社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

ホームネットワークシステムにおけるエージェントを用いた対話型インタフェースフレームワークの開発

堀内　大祥†　　佐伯　幸郎†　　松本　真佑†　　中村　匡秀†

† 神戸大学　〒 657–8501 神戸市灘区六甲台町 1–1
E-mail: †horihori@ws.cs.kobe-u.ac.jp, ††sachio@carp.kobe-u.ac.jp, †††{shinsuke,masa-n}@cs.kobe-u.ac.jp

あらまし　我々の研究室では，バーチャルエージェントを HNS の操作インターフェイスに用いる HNS-バーチャルエージェントユーザインタフェース（HNS-VAUI）として，MMDAgent を用いたプロトタイプを開発している．しかしながら，プロトタイプは MMDAgent の仕様に強く依存しているため，ユーザとの高度なインタラクションが実現できない．そこで，先行研究において，高度な HNS 操作のためのサービス指向 MMDAgent 制御フレームワーク（SO-MACHO）を提案した．SO-MACHO では，HNS-VAUI システムの制御をキャラクタの動作などの制御機能と HNS 操作に必要な外部 Web サービス実行機能に分離し，これらを Web サービス化することにより，外部からの入力に応じたステートフルなインタラクションを実現できる．本稿では，SO-MACHO に基づいた Web サービスの実装を行う．ケーススタディとして，実際の HNS 環境に SO-MACHO を適用した HNS-VAUI を実装し，SO-MACHO の有用性を考察する．
キーワード　ホームネットワークシステム，対話型ユーザインタフェース，バーチャルエージェント

Implementing Service Framework for agent-based interactive interface in Home Network System

Hiroyasu HORIUCHI†, Sachio SAIKI†, Shinsuke MATSUMOTO†, and Masahide NAKAMURA†

† Kobe University　Rokko-dai-cho 1–1，Nada-ku，Kobe，Hyogo，657–8501 Japan
E-mail: †horihori@ws.cs.kobe-u.ac.jp, ††sachio@carp.kobe-u.ac.jp, †††{shinsuke,masa-n}@cs.kobe-u.ac.jp

Abstract　We have developed the prototype of interactive voice user interface with the virtual agents as a new HNS user interface (HNS-VAUI) using MMDAgent to virtual agent. However, the prototype is strongly dependent on the implementation specifications of MMDAgent, and it is not possible to perform advanced interactions with the user. Therefore, in previous studies, we proposed the framework of service-oriented MMDAgent control for advanced HNS operation (SO-MACHO). In SO-MACHO, there are two Web services that controls the operation of the character, and that controls the invocation of the external Web service. Those two Web services control HNS-VAUI system and realize the interactive user interface. In this paper, we implement the each Web Service based on SO-MACHO. Moreover, as a case study, we implemented the HNS-VAUI with SO-MACHO in our HNS to evaluate the usability as a interface.
Key words　home network system, interactive voice interface, virtual agent

1. はじめに

近年，ネットワーク技術の発展に伴い，宅内に配置された家電やセンサをネットワークに接続し付加価値の高いサービスを実現する次世代のユビキタスアプリケーションの枠組みとして，**ホームネットワークシステム**（HNS）[1] が盛んに研究・開発されている．HNS における重要な課題として，ユーザインタフェース（UI）のユーザビリティの向上があげられる．HNS では日常的に使うという特性上，ユーザにとって学習コストが低く操作しやすい，かつ簡単で便利であるユーザフレンドリな UI がより強く求められている．

HNS に適したこのような UI として，**バーチャルエージェント**（VA）を UI に用いた VAUI がある．VA とは，ユーザとシステムとの連携を，擬人化されたキャラクタとの対話を通じて

行うインタフェースである．VAUI では，ユーザからの入力に対して音声だけではなくキャラクタの動作等の映像をユーザへのフィードバックに用いることで，ユーザはより自然な対話を通じてシステムを操作可能になる．

我々の研究室では，HNS の UI として VAUI を適用した，**HNS-バーチャルエージェントユーザインタフェース**（以下，HNS-VAUI）を提案し，**CS27-HNS** [2] において，**MMDAgent** [3] を利用したプロトタイプ（CS27-HNS-VAUI）を運用している [4]．しかしながら，プロトタイプでは，VA の制御は MMDAgent 独自の実装仕様に強く依存しており，HNS-VAUI の制御とキャラクタの動作の制御・インタラクションの制御が分離できないことや，実行できる機能が少ないという問題点があった．

そこで先行研究において，キャラクタの動作，及びインタラクションの制御が，他のサービスと連携可能な新たなフレームワーク（**SO-MACHO**）[5] を提案した．SO-MACHO では，「ソフトウェアの機能一つ一つをサービスとし，それらのサービスを組み合わせることで新たなサービスを開発する」というサービス指向アーキテクチャの考えに基づき，キャラクタの制御機能，及びインタラクションの制御機能をそれぞれ別のサービスとして構成する．VA の制御を，キャラクタの制御機能と外部 Web サービスの実行の制御機能を，それぞれの機能を持つ Web サービスに分離することで，個別の制御を可能にする．SO-MACHO を適用することでより高度な HNS-VAUI を実現できるようになる．

本稿では，SO-MACHO を実装し，VA の制御及び管理を行い，より高度な HNS-VAUI を実現する．また，SO-MACHO の可用性を検討するため，ケーススタディとして，SO-MACHO を実際の HNS に適用し，HNS における UI としての操作性の考察を行う．

2. 準　　備

2.1 ホームネットワークシステム（HNS）

ホームネットワークシステムは，照明やテレビなど家庭内における様々な家電機器と，温度計や湿度計などセンサをネットワークに接続することで構築される，家庭内でユーザにサービスを提供するためのプラットフォームである．HNS では，家電機器にユーザや外部エージェントがネットワーク越しに操作できるように制御 API を備えることで，音声による機器操作インタフェースや，外部環境やユーザの状況といったコンテキストに応じた機器の自律制御などの付加価値サービスを提供可能である．これらの特徴から，HNS はユーザの快適な生活を実現する次世代のスマートホーム技術として期待されている．我々の研究室で運用している CS27-HNS では，サービス指向アーキテクチャの考えを取り入れ，すべての制御 API を，機種や実行環境に依存しない標準的な Web サービスとして公開している．例えば，テレビのチャンネルを 6ch にするには，http://. . . /HNS/TVService/setChannel?channel=6 といった URL にアクセスすればよい．

2.2 バーチャルエージェント（VA）

VA は，ユーザと画面内のキャラクタとのインタラクションを通じてシステムの操作を行う UI である．親近感や対話の現実感から，無機質で情報量の少ない従来の音声インタフェースに替わる，次世代の UI として注目を集めている [6] [7]．VA を実現するアプリケーションとして，バーチャルエージェント音声対話システム用ツールキット MMDAgent は有力なソフトウェアである．MMDAgent では，音声認識，音声合成，3D 描画，リップシンク等の技術を組み合わせ，独自の MMDAgent オートマトンを用いたシナリオを作成することで，画面内のキャラクタとの音声対話システムを構築することができる．

2.3 HNS-VAUI

HNS-VAUI は，HNS の UI として VAUI を適用したものである．HNS における機器の操作に VAUI を用いることで，ユーザは機器に向かって発話するといった不自然なインタラクションではなく，人と会話しているような自然なインタラクションを通じて HNS 機器を操作することができる．また，機器の種類や操作が多数ありシステムからユーザへの情報量が増える場合において，音声フィードバックだけではなく操作可能な機器や操作内容を画面に表示することで，フィードバックの長大化を防ぎ，ユーザに優しいフィードバックを行うことが可能である．

2.4 MMDAgent を用いた HNS-VAUI（プロトタイプ）

我々は，CS27-HNS において，MMDAgent を用いたプロトタイプ（CS27-HNS-VAUI）を実現している．CS27-HNS に適用させるため，MMDAgent を以下のように改良した．

2.4.1 外部からの入力

オリジナルの MMDAgent は音声入力を処理するプロセスを内部に持っている．そこで，外部からのコマンドを MMDAgent に入力するために，標準の入力処理プロセスを MMDAgent から分離し，替わりに Web API を用いて MMDAgent オートマトンに直接コマンドを入力する Web サービス **Virtual Agent Service** を実装した．また，認識した音声を解析しコマンドを生成する機能を持つ，音声入力システムを作成した．

2.4.2 家電操作の実行

オリジナルの MMDAgent には外部の Web サービスを呼び出す機能がない．そこで，CS27-HNS における家電制御 API を実行するため，外部 Web サービスを用いて家電操作を実行する機能を新たに MMDAgent に追加した．キャラクタの動作と Web サービスの実行の両方を MMDAgent オートマトンに規定することで，HNS-VAUI を実現している．

2.5 プロトタイプの問題点

プロトタイプは MMDAgent のオートマトンの仕様に強く依存している．そのため，以下の P1,P2 のような問題点があり，実行できるキャラクタの動作やインタラクションは制限されている．

P1: 機能同士の密結合

プロトタイプでは，MMDAgent オートマトンがシステムの制御をすべて決定しており，MMDAgent オートマトンとキャラクタの制御・Web サービスの実行制御が密結合している．その

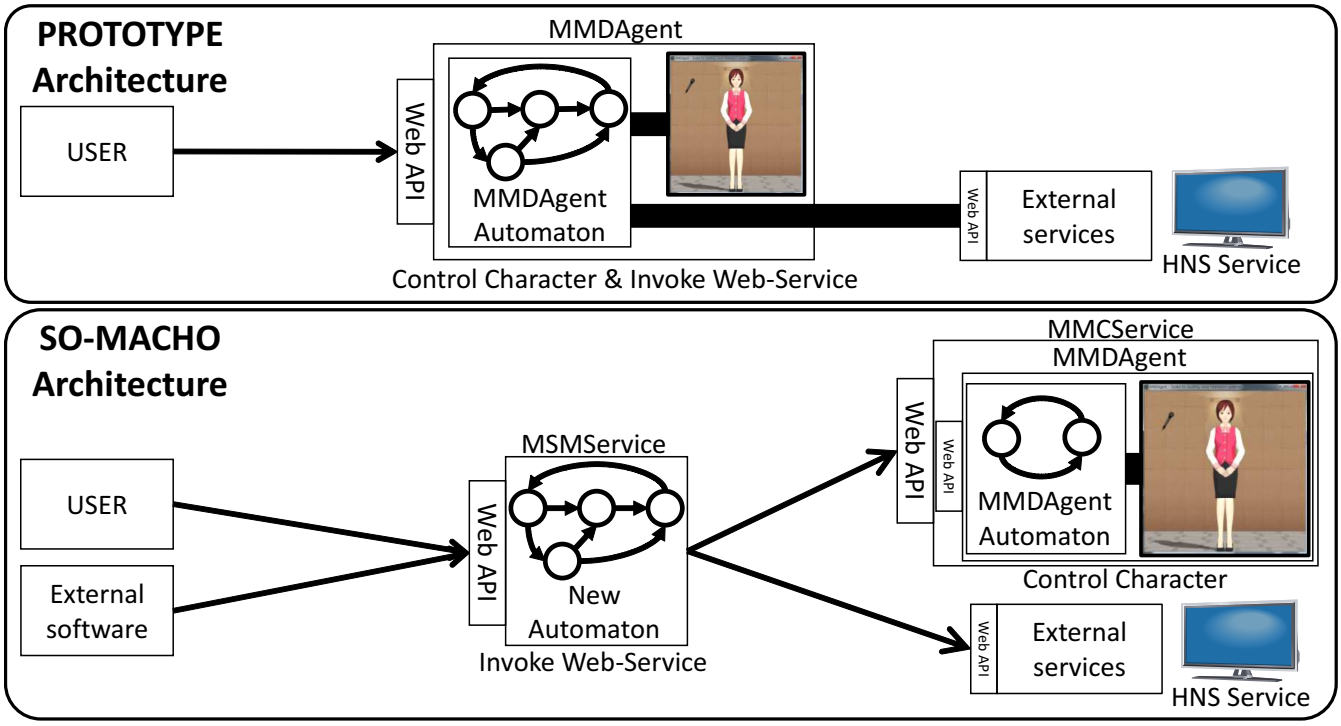

図 1　プロトタイプと SO-MACHO のアーキテクチャ

ため，キャラクタに任意のタイミングで任意の動作を実行させることや，外部から動的に遷移の追加や変更を行うことができない．

P2: MMDAgent オートマトンの記述能力の低さ

プロトタイプにおいて，システムは MMDAgent オートマトンに記述されている動作を実行する．しかし，MMDAgent オートマトンで記述できる動作は簡単な動作に限られている．そのため，複雑な処理を要するシステムの動作を実行することができない．例えば，MMDAgent オートマトンでは Web サービスの呼び出ししかできず，センササービス等から取得できる情報を用いたフィードバックができない．

3. 高度な HNS 操作のためのサービス指向 MMDAgent 制御フレームワーク

本節では, [5] で提案している，高度な HNS 操作のためのサービス指向な MMDAgent 制御フレームワーク（SO-MACHO）について述べる．SO-MACHO では，プロトタイプの問題点を改善し，種類の豊富なキャラクタの動作や高度なインタラクションが実行可能なユーザビリティの高い HNS-VAUI を実現できる．

3.1 SO-MACHO の概要

図 1 にプロトタイプと SO-MACHO のアーキテクチャを示す．プロトタイプでは，MMDAgent が独自の記法で記述された MMDAgent オートマトンを用い，キャラクタの動作及び外部 Web サービスの実行を制御しているため，**2.5** であげたような問題点がある．そこで，SO-MACHO では，サービス指向アーキテクチャに基づき，キャラクタの動作の制御と外部 Web サービスの実行の制御をそれぞれ別の機能として捉え，キャラクタの動作の制御を行う **MMCService**（Miku Miku Command Service）と，外部 Web サービスの実行の制御を行う **MSMService**（Miku State Machine Service）をあらたに作成している．MMCService はキャラクタの動作（発話・モーション）の実行を管理する Web サービスである．他のアプリケーションは API を通じ MMCService を呼び出すことで，キャラクタに動作をさせることができる．MSMService は，入力に応じて，外部 Web サービスの実行をステートフルに制御する Web サービスである．MSMService から，ユーザからの入力に対して MMCService を用いたキャラクタの動作や HNS Service を用いた家電操作を実行することでインタラクションを構成し，HNS-VAUI を実現する．また，外部アプリケーションから MSMService を呼び出すことで，ユーザからの入力以外をトリガとしたインタラクションも実行することができる．

以下に，この 2 つの Web サービスの機能について述べる．

3.2 MMCService

MMCService は，MMDAgent におけるキャラクタの発話・モーション機能をサービスとして捉え，プロトタイプで実装した Virtual Agent Service をラップした Web サービスである．キャラクタの発話とモーションを Web API として外部から制御することが可能な Web サービスである．MMCService は内部で Virtual Agent Service を呼び出し，MMDAgent と通信することで，画面内のキャラクタの発話とモーションの制御を行う．MMCService は，内部でキャラクタの発話やモーションの状態を管理しているため，外部サービスは MMCService 内部の状態を考慮する必要はなく，任意のタイミングで各 API を実行することでキャラクタに動作させることができる．

3.3 MSMService

MSMService はオートマトンを持っており，状態に応じたステートフルなフィードバックを行い，インタラクションを構成

する Web サービスである．入力に対してステートフルにフィードバックを行うことで，ユーザとの会話とシステムの制御を同時に管理し，ユーザとシステム間のインタラクティブなインタフェースを実現する．

MSMService の Web API を用いて，外部から MSMService のオートマトンへの入力を行うことができる．MSMService は入力を受けると遷移を行い，同時にその遷移に登録されているシステムの動作（MMCService を用いたキャラクタの制御や HNS Service を用いた家電操作）を実行する．

4. SO-MACHO の実装

本節では，SO-MACHO の仕様に従った MMCService と MSMService の実装について説明する．実装環境は以下の通りである．

・開発言語：Java 1.7.0_45
・音声認識エンジン：Julius 4.2.3
・サーブレットコンテナ：Apache Tomcat 7.0.39
・Web サービスフレームワーク：Apache Axis2 1.6.2

4.1 MMCService の実装

MMCService は，主なメソッドとして “say メソッド” と “doMotion メソッド” を持ち，それぞれ外部から発話 API，モーション API として実行することができる．また，キャラクタの状態を内部で管理するため，発話状態（sayingState）とモーション状態（doingState）を持ち，MMDAgent と常に同期している．

図 2 に発話 API 実行時における MMCService の状態遷移を示す．ただし，図中の “?” は条件（入力）を，“！” は動作（出力）を表し，VAS は Virtual Agent Service の略である．キャラクタに発話させたい文字列を引数として指定した発話 API を実行すると，MMCService は引数をもとに MMDAgent に送るコマンドを生成する．次に，Virtual Agent Service を用いてそのコマンドを MMDAgent に送ると同時に sayingState の状態を “saying” にする．MMDAgent は受けとったコマンドをもとに音声合成を実行し，キャラクタが発話を開始する．キャラクタの発話が終了すると，MMDAgent が有するキャラクタの発話検知できる機能を利用し，MMCService に発話終了（“SAY_OVER”）を通知する．MMCService は発話終了を受けとると，sayingState を “stand by” に戻す．“saying” 状態時に他の発話 API が実行された場合には，MMCService は Virtual Agent Service を通じ MMDAgent に発話停止コマンドを送ると同時に発話すべき内容をキューに保存し，一時的に sayingState を “waiting” 状態にする．MMDAgent は発話停止コマンドを受けると発話を停止し，MMCService に発話終了を通知する．MMCService は，発話終了の通知を受けると，キューに保存されている内容を Virtual Agent Service を通じ発話させ，sayingState を “waiting” から “saying” にする．また，モーション API についても同様である．

4.2 MSMService の実装

MSMService は input メソッドを持ち，外部からオートマトンへの条件の入力 API として実行できる．また，オートマトンの現在の状態を State という変数に格納している．オートマトンは遷移 ID，遷移前状態，遷移後状態，遷移条件，遷移アクションの 5 つの要素を持つ遷移が多数集まって構成されている．遷移前状態・遷移後状態はそれぞれオートマトンの状態を表す値であり，遷移条件は外部から入力を受けた際に遷移を行う条件となる文字列である．遷移アクションには，遷移する際に外部 Web サービスや，MMCService により実行するシステムの動作が登録されている．表 1 に遷移の例を, また図 3 にその状態遷移図を示す.

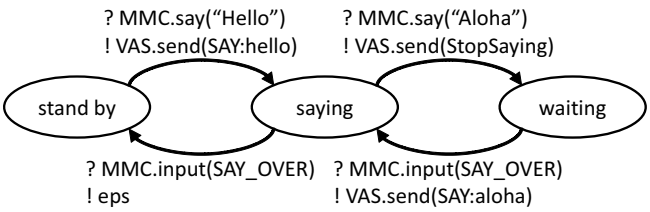

図 2　sayingState の状態遷移

入力する条件となる文字列を引数にとり入力 API を実行すると，MSMService はまず現在のオートマトンの状態（State の値）を確認し，登録されている遷移から，遷移前状態が現在の状態と一致し，さらに，遷移条件が入力された文字列と一致する遷移を 1 つ選択する．次に，MSMService は，State の値を選択された遷移の遷移後状態に更新し，同時にその遷移の遷移アクションを実行する．MSMService は，MMCService や HNS Service 等の外部 Web サービスにより遷移アクションに登録されているキャラクタを用いたフィードバックや HNS 操作を実行する．

5. ケーススタディ

ケーススタディとして，SO-MACHO を適用した HNS-VAUI による実際の操作・実行例をあげる．遷移に関しては，表 1, 図 3 の遷移を用いる．ただし，以降ではオートマトンの状態を *state* のように，オートマトンの遷移条件を “CONDITION” のように表記する．

MSMService のオートマトンは基本状態 *idle* を持ち，ユーザや外部サービスからの入力により，他の状態に遷移する．例えば，*idle* 状態時にユーザからの呼びかけられると *active* 状態に遷移する．*active* は主にユーザからの命令を待っている状態であり，ユーザはキャラクタに命令し，家電操作等を実行させることができる．

5.1 TV をつける

MSMService の State の値は，現在 *active* とする．ユーザが “TV をつけて” と発話すると，それを音声認識プログラムが解釈し，MSMService の入力 API を条件となる “TV_ON” を引数にとり，以下のように実行する．

http://. . . /MSMService/input?str=TV_ON

この Web サービスを実行することで，MSMService に “TV_ON” が入力される．MSMService は，その入力を受け，現在状態からの遷移のうち “TV_ON” を条件に持つ遷移を選択し，現在の状態をその遷移の遷移後状態に変更する．(このケー

表 1　MSMService が持つ遷移の例

No.	Transition Name	Current State	Next State	Condition	Action
1	Activate	idle	active	ACTIVATE	MMCService.say(May I help you?);
2	Disactivate	active	idle	DISACTIVATE	MMCService.say(Bye.); MMCService.doMotion(bye);
3	Tuning on TV	active	active	TV_ON	MMCService.say(I turn on the TV.); TVService.on();
4	Get today weather	active	active	TODAY_WEATHER	[weather] = WeatherService.get(today); MMCService.say(It is [weather], today.);
5	Recommend fan	idle	Recommend fan	RECOMMEND_FAN	MMCService.say(Shall I turn on the fan?);
6	Recommend fan yes	Recommend fan	idle	YES	MMCService.say(I turn on the fan.); FanService.on();
7	Recommend fan no	Recommend fan	idle	NO	MMCService.say(Okey.);
8	Recommend fan over	Recommend fan	idle	TIMEOUT	

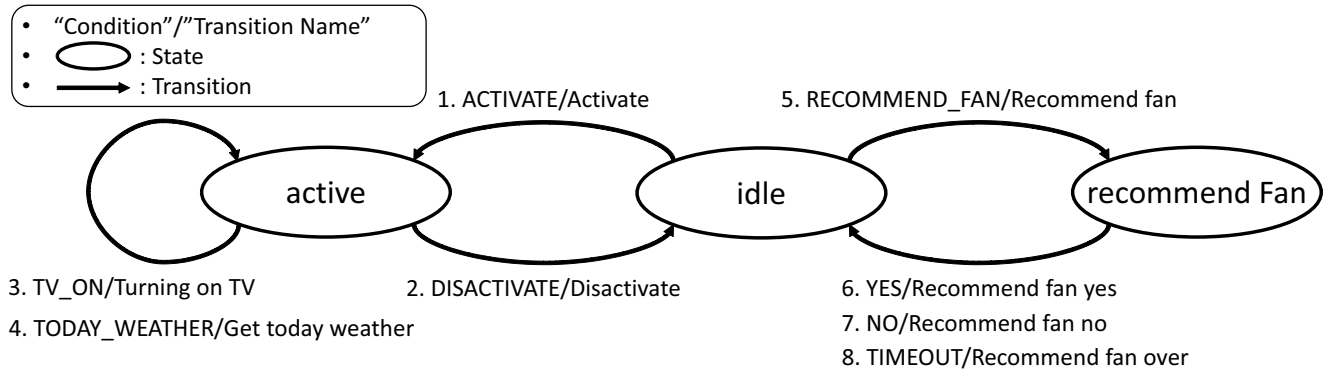

図 3　表 1 の状態遷移図

スでは，*active* から変化しない.）遷移と同時に，登録されている遷移アクションを実行する．この遷移アクションでは，まず，HNS Service を用いて TV が起動しているかどうかを確認するため，以下のような TV の状態を取得する Web サービスを実行する．

http://. . . /HNS/TVService/getStatus

次に，得られた情報より TV の状態が OFF であった場合，更に以下の 2 つの Web サービスを実行する．

http://. . . /HNS/TVService/on

http://. . . /MMCService/say?str=TV をつけます

以上の動作で，HNS の TV Service によりテレビが起動され，MMCService の発話 API によりキャラクタが “TV をつけます” と発話する．

5.2　今日の天気を聞く

MSMService の State の値は，現在 *active* であるとする．ユーザが “今日の天気は?” と発話するとそれを音声認識プログラムが解釈し，MSMService の入力 API を “TODAY_WEATHER” を引数にとり実行する．

http://. . . /MSMService/input?str=TODAY_WEATHER

MSMService は，入力を受け，現在の状態からの遷移のうち “TODAY_WEATHER” を条件に持つ遷移を選択，現在の状態をその遷移の遷移後状態に変更し，遷移アクションを実行する．（このケースでは，*active* から変化しない.）この遷移アクションでは，まず，外部 Web サービスである Weather Service を用いて，今日の天気の情報を取得する．

http://. . . /weatherService/getTodayWeather

次に，取得した情報より，今日の天気が “晴れ” であった場合，MMCService の発話 API を以下のように実行する．

http://. . . /MMCService/say?str=今日の天気は晴れです

Web サービスが実行されると，キャラクタが “今日の天気は晴れです” と発話し，ユーザの質問に返答する．

5.3　暑くなると，扇風機を推薦する

我々の研究室では，CS27-HNS のサービスの一つとして**コンテキストアウェアサービス（CAS）**を利用することを提案している．CAS とは，様々なシステムやセンサなどにより得られる情報からコンテキストの変化を認識し，その変化に対応した機能や情報をユーザに提供するサービスのことである．

ここで,「部屋の温度センサの値が 28 ℃以上」という情報から「暑い」というコンテキストが推定された場合に，扇風機の使用を推薦するケースを考える．また，MSMService の State の値は，現在 *idle* であるとする．室温が 28 ℃以上になると，コンテキストアウェアサービスが MSMService の入力 API を以下のように用いて “RECOMMEND_FAN” を入力する．

http://. . . /MSMService/input?str=RECOMMEND_FAN

MSMService は，現在の状態 *idle* からの遷移のうち “TODAY_WEATHER” を条件に持つ遷移を選択，現在の状態を選択された遷移の遷移後状態 *recommend fan* に変更し，遷移アクションを実行する．この遷移アクションでは，MMCService の発話 API を以下のように実行する．

http://. . . /MMCService/say?str=扇風機をつけませんか

発話 API を実行することで，キャラクタが発話し，ユーザに扇風機の使用を推薦する．

キャラクタの推薦に対し，ユーザが “はい” と回答すると，音声認識プログラムが解釈し，以下のように MSMService の入力 API を実行する．

http://. . . /MSMService/input?str=YES

MSMService は，入力を受け，現在の状態からの遷移のうち “YES” を条件に持つ遷移を選択，現在の状態を選択された遷移の遷移後状態 *idle* に変更し，遷移アクションを実行する．この遷移アクションでは，以下の 2 つの Web サービスを実行する．

http://. . . /HNS/FanService/on

http://. . . /MMCService/say?str=扇風機をつけます

HNS Service により，扇風機が起動し，MMCService の発話 API によりキャラクタが発話する．

5.4 考　　察

SO-MACHO に基づいた HNS-VAUI システムを実装し，実際に我々の HNS に適用した結果，SO-MACHO を適用することでプロトタイプでは実行できない高度なインタラクションが可能な HNS-VAUI を実現できることが確認できた．しかし，遷移が家電の機器や操作毎に必要となり，規模の増加に伴い，遷移の管理や編集が非常に複雑になるという問題が明らかになった．この問題に対し，遷移のパターン化や，遷移を管理するサービスを開発することでこの問題を解決が可能である．

複数のキャラクタを，サービス毎に担当するキャラクタを決め同時に利用したい，または，ユーザの気分などにより使い分けしたいといった要求が考えられるが，複数のキャラクタを同時に制御するためには，それぞれのキャラクタに対応したオートマトンを管理する必要がある．現在の SO-MACHO では，複数のオートマトンを制御することができないが，キャラクタ毎のオートマトンの管理や複数オートマトン間での状態の同期などの拡張を行うことで，システム的な対応が可能である．しかし，複数オートマトンを制御可能に拡張した場合であっても，ユーザからの入力がどのキャラクタのオートマトンに対して行われたのかを一意に定めなければならないことや，オートマトンの遷移アクションによる HNS 操作がサービス競合を起こす可能性があるといった問題点が更に発生する．前者は，対象とするキャラクタの名前を呼びかけるなどの方法で，VAUI としてのユーザビリティを損なうことなく対応が可能である．後者のサービス競合の問題については, [8] で示されている改善方法を適用することで，解決が可能である．

HNS では，各ユーザにより対象となる機器や期待するインタラクションが異なるため，HNS-VAUI は各ユーザの要求にあった個人適応型であることが望ましい．ユーザが実際に必要としているサービスやインタラクションを検討し，それらを HNS-VAUI において実現可能にすることや，ユーザ自身が自分にあったインタラクションを簡単に HNS-VAUI に追加できるようなサービスや GUI を提供することが重要である．他にも，外出先等から携帯端末に表示されるキャラクタとのインタラクションを通して，家の現在の情報を取得するといったことが可能なモバイル型 HNS-VAUI の実現や，ユーザの操作や行動を学習しシステムが自動でユーザを満足させるようなインタラクションを実行可能にするサービスの開発が HNS-VAUI を広く普及されるためには必要と考える．

6. おわりに

SO-MACHO を適用することで，プロトタイプでは不可能であった高度なインタラクションを行うことができる HNS-VAUI を実装した．また，ケーススタディを通して，CS27-HNS において，SO-MACHO を適用した HNS-VAUI が，家電操作や情報取得，推薦サービスを実行できることを確認した．今後の課題として，SO-MACHO を適用した HNS-VAUI の実験を行い，この HNS-VAUI のユーザビリティを評価することがあげられる．また，エンドユーザ視点から HNS-VAUI システムの考察を行い，個人適応型の HNS-VAUI を実現することもあげられる．

謝辞　この研究の一部は，科学技術研究費（基盤研究 C 24500079, 基盤研究 B 23300009），及び，積水ハウスの研究助成を受けて行われている．

文　　献

[1] H. Igaki, M. Nakamura, and K. Matsumoto, “Design and evaluation of the home network systems using the service oriented architecture,” Proc. International Conference on E-Business and Telecommunication Networks(ICETE04), vol.1, pp.62–69, Aug. 2004.

[2] M. Nakamura, A. Tanaka, H. Igaki, H. Tamada, and K. Matsumoto, “Constructing home network systems and integrated services using legacy home appliances and web services,” International Journal of Web Services Research, vol.5, no.1, pp.82–98, 2008.

[3] MMDAgent Project Team, “Mmdagent - toolkit for building voice interaction systems,” Nagoya Institute of Technology. `http://www.mmdagent.jp`.

[4] S. Soda, M. Nakamura, S. Matsumoto, S. Izumi, H. Kawaguchi, and M. Yoshimoto, “Implementing virtual agent as an interface for smart home voice control,” In Asia-Pacific Software Engineering Conference (APSEC2012), pp.342–345, Dec. 2012.

[5] 堀内大祥，佐伯幸郎，まつ本真佑，中村匡秀，“ホームネットワークシステムにおけるバーチャルエージェントのためのサービスフレームワーク,” 電子情報通信学会技術報告，第 113 巻電子情報通信学会，pp.83–88 Nov. 2013.

[6] M. Ochs, C. Pelachaud, and D. Sadek, “An empathic virtual dialog agent to improve human-machine interaction,” Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1, pp.89–96, 2008.

[7] J. Cassell, “Embodied conversational interface agents,” Communications of the ACM, vol.43, no.4, pp.70–78, 2000.

[8] 池上弘祐，吉村悠平，井垣宏，中村匡秀，“サービス期間を考慮したホームネットワークサービス競合検出・解消システムの実装,” 電子情報通信学会，vol.108，no.462，pp.007–012，March 2009.