한국음향학회지 제23권 제5호 pp. 390-399 (2004)

MVC 프레임 워크를 사용한 VoiceXML 다중 뷰 편집기의 설계 및 구현

A Design and Implementation of the VoiceXML Multiple-View Editor Using MVC Framework

염 세 훈*, 유 재 우**
(Sae-Hun Yeom, Chae-Woo Yoo)

*동서울대학교 컴퓨터소프트웨어과 **숭실대학교 컴퓨터학부

(접수일자: 2003년 9월 29일; 채택일자: 2004년 7월 12일)

본 논문에서는 음성 웹 언어인 VoiceXML의 작성 효율을 향상하기 위한 다중 뷰 편집기를 설계 및 구현하였다. VoiceXML 다중 뷰 편집기는 다중 뷰를 제공하기 위해 MVC (Model-View-Controller) 프레임워크을 이용하였다. MVC 프레임워크를 이용한 다중 뷰 편집기는 핵심 자료구조인 모델과 인터페이스인 뷰, 모델과 뷰를 제어하기 위한 제어기로 구성된다. MVC 프레임워크에서 모델은 추상 구문 트리와 추상 문법으로 구성되며 뷰는 역파싱 규칙과 역파서로 구성되고 제어기는 명령어 처리기와 트리 조작기로 구성된다. VoiceXML 다중 뷰 편집기는 문서의 구조, 내용, 흐름을 동시에 보여주어 기존 XML 편집기의 단점을 극복할 수 있다. MVC 프레임워크가 적용된 VoiceXML 다중 뷰 편집기는 여러 편집기를 통해 동시에 다양한 편집 뷰 (View)를 제공함으로써 사용자에게 음성 웹 문서 작성의 편의성을 제공하여 효율을 높일 수 있으며 여러개의 뷰가 하나의 모델을 가짐으로써 편집기들의 무결성을 보장하도록 하였다.

핵심용어: VoiceXML, MVC 프레임워크, 다중 뷰, 추상구문트리, XML, 구문분석
투고분야: 음성처리 분야 (2,9)

In this paper, we design and implement a multiple-view VoiceXML editor to improve editing efficiency of the VoiceXML. The VoiceXML multiple-view Editor uses a MVC framework to support multiple views and paradigm. Our multiple-view editor consists of Model, View and Controller using MVC framework. A model, core data structure, is constructed of abstract syntax tree and abstract grammar. A view, user interface, is formalized in unparsing rules and unparser. A controller, to control model and view, is made of command interpreter and tree handler. The VoiceXML multiple-view editor overcomes a drawbacks of existing XML editors by showing document structure and context concurrently, as well as document flows. Our VoiceXML multiple-view editor, which MVC framework has been applied, provides various editing views concurrently to users. Thereby, it supports efficient and convenient editing environments for voice-web documents to users and it guarantees transparency of editors, as various views have a same consistent model.

Keywords: VoiceXML, MVC framework, Multiple-view, Abstract syntax tree, XML, Syntax analysis
ASK subject classification: Speech signal processing (2,9)

I. 서 론

WWW의 보편화로 많은 사용자들이 인터넷에 쉽게 접근하고 사용할 수 있게 되면서 많은 정보들이 웹 상에 존재하게 됐다. 이러한 추세와 함께 휴대 이동 통신의 발전이 더욱 가속화되면서 휴대 이동 통신과 웹이 결합되고 있고 이를 위해 음성을 웹에서 이용하고자 하는 요구가 발생하고 있다. 그러나 현재 인터넷상에서 정보를 표현하는 방법은 HTML에 국한되어 있으며 이를 음성으로 이용하기에는 어려움이 많은데, 이러한 요구로 인터넷상에서 문서표현 및 검색을 음성으로 제공하기 위해 제시된 것이 음성지원 마크 업 언어이다. 음성지원 마크

책임저자 : 염 세 훈 (shyeom@dsc.ac.kr)
416-714 경기도 성남시 수정구 복정동 423
동서울대학교 컴퓨터소프트웨어과
(전화: 031-720-2091; 팩스: 031-720-2287)

업 언어는 인터넷상의 정보를 음성으로 이용하여 표시할 수 있게 한다.

음성을 지원하는 대표적인 웹 언어는 NOTE-Voice, SABLE, SpeechML, VoxML, VoiceXML[1] 등이 있다. 본 연구에서는 이들 중 현재 W3C에서 음성을 지원하기 위한 표준 마크 업 언어인 VoiceXML을 대상으로 한다. VoiceXML은 AT&T, IBM, Lucent, Motorola사 등이 기반이 되어 설립한 VoiceXML 포럼에서 설계되었으며 현재 W3C(World Wide Web Consotium)에서 음성 응답 응용 프로그램을 작성하는 기반 언어로 채택하고 있다[2]. 하지만 VoiceXML은 고유의 문법을 가지고 있기 때문에 일반 문서 편집기로 문서를 작성하기에는 어렵다. 게다가 문서를 작성했다고 하더라도 그 문서의 적합성을 검증할 수 없다는 단점이 있다. 더욱이 VoiceXML의 경우 음성 웹 언어의 특성 상 문법적인 적합성뿐만 아니라 문서내의 흐름, 즉 시나리오의 중요성도 무시할 수 없으며[3] 대표적 VoiceXML 편집기인 뉴앙스의 V-Builder 1.0은 VoiceXML 문서 작성자에게 쉬운 문서 편집을 위하여 디자인과 소스, 두가지 뷰를 제공하고 있다. 사용자는 디자인을 통해 전체적인 문서 구조를 디자인하고 소스를 통해 상세한 속성들을 입력하게 해준다. 하지만, 문서 디자인 단계에서 상세한 속성을 삽입하기 어려워 사용자는 문서 디자인을 한 후 소스에서 그 이외의 속성들을 입력하게 하여 이중의 작업을 초래한다. XML Prov2.0이나 EXml, XMLSpy와 같은 기존의 XML 편집기도 문서구조를 표시하는 트리 뷰와 소스 뷰를 제공하지만 이 트리와 소스 뷰를 동시에 보여주지 않아 문서 작성 효율을 떨어트린다. 또한, 기존의 XML[4] 편집기들은 문서구조의 적합성만을 제공할 뿐 VoiceXML에서 중요한 음성의 흐름인 시나리오의 중요성은 무시되고 있어 VoiceXML 문서의 작성에는 적합하지 않다.

위와 같은 단점을 해결하기 위해 본 연구의 VoiceXML 다중 뷰 편집기는 문서의 적합성 검증을 제공하는 한편 문서의 구조와 내용, 시나리오 흐름을 동시에 지원하는 다중 뷰 기능을 제공하였다. 본 논문에서는 동시성을 가진 다중뷰를 제공하기 위해 MVC 프레임워크를 이용하였다. VoiceXML 다중 뷰 편집기에 적용된 MVC 프레임워크는 Model로 핵심 자료구조를 사용하였으며, View로 화면에 나타나는 인터페이스를 정의하였으며, Controller로 사용자 인터페이스로부터 입력을 받는 방법을 정의한다. MVC 프레임워크를 이용하여 개발함으로써 하나의 문서를 다양한 형태로 동시에 표현하여 사용자가 문서작성을 효율적이고 편리하게 작성할 수 있도록 도와준다. 본 논문에서는 VoiceXML 편집기에 동시성을 가진 다중 뷰를 제공하기 위한 MVC 프레임워크를 제시하고 MVC 프레임워크를 이용한 다중 뷰 VoiceXML 편집기를 설계, 구현한다.

II. VoiceXML 다중 뷰 편집기와 MVC 프레임워크 구조

VoiceXML 다중 뷰 편집기는 음성 지원 웹 언어에 익숙하지 않은 초보자나 언어의 구조를 정확히 모르는 사용자가 정확한 음성 지원 웹 언어의 구조를 모르고도 오류 없이 쉽게 음성 지원 웹 응용 프로그램을 작성할 수 있게 하는 대화식 프로그래밍 지원 시스템이다.

음성 지원 웹 언어의 작성 시 편집과 동시에 문서의 적합성이 보장되어 오류 없는 문서 작성을 지원하며 MVC 프레임워크를 이용한 다양한 편집 형식을 제공한다. 제공하는 편집 형식은 문서의 내용과 구조를 보여주는 형식과 음성 지원 웹 언어 작성 시 중요시되는 문서내의 시나리오의 흐름을 보여주는 것이 있다.

본 편집기에서 다중 뷰로 제공되는 편집 뷰는 첫째로 문서의 내용을 나타내기 위하여 구조적 텍스트 편집이 가능한 구문 지향 편집기[5]를 제공하였다. 구문 지향 편집기는 문서의 내용을 문법에 따른 구조와 함께 텍스트 형식으로 보여줌으로써 사용자가 작성하고 있는 문서의 내용을 쉽게 파악할 수 있도록 한다. 두 번째로 문서의 구조를 나타내기 위하여 트리 기반 편집기를 제공한다. 트리 기반 편집기는 현재 작성하고 있는 문서의 구조를 트리형식으로 표현하여 문서 작성자가 쉽게 문서의 구조를 파악할 수 있도록 한다. 마지막으로 문서의 흐름, 즉 시나리오를 표현하기 위하여 다이어그램 기반 편집기를 제공한다. 다이어그램 기반 편집기[6]는 문서의 내용이나 구조와 달리 문서 내에서의 문서 내용의 흐름을 보여줌으로써 문서가 어떤 흐름을 가지고 있는지 나타낸다. 이와 같이 다양한 편집 환경을 제공함으로써 문서 작성자는 일목요연하게 문서를 파악할 수 있게 한다. 기존의 XML 편집기들은 문서의 인터페이스 형식이 단일화되어 사용자가 문서의 여러 가지 모습을 동시에 파악하기가 어려웠다. 본 시스템에서는 이러한 단점을 보완하고자 MVC 프레임워크를 적용하여 다중 뷰를 제공하였다.

MVC 프레임워크를 제공함으로써 사용자는 문서의 다양한 측면 (View) 중 하나의 면을 중심으로 편집을 하면서 다른 면을 동시에 파악할 수 있는 환경을 제공한다.

VoiceXML 다중 뷰 편집기는 MVC 프레임워크를 기반으로 하고 있다. MVC 프레임워크는 시스템을 모델(Model)과 뷰 (View)와 제어기 (Controller)로 나누어 시스템을 설계하는 방법으로 Smalltalk-80에서 사용자 인터페이스를 개발하는데 사용하였다. MVC 프레임워크는 하나의 핵심 자료구조인 모델을 기반으로 인터페이스인 뷰와 인터페이스와 모델을 제어하는 제어기를 따로 둠으로써 하나의 자료 구조로 다양한 형식의 인터페이스를 가질 수 있도록 한다.

본 논문에서는 VoiceXML 다중 뷰 편집기에 맞도록 MVC 프레임워크를 그림 1과 같이 적용하였다.

2.1. 모델

모델은 문서의 추상 구문 트리 (Abstract Syntax Tree)로 구성된다. 추상 구문 트리는 편집기에서 읽은 VoiceXML의 문서를 트리구조로 표현한 것으로 편집기의 모델이 되는 부분이다. 추상 구문 트리[7]는 추상문법을 이용하여 생성한다. 추상 문법은 추상 구문 트리를 생성하기 위한 문법이다. VoiceXML 문서가 VoiceXML DTD를 기반으로 작성되기 때문에 VoiceXML 문서를 트리형식으로 표현한 추상 구문 트리도 VoiceXML DTD를 기반으로 하는 문법을 필요로 한다. 이를 위해 추상 문법을 제공한다. 본 논문에서는 추상 문법을 문맥 자유 문법 형식(CFG; Context-Free Grammar)[7]으로 작성하였다. 이는 VoiceXML DTD(Document Type Definition)를 문맥 자유 문법으로 변환하기 용이하기 때문인데 이러한 특성을 이용하여 VoiceXML의 DTD를 편집환경에서 사용하기 위한 추상 문법(Abstract Grammar)으로 변환하였다. 추상 문법은 파서에 의해 추상 구문 트리를 변경할 때 사용된다.

2.1.1. 추상 문법 (Abstract Grammar)

추상 문법은 VoiceXML 개발 환경의 핵심 자료구조인 추상 구문 트리를 생성하기 위한 문법으로 편집기에서 만들어지는 문서는 추상 문법에 의해 문법이 검증되고 생성된다. 추상 문법은 개발 환경에서 문서를 작성할 때 현재 선택된 문법에 삽입 가능한 문법들을 얻거나 파서에서 수정된 문서를 검증하는 자료로 사용된다. 이러한 작업을 하기 위하여 VoiceXML의 DTD를 편집 환경에 맞게 CFG 형식 문법으로 변경하였다. VoiceXML의 DTD를 CFG형식의 문법으로 변환함으로써 본 논문에서 구현한 편집기들이 공통된 자료구조를 가지게 할 수 있어 편집기들 간에 무결성을 제공할 뿐만 아니라 CFG 형식으로 변환함으로써 추상 구문 트리를 쉽게 구성할 수 있다. 또한 CFG형식의 문법은 파서에서 문서의 유효성을 검사하기에 적절한 구조를 가지고 있다.

추상문법 G = {T, D, R, S}로 정의할 수 있으며, 여기서
T는 Tag들의 집합,
D는 PCDATA 혹은 CDATA들의 집합
R은 DTD Rule들의 집합
S는 문법의 시작기호를 나타낸다.

또한 DTD 규칙들의 집합 R은
S → Vxml_RootTag
Vxml_RootTag → α | D
여기서,
α 는 (T)+ or (T | D)+의 형태로 정의할 수 있는데 이를 이용하여 VoiceXML DTD를 추상 문법으로 변환하는 알고리즘을 다음과 같이 구성하였다.

[알고리즘 1] VoiceXML DTD의 추상 문법으로의 변환
입력 : VoiceXML DTD
출력 : VoiceXML 추상 문법
방법 :

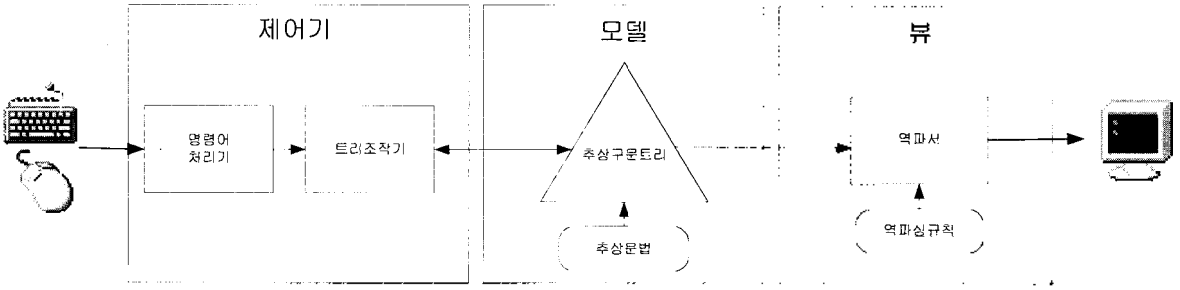

그림 1. VoiceXML 다중 뷰 편집기의 MVC 프레임워크
Fig. 1. MVC framework of VoiceXML multi-view editor

Step 1. VoiceXML DTD를 어휘분석과 구문 분석을 한다.

Step 2. Step 1을 수행하며 DTD에 대한 파스트리를 생성하고 DTD의 엔티티 정보를 저장한다.

Step 3. DTD 파싱 결과를 가지고 엘리먼트 노드 정보만을 추출하여 그래프 구조로 저장한다.

Step 4. 그래프 구조를 이용하여 추상 문법을 생성한다.

1) 루트노드부터 순환하면서 레벨별로 부모노드와 자식노드간의 관계를 CFG의 생성규칙을 만들면서 초기 문법을 생성한다.
2) 생성된 초기 문법 중에서 생성 규칙의 오른쪽에 한 개 비단말 기호만을 갖는 규칙을 대상으로 생성 규칙들을 결합하여 불필요한 생성규칙을 제거한다.
3) 발생자(+, *)에 따른 생성규칙 처리단계로 + 발생자를 먼저 처리한 다음 * 발생자에 따른 생성 규칙을 정리한다. ▮

알고리즘 1을 이용하여 변환한 VoiceXML DTD를 다음과 같은 추상 문법으로 변환할 수 있다.

main -> vxml
vxml -> Element vxml
Element -> Catch | Help| Noinput | Nomatch | Error | Form | Link | Menu | Meta | Property | Script | Var | Nil
Form -> formElement | Nil
formElement -> Grammar | Initial | Subdialog | Field | Filled | Record | Dtmf |Block | Var | Catch | Help | Noinput | Nomatch | Error| Object | Link | Transfer | Property | Comment

2.1.2. 추상 구문 트리 (Abstract Syntax Tree)

추상 구문 트리는 편집기에서 작업 중인 문서의 파싱한 결과를 트리 구조로 나타낸 것이다. 추상 구문 트리의 생성과 수정은 추상 문법을 이용한다. 추상 구문 트리는 편집기에서 VoiceXML의 문법을 더욱 쉽게 다룰 수 있게 한다. XML의 문법인 DTD는 문법인 엘리먼트들의 관계와 이들간의 속성을 따로 정의하고 있다. 이러한 구조는 편집기 구현에 적합하지 않다.

XML은 문서를 구성하는 문법들만으로 구성된 것이 아니라 문법을 구성하는 엘리먼트와 이들의 속성들로 구성되어있다. 이러한 이유로 XML을 작성하기 위한 편집기는 엘리먼트들 간의 관계 뿐 만 아니라 각 엘리먼트에 속하는 속성들도 함께 보여주어야 편집기의 편의성이 높아준다. 이런 구조를 가지기 위해서 문법을 구성하는 엘리먼트와 속성을 함께 표시할 수 있는 자료구조가 필요하게 되었다. 이런 필요성에 의해 본 논문에서는 엘리먼트와 속성을 함께 저장할 수 있는 트리 구조인 추상 구문 트리를 사용하였다. 추상 구문 트리를 사용함으로써 편집기 사용의 편의성은 물론 문서 편집 중 필요한 정보를 한꺼번에 가지고 올 수 있어 편집 속도의 향상을 가져올 수 있다.

문서를 편집하는 동안 문서를 수정하면 추상 구문 트리 핸들러는 추상 구문 트리의 수정되는 부분을 추상문법에서 참조하여 추상 구문 트리를 재구성한다.

추상 구문 트리 생성 과정은 알고리즘 2와 같다.

[알고리즘 2] 추상 구문 트리 생성
Algorithm make_abstract_syntax_tree()
입력 : 생성규칙 번호
출력 : 추상 구문 트리
방법 :

Step 1. 구문 분석기 호출하여 입력토큰을 받는다

Step 2. 트리의 루트를 생성한다.

Step 3. 생성된 트리의 맨 마지막 포인터를 ref에 저장한다.

Step 4. 입력된 토큰에 따라 다음 과정을 수행한다.

1) ref가 가르치는 노드(입력토큰)의 생성규칙 번호를 1 증가한다.
2) ref가 가르치는 노드의 왼쪽에 새로운 노드를 추가하고 추가된 노드를 node에 저장한다.
3) node값을 parent에 저장한다.
4) ref가 가르치는 노드의 오른쪽에 새로운 노드를 추가하고 추가노드를 ref에 저장한다.
5) node가 가르치는 노드의 왼쪽에 Nilnode를 달고 node에 저장한다.
6) 입력 토큰에 대한 트리를 생성하고 node가 가르치는 곳에 추가한다.
7) 다음 입력 토큰에 대해 구문 분석기를 호출한다. ▮

추상 구문 트리는 각 편집기에 기본 자료구조로 사용하여 각 편집기간의 수정된 내용의 일관성을 유지 시켜

준다. 사용자가 편집 연산을 수행할 경우 이것이 즉각적으로 추상 구문 트리에 반영된다. 추상 구문 트리가 변경되면 변경 사항이 즉각적으로 다른 뷰에 적용되어 각 편집기간의 무결성을 제공한다. 아래 과정은 왼쪽의 VoiceXML 문서가 추상문법을 통해 파싱된 추상 구문트리로 변경한 예이다.

2.2. 뷰

뷰는 추상 구문 트리를 사용자 인터페이스로 보여주는 역파싱 규칙 (Unparsing scheme)과 역파서로 구성된다. 역파싱 규칙은 사용자 인터페이스를 통해서 변경한 추상 구문 트리를 화면에 보여줄 때 사용된다. 이를 위해 각 편집기에서 사용되는 역파싱 규칙을 정의하고 역파서가 역파싱 규칙을 읽어 각 편집기에 다양한 형식의 인터페이스를 제공한다.

2.2.1. 역파서 (Unparser)

역파서는 사용자에게 문서의 다양한 뷰를 제공한다. 추상 구문 트리를 모델로 하여 각 편집기에 맞는 인터페이스 형식으로 출력한다. 이를 위해 본 논문에서는 추상 구문 트리를 화면에 출력하는 규칙을 기술한 역파싱 규칙을 정의하였다. 역파싱 규칙은 추상 구문 트리의 뷰를 결정하는 요소로 작동한다.

역파싱 규칙을 두가지를 이용하였다. 하나는 텍스트 기반 편집기를 위한 역파싱 규칙이고 다른 하나는 트리와 다이어그램 기반 편집기를 위한 역파싱 규칙이다. 이 두가지 역파싱 규칙들이 각각 규정되어 있고 이들이 각 편집기의 인터페이스를 결정한다.

텍스트 기반 편집기에서는 선택된 태그의 시작태그와 끝태그를 자동으로 나타나게 하여 태그의 타이핑을 줄이며 현재 태그의 서브 태그를 표시하여 준다. 텍스트 기반 편집기에서 사용되는 역파싱 규칙은 다음과 같다.

$$Vxml\_Tag \rightarrow \alpha \quad @ \ \beta$$

위의 역 파싱 규칙에서 α 와 β 는 출력 태그 정보로써 화면에 출력되는 태그명의 출력 형식을 나타내는 템플릿이며 @은 문법 상 현재 VoiceXML 태그에서 나타날 수 있는 하위 태그를 나타내는 플레이스홀더 (placeholder)를 나타낸다.

텍스트 기반 편집기의 역파싱 규칙에서 출력 태그 정보는 화면에 각 태그를 어떻게 화면에 출력할 것인가를 결정한다. 이를 위해 출력 정보는 시작 태그부와 끝 태그부로 나뉘고 이들은 각각 출력을 제어하는 제어문자와 출력 태그 정보들로 구성 되어 있다. 출력 제어문자로는 탭 문자 (₩t), 개행문자 (₩n), 백탭문자 (₩b)가 있으며 출력 태그 정보로는 출력 태그명를 표시하기 위한 따옴표 (“”)와 태그에 나타나야 할 속성을 표시하기 위한 #이 있다.

두 번째, 트리와 다이어그램 기반 편집기의 역파싱 규칙에서는 문자태그를 출력하는 대신 이미지 아이콘을 출력한다. 이러한 특성으로 출력 제어문자를 기술하지 않고 각 태그를 대표하는 이미지 아이콘만 기술하면 된다. 트리/다이어그램 기반 편집기의 역파싱 규칙은 다음과 같다.

$$Vxml\_Tag \rightarrow \alpha \quad @$$

트리/다이어그램 기반 편집기의 역파싱 규칙에서 α 는

```
〈?xml version="1.0"〉
〈!DOCTYPE vxml SYSTEM "vxml.dtd〉
〈vxml version="1.0"〉·
 〈form〉
  〈field〉
   〈prompt〉
     Hello, World
   〈/prompt〉
  〈/fie d〉
 〈/form〉
〈/vxml〉
```

⇒

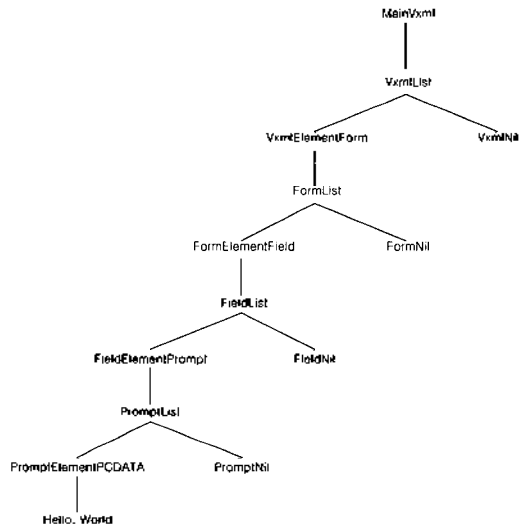

표 1. 텍스트 기반 편집기 역 파싱 규칙의 예
Table 1. Example of text-based editor unparsing rule

vxml-> VxmlList(vxmlElement vxml) { @ \n @} vxmlElement -> VxmlElementCatch (catch) {"(catch"#")" \n\t @ \n\b "(/catch)"}

표 2. 트리/다이어그램 편집기 역 파싱 규칙의 예
Table 2. Example of tree/diagram editor unparsing rule

vxml-> VxmlList(vxmlElement vxml) {@ @} vxmlElement -> VxmlElementCatch (catch) {catch @}

화면에 출력할 이미지 명을 표시하며 @는 현재 VoiceXML 태그에서 나타날 수 있는 하위 태그를 나타내는 플레이스 홀더를 표시한다.

위의 두가지 역파싱 규칙은 표 2와 같은 역파싱 알고리즘을 이용하여 추상 문법과 추상 구문 트리에 적용된다.

[알고리즘 3] 역파싱 알고리즘
Algorithm Unparsing(parent, current)
입력 : 노드
출력 : 역파싱 규칙
정의 :
parent는 parent node
current는 current node
방법 :

```
if (current가 리스트 노드) {
  do {
    추상문법에서 current의 프로덕션에 대한 역파
    싱 규칙을 읽는다.

    if (읽은 역파싱 규칙이 첫 번째 @)
        Unparsing(current, current의 왼쪽 노드
        링크);
    else if (읽은 역파싱 규칙이 두 번째 @)
        Unparsing(current, current의 왼쪽 노드
        링크);
  } while (역파싱 규칙이 남아 있을 경우)
} else {
  do {
    추상문법에서 current의 프로덕션에 대한 역파
    싱 규칙을 읽는다.

    if (추상문법에서 current의 프로덕션 메뉴값이
    STRING)
        parent의 자식으로 STRING을 출력한다.
    else if (읽은 역파싱 규칙이 @) {
        if (current의 프로덕션에 대한 메뉴값이
        STRING)
          parent의 자식으로 current의 왼쪽 노드
          링크를 출력한다.
        else
          Unparsing(current, current의 왼쪽
          랑크 노드)
    }
  } while (역파싱 규칙이 남아 있을 경우) ■
```

알고리즘 3에 적용된 텍스트 기반 편집기의 역파싱 규칙의 예는 표 1과 같다.

알고리즘 3에 적용된 트리/다이어그램 편집기의 역파싱 규칙의 예는 표 2와 같다.

2.3. 제어기

제어기는 명령어 처리기와 트리 조작기로 구성된다. 트리 조작기는 VoiceXML 추상 문법을 참조하여 추상 구문 트리의 노드를 추가, 삭제, 수정을 한다. 명령어 처

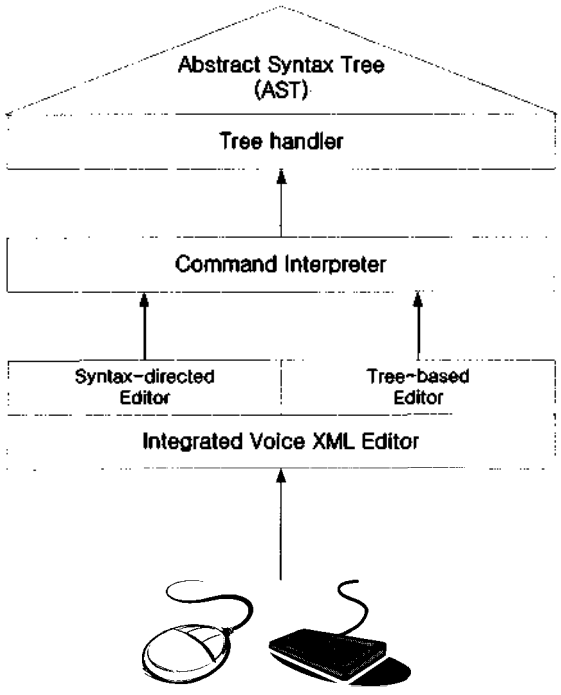

그림 2. 트리조작기, 명령어 처리기와 추상구문트리의 관계
Fig. 2. Relation of tree handler, command interpreter and abstract syntax tree

리기는 사용자 인터페이스를 통해서 내려진 명령을 해석하여 트리 조작기를 호출한다. 그림 2는 추상 구문 트리와 트리 조작기와 명령어 처리기간의 관계도 이다. 명령어 처리기는 입력 장치로부터 받은 명령을 해석하고 이에 따라 상응하는 조건의 명령으로 트리 조작기를 이용하여 추상 구문 트리를 조작한다.

트리 조작기는 사용자가 편집기를 이용하여 문서를 수정하였을 때 작동되는 부분으로 편집기의 핵심 자료구조인 추상 구문 트리를 조작할 때 사용된다. 문서가 수정되었을 때 트리 조작기는 VoiceXML 추상 문법을 참조하여 추상 구문 트리의 노드를 추가, 삭제, 수정을 한다. 노드를 추가할 때는 추상문법을 사용하여 선택된 노드에서 확장 할 수 있는 노드를 찾는다. 트리 조작기에서 가능한 연산은 크게 노드연산과 엘리먼트 연산으로 나눈다. 노드 연산은 파서에서 노드를 생성할 때 사용되는 연산으로 노드의 접근, 삽입, 삭제 연산이 있다. 엘리먼트 연산은 노드 연산의 집합으로 사용자가 입력한 것을 처리할 때 사용되며 엘리먼트의 접근, 삽입, 삭제, 이동, 편집, 속성 접근, 문자열 접근 연산이 있다.

트리 조작기의 기능은 명령어 처리기에서 받은 명령에 따라 현재 구성하고 있는 추상 구문 트리를 생성 및 수정한다.

명령어 처리기는 사용자 인터페이스를 통해서 내려진 명령을 해석하여 트리 조작기를 호출한다. 이때 명령어 처리기에 의해 호출된 트리 조작기는 엘리먼트 연산만 수행한다.

2.4. 다중뷰 편집기 사용자 인터페이스

본 논문에서 구현한 MVC 프레임 워크를 이용한 VoiceXML 다중 뷰 편집기는 텍스트 기반 구문 지향 편집기와 트리 기반 편집기, 다이어그램 편집기와 내부 파서로 구성되어 있다.

다중뷰 편집기는 VoiceXML 문서를 작성하는데 기존의 편집기들이 편집하는 순간 편집하는 문서의 한가지 정보만을 화면에 나타내는 단점을 극복하고 문서의 여러 측면의 정보를 동시에 여러 편집기로 보여 줌으로써 사용자가 문서에 대한 많은 정보를 습득하게 하여 좀 더 정확한 문서를 작성할 수 있게 도와준다. 예를 들어 사용자가 VoiceXML의 내용위주로 편집하면서 작성 중인 문서의 문서구조와 문서의 흐름을 동시에 파악 할 수 있게 한다. 다중 뷰 편집기를 통해서 사용자는 VoiceXML의 DTD에 대한 지식이 없어도 편집기와의 인터페이스를 통해서 문서편집이 가능하며 직관적으로 문서를 여러 측면을 이해할 수 있다.

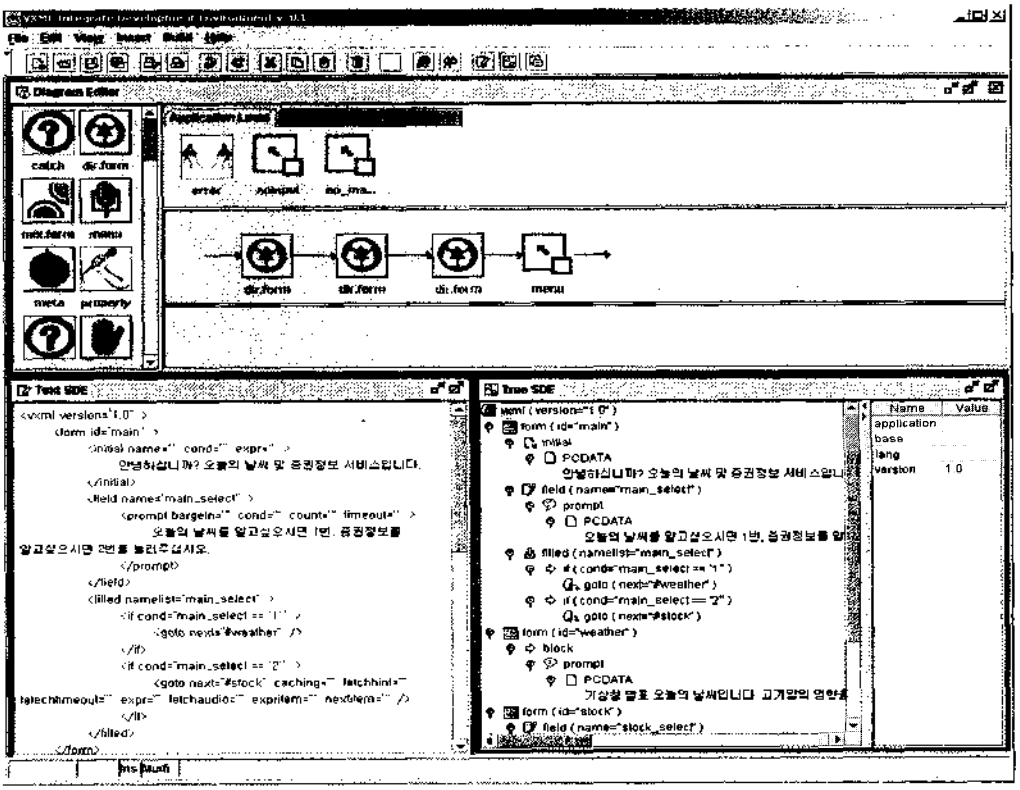

그림 3. VoiceXML 다중뷰 편집기
Fig. 3. VoiceXML multi-view editor

그림 3은 VoiceXML 다중 뷰 편집기 실행 화면이다.

편집기를 통하여 문서가 생성되거나 기존의 문서를 읽었을 때 VoiceXML 다중뷰 편집기는 핵심 자료를 구축하기 위하여 VoiceXML 파서를 이용하여 읽은 문서의 내용을 추상 구문 트리로 생성한다. 이렇게 구성된 추상 구문 트리는 각 편집기에서 읽은 후 역파싱 규칙을 이용하여 편집기에 맞는 인터페이스를 생성하고 이를 화면에 출력한다. 이때 하나의 편집기에서 문서 내용의 변경이 생기면 파서는 이를 즉각적으로 추상 구문 트리에 적용하고 추상 구문 트리가 변경이 되면 즉각적으로 각 편집기에게 알려 수정된 추상 구문 트리에 맞게 인터페이스를 재구성하고 이를 출력한다.

3개의 편집기는 하나의 문서 인스턴스 트리를 이용하기 때문에 한 편집기로 내용을 수정하면 전체 편집기에 동기화가 이루어져 보다 효율적인 편집이 가능하다.

2.4.1. 구문 지향 편집기

구문 지향 편집기는 내용 위주 편집을 제공한다. 이를 위해 편집기의 형태를 일반 텍스트 편집기와 같이 텍스트를 이용하여 문서를 작성할 수 있도록 하였다. 그러나 일반 텍스트 문서의 경우 문서를 작성하는 동안 문법에 대한 적합성을 알 수 없어 문서의 오류가 발생할 가능성이 크고 VoiceXML DTD를 정확히 모르는 사용자의 경우 문서작성이 어렵다. 이러한 이유로 구문 지향 편집기는 문서 작성 시간을 줄이고 올바른 형태의 문서를 작성할 수 편집기 구조를 제공하여야 한다. 이를 위해 구문 지향 편집기에서는 다음과 같은 기능들을 고려하여 구현하였다.

첫째, 사용자가 쉽고 편리하게 문서작성을 할 수 있어야 한다. 키보드 입력을 최소화하고 원하는 요소를 쉽게 넣을 수 있게 하여야 한다.

둘째, pretty printing 기능을 제공하여 VoiceXML 문서의 엘리먼트와 어트리뷰트, PCDATA의 값들을 구분하여 주어 문서의 가독성을 높여주어야 한다.

셋째, 문서를 작성하는 동안이나 문서 작성 후 바로 그 적합성을 검사할 수 있어야 한다.

본 구문 지향 편집기는 입력의 최소화와 원하는 정확한 요소를 쉽게 넣게 하기 위해 텍스트 형태의 문서를 마우스를 이용하여 제어할 수도 있게 하였다. 문서 작성 중 정확한 요소를 삽입하기 위해 마우스 우측 버튼을 이용하여 원하는 요소를 삽입할 수 있게 하였다. 또한, pretty printing 기능을 이용하여 엘리먼트와 속성을 구분하여 주었으며 문서 작성 중 새로운 요소가 추가될 때마다 문서의 적합성을 검사하게 하였다.

이러한 기능을 제공하여 본 구문 지향 편집기는 직관적으로 문서를 빠르고 효율적으로 작성 할 수 있게 하였으며 VoiceXML DTD 구조를 정확히 파악하고 있지 못하여도 쉽게 문서를 작성할 수 있게 하였다.

2.4.2. 트리 편집기

구문 지향 편집기와 같은 텍스트 위주의 문서 작성을 제공하는 시스템은 문서 작성 효율은 높아질 수 있지만 전체적인 문서 구조의 파악은 쉽지 않다. 즉 전체적인 DTD의 구성을 파악하기에는 어려움이 있다. 이러한 단점을 보완하기 위하여 본 논문에서는 트리 형식의 편집기를 제공하였다. 트리 편집기는 VoiceXML 문서를 문서 구조에 초점을 맞추어 작성할 수 있도록 문서를 트리로 표현하여 전체 문서의 구조를 일목요연하게 표시할 수 있게 한다. 그러나 단순한 문서구조의 파악만으로는 문서 작성에 크게 도움이 되지 않는다. 그래서 본 트리 편집기에서는 문서의 구조를 완성한 후 각 엘리먼트의 값이나 속성을 입력할 수 있도록 하여 문서의 편집도 가능하게 하여 효율을 높였다.

트리 편집기는 문서구조 창과 속성 창으로 구성되어 있다. 문서구조 창은 편집하고 있는 문서의 구조를 트리 형식으로 나타내고 있고 속성 창은 문서 구조 창의 각 태그별 속성을 입력, 수정할 수 있는 창을 표시한다. 트리 편집기의 특징은 WYSIWYG 방식을 이용하여 직관적인 편집을 가능하게 한다.

2.4.3. 다이어그램 편집기

VoiceXML은 XML 기반 언어이지만 다른 XML 기반 언어와 다른 특징을 가지고 있다. 그 중 큰 차이점이 시나리오가 있다는 것이다. VoiceXML은 전화 서비스나 음성을 인식하여 사용자에게 적절한 서비스를 제공하기 위해서 개발된 것이다. 이러한 특징 때문에 단순히 문서 내용의 나열이 아닌 사용자와의 상호작용이 필요하다. 이에 어떤 문서내용이 나타나면 그 다음에 어떤 내용이 나와야 하는가가 중요한 문제이다. 현재 많은 XML 편집기들이 있으나 이러한 기능을 제공하지 못하고 있다. 이에 본 논문에서는 VoiceXML 특성에 맞게 단순한 문서의 내용과 문서의 구조를 위주로 편집을 제공하는 이외에 문서의 흐름을 중심으로 문서를 작성할 수 있는 편집기를 추가하였다. 이런 기능을 제공하기 위하여 본 논문에서는 두가지 방식을 사용하였다.

표 3. VoiceXML 다중뷰 편집기 기능비교표
Table 3. A functional comparison table of VoiceXML editors

기능 / 편집기	다중뷰 편집	VoiceXML 문서 자동생성	VoiceXML 문서검증	문서 편집의 동시성	문서의 무결성
VoiceXML 다중뷰 편집기	●	●	●	●	●
Nuance VXML Builder	○	●	●	○	●
XMLSpy의 XML 편집기	○	○	◐	○	●

첫째, 문서의 흐름을 표현하기 위해 DFD (Data Flow Diagram) 방식을 사용하였으며 DFD방식을 표현하기 위해 VPL (Visual Prgramming Language)[6]을 이용하였다. 첫 번째로 DFD는 소프트웨어 공학에서 사용되는 설계 방법 중 하나로 시스템의 구성 요소들 사이의 흐름을 도식화하여 표현할 수 있는 방법으로 VoiceXML의 문서 흐름을 표현하기에는 최적의 방법이다. DFD 방식을 사용함으로써 VoiceXML의 엘리먼트들 사이의 흐름을 도식화하여 표현할 수 있다.

둘째, DFD의 흐름을 도식화하기 위해 VPL을 사용하였다. DFD 방식은 흐름을 표현하기에는 좋지만 일반인들이 이해하고 사용하기에는 어려움이 많다. 이러한 이유로 DFD를 표현하는 방식으로 VPL 방식을 이용하였다. VPL은 아이콘과 같은 시각정보를 이용하여 프로그래밍하는 방식이다. 아이콘과 같은 시각정보의 사용은 인간이 사고하는 방식과 유사하여 사용자가 프로그래밍을 할 때 그들의 생각을 쉽게 이해하고 쉽게 표현할 수 있는 장점을 가진다.

다이어그램 편집기를 이용하여 사용자는 아이콘화 되어 있는 엘리먼트들을 나열하고 엘리먼트들간의 순서를 화살표로 지정하여 문서의 흐름 위주로 문서를 작성할 수 있다.

III. 결 론

본 논문에서는 VoiceXML 문서를 효율적으로 작성하고 문서의 적합성을 검증하기 위한 다중 뷰 편집기를 설계, 구현하였다. 표 3에서와 같이 기존의 XML 편집기들이 문서의 구조 중심의 편집에 치중하여 문서를 작성하였다. 반면 다중 뷰 편집기는 문서의 여러 가지 정보를 사용자에게 동시에 제공하여 사용자가 문서 작성 시 다양한 문서 정보를 획득할 수 있도록 하였다. 편집기 사용자가 문서의 다양한 측면을 인지하면서 문서를 작성할 경우 문서에 대한 이해도가 향상되어 문서 작성 효율을 높일 수 있다. 이러한 이유로 다중 뷰 편집기는 동시에 문서의 내용, 문서의 구조 그리고 문서의 흐름을 보여주도록 하였다. 이런 동시성을 다중 뷰를 제공하기 위해 본 논문에서는 MVC (Model-View-Controller) 프레임 워크를 이용하였다. MVC 프레임워크는 시스템을 모델 (Model), 뷰 (View), 제어기 (Controller)로 분리, 설계할 수 있도록 하여 하나의 모델을 이용하여 여러 가지 다른 뷰를 동시에 보여줄 수 있게 하였다.

MVC 프레임워크를 VoiceXML 다중 뷰 편집기에 적용하여 개발환경을 모델, 뷰, 제어기로 분리하여 개발하였다. MVC의 구성 요소중 모델부분은 개발환경의 핵심 자료구조로 작성하는 문서의 내용을 트리로 표현한 추상 구문 트리와 추상 구문 트리를 작성하기 위해 VoiceXML의 DTD를 변경한 추상 문법으로 구성되며 뷰 부분은 핵심 자료구조를 다양한 형식으로 출력하기 위한 방법으로 각각의 편집기마다 역파싱 규칙과 역파서로 구성되었다. 마지막으로 제어기 부분는 뷰에서 사용자의 입력을 제어하는 부분으로 트리 조작기와 명령어 처리기로 구성되었다.

문서 개발 환경에서 MVC 프레임워크를 응용하면 하나의 문서를 동시에 여러 가지 형태로 보여줄 수 있다. 그럼으로써 사용자는 작성중인 문서의 다양한 정보를 인지하면서 문서를 작성할 수 있다. 이런 특성은 VoiceXML을 작성하는데 정확성과 편의성을 제공하여 문서 편집의 효율성을 증대시킨다. 또한 여러 뷰를 가진 시스템이 하나의 모델을 가짐으로써 다양한 뷰의 편집기들의 자료 구조의 일관성을 유지할 수 있어 신뢰성 있는 문서의 편집을 제공한다.

참 고 문 헌

1. W3C, "Voice Browser Activity - Voice Enabling the Web!", http://www.w3.org/voice.
2. VoiceXML Forum, "Voice Extensible Markup Language

(VoiceXML) Version 1.0", http://www.voicexml.org., March 7, 2000.
3. Paul Houle, et al., Early Adopter VoiceXML, Wrox Press, 2001.
4. David Hunter et al., Beginning XML, Wrox Press, 2000.
5. Thomas W. Reps and Tim Teitelbaum, *The Synthesizer Generator : A system for constructing language-based editors*, Springer-Verlag, 1989.
6. Martin Erwig, "A Visual Language for XML", *16th IEEE Symp.* on VL2000, 2000.
7. Alfred V. Aho, et al., Compilers: Principles, Techniques, and Tools, Addison-wesley, 1986.
8. Daive A. Watt, *Programming Language Processors*, Prentice -Hall, 1993.
9. Janet D. Hartman and Joaquin A. Vila, "VoiceXML Builder: A Workbench for investigating Voiced-Based Applications", 31st ASEE/IEEE Frontiers in Education Conference, October 2001.
10. Kang Zhang, Da-Qian Zhang and Jiannong Cao, "Design, Construction, and Application of a Generic Visual Language Generation Environment, IEEE Transactions on SE, 27(4), April, 2001.
11. Peter J. Danielsen, "The Promise of a Voice-Enabled Web", *IEEE Computer*, August, 2000.
12. 신현경, 강동남, 염세훈, 유재우, "음성 웹 서비스를 위한 VoiceXML 해석기의 설계 및 구현", 한국 음향 학회지, 20(4), pp.42-47, May, 2001.

감사의 글

본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음.

저자 약력

• **염 세 훈 (Sae-Hun Yeom)**

1992년 2월: 국립 서울 산업대학교 전자계산학과 졸업(공학사)
1992~1994년: 숭실대학교 컴퓨터학과(공학 석사)
1994~1999년: 숭실대학교 컴퓨터학과 박사과정 수료
2002년 3월~현재: 동서울 대학 컴퓨터소프트웨어과 전임강사
※주관심분야: 컴파일러, 프로그래밍언어, 인간과 컴퓨터 상호작용, Voice(Speech) Based Markup Language, XML

• **유 재 우 (Chae-Woo Yoo)**

1976년 2월 : 숭실대학교 전자계산학과(공학사)
1978~1985년: 한국 과학기술원 전산학과(공학석사, 공학박사)
1986~1987년: Cornell Univ. Visiting Scientist
1996~1997년: Univ. of Pittsburgh Visiting Scientist
1983년~현재: 숭실대학교 컴퓨터학부 정교수
※주관심분야: 컴파일러, 프로그래밍 환경, 인간과 컴퓨터 상호작용, SGML, XML