

國立交通大學

資訊科學與工程研究所



研究生：周鴻仁

指導教授：袁賢銘 教授

中華民國 九十七年六月

視覺化語音服務系統

學生：周鴻仁

指導教授：袁賢銘

國立交通大學資訊科學與工程研究所

摘要

本論文提出了 VSXML 格式及其相關技術,使傳統語音服務可以迅速的被轉換成爲手機上的應用程式,達成使用者不需全程聆聽語音服務的目標。整個系統包含了 VSXML 的格式定義、VSXML Designer 和 VSXML Renderer. VSXML 是一個可將語音服務中各個語音問題單元結構化敘述的標記語言, VSXML Designer 則爲一個方便語音服務設計者將傳統語音服務視覺化,同時生成行動裝置適用的應用程式。VSXML Renderer 則根據不同的 VSXML 檔案,在行動裝置上呈現出相對應的控制項,同時也負責記錄整個語音流程的使用者回應並執行自動撥出。

透過在 Motorola 和 Nokia 手機上的實作測試,我們證明了 VSXML 格式對於傳統語音服務的轉換能力與整體系統的可行性及簡易性。整體而言,經由本系統及 VSXML 格式,不需要中途查詢資料庫的語音服務都可以完全的被視覺化。

A Visualized Voice Service System

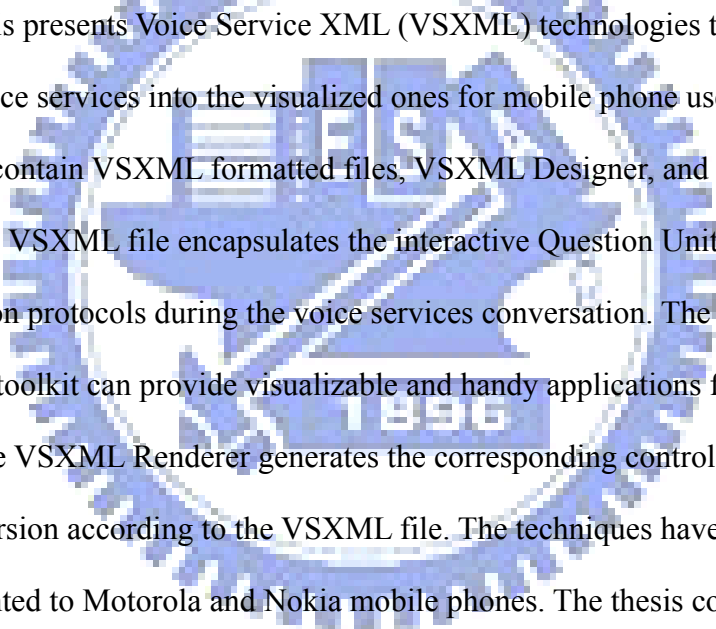
Student: Hung-Jen Chou

Advisor: Shyan-Ming Yuan

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

The logo of National Chiao Tung University is a circular emblem. It features a gear-like outer border. Inside the circle, there is a central shield or crest with various symbols, including what appears to be a book and a torch. Below the shield, the year '1896' is inscribed. The entire logo is rendered in a light blue, semi-transparent style, serving as a watermark behind the text.

The thesis presents Voice Service XML (VSXML) technologies to transform the traditional voice services into the visualized ones for mobile phone users. The technologies contain VSXML formatted files, VSXML Designer, and VSXML Renderer. The VSXML file encapsulates the interactive Question Units plus the communication protocols during the voice services conversation. The VSXML Designer is a toolkit can provide visualizable and handy applications for voice service designers. The VSXML Renderer generates the corresponding controls of the voice service conversion according to the VSXML file. The techniques have been designed and implemented to Motorola and Nokia mobile phones. The thesis concludes the technologies are flexible and easy to adapt. In general, the voice services can be visualizable completely if there is no database accessing. We also addressed the future work of the technologies.

Acknowledgements

首先，最重要的要感謝袁賢銘老師兩年來的指導，對於論文及專案，老師提供相當大的空間讓我自行發揮，同時從旁輔助給我建議，讓我得以順利的完成這篇論文。另外，感謝謝筱齡老師的協助，針對論文初稿的結構和文字做出了許多修正，也謝謝工研院的孫仲佑先生在我程式實作期間的所有幫忙。同時，謝謝實驗室的秉哲學長、繼弘學長、永威學長、家鋒學長，以及所有實驗室的碩班同學們，研究的過程中有許多不懂的地方，是你們幫我解惑。特別感謝我的室友瑋璿、書賢，還有我的好朋友田晏、奕丞、辰璞、捲毛，這兩年我們相互扶持打氣，是永遠的麻吉。

最後，感謝永遠在背後支持我，向我伸出溫暖的手的家人，我的爸媽。我將學生生涯中最後一個階段的成果獻給你們，沒有你們在背後做為我的推手，我沒有辦法任性的大步邁向自己選擇的路，我愛你們！



Table of Contents

Acknowledgements	III
Table of Contents	IV
List of Figures.....	V
List of Tables.....	VI
List of Codes	VII
1 Introduction.....	1
1.1 Preface.....	1
1.2 Motivation.....	2
1.3 Objectives	2
1.4 Research Contribution	2
2 Background.....	4
2.1 Voice Service and Telephony.....	4
2.2 The Patent	5
2.3 XML (Extensible Markup Language).....	6
3 System Design and Implementation.....	8
3.1 Overview.....	8
3.2 VSXML.....	9
3.3 VSXML Designer	14
3.4 VSXML Renderer	21
4 Scenarios Demonstration.....	24
4.1 System Flow Scenarios.....	24
4.2 Creating VSXML File.....	24
4.3 Rendering VSXML File.....	27
5 Results	30
5.1 Visualized Voice Service.....	30
5.2 VSXML.....	31
6 Future Works and Conclusion.....	33
6.1 Future Works.....	33
6.2 Conclusion	35
References.....	36
Appendix A	38

List of Figures

Figure 2-1 A large call center in Lakeland, Florida [3].....	4
Figure 3-1 System Architecture	8
Figure 3-2 Voice service description format of VSXML.....	10
Figure 3-3 NCTU voice service protocol of VSXML	12
Figure 3-4 Flow diagram of NCTU voice service	13
Figure 3-5 Screenshot of VSXML Designer.....	14
Figure 3-6 Screenshot of CONTAX IVR II.....	16
Figure 3-7 Same voice flow as figure 3-6 designed in VSXML Designer	17
Figure 3-8 Class diagram of module Entities	18
Figure 3-9 Rendering different types of Questions by VSXML Renderer	22
Figure 4-1 System Flow Scenarios Diagram	24
Figure 4-2 Voice Service Flow of Hsinchu Hospital	25
Figure 4-3 Add and Insert a Question.....	26
Figure 4-4 Screenshot of completed VSXML file of Hsinchu Hospital.....	27
Figure 4-5 Application Name on Motorola E1070 and Nokia 6210 Classic.....	28
Figure 4-6 Process of registering for Hsinchu Hospital with VSXML Renderer on Motorola E1070	28
Figure 5-1 Capabilities of VSXML	31
Figure Appendix- 1 Screenshot of completed VSXML file of Hwa Nan Bank.....	38
Figure Appendix- 2 Screenshot of completed VSXML file of Chang Hwa Bank.....	40

List of Tables

Table 4-1 Distribution of Question Type in Voice Service of Hsinchu Hospital	26
Table 5-1 Comparison between nonvisualized and visualized voice services (Relatively)	30



List of Codes

Listing 2-1 XML Document Sample	7
Listing 3-1 Implementation of Depth-first traversal algorithm	16
Listing 3-2 Loading VSXML file	19
Listing 3-3 Saving VSXML file.....	20
Listing 3-4 Building JAR file	21



1 Introduction

1.1 Preface

Customer Relationship Management (CRM) is a significant issue in many businesses, such as banking, food, and transportation. Most enterprises usually establish a 24-hour voice service system for customers to improve users' satisfaction. However, a regular voice system exists drawbacks, e.g. long instructions listening, service holding time. It is not convenient for mobile users; the phone line is busy frequently. In 2007, a patent proposed the idea to improve presentation of traditional voice services, which means to “visualize” voice services, and indicate this idea can solve problems of traditional voice services. Therefore, based on the patent, we develop an application that turns the voice service flow into a menu-driven interface. In other words, the voice service can be “visualized”. The application collects each option the user selected and dials the request sequence at once automatically.

In the paper, first, we analyze the general Question Unit over voice service systems. According to the communication protocols between mobile phones and the voice services system, we define and provide the corresponding scripts. Secondly, we describe and construct a XML standard format file for the voice services called VSXML (Voice Service XML). The file encapsulates the interactive Question Units plus the communication protocols. In this approach, we can improve the algorithm presented in the patent to achieve additional flexibilities. Therefore, we design a user-defined VSXML toolkit that can provide the visualizable and handy applications for developers.

1.2 Motivation

Voice service can be sensed in many places in our life, for example, asking about company's information、ordering food and applying for credit card...etc. Customers communicate with companies via phone line has become a habit. Taiwan Patent 200714012[1] proposed an algorithm to solve the traditional voice service system problems for mobile users. It utilizes the characteristic of mobile phone which can pre-dial DTMF (Dual-Tone Multi-Frequency)[2] tones prior to the actual dialing. For that reason, we start to cooperate with the author of patent to implement it. During design process, we realized that it is not easy to spread the idea if there is no common interface and useful tools. Therefore, we do the research into extension of the original idea and expect to make most voice services visualized more easily.

1.3 Objectives

There are several objectives in this research. First, we are going to implement the visualize voice service idea. We have to establish stable system architecture to make the system work well in reality. Second, try to visualize existing voice service quickly and easily as well as design a new one. Because of that, a description protocol or format is necessary. The protocol must be capable of representing general voice service. Last but not least, service designers should have an easy-to-use tool to design their services in this protocol.

1.4 Research Contribution

The thesis discussed how to visualize existing voice service efficiently. A complete solution was proposed including the description language of voice service – VSXML and its extension programs. Additionally, we proved that VSXML is able to

cover most voice services through experiments, and service designer could transform original voice service into VSXML file in a short period of time with VSXML Designer. On the other side, we tried to walk out the laboratory and to prove practicability of our solution via many real cases.



2 Background

2.1 Voice Service and Telephony



Figure 2-1 A large call center in Lakeland, Florida [3]

At beginning, the call center is bridged between companies and customers via telephone, or voice services. The customer service representatives need to answer all questions. After computer technologies advancing, lots of researches that integrate computer and phone technology are proposed [3], for examples, applying Text-to-Speech [4] to voice services as well as Interactive Voice Response (IVR) technology [5]. IVR systems make voice service automatic by analyzing DTMF tones [2]. It determines the phone button while customers pressed. In the mean time, it can

query database for the information customers requested. In the recent decade, the advanced voice services are based upon Internet and VoIP [6]. For example, VoiceXML [7] is a standard designed, supported by several worldwide companies, i.e., AT&T, Motorola and IBM. The goal of VoiceXML is using the accessibility of Internet to get services provided by voice sites. Another research topic is “Telephony Engine”. A telephony engine is a system which implements telephone exchange with software. The two most famous applications are Asterisk [8] and Yate [9]. Additionally, researches about Peer-to-Peer [10] Internet telephony become popular due to success of Skype [11] and instant messaging software, such as Windows Live Messenger [12]. These applications provide users to communicate with their friends using text-messaging or voice.

2.2 The Patent

According to the background section of patent “*Automatic Dial-Up System for Communication Apparatus*”, the best way to improve customers’ user experience is to let customer spend less time listening to voice service. Researches about user experience of voice service are not many recently. U.S. patent publication No. 20040153322 proposed an idea “An interactive voice response system has speak-ahead capabilities similar to type-ahead IVR systems by determining multi-level grammars for responses. Preferably, an existing IVR application is processed automatically to generate a multi-level grammar database that can then be used in recognizing multi-level responses by a user.”(Neuberger, Marc J., Panttaja, Erin M., 2004)[13] The idea relies on the speech recognition so much, but speech recognition technology is not perfect so far, especially in noisy environment.

The Patent “*Automatic Dial-Up System for Communication Apparatus*” proposed a system described as “The automatic dial-up system uses a server and a

communication apparatus for completing its services. The server obtains a working content of a voice service system, converts the working content to a corresponding table.“ (Sun, Chung-yo, Chueh, Shih-nan, 2006)[1] The concepts are “visualized” voice services, which means the content should be presented by voice is displayed by text on the screen.

During the implementation, we realized it is difficult to visualize voice services without a general solution. Hence we introduce, design voice service XML standards, VSXML, a client display application, VSXML Renderer, as well as a toolkit, VSXML Designer. We anticipate convert visualizable voice services by VSXML will be easy.

2.3 XML (Extensible Markup Language)

Definition of XML by W3C is “Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere“(W3C, 2006) [14]. In the early Internet development, information on the Internet are mostly text-based. Absence of definitions such as music and images limits capability of HTML, which is derived from SGML. In addition, HTML still has readable and poor scalability issues. The researchers developed XML, expectations for a wide range of data to provide a more accurate description of the way.

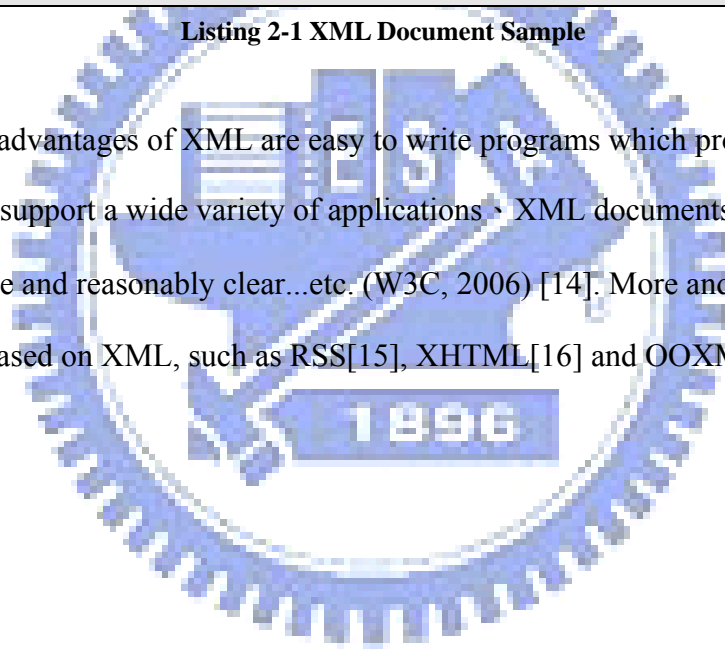
XML itself is not a markup language. It is a plain text format that lets users define their custom markup language. XML form is a tree-shaped structure. Listing 2-1 shows a typical XML document. The first line of a XML document is XML declaration, which defines the XML version and the encoding used. The second line describes the **root element** of this document (also called **document element**) which is <book> here, and the **attribute** “category” of <book> element represents that this

book is suitable for children. And the <book> element has four children : <title> , <author> , <year> and <price>.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<book category="Children">
  <title>Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2008</year>
  <price>19.99</price>
</book>
```

Listing 2-1 XML Document Sample

Primary advantages of XML are easy to write programs which process XML documents , support a wide variety of applications , XML documents should be human-legible and reasonably clear...etc. (W3C, 2006) [14]. More and more format are defined based on XML, such as RSS[15], XHTML[16] and OOXML[17].



3 System Design and Implementation

3.1 Overview

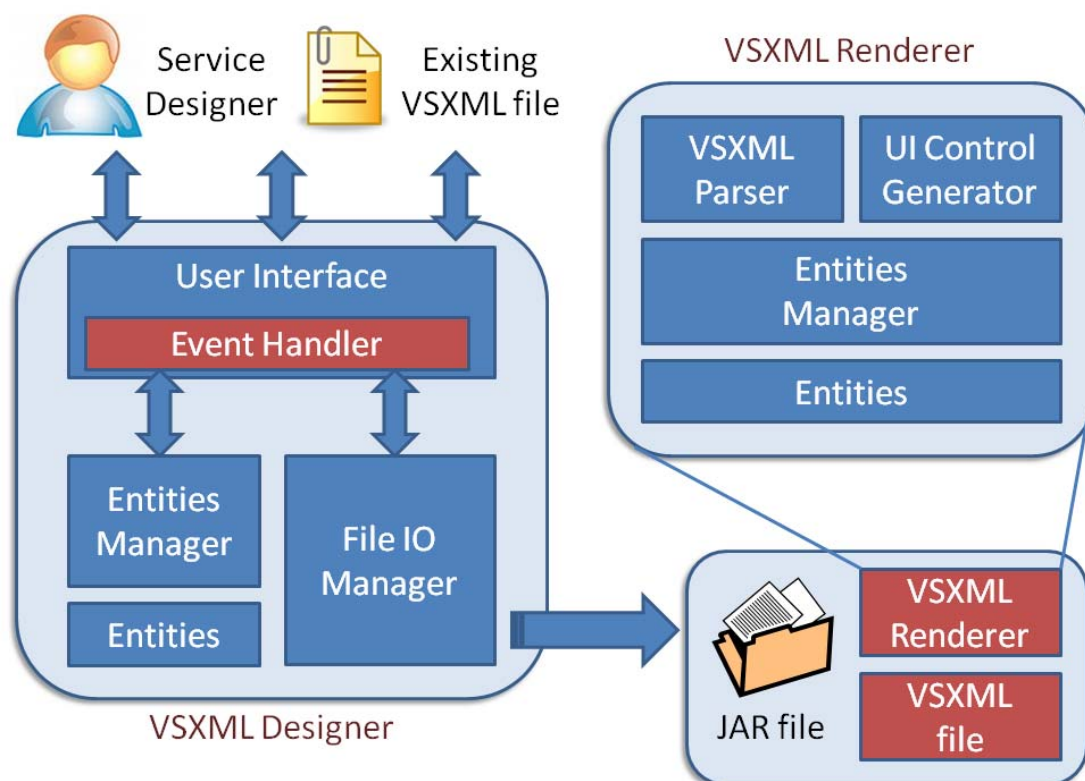


Figure 3-1 System Architecture

The system architecture is shown in Figure 3-1. We divide our system into three parts: VSXML, VSXML Designer and VSXML Renderer. VSXML Designer provides a friendly user interface for service designer designing new service in VSXML or transforms existing voice service into VSXML file. The JAR file generated from VSXML Designer contains not only VSXML file but also VSXML renderer, which renders text description to visual controls on the screen. The detailed descriptions of each part are introduced as followings.

3.2 VSXML

3.2.1 Analysis of Voice Service

Currently, there is no standardized voice service description. In order to design a standard format to fit in with most voice service, we collect several voice service samples from different business such as hospital、bank、school and travel agency. After analyzing several companies' voice services, we found out the similar characteristics in these services:

1. A voice service can be seen as a step-by-step flow. In each step, users need to listen for a short period, and it usually requires users to make a selection in order to enter the next step. We indicate the step as a Question Unit or a Question. In other words, a typical voice service is composed of many Question Units.

2. Normally, users response to the Question Units consists of traditional phone buttons, including numbers (0-9), # and *.

3. There are selections become meaningless after the voice service is visualized. For example, we notice the choice "Repeat" appears in many steps during a voice conversation. However, it is meaningless while users making selections through a visualized user interface.

4. The users' inputs to each Question Unit can be separated into two categories. One is to select a choice from a menu-driven interface. The other requires users to input information with phone buttons as indicated in the characteristic 2.

3.2.2 Definition

According to the above analyses, we can define the VSXML file as a XML formatted descriptions which document the voice services. A VSXML format

example is shown in Figure 3-2. In the diagram, each element and attribute is defined as following:

```
<?xml version="1.0" encoding="utf-8"?>
<Project programName="" fileName="" phoneNumber="">
  <Question number="" type="" pause="" description="" endchar=""
    limit="" directNextQuestion="">
    <Choice description="" value="" />
    <Choice description="" value="" />
    <NextQuestion number="">
      <Condition number="" value="">
      <Condition number="" value="">
    </NextQuestion>
  </Question>
</Project>
```

Figure 3-2 Voice service description format of VSXML

- Project := Represents the VSXML project; it contains many Question elements as listed below.
 - programName := Program name will be displayed after installed at mobile phones.
 - fileName := JAR file name is packed by the VSXML Renderer.
 - phoneNumber := phone number of the voice services.
- Question := Question Unit, this element must be included in the Project element.
 - number := Question Identification, we implement it as GUID (Globally Unique Identifier) in C#.
 - type := There are 3 different types of questions, two of them are described according to the characteristic 4 as mentioned above.
 - * SingleChoice := This Question requests users to select a choice from the Choice element in order to enter the next Question.
 - * Text := This Question requests users to input corresponding

information, such as extension number or bank account.

- * DescriptionOnly := This Question is a statement and does not require users' inputs or selections. For example, the welcome banner belongs to this type.
- pause := A Boolean value. This value is usually set in TRUE. It means a Pause tone should be inserted into the beginning of a dial-up sequence. In other words, it will add a short waiting period between the previous Question and the current Question. Mistakes may happen if the dial-up sequence of two consecutive Questions are too close.
- description := Description of the Question (not included in the selections)
- endchar := During the voice services, there are Questions, especially while entering Text data, that require users to input * or # for confirmation. However, after the voice services are visualized, service provider can ignore these characters.
- limit := Characters count limitation while the Question type is in Text.
- directNextQuestion := Direct the next Question number. If this attribute is empty, it means the voice service will be terminated after this Question. The attribute exists only if there is no Choice element in the Question element.
- Choice := Represents the selections of the Question. If the type of this Question is SingleChoice, the element must be included in the Question element.
 - description := Description of the selection.
 - value := Corresponding buttons of the selection. This attribute must be composed by numbers (0-9), *, or #.
- NextQuestion := Indicates the next Question of the current one and its conditions.

The element must be included in the Question element. After the user answers the current Question, it should scan all the NextQuestion elements that included in this Question and check if any NextQuestion element and its condition match users' selection.

- number := The next Question number of the current Question, if all conditions in the NextQuestion element are satisfied.

- Condition := The element must be included in the NextQuestion element. A Condition element represents a name-value pair - Question number and users' input of the Question. If all conditions in the NextQuestion element are satisfied, the NextQuestion number will become the number indicated in the Next Question.

- number := Question number
- value := Users input of the Question.

3.2.3 Example

```
<Project programName="NCTU" fileName="NCTU" phoneNumber="88635712121">
  <Question number="1" type="DescriptionOnly" pause="True"
    description="Welcome to National Chiao-Tung University"
    endchar="" limit="0" directNextQuestion="2">
  </Question>
  <Question number="2" type="SingleChoice" pause="True"
    description="Please choose a service"
    endchar="" limit="0" directNextQuestion="">
    <Choice description="Dial Extension Number" value="" />
    <Choice description="Check University Address" value="6" />
    <Choice description="Operator Service" value="9" />
    <NextQuestion number="3">
      <Condition number="2" value="" />
    </NextQuestion>
  </Question>
  <Question number="3" type="Text" pause="True"
    description="Please input extension number directly"
    endchar="" limit="5" directNextQuestion="">
  </Question>
</Project>
```

Figure 3-3 NCTU voice service protocol of VSXML

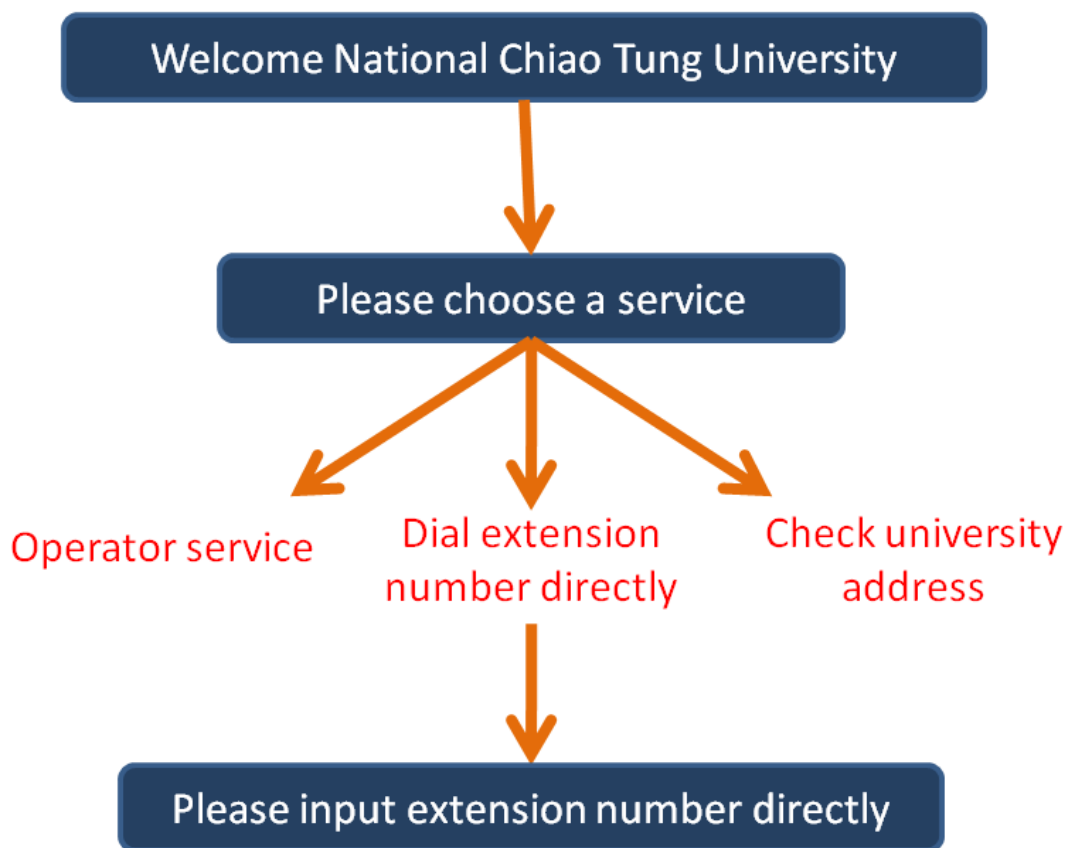


Figure 3-4 Flow diagram of NCTU voice service

Figure 3-3 provides an example of VSXML project file for the NCTU voice service. It consists of 3 Question Units. The type of Question 1 (Q1) is DescriptionOnly. It means the Question does not require users' inputs or selections. It only displays the welcome banner of NCTU. In addition, the Question contains no NextQuestion element, thus the next Question becomes Question 2 because of the attribute of the directNextQuestion in Q1. Question 2 (Q2) is a SingleChoice type Question. There are three choices in it: Dial Extension number, Check University Address and Operator Service. We notice there is a NextQuestion element in Q2. The value of the attribute Number is 3. Moreover, the number-value pair in the Condition is "2-empty". It indicates if user's selection in Q2 is "Dial Extension Number" with empty value, then Question 3 will become the next Question. Otherwise, the service

will be terminated if user selects either “Check University Address”, having number-value pair “2-9”, or “Operator Service”, having number-value pair “2-6”. It does not match the Condition “2-empty”. The type of the Question 3 (Q3) is Text. It needs user’s input, the extension number, as the description indicated. The attribute directNextQuestion of Q3 is empty. Thus the whole voice service will be ended after this Question. Figure 3-4 is the corresponding flow diagram of NCTU voice service.

3.3 VSXML Designer

Although the regular XML editors can provide the abilities of editing the VSXML files, in here, we support a user friendly, localized editor, i.e. the VSXML Designer to construct visualized VSXML files. It is implemented with C# and Microsoft Visual Studio 2005 because of completed framework and high productivity of Visual Studio 2005.

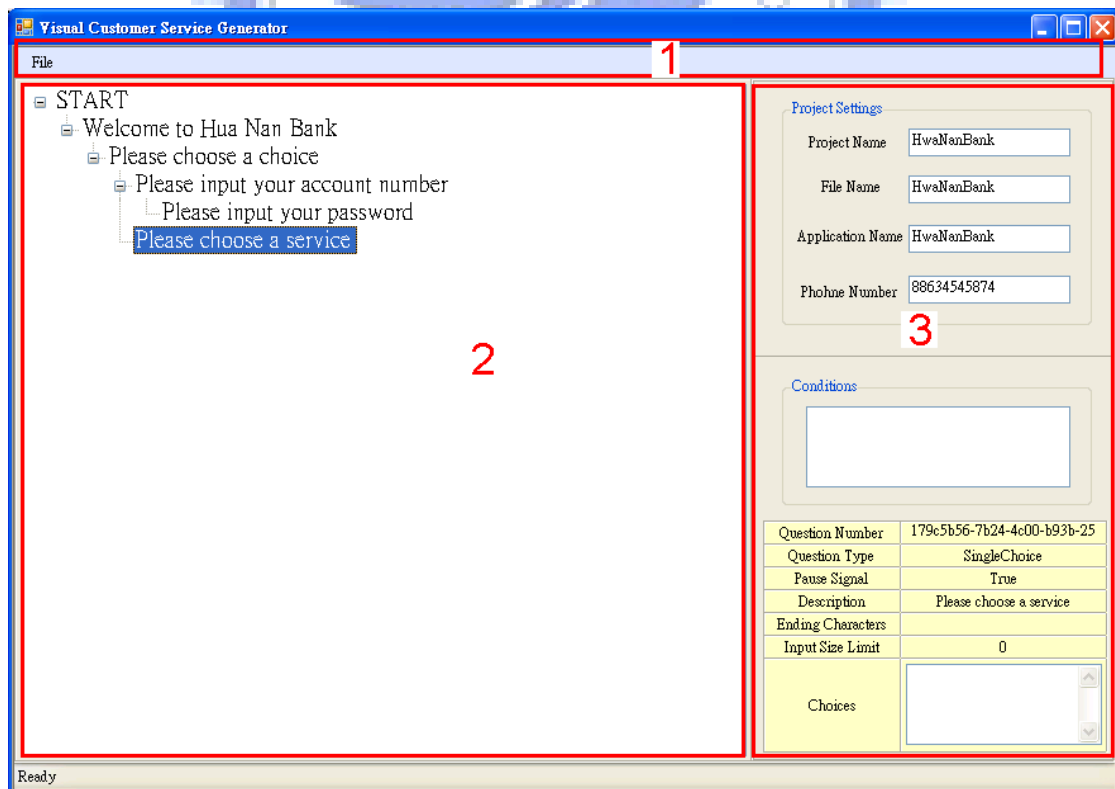


Figure 3-5 Screenshot of VSXML Designer

3.3.1 User Interface

The user interface of VSXML Designer application contains three components: MenuBar, QuestionTree, and Information Area. The screenshot of the VSXML Designer is depicted in Figure 3-5. The detailed descriptions of the components are illustrated as followings:

➤ MenuBar (1)

In the diagram, the MenuBar has one module, FileIOModule. It provides functions such as storing or loading VSXML project files as well as packing VSXML project files and the VSXML Renderer into a JAR file.

➤ QuestionTree (2)

In VSXML file, the order of the Questions is decided by the NextQuestion element. It is uni-direction; it indicates if Question B is the next Question of Question A, then Question A will not be the next Question of Question B. Otherwise, the application can be trapped in a loop. Hence, we construct a tree structure, as shown in Figure 4-4 (2), by connecting Questions with ordered numbers. Each node in the tree represents a Question Unit. A voice service designer can add, insert, modify and delete a tree node. Moreover, we use a depth-first traversal algorithm here to build the Question tree when the user opens a VSXML file. Listing 3-1 shows the implementation of the algorithm:

```
void BuildTree(Question q, TreeNode parent){  
    TreeNode node = new TreeNode(q.Description);  
    node.Tag = q;  
    parent.Nodes.Add(node);  
}
```

```

foreach(NextQuestion nq in q.NextQuestions){

    if (nq.Number == "" || nq.Number == null || nq.Number == "0")

    { } //Do nothing. It means there is no NextQuestion

    else{

        BuildTree(FindQuestion(nq.Number, questions), node);

    }

}

}
}

```

Listing 3-1 Implementation of Depth-first traversal algorithm

➤ **Information Area (3)**

A voice service designer needs to utilize the VSXML Designer to fill out the Project Information, assign values to attributes of the project, e.g., Program Name and Phone Number, as indicated in Figure 3-2. The designer also needs to enter the property grid of Question Units as shown in the diagram (3).

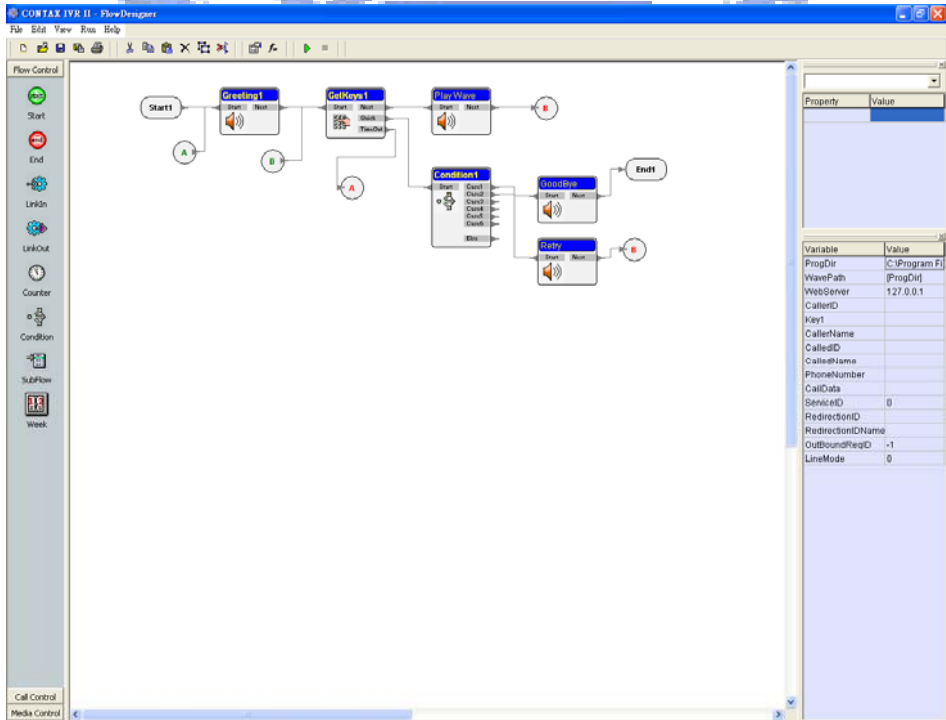


Figure 3-6 Screenshot of CONTAX IVR II

The primary part of user interface of VSXML Designer is the Question Tree which illustrates relationships of each Question Units, and the design concept of remaining part is east-to-understand. Figure 3-6 shows a demo version of voice service flow designer called CONTAX IVR II[18], developed by CALLURL Inc... In the middle of figure, a circuit-like drawing represents a voice service flow “decide which phone button that the user presses”. Though CONTAX IVR II is similar to VSXML Designer, the functions provided by it is more complicated. All controls on the left toolbox are about 30. Originally, we hoped users could complete a VSXML file through dragging controls onto a blank canvas, just like CONTAX IVR II. However, according to the analysis of the design process, we realize that “design a user calling flow” is much simpler than “design an IVR system”. Therefore, we use intuitive tree structure to show relationships of each Question Units. Figure 3-7 depicts the voice service flow as same as Figure 3-6 designed in VSXML Designer.

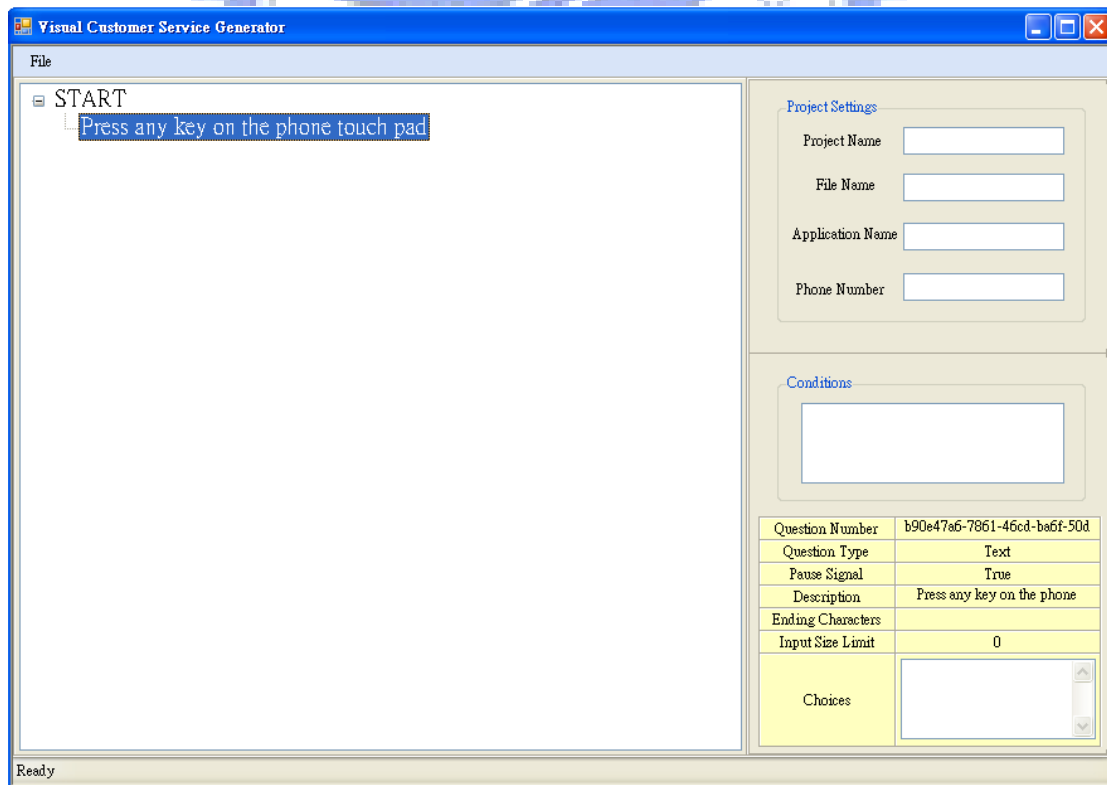


Figure 3-7 Same voice flow as figure 3-6 designed in VSXML Designer

3.3.2 Modules

The relationship between user interface and modules is depicted as Figure 3-1. According to the diagram, user interface bridged users and kernel modules of VSXML Designer, like Entities and File IO Manager. Besides, VSXML Designer is an Event-Driven application. In other words, the bottom modules will be executed after an event was triggered. The detailed descriptions of these modules are illustrated as followings:

➤ Entities

Module Entities is composed of many basic entity classes such as class Question, class Choice and class NextQuestion. The class diagram of these classes is shown as Figure 3-8. Two main functionalities of module Entities are keeping the value of each element and becoming parameters between event handler and File IO Manager.

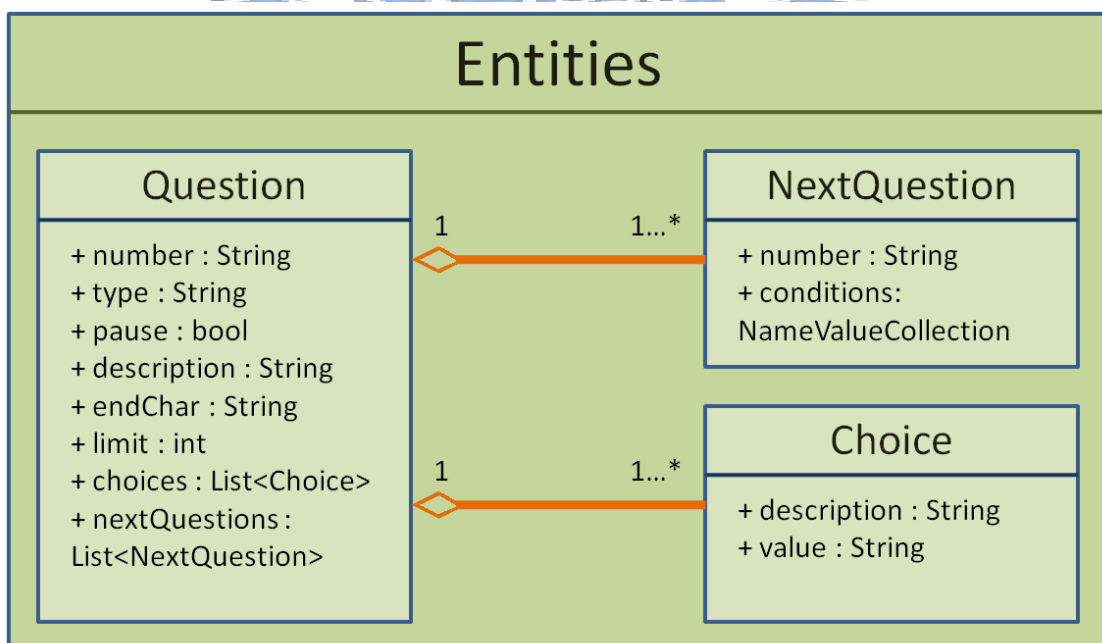


Figure 3-8 Class diagram of module Entities

➤ File IO Manager

File IO Manager deals with events related to file input and output. There are three functions in this class: SaveVSXML(), LoadVSXML() and BuildJAR().

The detailed descriptions of these functions are illustrated as followings:

■ Save/Load VSXML

In C# programs, the System.Xml.dll(dynamic link library) file must be referenced if it has to access a XML file. The following lines of code are part of function LoadVSXML():

```
XmlReader reader = xmlTextReader.Create(path);  
//path: absolute path of VSXML file  
while(reader.Read()) {  
    //start reading VSXML file until end of file  
    if(reader.NodeType == xmlNodeType.Element){  
        if(reader.Name.Equals("Project")) {  
            // Retrieve information of Element Project  
        }  
        else if(reader.Name.Equals("Question")) {  
            // Retrieve information of Element Question  
        }  
        //Other codes  
    }  
}
```

Listing 3-2 Loading VSXML file

The code of SaveVSXML() is very intuitive as well. It writes elements and attributes to VSXML file orderly:

```
XmlTextWriter writer = new XmlTextWriter(path);  
  
//path:absolute path of VSXML file  
  
loWriter.Formatting = Formatting.Indented;  
  
loWriter.WriteStartDocument();  
  
loWriter.WriteStartElement("Project");  
  
loWriter.WriteAttributeString("programName", programName);  
loWriter.WriteAttributeString("fileName", fileName);  
loWriter.WriteAttributeString("phoneNumber", phoneNumber);  
  
//Writing Question elements to VSXML file  
  
loWriter.WriteEndElement(); // Close the corresponding element
```

Listing 3-3 Saving VSXML file

■ Build JAR

Function BuildJAR() packs VSXML file and VSXML Renderer into a JAR file.

Listing 3-4 is part of codes of the function. To achieve that, the main idea is to call execution file *jar* externally through the Process object under namespace

System.Diagnostics. The Process object can “Provides access to local and remote processes and enables you to start and stop local system processes.”[摘自 MSDN]

Method write() of the Process object simulates condition of entering commands in Windows console mode, hence we can execute *jar.exe* with parameters through it as shown in Listing 3-4.

```
Process p = new Process();
```

```
p.StartInfo.FileName = "cmd.exe";  
  
p.StartInfo.CreateNoWindow = true; // Set no pop-up window  
  
p.Start(); // Start this process  
  
p.StandardInput.WriteLine(/* Command for packing jar file */);  
  
p.StandardInput.WriteLine("exit");  
  
p.Close();
```

Listing 3-4 Building JAR file

3.4 VSXML Renderer

VSXML Renderer is a program rendering the VSXML file displayed on mobile phones. The program is implemented under Java Micro Edition environment. The VSXML Designer packs the VSXML project file and the VSXML Renderer into a JAR file, later installed it in mobile phones. The Renderer executes the VSXML project file directly. The Renderer using an open source XML parser for Java Micro Edition device called Xparse-J[19] to decompose the VSXML file and transform the Question elements into a Question object array. The detailed information of implementation is described below.

3.4.1 Rendering VSXML

While the mobile phone, or the client application executing the VSXML Renderer, the first Question Unit will pop up. Figure 3-9 illustrates VSXML Renderer renders two different types of Questions. According to the types of the Questions, the Renderer interprets them differently. If the type is SingleChoice, the Renderer generates a Menu Driven form. If the type is Text, the Renderer generates an input box including the count limitation of the input characters. A dial-up sequence is kept

inside the Renderer to record the attribute values of the Questions that user entered orderly. The sequence will not be activated until the directNextQuestion of the current Question is empty.



Figure 3-9 Rendering different types of Questions by VSXML Renderer

3.4.2 Dial-Up Sequence

Assuming, a VSXML project file including n Question Units, we can express the dial-up sequence in a normal expression as followings:

$$\text{Dial-up Sequence } D = \text{Phone number} + \sum_{i=1}^n [\text{pause of } Q_i][\text{answer of } Q_i][\text{endchar of } Q_i]$$

For example, the VSXML project file of the NCTU voice service, a user is trying to reach the extension 56631 under the NCTU main phone directory, i.e., +886-3-571-2121. According to the normal expression, the dial-up sequence is presented as: “88635712121p56631”. This dial-up sequence will be automatically

activated, and the voice connection will be established to the extension directly.



4 Scenarios Demonstration

4.1 System Flow Scenarios

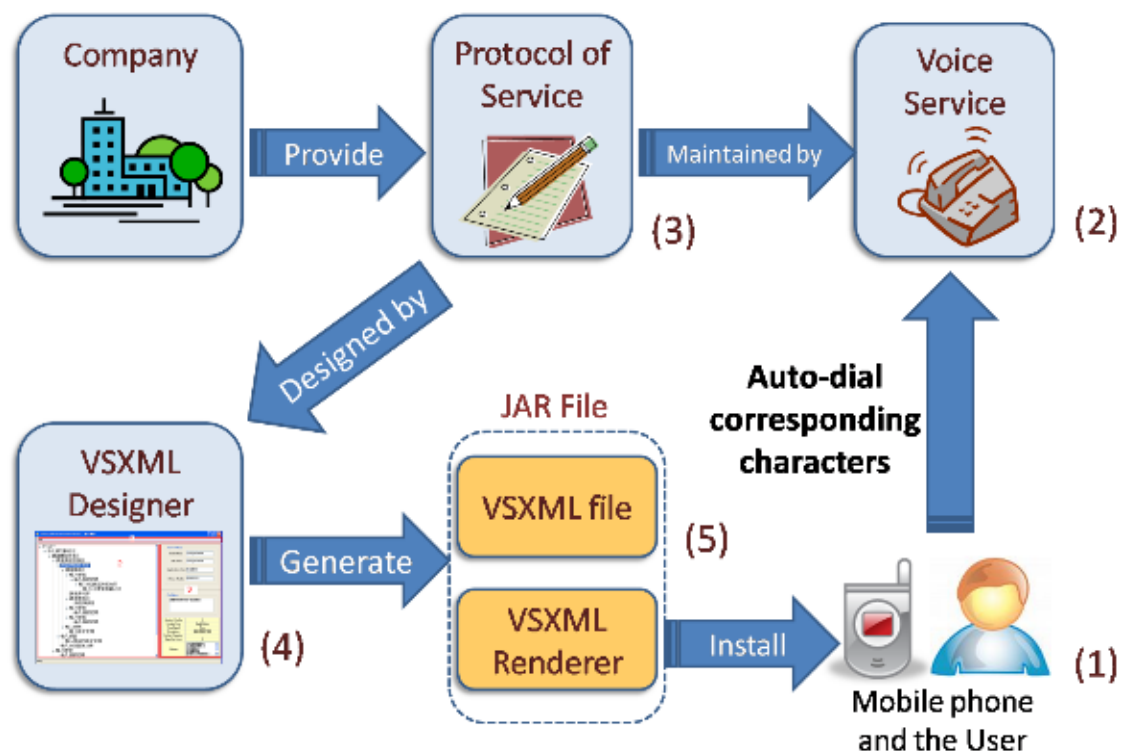


Figure 4-1 System Flow Scenarios Diagram

The scenarios of the system flow diagram are shown in Figure 4-1. In the diagram, according to the communication protocols between mobile phones (1) and the voice services system (2), initially, the service designer uses the toolkit, VSXML Designer (4) to implement the voice service flow scripts (3). When the process is completed, the VSXML Designer transforms the scripts into a VSXML format file (in 5). The JAR Builder Module in the Designer will pack the client display application, VSXML Renderer (in 5) implemented in Java Micro Edition environment, as well as the VSXML file into a JAR file (5) as depicted in the diagram.

Once the mobile device users install and execute the JAR file, the VSXML Renderer will generate the corresponding controls according to the VSXML file. Later, the executed program records users' selections as a dial-up sequence. These selections are displayed, presented as a visualized interface at the mobile phone. At the end, the dialing-sequence will be executed by the program, and the user can hold on and wait for the responses as requested.

4.2 Creating VSXML File

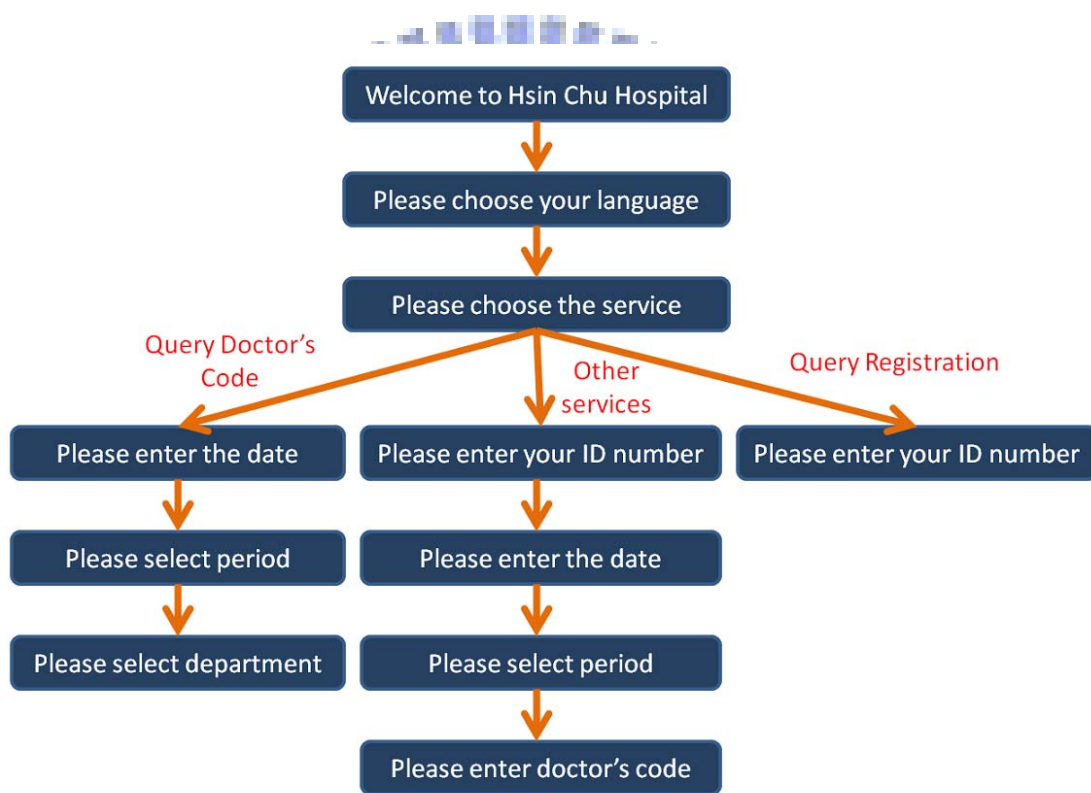


Figure 4-2 Voice Service Flow of Hsinchu Hospital

In this section, we demonstrate how to design a VSXML project by VSXML Designer. Before executing the application, we have to set up a real voice service script which is provided by voice service provider as a sample. Figure 4-2 depicted the voice service flow of Hsinchu Hospital in Taiwan. We can tell from this figure that there are 11 Question Units in voice service of Hsinchu Hospital, and the distribution

of Question Type is listed in Table 4-1.

Types	DescriptionOnly	SingleChoice	Text
Numbers	1	5	5

Table 4-1 Distribution of Question Type in Voice Service of Hsinchu Hospital

A new VSXML project will be presented after executing VSXML Designer. First of all, the root of Question Tree is “START”, which means the entry point of the voice service. Right click tree node and a context menu will pop up. Four functions can be chose in context menu: Add Question · Insert Question · Modify and Delete. Figure 4-3 illustrates difference between Add Question and Insert Question. Add Question adds a new Question to the child of selected Question. On the other hand, Insert Question inserts a new Question to the parent of selected Question.

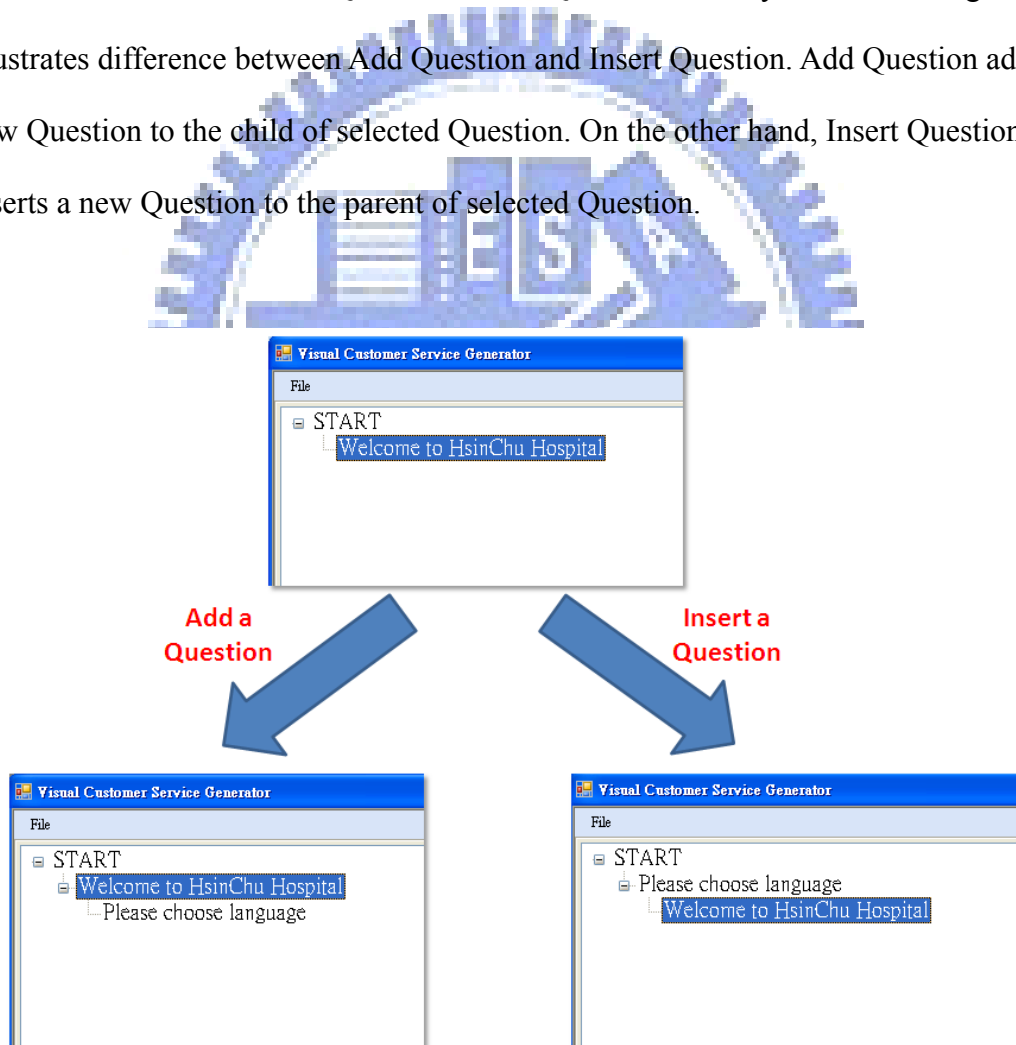


Figure 4-3 Add and Insert a Question

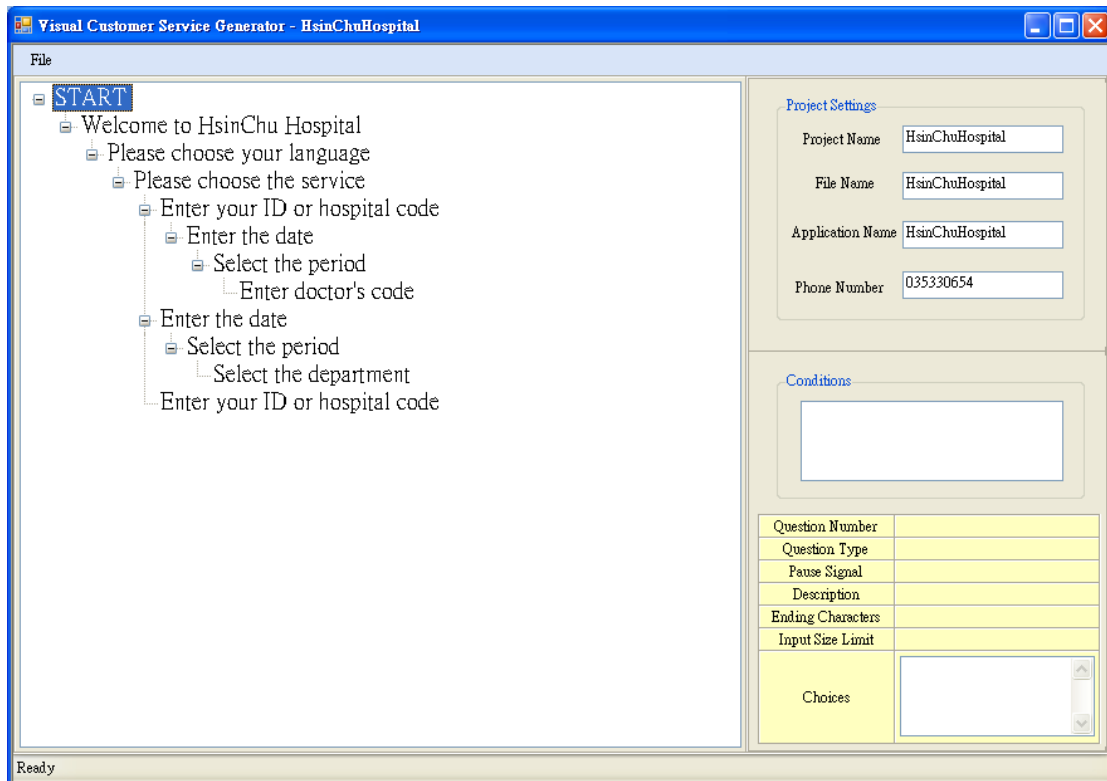


Figure 4-4 Screenshot of completed VSXML file of Hsinchu Hospital

Furthermore, function Modify and Delete do modification and deletion to the selected tree node (Question). Through the functions above-mentioned, we can finish entire voice service design easily and service flow will be easy to understand because of the Question Tree as shown in Figure 4-4. Finally, clicking “Build JAR” button in Menu Bar will create a JAR file after filling out the required information of this VSXML project.

4.3 Rendering VSXML File

Mobile device users can get JAR file which is composed of VSXML file and VSXML Renderer via Bluetooth or Over The Air (OTA) technique. The two mobile devices we used to do experiment are Motorola E1070 and Nokia 6120 Classic. As shown in figure 4-5, the application name “Hsinchu Hospital” is named after the Program Name field set in VSXML Designer.



Figure 4-5 Application Name on Motorola E1070 and Nokia 6210 Classic

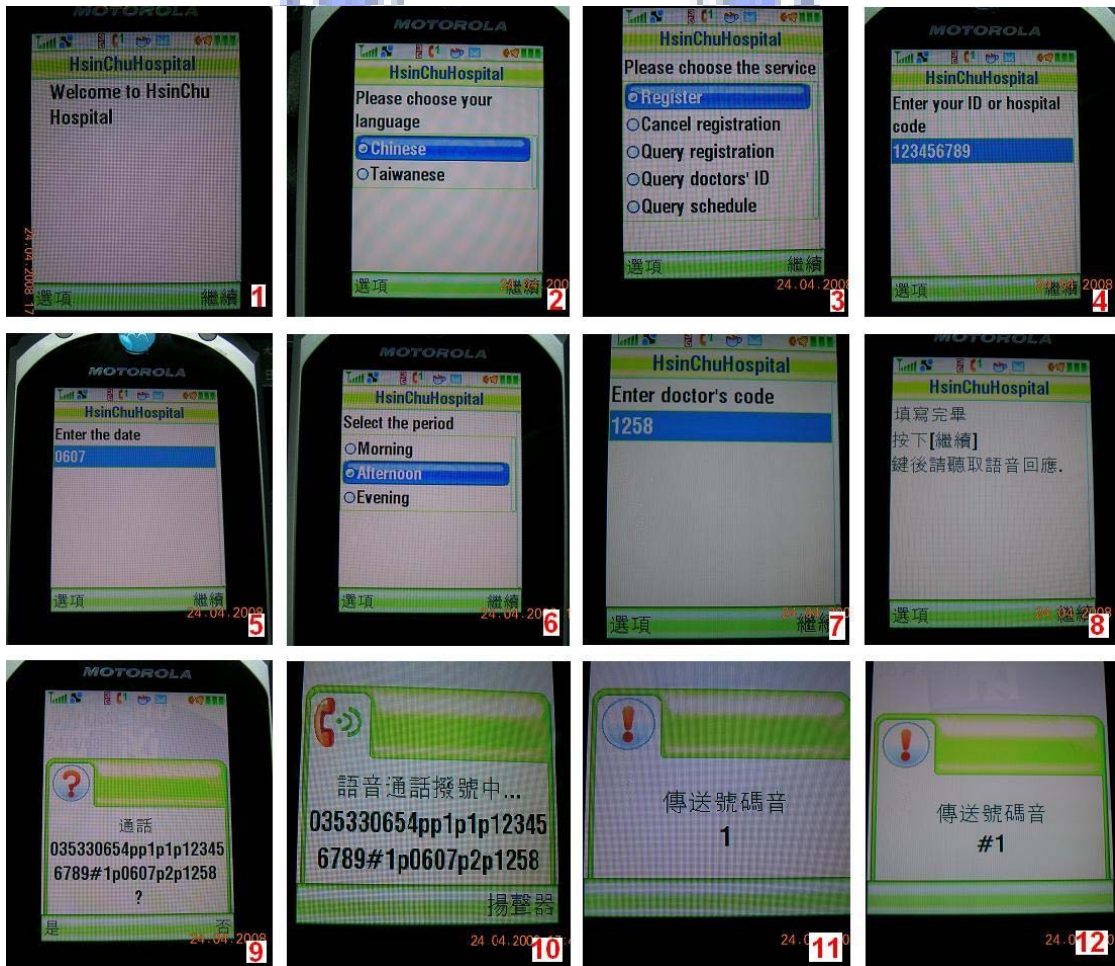


Figure 4-6 Process of registering for Hsinchu Hospital with VXML Renderer on Motorola E1070

Assume we are going to register for hospital service of Hsinchu Hospital; we enter the requested information and choose corresponding choice according to description of each Question Unit, then press “Continue” button until the end of flow. Before dialing, VSXML Renderer will remind users that they still have to listen to the voice service after pressing “Dial” button. VSXML Renderer invokes built-in dialing program of device and dials corresponding dial-up sequence automatically. A dialog will pop up to ask user to confirm the dialing. Figure 4-6 shows whole process of the user registering for Hsinchu Hospital on Motorola E1070.



5 Results

5.1 Visualized Voice Service

The comparison between traditional (nonvisualized) voice services and visualized voice services is shown in Table 5-1. In terms of users, the most important achievement of the visualized voice services is the users do not need to listen to the long instructions as well as holding time. It can reduce the phone bill dramatically. It also decreases the probability of pressing the wrong button. The users can read the lines based on their own tempo rather than catch up the speed of the voice services. For mobile phone users, they do not need to take the mobile phone away from ears while pressing the corresponding buttons requested by the service.

In addition, from the service provider's point of view, the phone lines occupancy will be decreased. Consequently, the customer satisfaction will be enhanced. It is the goal that enterprises prefer to achieve.

Table 5-1 Comparison between nonvisualized and visualized voice services (Relatively)

	Nonvisualized Voice Service	Visualized Voice Service
Presentation	Voice	Selections on the screen
Interactive period	Long	Short
Cost	High	Low
Line availability	Low	High
Program installed at client side	No	Yes

5.2 VSXML

At present, we are coordinating and cooperating with three companies, Hwa Nan Bank (A), Chang Hwa Bank (B), and Hsinchu Hospital (C), in Taiwan, to establish the visualized voice services via VSXML. The content of VSXML file of company A and company B are both presented in Appendix A. Furthermore, the VSXML technologies have been applied and validated to Motorola E1070, Nokia 6120 Classic mobile phones and the Nokia Series 60 3rd Edition Emulator as well.

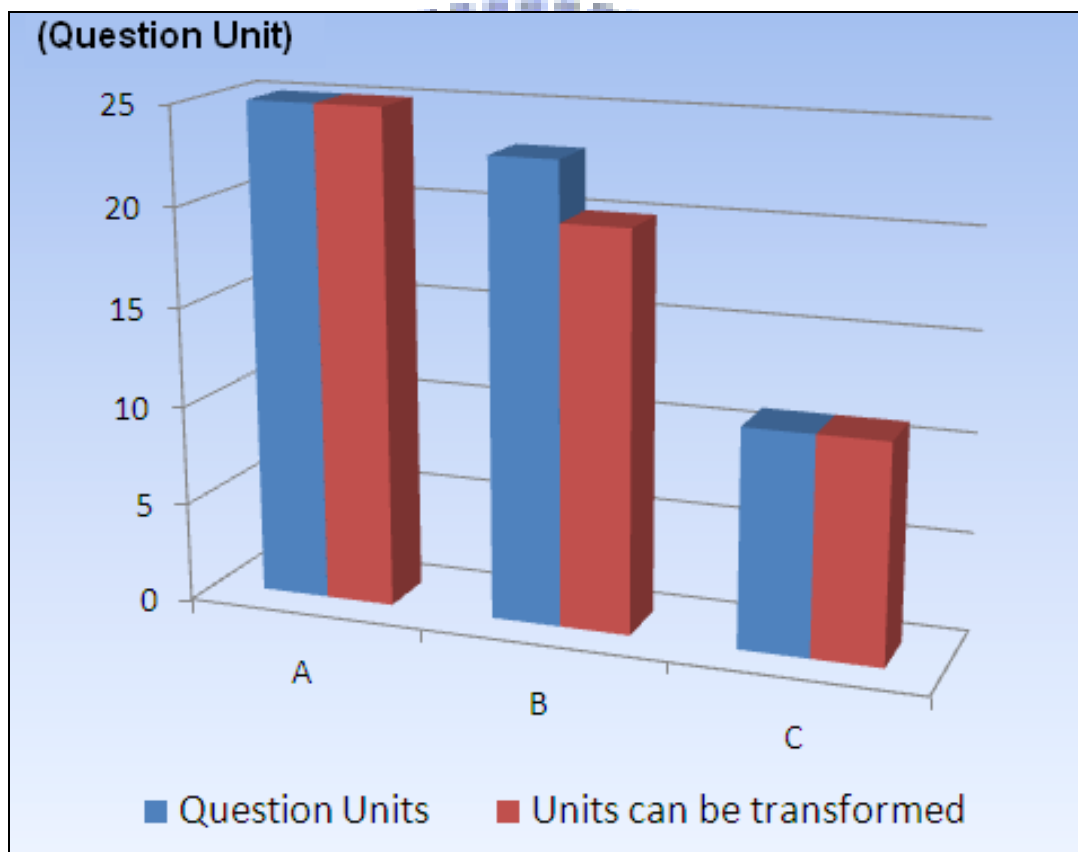


Figure 5-1 Capabilities of VSXML

Figure 5-1 illustrates the capabilities of converting the traditional voice services into visualized ones. In Company A, we defined 25 Question Units. Similarly, in Company C, 11 Question Units are identified. In these two companies, the Question Units can be visualized via VSXML completely. However, in Company B, there are

around 23 Question Units declared, among them, only 20, or 87%, Question Units are visualizable. The rest 13% cannot be converted due to database access requirements during the voice services. For instance, there is a Question Q1 during the flow of voice service of Company B asks for user's identification number. While IVR gets the identification number, it will query database to identify who the caller is. After that, the voice service proceeds with next Question Q2. Therefore, in Company B, the voice services cannot be transformed to VXML completely; customers need to listen to the traditional services after Q1.



6 Future Works and Conclusion

6.1 Future Works

6.1.1 Modification of VSXML

In the near future, we plan to extend the VSXML technologies to be capable of including databases accessing during the voice services. Currently, several solutions have been integrated on mobile devices: First, RMS (Record Management System) is a technology implemented in mobile devices to store records. Different MIDlet can access the same records through RMS. The main concept is similar as the well-known database model. Therefore, a small database can be created in mobile devices, and program developers can access the database by calling J2ME APIs. Secondly, the mobile devices can query database through mobile networks, i.e., GPRS and 3G (3rd Generation) mobile phone technologies. The main advantage of utilizing remote database access is that we do not need to keep a copy of data in mobile devices. Multiple copies of data can induce the data synchronization problems. On the other hand, if a mobile user does not apply for mobile network services, the second solution will not be achievable. At last, the data can be contained in an external parameter file or a manifest file downloadable into the mobile devices.

In our system, each solution presented above has problems. Although RMS is a common embedded mobile database, it can be integrated into our system. Unfortunately, we have to introduce another module to handle database interactions. In addition, the communications among modules can induce complexities and

complications. The RMS database cannot store personal data to protect privacy. The second approach can be implemented relatively easy. However, a corresponding web server ought to handle database interfaces, as well as the mobile devices must include the connectivity to a mobile network. Finally, attaching a downloadable external file into a mobile device seems more intuitive. But the solution can violate privacy and produce security issues. According to these problems, it is difficult to find a suitable implementation to achieve the database accessing between two Question Units in our system.

6.1.2 Compatibility of Mobile Devices

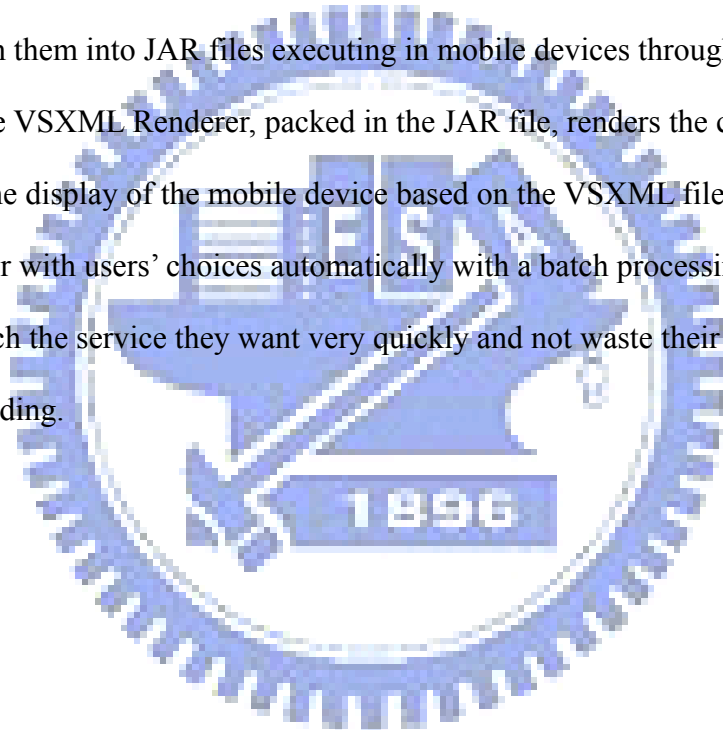
During the regression test, we realized the dial-up sequence may not be applicable to all mobile devices although the devices are compatible with JAVA Micro Edition MIDP 2.0(JSR118). After we installed the VSXML Renderer component, it functions properly on Motorola E1070 model. However, it fails on SonyEricsson Z800i model. During the service execution, the SonyEricsson Z800i model operated normally until the pause signal “p” dialed at end. Therefore, additional verifications and tests on different models of mobile devices are necessary.

6.1.3 Cooperation with CTI Companies

Currently, we continue searching for further cooperation with enterprises as well as the CTI (Computer Telephony Integration) companies to verify our design and system. There is no common standard for voice service formats or regulation. In the near future, we anticipate to enabling the voice service created by the generator of CTI companies being transformed into the VSXML formats simultaneously. In other words, the voice service providers can distribute our VSXML solutions in mobile devices industry.

6.2 Conclusion

The 24-hour voice service plays an essential role in customer relationship management. No matter how popular the Internet is, the phone line is still a faster and more convenient choice while needed. After analyzing several voice services, we design and develop VSXML technologies, a XML formatted standard encapsulating the contents as well as the description languages of the voice services. Furthermore, the voice service providers can easily design and convert the original voice services and transform them into JAR files executing in mobile devices through the VSXML Designer. The VSXML Renderer, packed in the JAR file, renders the corresponding controls on the display of the mobile device based on the VSXML file and dials the phone number with users' choices automatically with a batch processing. Therefore users can catch the service they want very quickly and not waste their time waiting for the speech ending.



References

- [1] Sun Chung Yo and Chueh Shih Nan, “Automatic Dialing-up System For Communication Apparatus”, Taiwan Patent 200714012 , Jan. 2006.
- [2] Schenker, L, “Pushbutton Calling with a Two-Group Voice-Frequency Code”, The Bell system technical journal, 39 (1): 235-255, ISSN 0005-8580, 1960.
- [3] Wikipedia, Call Centre, Retrieved on April 8, 2008, http://en.wikipedia.org/wiki/Call_centre
- [4] Jonathan Allen, M. Sharon Hunnicutt, and Dennis Klatt, “From Text to Speech: The MITalk system”. Cambridge University Press: 1987, ISBN 0521306418.
- [5] Michael B. Crane and Neil W. Sullivan, “Interactive computerized communications systems with voice input and output”, U.S. Patent 4866756, Apr. 1988.
- [6] Goode, B., “Voice over Internet protocol (VoIP)”, Proceedings of the IEEE, Volume 90, Issue 9, Sept. 2002, Page(s):1495 - 1517.
- [7] W3C, Voice Extensible Markup Language (VoiceXML) Version 2.0, Retrieved on April 8, 2008, <http://www.w3.org/TR/voicexml20/>
- [8] Official site of Asterisk, Retrieved on April 8, 2008, <http://www.asterisk.org/>
- [9] Official site of Yate, Retrieved on May 14, 2008, <http://yate.null.ro/pmwiki/>
- [10] Stephanos Androutsellis-Theotokis and Diomidis Spinellis. “A survey of peer-to-peer content distribution technologies.” ACM Computing Surveys, 36(4):335–371, December 2004.
- [11] Official site of Skype, Retrieved on May 14, 2008, <http://www.skype.com>
- [12] Official site of Windows Live Messenger, Retrieved on May 14, 2008,

<http://messenger.live.com>

[13] Neuberger, Marc J., Panttaja, Erin M., "Menu-based, speech actuated system with speak-ahead capability", U.S. Patent 20040153322, 2004.

[14] W3C, Extensible Markup Language (XML), retrieved on May 14, 2008,
<http://www.w3.org/XML/>

[15] RSS 2.0 Specification, retrieved on May 14, 2008,
<http://cyber.law.harvard.edu/rss/rss.html>

[16] W3C, XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition), retrieved on May 14, 2008, <http://www.w3.org/TR/xhtml1/>

[17] Open Office XML (OOXML), retrieved on May 14, 2008,
<http://xml.openoffice.org/general.html>

[18] CallURL Inc., CONTAX IVR II Demo Version, downloaded from
<http://www.callurl.net/IVRIIDEMO.asp>

[19] Xparse-J, downloaded from
<http://www.webreference.com/xml/tools/xparse-j.html>



Appendix A

Case A : Hwa Nan Bank

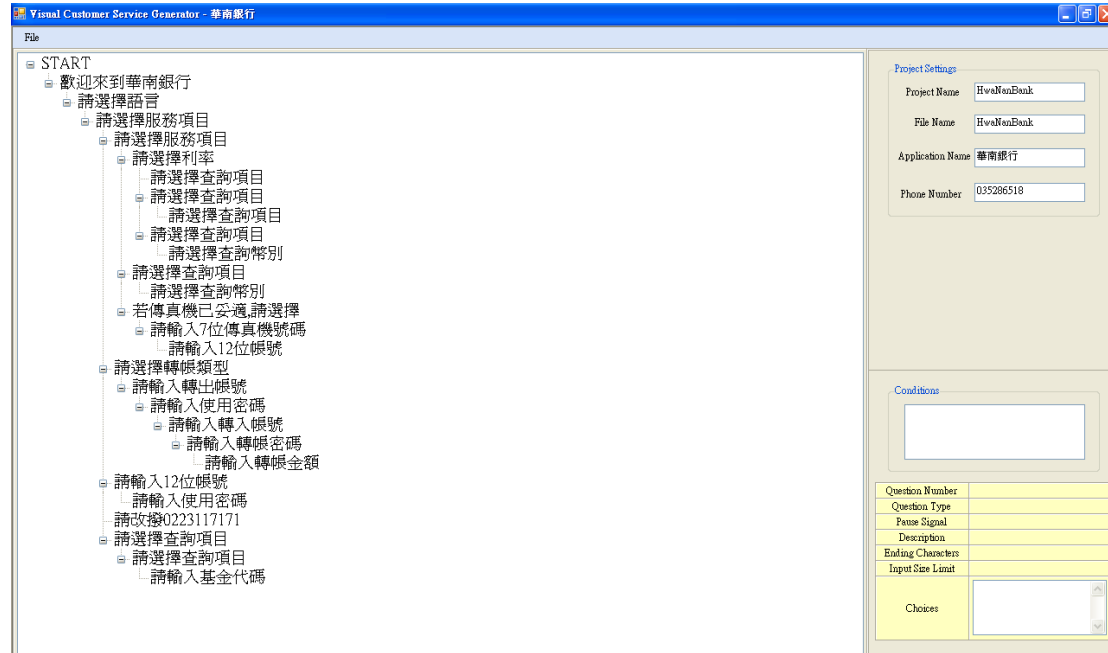


Figure Appendix- 1 Screenshot of completed VSXML file of Hwa Nan Bank

VSXML of Hwa Nan Bank

```
<Project programName="華南銀行" fileName="HwaNanBank" phoneNumber="035286518">
<Questions>
<Question number="1" type="DescriptionOnly" pause="False" description="歡迎來到華南銀行" endchar="" limit="0" directNextQuestion="2">
</Question>
<Question number="2" type="SingleChoice" pause="True" description="請選擇語言" endchar="" limit="0" directNextQuestion="3">
<Choice description="國語" value="1" />
<Choice description="台語" value="2" />
</Question>
<Question>
<Question number="3" type="SingleChoice" pause="True" description="請選擇服務項目" endchar="" limit="0" directNextQuestion="0">
<Choice description="台幣轉帳" value="1" />
<Choice description="餘額查詢" value="2" />
<Choice description="金庫卡掛失" value="3" />
<Choice description="預約轉帳" value="4" />
<Choice description="更改密碼" value="5" />
<Choice description="外匯轉帳" value="6" />
<Choice description="其它服務" value="7" />
<Choice description="理財服務" value="8" />
<Choice description="轉帳繳稅" value="9" />
<NextQuestion number="10">
<Condition number="3" value="7" />
</NextQuestion>
<NextQuestion number="4">
<Condition number="3" value="1" />
</NextQuestion>
<NextQuestion number="21">
<Condition number="3" value="2" />
</NextQuestion>
<NextQuestion number="23">
<Condition number="3" value="3" />
</NextQuestion>
<NextQuestion number="24">
<Condition number="3" value="8" />
</NextQuestion>
</Question>
</Questions>
```

```

<Question number="4" type="SingleChoice" pause="True" description="請選擇轉帳類型" endchar="" limit="0" directNextQuestion="5">
  <Choice description="轉入華銀帳號" value="1" />
  <Choice description="轉入跨行帳號" value="2" />
</Question>
<Question number="5" type="Text" pause="True" description="請輸入轉出帳號" endchar="#" limit="12" directNextQuestion="6">
</Question>
<Question number="6" type="Text" pause="True" description="請輸入使用密碼" endchar="" limit="4" directNextQuestion="7">
</Question>
<Question number="7" type="Text" pause="True" description="請輸入轉入帳號" endchar="###" limit="12" directNextQuestion="8">
</Question>
<Question number="8" type="Text" pause="True" description="請輸入轉帳密碼" endchar="" limit="4" directNextQuestion="9">
</Question>
<Question number="9" type="Text" pause="True" description="請輸入轉帳金額" endchar="#p1" limit="12" directNextQuestion="10">
</Question>
<Question number="10" type="SingleChoice" pause="True" description="請選擇服務項目" endchar="" limit="0" directNextQuestion="0">
  <Choice description="查詢利率" value="1" />
  <Choice description="匯率查詢" value="2" />
  <Choice description="即期外匯匯率表傳真" value="4" />
  <Choice description="交易明細表傳真" value="5" />
  <Choice description="各項業務代碼查詢" value="6" />
  <Choice description="查詢業務說明" value="7" />
  <NextQuestion number="11">
    <Condition number="10" value="1" />
  </NextQuestion>
  <NextQuestion number="17">
    <Condition number="10" value="2" />
  </NextQuestion>
  <NextQuestion number="18">
    <Condition number="10" value="4" />
  </NextQuestion>
</Question>
<Question number="11" type="SingleChoice" pause="True" description="請選擇利率" endchar="" limit="0" directNextQuestion="0">
  <Choice description="存摺存款利率" value="1" />
  <Choice description="定期存款利率" value="2" />
  <Choice description="放款基本利率" value="3" />
  <Choice description="外匯存款利率" value="4" />
  <NextQuestion number="12">
    <Condition number="11" value="1" />
  </NextQuestion>
  <NextQuestion number="13">
    <Condition number="11" value="2" />
  </NextQuestion>
  <NextQuestion number="15">
    <Condition number="11" value="4" />
  </NextQuestion>
</Question>
<Question number="12" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="0">
  <Choice description="活期存款利率" value="1" />
  <Choice description="活儲存款利率" value="2" />
</Question>
<Question number="13" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="14">
  <Choice description="固定利率" value="1" />
  <Choice description="機動利率" value="2" />
</Question>
<Question number="14" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="0">
  <Choice description="定期存款利率" value="1" />
  <Choice description="定儲存款利率" value="2" />
</Question>
<Question number="15" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="16">
  <Choice description="外匯活存利率" value="1" />
  <Choice description="外匯定存利率" value="2" />
</Question>
<Question number="16" type="SingleChoice" pause="False" description="請選擇查詢幣別" endchar="#" limit="0" directNextQuestion="0">
  <Choice description="美金" value="01" />
  <Choice description="港幣" value="02" />
  <Choice description="英鎊" value="03" />
  <Choice description="馬克" value="04" />
</Question>

```

```

<Question number="17" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="16">
  <Choice description="即期匯率" value="1" />
  <Choice description="遠期匯率" value="2" />
</Question>
<Question number="18" type="SingleChoice" pause="True" description="若傳真機已妥適,請選擇" endchar="" limit="0" directNextQuestion="19">
  <Choice description="傳真機已準備" value="1" />
</Question>
<Question number="19" type="Text" pause="False" description="請輸入7位傳真機號碼" endchar="1" limit="7" directNextQuestion="20">
</Question>
<Question number="20" type="Text" pause="False" description="請輸入12位帳號" endchar="#" limit="12" directNextQuestion="0">
</Question>
<Question number="21" type="Text" pause="False" description="請輸入12位帳號" endchar="#" limit="12" directNextQuestion="22">
</Question>
<Question number="22" type="Text" pause="False" description="請輸入使用密碼" endchar="" limit="4" directNextQuestion="0">
</Question>
<Question number="23" type="DescriptionOnly" pause="False" description="請改撥0223117171" endchar="" limit="0" directNextQuestion="0">
</Question>
<Question number="24" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="0">
  <Choice description="基金理財" value="1" />
  <Choice description="存款理財" value="2" />
  <NextQuestion number="25">
    <Condition number="24" value="1" />
  </NextQuestion>
</Question>
<Question number="25" type="SingleChoice" pause="True" description="請選擇查詢項目" endchar="" limit="0" directNextQuestion="0">
  <Choice description="基金淨值查詢" value="1" />
  <Choice description="基金代碼查詢" value="2" />
  <Choice description="個人基金資產查詢" value="3" />
  <NextQuestion number="26">
    <Condition number="25" value="1" />
  </NextQuestion>
</Question>
</Questions>
</Project>

```

Case B : Chang Hwa Bank

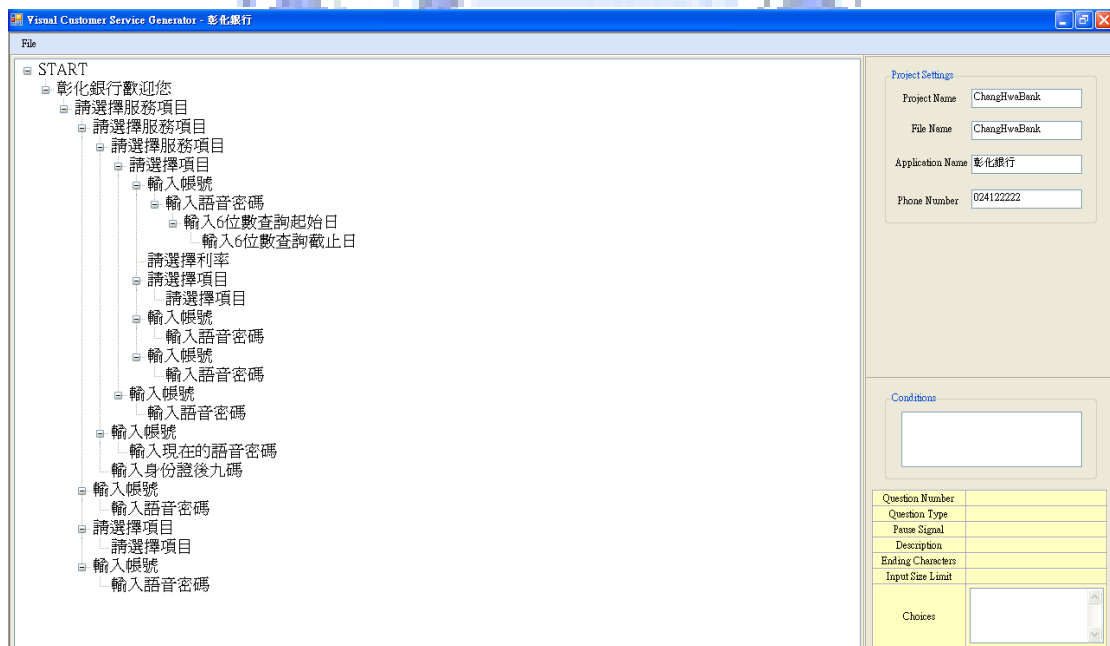


Figure Appendix- 2 Screenshot of completed VSXML file of Chang Hwa Bank

VSXML of Chang Hwa Bank

```
<Project programName="彰化銀行" fileName="ChangHwaBank" phoneNumber="024122222">
  <Questions>
    <Question number="1" type="DescriptionOnly" pause="False" description="彰化銀行歡迎您" endchar="" limit="0" directNextQuestion="2">
    </Question>
    <Question number="2" type="SingleChoice" pause="True" description="請選擇服務項目" endchar="" limit="0" directNextQuestion="0">
      <Choice description="信用卡業務" value="1" />
      <Choice description="基金業務" value="2" />
      <Choice description="銀行業務" value="3" />
      <Choice description="貸款業務" value="4" />
      <Choice description="外匯業務" value="5" />
      <Choice description="快速-存款餘額查詢" value="830" />
      <Choice description="快速-餘額不足明細查詢" value="839" />
      <Choice description="快速-匯率查詢" value="850" />
      <NextQuestion number="3">
        <Condition number="2" value="3" />
      </NextQuestion>
      <NextQuestion number="5">
        <Condition number="2" value="830" />
      </NextQuestion>
      <NextQuestion number="8">
        <Condition number="2" value="850" />
      </NextQuestion>
      <NextQuestion number="10">
        <Condition number="2" value="839" />
      </NextQuestion>
    </Question>
    <Question number="3" type="SingleChoice" pause="True" description="請選擇服務項目" endchar="" limit="0" directNextQuestion="0">
      <Choice description="查詢服務" value="1" />
      <Choice description="轉帳服務" value="2" />
      <Choice description="語音密碼變更" value="3" />
      <Choice description="繳付本行信用卡費" value="4" />
      <Choice description="繳交中華電信資費" value="5" />
      <Choice description="存摺掛失" value="7" />
      <Choice description="金融卡掛失" value="8" />
      <Choice description="轉接專人" value="9" />
      <NextQuestion number="4">
        <Condition number="3" value="1" />
      </NextQuestion>
      <NextQuestion number="19">
        <Condition number="3" value="3" />
      </NextQuestion>
      <NextQuestion number="21">
        <Condition number="3" value="9" />
      </NextQuestion>
    </Question>
    <Question number="4" type="SingleChoice" pause="True" description="請選擇服務項目" endchar="" limit="0" directNextQuestion="18">
      <Choice description="存款餘額查詢" value="1" />
      <Choice description="交易明細查詢" value="2" />
      <Choice description="利率查詢" value="3" />
      <Choice description="匯率查詢" value="4" />
      <Choice description="餘額不足明細查詢" value="5" />
      <Choice description="約定轉帳帳號之查詢" value="6" />
      <NextQuestion number="5">
        <Condition number="4" value="1" />
      </NextQuestion>
    </Question>
    <Question number="5" type="Text" pause="False" description="輸入帳號" endchar="#" limit="14" directNextQuestion="6">
    </Question>
    <Question number="6" type="Text" pause="False" description="輸入語音密碼" endchar="" limit="4" directNextQuestion="0">
    </Question>
    <Question number="7" type="SingleChoice" pause="True" description="請選擇利率" endchar="" limit="0" directNextQuestion="0">
      <Choice description="活期存款" value="1" />
      <Choice description="定期存款" value="2" />
      <Choice description="放款" value="3" />
      <Choice description="外匯存款" value="4" />
      <Choice description="外匯授信利率" value="5" />
    </Question>
    <Question number="8" type="SingleChoice" pause="True" description="請選擇項目" endchar="" limit="0" directNextQuestion="9">
      <Choice description="即期" value="1" />
      <Choice description="遠期" value="2" />
    </Question>
    <Question number="9" type="SingleChoice" pause="True" description="請選擇項目" endchar="" limit="0" directNextQuestion="0">
      <Choice description="單一匯率" value="1" />
      <Choice description="常用匯率" value="2" />
      <Choice description="全部匯率" value="3" />
    </Question>
  </Questions>
</Project>
```

```

<Question number="10" type="Text" pause="False" description="輸入帳號" endchar="#" limit="14" directNextQuestion="11">
</Question>
<Question number="11" type="Text" pause="False" description="輸入語音密碼" endchar="" limit="4" directNextQuestion="0">
</Question>
<Question number="12" type="Text" pause="False" description="輸入帳號" endchar="#" limit="14" directNextQuestion="13">
</Question>
<Question number="13" type="Text" pause="False" description="輸入語音密碼" endchar="" limit="4" directNextQuestion="0">
</Question>
<Question number="14" type="Text" pause="False" description="輸入帳號" endchar="#" limit="14" directNextQuestion="15">
</Question>
<Question number="15" type="Text" pause="False" description="輸入語音密碼" endchar="" limit="4" directNextQuestion="16">
</Question>
<Question number="16" type="Text" pause="False" description="輸入6位數查詢起始日" endchar="" limit="6" directNextQuestion="17">
</Question>
<Question number="17" type="Text" pause="False" description="輸入6位數查詢截止日" endchar="1" limit="6" directNextQuestion="0">
</Question>

<Question number="18" type="SingleChoice" pause="True" description="請選擇項目" endchar="" limit="0" directNextQuestion="0">
  <Choice description="查詢" value="1" />
  <Choice description="傳真" value="2" />
  <NextQuestion number="14">
    <Condition number="4" value="2" />
    <Condition number="18" value="1" />
  </NextQuestion>
  <NextQuestion number="7">
    <Condition number="4" value="3" />
    <Condition number="18" value="1" />
  </NextQuestion>
  <NextQuestion number="8">
    <Condition number="4" value="4" />
    <Condition number="18" value="1" />
  </NextQuestion>
  <NextQuestion number="10">
    <Condition number="4" value="5" />
    <Condition number="18" value="1" />
  </NextQuestion>
  <NextQuestion number="12">
    <Condition number="4" value="6" />
    <Condition number="18" value="1" />
  </NextQuestion>
</Question>
<Question number="19" type="Text" pause="False" description="輸入帳號" endchar="#" limit="14" directNextQuestion="20">
</Question>
<Question number="20" type="Text" pause="False" description="輸入現在的語音密碼" endchar="" limit="4" directNextQuestion="0">
</Question>
<Question number="21" type="Text" pause="False" description="輸入身份證後九碼" endchar="#" limit="9" directNextQuestion="0">
</Question>
</Questions>
</Project>

```