Vol.2018-AVM-100 No.1
2018/3/6

移動型ロボットを活用した接客ロボットシステムの検討

後藤 充裕[1,a)]　松本 猛[1]　倉橋 孝雄[1]　布引 純史[1]　山田 智広[1]

概要：ロボットによる実空間での接客サービスを実現するには，単体のロボットではなく各種センサデバイスや複数のロボットを組み合わせ，ユーザと高度なインタラクションを行うことが求められる．しかしながら，このようなデバイス連携制御は開発者にとって，非常に煩雑な作業となり，システム開発コストの増大に繋がる．そこで，本研究では，複数の機器を簡単に連携制御する技術 R-env：連舞を用いて，移動型ロボットを活用した接客システムのプロトタイピングを行い，連携技術によるサービス開発コストの削減効果を検証した．また，実際の展示会のシステムログの分析結果について報告する．

A Study for Customer Service Robot Systems Using Mobile Robots

Mitsuhiro Goto[1,a)]　Takeshi Matsumoto[1]　Takao Kurahashi[1]　Tadashi Nunobiki[1]
Tomohiro Yamada[1]

1. はじめに

現在，接客や介護，教育など様々な分野におけるインタラクションロボットを活用したサービスが登場している．これらサービスの開発には，単体のロボットだけではなく，距離センサやカメラなどの各種センサを用いて外部状況を正確に認識したり，サービスに合わせて機能の異なるロボットを複数台組み合わせたり，Web サービスと連携させてユーザとの高度なインタラクションを実現する必要があることが多い．特にロボットによる接客サービスにおいては，お客様との音声対話だけではなく，実物体の運搬や目的位置までのナビゲーションといった様々なタスクを実施する必要があり，これらタスクを完遂可能なロボットやセンサ，Web サービスを組み合わせてロボットサービスを実現することが求められる．このような複数のロボットやセンサ，Web サービスを連携したロボットサービスの実現には，(1) 連携する各機器のインタフェース仕様に合わせて相互に情報を送受信する連携プログラムの開発，(2) 連携プログラムを用いてロボットやセンサの挙動を設定しサービス内容の動作シナリオの作成，(3) 現場でのサービス運用結果をフィードバックした動作シナリオの改良といった各種作業工程を経る必要がある．しかしながら，組み合わせるロボットやセンサの数が多くなると，連携プログラムが対応しなければならないインタフェース仕様が多岐に渡り，プログラムの開発が非常に煩雑となる．さらに，このような複雑な連携プログラムを用いて動作シナリオの作成や修正するには，各機器の挙動や依存関係を熟知し，必要に応じて連携プログラム自体の改修も行うプログラミングスキルがシナリオ作成者に求められることになり，各種工程におけるコスト増に繋がる．

そこで，我々は複数のロボットやセンサ，Web が連携したロボットサービスを簡単に作成可能なクラウド対応型インタラクション制御技術 R-env:連舞 ®[1](以下，R-env) の研究を進めてきた．R-env では，各機器が WebSocket をベースにした連携プログラム（以下，R-env アダプタ）を用いてプラットフォームに接続し，規定されたメッセージを R-env アダプタ経由で送受信することで各機器の連携を実現している．また，規定されたメッセージの送受信ルールをグラフィカルプログラミング環境を用いて記述可能としており，高度なプログラミングスキルを開発者に要求しない動作シナリオの作成・改良が可能になっている．

本稿では，この R-env を用いて，トヨタ自動車株式会社

[1] NTT サービスエボリューション研究所
NTT Service Evolution Laboratories
[a)] gotou.mitsuhiro@lab.ntt.co.jp

の生活支援ロボット Human Support Robot(以下，HSR)[*1] やヴィストン社の Sota[*2] といった複数のロボットや測距センサ，タブレット端末などの各機器を連携した接客ロボットサービスシステムの試作を通して得た開発コストの削減効果について述べる．さらに，試作したシステムを実際の展示会へ出展して得たシステムログの分析結果についても述べる．

2. 関連研究

本章では，ロボットサービスの作成に用いられる関連技術について整理する．単体のロボット制御を効率化する技術として Robot Operating System（ROS）[2] や RT-middleware[3] が提案されている．ROS はロボットの各機能をノードという単位でライブラリ化するフレームワークを提供し，ノード間でのデータ送受信を Pub/Sub モデルにて実現している．RT-middleware は，ロボットの機能を標準化した RT コンポーネントという単位でプログラムを実装し，別のロボットでも同じ機能を利用する際の効率化を図っている．また，コンポーネント間でのデータ送受信を同一のインターフェースを持ったポートを接続することにより実現している．これらの技術は，どちらも単体のロボット制御を主目的においたフレームワークであり，複数のロボットやセンサを連携する仕組みまではサポートされていない．一方，R-env では，個々のロボット機能追加は扱わず，ROS や RT-middleware を利用して制御されるロボットも含めて，複数のロボットやセンサを連携したサービス開発をサポートする．

複数のロボットやセンサを連携制御するプラットフォームとして，ユビキタスネットワークロボットプラットフォーム (UNR-PF)[4] や Node-RED[5] が提案されている．UNR-PF はロボットの各機能をモジュールとして抽象化し，モジュール間の連携を HTTP や SOAP プロトコルなどの一般的な通信プロトコルを用いて規定することで，個々のロボット仕様を気にすることなくサービス開発が可能としている．しかしながら，ロボットサービスで活用したい個々のロボットやセンサの持つ機能は様々な種類があり，これら全てをプラットフォーム側で抽象化するのは難しく，規定された抽象化の枠組みでは扱えないデバイスが発生する問題があった．Node-RED は，ロボットやセンサなどのハードウェアデバイスや Web サービスの API を Node.js ベースで連携するプラットフォームである．Node.js の豊富なモジュールを活用してデバイスの機能を簡単に拡張できるのが特徴である．しかしながら，ハードウェアデバイスの制御 API が Node.js をネイティブにサポートしていない場合には，わざわざ Node.js を経て制御 API を実行する方式となり，制御時のオーバヘッド増加や詳細な制御を実施できない問題が発生する．一方，R-env では，WebSocket という通信プロトコルを利用できるプログラミング言語を利用できれば良く，機器の制御 API に合わせて様々な言語を利用して機器連携を実現できる．

ロボットやセンサ，Web サービスを連携する動作シナリオを簡単に記述可能なグラフィカルプログラミング環境として，Choregraphe[6] が提案されている．プログラミング言語に精通していないユーザでも単体のロボットやデバイスの動作シナリオをマウスを用いてグラフィカルに記述できる環境となっているが，複数の機器を連携する仕組みは提供されていない．一方，R-env では，複数機器を連携することを前提としており，それら機器の動作シナリオを 1 つの画面上でグラフィカルに記述でき，簡単にロボットやセンサ，Web サービスを連携したサービス開発が可能である．

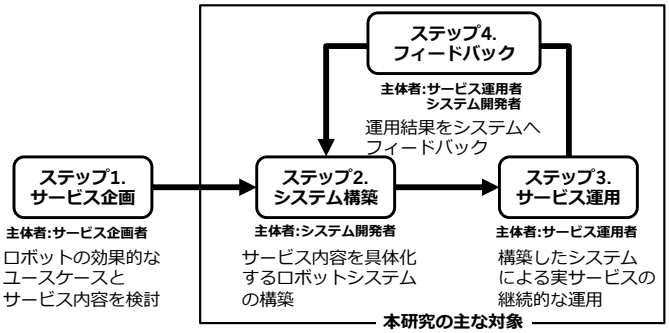

図 1 サービス開発のフロー
Fig. 1 Overview of robot service development.

3. ロボットサービス開発

本章では対象としている複数のロボットやセンサを活用したロボットサービス開発の現状や課題について述べる．

3.1 サービス開発フロー

ロボットサービスの開発フローは，**図 1** に示す通り 4 ステップにより実施され，実施する主体者が各ステップにより異なる．

ステップ 1:サービス企画 サービス企画者がそのサービスが対象とするユーザの選定やロボットが効果的なユースケースを検討し，サービス内容を決定するステップである．

ステップ 2:システム構築 システム開発者がロボットやセンサを連携に必要な連携プログラムの開発したり，システム全体の動作の流れを定義する動作シナリオ作成を行い，システムとして具体的に構築するステップである．

ステップ 3:サービス運用 サービス運用者が構築システムを用いて対象ユーザへのサービス提供を行い，必要に

[*1] http://www.toyota.co.jp/jpn/tech/partner_robot/family_2.html#h21
[*2] https://sota.vstone.co.jp/home/

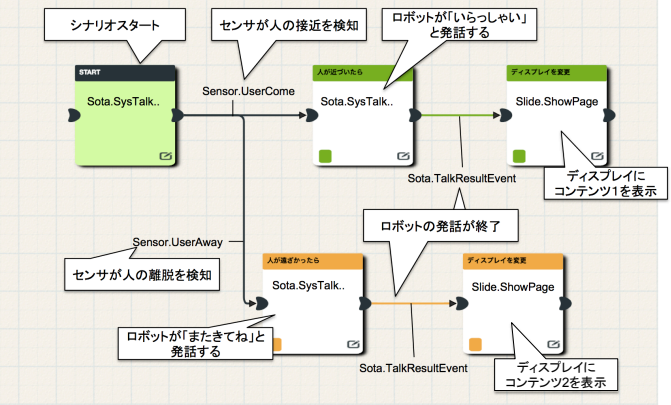

図 2　R-env による動作シナリオ記述の例
Fig. 2　Example of scenario description by R-env.

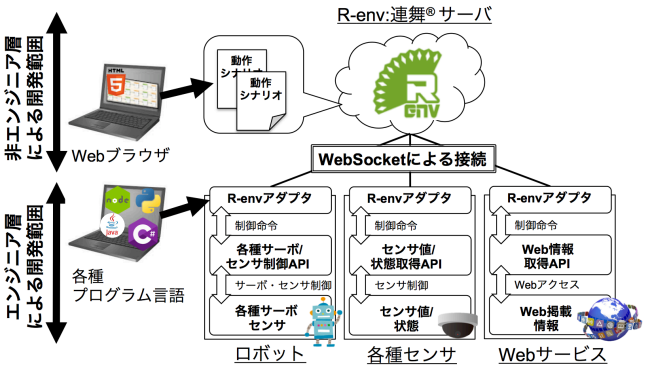

図 3　R-env:連舞 Ⓡ の概要
Fig. 3　Overview of R-env.

応じて故障した機器を交換しながらシステムを継続的に運用するステップである．

ステップ 4:フィードバック　サービス運用者がシステム運用により得た修正点や改善点をもとに，システム開発者がシステムの動作シナリオや連携プログラムを改良するステップである．

以上の通り，このようなサービス開発フローでは前のステップにおける実施結果を次のステップに用いてサービス開発を進めていく．そして，このフローをステップ 1 からステップ 3 の各ステップを 1 回ずつ実施するだけではなく，図 1 上部の矢印で示した通りステップ 4 のフィードバックを何度も繰り返しながら，対象とするユーザを満足させるサービス開発を行うことになる．

なお，本研究では，企画されたサービス内容をもとにシステム構築や運用をサポートするかを主眼に置き，ステップ 1 を除いたステップ 2〜4 を主な対象としている．そのため，以降では，ステップ 2〜4 について詳述していく．

3.2　ロボットサービス開発における課題

現状のロボットサービス開発における課題について，ステップ毎に述べていく．

ステップ 2 においては，多くの機器を連携するシステム構築に高度なプログラミングスキルが求められることが挙げられる．多くの機器を連携する場合には，各機器の連携プログラムが対応しなければならないインタフェース仕様が多岐に渡り，プログラム開発が非常に煩雑になり，開発コストの増加に繋がる．そして，そのような連携プログラムを組み合わせてシステム全体の動作シナリオを作成するには，各機器の挙動や依存関係を熟知して，各機器の制御を行う必要があり，これもシステム構築を複雑にしてしまう要因となる．

ステップ 3 においては，エンジニア層であるシステム開発者と，異なるスキルや知識を保有したサービス運用者によるシステムの運用が挙げられる．サービス運用者は，エンジニア層であるとは限らず，非エンジニア層がステップ 2 で構築したシステムを運用しながらサービス提供を行うことも多い．そのため，システム起動方法や故障した機器の交換などを簡易に実施できる必要がある．

ステップ 4 においては，実運用結果のフィードバックによる動作シナリオの改良にも高度なプログラミングスキルが求められることが挙げられる．ロボットの発話内容や情報提示のタイミングなどの軽微なシナリオの改良を行う場合にも，上述した通り連携したい各機器の挙動や依存関係の熟知が求められる．また，改良したい動作シナリオを実現する上で，連携プログラムの改修が必要な場合には，改修作業も実施する必要がある．これらをまとめると以下の通りとなる．

課題 1　エンジニア層による連携プログラムの開発・改修を容易とすること．

課題 2　非エンジニア層による動作シナリオの作成・改良を容易とすること．

課題 3　非エンジニア層によるシステム運用を容易とすること．

3.3　R-env:連舞 Ⓡ によるアプローチ

上述した課題を解決するための R-env のアプローチについて述べる．まず，課題 1 の解決には，連携したい各機器を疎結合として，R-env サーバ経由で各機器を連携するサーバクライアント方式を採用している．この方式を採ることで，規定されたフォーマットによるメッセージを WebSocket 経由でサーバ・機器間でやり取りし，そのメッセージを機器の制御・情報取得 API のインタフェース仕様に基づき制御命令に変換して機器制御を行う R-env アダプタを各機器用に開発するだけで連携を実現できる．また，R-env アダプタは WebSocket を利用可能な各種プログラミング言語を利用して開発すれば良いため，Node.js や Java, Python, C#といった主要な言語での開発が可能となる．

次に，課題 2 の解決には，各機器が有している機能を選

択するだけでその機能を実行したり，各機器間での連携に必要なメッセージの送受信をどのように制御するかをグラフィカルプログラミング環境で提供する方式を採用している．この方式を採ることでシステム全体の動作シナリオを状態遷移図としてグラフィカルに記述するだけで各機器の制御や連携を実現できる (**図 2**)．また，このプログラミング環境は HTML5 対応の Web ブラウザ経由で利用可能であり，特別なアプリケーションのインストールを不要として開発を行うことが可能である．

最後に，課題 3 の解決には，課題 2 の解決で提供した開発環境が同時にシステムの運用環境となる方式を採用している．システム動作シナリオの作成・改良時にシナリオをデバッグしていた開発環境がそのまま運用環境となるため，高度なシステム操作を要求せず，非エンジニア層でも簡単にシステム起動などの運用作業を実施できる．また，機器の交換作業も，故障機器と予備機器を入れ替えて，動作シナリオ上での機器の対応を書き換えるだけで実施できるため，運用が非常に容易となる．

このようなアプローチにより，**図 3** の R-env の概要で示した通り，ロボットシステム構築におけるエンジニア層と非エンジニア層の開発範囲を分担することが可能になり，サービス運用から得た修正点・改善点を素早くシステムへフィードバックできるようになる．例えば，動作シナリオの軽微な改良は，非エンジニア層により実施可能となり，ロボットの発話内容を毎日変更するといったことができる．R-env アダプタの開発を伴う新規デバイスの追加についても，他デバイスの依存関係を気にせずにエンジニア層が短期間での開発が実施でき，運用上システムに必要な機能を簡単に追加できる．

4. 接客ロボットシステム

本章では R-env を活用してロボット接客サービスの実現に向けて試作したシステム概要やシステムの試作で実際に連携したロボット・センサについて述べる．本研究で対象としたサービスは，展示会や街頭での商品宣伝ブースにおける接客サービスである．このようなサービスでは，呼び込みを行いながら，来訪者がブースに接近したのを契機として，宣伝対象となる技術や商品をブース来訪者に説明する．そして，一定時間以上の説明を聞いた来訪者に対してノベリティを配布し，最後にブース内容に関するアンケートを実施するといった流れで接客を行うことを想定している．今回は技術紹介を行う展示会ブースを対象として，これらブースにおける接客サービスをロボットにより自動化したシステムを構築した．

4.1 システム概要

本システムは**図 4** に示した通り，R-env サーバと音声合成・認識サーバ，Web コンテンツ提供サーバの各種サーバと，測距センサや音声対話ロボット (Sota)，スライド表示デバイス，移動型ロボット (HSR)，タブレット端末といった各種機器により構成される．なお，音声合成・認識サーバは，Sota が来訪者と音声対話する際に利用され，Sota の R-env アダプタ経由で音声発話や認識時にアクセスされる．Web ページ提供サーバは，スライド表示デバイスやタブレット端末で必要なコンテンツを表示する際にアクセスされる．また，各種機器は R-env により連携されており，R-env 上で記述した動作シナリオに従った動作を実行する．

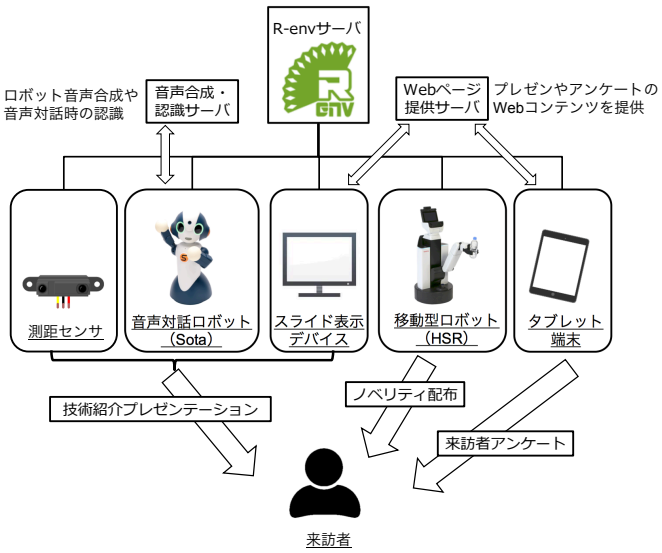

図 4 試作した接客システムの概要
Fig. 4 Overview of prototyping system for customer service.

4.1.1 技術紹介プレゼンテーション

Sota が来訪者への呼び込みをしながら，測距センサを用いて来訪者のブースへの接近をチェックする．来訪者の接近を検知すると，Sota がブースで展示している技術に関するプレゼンテーションを開始する．このとき，音声での説明だけではなく，スライド表示デバイス上にプレゼンテーション用のスライドを表示しながら，音声による説明とスライドを連動したプレゼンテーションを実施する．プレゼンテーション中は，常時測距センサによりユーザが滞在しているかを確認しており，ユーザが途中でブースから離脱するとプレゼンテーションを途中で終了して，呼び込み状態に戻るようになっている．また，プレゼンテーション中に来訪者からの質問への対応が可能となっており，来訪者が不明点などを質問をすると，Sota がプレゼンテーションを中断して音声対話による質疑応答が行われる．そして，質疑応答が終わると，中断していた箇所からプレゼンテーションを再開するようになっている．

4.1.2 ノベリティ配布

来訪者が一定時間ブースに滞在すると，Sota と HSR が連動しながら，棚に陳列してあるノベリティのペットボトルを来訪者近くのテーブルまで運搬し，ノベリティ配布を

行う．今回のシステムでは，Sota との音声対話を通して来訪者が 3 種類のノベリティから 1 つを選択すると，HSR がユーザに選択されたノベリティを棚から 1 つピックアップして運搬する．ノベリティをピックアップする HSR は，移動機能だけではなく物を把持するアームを有し，任意の物体の運搬が行える．また，今回はノベリティを置いた棚にアーム位置を調整するマーカを設置し，そのマーカをステレオカメラで認識し，アーム位置を正確に調節しながらノベリティのピックアップを実現している．

4.1.3 来場者アンケート

ノベリティ配布が終わると，来訪者に対してブース内容に関するアンケートをタブレット端末を用いて取得する．今回のシステムでは，「ブースの展示内容が分かりやすかったか」や「ブースのコンセプトに可能性を感じるか」，「ブース説明に満足できたか」といった設問に対して 5 段階評価で回答するアンケートをタブレット端末を介して来訪者へ提示する．そして，提示されたアンケートに対して，来訪者が設問に全て回答するか，回答をスキップするボタンを選択すると，Sota と HSR が来訪者に対して御礼を述べて，ブースによる接客を終了する．

5. ケーススタディ

前章で述べてきたロボット接客システムの試作を通して，R-env によるロボットサービス開発時のコスト削減について検証した．また，実際行われた展示会において，本システムによる接客サービスを 4 日間運用し，来訪者がロボットによるプレゼンテーション後にどれだけアンケート回答に応じてくれるかについて評価した．

5.1 R-env アダプタ経由での HSR 制御の妥当性

本システムで利用した HSR は ROS による制御 API(C++と Python) を有して [7] おり，本システムでは，この ROS-API を R-env アダプタ経由で実行し制御する方式を採っている．エンジニア層が R-env アダプタ開発を容易に行い，かつ，制御方式が妥当であるか検証することを目的として，本システムで採用している R-env アダプタ経由で制御した場合と，直接 ROS を利用して制御した場合の制御ブログラムの実装量や各方式での処理時間の比較からオーバヘッドを計測した．

以下に示す方式 1 と方式 2 において同一のタスクを実行し，その処理時間を計測した．

方式 1 R-env アダプタ経由で ROS-API(Python 版) を実行して HSR を制御する

方式 2 ROS-API(Python 版) を直接実行して HSR を制御する

1 回のタスクとしては，**図 5** に示した通り，指定座標への移動やマーカによる移動，物体の把持や開放といった各種制御を座標 P_0 から座標 P_4 にてそれぞれ実施する．このタスクを 5 セット繰り返し，各制御に要した処理の平均時間を比較した．また，方式 1 で利用する R-env アダプタ経由でのプログラムは，直接 ROS を利用する方式 2 の制御プログラムをベースとして，R-env サーバと送受信するメッセージを規定する処理を 250 行程度追加することで実装している．

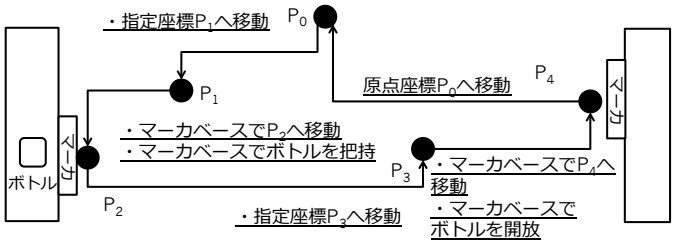

図 5 処理オーバヘッドの検証タスク
Fig. 5 Verification task of processing overhead.

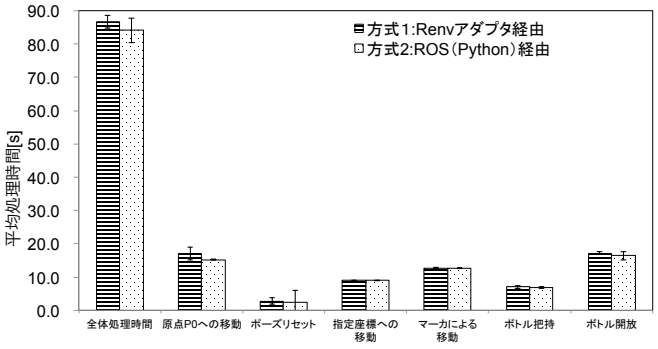

図 6 各方式における平均処理時間
Fig. 6 Average processing time with each method.

図 6 に各方式における平均処理時間を示す．全体処理時間の平均を比較すると，方式 1 が方式 2 に比べて平均で 2.6 秒程度遅延したが，大きな差が生じないことが分かった．また，各種制御の処理時間ごとの平均を比較すると，原点位置への移動処理において，方式 1 が方式 2 に比べて平均で 2.1 秒程度の遅延となった．他の制御の処理時間においては，1.0 秒以下の遅延となり，ほぼ遅延が発生することが無かった．従って，R-env アダプタ経由で ROS-API を実行し，HSR を制御する方式 1 を採用した場合にも，処理時間のオーバヘッドはほぼ発生せずに制御可能である．そして，プログラムの実装量としても，方式 2 に対して高々 250 行程度の追加で R-env アダプタ化が可能であることを確認した．このことから，R-env を活用してシステム構築を行うことは，現状のサービス開発フローの課題 1 における「エンジニア層による連携プログラムの開発・改修の容易化」に寄与できることが分かった．

5.2 非エンジニアによる動作シナリオ作成・改良

非エンジニア層が動作シナリオを R-env を用いて実行できるかを検証するために，ROS プログラミングが未経験

で，日常業務においてプログラミングに携わっていない弊社社員を実験者として，5.1 節で述べた HSR の制御タスクを R-env の動作シナリオとして作成してもらい，その所要時間を求めた．

実験者には，動作シナリオの前提となる HSR と 5.1 で利用した R-env アダプタに関するマニュアルを提示し，「HSR がペットボトルが置かれた机から反対側にある机まで把持して移動させる動作シナリオを作成してください．ペットボトルボトルの把持にはマーカを利用して下さい」という指示を与えた．また，HSR や R-env に関する事前レクチャを実験者に対して行わず，実験者がマニュアルベースでシナリオ作成を進めた．

実験者が指示した動作シナリオを作成するのに要した時間は約 7 時間であった．内訳は以下の通りとなる．

HSR の前提知識の習得　約 2 時間
R-env の前提知識の習得　約 2 時間
R-env による動作シナリオ作成　約 3 時間

このように短時間で動作シナリオ作成が行えた要因として，ROS-API を用いて実装する場合に比べて，制御したいロボットの機能（移動や把持など）のみに集中してシナリオ作成ができたことが挙げられる．また，シナリオ作成時に，例えば「座標 P_3-座標 P_4 間で実施する制御」といったタスクの一部を連続してテストしたり，移動したい座標のパラメータを微調整するといったデバッグ作業も簡単に行えたという意見も得た．これらから，R-env の活用により現状のサービス開発フローの課題 2 における「非エンジニア層による動作シナリオの作成・改良の容易化」に寄与できることが分かった．また，動作シナリオを作成した実験者によるシステム起動や異常停止時の再起動についても実験者自身で容易に行えることも確認し，R-env が課題 3 における「非エンジニア層によるシステム運用を容易化」へも寄与できることが分かった．

5.3 ロボットシステムによる来訪者アンケート回答率

4 日間の展示会に本システムを展示し，来訪者に対してタブレット端末によるアンケートへの回答を求めた．この際に，実際にアンケートへ回答した来訪者数と回答をスキップした来訪者数を比較して，ロボットによる接客サービスを自動化しても，どの割合でアンケート取得が可能かを検証した．なお，このアンケートはダミーの設問であり，実際に各設問の回答は集計せずに，アンケートに回答した来訪者数とスキップした来訪者数のみを計測している．

計測結果は表 1 に示した通りであり，4 日間合計で 352 名の来訪があり，その内の約 85%の来訪者が実際にアンケート回答を行った．日単位で見ると，概ね 80%以上の来訪者がアンケート回答を行ったという結果を得た．この結果より，ロボットを用いた接客システムにおいてアンケート機能を実装した場合に，多くの来訪者のアンケート回答を取得できる可能性が高いことが分かった．

表 1　来訪者数とアンケート回答数
Table 1　Number of visitors and questionnaire responses.

日付	1 日目	2 日目	3 日目	4 日目	合計
(a) 来訪者数	96	103	66	87	352
(b) アンケート回答数	78	89	52	80	299
回答率 ((b) / (a))	81.3%	86.4%	78.8%	92.0%	84.9%

6. まとめ

本稿では，ロボットやセンサなどの複数の機器を連携し，移動型ロボットを活用した接客ロボットシステムの実現に向けたサービス開発フローについて述べるとともに，複数の機器を連携制御する技術 R-env:連舞 ® が現状のロボットサービス開発フローにおける課題解決に寄与することを検証した．また，実際の展示会におけるシステムログの分析結果より，試作した接客ロボットシステムがブースに訪れた全来訪者のうち，約 85%の来訪者からアンケート回答を得ること可能であることを明らかにした．

今後の課題として，ロボットサービス開発フローにおける R-env による開発コスト削減に関する詳細な効果検証や，各ロボットの搭載されたカメラを用いて来訪者の状態を検知して高度なインタラクションを実現する手法の考案がある．

謝辞　本研究を進めるにあたり，HSR に関して度重なるサポートを実施して頂いたトヨタ自動車株式会社 未来創成センターのみなさまに感謝致します．

参考文献

[1] 松元 崇裕, 松村 成宗ら：「*R-env*:連舞 ^{TM}」クラウド対応型インタラクション制御技術, 2017 年度人工知能学会全国大会予稿集, (2017).
[2] Eitan, M. E., Eric, B., Tully, F., Brian, G. and Kurt, K.: *The Office Marthon : Rubust Navigation in an Indoor Office Environment*, International Conference on Robotics and Automation 2010(ICRA 2010), (2010).
[3] Ando, N., Suehiro, T. et.al : *RT-Middleware: Distributed Component Middleware for RT (Robot Technology)*, International Conference on Intelligent Robots and Systems 2005 (IROS2005), pp.3555-3560, (2005).
[4] Nishio, S., Koji, K. and Norihiro, H.: *UbiquitousNetwork Robot Platform for Realizing Integrated RoboticApplications*, International Conference on Intelligent Autonomous Systems(IAS-12), (2013).
[5] Blackstock, M. and Lea, R.: *Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED)*, International Workshop on Web of Things 2014(WoT2014), pp34-39, (2014).
[6] Pot, E., Monceaux, J., Gelin, R. et.al : *Choregraphe: a graphical tool for humanoid robot programming*, International Conference on Robot and Human Interactive Communication 2009(RO-MAN2009), (2009).
[7] 寺田 耕志, 村瀬 和都, 山本 貴史.: 生活支援ロボット *Human Support Robot*（*HSR*）における *ROS* 活用, 日本ロボット学会誌, Vol. 35, pp280-283, (2017).