周辺環境とインタラクション可能な音声対話用 BLE ビーコンの開発

佐野　敦志† 　堤　修平† 　山本　大介† 　高橋　直久†

† 名古屋工業大学大学院工学研究科情報工学専攻　〒 466–8555 愛知県名古屋市昭和区御器所町
E-mail: †cko17545@stn.nitech.ac.jp, ††{tsutsumi.shuhei,yamamoto.daisuke,naohisa}@nitech.ac.jp

あらまし　近年, BLE(Bluetoooth Low Energy) 機能を搭載した機器や端末が普及しており, 特にビーコンを利用したシステムが急増している. また我々の研究室では, 音声対話システム構築ツールキット MMDAgent を Android 向けに移植した Android 版 MMDAgent の研究を行なっている. MMDAgent の音声対話は, 人とエージェント間で行われる. エージェントは人間のように話すことはできるが, 人間のように話者の周辺環境やその変化を知る術は持たない. 加えて Android 版 MMDAgent は移植元の PC 版とは異なり, ポータブル端末での使用が前提にあるため, 利用場所により周辺環境は大きく異なる. この問題を解決するために, MMDAgent の従来の音声対話にセンサからの取得情報を用いた「環境」を追加し, 人・エージェント・環境の三者間での音声対話を試みた. また「環境」については, 端末ごとに差異が生じないように, それを専用に取り扱うマイコンを別途用意した.

本論文では,「環境」を含めた音声対話が可能なプラットフォームの開発及びその「環境」を取り扱うビーコンを制作した. また提案システムの有用性についての評価実験と共に, これらを利用した応用例の紹介も行う.

キーワード　MMDAgent,　Arduino,　BLE,　ビーコン

1. はじめに

近年, BLE(Bluetooth Low Energy) 機能を搭載した機器や端末が著しく普及している. それに伴い, BLE を利用したサービス, 特に iBeacon を始めとするビーコンを用いたサービスが多く見受けられるようになってきた. 例としては, 株式会社エンプライズが開発した「HMV アプリ」[3] や, 本学 (名古屋工業大学) で運用されている「Nitech ピロリン」[5] などがある. その他ビーコンを利用したもの以外にも, 株式会社ニコンのカメラと一緒に使用するアプリ「SnapBridge」[9] のようなサービスもある.

また, 我々の研究室では, PC 向け音声インタラクションシステム構築ツールキット MMDAgent [10] を Android 向けに移植した Android 版 MMDAgent [13](図 1) を開発し, Android 版 MMDAgent を用いて携帯端末向け音声対話システムに関する研究を行っている. MMDAgent ではユーザが FST(Finite State Transducer) 形式 [11] に基づいたスクリプト (以下, FST スクリプト) を編集することで任意の対話内容を設定できるという特徴がある. FST スクリプトとは, ユーザの入力内容に応じて, 設定した内容を音声出力したり, 3D キャラクタに動作をさせたりするといった流れを FST ファイルと呼ばれる外部ファイルに記述した台本のようなものである. Android 版 MMDAgent において, 対話内容はスマートフォン内にある FST ファイルの記述内容に依存するという性質をもつ. そのため利用者の置かれている状況に応じた対話を適切に行う事は簡単ではない. 例えば, 質問する部屋の名前を指定して質問した場合は, その部屋についての詳細な説明も可能だが, MMDAgent に「ここは何処か. 」という部屋自体のことを聞かれても, それを判断する術を持たないためである. また他に, 会話しているその部屋がとても寒いとした場合, 人間はその状況を理解することが可能だが, MMDAgent はそれを理解できない.

そこで本論文では, 利用者の周囲の状況に応じた音声対話可能なシステムを提案する.

2. 提案システムの概要

2.1 Android 版 MMDAgent の特徴

MMDAgent は, 音声対話のための高度な機能を備えたツールキットであり, 音声認識 [14], 音声合成, 3D モデルの描画や物理演算などを統合したシステムである. これを Android 向けに移植したものが, Android 版 MMDAgent(図 1) である. 本研究では, Android 版 MMDAgent を使用する.

また MMDAgent は, Apple の Siri [1] を始めとする他の多くの音声対話システムと異なり, ネットワークを介さないため, 遅延の少ない対話が可能である.

その他, MMDAgent は FST ファイルを読み込むことで, その内容に従った音声対話を実現している. FST スクリプトは, 4 つのフィールド, 状態番号, 次状態番号, 条件メッセージ, 実行メッセージから構成され, この順で記述される. FST スクリプトは, 有限状態遷移機械に基づいているので, 状態遷移図で表すこともできる. 図 2, 3 は, MMDAgent が「こんにちは」と認識した際に, お辞儀をしながら「こんにちは」と返答する例を, それぞれ FST ファイルに記述された形式と状態遷移図に示したものである. なお FST スクリプト例では, 4 つのフィールドをそれぞれ黒 (状態番号), 緑 (次状態番号), 橙 (条件メッセージ), 青 (実行メッセージ) で色分けして表記しているが, 実際に色分けされているわけではない. また今後, 本論文での FST スクリプト例の色分けは, これに従う.

図 1　Android 版 MMDAgent の画面例

```
1  10  RECOG_EVENT_STOP|こんにちは   <eps>
10 11  <eps>                        MOTION_ADD|mei|greet|greet.vmd
11 12  <eps>                        SYNTH_START|mei|normal|こんにちは
12 1   SYNTH_EVENT_STOP|mei         <eps>
```

図 2　FST スクリプトの例

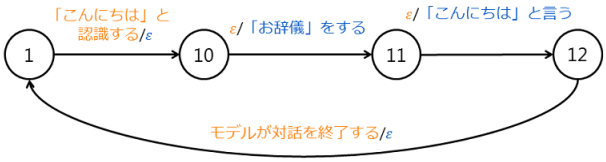

図 3　状態遷移図

2.2　既存技術の問題点と本研究の目的

既存の MMDAgent において多くの音声対話は，ユーザとエージェントの間でのみ行われていた．この音声対話では，予め用意した FST ファイルの内容に依存するために，周囲の状況を捉えて会話することは，ほぼ不可能である．関連研究として，MMDAgent の利用者の位置に応じて音声対話内容を更新する研究 [15] がある．関連研究では，スマートフォンの GPS 機能を利用し，その位置情報に適した FST ファイルを逐次配信することで，現在地に適した音声対話を行う事を可能とした．しかしながら，GPS を利用した屋外での利用が前提のため，室内で会話をしている状況での利用は困難である．そこで本研究では，現行の音声対話システムに，新たに「環境」という要素を関与させ，「ユーザ・エージェント・環境」の三要素間での音声対話を試みる．ここでいう「環境」とは，MMDAgent ユーザ周囲の状況や状態を表し，本研究では特に人が会話する範囲を中心とした．「ユーザ・エージェント・環境」の三要素間での音声対話の例としては，部屋のドアに特定の装置を取り付けることによって，音声対話でドアを開けたり，或いは，そのドアの前にユーザが来たことを感知して MMDAgent が音声対話を開始したりするといった事である．この例のように，提案システムでは「環境」の影響を音声対話に取り込むのみではなく，音声対話を利用して「環境」へ働きかけることも可能とする．

即ち本研究の目的は，従来の MMDAgent の音声対話に「環境」を含め，前述の例のような音声対話が可能なプラットフォームの開発である．

2.3　アプローチ

目的への実現に向けて，以下のアプローチで解決を図った．

アプローチ 1　「環境」とのインタラクションを実現するため、Arduino [2] と連携
アプローチ 2　Arduino と MMDAgent の通信に BLE を利用
アプローチ 3　MMDAgent の仕様に準拠

本研究では「環境」とのインタラクションを，Arduino と連携することで実現した．使用した Arduino についての詳細な説明は，2.4 節にて行う．しかしながら，Arduino と通信する機能が別途必要になるという新たな課題が発生した．Arduino には元々，USB ケーブルを用いて有線接続で行うシリアル通信機能が標準で備わっているが，本研究においては Android 版 MMDAgent を使用するため，使用者はスマートフォンを持って自由に移動できるという前提がある．そのため，行動範囲に制限のかかる有線接続は適さないと考えられる．そこで本研究では使用者がスマートフォンを持ったまま自由に移動しながら「環境」とインタラクション可能なように，近距離無線通信に適した BLE 通信を使用することにした．

そして，MMDAgent はイベント実行型プログラムの FST スクリプトで制御が行われている．そのため，FST スクリプトで制御可能なコマンドを定義することで、「環境」とのインタラクションを FST 形式で記述可能にした．

2.4　Arduino

Arduino とは，マイコンチップを実装した基盤，マイコンボードと，Processing から派生したプロジェクトである Wiring [12] をベースとした，Arduino 独自のプログラミング言語から構成される開発システムである．また単純に基盤であるマイコンボードの事のみを指して呼ぶ場合もある．Arduino の特長としては，ハードウェア自体は非常に安価なものが多く，更にソフトウェアの開発環境は無料で使用できる．また低消費電力で稼動できる．

さて 2.3 節で述べた通り，本研究では Arduino と MMDAgent との通信方法に BLE 通信を使用することにした．そこで本研究においては，Arduino のボードとして「RedBearLab nRF51822」[8](図 4) を使用した．これは，マイコンボード内に BLE 通信を可能にする Nordic 社製のチップ”nRF51822” [6] を予め搭載しており，mbed や Arduino の互換ボードとしての使用ができる．また BLE 通信を用いることで，データ処理がスマートフォンとマイコン間のみで完結し，その他の外部ネットワークやサーバを経由しないため，MMDAgent の特徴であるネットワークを介さない運用が可能で，マイコンボード設置場所の自由度が高いという利点もある．

本研究では，MMDAgent 内のマイコンに関わる拡張機能及びマイコン本体について「MeiBeacon」と命名した．

2.5　提案システムの構成

本研究のシステム構成図を図 5 に示す．図 5 内では都合上，一部の機能名を省略した．システムは大きく分けて次の 3 エリ

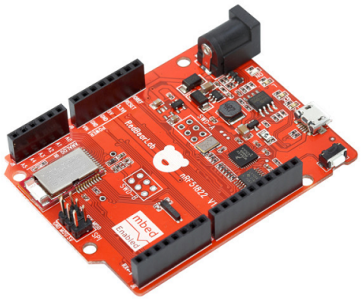

図 4　RedBearLab nRF51822

アから構成される.

（1） MMDAgent の機能の拡張を行った「MMDAgent-BLE 連携機能」「イベント生成機能」「MeiBeacon 操作機能」の部分
（2） Arduino 内の制御を執り行う部分
（3） Arduino を MMDAgent で扱うようにしたこれらを取りまとめた部分, 即ち MeiBeacon と呼称する領域.

この内, MMDAgent の拡張機能となる「MMDAgent-BLE 連携機能」「イベント生成機能」「MeiBeacon 操作機能」についての詳細な説明は, 第 3. 章で示す.

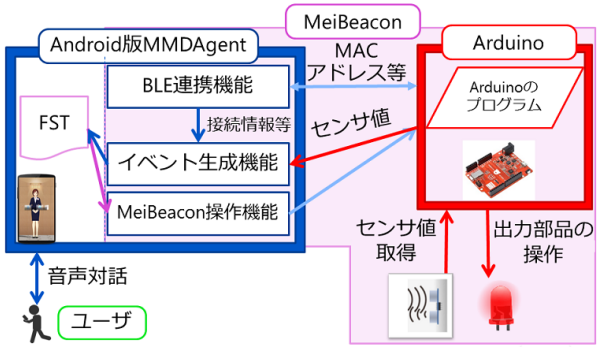

図 5　提案システムの構成図

3.　提案システムの実現法

提案システムは図 5 に示した端末及び機能郡からなる. それぞれの実現法を以下に示す.

3.1　MMDAgent-BLE 連携機能

本節では BLE 通信のために必要な, 通信の確立から接続, 各種データの送受信を行う機能を説明する.

3.1.1　BLE 通信の確立から接続について

通信の確立から接続の流れを図 6 に示す. 今回のシステムでは, MMDAgent 側は常に最も近い MeiBeacon との接続を前提としている. これはモバイル端末 MMDAgent を使用するユーザは, その端末を手に持って使用すると考えられ, そのユーザと関連性の高い「環境」とインタラクションするためである. ここでユーザの端末と MeiBeacon とのおおよその距離を測るために, RSSI(受信信号強度) を利用した. RSSI 値は, 端末と MeiBeacon の距離が近いほど大きくなる. システムの流れは次の通りである.

（1） 起動時又はユーザ側のリクエストなどに応じて, 周辺 MeiBeacon の探索を開始する.
（2） 発見した MeiBeacon が, 初めて発見した機器ならば MeiBeacon リストに保存する.
（3） 発見した MeiBeacon の RSSI 値をチェックし, その MeiBeacon で過去に記録した値より大きい場合はその値を上書きして記録する. 初めて発見した機器だった場合は無条件でその値を記録する.
（4） RSSI 値を（3）で記録した場合, 接続候補に選定し, （5）以降の処理を行う. 記録していない場合は, 次に発見した MeiBeacon の処理をするために（2）へ戻る.
（5） 探索開始後, 接続候補を選定する処理が 1 回目なら MeiBeacon とその RSSI 値を記録する. 2 回目以降なら, 過去に記録した接続候補機器との RSSI 値を比較し, 大きい場合はその MeiBeacon とその値を上書きして記録する.
（6） 終了後, 次に発見した MeiBeacon の処理をするために（2）へ戻る.
（7） 設定された探索時間終了後, 接続候補に記録されている MeiBeacon との接続を開始する.

なお探索開始時に, 記録されている各 RSSI 値は別途保存された後にリセットされ, その探索では前回探索時の RSSI 値の影響を各 MeiBeacon は受けない. 即ち各探索開始後, 発見された各 MeiBeacon における最新の RSSI 値を再度取得し直している. 但し MeiBeacon リストはリセットされず, 今回の探索で新たに発見された MeiBeacon はリストの末尾に追加される. またアプリ終了時には, 全ての記録内容はリセットされる.

コネクション確立後も, 接続中 MeiBeacon の RSSI 値を一定間隔で常に受け取る. 一定時間, 接続中 MeiBeacon から一定距離ユーザが離れている場合には自動的にその MeiBeacon と切断する. 加えて, 現在接続中の MeiBeacon より近くに, 他の MeiBeacon が発見された場合には, それに接続先を自動的に変更する. このように, 自動で切断及び最も近い機器に接続しなおすことで, 関連性の低い「環境」とインタラクションする可能性を軽減した.

なお, MMDAgent 起動時のコネクションの確立は, BLE 機器との接続を優先しており, 探索時間修了前に, 接続を開始する場合がある. また, 起動時にコネクションの確立を行うのは本システムの仕様である.

3.1.2　BLE 通信におけるデータの送受信について

MeiBeacon とのコネクション確立後, 各種データの送受信においては配列を用い, 0 番に用途識別用のデータを, 他の部分に各種必要データを格納している. 但し, コネクション確立後も BLE 通信の確立及び接続等に必要な各種データは, この方法で取り扱わず, 接続時の方法で使用した.

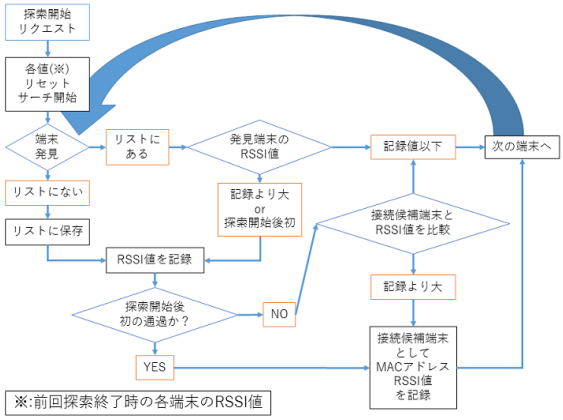

図 6　BLE 通信の確立から接続

3.2　イベント生成機能

MMDAgent はイベント実行型プログラム制御のため, 直接「環境値」を参照しての処理には不向きである. 「環境値」とは, Arduino から「環境」として取得した値や情報の本論文固有の呼称である.

そこで, センサから送られてきた「環境値」を読み取り, それを MMDAgent 内のプログラムで, FST スクリプトで使用する各種イベントへの変換機能を実装する. 「環境値」は, 使用するセンサによって意味合いは異なるため, 予め「環境値」の扱い方を MMDAgent 内のプログラムで定義しておく必要がある. この個別の FST イベントを MMDAgent が認識することで, MMDAgent が音声対話を始めとする出力メッセージの内容に応じた動作を行う. 実装した FST イベントの例と使用例を図 7 に示す. 使用例の FST スクリプトの見方は 2.1 節に示したとおりである.

BLE_XX_(DIS)CONNECT

- BLE機器との接続/切断にXXした際に発行されるイベント
  - XX=ALREADY(既に), MISTAKE(失敗), OK(成功)

TEMP_INFO|XX

- 温度を参照した際に発行されるイベント
  - XX=温度の数値

例> MeiBeacon接続におけるFST

```
1  10  BLE_OK_CONNECT        SYNTH_START|mei|normal|接続しました
```

図 7　FST イベントの例

3.3　MeiBeacon 操作機能

3.2 節で述べた通り, MMDAgent はイベント実行型プログラムである. そのため同様に, その逆の処理も, FST スクリプト上のみで処理を完結させることは困難である.

そこで, MeiBeacon に接続された出力部品等を MMDAgent から制御可能なように, MMDAgent 内のプログラムで MeiBeacon 用の情報に変換機能を実装する. この機能は FST スクリプトから特定の出力メッセージを受け取った場合に, MMDAgent 内のプログラムで MeiBeacon 用の情報に変換するプログラムである. 特定の出力メッセージは, 本機能で扱うために作成した専用 FST コマンドであり, これらを FST スクリプトの実行メッセージに指定することで MeiBeacon の操作が可能となる. 実装した FST コマンドの例と使用例を図 8 に示す. 使用例の FST スクリプトの見方は 2.1 節に示したとおりである. FST スクリプトから特定の出力メッセージに対応した条件メッセージを音声入力の認識にすることで, MeiBeacon に取り付けている出力部品等を音声対話で操作可能である.

BLE_CP|(CONNECT or DISCONNECT)

- 最も近いBLE機器と接続/切断するコマンド

BLE_LIGHT|(DigitalPin number)|(ON or OFF)

- デジタル値出力用のコマンド
  - 各要素は省略できない

例> LEDを点灯させるFST

```
1  10  RECOG_EVENT_STOP|点灯        BLE_LIGHT|2|ON
```

- 上記例では「2」番ピンをONにする
  - 即ち「2」番ピンに取付けられたLEDが点灯する

図 8　FST コマンドの例

4.　プロトタイプシステム

4.1　プロトタイプシステムの概要と動作

プロトタイプシステムは, Android 版 MMDAgent のプラグインとして実装を行った. Android 版 MMDAgent のプラグインは, Java 言語 [4] で記述がされており, 本研究においてもそれに倣った. Android 端末には LG エレクトロニクス社製の Nexus 5 を用いた. MMDAgent については公開されているバージョン 1.5 を使用している.

また, MeiBeacon となる Arduino については,「RedBearLab nRF51822」を使用した.「環境値」のアナログセンサから取得できるアナログ値は, 直接の取得値 (0 から 1023 の 1024 段階) を送信したが, これは利用するセンサにより, それぞれ適切な変換式を適用する必要があり, それを MeiBeacon 上のプログラムに記述することは手間になるからである. また変換式を先に適応することで, 送信する値が整数以外になり, MeiBeacon の送信部分にも手間を加える必要があるため, アナログ値を扱う場合, その値の変換は MMDAgent 内で行った.

プロトタイプシステム使用時は, まず MMDAgent を起動させ, 3.1.1 節で述べた方法で MeiBeacon を探索し, コネクションを確立する. 接続に成功すると図 9 に示すように, 接続先のアドレスと共にその旨を知らせるメッセージが出現する. コネクション確立後, 4.2 節に述べるような使い方が可能になる.

4.2　プロトタイプシステムを用いた例

例えば, 次に挙げるようなことが可能になる. 全ての例において使用するためには, 予め MeiBeacon に対応するプログラムを書き込みし, センサ類を取り付けておく必要がある.

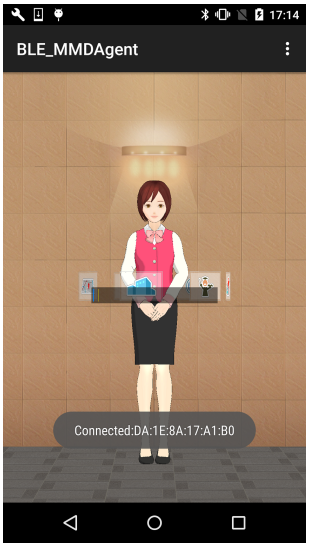

図 9　起動時に MeiBeacon に接続した様子

4.2.1　部屋の室温を尋ねる例

音声対話で現在の室温を尋ねる例である．MeiBeacon に温度センサを取り付けておき, 温度の値を「環境値」として取得する．この例の温度センサには, ”LM61BIZ”(図 10) を使用した．使用した温度センサは安価で精度がそれほど良くないため, ユーザには 1 度単位で知らせるようにした．ここでは, 3.2 節の例にある’TEMP_INFO|’ の FST イベントを使用した．

図 10　温度センサ”LM61BIZ”

4.2.2　部屋の室温を尋ねる例の構成

実際に使用するための MeiBeacon の例を図 11 に, その回路図を図 12 示す．なお都合上, 回路図では一部実際のパーツと異なる類似パーツで表現している箇所がある．その場合は, 図中に注釈を添えた．(以後の本節内の回路図についても同様である．)

また, FST スクリプトの例を図 13, 本例の構成図を図 14 に示す．図 13 では, FST スクリプト内で変数を用いている．FST スクリプトでは, ローカル変数を利用でき, 変数への値の代入は第 5 フィールドで行う．第 5 フィールドについては, 桃色で示した．代入した変数の値は第 3, 4 フィールドで利用できる．

続いて図 14 の構成図の流れを説明する． まずユーザが MMDAgent に「部屋の温度を教えて」と言うことで, 図 13 の 1 行目, ’BLE_ANALOG_READ’ の FST コマンドが実行され MeiBeacon にアナログ値の要求を行う．その後アナログ値を受信すると, それをセルシウス温度に計算し直し FST イベント’TEMP_INFO|’ の引数に温度を代入して FST に出力する．該当する FST イベントを受け取る事で, 変数 temp に温度が代入され, MMDAgent が温度についての対話を開始する．

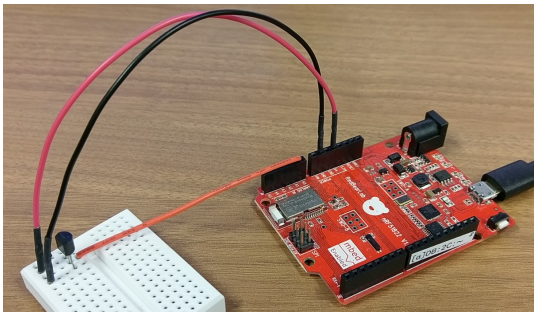

図 11　MeiBeacon に温度センサを取り付けた例

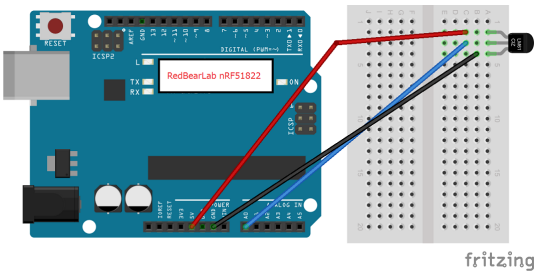

図 12　MeiBeacon に温度センサを取り付けた例の回路図

```
1  10   RECOG_EVENT_STOP|部屋,温度   BLE_ANALOG_READ|TEMP
10 11   TEMP_INFO|15                 <eps> ${temp}=15
10 11   TEMP_INFO|16                 <eps> ${temp}=16
                  ⋮
10 11   TEMP_INFO|34                 <eps> ${temp}=34
10 11   TEMP_INFO|35                 <eps> ${temp}=35
11 12   <eps>                        SYNTH_START|mei|normal|部屋の温度は${temp}度です。
12 1    SYNTH_EVENT_STOP|mei         <eps>
```

図 13　部屋の室温を尋ねる FST スクリプト例

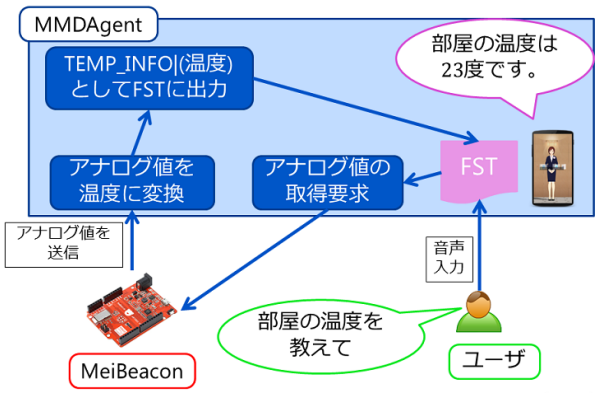

図 14　部屋の室温を尋ねる例の構成図

4.2.3　合言葉でドアの鍵を開ける例

対応した MeiBeacon との接続中に, 合言葉を使用してドアを解錠できる．ドア解錠のイメージとしては, Qrio Smart Lock [7] のようなスマートロックと考えて問題ない．ドアの鍵のサムターンに, それを回す部品を取り付ける．実際に使用した部品とその使用例を図 15 に示す．これは, MeiBeacon とサーボモータを接続し, サーボモータを制御してドアの鍵の開閉を行う．

また合言葉は音声入力を用いて行う．MMDAgent で認識されたデータを MeiBeacon に送信し, MeiBeacon 上で判定を行う．判定後, その成否の結果のみを MMDAgent に返却し, MMDAgent は受け取った FST イベントに応じて動作する．

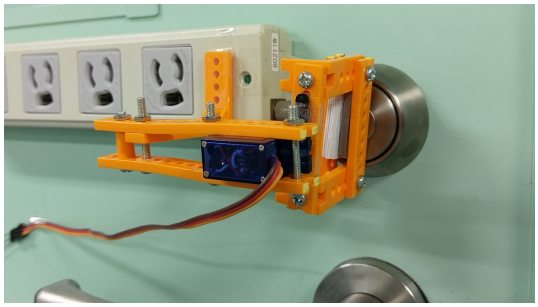

図 15　サムターンを回す部品の例

4.2.4　合言葉でドアの鍵を開ける例の構成

実際に使用するための MeiBeacon の例を図 16 に, その回路図を図 17 示す. 加えて, FST スクリプトの例を図 18, 本例の構成図を図 19 に示す. また構成図の流れを説明する.

まずユーザが MMDAgent に「ドアを開けて」と言うと, 図 18 の 1, 2 行目より, MMDAgent はユーザに合言葉を要求する. それと同時に合言葉を受け付ける時間を決めたタイマーイベントを FST スクリプトで動作させる. (図 18 では, 15 秒間.) 更にこれに連動して, この時間内の音声認識結果を合言葉として扱うように MMDAgent 内のプログラムで設定をしている. 合言葉としての扱いは, 正しい合言葉が認識されるか, 15 秒経過で終了する. 但し図 19 の構成図では, スペースの都合上ここまでの流れの表記を割愛した. その後, MMDAgent に音声入力を行うと, その認識内容を MeiBeacon に送信する. 次に, MeiBeacon 内で送信されてきた認識内容を判定し, 合言葉が正しいならば, MMDAgent に FST イベント'DOOR_LOCKCODE|' の引数を TRUE として FST に出力するよう, その情報を MeiBeacon から送信する. それと同時にサーボモータを動作させ, ドアの解錠を行う. 解錠成功 FST イベントを受け取ると, MMDAgent はその事をユーザに知らせる. なお合言葉と認識内容が異なるならば, 引数を FALSE として FST に出力し, 解錠失敗をユーザに知らせる.

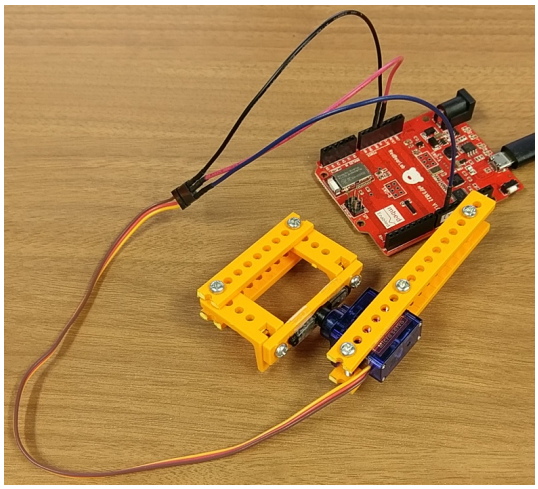

図 16　MeiBeacon にドアの開錠用部品を取り付けた例

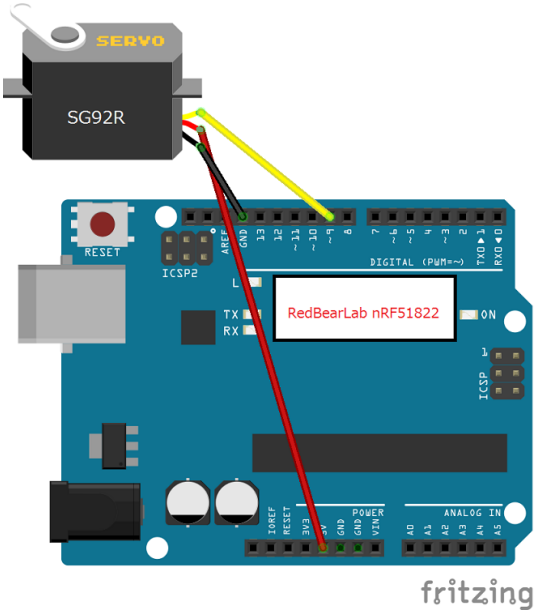

図 17　MeiBeacon にドアの開錠用部品を取り付けた例の回路図

```
1  10  RECOG_EVENT_STOP|ドア,開け   SYNTH_START|mei|normal|開けるには合言葉が必要です。
10 11  <eps>                        SYNTH_START|mei|normal|合言葉をどうぞ。
11 12  SYNTH_EVENT_STOP|mei         TIMER_START|wordwait|15
12 13  TIMER_EVENT_START|wordwait   <eps>
13 13  DOOR_LOCKCODE|FALSE         SYNTH_START|mei|sad|違います。
13 14  DOOR_LOCKCODE|TRUE          SYNTH_START|mei|happy|ドアを開きました。
13 15  TIMER_EVENT_STOP|wordwait   SYNTH_START|mei|normal|合言葉の受付を終了しました。
14 1   SYNTH_EVENT_STOP|mei        <eps>
15 1   SYNTH_EVENT_STOP|mei        <eps>
```

図 18　合言葉でドアの鍵を開ける FST スクリプト例

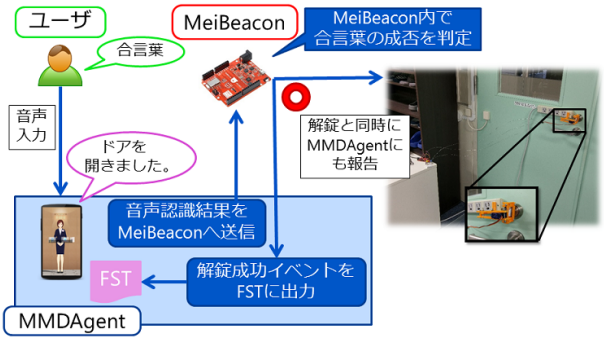

図 19　合言葉でドアの鍵を開ける例の構成図

5.　提案システムの評価

本節ではプロトタイプシステムの評価実験について述べる.

5.1　評価実験の目的

提案したプロトタイプシステムを使用し, BLE 関連機能が有用なものかどうかを調査する. また, 対話に「環境」を含んだ状態でも MMDAgent はインタラクティブな対話を行うことができるか評価する. なおここでいうインタラクティブな対話とは, MMDAgent を Android へと移植した文献 [16] を参考にし, 2 秒以内に応答があるもののことを指す. そのための実験として, 今回は提案システムを用いた例の作成と動作時間の計測を行った. 提案システムを用いた例は, 4.2 節に述べたとおりである. また動作時間の計測実験では, 次の 2 種類を計測した.

- コネクションの確立に要する時間
- コマンド制御に要する時間

それぞれの実験内容の詳細を 5.2, 5.3 節に示す.

5.2 コネクション確立実験

システムがリクエストを出してから実際にコネクションの確立を完了させるまでの時間を計測する. 実験でのシステムの流れと時間の計測を行う部分を, 図 20 に示す. 図 20 での時間の流れは上から下であり, それぞれ FST スクリプト, MMDAgent 内のプログラム, MeiBeacon 上での時間軸を示し, 各時間軸同士を繋ぐ矢印は, 始点から送信される主なデータや命令を表す. また, 「※」での接続先選択処理を含む主な処理は, 3.1.1 節で示した通りであり, 図 6 の「端末発見」は図 20 における「callback(BLEAddress)」に該当する. 更に, 図 20 の「CONNECT」は, 3.1.1 節で示したシステムの流れの（7）に当たる.

それから, 図の橙の両方向矢印はその間の時間計測を行ったことを示し, 実験での応答時間は, 図 20 の「計測時間」である.

実験は, 最大 3 台の MeiBeacon を用い, 周囲に存在する台数及び各 MeiBeacon までの距離毎に条件を分けて行った. 実際の実験風景を図 21 に示す. 便宜上 3 台の MeiBeacon はそれぞれ A 機, B 機, C 機とし, 図 21 の A, B, C はそれらに該当する. 以降, 本節ではこの呼び分けを用いる. 実験では, 1 台の時は A 機のみを, 2 台の時は A, B 機を使用し, 何れの実験でも 3 台の中心位置 (図 21 中の緑丸の位置) に立ち, 本システムを起動した. その後, 機器との BLE 接続を切断した後, 再接続の実験を行った.

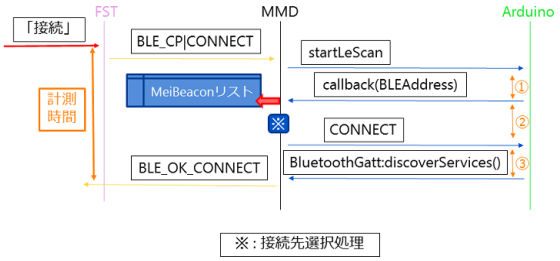

図 20　実験でのシステムの流れ (BLE 接続)

5.3 コマンド制御実験

今回は LED ライトを点灯する実験を行った. MMDAgent に「電気を点けて」と要求し, 実際に LED が点くまでの時間を計測する. 本実験でのシステムの流れと時間の計測を行う部分を, 図 22 に示す. 図 22 の見方は図 20 と同様である. ここで実験での応答時間は, 音声認識から実際に LED が点灯するまでとした. しかし, 実際に点灯するまでの時間の計測は困難であったため, MMDAgent 側で点灯リクエストを出した時間から MeiBeacon からの点灯報告を受け取った時間までの時間 (図 22 内の ⑤) を計測し, その半分の時間を使用した. なぜなら, 相互にデータをやり取りする時間は等しいと考えられるためである.

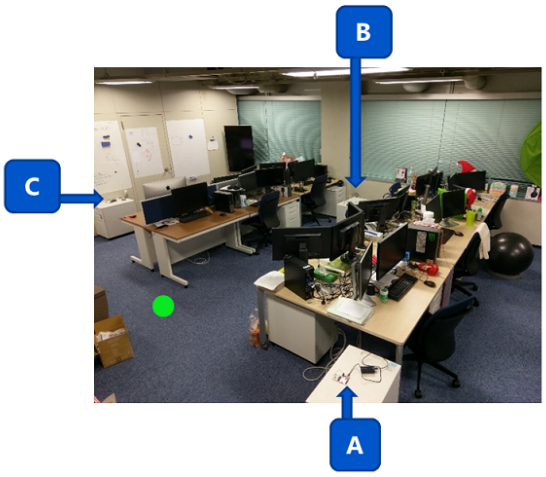

図 21　実 験 風 景

実験は 5.2 節と同じ配置で実施し, MeiBeacon との距離で条件を分けて行った. 但し, 本システムは MeiBeacon1 台との接続が前提のため, 1 台の場合のみで実験を行った. 実験では, システムを起動し BLE 接続が完了した後, 前述のように「電気を点けて」と MMDAgent に話しかけた.

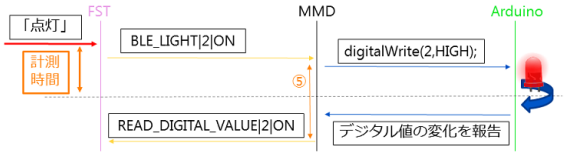

図 22　実験でのシステムの流れ (LED 点灯)

5.4 実験結果と考察

それぞれの実験における計測時間の結果を示す. 実験は, BLE 接続実験については, 各項目それぞれ 5 回ずつ行い, その結果を表 1, 2 で, 条件ごとに最速時間と最遅時間, 平均時間を示した. なお 2 台以上の距離が異なる場合は, A 機が 5m, C 機が 1m の条件で何れも実験を行った. LED 点灯実験については, 各項目それぞれ 10 回ずつ実施し, その結果を表 3 で条件ごとに最速時間と最遅時間, 平均時間を示した.

但し, 図 20 の ① から ③ と図 22 の ⑤ の計測時間については, 計測時間の参考であるためその結果の掲載については割愛した.

表 1　BLE 接続・起動時の計測結果

台数	距離	平均 [秒]	最遅 [秒]	最速 [秒]
1 台	50cm	1.75	2.11	1.32
	5m	1.76	2.05	1.61
2 台	各 3m	1.80	2.38	1.61
	3m,5m	1.91	2.37	1.74
3 台	各 3m	1.80	1.91	1.74
	1m,3m,5m	1.77	1.83	1.70

表 1 の結果から, 平均時間はどれも 2 秒を下回っており, インタラクティブな対話を行える.

表 2　BLE 接続・再接続時の計測結果

台数	距離	平均 [秒]	最遅 [秒]	最速 [秒]
1 台	50cm	2.45	2.53	2.34
	5m	2.51	2.75	2.31
2 台	各 3m	2.44	2.74	2.34
	3m,5m	2.49	2.64	2.33
3 台	各 3m	2.46	2.58	2.31
	1m,3m,5m	2.43	2.61	2.33

表 3　LED 点灯の計測結果

距離	平均 [秒]	最遅 [秒]	最速 [秒]
50cm	0.42	0.67	0.28
5m	0.42	0.55	0.36

続いて表 2 の結果では, 全ての場合において平均時間が 2 秒を超えており, 最速でも 2 秒を下回っていない. しかしながら今回の実験では, 探索時間設定が 2 秒のため, そもそも 2 秒を下回ることは不可能である. そこでこのログを確認したところ, 最初の MeiBeacon の情報を受け取るまで平均して約 0.2 秒要している事がわかった. そのため最短手順で接続しようとすると, 平均約 0.7 秒で完了すると考えられる. 但し, 探索時間を短く設定した場合, 最も近い MeiBeacon に接続する確率も低くなるため, 実験結果より 1 秒から 1.5 秒程度に設定することで, インタラクティブな対話を行えると考えられる.

一方で, 使用した MeiBeacon の台数や距離による, 計測時間の違いは共に現れなかった. だが, 複数台を使用した場合の実験における接続先は, 必ずしも最も近い MeiBeacon に接続する結果とはならなかった. 理由は実験での MeiBeacon の配置や, 使用者の向きなどで電波受信強度が低下したことであった.

最後に表 3 の結果を確認すると, どちらの条件でも平均時間は 2 秒を大きく下回る結果となった. これより BLE 接続時と同様で, 距離による計測時間の違いはほぼないものであると言える.

6. 終わりに

本論文では, スマートフォン向け音声対話システムである Android 版 MMDAgent を用いて, 利用者の周囲の環境に応じた音声対話可能なシステム及びそれを可能とするプラットフォームの開発を提案し, その実現法について述べた. また提案した実現法をもとにした, Android 端末と Meibeacon から成るプロトタイプシステムを実装し, その使用例について述べた. 今後の課題としては, 現在のシステムは 1 台と接続することを前提としているため, MeiBeacon 同士の連携を行う事ができない点が挙げられる. MeiBeacon 同士の連携を行う事で, 例えば, 部屋を移動した際に移動前の部屋の電気がついてるなら, その電気を消すように促す会話ができ, 移動前の部屋の電気を消すことが可能となる. また, 現在は接続した MeiBeacon との距離に応じたイベントが存在しない. 上記課題の解決と併せることで, ユーザのおおよその位置推定も可能となり, それに応じた対話が可能になる事が期待できる. その他として, Android 版 MMDAgent の画面内の情報がユーザビリティに欠けていると思われる点が挙げられる. 例えば, 現在のシステムでは接続されている端末は接続時にしか確認できない. また, 対話可能な内容が明示されていないため, 環境に応じた対話と言っても何を話せばよいかわからないということも課題として挙げられる. .

謝辞　本研究は JSPS 科研費 26330136, 25700009, 科学技術振興機構 CREST, および, 総務省 SCOPE の助成を受けたものです.

文　献

[1] Apple Inc., Siri, http://www.apple.com/ios/siri/(2017.1.12 参照)
[2] Arduino, https://www.arduino.cc(2017.1.15 参照)
[3] 株式会社エンプライズ, ”自分のスマホが試聴機に！” HMV & BOOKS TOKYO に 150 個のビーコンを設置！ HMV アプリをリリース！, http://www.emprize.jp/news/1151/(2017.1.12 参照)
[4] Java, http://www.oracle.com/(2017.1.15 参照)
[5] 国立大学法人名古屋工業大学情報基盤センター, 授業打刻アプリ (Nitech ピロリン) 提供サービス, https://www.cc.nitech.ac.jp/service/common/pyrroline.html(2017.1.12 参照)
[6] Nordic Semiconductor, nRF51822, https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF51822(2017.1.15 参照)
[7] Qrio 株式会社, Qrio Smart Lock, http://qrio.me/smartlock/(2017.1.15 参照)
[8] RedBear, RedBearLab nRF51822, http://redbearlab.com/redbearlab-nrf51822/(2017.1.15 参照)
[9] 株式会社ニコン, SnapBridge アプリ, http://www.nikon-image.com/products/software/lineup/snapbridge/(2017.1.12 参照)
[10] Akinobu Lee, Keiichiro Oura, Keiichi Tokuda, MMDAgent - A fully open-source toolkit for voice interaction systems, Proceedings of the ICASSP 2013, pp. 8382-8385, 2013.5
[11] Allauzen C., Riley M., Schalkwyk J., Skut, W., Mohri M., OpenFst: A general and effi- cient weighted nite-state transducer library, Implementation and Application of Automata Springer Berlin Heidelberg, pp. 11-23, 2007
[12] Barragan, H., *Wiring: Prototyping Physical Interaction Design*, Interaction Design Institute Ivrea, Italy, 2004.
[13] Daisuke Yamamoto, Keiichiro Oura, Ryota Nishimura, Takahiro Uchiya, Akinobu Lee, Ichi Takumi and Keiichi Tokuda, Voice Interaction System with 3D-CG Human Agent for Stand-alone Smartphones, Proc. of the 2nd International Conference on Human Agent Interaction, ACM digital library, pp.320-330, 2014
[14] Lee, A. and Kawahara, T., *Recent Development of Open-SourceSpeech Recognition Engine Julius*, APSIPA, pp. 131-137, 2009.
[15] 田中 亮佑, 堤 修平, 山本 大介, 高橋 直久, 領域グラフと利用者の位置に基づく音声対話シナリオ更新手法, マルチメディア, 分散協調とモバイルシンポジウム 2016 論文集 2016, pp.969-976, 2016.
[16] 山本 大介, 大浦 圭一郎, 西村 良太, 打矢 隆弘, 内匠 逸, 李 晃伸, 徳田 恵一, スマートフォン単体で動作する音声対話 3D エージェント「スマートメイちゃん」の開発, インタラクション 2013, IPSJ Symposium Series Vol. 2013, No. 1, pp. 675-680