G-001

同一アプリケーションを異なる端末から利用する際のMMI記述における再利用性の向上

Improvement in reusability of MMI description language for developing an application used on different terminals

青木一峰† 桂田浩一† 山田博文‡ 新田恒雄†

Kazumine Aoki Kouichi Katsurada Hirobumi Yamada Tsuneo Nitta

1 はじめに

近年，携帯電話やPDA・タブレットPC等のユーザ端末多様化に伴い，Webアクセスへのマルチモーダル対話（MMI）技術応用が活発に議論されている．その中で，SALTやXHTML＋Voiceなど，既存のマークアップ言語のマルチモーダル対応が試みられている．一方，我々もモダリティ拡張性の高いMMI記述言語XISL1.1（eXtensible Interaction Scenario Language）[1][2]を提案してきた．しかし，SALT[3]やXHTML＋Voice[4]は，扱うことのできるモダリティが限られ，またXISL1.1も多様な端末間での再利用性が高いといえないため，多様な端末で同時に実行可能なアプリケーションを効率よく開発できる記述言語が望まれている．

本報告では，MMI記述言語XISL2.0を提案する．XISL2.0は，音声対話記述で標準的言語となっているVoiceXMLをベースとしているが，入力データの受け渡しのための意味表現（EMMA）[5]を採用するとともに，端末環境に応じた複数モダリティ制御を可能にするルール記述を導入している．これによって，MMI記述の再利用性向上と，多様な端末で一つの対話シナリオが動作することを実現する．

2 XISL2.0の概要

2.1 言語仕様

XISL2.0（以下XISL）の設計にあたっては，VoiceXMLの対話記述能力を導入し，同時にマルチモーダル化に必要な拡張を行った．VoiceXMLは対話制御に必要な条件分岐や算術演算といった基本的命令セット，および対話の階層（セッション，アプリケーション等）や遷移といった概念を備えている．XISLは，同様の命令や概念を導入した．一方，VoiceXMLは入力モダリティとして音声とDTMF，出力メディアとして合成音声とオーディオのみの利用を想定しており，入出力の拡張性を持っていない．そこでXISLでは，新たな入出力の導入を容易にするため，モダリティ依存の記述をXISLの仕様外とし，開発者が自由に入力モダリティや出力メディアを使用できるようにした．図1にXISLの記述例を示す．この例は路線検索システムにおいて，出発駅と到着駅をユーザに尋ねる対話である．先頭の<form>①を訪れると，まずフォームレベルにある<fe>②が実行される．<fe>は，フロントエンドのモダリティ制御を定義するタグで，この例では，詳細を省略しているが，htmlを表示し音声認識文法を有効にしている．次に，最初の<field>③が実行され，その中に定義された

```
<form id="Select_Section">・・・①
  <fe><!--html 表示--></fe>
  <fe><!--音声認識文法--></fe>  }・・・②
  <field name="Dep" semantics="出発地/駅名/">・・・③
    <prompt>・・・④
      <fe><!--合成音声「出発駅を選択してください」--></fe>
    </prompt>
    <filled>・・・⑤
      <fe><!--合成音声「○○発が選択されました」--></fe>
    </filled>
  </ field>
  < field name="Arr" semantics="到着地/駅名/">・・・⑥
    <prompt>
      <fe><!--合成音声「到着駅を選択してください」--></fe>
    </prompt>
    <filled>
      <fe><!--合成音声「○○行が選択されました」--></fe>
    </filled>
  </ field>
</form>
```

図1．XISLの記述例

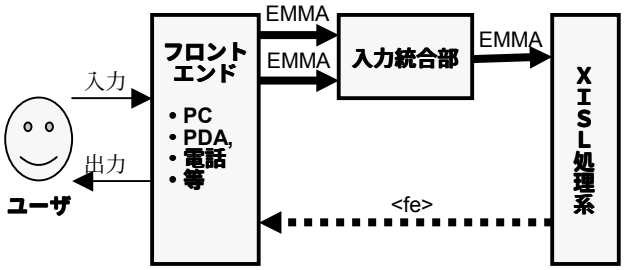

図2．実行システムの概要

<prompt>④によってガイド文が音声合成で読み上げられる．ユーザ入力の後，VoiceXMLのFIA（Form Interpretation Algorithm）と同様のアルゴリズムによって<field>に値が埋まると，<filld>⑤が実行され，応答文が音声合成で読み上げられる．続いて次の<field>⑥が実行され，同様の処理が行われる．

2.2 実行システム

以下にXISLの実行システムを簡単に説明する．図2は実行システムの概要を示している．実行システムは，大きく分けてXISL処理系とフロントエンドからなる．XISL処理系は，対話シナリオを処理し，外部処理系であるフロントエンドに<fe>を渡す．<fe>を受け取ったフロントエンドは，その内容に応じて入力モダリティと出力メディアの設定を更新する．このようにモダリティ／メディアの制御を全てフロントエンドに任せることで，新たなモダリティ／メディアの追加によるXISL処理系の修正を不要にしている．また，フロントエンドからXISL処理系への入力データの表現が，モダリティ毎に異なることを防ぐための統一

† 豊橋技術科学大学　大学院工学研究科

‡ 豊橋技術科学大学　マルチメディアセンター

```
<fe type="HTML" cond="FEtype=='PC' && User.Age>=60">
  <param name="document">
    <!--PC 用 html ドキュメント(高齢者向け)-->
  </param>
</fe>
<fe type="HTML" cond=" FEtype=='PC' && User.Age<60">
  <param name="document">
    <!--PC 用 html ドキュメント-->
  </param>
</fe>
<fe type="HTML" cond="FEtype=='PDA'">
  <param name="document">
    <!--PDA 用 html ドキュメント-->
  </param>
</fe>
```

図 3．フロントエンドに合った入出力仕様の選択

```
<emma:emma version="1.0"
          xmlns:emma="http://www.w3.org/2003/04/emma">
  <emma:interpretation id="Select_Section">
    <出発地>
      <駅名>東京</駅名>
    </出発地>
    <到着地>
      <駅名></駅名>
    </到着地>
  </emma:interpretation>
</emma:emma>
```

図 4．EMMA の例 （データモデルのみ)

表現として，EMMA（Extensible MultiModal Annotation markup language）を採用した．EMMA は，現在 W3C で検討中の仕様で，入力データの意味構造を表すモデルと入力情報を表現するための統一規格である．フロントエンドから送られる各種モダリティに対応した EMMA は，入力統合部で適切に統合された後，XISL 処理系に送られる．

3　XISLの再利用性向上

次世代携帯端末やカーナビゲーションなどでは，入力モダリティ／出力メディアの多様化にともない，異なる端末環境で実行可能なMMI記述が求められている．XISLはこの要求を満たすため，EMMA導入とともに，モダリティ（メディア）の記述とその選択ルール記述の導入を行った．

入力モダリティ（出力メディア）の記述とその選択ルール記述の導入により，端末の違いは若干の条件分岐で吸収できるようになった．この結果，異なる端末のシナリオも簡単に記述でき，シナリオの再利用性が格段に向上した．図3は，フロントエンドに合った入出力仕様をif-thenルールで選択する記述例を示している．If-thenルールは<fe>の属性condに条件式を記述する形で実現した．条件が真の場合その<fe>が実行される．この例では，ユーザが使用する端末がPCで，かつユーザが60歳以上の場合に，高齢者向けPC用htmlを表示するというルールが記述されている．

EMMAは，入力結果をモダリティ非依存なデータと意味構造に表現することができる．図4は，路線検索の際に「東京から」とユーザが発話した際に生成された例を示している．「東京」というデータは，「出発地」の「駅名」を意味することを表現している．EMMAで表現されたデータが，<field>の属性semanticsに定義された意味構造と一致した場合，その<field>の属性nameに定義された変数にデータの内容が埋められる．例えば図4のEMMAに表現された「東京」というデータは，意味構造が一致する図1の<field>③に埋められる．モダリティ／メディア選択ルールの導入に加えてEMMAを導入したことで，多様なモダリティに対して統一的な入力表現が得られるようになった．

4　他言語との比較と考察

XISLを他の言語と比較してその特長を明らかにする．比較対象としてSALT，XHTML＋Voiceを取り上げる．尚，これらの言語は再利用性に焦点を当てた言語ではないため，本節では言語全体の特長を比較する．

4.1　SALTとの比較

SALTはWebページ上に音声インターフェースを付加するためのタグセットで，XHTML等のマークアップ言語に埋め込む形で記述される．SALTタグは，XHTMLの<input>タグ等への音声認識結果のバインドや，XHTML文書内のテキストの読み上げを可能にする．このためXHTMLと親和性が高く容易に音声インターフェースを付与できる．

一方，SALTでは対話制御の記述ができず，制御はSMILもしくはスクリプトで記述する必要がある．これに対して，XISLは自身で対話を制御している．また，SALTは既存の文書にSALTタグを付与することで，その文書に音声モダリティを拡張するのに対し，XISLは，自由にモダリティ（メディア）を設定することができる．

4.2　XHTML＋Voiceとの比較

XHTML＋Voiceは，XHTML文書にVoiceXMLのタグセットを埋め込むことにより，音声によるフォームへの入力等を可能にする．XHTML文書に埋め込むという点で，SALTと同様のアプローチとみなせるが，VoiceXMLが対話制御を行うため，複雑な対話シナリオを容易に記述できる．

しかしながら，XHTML＋Voiceは，SALT同様に音声とポインティング等に限られたモダリティを想定しているのに対して，XISLはモダリティの拡張性を考慮し，モダリティの記述に自由度を持たせている点で異なる．また，同じVoiceXMLを拡張した言語であるが，XHTML＋VoiceはXHTMLが主に対話全体の制御を行うため，画面にまたがる対話記述が複雑になる．XISLはこの点，画面に関係なく対話を記述することができる．

5　まとめ

XISLでは，入力モダリティ／出力メディア選択ルールの導入に加えて，入力結果を表現するEMMAの導入によって，単一のシナリオを異なる端末上で実行することを可能にした．この結果，シナリオの再利用性が向上し，アプリケーション開発者の負担が軽減され，更にシナリオ変更に伴う保守性も向上した．今後，XISLの有用性を示すため，実行システムを開発して実証実験を行う予定である．

参考文献

[1] 桂田浩一，中村有作，山田真，山田博文，小林聡，新田恒雄：“ MMI記述言語XISLの提案 ”，情報処理学会論文誌，Vol.44, No.11，pp.2681－2689（2003）
[2] http://www.vox.tutkie.tut.ac.jp/XISL/XISL.html
[3] http://www.saltforum.org/
[4] http://www.voicexml.org/specs/multimodal/x+v/12/spec.html
[5] http://www.w3.org/TR/emma/