端末非依存なマルチモーダル対話記述言語の検討

青木一峰† 桂田浩一† 山田博文‡ 新田恒雄†
†豊橋技術科学大学 大学院工学研究科 ‡豊橋技術科学大学 工学部

1 はじめに

これまで我々はマルチモーダル対話(MMI)記述言語 XISL（eXtensible Interaction Scenario Language）[1][2]を提案してきた．XISL は，モダリティ記述に関する拡張性が高いため，多様な端末上での対話記述に利用できる．また XISL は対話シナリオが web コンテンツから分離して記述できるため，双方の再利用が可能であるという特徴を持つ．しかし実際には，端末毎に対話シナリオ中の入出力モダリティ記述が異なるため，XISL の再利用性は高いとは言えない．また，XML コンテンツの操作には CGI が必要であり，その作成は，アプリケーション作成者にとって大きな負担となっていた．これらを改善するために，まず端末依存の具体的な入力記述を XISL と分離することで，XISL の再利用性向上を図った．また，XForms[3]を参考にしたデータモデルと，新たなタグセットにより，XML コンテンツの操作を容易にする方法も検討した．

2 XISL1.1 の特徴と構造

これまで検討してきた XISL1.1 の記述例を図 1 に示す．<body>内には一組の対話を表す<dialog>が列挙される．<dialog>は，対話の最小単位である<exchange>の集合と，対話の導入処理を行う<begin>，終了処理を行う<end>をそれぞれ一つずつ内部に含む．

<exchange>は，<operation>（入力記述部）と，<action>（アクション記述部）をそれぞれ一つずつと，<prompt>（ユーザへのプロンプト提示）を複数記述できる．<operation>は単一の入力を記述する<input>の集合からなる．<action>，<prompt>，<begin>，<end>には，単一の出力を記述する<output>の集合が記述できる．加えて，<action>，<begin>，<end>には，CGI 実行のための<submit>の他，条件処理，繰り返し処理，演算処理等のための要素を記述できる．<input>と<output>の仕様は，モダリティの拡張性を高めるために，XISL の仕様外となっている．また，複数の<input>を逐次的，並列的，択一的に組み合わせた入力の統合方法や，複数の<output>を逐次的，並列的に組み合わせた出力の同期方法を記述できるため，複雑な対話を構成できる．

```
<xisl>
<head>･･･</head>
<body>
  <dialog id="sample" scope="dialog">
    <begin>対話の導入処理記述部</begin>
    <exchange>
      <prompt>･･･</prompt>
      <operation comb="par">
        <input type="touch" event="click"
          match="goods" target="ols.htm" return=”vgoods”/>
        <input type = "speech" event="recognize"
          match="spgram.gram#num" return=”vnum”/>
      </operation>
      <action>
        <submit next=”set_goods.cgi” namelist=”vgoods”/>
        <output type="agent" event="speech">･･･</output>
        ･･･
      </action>
    </exchange>
    <exchange>･･･</exchange>
    <end>対話の終了処理記述部</end>
  </dialog>
  ･･･
```

図 1. XISL1.1 の例

3 XISL1.1の問題点

XISL1.1では入力モダリティの指定と入力の統合方法の記述を<operation>に直接記述する必要がある．また，前述したように，端末ごとに<input>の仕様が異なるため，他端末でXISLドキュメントを再利用しようとした場合，<operation>内の記述を修正する必要がある．そのため，XISLドキュメントの再利用性や保守性が高いとは言えない．

また，XISL1.1では，XMLコンテンツからデータを取得する場合，図1に示すように<submit>を用いてCGIを実行する必要がある．CGIの作成は開発に大きな負担となるだけでなく，サーバー/クライアント間の通信も頻繁になるため，パフォーマンスが劣化する原因となる．

A Study on Terminal-Independent MMI Description Language
Kazumine AOKI †,Kouichi KATSURADA †,Hirobumi YAMADA ‡,Tsuneo NITTA †.
†Graduate School of Engineering, Toyohashi Univ.of Tech.
‡Faculty of Engineering, Toyohashi Univ.of Tech.

4 XISLの改良検討

4.1 改良方針

再利用性向上のために，これまでXISLドキュメント内に記述してきた入力記述をマルチモーダルグラマーとして，外部ファイルへ分離する．これによりグラマーとシナリオの双方の再利用性が高められるだけでなく，保守性も向上する．

また，XMLコンテンツを容易に操作できるように，XFormsを参考にしたデータモデルをXISLに導入した．XISL内のデータモデルでは，XMLコンテンツと，そのコンテンツデータ同士の依存関係を定義する．データモデル中に定義されたXMLコンテンツは，従来のデータとしての役割だけでなく，対話シナリオからの参照や書き換えを可能にすることで，一時変数としての役割も果たせるようにした．この結果，データ操作に係わるCGIを作成する必要がなくなり，更に，サーバー/クライアント間の通信によるパフォーマンスの劣化も防止した．

4.2 新XISLの特徴と構造

前節で述べた改良方針を基に，新たなXISLの仕様を検討した．図2に記述例を示す．新XISLでは，<head>内にデータモデルを定義する<model>を複数記述できる．各<model>は，一つの<instance>と複数の<bind>を持つ．<instance>にはXMLコンテンツを定義する．図2のように，XISLドキュメント内に直接記述することもできるが，外部参照にすることもできる．<bind>には，XMLコンテンツ中のあるデータと他のデータとの依存関係を記述する．

<body>内には，XISL1.1同様，<dialog>が列挙される．<dialog>は，<exchange>の集合と，一つの<begin>の他に，新たに<prompt>と<action>をそれぞれ複数持てるようにした．また<exchange>内部には，<dialog>と同様の内容（<exchange>の入れ子は除く）を記述できるよう変更した．更に，<begin>，<prompt>，<action>内には，XISL1.1と同様の要素の他に，<grammar>を複数持てるようにした．<grammar>には，入力モダリティとそれらの統合方法を規定している既存のマルチモーダルグラマーファイルを指定する．図2の<grammar>で指定しているグラマーファイルの記述例を図3に示す． <begin>，<action>には，データモデル中のXMLコンテンツのデータを参照する<get_element>や，データを書き込む<set_element>などを記述できるようにした．

```
<xisl>
  <head>
    <model id="Shopping">
      <instance>
        <SHOP>
          <SELECT>
            <GOODS/><PRICE/><NUM/><SUBTOTAL/>
          </SELECT>
          <LIST>
            <GOODS name="りんご" price="100"/>
            ・・・
      </instance>
      <bind ref="SHOP/SELECT/SUBTOTAL"
calculate="SHOP/SELECT/PRICE*SHOP/SELECT/NUM"/>
    </model>
  </head>
  <body>
    <dialog id="sample" scope="dialog">
      <begin>対話の導入処理記述部</begin>
      <prompt>
        <output type="agent" event="speech">・・・</output>
        <grammar rule="mmigram.xml#goods_num"/>
        ・・・
      </prompt>
      <action>
        <set_element var="vgoods"
            ref=" SHOP/SELECT/GOODS"/>
        ・・・
      </action>
      <exchange>・・・</exchange>
      ・・・
```

図 2. 新 XISL の例

```
<grammar id="goods_num">
  <operation comb="par">
      <input mode="touch" match="ols.htm#goods"/>
      <input mode="speech" match="spgram.gram#num"/>
  </operation>
</grammar>
```

図 3. マルチモーダルグラマーの例

5 まとめ

新しいXISLでは<grammar>要素とマルチモーダルグラマーの導入により入力記述の外部ファイル化を実現し，XISLドキュメントの再利用性を高めた．またXFormsを参考に，<model>，<instance>，<bind>等の要素を導入することで XMLコンテンツの操作を容易にした．今後は，XISLの細部の仕様について検討しXISL2.0として提案したい．また，有用性を示すために，実行システムを開発し，実証実験を行う予定である．更に，マルチモーダルグラマーの仕様を，現在W3Cで議論されているmultimodal integration grammarsを参考に詳細化したい．

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.w3.org/TR/xforms/