2C2-04

擬人化エージェントGalateaのためのVoiceXML処理系

The VoiceXML Interpreter for the Anthropomorphic Agent Software "Galatea"

西本 卓也　　嵯峨山 茂樹
Takuya Nishimoto　Shigeki Sagayama

東京大学大学院
The University of Tokyo

This paper reports the development of the VoiceXML interpreter Galatea DM, which controls the anthropomorphic spoken dialog agent (ASDA). Although various functions are required in research and development of the dialog system using VoiceXML, our ASDA system can customize it's configuration and add new modules easily. The dialog manager can also manage such functions as an extension of VoiceXML interpreter, including face animation control, multi-modal input and output, logging of the dialogs, and assisting the human-machine dialogs in WOZ (Wizard of Oz) experiments.

1. はじめに

近年，音声対話システムの研究や開発が盛んに行なわれている．特にユーザの自由な発話を許容する対話システムにおいては，誤認識を想定した確認対話など，効率的な対話制御を含むさまざまな考慮が求められる [1] ．

我々は，GUI (Graphical User Interaface) に限定された従来の HMI (Human Machine Interface) の限界を超える手段として，擬人化音声対話エージェントの利用を目指しており，特に擬人化エージェントによる対話実験を効率的に行うために，VoiceXML *1 による対話記述を検討している．本報告では，擬人化エージェントのツールキットである Galatea [2] の基本設計を用いて，入出力モダリティの追加や対話モニタ機能の実現など，さまざまなモジュールの追加について検討する．また，VoiceXML 処理系 Galatea DM (Dialog Manager) の実装と，Galatea DM におけるモジュール制御の詳細についても合わせて報告する．

紹介される各ツールはオープンソースソフトウェアとしての公開を予定している*2 ．

2. 音声対話の研究ツール

2.1 VoiceXML によるシステム開発

電話応答システムを主な対象とした対話パターン記述言語 VoiceXML が W3C (World Wide Web Consortium) によって標準化されており，いくつかの VoiceXML 処理系が製品化されている．

VoiceXML は音声認識や音声合成などに関しては成熟した技術のみを使用しているが，Web 開発のアーキテクチャを採用しており，特にブートストラップ段階でのアプリケーション開発が容易になる．簡単なシステム主導対話においてはタスク記述の可読性が考慮されており，メニューおよびフォーム入力を対話の単位とする状態遷移モデルに基づいた構造となっている．一方で，対話の流れをカスタマイズしたい場合には，変数や条件分岐など，手続き型プログラミングの機能を埋め込むことができる．

連絡先: 西本卓也，東京大学大学院　情報理工学系研究科，〒 113-8656 東京都文京区本郷 7-3-1，電話/Fax:03-5841-6902，nishi@hil.t.u-tokyo.ac.jp

∗1 http://www.w3.org/Voice/

∗2 http://hil.t.u-tokyo.ac.jp/∼galatea/ に情報を掲載予定.

VoiceXML ファイルを動的に生成することで，タスクに依存したデータベース操作や言語処理など，VoiceXML が備えていない機能を補うことも可能である．例えば，ドメイン知識からの対話パターンの自動生成 [4]，ユーザに対する協調的な応答の動的生成 [5]，汎用的な対話システムのフレームワークの実現 [6]，対話中のタスク間遷移 [7] などの試みがなされている．また，対話設計の教育に関する事例 [8] では，VoiceXML の利用によって対話設計の効率的な評価が可能となった反面，ログ機能などの充実が必要である，と報告されている．

2.2 VoiceXML を用いた対話実験

音声対話システムの反復的な開発手法として，荒木らによる提案 [9] では，商用の VoiceXML 処理系を用いた WOZ (Wizard of Oz) 実験における対話コーパス収集とシステム改良の効率化を試みている．

荒木らの手法ではユーザと対話システムの対話を実験者が監視しており，ユーザが文法外の発話を行った場合などに実験者が数字入力 (DTMF) を用いて操作を代行する．しかし，標準的な VoiceXML 処理系は対話中の実験者の関与を想定しない設計であるため，このような実験を行う場合には，対話パターン記述内に対話タスク機能に関する要素と評価実験支援に関する要素が混在する．また，VoiceXML が標準的に備えている DTMF 入力機能のみでは，実験者が可能な関与のバリエーションに限界がある．

2.3 対話処理系への要求

擬人化エージェントを用いた音声対話システムでは，エージェントの非言語的な振舞いなどが快適な HMI の実現に貢献すると期待される．しかし，非言語的表現をどのような方針に基づいて用いるべきか，といった指針はまだ明らかではなく，さまざまな試行錯誤が必要である．

そこで我々は，以下の特長を持つ擬人化音声対話エージェントの実験環境が必要であると考えている．

- ブートストラップ対話システムを効率的に実装でき，反復的開発が容易であること．
- モジュール構成が柔軟で，複数モダリティの入出力を容易に追加できること．
- 実験監視者のためのログ取得やエージェント操作機能を備えること．

表 1: Galatea のモジュール構成

モジュール名	機能
DM	VoiceXML 処理系
AM (AM-DCL)	各モジュールの制御
AM-MCL *	出力同期およびマクロ処理
DM-MCL *	DM 関連のイベント処理
FSM	顔画像合成
SSM	テキスト音声合成
SRM	音声認識
SIM	意味解釈（スケルトン）
GUI	ユーザ向け画面
MON	対話監視者向け画面
SND	音声ファイル出力

*はブロードキャスト指定のモジュール.
下線は新たに追加・拡張したモジュール.

2.4 Galatea ツールキット

前述したシステムを開発するに当たって，我々は Galatea ツールキット（Linux 版）[3] のアーキテクチャを使用する．Galatea は音声認識モジュール (SRM)，音声合成モジュール (SSM)，顔画像生成モジュール (FSM)，エージェント管理部 (AM)，対話管理部 (DM) から構成され，それぞれのモジュールは共通の仮想マシンモデルに基づいて通信を行う．DM は AM をサブプロセスとして起動する．AM は各モジュールを AM のサブプロセスとして起動し，標準入出力を介して通信を行う．

AM は，DCL (Direct Command Layer) および MCL (Macro Control Layer) から構成されている．DCL は各モジュールを直接制御するコマンドセットを提供し，主に DM からみた利便性を実現する．これに対して MCL は，エージェントからの音声出力とリップシンクなどの同期処理を行い，これに必要な状態遷移の管理を行う．また，DM によって定義されたマクロコマンドの処理を行う．さらに，各モジュールが発したメッセージの中で，送信先が指定されていない（ブロードキャスト）メッセージの配送処理などを行なう．

Galatea はカスタマイズが容易で拡張性が高い点が特長とされており，各モジュールは単体での実装，テスト，利用が容易に行える．拡張性や柔軟性に関する検討として，AM を利用する複数の対話管理部（タスクプログラム）の実装が可能であることが確認されているが [3] ，モジュール構成のカスタマイズや対話タスク記述の柔軟性に関しては検討が行われていない．

3. モジュールの追加

本章では，Galatea のアーキテクチャを活かして我々が行なうモジュール追加作業について述べる．本報告における Galatea のモジュール構成を表 1 に示す．ただし，下線は新たに追加・拡張したモジュールである．

3.1 出力手段の追加

擬人化エージェントと人間の対話を検討するための実験には次のような出力手段が必要となる．

(1) 音声合成（リップシンクを含む）

(2) ユーザ向け出力（テキストや画像の表示，オーディオ出力，状況に応じた顔画像の制御など）

(3) 実験者向けモニタ出力（各サブモジュールのログ出力，対話管理部の内部状態の出力）

図 1: ユーザ用画面（ボタン配置は実装の一例）

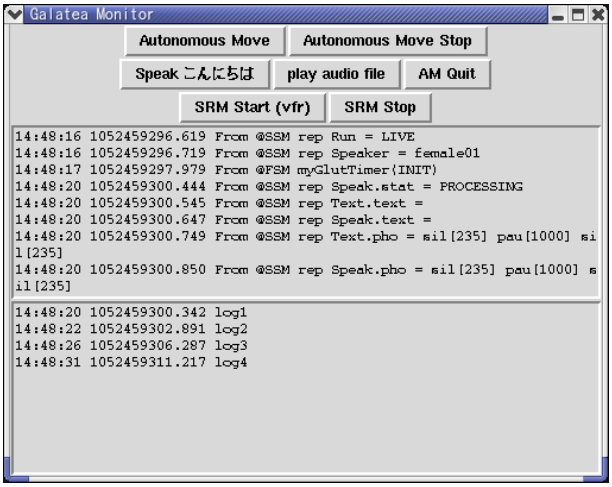

図 2: 実験者用対話モニタ画面の例（上段：ボタン入力，中段：AM のログ表示，下段：DM のログ表示）

これらのうち音声合成と顔画像制御はすでに SSM, FSM によって実現されている．そこで以下のモジュールを新たに実装した．

SND 効果音や録音音声の出力を行う．ファイル名を指定して再生を開始し，出力の開始および終了がモジュールから返答される．

GUI ユーザ向けの画面表示（図 1）において，テキスト表示やボタン操作による入力機能（後述）を提供する．

MON 実験者向けのモニタ表示（図 2）において，ボタン入力，AM のログ表示，DM のログ表示を行う．

SND は Perl 言語により実装し，音声ファイルの再生は外部プログラムの呼び出しによって行う．また，GUI および MON は Ruby/TK によって実装する．追加した各モジュールの入出力コマンド例を図 3 に示す．

3.2 入力手段の追加

GUI および MON の各モジュールにおいては，被験者による音声 (SRM) と等価なマウス入力およびマルチモーダル操作，実験者による各モジュールの直接制御などを行える機能が望まれる．

```
[to SND]
set Play = /path/file.wav
[from SND]
tell start /path/file.wav
tell end /path/file.wav

[to GUI]
set Text = message

[to MON]
set LogText = message
set AppLogText = message
```

図 3: モジュール SND, GUI, MON の入出力例

```
[from SRM to SIM]
set RecogStatus = STATUS="STARTREC"
set RecogOut = <RECOGOUT>
set RecogOut = <SHYPO RANK="1" SCORE="-3556.4">
set RecogOut = <WHYPO WORD="silB"/>
set RecogOut = <WHYPO WORD="はい"/>
set RecogOut = <WHYPO WORD="silE"/>
set RecogOut = </SHYPO>
set RecogOut = </RECOGOUT>

[from SIM (broadcast)]
tell status input start from SRM
tell result はい

[from GUI to SIM]
set ButtonPressed = はい

[from SIM (broadcast)]
tell status input start from GUI
tell result はい
```

図 4: モジュール SIM の入出力例

我々はこれらの機能を，W3C が提案する MMI Framework [10] を参考にしながら実現することにした．

マルチモーダルの入力操作においてはモダリティ統合処理が必要である．また W3C は現在，音声認識文法へのタグの埋め込みによる意味解釈処理について仕様を策定中である[*3]．我々はこれらの機能について，現時点ではスケルトンのみを提供し，実装はアプリケーション開発者に委ねる．

我々は，MON および GUI モジュールからのボタン操作と SRM からの音声認識結果を統合する役割を，新たなモジュール SIM (Semantic Interpretation Module) に持たせた．また，SRM や GUI からのユーザ入力イベントを SIM に転送する役目を持つ DM-MCL モジュールを実装した[*4]．

SRM, MON, GUI は各モジュールにおけるユーザ入力イベントを送り先の指定なしに出力する．DM-MCL はブロードキャスト指定モジュールとしてこれらのイベントを監視し，逐次 SIM に送る．SIM は受理したイベントから意味的情報を抽出する（現在の実装では単に認識結果から文字列を抜き出す）．DM はユーザ入力として SIM からのメッセージを監視する．

SIM のスケルトンの実装は Perl で行なうが，使用する文法に応じた拡張や置き換えを想定している．図 4 に SIM の入出力の例を示す．

なお，GUI および MON モジュールにおいて，特定のアプリケーションを前提としてボタンを追加修正する場合には，Ruby/TK コードの改変のみで簡単に対応できる．

4. Galatea DM の実装

我々は Galatea のための VoiceXML 処理系 (Galatea DM) を開発している [11]．使用言語は Java (J2SE 1.4) であり，前章で述べた AM をサブプロセスとして実行する．

VoiceXML はシステム主導型のメニュー選択およびスロットフィリングを行うための制御機能を持つが，変数などのオブジェクトは ECMAScript [*5] の仕様に基づいており，多くの VoiceXML 要素で ECMAScript の実行文や条件式を記述することが可能である．そこで我々は ECMAScript エンジンとして Mozilla Rhino [*6] を使用している．

Galatea DM では VoiceXML 文書の解釈において，XML の DOM (Document Object Model) API を直接用いず，スキーマコンパイラ Relaxer [12] によって生成された VoiceXML パーザクラスを使用する．処理系はまず，整形済み XML ファイルであるか否かのチェックを行ない，DTD (Document Type Definition) に従った適正な VoiceXML ファイルであるかをチェックし，その後，VoiceXML 文書としての解釈と実行を行なう．各段階でのエラーは処理系のユーザに通知される．

対話インタプリタとしての拡張性を高め，処理系自身の実装やデバッグを容易にするために，Galatea DM では VoiceXML の制御機能を ECMAScript を含んだ内部命令に変換し，対話状態の遷移に応じて内部命令を実行する．内部命令への変換のみを行い出力することも可能であり，VoiceXML 文書が記述者の意図通りに解釈されているかをチェックできる．

新たなタグの追加は以下のような作業となり，言語仕様の拡張は比較的容易である．

1. RELAX スキーマ定義（または DTD）へのタグの追加と，Relaxer によるパーザクラスの自動生成.
2. VoiceXML 解釈部 (Relaxer の Visitor デザインパターンを使用）に，新たなタグを内部命令に変換するコードを追加.
3. 必要ならば内部命令処理系に機能を追加.

Galatea DM は以下のように出力項目 (OutItem) を分類している（カッコ内は対応する VoiceXML 要素）.

VoiceOutItem リップシンクを含む音声合成 (prompt)

AudioOutItem 音声ファイル出力 (audio)

LogOutItem ログ出力 (log)

NativeOutItem その他の Galatea 制御コマンド

これらは出力キューによって管理される．VoiceOutItem, AudioOutItem に関しては，出力終了イベントを受け取るまで次の出力を行なわない．NativeOutItem は VoiceXML が定義していない顔画像などの制御や拡張された各モジュールへの出力を行なう．

各 OutItem は出力内容および実行条件を ECMAScript 式として保持し，実際に出力される時点で Rhino によってこれらの式を評価し，その値を用いて出力を実行する．

Galatea DM は以下の手順で出力の処理を行う．

∗3 Semantic Interpretation for Speech Recognition, W3C Working Draft, http://www.w3.org/TR/semantic-interpretation/

∗4 DM-MCL の諸機能は AM-MCL に統合が可能であるが，AM-MCL のコードを改変しないためにモジュールを分割した．

∗5 http://www.ecma-international.org/publications/standards/ECMA-262.HTM

∗6 http://www.mozilla.org/rhino/

```
<block>
  <log>greeting begin</log>
  <native>to @FSM set HedRotAbs.1 = 0 10 0</native>
  <prompt>こんにちは</prompt>
  <native>to @FSM set HedRotAbs.1 = 0 0 0</native>
  <log>greeting end</log>
</block>
```

図 5: VoiceXML 拡張によるエージェント制御の例（顔の向きを変えて「こんにちは」と発話し，顔の向きを元に戻す）.

ドキュメント解釈処理 出力に関する VoiceXML 要素を，出力項目 OutItem を出力キューに追加するコマンド (AddOutItem) に変換する.

状態遷移処理 AddOutItem コマンドを実行し，出力キューに OutItem を追加する.

出力キュー処理 OutItem の実行条件を ECMAScript 値として評価し，実行条件が真であれば，出力内容を ECMAScript 値として評価し，評価結果の文字列を出力する.

5. 検討

各モジュールの動作検証を Redhat Linux 8.0 上で行なった.

本実装においては，SRM, SSM, FSM, AM, AM-MCL の既存モジュールを改変することなく，ただ新たなモジュール (DM-MCL, SIM, GUI, MON, SND) を AM の設定ファイルに追加するだけで所望の機能を実現できた.

コンソールから AM を起動して，標準入力から各モジュールにコマンドを送ることにより，全モジュールの動作確認が可能であった．また，GUI モジュールにテンキーや「はい」「いいえ」などのボタンを追加し，これらのボタンを押すことで，SIM から音声入力と同形式のユーザ入力イベントを発生させることができた.

MON モジュールには「顔の自律動作の ON/OFF」「音声認識の ON/OFF」など AM に直接コマンドを送信するためのボタンを設けて実行することができた．入出力に関するログは，ミリ秒単位での時刻と合わせて MON の画面に表示できた.

さらに，Galatea DM を各モジュールの機能に対応させることで，VoiceXML で記述された対話を実行中に，音声入力待機時のボタン操作が可能となったほか，VoiceXML の仕様に準拠した audio や log などのタグによって SND, MON モジュールを制御することができた.

現在は Galatea のコマンドを図 5 のように直接 VoiceXML に埋め込むための機能を実装中である.

6. まとめ

擬人化エージェントツールキット Galatea の高い拡張性を活かして，音声対話インタフェースの試作や実験を想定した VoiceXML 処理系を実装した．Galatea の既存モジュールに GUI 対応などいくつかのモジュールを追加し，マルチモーダル入出力，実験者の対話への関与，ログの表示などが容易に実現できることを確認した.

Galatea DM の開発に関しては，VoiceXML 2.0 仕様にできるだけ準拠して機能を追加すること，音声認識結果の意味処理やマルチモーダル入力統合などの機能を実現すること，などを今後予定している．さらに，SRM の認識文法の切り替えに連動して SIM の処理を切り替えるなど，入力関連の処理を充実させる必要がある．また，表情制御と音声出力を並行して行なう場合の記述や，GUI モジュールを用いた画像表示なども検討課題である.

今後，多くの開発者がさまざまなモジュールを Galatea に実装する場合には，モジュール名やコマンド名などの重複が生じるおそれがある．このような問題に対して松坂ら [13] はバージョン管理システム CVS の登録モジュール名を用いることを提案している．Galatea の枠組においても何らかの命名規則を検討する必要があろう.

また今後は本ツール群を用いて対話実験を行ない，擬人化エージェントをどのように制御することが対話において効果的であるかを検討する予定である.

謝辞

本研究の一部は情報処理技術振興協会 (IPA) の支援を受けた．また VoiceXML 処理系の開発に協力していただいた京都工芸繊維大学の荒木雅弘助教授および岐津三泰氏，Galatea の開発を通じて御議論いただいた Galatea プロジェクトメンバーの皆様および嵯峨山研究室の皆様に感謝します.

参考文献

[1] 宮崎昇: 話しことばを扱う音声対話システム, 人工知能学会研究会資料, SIG-SLUD-A203-04, pp.21-27, 2003.

[2] 嵯峨山茂樹, 他: “擬人化音声対話エージェントツールキット Galatea,” 情処研報 2003-SLP-45-10, pp.57-64, 2003-02.

[3] 川本真一, 下平博, 他: “ カスタマイズ性を考慮した擬人化音声対話ソフトウェアツールキットの設計, ” 情報処理学会論文誌, vol.43, no.7, pp.2249-2263, Jul 2002.

[4] 荒木雅弘, 小野佑, 植田喜代志, 西本卓也, 新美康永: XML-VoiceXML 変換ツールの開発, 情処研報 2000-SLP-34-33, 2000.

[5] 安達史博, 河原達也, 岡本隆志, 中嶋宏: VoiceXML の自動生成に基づく協調的な電話自動応答システム，音講論 1-5-18, pp.35-36, 2002-03.

[6] Bob Carpenter, Sasha Caskey, Krishna Dayanidhi, Caroline Drouin, Rberto Pieraccini: “A portable, server-side dialog framework for VoiceXML,” ICSLP-2002, pp.2705-2708, Denver, 2002.

[7] 畑岡信夫, 渡邊純一郎, 赤堀一郎, 立石雅彦, Eric Nyberg, Teruko Mitamura, Yasunari Obuchi: VoiceXML をベースにした音声対話管理方式の開発, 音講論 2-9-12, pp.85-86, 2002-09.

[8] Christina Bennet, Ariadna Font Llitjós, Stefanie Shriver, Alexander Rudnicky, Alan W Black: “Building VoiceXML-based applications,” ICSLP-2002, pp.2245-2248, Denver, 2002.

[9] M. Araki, J. Kurahashi, T. Matsumoto: “ Extreme experiment (XE) method for developing mixed initiative dialogue systems, ” SSPR-2003, pp.107-110, Tokyo, 2003.

[10] 中村有作, 桂田浩一, 山田博文, 新田恒雄: “ MMI 記述言語の標準化動向と XISL の対応について, ” 信学技報 SP2002-160, pp.73-78, 2002.

[11] 岐津三泰, 西本卓也, 荒木雅弘: 擬人化エージェントのための VoiceXML 処理系の開発, 人工知能学会研究会資料 SIG-SLUD-A201-01, pp.1-6, 2002-06.

[12] 浅海智晴: Relaxer Java/XML による Web 開発, ピアソンエデュケーション, 2001.

[13] 松坂要佐, 於久健太郎, 小林哲則: “ 多機能ロボット開発のための情報共有アーキテクチャの設計と実装, ” 電子情報通信学会論文誌 D-I, Vol.J86-D-I, No.5, pp.318-329, 2003.