3H4-NFC-03b-1

RSNP クライアントの実装を簡略化するための追加ライブラリ

An additional library for implement cost reduction of RSNP clients

加納 政芳 *1　山本 悠策 *1　中村 剛士 *2
Masayoshi Kanoh　Yusaku Yamamoto　Tsuyoshi Nakamura

*1中京大学　*2名古屋工業大学
Chukyo University　Nagoya Institute of Technology

The Robot Service Network Protocol (RSNP) is a common platform for providing services with robots. Developing a service in reference to RSNP costs its developers programs for server and robot control. In this paper, we propose an additional library for implement cost reduction of RSNP clients.

1. はじめに

近年，情報端末やネットワーク技術の発展，普及に伴い，様々な情報へのアクセスが容易になった．ロボット分野においても，コンテンツの提供やセンサ情報の集約といったネットワークを利用したサービスが提案されており [1, 2, 3]，今後，サービスが拡大していくと考えられる．ロボットを用いた様々なサービスを提供するための共通プラットフォームとして Robot Service Network Protocol（RSNP）[4, 5] がある．RSNP に準拠して開発されたサービスは，RSNP が利用可能な全てのロボットで使用できる利点がある．

RSNP に準拠したサービスの開発には，それに準拠したサーバの構築，ロボットの開発および，サーバとロボット用のプログラム開発など，要求される技術が多岐にわたる．このようにサービス開発には大きなコストがかかることから，サービス開発は敬遠されがちである．RSNP に準拠したサービス開発の環境を改善する試みとして，文献 [6] では，RSNP Server Container を提案している．サーバ仮想化技術である Linux Containers を利用して RSNP サーバ環境をゲスト OS として提供することによって，RSNP に準拠したサーバを容易に構築できるようにしている．文献 [7] では，リアルタイム音声通信のために MultimediaProfile を拡張し，リアルタイムで音声データを通信可能にする機能を追加している．このようにサービス開発環境改善の試みは数多くなされているが，これまでに RSNP クライアントのプログラム開発を改善する提案は行われていない．そこで本稿では，RSNP クライアントのためのサービス実装を簡略する追加ライブラリを提案する．本ライブラリは，サービスの開発に必要となる基本的な処理をパッケージとして提供するため，RSNP サービスの開発コストの削減が見込める．

2. RSNP クライアント開発用ライブラリ

従来の RSNP ライブラリ構造では，複数のクラスにロボットの動作指示，サーバへの依頼送信やサーバからの依頼要求対応といった処理を記述している．そのため，開発者にとって，ソースコードの加筆修正箇所やデータ通信の流れがイメージしづらいといえる．これに対して提案ライブラリでは，サービスの開発に必要な接続・認証・切断といった基本的な処理をパッケージとして提供しつつ，簡易な実装を実現する．

連絡先: 加納 政芳，中京大学，〒 466-8666 名古屋市昭和区八事本町 101-2

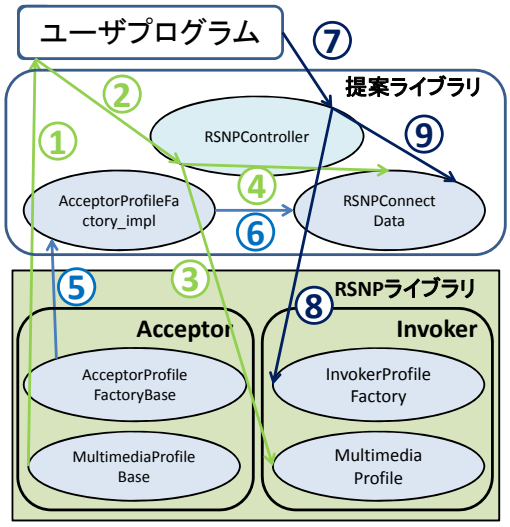

図 1: 提案ライブラリ

簡易な実装を実現するために，RSNPController クラス，RSNPConnectData クラス，AcceptorProfileFactory_impl クラスを開発した（図 1）．RSNPController クラスは，接続・切断や認証といった通信の管理と各プロファイルの利用処理を行う．RSNP クライアントの開発者は基本的にこのクラスのみを用いてプログラムを開発すればよい．RSNPConnectData クラスは，通信に必要なデータおよび，サーバからの要求に対する各プロファイルの応答処理を登録する．AcceptorProfileFactory_impl クラスは，サーバからの要求に対する各プロファイルの応答処理を管理する．

図 1 の矢印は処理の流れを表しており，①〜④はサーバからの要求に対する処理，⑤，⑥はサーバからの要求に対する各プロファイルの応答処理の管理，⑦〜⑨は通信の接続・切断や認証処理である．従来の RSNP ライブラリを用いたクライアントの開発では，①②③⑤⑦⑧をコーディングする必要があるが，提案ライブラリを用いた場合は，①，⑦のみコーディングすればよい．なお，図 1 は MultimediaProfile を例にしているが，他のプロファイルにおいても提案ライブラリは利用可能である．

```
 1: public static void main(String[] args) throws IOException {
 2:     if (args.length >= 3) {
 3:         robot_id = args[0];
 4:         password = args[1];
 5:         port = Integer.parseInt(args[2]);
 6:     }
 7:     ServerSocket serversock = new ServerSocket(port);
 8:     Socket sock = null;
 9:     RSNPController rsnp = new RSNPController(epn, robot_id, password);
10:     rsnp.connect();
11:
12:     try{
13:         if (rsnp._isConnect()) {
14:             long conv_id = rsnp.getConv_id();
15:             IContents_profile cp = rsnp.startContents_profile(new ContentsProfileBase());
16:             IAsyncCallBack callback = new IAsyncCallBack() {
17:                 @Override public void doEvent(Ret_value ret, boolean isLast) {
18:                     ContentsProfileHelper helper = new ContentsProfileHelper(ret);
19:                     String message = helper.getDetail();
20:                 }
21:                 @Override public void doException(Exception e) {}
22:             };
23:             cp.distribute_contents(conv_id, "", -1, 0, callback);
24:
25:             Multimedia_profile_impl mpi = new Multimedia_profile_impl();
26:             IMultimedia_profile mp = rsnp.startMultimedia_profile(mpi);
27:             mp.start_profile(conv_id);
28:             sock = serversock.accept();
29:
30:             rsnp.disconnect();
31:             sock.close();
32:             serversock.close();
33:         } else { }
34:     }catch(RSiException e){
35:         e.printStackTrace();
36:     }
37: }
```

図 2: RobotMain.java の開発例

3. 開発例

本節では，http://www.robotservices.org/wiki/jp において公開されているサンプルプログラム RobotSample.zip を例にして，提案ライブラリを用いた場合のソースコードを示す．RobotSample.zip は画像送信サービスのサンプルプログラムであるが，このサービスを実現するために，AcceptorProfileFactory_impl.java，Multimedia_profile_impl および，RobotMain.java の 3 つのソースコードを作成している．提案ライブラリを用いた場合は，AcceptorProfileFactory_impl.java は作成せずに，Multimedia_profile_impl.java と同一のソースコードを用いて，RobotMain.java のみ書き換えることで同様の機能を有したサービスを開発できる．図 2 に提案ライブラリを用いた場合の RobotMain.java のソースコードを示す．

まず，RSNPController クラスをインスタンス化する（同図 l.9; 図 1 の⑦）．つぎに，RSNPController クラスのインスタンスを通じてサーバへの接続およびロボットの認証を行い，conv_id を同インスタンスから取得する（ll.10，14，15; ⑦）．そして，MultimediaProfile をインスタンス化し（l.25; ①），同プロファイルの開始処理を行う（l.26; ⑦）．最後に，RSNPController クラスのインスタンスを通じてサーバとの接続を切断する（l.30; ⑦）．

以上の手続きによって容易にサーバとの通信管理と MultimediaProfile の利用ができる．

4. おわりに

本稿では RSNP を用いたサービスの実装を簡略化するための追加ライブラリを提案した．本ライブラリを使用することでクライアント用のプログラムの開発が容易になり短時間でサービスの提供が可能になると考える．今後は，サーバ用の追加ライブラリの提案を行う予定である．

参考文献

[1] ジメネスフェリックス，加納政芳：RSNP を用いた英単語学習支援ロボットの開発, 日本ロボット学会学術講演会 2012, 2B2-6, 2012.

[2] 牟田真介, 山本悠策，加納政芳，中村剛士：動画からの特徴的な表情抽出による高齢者の心の見守りシステム, 日本ロボット学会学術講演会 2012, 2B2-5, 2012.

[3] 石田真一, 荻谷浩史，松日楽信人：ロボットサーフィンサービスの提案, 日本ロボット学会学術講演会 2013, 3R3-05, 2013.

[4] 成田雅彦，村川賀彦，植木美和，中本啓之，日浦亮太，平野線治，蔵田英之，加藤由花：普及期のロボットサービス基盤を目指す RSNP（Robot Service Network Protocol）2.0 の開発，日本ロボット学会誌，vol.27，no.8，pp.857-867，2009.

[5] 成田雅彦，村川賀彦：ロボット技術の標準化と RSi（Robot Service Initiative）の取り組み，日本ロボット学会誌，vol.29，no.4，pp.353-356，2011.

[6] 三浦秋人, 小川康一, 加藤由花：RSNP Server Container の提案, 日本ロボット学会学術講演会 2013, 3R1-04, 2013.

[7] 大澤秀也，朝倉健介，小原範子，佐藤健，藤田尚宏，成田雅彦：RSNP を利用したリアルタイム音声の実現, 日本ロボット学会学術講演会 2013, 3R2-02, 2013.