音声言語情報処理 37－8
（2001．7．14）

マルチモーダル対話記述言語XISLの提案

○小林 聡*、中村 有作**、桂田 浩一**、山田 博文***、新田 恒雄**

* 豊橋技術科学大学 情報処理センター, ** 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,
〒 441-8580 愛知県 豊橋市 天伯町 雲雀ヶ丘 1-1
skoba@cc.tut.ac.jp

あらまし: 本報告では、マルチモーダル対話記述言語 XISL (Extensible Interaction-Sheet Language) について述べる。XISLは、マルチモーダル対話 (MMI) の 記述と、XMLコンテンツからのインタラクション記述の分離を目的として開発されている。XISLにより、XMLドキュメントは端末間で異なる操作環境から独立し、シームレスなサービスを可能とする。XISLは、複数のモダリティーを組み合わせた様々な対話を可能にするため、XMLコンテンツ開発者はMMIシナリオを容易に作成できるようになる。我々は、XISLのインタプリタをMMI講義システムのプロトタイプシステム上に実装し、動作を確認した。

キーワード: ヒューマンインタフェース, マルチモーダル対話, インタラクション記述, XML, XISL

A Proposal of Multimodal Interaction Description Language XISL

○ Satoshi KOBAYASHI*, Yusaku NAKAMURA**, Kouichi KATSURADA**,
Hirobumi YAMADA***, Tsuneo NITTA**

* Computer Center, Toyohashi Univ. of Technology ** Graduate School of Technology,
Toyohashi Univ. of Technology, *** Multimedia Center,Toyohashi Univ.of Technology
〒 441-8580 1-1 Hibarigaoka Tenpaku-cho Toyohashi-shi Aichi-ken

Abstract: In this paper we outline a multimodal interaction description language XISL (Extensible Interaction-Sheet Language) that is developed to describe multimodal interactions (MMI), and to separate the description of interactions from XML contents. XISL makes an XML document independent of interactions that may differ between each terminal, and so enables such seamless services as web-browsing to be constructed easily. Since XISL also provides various combinatorial usage of modalities, a developer can describe a MMI scenario easily. We implemented an interpreter of XISL on prototype systems for multimedia lectures with different types of MMI and proved the viability of XISL by experiments using the systems.

Keyword: Human-Computer Interface, Multimodal Interaction (MMI), Interaction Description, XML, XISL

1 はじめに

近年、マルチメディア環境が急速に整備されるにつれて、マルチメディア・アプリケーションのユーザ・インタフェース (UI) は次第に重要になりつつある。マルチモーダル・インタラクション (MMI) は親しみやすく、円滑かつロバストな操作を可能にするが、ユーザ端末の違いへの対応や、MMIデザインの難しさなど多くの問題がある。例えば、ネットワークを介したマルチメディア・アプリケーションでは、サーバはユーザ端末に応じて入出力モダリティを提供しなければならない。一つの解決方法は、異なるモダリティーの組み合わせに対応する、複数のサーバを構築しておくことであるが、このようなアプローチは非効率的で高くつく。

近年、XMLドキュメントが様々なwebサービスのコンテンツに利用され始め、様々な端末を通して閲覧できるようになった[1]。この新しいwebサービスでは、開発者は個々の端末に応じたサーバとスタイル・シート (XSL[2] など) を用意することで、画面表示の差異に対応できるようになった。一方インタラクションの記述では、VoiceXML[3]が提案され、電話を介してのインタラクションを記述できるようになった。しかし、VoiceXMLではインタラクションとコンテンツが混在して記述され、また利用可能なモダリティーも音声とDTMFに限られている。このように、MMIシステムを異なるUI環境に対応させ、シームレスなサービスを実現することは、現状の技術では困難である。

我々は、インタラクション記述言語XISL (Extensible Interaction-Sheet Language) を提案する。XISLはコンテンツからインタラクションに関する記述を分離し、コンテンツを独立に利用できるようにすると共に、ネットワークを介したシームレスなMMIサービスの構築を支援することを狙っている。

先行研究 [4] ではシステム動作のみが分離されているが、XISLでは、ユーザーの操作とシステムの動作をXMLコンテンツとは独立に記述する。図1は、XSLとXISLというUIに関係する2種類のシートをXMLコンテンツと共にサーバに用意することで、PC、携帯情報端末、携帯電話、デジタルTV、カーナビなど様々な端末を介したシームレスなサービスを提供するコンセプトを示している。

以下では、2. でシステム・アーキテクチャの概要を述べる。3. ではXISLについて解説する。また4. において、MMI講義システムプロトタイプ上でのXISLの実装について説明した後、5. で今後の課題などをまとめる。

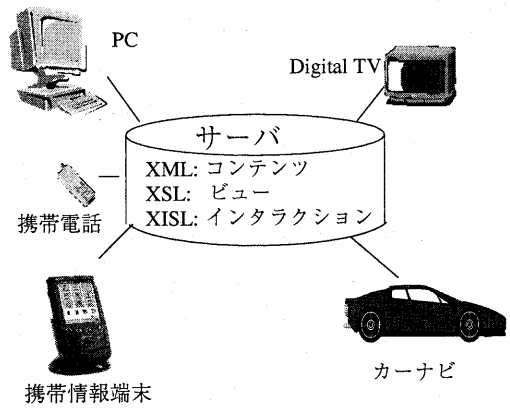

図 1: シームレス・サービス

表 1: 入力統合表

\op.	Pointing		Voice	
state	command	object ...	command	object...
0	s4	s1 ...	s5	s2 ...
1	r1	p1 ...	r2	p2 ...
2	r3	p3 ...	r4	p4 ...
:	:	:	:	:

s: 遷移, r:受理&reduce, p:処理

2 システム・アーキテクチャ

図2に、MMIシステムの構成図を示す。この例では、入力として音声とポインティングジェスチャを、また出力としてグラフィックスと音声を持つ。音声入力は指定された文法に従って認識され、結果が入力統合部へ送られる。入力統合部では、認識結果を意味コードの並びに変換する。この際、XISLから作成した統合表を用いて、モダリティーの統合、競合解決、タイムアウト処理などを行なっている (表1[5])。マルチモーダル対話マネージャは、XISLに記述された入力操作 (operation) を解釈し、結果を入力統合部に統合表の形で送る。また、入力の結果として得られる出力動作 (action) に関する情報を出力制御部に送る。出力制御部では、応答戦略決定木によりユーザーの設定、使用された入力モダリティー、出力のカテゴリー (System Error, Warning, Notice, Prompt, etc...) に基づいてプランを立て [6]、モダリティーと応答 内容を出力モジュールに伝える。

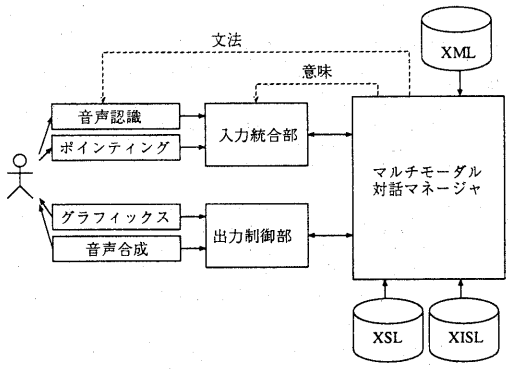

図 2: MMI システムのブロック図

3 Extensible Interaction-Sheet Language

3.1 設計の方針

XISL 策定の第一の目的は、XML コンテンツからインタラクション記述を分離することである。XML ドキュメントにおいて、インタラクションは個々のオブジェクトに対して定義されるため、XISL においても特定のオブジェクトに対する入力とその動作をペアにして記述する。

第二の目的は、XISL を用いることで MMI シナリオを簡単に記述できるようにすることである。複数のモダリティがインタラクションの中で使用される場合、それらは、同時、もしく逐次的に、また時には択一的に組み合わせられることが多い。出力の場合も同様である。このような多様な組み合わせを簡潔に記述するため、対話の流れを制御するいくつかのタグと属性を用意した。

3.2 XISL の構造と要素

XISL ドキュメントの例を図 4から図 6に示す。

XISL ドキュメントは、head 部と body 部からなる (図 4〜図 6内の a,b,c。以下同じ。)。head 部には、XISL ドキュメントに関する情報を記述し、body 部は対話を記述する。

body 部は dialogue の集合である。dialogue には、初期設定を行なうもの (d)、割込みなどを設定するもの (g)、そして一般の入出力を行なうもの (n) がある。また dialogue は、enter 部 (e)、exchange 部 (h)、exit 部からなる。対話の最小単位を構成する exchange 部は、operation 部 (i) と action 部 (k) からなる。enter 部と exit 部は、それぞれ dialogue に入った時、および出る時に実行される特殊な action 部である。operation 部には、入力イベントを (j)、そして action 部にはその入力に対応する動作を記述する。

operation 部は input からなり、現在はタッチ (q) と音声 (j,u) が記述可能である。音声の場合は、認識語彙や文法を指定できる。入力イベントの結果は、指定した変数に保存することによって、action 部で利用できるようになっている。また、複合した入力の流れの制御も、comb 属性やタグを用いて記述できる (t)。さらに、repeat 属性により、繰返しの制御も可能である (r)。operation 部は、インタプリタにより入力統合表 (表 1) へ変換され、入力統合部へと送られる。

action 部では、他の dialogue を呼び出す (l)、呼び出し元へ復帰する (m)、外部ファイルや変数からオブジェクト、値、属性値を読み込む (o,v,s) 機能の他、、出力 (p)、条件分岐 (w)、計算 (x,y) なども記述できる。また、変数も、"$"記号を付けて参照できるようになっている (f,y,z)。出力の差異には、埋め込まれた変数は自動的に展開される (p)。

4 XML と XISL の例

XML ドキュメントの例を図 3に、XISL ドキュメントの例を図 4から図 6に示す (XSL に関しては省く)。これはハンバーガーショップの例である。

図 3ではまず、「いらっしゃいませ」と初期画面が表示される。次に画面下の「注文に進む場合はここを押して下さい」という表示部分にタッチすることで、次の注文画面に進む。

注文画面では、商品の名前にタッチして、個数を言うことで、注文が累積される。また、システムは注文を復唱する。注文画面では、その他、商品の注文数と合計金額を表示する。

この例では、ユーザーは画面へのタッチと音声を用いて対話することが可能になっている。

5 プロトタイプ・システム

我々は、XISL インタプリタを Visual C++[1] を用いて実装し、MMI 講義システム上で XISL の有効性を確認した。図 7に、MMI 講義システムの概要を示す。コンテンツは XML で、ビューは XSL で、インタラクションは XISL で記述する。このシステムでは、音声、ポインティング、およびデジタルインクを

[1] Visual C++は Microsoft 社の登録商標である。

```
<?xml version="1.0" encoding="Shift-JIS"?>
<?xml-stylesheet type="text/xsl" href="ham.xsl"?>
<!DOCTYPE hamburger SYSTEM "ham.dtd">
<hamburger>
  <page id="1">
    <title>いらっしゃいませ</title>
    <button> 注文に進む場合はここを押して下さい</button>
  </page>
  <page id="2">
    <title>商品選択</title>
    <table>
      <tr>
        <th>商品名</th><th>価格</th><th>数量</th>
      </tr>
      <tr>
        <name>ハンバーガー</name>
        <price id="ham" value="200">200 円</price>
        <quant>$HAMBURGER</quant>
      </tr>
      <tr>
        <name>チーズバーガー</name>
        <price id="cheese" value="250">250 円</price>
        <quant>$CHEESE</quant>
      </tr>
      <tr>
        <name>コーラ</name>
        <price id="cola" value="150">150 円</price>
        <quant>$COLA</quant>
      </tr>
      <tr>
        <name>オレンジ・ジュース</name>
        <price id="oj" value="150">150 円</price>
        <quant>$ORANGE</quant>
      </tr>
      <tr>
        <name>合計</name>
        <price>$TOTAL</price>
        <quant></quant>
      </tr>
    </table>
    <p>ご注文は以上でよろしいですか?</p>
    <button>[yes]</button>
  </page>
  <page id="3">
    <title>ありがとうございました</title>
    <p>奥に進み、ご会計の後、商品をお受け取り下さい。</p>
  </page>
</hamburger>
```

図 3: XML ドキュメントの例

```
<?xml version="1.0" encoding="Shift-JIS"?>
<!DOCTYPE xisl SYSTEM "xisl.dtd">
<xisl>      ------------------------------------ (a)
  <head>    ------------------------------------ (b)
    <content>hamburger.xml</content>
    <name>hamburger.xisl</name>
    <author>小林 聡</author>
  </head>
  <body>    ------------------------------------ (c)
    <dialogue id="initial" type="initial"> ----- (d)
      <enter>   ------------------------------- (e)
        <var name="TIMEOUT" value="1000"/>  ---- (f)
        <call dialogue_name="page1"/>
      </enter>
    </dialogue>
    <dialogue id="interrupt" type="interrupt"> - (g)
      <exchange>    ---------------------------- (h)
        <operation> --------------------------- (i)
          <input namelist="speech"  match="page"
            type="speech" event="へるぷ"/>    --- (j)
        </operation>
        <action>  ----------------------------- (k)
          <call file_name="help.xisl"
            dialogue_name="initial"/> ---------- (l)
          <return/> ----------------------------- (m)
        </action>
      </exchange>
    </dialogue>
    <dialogue   id="page1" type="normal">   ---- (n)
      <enter>
        <get_object target="hamburger.xml"  ---- (o)
          var="hamburger" match="page[@id=1]"
          place="temp"/>
        <output type="display"  ---------------- (p)
            event="show_page">
          <![CDATA[
            <?xml version="1.0" encoding="Shift-JIS"?>
            <?xml-stylesheet type="text/xsl"
                href="ham.xsl"?>
            <!DOCTYPE hamburger SYSTEM "ham.dtd">
            <hamburger>
              $hamburger
            </hamburger>
            ]]>
        </output>
      </enter>
      <exchange>
        <operation> --------------------------- (q)
          <input match="page[@id=1]/button"
            type="touch" event="click"
            namelist="touch"/>
        </operation>
        <action>
          <call dialogue_name="page2"/>
          <return/>
        </action>
      </exchange>
    </dialogue>
```

図 4: XISL ドキュメントの例 (1)

```
<dialogue id="page2" type="normal"
          repeat="*">   -------------------- (r)
  <enter>
    <get_object target="hamburger.xml"
      var="hamburger" match="page[@id=2]"
      place="temp"/>
    <output type="display" event="show_page">
      <![CDATA[
        <?xml version="1.0" encoding="Shift-JIS"?>
        <?xml-stylesheet type="text/xsl"
            href="ham.xsl"?>
        <!DOCTYPE hamburger SYSTEM "ham.dtd">
        <hamburger>
          $hamburger
        </hamburger>
        ]]>
    </output>
    <get_attr target="$hamburger"  --------- (s)
      var="price_ham"
      match="price[@id='ham']/@value"
      place="variable"/>
    <get_attr target="$hamburger"
      var="price_cheese"
      match="price[@id='cheese']/@value"
      place="variable"/>
        :
  </enter>
  <exchange>
    <operation comb="par"> ----------------- (t)
      <input namelist="touch"
        match="page[@id=2]/name"
        type="touch" event="click"/>
      <input namelist="speech"  ------------ (u)
        match="page[@id=2]"
        type="speech" event="(いっ|に|さん) こ"/>
    </operation>
    <action>
      <get_value target="touch"    --------- (v)
        var="product" match="name"
        place="temp"/>
      <switch var="speech"> ---------------- (w)
        <case value="いっこ">
          <eval var="quant" expr="1"/> ----- (x)
        </case>
        <case value="にこ">
          <eval var="quant" expr="2"/>
        </case>
        <case value="さんこ">
          <eval var="quant" expr="3"/>
        </case>
      </switch>
      <switch var="product">
        <case value="ハンバーガー">
          <eval var="HAMBURGER" ------------ (y)
                expr="$HAMBURGER+$quant"/>
          <output type="speech"
            event="speech_synth">
            はんばーがーを
            $HAMBURGER こです。
          </output>
        </case>
        <case value="チーズバーガー">
              :
        </case>
```

図 5: **XISL**ドキュメントの例 **(2)**

```
        <case value="コーラ">
              :
        </case>
        <case value="オレンジジュース">
              :
        </case>
      </switch>
      <eval var="TOTAL" -------------------- (z)
        expr="$HAMBURGER * $price_ham +
              $CHEESE * $price_cheese +
              $COLA   * $price_cola   +
              $ORANGE * $price_oj"/>
      <output type="display" event="show_page">
        <![CDATA[
          <?xml version="1.0" encoding="Shift-JIS"?>
          <?xml-stylesheet type="text/xsl"
            href="ham.xsl"?>
          <!DOCTYPE hamburger SYSTEM "ham.dtd">
          <hamburger>
            $hamburger
          </hamburger>
          ]]>
        </output>
      </action>
    </exchange>
    <exchange>
      <operation>
        <input namelist="touch"
          match="page[@id=2]/button"
          type="touch" event="click"/>
      </operation>
      <action>
        <call dialogue_name="page3"/>
        <return/>
      </action>
    </exchange>
  </dialogue>
  <dialogue id="page3" type="normal"
            repeat="*">
    <enter>
      <get_object target="hamburger.xml"
        var="hamburger" match="page[@id=3]"
        place="temp"/>
      <output type="display" event="show_page">
        <![CDATA[
          <?xml version="1.0"
            encoding="Shift-JIS"?>
          <?xml-stylesheet type="text/xsl"
              href="ham.xsl"?>
          <!DOCTYPE hamburger SYSTEM "ham.dtd">
          <hamburger>
            $hamburger
          </hamburger>
          ]]>
      </output>
      <sleep time="2000"/>
      <return/>
    </enter>
  </dialogue>
 </body>
</xisl>
```

図 6: **XISL**ドキュメントの例 **(3)**

入力に持つ。また出力としては、スライド、擬人化エージェント、および音声合成を持つ。

グラフィックスの表示には Microsoft InternetExplorer[2]を、擬人化エージェントには Microsoft Agent[3]を、音声認識には LaLa Voice を用いた[4]。ポインティングやデジタルインクは、透明なウィンドウをブラウザ画面に重ねて取得している。

マルチモーダル入力がシステムに与えられると、対応する意味コードに変換され、表1に示す入力統合表 (XISL の operation 部から生成) により統合される。その結果、XISL 記述の対応する exchange が確定され、出力動作 (action 部) が実行される。最後に action 部の記述を解釈しながら、出力が実行される。

6 まとめ

本報告では、MMI 記述言語 XISL を提案するとともに、MMI 講義システム上での実装について述べた。コンテンツ開発者は、XISL を利用することで MMI システムを容易に構築することが可能になる。また、より迅速に MMI システムを構築できるように、GUI ベースのプロトタイピングツールについても検討している [7]。XISL の仕様は、現在、様々なドメインに適合させるための拡張作業を行なっている。ドメインとしては、旅行案内、カーナビ、オンラインショッピング、ロボットとのインタラクションなどを対象にしている。マルチモーダル対話マネージャには、対話の流れの制御など多くの課題があるが、今後、順次検討していきたい。加えて MMI システムでは、今後加わるであろう未知のモダリティ、あるいは不特定エンジンへの対応を考慮する必要がある。今後はこうしたインタフェースについても検討したい。

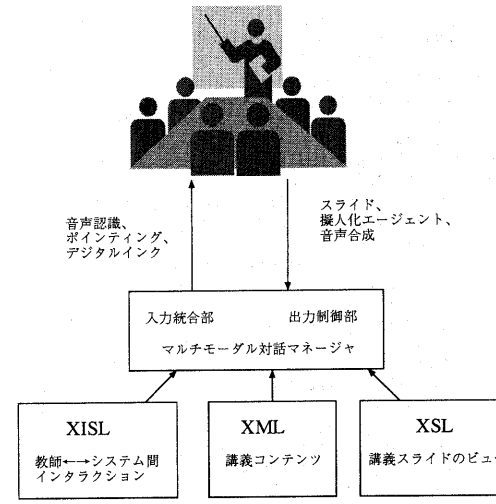

図 7: MMI 講義システム

参考文献

[1] http://www.w3.org/XML

[2] http://www.w3.org/Style/XSL/

[3] http://www.voicexml.org/,
http://www.w3.org/Voice/

[4] Wang, K. : "Implementation of a Multimodal Dialogue System using Extended Markup Languages", ICSLP2000, II-138-141, 2000.

[5] 山田真, 村上太一, 山田博文, 桂田浩一, 新田恒雄 : "マルチモーダルインタフェースにおける入力統合方法の検討", 情報処理学会第 62 回全国大会講演論文集 (分冊 4) ,1U-04,pp.87-88, 2001.

[6] 大谷佳彦, 鈴木浩和, 小林聡, 桂田浩一, 新田恒雄 : "マルチモーダルインタフェースにおける応答戦略の検討", 情報処理学会第 62 回全国大会講演論文集 (分冊 4) ,1U-05, pp.89-90, 2001.

[7] Kamio, H., Amamiya, M., Matsu'ura, H., and Nitta, T.: "Muse, a rapid prototyping tool", Advances in Human Factores/Ergonomics 21B - Design of Computing System (Proc. HCI International '97), pp. 569-572, 1997.

[2] Microsoft Internet Explore は Microsoft 社の商標である。
[3] Microsoft Agent は Microsoft 社の登録商標である
[4] Lala Voice は (株) 東芝の登録商標である。