XISL: A Devices/Contents Independent MMI Description Language

Kouichi Katsurada Hirobumi Yamada Yusaku Nakamura Satoshi Kobayashi Tsuneo Nitta

Toyohashi University of Technology
1-1 Hibarigaoka, Tempaku-cho, Toyohashi 441-8580, Japan
katurada@tutkie.tut.ac.jp

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 make the description of interactions independent of 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 is also independent of the devices, and has the flexibility to describe modalities, the system developer can easily change the user interface without introducing a new set of description language. We implemented an interpreter of XISL, and proved the viability of XISL by experiments using an online shopping application described by XISL.

1. Introduction

Recently, discussion on standardization of multimodal interaction (MMI) has just started, and several languages are specified to describe MMI. One language is SALT [1] which adds modalities of speech and dtmf to html pages. The others are the languages which expand current languages such as VoiceXML and SMIL [2][3]. We also have specified an MMI description language XISL (Extensible Interaction-Sheet Language) [4][5], and proposed it at the multimodal working group organized by W3C [6].

XISL is a multimodal description language which has two features that: 1) XISL description is separated from XML contents, and 2) XISL has flexibility to describe input/output modalities while previous languages only deal with speech and dtmf in addition to current PC's modalities. These features enable the system developers to reuse both the XML and XISL documents, and to change the user interface without introducing a new set of description language. Thus they can construct seamless services over networks easily. Figure 1 shows the concept of providing seamless services using PCs, mobile phones, digital TVs, and car navigators as terminals.

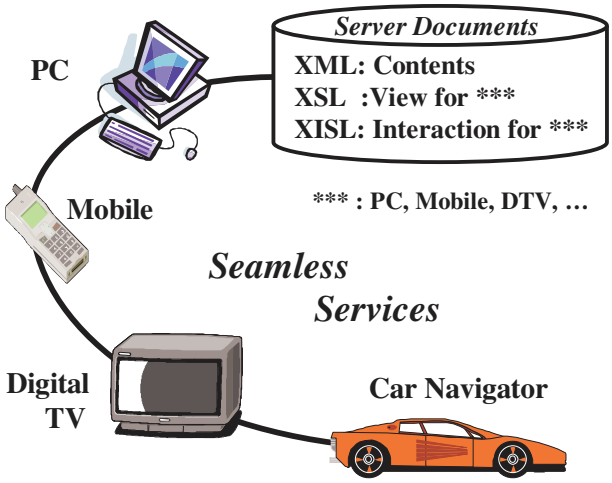

Figure 1 **Seamless Services over Networks**

This paper provides the latest version of XISL. In section 2, we outline the system architecture. In section 3, we describe the latest specification of XISL. In section 4, we explain the online shopping application described by XISL. And finally, in section 5, we discuss future works.

2. System Architecture

Figure 2 shows an outline of the MMI system composed of three modules: a front-end module, a dialog manager module, and a document server module. The document server is a general web server. It holds XISL, XML, and other documents (e.g. XSL style sheet, speech grammar file, and so on), and sends them to the dialog manager according to requests. The dialog manager interprets XISL documents, manages dialog flow, and controls inputs and outputs. The front-end is a user interface terminal that supports various modalities. Although the front-end has flexibility to support arbitrary modalities, we assume that the system in this paper supports microphone, speaker, mouse, and screen.

The XISL document is firstly sent to an XISL interpreter in the dialog manager module. The XISL interpreter interprets XISL, divides it into input operations and actions, and then sends them to the input integrator and the action module, respectively. The user's inputs from microphone and mouse are transferred to the input integrator, and integrated using an integration table. The integration table, generated from input operations of XISL, is represented in the form of a transition table that integrates input modalities, solves competitive input modalities, and handles time-out processing. After the input integration, the action module executes corresponding actions, and sends output information to output engines of graphics and TTS that are used for agent actions, balloons, and speech of an anthropomorphic agent, etc.

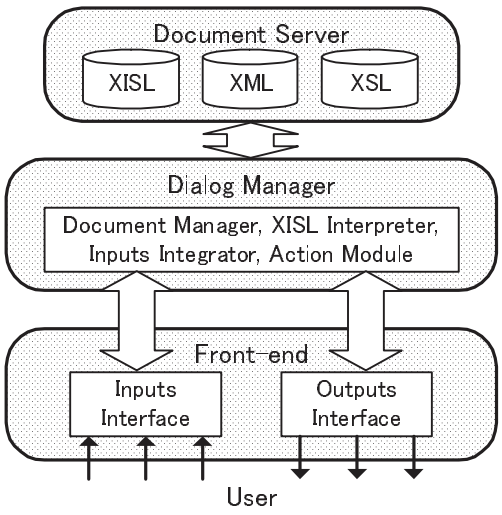

Figure 2 **Architecture of an XISL Execution System**

3. Extensible Interaction-Sheet Language

Figure 3 shows an example of an XISL document. The root of the XISL document is an <xisl> element ((a) in Fig. 3) that is divided into two elements: a <head> element ((b) in Fig. 3) and a <body> element ((c) in Fig. 3). The <head> element is a header of the document, and the <body> element is a container of <dialog> elements. Each <dialog> element contains at most one <begin> element ((e) in Fig. 3), one or more <exchange> elements ((f) in Fig. 3), and at most one <end> element, which are used to describe initial processes, a unit of interaction, and terminal processes, respectively.

An <exchange> element is composed of at most one <prompt> element, an <operation> element ((g) in Fig. 3), and an <action> element ((i) in Fig. 3). If an <exchange> element contains a <prompt> element, it is a system initiative interaction. On the other hand, if an

```
<?xml  version="1.0"  encoding="Shift-JIS"?>
<!DOCTYPE  xisl  SYSTEM  "xisl.dtd">
<xisl  version="1.0"> ---------- (a)
  <head> ..... </head> ---------- (b)
  <body> ---------- (c)
    <dialog  id="order"> ---------- (d)
      <begin> ---------- (e)
        <output  type="window"  event="open">
          <![CDATA[
            <param  name="window-name"> top-page </param>
            <param  name="uri"> ./OLS.xml </param>
          ]]>
        </output>
        <output  type="speech"  event="play">
          <![CDATA[  <param  name="speech_text">
            May I help you?
            </param>    ]]>
        </output>
      </begin>
      <exchange> ---------- (f)
        <operation  comb="par"> ---------- (g)
          <input  type="pointing"  event="l_click" ---------- (h)
                  match="/object/goods[item:=@id]/fig"/>
          <input  type="speech"  event="recognize"
                  target="count.grxml"
                  match="/grammar/rule[@id=count]"
                  return="number">
            <param name="mode"> data </param>
          </input>
        </operation>
        <action  comb="par"> ---------- (i)
          <get_value  target="Goods.xml"  var="item_name"
                      match="/item[@id=$item]/name"> ---- (j)
          <output  type="speech"  event="play"> ---------- (k)
            <![CDATA[  <param  name="speech_text">
              You ordered <value expr="number">
              <value expr="item_name">.---------- (l)
              </param>    ]]>
          </output>
          <switch  var="number"> ---------- (m)
            <case  value="one">
              <assign  var="Qty"  expr="1"/> ---------- (n)
            </case>
              :
          </switch>
            :
          <goto  next="payment"> ---------- (o)
        </action>
      </exchange>
        :
    </dialog>
  </body>
</xisl>
```

Figure 3 **Example of XISL**

<exchange> element contains no <prompt> element, it is a user initiative interaction. A <prompt> element, an <operation> element, and an <action> element represent a prompt to the user, a user's operation and a system action, respectively.

In a <prompt> element and an <operation> element, the author can describe <output> elements and <input> elements ((h) in Fig. 3), respectively. In an <action> element, the author can describe some elements such as

<output> elements ((k) in Fig. 3), <get_value> elements ((j) in Fig. 3), <switch> elements ((m) in Fig. 3), <assign> elements ((n) in Fig. 3), <value> elements ((l) in Fig. 3), <goto> elements ((o) in Fig. 3), and so on.

For the purpose of increasing the flexibility of modalities, strict attributes and contents of <input> and <output> elements are not specified by XISL. This enables us to expand or modify modalities easily. The descriptive details of them are intended to be specified by the platform developers.

An interaction flow is controlled by a "comb" attribute of the <dialog> tag whose value is either *par* (parallel), *seq* (sequential) or *alt* (alternative). When the "comb" is *par*, all <exchange> elements are executed in parallel. If the "comb" is *alt*, one of the <exchange> elements is performed alternatively, and if the "comb" is *seq*, all <exchange> elements are performed in sequence. We can also control interaction flows by means of <par…>, <seq…> and <alt…> tags. The elements bound by these tags are equivalent in functions to the corresponding attributes of the <dialog> tag. These tags and attributes can also be used in an <operation> tag and an <action> tag (*alt* and <alt…> do not appear in an <action> tag).

4. A Prototype System

We implemented an OnLine Shopping (OLS) application by XISL. The OLS application is composed of the dialog scenario for shopping (XISL), the data of items and customers (XML), and the style to display on the screen (XSL). Upon starting the system, a user firstly inputs his ID for authentication. The page shown in Figure 4 is then displayed. This page presents a shopping cart ((1) in Fig. 4), an item list ((2) in Fig. 4), and the list of categories of the items ((3) in Fig. 4). The user can modify the shopping cart, modify the item list, and go to the explanation page. After deciding the items to buy, the user determines the number of items, inputs his personal information, and ends the shopping by registering his personal information. The help page can be called up whenever the system is running. All of the user's inputs are made by speech, pointing, or both of them.

Figure 4 **An Online shopping system**

5. Discussion

In this paper, we described an MMI description language XISL, its elements and document structure. A system designer can develop an MMI system more easily by using XISL, but we consider that a GUI-based prototyping tool is needed for designing such systems quickly [7]. The tag set of XISL must also be expanded to apply XISL to various domains. There are many issues concerning the multimodal dialog manager; in further MMI studies, we will examine dialog flow control by using dialog context and task switching in multi-task applications.

References

[1] http://www.saltforum.org/

[2] http://www.w3.org/TR/xhtml+voice/

[3] Beckham, J.L., Fabbrizio, G.D., and Klarlund, N., "Towards SMIL as a Foundation for Multimodal, Multimedia Applications", Proc. of EUROSPEECH 2001, pp.1363-1366 (2001).

[4] Nitta, T., Katsurada, K., Yamada, H., Nakamura, Y., and Kobayashi, S.: "XISL: An Attempt to Separate Multimodal Interactions from XML Contents", Proc. of EUROSPEECH 2001, pp.1197-1200 (2001).

[5] http://www.vox.tutkie.tut.ac.jp/XISL/XISL-E.html

[6] http://www.w3.org/2002/mmi/

[7] Kamio, H., Amamiya, M., Matsu'ura, H., and Nitta, T., "Muse, a rapid prototyping tool", Proc. HCI International'97, pp.569-572 (1997).