SALT 기반 음성 웹 페이지의 자동 생성
(Automatic Generation of Voice Web Pages Based on SALT)

고 유 정 † 김 윤 중 ††
(Youjung Ko) (Yoonjoong Kim)

요 약 음성 브라우저가 등장함에 따라, 음성 대화 어플리케이션이 웹 환경에서 사용이 가능하게 되었다. 음성 대화 어플리케이션은 음성 웹 페이지로 구성되어 있다. 음성 웹 페이지의 대화 스크립트는 SALT(Speech Application Language Tags) 언어 등으로 기술되어야 한다. 기존 웹 페이지들은 음성 대화를 고려하지 않고 시각용(visual)용으로 제작되었지만, 이들 웹 페이지에도 음성 대화를 이용하여 처리할 수 있는 요소들이 있다. 따라서 본 논문에서는 시각용으로 제작된 HTML 웹 페이지로부터 대화처리가 가능한 요소들을 추출하고 해당대화를 SALT로 생성해내는 음성 웹 페이지의 자동 생성방법을 제안하였다. 제안한 음성 웹 페이지의 자동 생성기는 어휘 분석기와 구문 분석기로 구성된 번역기로, HTML로 기술된 웹 페이지를 HTML+SALT로 기술된 음성 웹 페이지로 변환한다. 변환된 음성 웹 페이지는 기존의 마우스, 키보드를 이용한 처리도 가능하고 음성 대화 처리도 가능하도록 설계되었다.

키워드 : 음성 웹 페이지, SALT, 음성 사용자 인터페이스, 음성 브라우저

Abstract As a voice browser is introduced, voice dialog application becomes available on the Web environment. The voice dialog application consists of voice Web pages that need to translate the dialog scripts into SALT(Speech Application Language Tags). The current Web pages have been designed for visual. They, however, are potentially capable of using voice dialog. This paper, therefore, proposes an automated voice Web generation method that finds the elements for voice dialog from Web pages based HTML and converts them into SALT. The automatic generation system of a voice Web page consists of a lexical analyzer and a syntactic analyzer that converts a Web page which is described in HTML to voice Web page which is described in HTML+SALT. The converted voice Web page is designed to be able to handle not only the current mouse and keyboard input but also voice dialog.

Key words : Voice Web Page, SALT, Voice User Interface, Voice Browser

1. 서 론

음성합성 및 인식기술의 발전으로 음성기술을 이용한 웹 브라우저의 개발 연구가 가속화되고 있으며, W3C에서는 그 중요성을 인식하여 음성 브라우저(Voice Browser)에 대한 권고안을 발표하였고 지속적으로 갱신되고 있다[1,2]. 음성 브라우저 작업그룹에서는 음성 브라우저를 위한 음성 인터페이스 구조(Speech Interface Framework)를 정의하였다. 음성 인터페이스의 한 구성요소인 대화처리기(Dialog Manager)는 대화스크립트를 받아, 그 내용에 따라 대화를 진행시킨다. 이 스크립트를 기술할 수 있는 음성 언어로는 W3C의 VoiceXML(Voice eXtensible Markup Language)과 Microsoft의 SALT (Speech Application Language Tags)가 있다[3,4].

이와 같은 음성 언어의 출현과 함께 웹 기반 대화형 응용프로그램에서는 음성 언어가 HTML을 대치할 것이며, 마우스와 키보드 대신 음성이 사용될 것이다[5]. 이러한 연구 사례로 Hao Shi(2006)은 SALT 언어를 이용하여 음성 웹 사이트를 구축하였다[6]. 이 논문에서는 음성 웹 사이트를 구축함으로써 웹 기반에서 음성으로

† 정 회 원 : 한밭대학교 컴퓨터공학과
youlony@hanbat.ac.kr
†† 종신회원 : 한밭대학교 컴퓨터공학과 교수
yjkim@hanbat.ac.kr
논문접수 : 2009년 2월 5일
심사완료 : 2009년 12월 18일

정보과학회논문지: 소프트웨어 및 응용 제37권 제3호(2010.3)

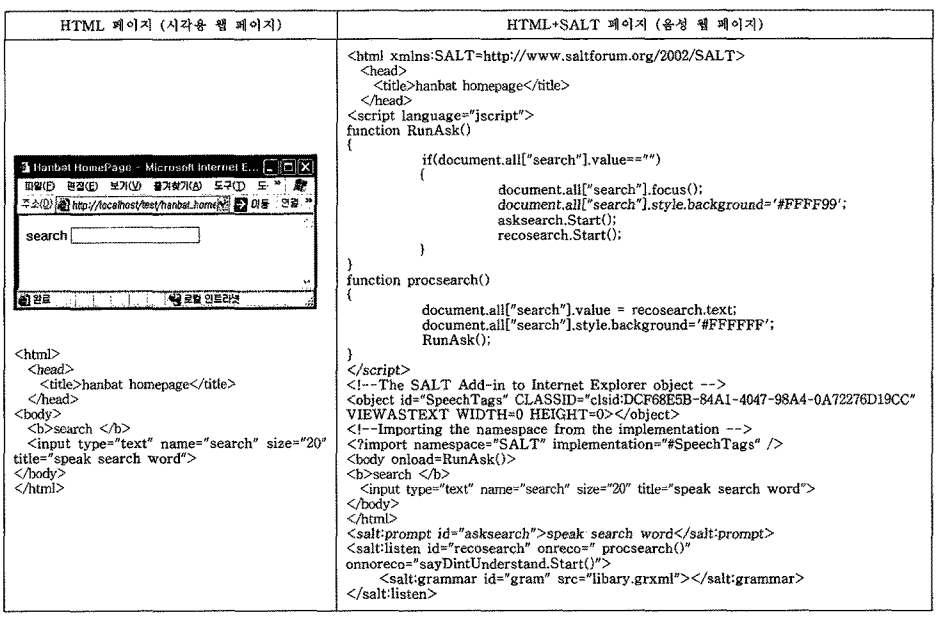
물건을 주문할 수 있는 보다 편리한 인터페이스를 제공하지만, 수많은 시각용 웹 페이지를 수동으로 음성 웹 페이지로 구축하기에는 많은 시간과 비용이 따른다. 기존의 웹 페이지를 음성 웹 페이지 환경에 맞게 효과적으로 자동 변환할 수 있다면 비용의 절감을 가져 올 수 있다. 이러한 필요성에 의해 기존 웹 페이지를 음성 웹 페이지로 자동 변환하기 위한 연구가 진행되고 있다. 국외논문으로 Hiyan Shao(2003)은 HTML을 VoiceXML로 변환하는 방법을 제안하고 있다[7]. 이 연구에서는 IBM의 WTP(WebSphere Transcoding Publisher)을 이용하여 변환하고 있다. 그러나 WTP를 사용할 경우 HTML 문서에서 VXPL(VoiceXml Precursor Language)로, VXPL에서 VoiceXML로 변환하는 두 단계를 거쳐서 번역해야 하며, WTP에서 정의된 요소들만 번역할 수 있다는 한계점이 있다. 국내논문으로 오지영(2008)은 HTML 웹 페이지를 VoiceXML 웹 페이지로 자동 변환하는 대화 스크립트 생성기를 개발하였다[8]. 하지만 변환한 VoiceXML 웹 페이지는 인터넷 익스플로러 브라우저에서는 지원되지 않아 별도의 응용 프로그램이나 오페라(Opera) 브라우저가 필요하다. 또한 HTML 웹 페이지의 링크 태그만을 대상으로 하고 입력 태그는 고려되지 않았으며, 번역기로 구현되어 있지 않아 확장의 어려움이 있다.

따라서 이와 같은 문제점을 보완하기 위하여 본 논문에서는 인터넷 익스플로러 브라우저에서 지원하는 SALT 언어를 이용하여 HTML 웹 페이지를 HTML+SALT 웹 페이지로 변환 하고, 음성 웹 페이지 자동 생성시스템을 번역기로 구현하여 확장이 용이하도록 하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 설계한 음성 웹 페이지 자동 생성 시스템의 구성을 살펴보고, 3장과 4장에서는 HTML로 기술된 시각용 웹 페이지를 HTML+SALT로 기술된 음성 웹 페이지로 자동 변환하는 시스템의 설계 및 구현 방법에 대해 설명한다. 5장에서는 실험 및 결과를 통해 시스템을 성능을 알아보며 6장에서는 결론 및 향후 연구 방향에 대해 설명하도록 한다.

2. 음성 웹 페이지 자동 생성 시스템

그림 1은 시각용 웹 페이지를 음성 웹 페이지로 변환하는 과정을 설명하기 위한 샘플 예제이다. HTML 페이지는 하나의 입력박스가 있는 시각용 웹 페이지이고, HTML+SALT 페이지는 시각용 웹 페이지에 SALT로 구성된 음성 대화 스크립트와 대화의 흐름을 제어하는 자바스크립트가 삽입된 음성 웹 페이지이다. 그림 1의 시각용 웹 페이지에서 굵은 글씨는 음성요소 정보를 포함하고 있으며 음성 웹 페이지에서 굵은 글씨는 추출된 음성 요소 정보를 이용하여 생성된 코드이다. 시각용 웹 페이지는 정보를 검색하기 위해 키보드와 마우스를 이용하여 입력 박스에 단어를 입력하지만, 음성 웹 페이지는 "speak search word"라는 음성 안내 메시지가 나오



HTML 페이지 (시각용 웹 페이지)


```
<html>
  <head>
    <title>hanbat homepage</title>
  </head>
<body>
  <b>search </b>
  <input type="text" name="search" size="20"
title="speak search word">
</body>
</html>
```

HTML+SALT 페이지 (음성 웹 페이지)

```
<html xmlns:SALT=http://www.saltforum.org/2002/SALT>
  <head>
    <title>hanbat homepage</title>
  </head>
<script language="jscript">
function RunAsk()
{
        if(document.all["search"].value=="")
        {
                document.all["search"].focus();
                document.all["search"].style.background='#FFFF99';
                asksearch.Start();
                recosearch.Start();
        }
}
function procsearch()
{
        document.all["search"].value = recosearch.text;
        document.all["search"].style.background='#FFFFFF';
        RunAsk();
}
</script>
<!--The SALT Add-in to Internet Explorer object -->
<object id="SpeechTags" CLASSID="clsid:DCF68E5B-84A1-4047-98A4-0A72276D19CC"
VIEWASTEXT WIDTH=0 HEIGHT=0></object>
<!--Importing the namespace from the implementation -->
<?import namespace="SALT" implementation="#SpeechTags" />
<body onload=RunAsk()>
<b>search </b>
  <input type="text" name="search" size="20" title="speak search word">
</body>
</html>
<salt:prompt id="asksearch">speak search word</salt:prompt>
<salt:listen id="recosearch" onreco=" procsearch()"
onnoreco="sayDintUnderstand.Start()">
      <salt:grammar id="gram" src="libary.grxml"></salt:grammar>
</salt:listen>
```

그림 1 HTML 웹 페이지와 HTML+SALT 웹 페이지

고 사용자는 음성으로 검색하고자 하는 단어를 발음하면 자동으로 입력박스에 입력된다.

이와 같이 시각용 웹 페이지에는 음성으로 처리될 수 있는 선택 박스, 체크 박스, 정보를 입력하는 텍스트 박스 등이 있다. 다음은 HTML 페이지에서 음성으로 처리 가능한 요소들의 규칙을 정의하였다. 본 논문에서는 음성 안내 메시지를 프롬프트라 한다.

규칙 1. input태그이고 type="text"이거나 type="password"인 경우와 textarea태그인 경우

- input태그와 textarea태그의 배경 색깔을 변경하고 포커스를 셜정한다.
- title 속성에 값이 있으면 이 속성의 문자 정보를 추출하여 프롬프트를 출력한다.
- 음성출력 요소가 없는 경우 "포커스가 있는 곳에 입력할 단어를 말하시오"라는 프롬프트를 출력한다.
- type="password"인 경우는 비밀번호를 음성으로 발음할 경우 유출될 수 있는 요소가 있기 때문에 변환시 필요에 따라 프롬프트 출력 여부를 결정할 수 있게 한다.

규칙 2. input태그이고 type="radio"이거나 type="checkbox"인 경우

- 선택 항목 정보는 <label></label>태그사이나 input 태그가 끝난 다음에 문자로 표기되어 있다.
- 프롬프트는 선택 항목 정보를 추출하여 출력한다.

규칙 3. select태그인 경우

- <option></option>태그 사이의 선택 항목 정보를 추출하여 프롬프트로 출력한다.
- 예를 들어, 학과를 선택하는 경우 "컴퓨터 공학과, 기계공학과, 정보통신공학과 중 하나를 선택하세요." 라는 프롬프트 를 출력한다.

공통 규칙으로는 안내가 길어지는 프롬프트를 제어할 수 있도록 bargin 기능을 사용하여 프롬프트가 끝나기 전에 응답할 수 있는 기능이 있다. 이와 같은 규칙에 의해 HTML 웹 페이지로부터 음성 요소를 추출한다.

그림 2는 음성 웹 페이지 자동 생성 시스템의 전체 구성도이다. 음성 웹 페이지 자동 생성 시스템은 컴파일러 생성기과 자동 생성기로 구성되어 있다. 컴파일러 생성기(IISPLGenerator)는 어휘 분석기와 구문 분석기를 생성하는 기능을 가진 도구로, 본 연구실에서 개발하였다[9]. 컴파일러 생성기는 C# 언어로 쓰여진 파일을 출력하며, 필수적인 사항은 클래스에서 상속받아 쓸 수 있도록 구성되어 있다. 자동 생성기는 하나의 입력과 출력을 가지며 어휘 분석기와 구문 분석기로 구성되어 있다.

HTML+SALT 어휘 분석기는 어휘 분석기 생성기에서 HTML 토큰 기술 파일을 입력으로 받아 생성된다. 생성된 어휘 분석기는 HTML 웹 페이지를 입력으로 받

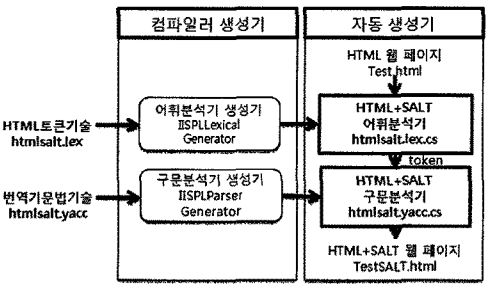


그림 2 음성 웹 페이지 자동 생성 시스템의 구성

아 음성요소에 필요한 태그들을 분석하고, 태그와 텍스트를 분리한 후 불필요한 태그들과 주석을 제거하고 토큰(token)을 반환한다. HTML+SALT 구문 분석기는 구문 분석기 생성기에서 생성규칙을 정의한 번역기 문법 파일을 입력으로 받아 생성된다. 생성된 구문 분석기는 토큰을 입력으로 받아 각 태그에 해당하는 속성들과 문법을 검사한다. 구문분석(parsing)이 완료되면, 생성규칙에 의해 대응되는 SALT 코드를 생성한다.

제안한 음성 웹 페이지의 자동 생성 시스템은 다음과 같은 기능을 가진다.

1) HTML로 기술된 시각용 웹 페이지에서 음성 대화로 처리 가능한 선택 및 입력태그 요소를 추출한다.
2) 추출된 요소 정보를 이용하여 SALT로 음성 대화 스크립트를 생성한다.
3) 이 스크립트를 시각용 웹 페이지에 삽입하여 음성 웹 페이지를 완성한다.

3. HTML+SALT 어휘 분석기

HTML+SALT 어휘 분석기는 HTML 웹 페이지에서 SALT 코드로 변환시 필요한 토큰을 찾아내는 기능을 한다. HTML에서 음성 요소들을 포함하는 input, select, textarea태그와 속성을 정의하기 위해 19개의 토큰을 정의한다. 그림 3은 정의된 토큰의 형태를 묘사한 정규표현과 각 정규 표현이 매칭되었을 때 처리를 나타내는 액션(action)을 기술한 코드이다. 즉, 규칙 부분에서 html return("HTML");}은 html과 일치되는 문자열이 인식되었을 때 HTML토큰을 반환하도록 기술된 부분이다.

사용자 프로그램 부분은 Test함수로 구성되어 있다. Test함수는 HTML 페이지를 입력으로 받아 입력 파일의 끝에 도달 할 때까지 토큰을 생성하며 파일의 끝에 도달하면 null을 호출한다. 스캐너 역할을 담당하는 lexan함수는 정규표현과 일치하는 토큰을 입력 문자열에서 찾을 때까지 한 번에 한 문자씩 계속 읽어 들인다.

어휘 분석기 생성기는 어휘 분석기를 자동으로 생성하는 도구이며, HTML 토큰 기술을 입력으로 받아 기술된

```
%{
        //   htmlsalt.lex
%}
//이름 정의 부분
str         \"[^"\n]*\"
delim       [ \t\r\n]
ws          {delim}+
letter      [A-Za-z]
digit       [0-9]
idl         {letter}({letter}|{digit})*
code        [<>=/]
%%
//규칙 부분
{ws}        {return("WS");}
html        {return("HTML");}
body        {return("BODY");}
input       {return("INPUT");}
label       {return("LABEL");}
name        {return("NAME");}
type        {return("TYPE");}
title       {return("TITLE");}
id          {return("ID");}
select      {return("SELECT");}
option      {return("OPTION");}
textarea    {return("TEXTAREA");}
{str}       {return("STR");}
{idl}       {return("IDL");}
{code}      {return lexbuf;}
:           {return("COLON");}
;           {return("SEMICOLON");}
.           {return("WS");}
%%
//사용자 프로그램 부분
public void Test(string fn)
{
        install(fn);
        string t;
        do
        {
                t=(string)lexan();
                Console.WriteLine("["+t+"]"+lexbuf);
        } while(!isDone());
}
```

그림 3 HTML의 토큰 기술

```
%{
        /*   htmlsalt.yacc */
        SpeechWeb speech = new SpeechWeb();
%}

%token  HTML BODY INPUT NAME TYPE TITLE ID IDL STR < > / =  SELECT OPTION TEXTAREA LABEL
COLON SEMICOLON

%%
pages       :  < HTML xml > htmlcontent < /  HTML >
            |
            |error < {}
            ;
htmlcontent : htmlcontent < BODY attr > body < /  BODY >  {speech.SaltPage();}
            |
            | error < {}
            ;
body        : body < INPUT attr closetag        {speech.AddInput("input",T4); }
            | body < INPUT attr closetag IDL {speech.AddInput("input",T4+"str:"+T6); }
            | body < SELECT attr > optionlist < / SELECT >{speech.AddSelect("select", T4, T6); }
            | body < TEXTAREA attr > anystr < / TEXTAREA >    {speech.AddInput("textarea",T4); }
            | body < LABEL attr > anystr < / LABEL > {T0="label:"+T6; }
            |
            | error < {}
            ;
attr        : attr NAME = attrval     {T0=T1+T2+T3+T4+";";}
            | attr TYPE = attrval     {T0=T1+T2+T3+T4+";";}
            | attr TITLE = attrval    {T0=T1+T2+T3+T4+";";}
            | attr ID = attrval       {T0=T1+T2+T3+T4+";";}
            | attr IDL =  attrval
            |
attrval : IDL   {T0=T1;}
        | STR   {T0=T1;}
        ;
closetag : > | / > ;
optionlist : optionlist < OPTION attr > anystr < / OPTION > {T0= T4+"str:"+T6+"/";}
            |
            ;
anystr      : anystr anystr0
            |
            ;
anystr0 : HTML | BODY | INPUT | NAME | TYPE | TITLE | ID | IDL | STR
        | / | = | SELECT | OPTION | TEXTAREA | LABEL | COLON | SEMICOLON
        ;
xml         : IDL COLON anystr | ;
%%
%{
public void Test(string fn)
{
        lexan.install(fn);
        lexan.ToLower();
        Parse();
}
%}
```

그림 4 HTML+SALT 번역기 문법 기술 코드

형태의 토큰을 찾아내는 HTML+ SALT 어휘 분석기를 생성한다. 생성된 어휘 분석기는 61개의 상태를 가지며 시작은 0번째 상태부터 시작하고 도달상태(initial state)는 60개를 가지고 있으며, 전이 테이블(transition table) 기능을 하는 Install함수와 도달 상태에 따라 토큰을 반환하는 Action함수로 구성되어 있다.

4. HTML+SALT 구문 분석기

HTML+SALT 구문 분석기는 토큰을 입력으로 받아 HTML+SALT 웹 페이지를 생성하는 번역기이다. 정의된 생성 규칙에 의해 HTML 문법을 파싱하며 패턴(pattern)이 발견되면 의미 규칙에 의해 SALT 코드를 생성하는 번역함수를 호출한다.

그림 4는 HTML의 SALT로 번역하는 문법 기술 코드이며, HTML의 오류 복구 생성 규칙에 정의된 문법 심벌을 이용하여 그 생성 규칙에 해당하는 의미 규칙을 기술하였다. 이와 같은 의미 규칙은 Parse함수에 의해 호출되어 실행된다. 본 논문에서 구문 분석기는 LR(Left to Right scanning) 파싱 알고리즘을 사용하였고, shift 행동에 HTML 어휘 분석기를 호출하여 토큰을 얻고 reduce 행동에 적용된 생성 규칙과 일치하는 의미 수행 코드를 실행한다. 생성 규칙은 HTML 웹 페이지에서 음성 정보 요소를 포함하고 있는 input, select, textarea, label 태그의 구조를 파악하여 10개 규칙을 정의하였고, 의미 규칙은 생성 규칙에 있는 기호와 속성을 이용하여 SALT로 변환하기 위해 기술하였다.

구분 분석기는 입력받은 토큰이 정의된 문법의 문장이라면 구문 분석 정보를 생성하고 정의된 문법의 문장이 아니라면 에러 메시지를 출력하는 작업을 한다. pages 생성규칙에서 오류 복구 규칙은 HTML 구조가 아닌 입력을 배제하기 위하여 error처리를 수행하며, htmlcontent 오류 복구 규칙은 HTML 문서의 body 구조가 아닌 입력을 배제하기 위하여 error처리를 수행한다. body 오류 복구 규칙은 input, select 및 textarea, label 구조가 아닌 모든 구조를 버리도록 동작하기 위해 error처리를 수행하도록 설계하였다. 여기서 body생성규칙은 body태그 안에 select, input, textarea, label태그가 올 수 있으며 이에 해당하는 문법의 구조를 정의하고 있다. 이와 같이 정의된 태그를 만나면 SALT의미 규칙을 호출한다.

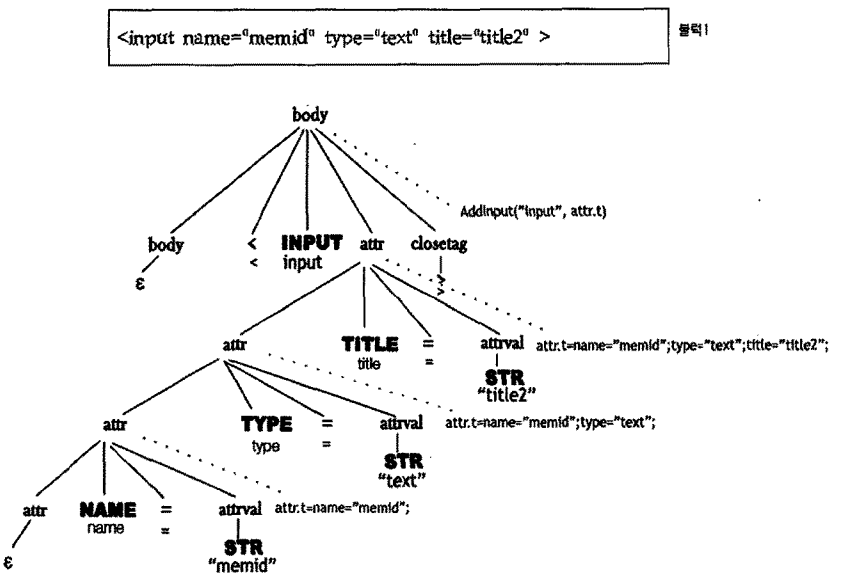


그림 5 SALT 의미 규칙에 의해 번역하는 구문트리

그림 5는 블럭1의 코드가 입력되었을 때 SALT 의미 규칙에 의해 번역되는 과정을 보여주는 구문트리(parse tree)이다. 먼저 입력된 토큰열을 파싱하고 구문트리를 만들며, 구문트리를 운행하면서 각 노드에 있는 SALT 의미 규칙을 계산한다. 생성규칙 attr에 의해 블럭1의 패턴이 발견되면 attr.t의 의미규칙을 호출한다. 의미규칙에서 첨자 val은 계산된 값을 기억하고 있는 속성이며, t는 기억장소를 나타낸다. 이 노드에 의미 규칙 attr.t :=attr.t+name.val+"="+attrval.val+":"을 적용하면 attr.t은 공백 문자열값, name.val은 name값을 가지며, attrval.val은 "memid"를 가진다. 그러므로 arrt.t은 이 노드에서 값 name="memid"을 가지게 된다. 이와 같이 구문트리를 상향(bottom up)으로 운행(traverse)하면서 각 노드에 있는 의미규칙을 계산한다. 최종적으로 트리의 루트노드인 body.t 의미규칙에 의해 번역함수인 AddInput을 호출함으로써 SALT코드를 생성한다.

구문 분석기 생성기는 언어의 문법 표현으로부터 구문 분석기를 자동으로 생성하는 도구이며, HTML문법 규칙과 그에 해당하는 액션을 기술한 코드를 입력으로 받아 HTML+SALT 구문 분석기를 생성한다. 생성된 구문 분석기는 Action함수와 Install함수로 구성된다. Action함수는 생성 규칙에 의해 대응되는 번역함수를 호출하는 기능을 수행하고 Install함수는 파싱테이블 기능을 한다. Install함수는 19개의 토큰과 46개의 생성 규칙을 갖는 문법 및 11개의 비단말 생성 규칙과 3개의 오류복구규칙(error 규칙)으로 구성되었다. HTML 문서에서 input, select 및 textarea 태그를 인식하고 번역하도록 구성되었다.

생성된 구문 분석기의 파싱테이블은 101개의 상태수와 19개의 입력 심볼 및 11개의 비단말기호로 구성된다. HTML 문서 중에서 인식대상의 태그가 아닌 것은 모두 버리도록 하기 위하여 오류복구용 특별생성 규칙 error를 사용하였다. 파싱테이블의 상태 0, 8, 38에서 오류복구처리를 수행한다. 오류 복구 처리 기능은 해당 구조의 패턴이 아니면 즉, 파싱 에러가 발생되면 파싱 스택의 상태를 오류 복구 상태가 나올 때까지 버리고 입력은 < 문자가 나타날 때까지 다시 파싱을 계속한다.

첫 번째 pages → error < 의 경우는 0번 상태이후 HTML의 구조가 아닌 입력이 발견되면 실행을 중지하고 스택을 0번까지 추출(pop)하여 버리고 < 문자가 입력될 때까지 전진하고 다시 실행을 계속한다.

두 번째 htmlcontent → error < 의 경우는 8번 상태에서 HTML 문서의 body 구조가 아닌 입력을 배제하기 위하여 상기와 같은 처리를 수행한다.

세 번째 body → error < 의 경우는 38번 상태에서 input, select, textarea 및 label 구조가 아닌 모든 구조를 버리도록 동작한다.

SALT 의미 규칙에 의해 요청되는 번역함수는 SpeechWeb클래스에 정의하였다. 그림 6은 HTML+SALT 코드를 생성하는 SpeechWeb 클래스의 구조이다. SpeechWeb클래스는 구문 분석기에서 HTML+SALT 코드로 번역하기 위해 참조되며, 각 함수의 주요 기능은 표 1과 같다.

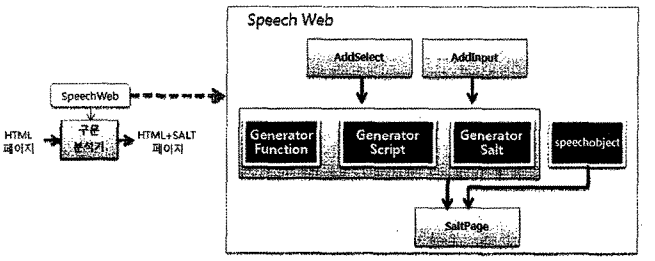


그림 6 HTML+SALT 코드 생성하는 SpeechWeb 클래스 구조

표 1 번역 함수의 기능

함수명	입력	출력
void AddSelect, AddInput	tagname, info	GeneratorScript,GeneratorFunction, GeneratorSALT 호출
string GeneratorFunction	tagname, type, name	음성 인식된 후에 수행할 기능을 정의한 자바스크립트 함수를 생성
string GeneratorScript	tagname, type, name, title	대화 흐름 제어하는 자바스크립트 생성
string GeneratorSALT	tagname, type, name, title	음성 출력하는 prompt와 음성 입력받는 listen 엘리먼트 생성
string speechObject		Microsoft Speech Add in 코드 생성
void SaltPage		상기에 기술한 함수들에 의해 반환된 값을 HTML 웹 페이지에 삽입

6. 실험 및 결과

본 연구에서는 시각용 웹 페이지를 음성 대화가 가능한 음성 웹 페이지로 변환하기 위한 자동 생성기 시스템을 설계 및 구현하였다. 본 장에서는 구현한 시스템의 성능을 확인하기 위하여 테스트용 웹 페이지를 생성한 후 음성 웹 페이지로 정확히 변환되고 인터넷 익스플로러 웹 브라우저에서 정상적으로 동작하는지를 확인한다.

6.1 실험 환경 및 절차

구현된 음성 웹 페이지 자동 생성기는 HTML+SALT 어휘 분석기와 HTML+SALT구문 분석기로 구성되어 있다. 먼저 각각의 어휘 분석기와 구문 분석기의 성능을 분석한 후 음성 웹 페이지 자동 생성기 시스템의 성능을 실험하였다.

음성 웹 페이지 자동 생성기 실험 절차는 다음과 같다.

단계 1. 자동 생성기 시스템에 시각용 웹 페이지를 입력한다.

단계 2. 자동 생성기 시스템이 입력된 시각용 웹 페이지를 음성 웹 페이지로 정확히 생성하는지 확인한다.

단계 3. 생성된 음성 웹 페이지는 Speech Add-in이 설치된 인터넷 익스플로러 브라우저에서 실행한다.

단계 4. 사용자가 음성을 통해 웹 페이지를 제어할 수 있는지 확인한다.

생성된 음성 웹 페이지는 윈도우 XP 운영체제에 Speech Add-in for Internet Explorer를 설치하고, 마이크는 UNI-DIRECTIONAL MIC DM-20SL를 사용하였다. Speech Add-in for Internet Explorer를 설치하면 로컬 컴퓨터에 음성 인식과 합성 엔진이 탑재되어 인터넷 익스플로러 브라우저에서 음성 웹 페이지의 실험이 가능하다.

6.2 음성 웹 페이지 자동 생성기 실험

먼저 구현된 HTML+SALT 어휘 분석기가 HTML 웹 페이지에서 SALT 코드로 변환시 필요한 토큰을 정확히 찾아내는 기능을 수행하는지 실험하였다. 그림 7은 블럭1의 HTML 웹 페이지 코드를 어휘 분석기에 입력하여 토큰이 추출된 결과의 일부이다. 어휘 분석기는 정의된 input태그와 속성들의 태그를 찾아 정확히 토큰으로 반환하는 것을 확인할 수 있다.

다음은 HTML+SALT 구문 분석기가 그림 7의 토큰을 입력 받아 HTML+SALT 음성 웹 페이지를 정확히

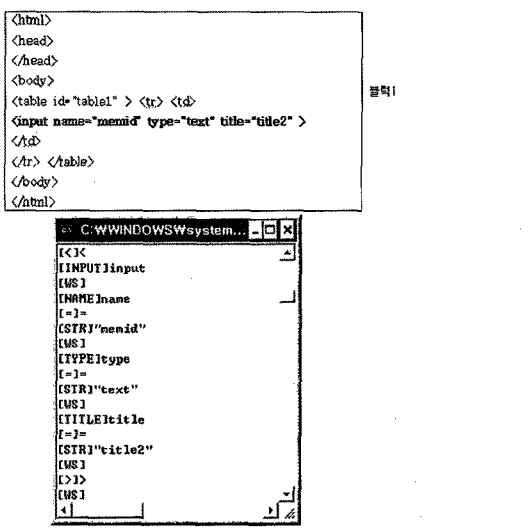


그림 7 HTML+SALT 어휘 분석기로 추출한 토큰

```
<html xmlns:SALT=http://www.saltforum.org/2002/SALT>
  <head>
  </head>
<script language="jscript">
function RunAsk()
{
        if(document.all["memid"].value=="")
        {
                document.all["memid"].focus();
                document.all["memid"].style.background='#FFFF99';
                askmemid.Start();
                recomemid.Start();
        }
}
function procmemid()
{
        document.all["memid"].value = recomemid.text;
        document.all["memid"].style.background='#FFFFFF';
        RunAsk();
}
</script>
<!--The SALT Add-in to Internet Explorer object -->
<object id="SpeechTags" CLASSID="clsid:DCF68E5B-84A1-4047-98A4-0A72276D19CC" VIEWASTEXT WIDTH=0 HEIGHT=0></object>
<!--Importing the namespace from the implementation -->
<?import namespace="SALT" implementation="#SpeechTags" />
<body onload=RunAsk()>
<table id="table1"> <tr> <td>
  <input name="memid" type="text" title="title2">
</td>
</tr> </table>
</body>
</html>
<salt:prompt id="askmemid">title2</salt:prompt>
<salt:prompt id="sayDintUnderstand">Sorry, I didn't Understand</salt:prompt>
<salt:listen id="recomemid" onreco=" procmemid()" onnoreco="sayDintUnderstand.Start()">
    <salt:grammar id="gram" src="libary.grxml"></salt:grammar>
</salt:listen>
```

그림 8 HTML+SALT 구문 분석기로 생성한 음성 웹 페이지

생성하는지 실험하였다. 그 결과 그림 8에서 SALT와 자바스크립트 코드가 HTML웹 페이지에 정확히 삽입되어 있는 것을 확인하였다.

마지막으로 음성 웹 페이지 자동 생성기가 HTML로 구성된 웹 페이지에서 음성요소를 추출하여 SALT 코드를 정확히 생성하는지 평가하기 위하여 입력 박스, 선택 박스, 라디오 박스, 텍스트 박스로 구성된 샘플 웹 페이지를 작성하였다. 작성된 HTML 샘플 페이지는 음성 웹 페이지 자동 생성기에 입력하여 HTML+SALT 음성 웹 페이지를 생성하였다. 생성된 음성 웹 페이지는 Speech Add-in이 설치된 인터넷 익스플로러 브라우저에서 테스트한 결과 브라우저에서 음성으로 안내해 주고 사용자는 음성으로 정보를 입력하거나 선택할 수 있었다. 생성된 HTML+SALT 코드와 음성 웹 페이지의 실험 결과는 동영상으로 촬영하여 http://www.wins.or.kr/salt 홈페이지에 첨부한다.

또한 현재 호스팅 중인 국외사이트 50개의 회원가입 페이지를 대상으로 실험하였다. 50개 웹 사이트에서 음성 웹 페이지를 생성하였으며, 생성된 음성 웹 페이지는 웹 브라우저에서 정상적으로 동작하는 것을 확인하였다.

7. 결 론

음성 브라우저가 등장함에 따라, 음성 대화 어플리케이션이 웹 환경에서 사용이 가능하게 되었다. 기존의 웹 페이지들은 음성 대화를 고려하지 않고 시각용으로 제작되었지만, 이들 웹 페이지에도 음성 대화를 이용하여 처리할 수 있는 요소들이 있다. 따라서 본 논문에서는 시각용으로 제작된 HTML 웹 페이지로부터 대화처리가 가능한 요소들을 추출하고 해당 대화를 SALT로 생성해내는 음성 웹 페이지 자동 생성 시스템을 설계 및 구현하였다. 구현한 시스템은 HTML+SALT 어휘 분석기와 구문 분석기로 구성되어 있다. 어휘 분석기는 HTML 웹 페이지에서 음성 대화로 처리 가능한 링크 및 입력 태그 요소를 추출하여 토큰을 반환하고, 구문 분석기에서는 토큰을 입력 받아 SALT 음성 대화 스크립트를 생성하였다.

구현된 시스템의 성능을 확인하기 위해 HTML 웹 페이지를 자동 생성 시스템에 입력한 결과 정확히 변환되어 SALT 코드를 포함한 음성 웹 페이지가 생성되는 것을 확인할 수 있었다. 생성된 음성 웹 페이지는 인터넷 익스플로러 브라우저에서 테스트한 결과 브라우저에서 음성으로 안내해 주고 사용자는 음성으로 정보를 입력하거나 선택할 수 있었다.

본 연구에서 제안한 시스템은 기존의 시각용 웹 페이지를 음성 웹 페이지로 변환해주므로, 개발자는 기존에 개발된 시각용 웹 페이지를 본 연구의 자동 생성기로 변환하도록 환경을 설정하기만 하면 되며, 사용자는 회원 가입이나 게시판 등 입력 박스가 있는 웹 페이지에 키보드와 마우스를 이용하여 글을 입력할 수 있을 뿐만 아니라 음성 명령으로도 글자를 입력함으로써 보다 편리한 인터페이스를 이용할 수 있다.

향후에는 한글 구문 분석기와 어휘 분석기를 개발하

여 한글이 포함된 웹 페이지의 구조를 분석하여 음성 시나리오를 구성할 수 있는 방안에 대해서도 연구할 계획이다.

참 고 문 헌

[1] Mecanovic, D, and Shi, H. "Voice User Interface Design for a Telephone Application Using VoiceXML," Lecture Notes in Computer Science, No. 3399, Web Technologies Research and Development, pp.1058-1061.
[2] Susan J.Boyce, "Natural Spoken Dialogue Systems for Telephony Application," *Communications of the ACM*, vol.43, no.9, September, 2000.
[3] Speech Application Language Tags Specification, http://www.saltforum.org/saltforum/downloads/SALT1.0.pdf, 2002.
[4] "Voice Extensible Markup Language Version," http://www.w3c.org/TR/voicexml20, W3C Recommendation, 16 March 2004.
[5] Michael Dunn, "Pro Microsoft Speech Server 2007: Developing Speech Enabled Applications with .NET," Apress, June 2007.
[6] Hao Shi and Alexander Maier, Speech-enabled windows application using Microsoft SAPI, International Journal of Computer Science and Network Security, vol.6, no.9A, September 2006.
[7] Hiyan Shao, Robert Capra, Manuel A. Pérez-Quiñones, "Transcoding HTML to voiceXML using annotation," *IEEE Computer Society*, pp.249-258, 2003.11.
[8] Jee Young Oh, "A study on the Automatic Generation of Dialog Script Generator for Voice Browser," Hanbat National University, 2008.08.
[9] IISPLLab, "IISPLGenerator Specification," http://www.wins.or.kr/IISPLGenerator, 2008.

고 유 정
2002년 한밭대학교 컴퓨터공학과 학사
2004년 한밭대학교 컴퓨터공학과 석사
2009년 한밭대학교 컴퓨터공학과 박사
현재 한밭대학교 강사. 관심분야는 음성 브라우저, 웹 서비스, 컴파일러

김 윤 중
1981년 충남대학교 전자공학교육과 학사
1983년 충남대학교 전자공학과 석사. 1989년 충남대학교 전자공학과 박사. 현재 한밭대학교 컴퓨터공학과 교수. 관심분야는 음성인식, 컴파일러, 디지털 신호 처리, 웹기술