P2P プロトコルの緩衝について

2003MT042 近藤 真弘
指導教員　張 漢明

1　はじめに

現在，いくつかの組織によって P2P ソフトウェア開発環境が提供されている． 開発環境として JPPP[2] や JXTA[3]，GTKt[1] がある．各 P2P ソフトウェア開発環境で定義されるプロトコルは，各組織が独自に定義したものであり，標準化がなされていない．仕様の異なるプロトコルが存在する事によって，同一の目的を持つアプリケーションであっても，異なるプロトコルを使用する事例が存在する．

異なる開発環境で作成されたアプリケーション同士は，各開発環境が定義するプロトコルの差異が障害となり，互いに通信をおこなう事ができない．アプリケーションプロトコルが同一であっても，P2P ソフトウェア開発環境が定義するプロトコルが異なることで，相互運用が図られない．この問題は本質的に回避不可能であるが，本研究では，新たにアプリケーションを作成する場合に状況を限定する．新たにアプリケーションを作成する場合に状況を限定することで，アプリケーション開発者は他プロトコル，ライブラリを知ること無くアプリケーションを開発できる可能性がある．

本研究の目的は，アプリケーション開発者が異なる開発環境を用いていても，相互運用可能な P2P アプリケーションを開発することのできる環境を構築することである． 本研究の基本的なアイデアは， 開発環境が提供するプロトコル間の差分を埋めることである． 本研究ではプロトコル緩衝手法として，フレームワークの構築を提案する． 下位プロトコルを定義する開発環境に，上位プロトコルを解決するフレームワークを構築することで プロトコル緩衝をおこなう． 本研究は以下の手順で進める．

- P2P プロトコルの比較
- プロトコル緩衝手法の提案
- JPPP と GTKt を例に，プロトコルレベルの異なる開発環境間のプロトコル緩衝の実現
- GTKt と JXTA を例に，同一レベルのプロトコルを定義する開発環境間への適用可能性を考察

本研究では，上位プロトコルを解決するフレームワークを構築する手法でのプロトコル緩衝を提案した．本研究で提案する手法を用いることで，ソフトウェア開発者は元々使用していた開発環境の形式を用いて，他プロトコルを実装することができる．一方フレームワークを使用して作成されるアプリケーション自体に対しては，本研究は関与しない．アプリケーションが持つネットワークトポロジやメッセージのリレー方式は，アプリケーション開発者が決めなければならない．

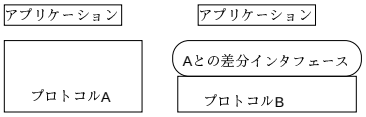

図 1: 基本的なアイデア

2　プロトコルの比較

本研究では JPPP，GTKt，JXTA を対象に研究を進める． JPPP，GTKt は共に Jnutella によって開発され，オープンソースである． JPPP は，携帯電話等構成要素が頻繁に変わるネットワークでの運用を 目指したプロトコルである． GTKt は，ファイル共有プロトコルである Gnutella プロトコルをフレームワーク化したものである．JXTA は，Sun が提供する P2P ソフトウェア開発フレームワークである． **図 2** に各環境が定義するプロトコルをまとめた．JPPP ではメッセージの基本形式は定義しているが，メッセージ内容までは定義していない． アプリケーション開発者がメッセージ内容と，メッセージの持つ意味を定義することを想定している．Gnutella，JXTA ではメッセージ形式に加えてメッセージ内容を定義し， ピアは定義されたメッセージの通信によってネットワークを構成する． JPPP は基礎レベルまでのプロトコルを定義する下位プロトコルであり，Gnutella，JXTA は応用レベルのプロトコルを定義する上位プロトコルである．

プロトコル	開発環境	プロトコルが定義する範囲
JPPP	JPPP	パケット構造
Gnutella	GTKt	パケット構造 メッセージ内容
JXTA	JXTA	パケット構造 メッセージ内容

図 2: プロトコルの比較

3　プロトコル緩衝手法の提案

上位プロトコルで定義される情報を下位プロトコルが受信した時，下位プロトコルでは受信した情報を適切に解決することができない． 上位プロトコル解決のためには，上位プロトコルを解決する手段が必要である．下位プロトコルまでを定義する開発環境を，上位プロトコルを解決できるように矯正する必要がある． 本研究では，下位プロトコルを定義する開発環境に，上位プロトコルを解決する フレームワークを作成することでプロトコル緩衝をおこなう．アプリケーション開発者は単一の開発環境で開発をおこない，単一のプロトコルを利用するつもりで複数のプロトコルを実装することができる．

4 JPPP と Gnutella による実現

JPPP と Gnutella を例に実現をおこなった．JPPP は下位プロトコルを定義し，Gnutella は，上位プロトコルまでを定義する．JPPP に Gnutella を解決するフレームワークを構築する．**図 3** にアーキテクチャを示した．

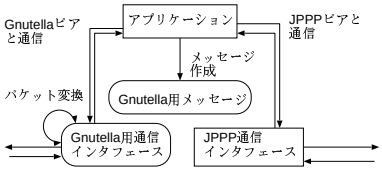

図 3: JPPP と GTKt による実現

4.1 メッセージに関するプロトコル緩衝

メッセージについてプロトコル緩衝をおこなう．Gnutella ではメッセージ内容が定義されているので，JPPP のメッセージ形式を用い，差分情報を追加可能な API を用意する．差分情報を追加可能な API を用意することで，アプリケーション開発者は JPPP のメッセージに対し，Gnutella との通信に十分な情報を追加することができる．JPPP のメッセージ形式を用いることで，JPPP 同士の通信には JPPP が用意する通信インタフェースを利用可能である．

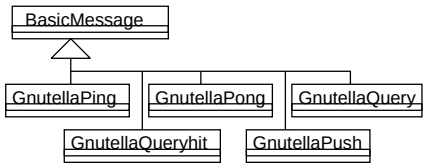

図 4: メッセージについてのフレームワーク

4.2 通信インタフェースに関するプロトコル緩衝

通信インタフェースについてプロトコル緩衝をおこなう．JPPP と Gnutella との通信には，下位プロトコルである JPPP に，対 Gnutella 用通信インタフェースを用意する．**図 5** は，作成した通信インタフェースを用いてメッセージを送信する際のシーケンス図である．JPPP に用意した Gnutella 用メッセージ作成 API を用いることで，Gnutella メッセージを作成する十分な情報を渡すことができる．通信インタフェースに通信相手の情報とメッセージを渡すと，JPPP メッセージから Gnutella メッセージを作成する．作成されたメッセージは，指定された相手に送信される．メッセージ受信時には，Gnutella メッセージを JPPP 形式に変換し，アプリケーションへ渡す．

5 考察

本研究では定義されるプロトコルレベルが異なる開発環境を対象に 研究を進めた．定義されるプロトコルレベルが同じである開発環境を対象に，本研究の適用可能性を考察する．

Gnutella と同様にメッセージ内容が定義されているプロトコルとして，JXTA がある．JXTA は Gnutella と同様にメッセージ内容が定義されているが，メッセージ形式が違いメッセージを互いに解決することができない．JXTA が Gnutella のメッセージを，Gnutella が JXTA のメッセージを解決することができるなら，JXTA と Gnutella は互いに通信をおこなえる．JXTA に Gnutella 解決用のフレームワーク，GTKt に JXTA 解決用のフレームワークを作成することで，相互運用が図られると考える．**図 6** に JXTA と GTKt へ本研究を適用した際のアーキテクチャを示した．JXTA と GTKt は共に，JXTA メッセージ，Gnutella メッセージ両方を持つ．

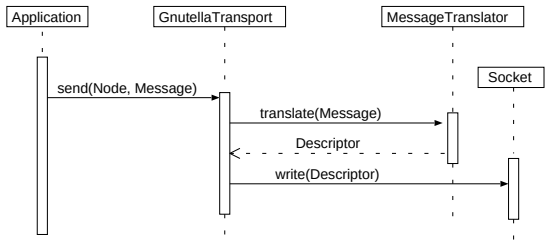

図 5: メッセージ送信時のシーケンス図

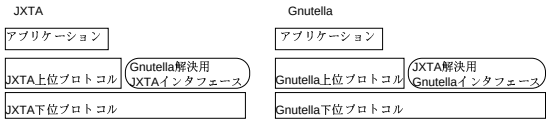

図 6: JXTA と GTKt への本研究の適用

6 おわりに

本研究では，下位プロトコルを定義する開発環境に，上位プロトコルを解決するフレームワークを構築する手法でプロトコル緩衝をおこなった．本研究で提案する手法では，アプリケーション開発者に対して 他のプロトコルとの通信を可能にする環境を提供できる．アプリケーション開発者は単一の開発環境で開発をおこない，単一のプロトコルを利用するつもりで複数のプロトコルを実装することができる．複数のプロトコルを実装できることで，異なる開発環境で作成されたアプリケーションの相互運用が図られる．

謝辞

本研究を進めるにあたり，2 年間熱心にご指導いただいた張漢明助教授，野呂昌満教授，有益なアドバイスを下さった蜂巣吉成先生，野呂研究室大学院生のみなさま，野呂研究室，張研究室のみなさまに深く感謝いたします．

参考文献

[1] GTKt. `http://www.jnutella.org/docs/other/what_is_gtkt11_14.shtml`. Jan, 2007

[2] JPPP. `http://www.jnutella.org/presentation/umeda/jppp/spec/`. Jan, 2007

[3] JXTA. `http://www.jxta.org/`. Jan, 2007