Vol.2014-DPS-158 No.14
Vol.2014-CSEC-64 No.14
2014/3/6

ロボットサービス基盤を実現するRSCSの実装と評価

三浦 彰人[1,a)]　小川 康一[1,b)]　加藤 由花[1,c)]

概要：本研究は，インターネット上のサーバを用いて，ネットワーク経由でロボットサービスを提供するインフラの提案である．ロボットサービスプロトコルのひとつに RSNP（Robot Service Network Protocol）がある．従来，RSNP を用いるシステムの構成には，構築手順が複雑で時間がかかるという課題があった．そこで，ロボットサービス基盤を構築するためのインフラをオールインワンで提供する RSNP Server Container System を開発した．本稿ではその実装結果を報告する．

1. はじめに

近年，ロボット技術とクラウド技術の発達により，ロボットとクラウドとの連携によるサービスの提供手法について，様々なものが提案されている．ロボットとクラウド間の連携については，独自仕様が制定されている場合が多く，異なるサービスやロボット間の連携を困難にしている．そこで，ロボットサービスのインターネット化を目指し，RSi[1] により Robot Service Network Protocol（RSNP）[2] の仕様化が進められてきた．

最も単純な RSNP によるロボットサービス上の通信の例を，**図 1** に示す．この図のように，ユーザ端末とロボットとの間に RSNP サーバが仲介し，ロボットと RSNP サーバとの連携により，ユーザにロボットサービスを提供する．

RSNP 準拠のライブラリ実装としては，富士通研究所で開発された FJLIB（Java SDK として提供）があり，これを利用することで，ロボットアプリケーション，ロボットサービスの開発が可能である．FJLIB を利用したサンプルプログラムは，RSNP チュートリアルサイトからダウンロード可能であり，2013 年度の RSNP コンテスト [3] においても，FJLIB を利用した多くのロボット，ロボットサービスが出展されている．

一方，ロボットサービス開発者の裾野が広がるにつれ，特にサーバサイドの環境整備に手間取り，開発が思うように進まない事例が報告されるようになってきた．本稿では，サーバ仮想化技術を利用してこの問題を解決する．具体的には，OS レベルの仮想化技術である Linux Containers（LXC）[4] を利用し，オールインワンの RSNP サーバサイドの環境をゲスト OS ごと提供する，RSNPServer Container Service（RSCS）を提案する．

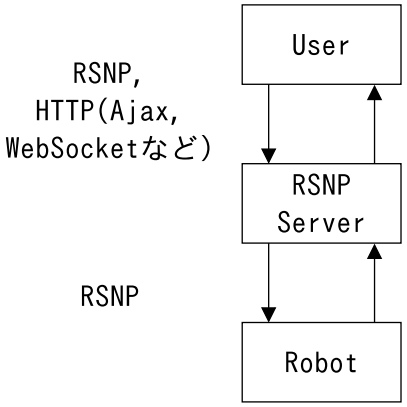

図 1　RSNP サービスにおける通信の例

コンテナ型仮想化技術を用いた既存研究としては，Linux-VServer を用いた学内ホスティングサービス [5] や，ロボット向けの Platform-as-a-Service フレームワークである Rapyuta[6] などがある．RSCS も，これらと同様の方向性を持つシステムである．

2. 課題

RSNP を用いたサービスを構築する場合，開発・運用環境に起因する以下の 3 つの課題が存在する．

- 導入に関する課題
  オペレーティングシステム（Windows，Linux など），HTTP サーバ（Apache HTTPD など），Web アプリケーションフレームワーク（Tomcat など）など，RSNP

[1] 産業技術大学院大学
Advanced Institute of Industrial Technology
[a)] a1235am@aiit.ac.jp
[b)] a1209ko@aiit.ac.jp
[c)] yuka@aiit.ac.jp

サーバの提供には多くの構成要素が存在し，導入と運用に手間がかかる．

- 開発に関する課題
  開発したサービスを運用環境に展開する際に，オペレーティングシステムやソフトウェアバージョン等の違いにより，予期せぬ問題が発生することがある．RSNP サーバとして用いられる環境として Linux が多いが，Windows 環境での Java アプリケーション開発に慣れている開発者には敷居が高い．
- 運用に関する課題
  システム更新の際に，構成要素間の依存関係の問題などにより不具合が発生することがある．

3. 設計

これら導入と開発，運用に関する課題を解決するために，RSC を設計した．RSC は，RSNP サーバサイドの環境設定が全て行われており，システムを立ち上げた直後から，RSNP サービスを実行可能なゲスト OS（コンテナ）である．コンテナとしてアーカイブの再配布が可能になるよう，Java VM，Web サーバを含め，全てのソフトウェアは再配布可能なライセンス形態のものを利用している（FJLIB を除く）．複数の RSC を一つのホスト上に構築することも可能であり，環境の異なる複数サーバ上での実験等が実施可能である．

本稿ではさらに，RSC，コンテナ内の RSNP サービスをインターネット上に公開するための基盤となる RSCS 管理コンテナ，各コンテナを管理する Web インタフェースとツール（管理ユーティリティ）を統合し，RSCS として提供する．

RSCS の構成の概要を**図 2** に示す．RSCS は，ホスト環境，RSCS 管理コンテナ，RSC の 3 要素に分けられる．ホスト環境はコンテナの管理を担当する．また，RSCS 管理コンテナでは，管理 Web インタフェースとリバースプロキシによる各 RSC へのアクセスの振り分けをおこなう．そして RSC はリバースプロキシ経由で RSNP サービスを提供する．RSCS を利用した際の通信の流れは，**図 3** のようになる．ユーザからの要求に対し RSNP サーバがロボットと連携しサービスを提供するという流れは基本的な RSNP 通信と同様だが，ユーザと RSNP サーバ間，RSNP サーバとロボット間にリバースプロキシサーバが挟まる形となる．

ホストの動作環境として，グローバル IPv4 アドレスをひとつ持つ Virtual Private Server（VPS）インスタンスなどを用いることを想定している．また，利用形態としては，その VPS 上の RSCS を一つの団体・プロジェクト内で共有し開発をおこない，サービスを提供することを想定している．

汎用的なクラウド基盤を用いた環境と比較した際の利点として，RSNP を用いたサービスの提供に特化し管理インタフェースを簡略化している点，また 1 インスタンスの VPS などのような小規模な環境でも利用できる点が挙げられる．

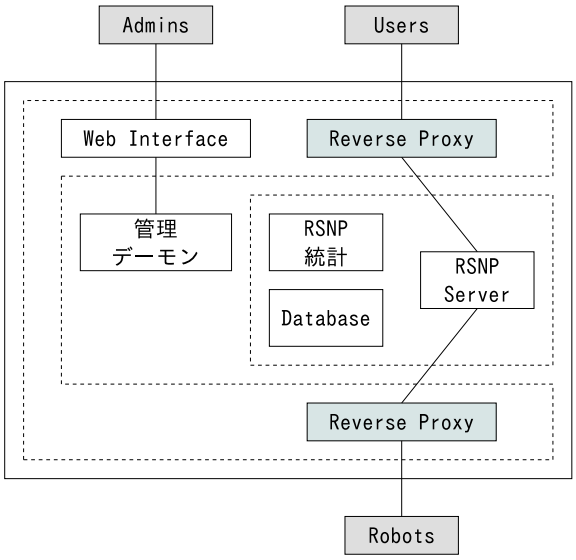

図 2 RSCS の構成

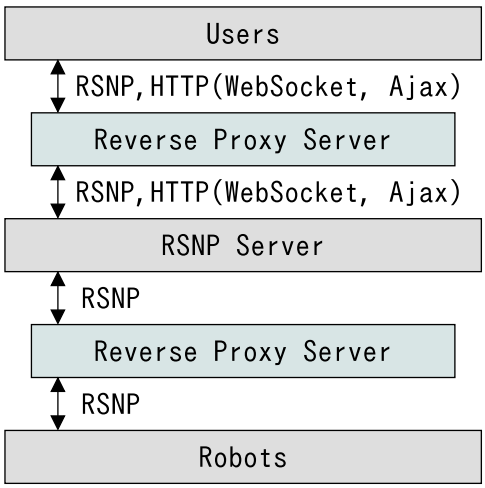

図 3 通信の流れ

4. 実装

4.1 ホスト環境

ホスト環境では，RSCS を維持するための最小限の機能を提供する．RSCS に必要となる機能は以下の通りである．

- ネットワーク
  - 外部公開用ネットワークインタフェース
  - コンテナ用ネットワークブリッジインタフェース
  - ファイアウォール
- アカウント
  - ホスト管理アカウント
  - SSH による管理アカウントのリモートログイン
- コンテナ
  - RSCS 管理コンテナ

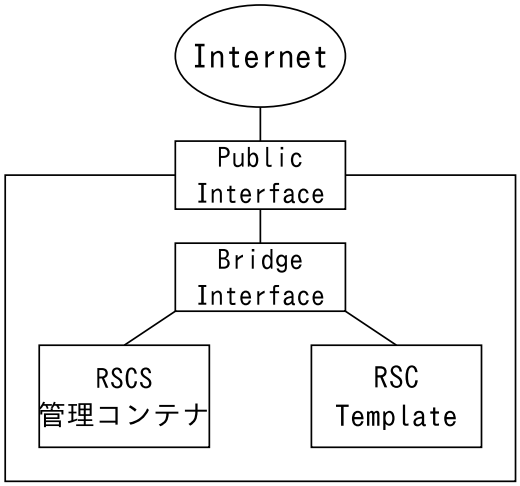

図 4　ホスト環境構成

– RSC テンプレート
– コンテナ管理デーモン

ホスト環境構成の概要を図 4 に示す．RSCS の動作に必要となる最小限のコンテナは，RSCS 管理コンテナと RSC テンプレートコンテナの 2 つである．RSCS 管理コンテナでは，RSCS を維持するためのサービスが実行される．RSC テンプレートコンテナは，本システム内で新たに作成される RSNP サービス環境の元となる，RSNP サーバ環境を用意する．

本稿のシステムでコンテナの仮想化に用いる仮想化技術は LXC である．LXC は Linux 上で利用できる OS 仮想化技術であり，同様の技術に Linux-VServer や FreeBSD Jail などがある．LXC を採用した理由としては，以下の 3 点が挙げられる．

- ハードウェアの制約
  本システムの動作環境として，VPS 上などを想定している．VPS 上では，Intel VT-x のようなハードウェア仮想化支援機構を利用するハイパーバイザ型の仮想化技術である Linux KVM などは利用できない場合が多い．LXC は，ハードウェア仮想化支援機構を必要としない．
- 性能
  OS 仮想化方式では，ひとつのカーネルの上でカーネルのリソース制御機構を利用し複数のコンテナを管理するため，オーバーヘッドが小さい．ただし，異なるカーネルの上でコンテナを動作させることはできない．本システムでは異なるカーネルを用いる必要がない．
- サポート状況
  LXC は標準の Linux カーネルが持つリソース制御機構を利用し開発されており，Linux ディストリビューション側でのサポートも活発である．競合技術となる Linux-VServer においては，標準の Linux カーネルにパッチを当てる形で提供されており，また Linux ディストリビューション側のサポートが無い場合が多いため，保守性に難がある．

ファイアウォールでは，外部公開用ネットワークインターフェースとコンテナ用ブリッジインターフェース上でのパケットフィルタリングと，管理コンテナの公開ポート（リバースプロキシ）へのポートフォワーディング，コンテナ側からの通信のアドレス変換などをおこなう．

アカウントについては，ホスト環境の維持に必要最低限な，ホスト管理者アカウントと SSH によるリモートログイン環境を用意する．このアカウントは，ホスト環境への RSCS の展開やソフトウェアの更新をおこなうためのものである．そのため，RSNP サービス開発者のアカウントなどを用意する必要はない．

コンテナ管理デーモンは，ホスト側からのみおこなえるコンテナの複製・削除や起動・停止といった操作を，RSCS 管理コンテナ側からの要求に従い実行するためのものである．また，コンテナの作成や起動などといった管理操作については，ホスト側の管理者権限が必要となる．

4.2　ゲスト環境（RSCS 管理コンテナ）

RSCS 管理コンテナでは，以下の機能を提供する．

- リバースプロキシサーバ
- DNS サーバ（内部向け）
- RSCS 管理 Web インタフェース

リバースプロキシサーバでは，ユーザと RSNP サーバ，ロボット間の通信を，各コンテナに割り当てられたホスト名を元に振り分けをおこなう．このホスト名の解決には，内部向け DNS サーバを用いる．また，リバースプロキシサーバを経由する際に利用される可能性のある HTTP ベースのプロトコルとして，WebSocket，RSNP などがある．このため，リバースプロキシサーバ側にてそれぞれのプロトコルに対応しており，それらに合わせてパラメータを調整する必要がある．本システムでは，リバースプロキシサーバに WebSocket などに対応した nginx-1.4[7] を用いる．

また，RSCS 管理 Web インタフェースを提供するサービスについても，RSCS 管理コンテナにて提供する．リバースプロキシサーバや RSCS 管理 Web インタフェースをコンテナ化する理由として，コンテナ化による設定や複製，バックアップなどといったシステム管理の簡略化が挙げられる．

4.3　ゲスト環境（RSC）

RSC では，以下の機能を提供する．

- RSNP サーバ
- RSNP 統計管理サーバ
- データベースサーバ

RSNP サーバでは，RSNP サービスの展開と提供をおこ

なう．RSNP サービスについては，Java のサーブレットエンジンである tomcat7 を用い展開する．

RSNP サービスで用いるプロトコルとして，ユーザ側は WebSocket などを用い，ロボット側は RSNP を用いるなどといったことも可能である．そのため，リバースプロキシ同様に tomcat 側にて WebSocket などに対応している必要がある．

また，この RSNP サービス実行時に RSNP 統計管理サーバに通信状況を報告するよう設計することで，RSNP 通信に関する統計情報の収集をおこなう．収集したデータは，データベースサーバに蓄積される．

RSNP 統計管理サーバに蓄積されるデータの例として以下のものがある．

- ロボット ID
  ロボットを一意に識別する ID．
- 発行コマンド
  ロボットに対して送られるコマンド．forward，stop など．
- 実行時間
  ユーザがコマンドを発行してから，RSNP サーバを経由しロボットが応答，RSNP サーバがユーザにレスポンスとして返すまでの時間を，ユーザ，RSNP サーバ，ロボットのそれぞれのチェックポイント毎に計測したもの．
- ロボットの装備データ
  ロボット内の装備（Wi-Fi 通信状況，電波強度，バッテリー残量など）のデータ．
- センサデータ
  センサ（照度，加速度，GPS など）のデータ．
- 利用プロトコル
  ユーザと RSNP サーバ間，RSNP サーバとロボット間のプロトコル．

これらのデータを RSNP サービス側の必要に応じて収集，解析することにより，サービスの検証や品質の向上を目指す．更に，コンテナ内にて RSNP サーバ機能を提供するにあたって不要であるアプリケーションや機能を排除することにより，コンテナの軽量化を図る．

4.4 Web インタフェース

RSCS 管理 Web インタフェースでは，RSNP サーバ基盤を維持するために必要な以下の機能を提供する．

- ネットワークの管理
  LXC ではホストのネットワークインタフェースと別に，複数の仮想ネットワークインタフェースを取り扱う必要があるため，構成が複雑になる．そのため，LXC ホストとなる環境と，ゲスト環境のネットワークの設定（IP アドレス割り当てなど）を簡略化するインタフェースを提供する．

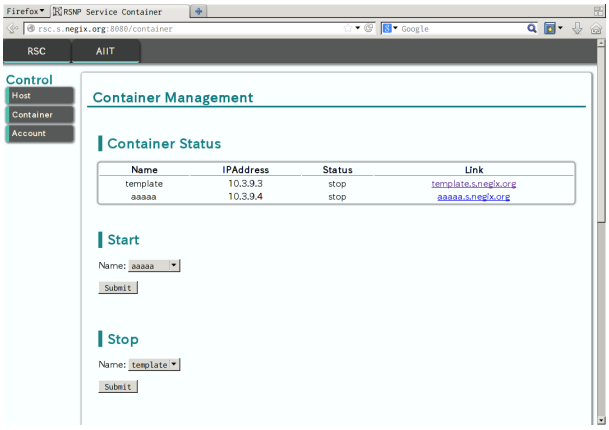

図 5　Web インタフェース

- アカウントの管理
  コンテナ数が多くなると，各コンテナ間でそれぞれ別々にアカウント管理をおこなうのは煩雑となってくる．そこで，LXC ゲスト環境のアカウントを統合管理するインタフェースを提供する．
- コンテナの管理
  RSNP サービスを展開するコンテナは，作業の簡略化と環境の統一化のため，RSC テンプレートを複製し，最小限の設定を適宜変更したものを用いる．これらの作業を更に簡略化するために，コンテナの作成や削除，起動と終了，複製といった操作をおこなうインタフェースを提供する．
- RSNP サービスの管理
  RSNP サービスの展開は，各コンテナに管理者権限にてログインしコマンドラインでの操作をおこなう必要がある．その手間を軽減するために，RSNP サービスの追加や削除をおこなうためのインタフェースを提供する．
- RSNP サービスログの管理
  RSNP サービスの開発においても，通信状況やエラーメッセージなどといったログに出力される情報は重要である．これらのコンテナ毎に出力されるログを統合管理するインタフェースを提供する．
- RSNP サービス統計情報の閲覧
  ロボットを利用したサービスでは，スループットやレイテンシなどがサービスの品質に大きく影響することがある．そこで，RSC 内の通信に関する統計情報を集計，参照できるインタフェースを提供する．

図 5 は，RSCS の管理 Web インタフェースの一部である．この Web インターフェース内で，RSNP サービスの展開に必要な操作は完結する．

5. 考察

RSCS の今後の課題として，以下の点が挙げられる．

- ホスト環境インストーラの作成

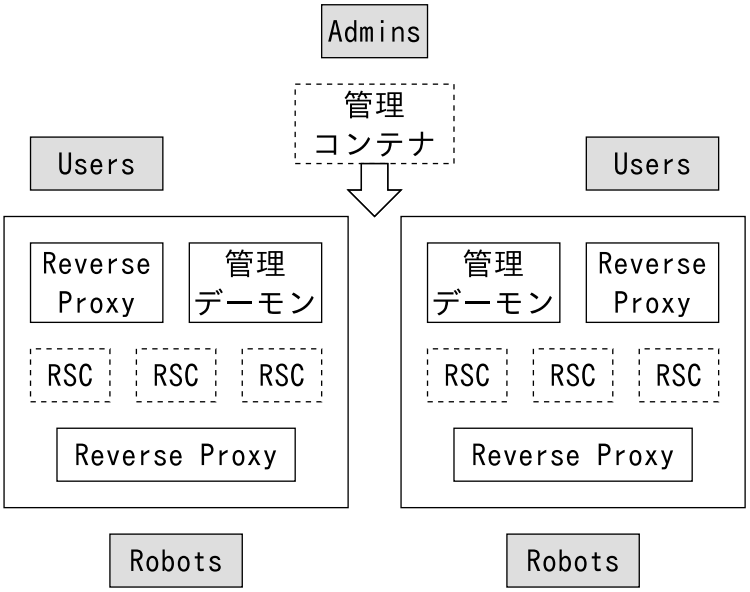

図 6 複数サーバの統合管理

RSCS の導入で最も手間がかかる部分は，サーバの置かれるシステムの環境に依存する，ホスト環境の構築である．ホスト環境の導入が完了すれば，管理コンテナと RSC テンプレートコンテナを展開するのみでシステム構築は完了する．そこで，ホスト環境を構築するためのインストーラを作成し，容易に導入できる形にする必要がある．

- 複数サーバの統合管理
  現状の RSCS では，単一ホスト内でのサービス提供のみを考慮した設計となっている．このため，ハードウェアリソース不足によるサーバの追加などをおこなった際は，別途 RSCS 環境を設定しなおさなければならない．そこで，**図 6** のように，管理コンテナを分離し，ひとつの管理コンテナから複数のホスト上の管理デーモンを管理する形での運用を可能にすることを検討している．
- リソース制御
  現状の RSCS では，コンテナ内プロセスの CPU やメモリ，ネットワーク帯域の使用量に関する制限はおこなっておらず，ひとつのコンテナに負荷が集中すると，ホストやその他のコンテナへの影響が大きく出るなどといった問題がある．コンテナ型仮想基盤内にてリソース制御をおこなっている例としては，文献 [8] などがある．本稿のシステムにおいては，CPU やメモリの使用量などについては LXC，ネットワーク制御についてはホスト側にて制御することが可能であるため，これらを容易に管理し，状態を確認できるインタフェースの追加が必要である．
- LXC の開発状況
  LXC は 2014 年 2 月現在不安定版として開発がおこなわれている状態であり，機能や挙動，コマンド体系などが変化する可能性が高い．このため，LXC 管理サービスの部分において今後変更が必要になる部分が発生すると思われる．
- セキュリティ
  LXC によるコンテナのセキュリティについては，未だ発展途上の部分も多く，コンテナ内からホスト環境に影響が出る操作や，ホスト環境の情報の閲覧などが可能になっている部分がある．（ゲスト環境内の管理者権限での reboot コマンド実行によるホストの停止など）この問題については，SELinux などの強制アクセス制御機構にて制御することも可能ではあるが，管理が煩雑となる．これらの点も含め，LXC や Linux カーネル側の機能追加や修正の状況により今後変化していくと考えられるため，注視が必要である．
- RSCS 以外の汎用コンテナ型仮想化基盤の利用
  LXC を利用した仮想化基盤については，Docker[9] や CloudStack[10] など，コンテナ型，ハイパーバイザ型共に様々なものが開発途上の状態にある．本システムでは，システムの複雑化を避け可能な限り RSNP サーバ環境として特化させ，インタフェースを簡略化することで，メンテナンス性を高めることを意識し開発をおこなったが，これらの汎用仮想化基盤の検証も継続しておこない，機能の取り入れなどを検討する必要がある．

6. おわりに

本稿では，RSNP を用いたロボットサービス基盤の構築・開発・管理を支援する RSCS の設計と実装について述べた．RSCS は現在さくらインターネットの VPS サービ

ス上で運用しており，産業技術大学院大学，はこだて未来大学において RSNP に関する研究用途で利用している．今後，更なる改良やオープンソースソフトウェアとしての公開をおこなう予定である．

謝辞 本研究を進めるにあたり，産業技術大学院大学の青柳一之さん，大橋修さん，落合瑛史さん，常盤嘉昭さん，藤田正典さんから貴重なご意見を頂きました．ここに感謝の意を表します．

参考文献

[1] RSi : Robot Service initiative (online), 入手先 ⟨http://robotservices.org/⟩
[2] RSi : RSNP Tutorial (online), 入手先 ⟨http://www.robotservices.org/wiki/jp/⟩
[3] RSi : RSNP を利用したロボットサービスコンテスト (online), 入手先 ⟨http://robotservices.org/contest/2013/award2013.html⟩
[4] Linux Containers (online), 入手先 ⟨http://linuxcontainers.org/⟩
[5] 土屋 雅稔 : 管理者が安全に交代できる学内ホスティングサービス, 電子情報通信学会論文誌 Vol.J95–B No.10 pp.1264–1272, (2012).
[6] Guoqiang Hu, Wee Peng Tay, Yonggang Wen : Cloud robotics: architecture, challenges and applications, IEEE Network, Vol.26, No.3, pp.21–28, (2012).
[7] Nginx, Inc. : WebSocket proxying (online), 入手先 ⟨http://nginx.org/en/docs/http/websocket.html⟩,
[8] 中田 秀基, 高野 了成, 竹房 あつ子, 工藤 知宏 : ネットワーク帯域予約を用いた分散アプリケーション実行環境の構築, 電子情報通信学会信学技報, NS2009-206, pp.254–258, (2010).
[9] Docker Inc. : Docker: the Linux container engine (online), 入手先 ⟨https://www.docker.io/⟩
[10] Apache CloudStack (online): 入手先 ⟨http://cloudstack.apache.org/⟩