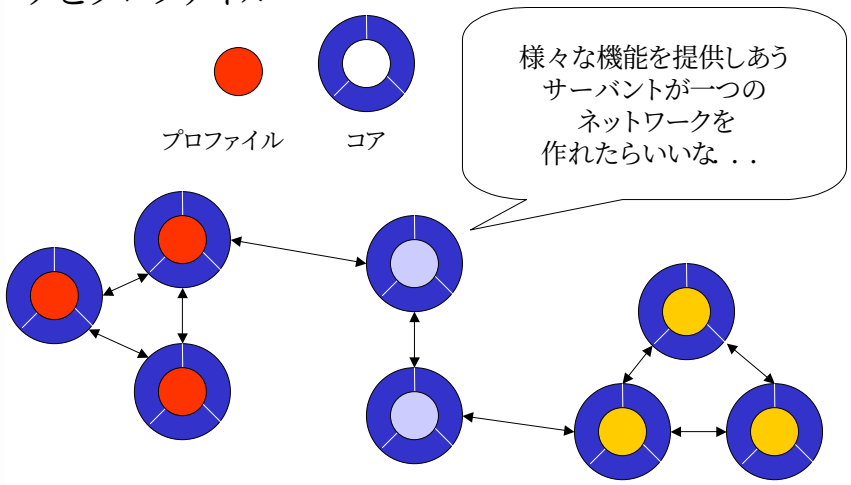
・企業、組織が独自にP2Pプロトコルを作った結果、相互接続性が失われている。
・ユーザは複数のソフト、サービスを使い分けなければならない。

不特定多数のPeerが自分の「パワー」の源になるという、P2Pのダイナミズムが失われてしまっている

ACCELERATING xP2P

アーキテクチャ概要

コアとプロファイル



ACCELERATING xP2P

アーキテクチャ概要

コアとプロファイル

コア　すべてのPeerが共有すべき機能と構造

- 基本データフォーマット（パケット構造）
- ネットワークへの参加、離脱、ネゴシエーション
- デフォルトルーティングとルーティングプラグイン

プロファイル　JPPPを用いて提供されるサービス

- セレクタの定義とその解釈
- プロファイル独自の拡張ルーティング

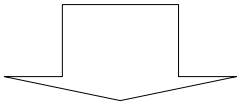
ACCELERATING xP2P

アーキテクチャ概要

エントリーポイント発見サービス

JPPPノードがP2Pネットワークの一部となるために最初に接続する相手先を、エントリーポイントと呼ぶ

GnutellaではIPアドレスを直接入力したり、サーバントアドレスのリストを別に用意していた

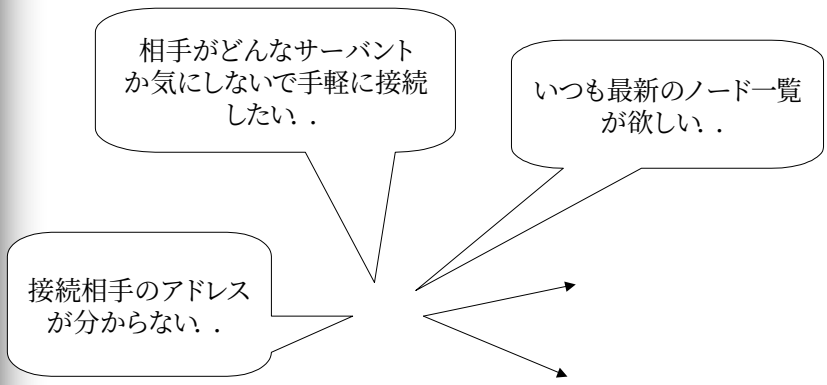


標準的な取得手段を提供

ACCELERATING xP2P

アーキテクチャ概要

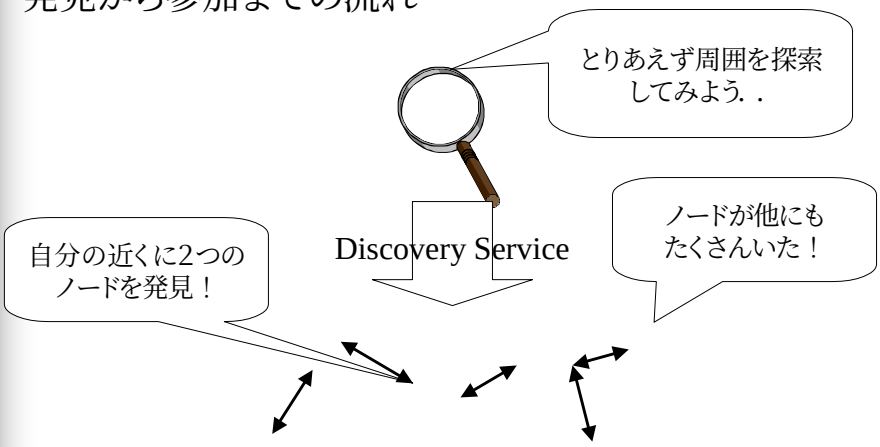
エントリーポイントを探す際に．



ACCELERATING xP2P

アーキテクチャ概要

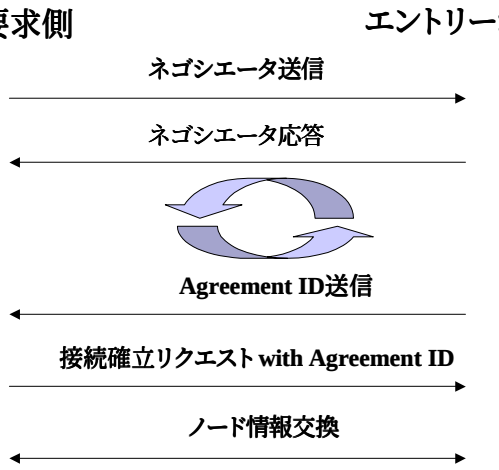
発見から参加までの流れ



ACCELERATING xP2P

アーキテクチャ概要

接続ネゴシエーション

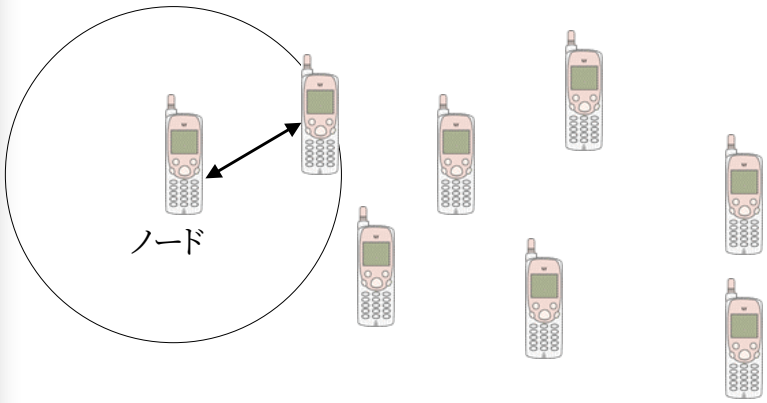


ACCELERATING xP2P

アーキテクチャ概要

マルチホップ

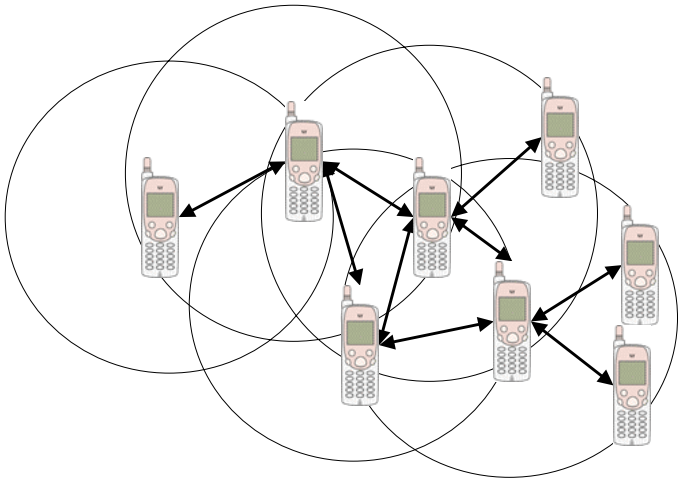
それぞれのノードは有限の通信半径をもつ



ACCELERATING xP2P

アーキテクチャ概要

有限の通信範囲がネットワークトポロジを規定する

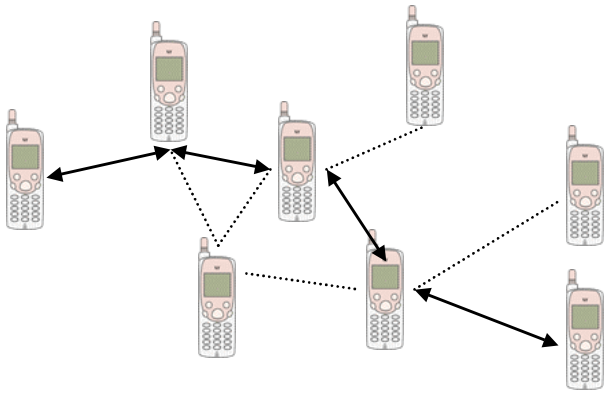


ACCELERATING xP2P

アーキテクチャ概要

マルチホップ

通信半径外の相手との通信は、他人に中継してもらう

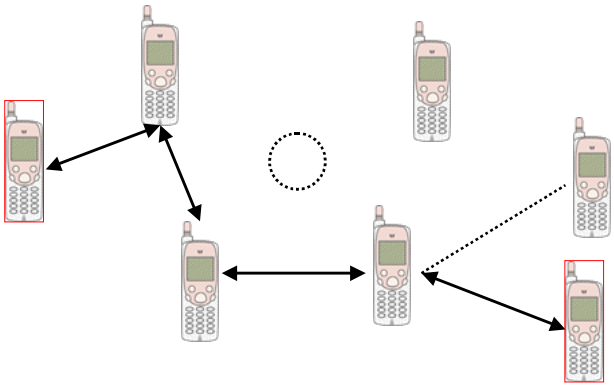


ACCELERATING xP2P

アーキテクチャ概要

Multihop unicast

途中経路が変化しても1対1の通信が継続する

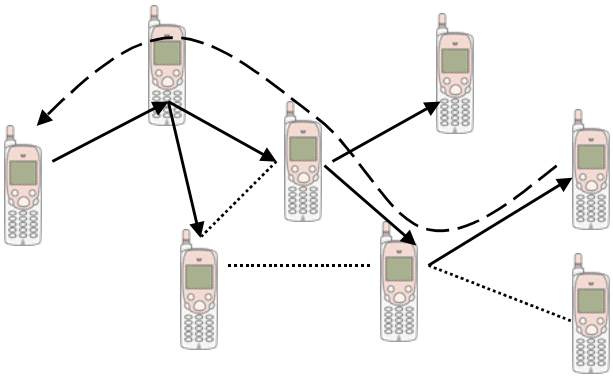


ACCELERATING xP2P

アーキテクチャ概要

Sendback unicast

メッセージが辿ってきた経路を逆向きに送り返す



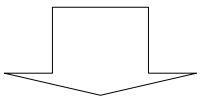
ACCELERATING xP2P

アーキテクチャ概要

ルーティング・プロトコル

Pure P2PネットワークでMultihop unicastを実現するために経路制御が必要

一方で、ノードの流動性が高く、経路が不安定



MANET（Mobile Ad hoc NETwork）で提案されているいくつかのルーティングプロトコルを試験的に採用

ACCELERATING xP2P

アーキテクチャ概要

ルーティング・プロトコル

Fisheye State Routing Protocol

・定期的に経路表をリフレッシュ（Proactive型）
・スコープの小さいノードほど頻繁にリンクステートを交換する

The Dynamic Source Routing Protocol

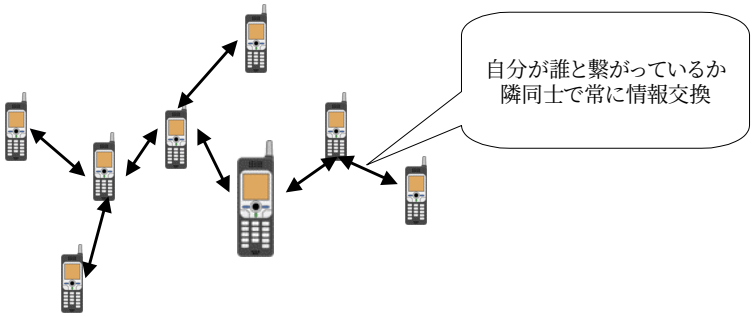
・データを送信する際に経路を発見（on-demand型）
・Route Discoveryをbroadcastし、Route Replyで経路情報を取得、Source Routingを用いて通信

ACCELERATING xP2P

アーキテクチャ概要

JPPP Routing Protocol

基本は、隣同士でリンク状態を交換するProactive型



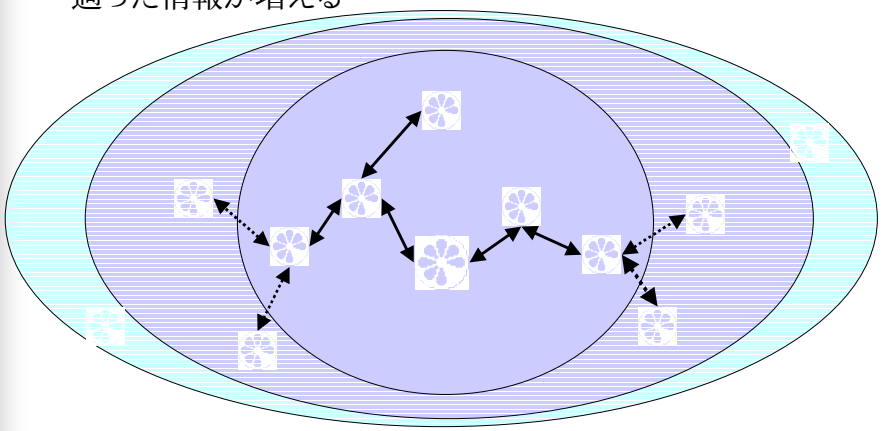
スコープによって更新サイクルを変化させることで
通信量を減らす

ACCELERATING xP2P

アーキテクチャ概要

JPPP Routing Protocol

自分に近いほど良く見え、周辺にいくほど確率的に
過った情報が増える



ACCELERATING xP2P

アーキテクチャ概要

JPPP Routing Protocol

リンクステート情報

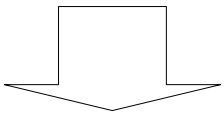
対象ノードID
シーケンス番号
隣接ノード数
隣接ノードID ….

ACCELERATING xP2P

アーキテクチャ概要

JPPP Routing Protocol

「よく見える範囲」の外にいる相手とはどう通信するのか？



JPPPパケット構造を利用した「戻り専門」のソースルーティング

ACCELERATING xP2P

アーキテクチャ概要

JPPPパケット構造

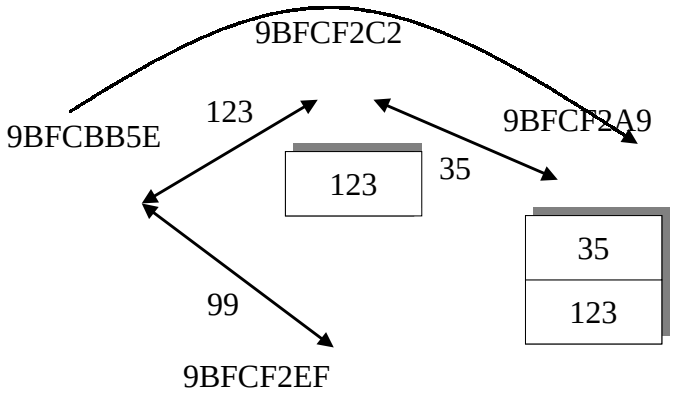
TTL	HOPS
ペイロード長	スタック長
セレクタ名	
送信先ID	
送信元ID	
メッセージID	
ペイロード...	
経路スタック...	

ACCELERATING xP2P

アーキテクチャ概要

経路スタック

メッセージがリレーされると、リンクIDがスタックに積まれる

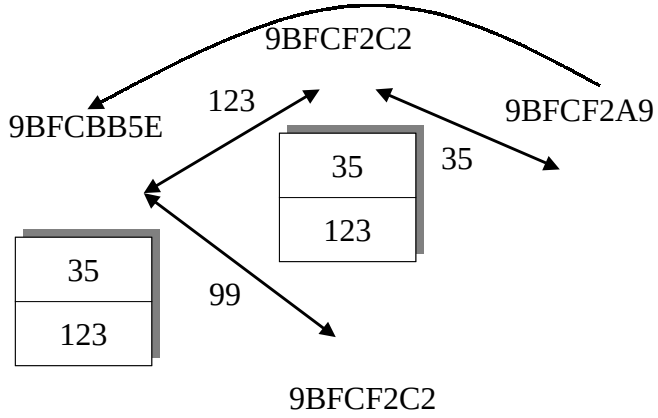


ACCELERATING xP2P

アーキテクチャ概要

経路スタック

メッセージが戻ってきた時には、スタックに経路情報が詰め込まれている。

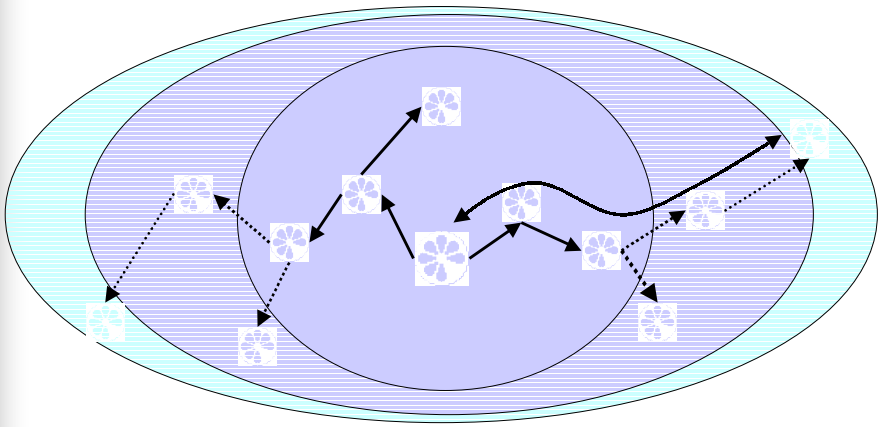


ACCELERATING xP2P

アーキテクチャ概要

ブロードキャストと経路スタック

スコープ外の相手へブロードキャストでメッセージを送信。戻ってきたメッセージの経路スタックを使って通信

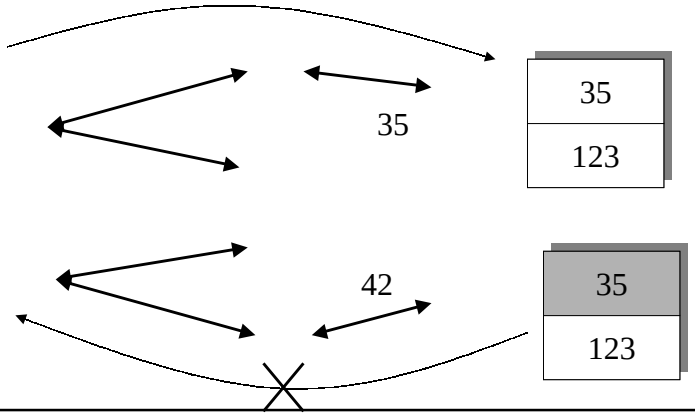


ACCELERATING xP2P

アーキテクチャ概要

スタックの再構築アルゴリズム

途中経路の変化に応じて、経路スタックの内容を適切に作り直す

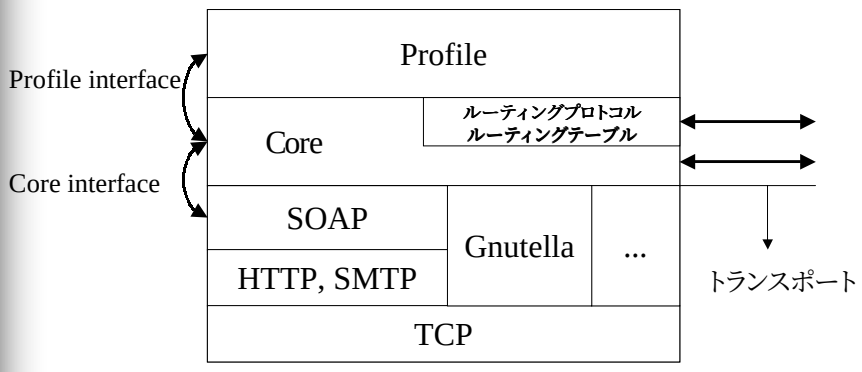


ACCELERATING xP2P

実装とSDK

プロトコルスタック

コアの直下にあってネットワークインターフェイスとの仲介を果たす部分をトランスポートと呼ぶ



ACCELERATING xP2P

実装とSDK

トランスポート

Ver0.2ではSOAP/HTTPを利用。

・Selializerがあることで、ストリーム内でのデータ表現をとりあえず考えなくて良い。

・通信手順をメソッドの設計という抽象的なレベルで思考できる。

・HTTPは Firewall を通らせるためだけに利用される。

・トランスポートにBluetoothを使った実装が進行中

ACCELERATING xP2P

実装とSDK

ランタイム構成

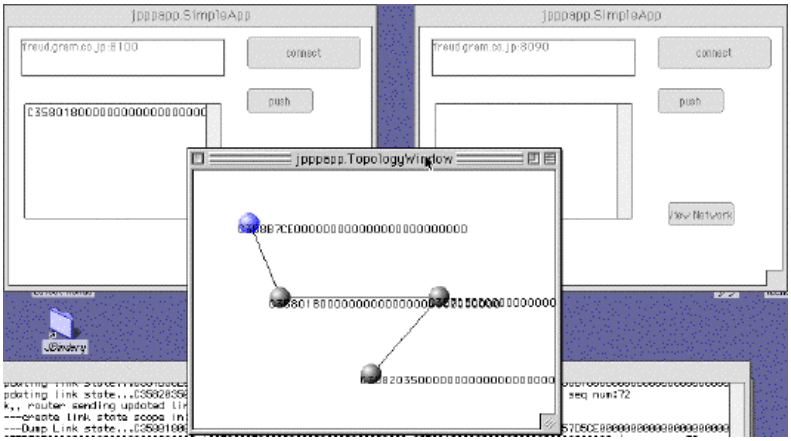
Java言語で構築

<table>
<tr><td>様々なプロファイル</td><td>…</td><td>…</td></tr>
<tr><td colspan="3">プロファイル　インターフェイス</td></tr>
<tr><td colspan="3">コア 実装</td></tr>
<tr><td colspan="3">コアインターフェイス</td></tr>
<tr><td rowspan="2">Apache-SOAP 2.0
JPPPBeanSerializer</td><td rowspan="2">Jasper3.0改
RPCRouter Servlet</td><td rowspan="2">GTKt
(GnutellaToolkit)</td></tr>
<tr></tr>
</table>

実装とSDK

スクリーンショット

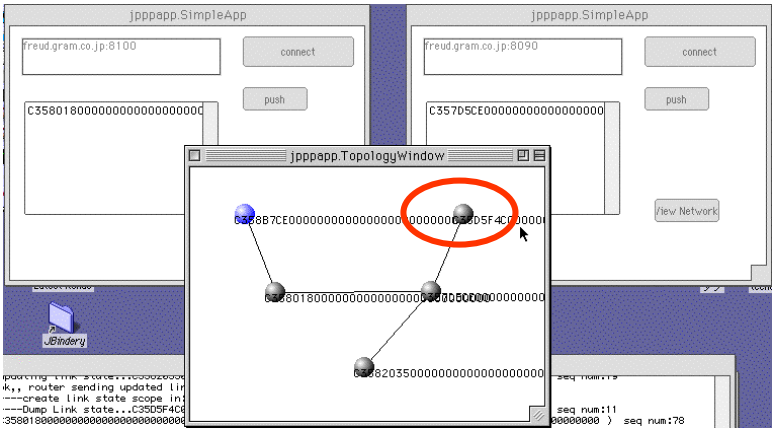
4つのノードで構成されているネットワークのルーティングテーブルをグラフ表示した図



実装とSDK

スクリーンショット

新しいノードが1つ参加

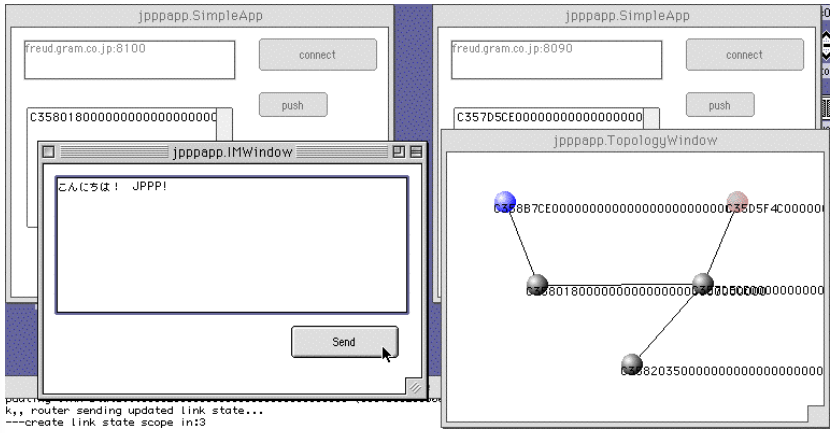


ACCELERATING xP2P

実装とSDK

スクリーンショット

新しいノード (赤) に対してメッセージを送信

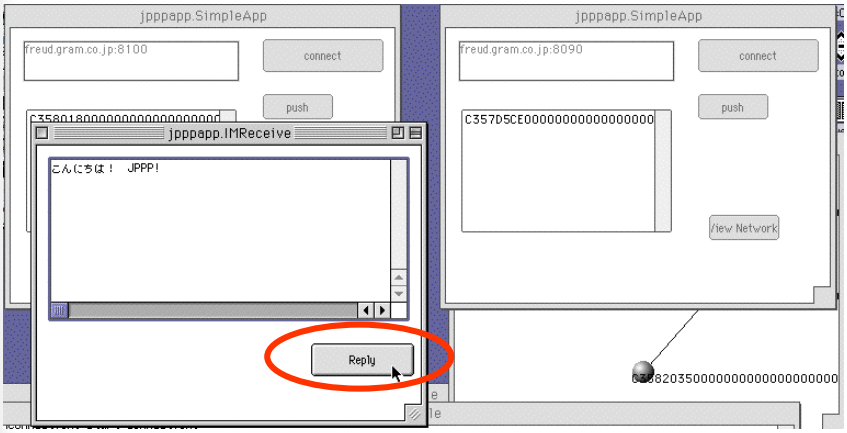


ACCELERATING xP2P

実装とSDK

スクリーンショット

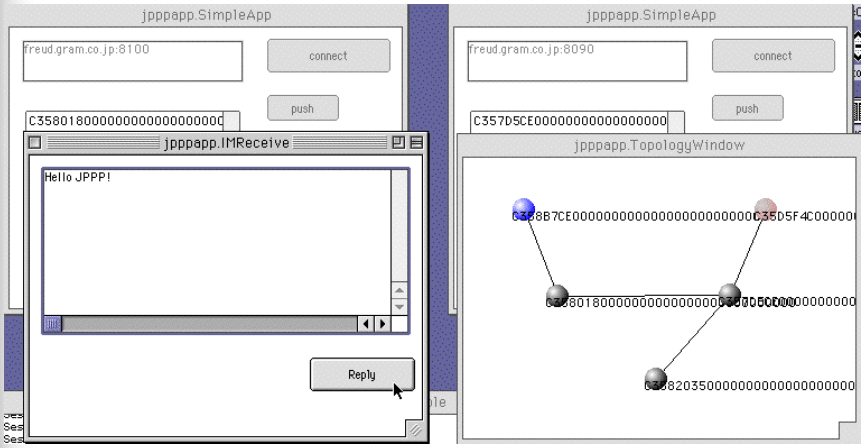
メッセージがルーティングされて到達。Broadcastではないことに注意。



実装とSDK

スクリーンショット

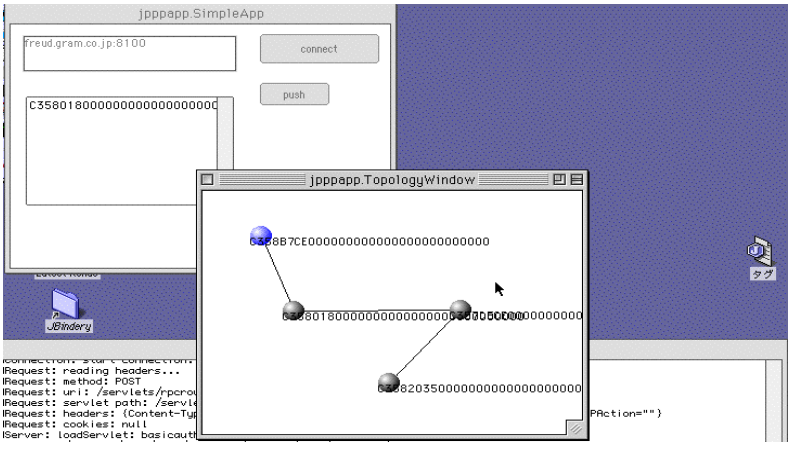
返事をsendback unicast で送信。Send back unicastの場合は Stack Routingのため、送信速度が速い。



実装とSDK

スクリーンショット

ノードを強制終了させると、変化した経路情報がきちんと伝播する。



ACCELERATING xP2P

実装とSDK

開発ロードマップ

・JPPP仕様Ver1.0へ向けて...

・C言語版
・Bluetooth Transport
・WLAN Transport

Milestone

・SDKとRuntime
・JPPP仕様Ver0.5

5月末　　9月末

ACCELERATING xP2P

これからの課題

短期的

•JPPPパケットのフィールドはこれで充分か？
•パケットの経路スタックはノードID（＝グローバルアドレス）？
それともリンクローカルな値？（トレードオフ）
•ルーティングの停止、再開、レート制限プロトコル

ACCELERATING xP2P

これからの課題

長期的

・セキュリティ
データをリレー転送することが本質的なPure P2Pでいかにしてセキュアな通信を行なうか

・認証
Central PointのないPure P2Pでどのように認証を確立するのか？　->分散認証サービスの必要性

・性能評価
数千規模のサーバントで構成されるネットワーク

ACCELERATING xP2P

参考文献

・Jnutella.org
http://www.jnutella.org/

・Jnudev ML
http://www.jnutella.org/mailinglist/

・JPPP 開発ガイドVer.0.2
http://www.jnutella.org/presentation/umeda/jppp/spec/

・MANET
http://www.ietf.org/html.charters/manet-charter.html

・Mobile IP
http://www.ietf.org/html.charters/mobileip-charter.html

・NTT未来ねっと研究所　PREFERENCE , SION
http://www.onlab.ntt.co.jp/jp/ni/preference/index.html

・Cybiko
http://www.cybiko.com/