한국음향학회지 제20권 제4호 pp. 42~47 (2001)

음성 웹서비스를 위한 VoiceXML 해석기의 설계 및 구현

Design and Implementation of the VoiceXML Interpreter for Voice Web-service

신 현 경*, 강 동 남**, 염 세 훈**, 유 재 우**
(Hyun-Kyung Shin*, Dong-Nam Kang**, Sae-Hun Yeom**, Chae-Woo Yoo)

*상지영서대학 인터넷정보과, **숭실대학교 컴퓨터학과

(접수일자: 2001년 3월 19일; 채택일자: 2001년 4월 18일)

본 연구의 목적은 비 시각 환경에서 웹 서비스를 위한 언어인 VoiceXML을 기존의 자동 응답 시스템에 적용하기 위해 VoiceXML 문서의 마크-업을 인식하고, 문서가 문서형정의 (DTD)에 적합한지를 검사하여 적합성이 확인되면 추상구문트리를 생성하는 DI 파서 (Document Instance Parser)와 생성된 추상구문트리를 이용하여, Voice-XML문서를 번역해주는 해석기를 제안하고자 한다. VoiceXML 해석기는 DI 파서와 실행기로 구성되어 있으며, DI 파서는 Recursive descent 파싱 기법을, 실행기는 VXML 포럼에서 제안한 FIA (Form Interpretation Algorithm)를 사용하였다. 본 시스템은 VoiceXML 언어를 효율적으로 실행할 수 있는 환경 제공 및 시스템 개발의 편의성과 효율성을 위해 모듈화 설계가 가능한 자바언어를 사용함으로써 이 기종간의 이식성이 뛰어난 특징이 있다.

핵심용어: 음성XML, 파서, 해석기, 추상구문트리, ECMAScript, JSGF
투고분야: 음성처리 분야 (2.7)

In this paper, we propose an interpreter, which recognizes the VoiceXML markups, verifies the validation of the document, and interprets the VoiceXML documents using DI parser and the generated AST by the parser. The VoiceXML interpreter consists of DI parser and executor, and the DI parser uses recursive descent parsing technology, and the executor uses FIA (Form Interpretation Algorithm) proposed by VXML forum. This system uses the Java language in order to develop the runtime environment for VoiceXML efficiently, thus this system has portability.

Keywords: *XML, VXML, AST, Parser, Interpreter, FIA, JSGF*
ASK subject classification: *Speech signal processing (2.7)*

I. 서 론

음성인식기술이 실용화 수준으로 발전함에 따라 음성 포털 (VoicePortal)을 비롯한 음성 인식 응용 분야가 새로운 이슈로 떠오르고 있다. 차세대 꿈의 통신을 실현시킬 것이라 하여 현재 업계에 관심이 집중되고 있는 IMT-2000 사업의 준비과정에서도 음성인식 관련 기술은 상당히 중요한 테마로 자리를 차지하고 있다. 이같은 분위기 속에서 최근에는 VoiceXML (Voice eXtensible Markup Language) 문서형식은 음성인식 및 음성합성 모듈을 결합해 음성포털, 차세대 UMS (Unified Messaging System), IP기반 ARS 시스템 등 그 응용 분야에 대한 연구가 활발히 진행되고 있다.

책임저자: 신현경 (hkshin@youngseo.ac.kr)
220-713 강원도 원주시 우산동 660
상지영서대학교 인터넷 정보과
(전화: 033-730-0765; 팩스: 033-730-0761)

음성을 지원하는 대표적인 웹 언어로는 NOTE-Voice, SABLE, SpeechML, VoxML, VoiceXML (이하 VXML) 등이 있다. 본 연구에서는 이들 중 모든 음성 지원 웹 언어의 장점만을 가진 VXML을 기반으로 한다. VXML은 IBM, AT&T, Motorola 등을 주축으로 한 VXML 포럼에서 제안된 대화형 음성 인터페이스의 표준으로 1999년 8월에 제안하여 현재 스펙버전1.0이 발표되었으며, ARS와 같은 전화를 이용한 정보 서비스 시스템에서 사용되는 어플리케이션을 작성할 수 있게 해주는 마크 업 언어이다. 이 VXML은 현재 전화 정보 서비스에서 사용되는 콘텐츠 및 어플리케이션 작성시 발생되던 문제, 즉 C, Assembler와 같은 비교적 저 수준 프로그래밍언어 (Low level programming language)의 사용 및 시스템에 의존적인 API를 사용함으로써 어플리케이션의 개발 속도와, 각 서비스 시스템간의 호환성을 해결하고자 연구 개발되었다. VXML을 사용하면 음성 서비스용 컨텐츠나 어플리케이션의 개발시 전문가가 아닌 일반인이 간단한 마크업 언어를 통하여 음성 서비스용 컨텐츠나 어플리케이션을 작성 할 수 있을 뿐만 아니라, 빠른 서비스 개발이 가능하다. 하지만, VXML을 이용하여 음성 서비스를 하려면, VXML 문서 서버 (Document server), 음성 입/출력기, 그리고 VXML 해석기 (Interpreter)와 같은 VXML 실행 요소가 필요하며, 현재 여러 분야에서 연구 진행중이지만, 아직 실용적으로 사용할 수 있는 표준 해석기와 같은 장치가 부족한 상태이다. 이에 본 논문에서는 VXML 해석기를 설계 및 구현하고자 한다. 이는 앞으로 VXML 해석기 개발 시 표준적인 방향을 제공 할 수 있을 뿐만 아니라, VXML 서비스 컨텐츠 개발시에도 도움을 줄 수 있으리라 기대된다.

II. VXML 어플리케이션 및 시스템 구성

2.1. VXML어플리케이션의 구성

VXML 문서 (Document)는 VXML 문서형정의 (DTD)에 맞게 작성된 문서로 다이얼로그 형태로 이루어졌다. VXML 어플리케이션은 하나의 root 문서와 그를 참조하는 다수의 VXML 문서로 구성되어 있고, 그 문서는 VXML DI 서버에 저장되어 보관되어진다. 실제로 서비스하기 위해서는 해석기 등을 포함한 VXML 브라우저가 필요하다. 그림 1은 이런 관계를 나타낸 것이다. 하나의 어플리케이션은 다른 어플리케이션과 연결되어 사용될 수도 있다.

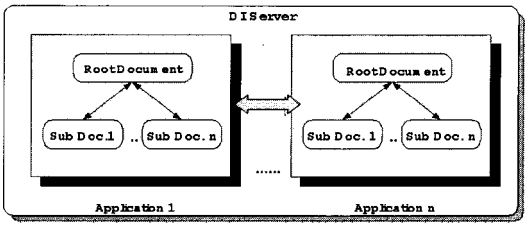

그림 1. VXML 어플리케이션 구성
Fig. 1. Architecture of the VXML application.

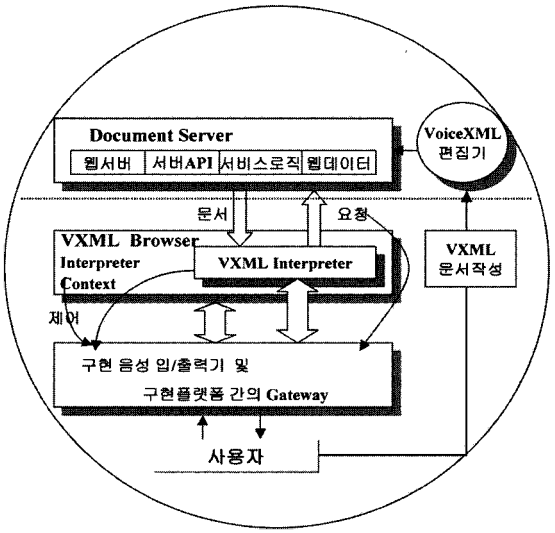

그림 2. VXML 시스템 구성
Fig. 2. Architecture of the VXML system.

2.2. VXML 시스템의 구성

VXML 시스템은 그림 2와 같이 크게 나누어 3가지로 분류할 수 있다. 첫째로 VXML 문서 서버로 사용자의 요구에 따라 VXML 문서를 브라우저로 전송해주는 역할을 한다. 둘째로, VXML 브라우저는 DI 서버에서 받은 문서를 해석하고 실행하는 역할을 하며, 파서 (Parser)와 해석기로 구성되어 있다. 마지막으로, 구현 플랫폼에 따른 음성 입/출력기와 전화 서비스 연결에 다른 게이트웨이 등으로 구성되어 있다. 이중 해석기는 VXML 어플리케이션을 입력으로 하여 실행하며, 사용자에게 서비스를 제공하는 역할을 한다. 이 때 사용자로부터의 입력은 음성 입력기를 통하여 해석기로 입력되어지고 출력은 음성출력기를 통하여 사용자에게 전해진다.

III. VXML 해석기의 설계

VXML 해석기는 VXML 문서를 일반 PC 환경 아래서 해석할 수 있는 환경을 제공하여 줌으로써, VXML 문서를

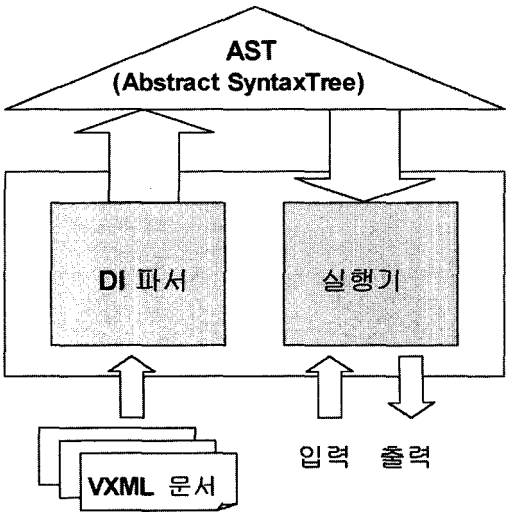

그림 3. VXML 해석기
Fig. 3. VXML Interpreter.

읽어 들여 문서의 적합성을 판단하고, 적합한 문서인 경우 실행하여, VXML 어플리케이션의 작성시 작성한 어플리케이션을 시험해 볼 수 있는 환경을 제공하여 준다.

3.1. VXML 해석기의 구성

VXML 해석기는 크게 그림 3과 같이 DI (Document Instance) 파서와 실행기로 구성되어 있으며, 시스템의 전체적인 흐름은 문서를 읽거나 작성시 최초 파서를 이용하여 문서의 유효성과 적합성을 점검하고 추상구문트리(AST: Abstract Syntax Tree)[9]를 구성하여 실행기에서 적절한 처리를 할 수 있게 하여 준다. 또한 실행시에 생성되는 자료 (심볼테이블 및 기타 자료)를 추상구문트리 조작기를 이용하여 추상구문트리 노드에 속성을 추가하며 사용자 인터페이스를 이용하여, 실행 시에 필요한 사용자의 입력을 받아들이거나 실행 결과를 표시하여 준다.

추상구문트리의 내부모듈은 기본 연산 모듈, 앨리먼트 단위연산 모듈, 되살리기 (undoable) 연산모듈의 세가지로 구성되며, 각 모듈은 클래스로 이루어지며, 기본연산으로부터 앨리먼트 연산, 되살리기 연산으로 계승 (inheritance)되어 사용된다.

3.2. DI 파서의 설계 및 구현

DI Parser는 VXML문서의 적합성을 확인하는 프로세서로 대표적인 하향식 구문분석 방식인 recursive descent 파싱기법을 사용하여 설계되었으며, 그림 4와 같이 주어진 문서에서 토큰 (token)을 추출하는 어휘 분석기, 문서의 적합성을 분석하는 구문 분석기 및 문서의 의미의 적

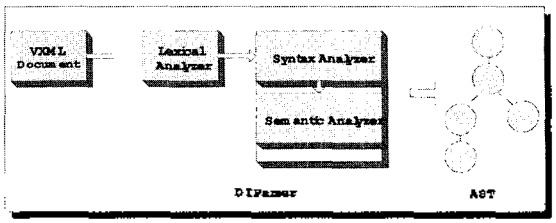

그림 4. DI 파서
Fig. 4. Document Instance Parser.

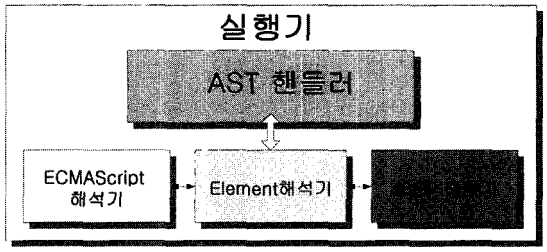

그림 5. 실행기의 구조
Fig. 5. Architecture of the executer.

합성을 분석하는 의미 분석기로 나누어 설계 구현되었다. DI 파서의 주 클래스는 파서 클래스이고, 구문 분석을 담당하며, 어휘분석기는 Yylex와 Yytoken 클래스가 담당하고, 추상구문트리를 생성하는 의미 분석기는 VxmlParseTree, VxmlNode, VxmlAttr, VxmlAttr, VxmlAttrVector 클래스가 담당한다. VXML DTD 와 Tag들이 가지고 있는 속성에 대한 정보는 VxmlAST 와 VxmlSyntax 클래스가 담당한다.

3.3. 실행기의 설계

실행기는 VXML 문서를 어휘분석, 구문분석, 의미분석 등을 거쳐 생성된 추상구문트리와 각 단계에서 생성된 정보를 토대로 실제 VXML 문서를 해석하는 부분이다. VXML포럼에서 제안한 FIA (Form Interpretation Algorithm)를 기본으로 하여 구현되었고, 그림 5와 같이 추상구문트리 인스턴스 핸들러, JSGF (Java Speech Grammar Format)를 해석하는 JSGF해석기, ECMAScript를 해석하는 ECMAScript 해석기, 그리고 VXML의 앨리먼트를 해석하는 앨리먼트 해석기로 구성되어 있다. 추상구문트리는 파서와 실행기가 공통으로 사용되는 것이고, 나머지는 실행기 내부에서만 사용된다.

3.3.1. 폼 해석 알고리즘 FIA (Form Interpretation Algorithm)

폼의 해석은 폼 아이템 단위로 처리되며, FIA는 폼 아이템의 기본적인 해석 알고리즘으로 3단계로 구성된다.

표 1. FIA 알고리즘
Table 1. Form Interpretation algorithm.

```
//Initialization Phase
f_i[] = form item;
f_i_g[] = form item guardcondition;
for (i=0 ; i<number of form item; i++){
  if ( f_i[i] = "var" || f_i[i] == "field")
    declare variable or field_item  with f_i[i]'s "expr" attribute, or undefined
}
//Main Loop
current_form_item= null;
end_of_loop = false;
event_exist = false;
do{
  // Select Phase
  if (last main loop iteration end with goto)
    process goto element;
    continue;
  for (i=0; i<number of form item; i++){
    if (f_i_g[i] == false){
      current_form_item = f_i[i];
      break;
    }
  }
  // Collect Phase
  if (current_form_item == null)
    end_of_loop = true;
    continue;
  switch(current_form_item){
    case field :
   process "field" element;
    break;
    case record:
      process "record" element;
    break;
  :
    //process all of form item with the case statement
  }
  f_i_g[i] = true;
  // Process Phase
  if (event_exist == true)
   :
process event handler;
    f_i_g[i] = false;
    continue;
else if (has_filled_element == true)
    process filled element
     :
}while(!end_of_loop)
    :
```

표 1은 FIA 슈도 코드 (Pseudo Code)이다.

첫 단계는 선택단계로 실행될 폼 아이템을 결정하는 단계, 두 번째는 수집단계로 결정된 폼 아이템을 실행하는 단계로 경우에 따라 사용자와의 상호작용 (Interaction)이 일어나며, 문서 실행에 필요한 정보를 수집한다. 세 번째 단계는 처리단계로 수집 단계에서 수집된 정보를 처리하는 단계로 유효성을 검사하며, 이벤트 (Event)발생시 처리를 한다.

3.3.2. JSGF 해석기

VXML에서 모든 사용자의 입력은 JSGF를 이용하여 정의한 문법에 의거하여 해석[1]해야 하기 때문에 VXML 시뮬레이터는 사용자의 입력을 바로 해석하기 위해서 JSGF 해석기가 필요하다. JSGF 해석기의 구조는 그림 6의 클래스 다이어그램과 같은 구조로 설계 구현되었다.

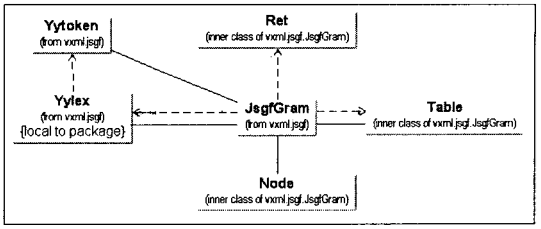

그림 6. JSGF 해석기
Fig. 6. JSGF Interpreter.

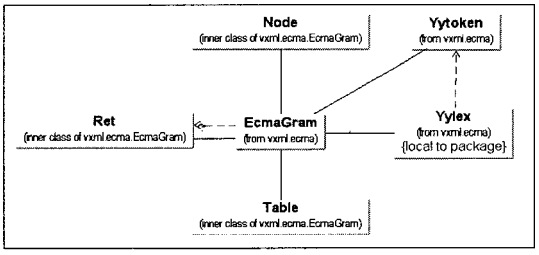

그림 7. ECMAScript 해석기
Fig. 7. ECMAScript Interpreter.

그림 6에서 보면 가장 기본적인 JSGF에 대한 처리를 하는 JsgfGram과, 입력된 JSGF 문법을 토큰으로 변환하는 Yytoken 및 Yylex가 있고, 입력된 문법을 정의한 내부 자료 구조 Tree와 이 자료구조에 접근 할 수 있게 해주는 Node, 그리고 실행된 결과를 전달해 주는 Ret 등으로 구성되어 있다.

3.3.3. ECMAScript 해석기

VXML은 마크업 언어의 특징상 수식 및 조건식의 처리를 할 수 없기 때문에, ECMAScript를 이용하여 이러한 처리를 수행한다[1]. ECMAScript 해석기는 그림 7과 같이 설계 구현되었다.

ECMAScript 해석기 역시 JSGF 해석기와 비슷한 구조로 구성되어 있고, 가장 기본적인 처리를 하는 EcmaGram과, 입력된 ECMAScript 수식을 토큰으로 변환 해주는 Yytoken 및 Yylex가 있고, 입력된 수식을 정의한 자료 구조 Tree와 이 자료구조에 접근할 수 있게 해주는 Node, 그리고 실행된 결과를 전달해 주는 Ret 등으로 구성되어 있다.

3.3.4. 앨리먼트 (Element) 해석기

앨리먼트 해석기는 DI Parser에서 생성한 추상구문트리를 이용하여 VXML 앨리먼트들을 해석하고, VXML 스펙

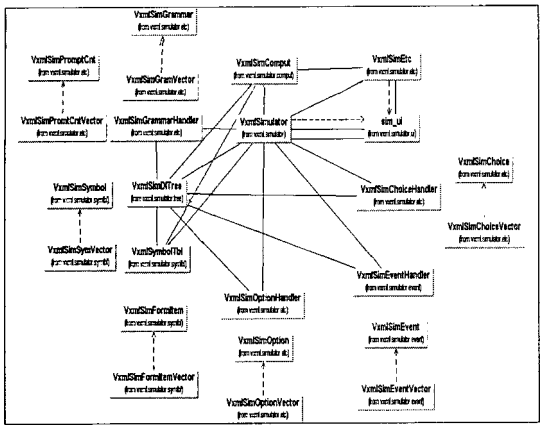

그림 8. 엘리먼트 해석기
Fig. 8. Element Interpreter.

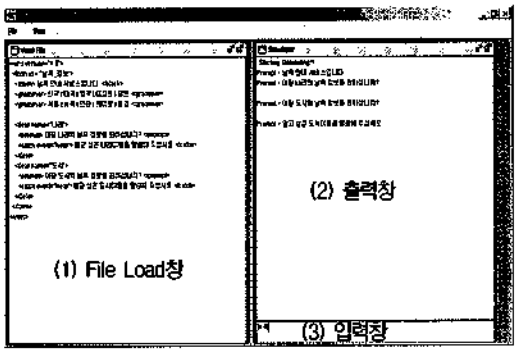

그림 9. VXML 해석기 결과화면
Fig. 9. Appearance of the VXML Interpreter execution.

에 맞게 시뮬레이팅하는 기능을 담당한다. VXML의 해석기는 VXML문서를 폼 단위로 나누고, 다시 그 폼의 폼 아이템을 FIA[1]를 이용하여 분석 실행한다. 엘리먼트 해석기는 그림 8과 같이 클래스 다이어그램과 같은 구조로 구성되어 있고, 각각의 클래스의 역할은 표 2에 나와 있다.

IV. 실험결과 및 시뮬레이션

본 연구에서 구현한 VXML 해석기는 Window NT/2000, Linux, Sun JDK 1.3, JMF2.1.1 beta2, Borland JBuilder4.0을 기반으로 구현하였다.

본 VXML 해석기가 실행되면, 자바 가상머신이 메모리에 로드 되어 VXML 해석기가 실행된다. 실행결과 화면은 그림 9와 같다. 화면의 (1)File Load 창은 사용할 VXML 어플리케이션을 보여주는 창이며, (2)출력 창은 실행된 결과를 사용자에게 보여주는 창이다. (3)의 입력 창은 실행시에 필요한 입력을 사용자로부터 받을 수 있는 창으로 설계되었다. 본 해석기가 제공하는 주요 특징으로는 간단한 GUI 형태를 가지고 있어 초보자라 할 지라도 직관적

표 2. 엘리먼트 해석기의 클래스
Table 2. Element Interpreter Classes.

Class 명	역 할
VxmlSimulator	시뮬레이터의 메인 class
VxmlSimcouput	ECMAScript처리를 위하 보조 Class
VxmlSimEvent	Event의 AST에 대한 자료구조
VxmlSimEventHandler	Event에 대한 AST 핸들러
VxmlSimEventVector	Event에 대한 AST노드 속성 벡터
VxmlSimFormItem	Form Item의 AST에 대한 자료 구조
VxmlSimformItemVector	Form Item에 대한 AST노드 속성 벡터
VxmlSimSymbol	Symbol의 AST에 대한 자료 구조
VxmlSimSymbolVector	Symbol에 대한 AST노드 속성 벡터
VxmlSimSymbolTbl	Form item과 Symbol에 대한 AST 핸들러
VxmlSimDiTree	전체 AST 핸들러
VxmlSimGrammar	Grammar에 대한 AST 자료 구조
VxmlSimGrammarVector	Gramamr에 대한 AST노드 속성 벡터
VxmlSimPromptCnt	Prompt counter에 대한 AST 자료 구조
VxmlSimPromptCntVector	Prompt countet에 대한 AST노드 속성 벡터
VxmlSimGrammarhandler	Gramamr 및 Prompt counter의 Ast 핸들러
VxmlSimEtc	시뮬레이터에서 사용하는 각종 Method들을 위한 Class

으로 사용이 가능하며, 다른 시스템의 도움 없이 거의 모든 환경에서 VXML 어플리케이션의 동작을 확인해 볼 수 있기 때문에 VXML 어플리케이션 작성을 보다 편리하게 할 수 있도록 도움을 줄 수 있다. 또한 숙련된 사용자를 위하여 Hot key도 지원한다. 본 연구는 단일 VXML 문서의 해석기를 기본으로 연구하였기 때문에 문서 혹은 어플리케이션 사이를 옮겨가며 수행되는 다중문서 (Multi Document) 어플리케이션을 위한 엘리먼트 처리는 구현 대상에서 제외되었으며, ECMAScript의 경우에 그 스팩의 범위가 VXML보다 방대하기 때문에 기본적인 수식 및 조건식 처리로 그 범위를 한정지어 구현하였으며 meta, link, object, param, pros, return, subdialog, submit, transfer 엘리먼트는 VXML 해석기에서 구현하지 못하였다.

V. 결 론

본 연구는 VXML문서를 기존의 자동 응답 시스템에 적용하기 위하여 VXML 문서를 번역하는 기술을 연구하고 구현하였다. 효과적인 번역을 위하여 VXML 언어 문서의 특성 및 문서의 번역에 필요한 어휘분석, 구문분석, 의미분석과정으로 구성된 파싱 기술 구현하여 VXML언어를 번역하기 위한 방법을 연구하였다. DI 파서는 VXML문서

의 적합성을 확인하는 프로세서로 대표적인 하향식 구문 분석 방식인 recursive descent 파싱 기법을 사용하여 설계되었으며, 실행기는 크게 JSGF를 해석하는 JSGF해석기, ECMAScript를 해석하는 ECMAScript해석기, VXML 앨리먼트를 해석하는 앨리먼트 해석기로 구성되어, DI 파서에 의해 생성된 추상구문트리를 토대로 실제 VXML 문서를 FIA 알고리즘을 이용하여 구현하였다. 번역하는 과정에서 VXML 언어로 표현되는 문서가 가지는 특성으로 인하여 번역이 불가능한 기능이 있음을 알 수 있었고, ECMA-Script의 경우 그 스펙의 범위가 VXML 보다 방대하므로 기본적인 수식과 조건식 처리로 그 범위를 한정지어 구현하였다. 본 VXML 해석기의 특징은 쉽고, 간단한 GUI형태로 설계되어 사용자가 직접 시뮬레이션하고 있는 VXML 문서를 확인할 수 있도록 실행 창 뿐만 아니라, VXML File 창을 따로 만들어 사용자 편의제고를 기하였다. 아울러 시스템 개발의 편의성과 효율성을 위해 모듈화 설계가 가능한 자바언어를 사용함으로써 이 기종간의 이식성이 뛰어난 장점을 가졌다. 이로 인하여 본 논문에서 개발한 VXML 해석기는 관련 어플리케이션 개발에 크게 기여 할 것으로 기대되며, 향후 연구과제로는 VXML 해석기를 확장하여 범용 VXML 해석기의 구현이 가능하도록 하고 다중 문서 어플리케이션에 대한 처리, 음성 입/출력기와의 연동 및 DI 서버에 대한 연구도 계속적으로 요구된다.

참 고 문 헌

1. VoiceXML forum, "Voice Extensible markup language, Version 1.0", http://www.voicexml.org.
2. Sun Microsystems, "JAVA Speech Grammar Formal Specification, Version 1.0," Sun Microsystems, Oct., 1998.
3. D. Stallard., Demonstration of BBN SPIN(Speech over the Internet), Presented at MIT, Cambridge, MA, 1997.
4. N. K. David 외, "Requirements for markup language for HTTP-mediated interactive voice response services", AT&T Lad-Research, Lucent Technologies-Bell Lab., Motorola ICSD, 1999.
5. Nakhimvsky and T. Myers., "Professional Java XML Programming-with Servlets and JSP," Wrox Press, 1999.
6. 김경란 외, "VXML 편집기와 음성브라우저의 설계 및 구현", *춘계 정보과학회 학술대회*, April, 2000.
7. 우상남, "VoiceXML 구동부터 애플리케이션 실행까지", *월간 마이크로 소프트*, Dec., 2000.
8. Daive A. Watt., Programming Language Processors., Prentice-Hall, 1993.
9. A. V. Aho et al., Compilers: Principles, Techniques and Tools., Addison-Wesley, 1986.

저자 약력

● 신 현 경 (Hyun-Kyung Shin)

1985년 2월: 숭실대학교 수학과 (이학사)
1988년 2월: 숭실대학교 전자계산학과 (공학사)
1991년 2월: 숭실대학원 전자계산학과 (공학석사)
1997년 8월: 숭실대학원 컴퓨터학과 (박사과정 수료)
1998년 2월~현재: 상지영서대학 인터넷정보과 조교수
※ 주관심분야: 컴파일러, 프로그래밍 환경, HCI, VoiceXML

● 강 동 남 (Dong-Nam Kang)

2000년 2월: 국립 산업대학 전자계산학과 졸업 (공학사)
2000년~현재: 숭실대학교 컴퓨터학과 석사과정
※ 주관심분야: 컴파일러, XML, Embedded Programming

● 염 세 훈 (Sae-Hun Yeom)

1992년 2월: 국립 서울 산업 대학 전자계산학과 졸업 (공학사)
1992~1994년: 숭실대학교 전자계산학과 (공학 석사)
1994~현재: 숭실대학교 컴퓨터학과 박사과정
※ 주관심분야: 프로그래밍 언어, 인간과 컴퓨터 상호작용, Speech (Voice) Based Markup Language

● 유 재 우 (Chae-Woo Yoo)

1976년 2월: 숭실대학교 전자계산학과 (공학사)
1978~1985년: 한국과학기술원 전산학과 (공학 석사, 박사)
1986~87, 96년~97년: Cornell Univ.와 Univ of Pittsburgh의 Visiting Scientist
1983년~현재: 숭실대학교 컴퓨터학부 교수
※ 주관심분야: 컴파일러, 프로그래밍 환경, 인간과 컴퓨터 상호작용