VXML 수행을 위한 ECMAScript 인터프리터의 설계 및 구현

신 동 혁† · 윤 영 선†† · 은 성 배†††

요 약

VXML에서는 시스템에 관련된 정보의 이용, 복잡한 수식의 해석, 반복적인 기능의 수행, 함수의 선언과 호출 등을 위하여 ECMAScript를 사용한다. 그러나 ECMAScript는 인터넷을 위한 표준 스크립트 언어이기 때문에 VXML과의 유기적 연동이 어렵다는 단점이 존재한다. 본 연구에서는 ECMAScript와 VXML의 유기적 연동을 위하여 ECMAScript의 요구사항을 만족시키는 인터프리터를 설계하고 구현하였다. VXML과의 연동을 위하여 VXML변수의 관리, 시스템 함수의 수행, 수식의 해석 및 함수의 호출 등의 인터페이스를 추가하여, VXML과 연동을 시도하였다. 연동 결과 VXML의 다양한 알고리즘을 처리할 수 있어 VXML의 응용 범위를 넓힐 수 있는 가능성을 얻었다.

Design and Implementation of ECMAScript Interpreter for VXML Execution

Dong-Hyeok Shin† · Young-Sun Yun†† · Sungbae Eun†††

ABSTRACT

ECMAScript can support VXML in utilizing the system information, analysis of complex equation, iterative execution, declaration of functions and their call, etc. However, since the ECMAScript is the standard script language for Internet, there is no way that the script lithely connects with VXML. In this paper, we presented the design and implemented the interpreter that meets the requirement of ECMAScript for its flexible connection with VXML. For connections, we added some functions in modified ECMAScript ; management of VXML variables, execution of system functions, analysis of equations and function calls. From the result of connection, it is shown that new ECMAScript can handle the various algorithms of VXML.

키워드 : VXML, ECMAScript, ECMAScript 인터프리터(ECMAScript Interpreter)

1. 서 론

인터넷을 이용한 정보의 교류가 활발해지면서 음성 입출력 기술을 이용한 보이스 포탈(voice portal)이 인터넷의 새로운 조류로 떠오르고 있다. 보이스 포탈이란 대표 전화번호로 전화를 걸면 인증 절차를 거쳐 전자우편, 일정, 메모 등 개인 정보와 인터넷 정보를 이용할 수 있는 서비스를 말하며, 메뉴를 이용한 방식으로 필요한 정보를 선택하여 음성이나 전화기의 톤으로 명령을 내리면 음성으로 정보를 들을 수 있다. 초기에는 제공되는 서비스의 양이 많지 않았기 때문에 원하는 정보를 쉽게 찾을 수 있었으나, 전달되는 정보량이 많아지면서 정보를 분류하거나 구분하는데 많은 어려움이 있다. 또한 음성으로 많은 양의 정보를 듣는 것은 한계가 있기 때문에 한정된 정보량 내에서 서비스를 빠르게 제공할 수 있는 컨텐츠 개발이 요구되고 있다.

보이스 포탈 서비스는 방대한 양의 컨텐츠와 특정 목적의 시나리오를 이용하여 시스템이 만들어지고 있기 때문에, 시나리오나 컨텐츠의 변경이 필요한 경우에는 전체 시스템을 재구성하여야 한다. 따라서 어떤 특정 서비스를 위하여 개발된 보이스 포탈 시스템은 다른 시스템에서 제공되는 서비스에 적용하기 힘들다는 문제점을 갖고 있다.

이러한 문제점을 해결하고 전화나 음성을 통한 인터넷 컨텐츠의 이용 및 제작이 용이하도록 고안된 언어가 VXML (Voice eXtensible Markup Language)이다[1]. VXML은 보이스 포탈 서비스의 시나리오를 작성 할 수 있도록 설계

† 정 회 원 : (주)코난테크놀로지
†† 종신회원 : 한남대학교 정보통신 · 멀티미디어공학부 교수
††† 정 회 원 : 한남대학교 정보통신 · 멀티미디어공학부 교수
논문접수 : 2002년 2월 22일, 심사완료 : 2003년 4월 11일

된 마크업(markup) 언어로서, 전화로부터의 DTMF(Dual Tone Multi-Frequency) 또는 음성 입력을 처리할 수 있다. 즉, HTML(Hyper Text Markup Language)이 시각적인 웹 페이지를 만드는데 사용되는 것이라면 VXML은 청각적인 웹 페이지를 만들 수 있도록 고안되었다. VXML은 1999년 8월 AT&T, IBM, Lucent 그리고 Motorola를 주축으로 구성된 VoiceXML 포럼에 의해 0.9 버전의 명세문이 발표되었고, 2000년 3월에 버전 1.0이 발표되었다. 현재는 보이스 포탈 서비스와 관련된 여러 회사에서 상용 서비스를 위한 제품 개발에 박차를 가하고 있으며, 개발 회사도 날로 증가하고 있다.

VXML은 프로그래밍 언어가 아닌 마크업 언어이기 때문에, 간단한 조건의 검사와 결과에 따른 분기 수행이 가능하나 클라이언트 시스템 정보의 이용 및 알고리즘 처리, 반복적인 연산이 어렵다. 즉, VXML은 정의된 내장함수가 없어 함수의 정의 및 호출이 어렵고 작성된 시나리오대로 동작하는 것이므로, 상황에 따른 시나리오의 변경과 적응이 어렵다. VXML에는 함수와 비슷한 개념을 갖는 'subdialog'라는 엘리먼트를 제공하여, 음성 대화의 한 단락으로 사용자 정의 함수에 가까운 개념을 갖고 있다. 그러나 사용 빈도수가 높은 함수들을 미리 정의한 내장함수를 지원하지 않기 때문에 강력한 기능의 연산을 수행하는데 많은 어려움이 있다. 이러한 단점을 해소하고자 VXML에 ECMAScript (European Computer Manufacturers Association Script)[2]를 연동하여 클라이언트 시스템의 정보를 이용하고, 알고리즘 및 반복적인 연산을 수행하는 애플리케이션을 작성할 수 있도록 하는 방식이 제안되었다. 이 방식은 VXML 수행 도중 실시간으로 클라이언트의 정보를 이용하여 VXML 시나리오를 구성할 수 있으며, 시나리오 작성 시 필요한 많은 내장함수를 지원함으로써 다양한 연산과 정보 처리를 지원한다. 또한 ECMAScript의 표현식은 VXML의 표현식과 동일하므로 표현식 해석에 따른 VXML의 부담을 상당히 줄일 수 있다.

본 논문에서는 웹 스크립팅[3]을 위해 설계된 ECMAScript 언어의 다양한 표현식과, 연산자, 내장함수들을 이용할 수 있는 인터프리터를 설계하고 구현 방법을 기술한다. 제안된 인터프리터는 ECMAScript 언어 명세를 만족하고, VXML과 ECMAScript의 유기적인 연동을 가능케하고 사용자 정의 변수를 서로 공유할 수 있다. VXML의 변수를 ECMAScript에서 이용하기위해서 모든 변수의 관리는 ECMAScript 인터프리터가 담당하고, VXML 변수에 대한 범주 조작도 가능하도록 한다. 또한 VXML 내부에서 사용되는 간단한 ECMAScript 표현식을 해석하기 위한 인터페이스를 제공하여 표현식 처리에 따른 VXML의 부담을 줄이도록 하였다. VXML의 요구사항을 만족시키기 위하여 여러 가지 API 함수를 두었으며 연동할 때 필요한 기능들을 추가적으로 설계하고 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 VXML 및 ECMAScript, VXML의 요구를 살펴보고, 3장에서는 설계 시 고려사항 및 자료구조를 살펴본다. 4장에서는 구현 환경 및 각 모듈을 설명하고, 마지막으로 5장에서 결론을 맺는다.

2. 배 경

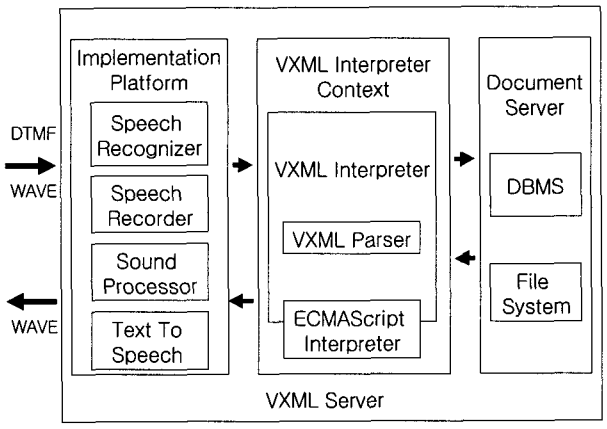
2.1 VXML

VXML은 AT&T, 루슨트 테크놀로지, 모토롤러 등 3사가 개발한 기술로, 전화와 음성처리 시스템을 통해 인터넷의 다양한 정보를 검색할 수 있도록 XML에 기반을 둔 마크업 언어이다. 음성 정보 기술이 지속적으로 발전함에 따라 보이스 포털을 비롯한 음성 응용 분야가 새로운 관심사로 떠오르고 있다. 차세대 꿈의 통신을 실현시킬 것이라 하여 현재 업계에서 관심이 집중되고 있는 IMT-2000 사업의 준비 과정에서도 음성 관련 기술은 상당히 중요한 역할을 차지하고 있다. 최근에는 VXML 문서 형식이 음성 합성과 음성 인식을 결합한 음성 서비스를 실현하기 위한 가장 유력한 방안으로써 기대를 모으고 있다. VXML 포럼은 1999년 8월 VXML 0.9 버전을 발표한 후 2000년 3월 이를 보완한 버전 1.0을 정식 제안하였다. 세계 인터넷 환경을 주도하고 있는 W3C 컨소시엄에서는 VXML 포럼의 제안을 받아들여 2000년 5월 VXML을 웹의 대화형 마크업 언어 표준으로 공인하였다.

일반적인 웹 정보는 시각적인 브라우징과 텍스트, 그래픽, 동영상 등의 내용을 전달하기 때문에, 인터넷을 통한 정보 전달은 시각적인 모니터 시스템을 갖는다. HTML[4]이 시각적 웹 문서를 만드는데 사용되는 것처럼 VXML은 청각적 웹 페이지를 위한 음성 대화를 정의하고 음성과 전화를 통한 인터넷 정보의 이용 및 컨텐츠를 저작하도록 고안된 언어이다. 또한 VXML은 전화로 인터넷의 전자우편, 날씨정보, 교통정보 등을 검색할 수 있고 웹 페이지를 통하여 다양한 음성정보를 전달할 수 있는 음성기반 기술이다.

보이스 포탈 서비스는 필요성이 증가하고, 꾸준한 연구와 개발이 이루어지고 있으나 지금까지의 시스템은 특정한 분야나 특정 서비스에 적합하도록 구현되어 있다. 컨텐츠는 독립적인 개발이 어려우며, 음성 인식 및 음성 합성 등과의 밀접한 관계가 필요하며, 현재까지의 시스템은 주로 C 언어를 이용하여 소스 코드 안에 컨텐츠를 삽입하여 개발되고 있다. 따라서 컨텐츠를 변경하기 위해서는 전체적인 시스템을 수정 또는 재구성해야 하는 어려움이 따르며, 다른 플랫폼으로의 이식이나 새로운 서비스의 추가는 많은 시간과 비용이 필요하다. 이런 문제점을 해결하기위하여 VXML

을 이용하여 특정 시스템에 종속하지 않는 보이스 포탈 시스템을 구축하여, 음성 처리와 컨텐츠 개발 그리고 플랫폼 개발이 독립적으로 이루어 질 수 있다. 또한 서비스 내용의 추가 또는 변경에 따른 컨텐츠의 독립적인 수정이 가능하여, 다른 플랫폼으로의 이식에 필요한 비용과 시간을 줄일 수 있다.



(그림 1) VXML-based voice portal service systems

VXML은 스크립트(script) 개발자가 여러 복잡한 문제로부터 벗어나 서비스 내용에만 집중할 수 있도록 한다. 지금까지의 음성 서비스 구현 과정과 달리 VXML은 음성 입출력의 기술적인 면에서 독립하여 서비스를 가능케 한다. 또한 스크립트를 통해 전달할 정보의 창출 과정이나 수집한 정보의 처리는 문서 서버(Document Server)에 연결된 CGI [5]나 DB 등을 통해 가능하므로 대화형 스크립트 개발자는 서비스 흐름 체계와 독립적으로 작업할 수 있다. (그림 1)처럼 플랫폼과 인터프리터 그리고 문서 서버가 독립적으로 되어있기 때문에 시나리오가 동작하는 플랫폼을 고려하지 않아도 된다. 시나리오를 쉽게 작성할 수 있고 여러 가지 상황을 표현하기에 쉬운 VXML의 문법 체계는 개발자가 다양하고 유연한 시나리오를 제공할 수 있도록 한다. 50여개에 달하는 요소는 다양한 상황의 음성 대화를 표현할 수 있도록 하며, ECMAScript를 이용함으로써 고급적인 표현을 가능하게 해주고 있다. 또한 프로그래머를 괴롭히는 프로그램에서의 자원 할당과 반납 문제도 인터프리터가 담당하기 때문에 프로그래머는 손쉽게 프로그램을 작성할 수 있다.

지금까지 살펴본 VXML의 장점은 컨텐츠 구축에 소요되는 인력을 최소화하여 음성 포탈의 구축을 가능하게 할 수 있다는 점이다. 또한 널리 설치되어 사용되고 있는 웹 서버를 문서 서버로 이용하기 때문에 음성 서비스 스크립트 작성의 확산 및 대중화가 가능해 개인 음성 서비스를 쉽게 제공할 수 있다. VXML 서버에서 말하는 문서 서버는 웹 서버와 성격이 거의 비슷하다. 현재의 웹 서버는 시각적인 HTML 문서를 제공하며 VXML 서버의 문서 서버는 청각적인 정보에 바탕을 둔 VXML 문서를 제공한다. 따라서 VXML이 상용화되면 음성을 활용한 웹 기반의 콘텐츠 서비스와 음성인식 애플리케이션간의 자료변환이나 상호 정보 교환이 중요하게 된다.

2.2 ECMAScript

초기의 HTML은 텍스트로 이루어진 정보 전달이 주목적이었으며, 점차 이미지나 사진과 같은 화상 정보도 지원하게 되었다. 그러나 이들 문서는 정적인 정보를 전달하기 때문에 시간에 따라 변하는 동적인 정보나 다양한 표현이 미흡하였다. 이에 따라 브라우저 공급업체들은 마크업 언어(Markup language)를 확장하여 테이블이나 스타일 시트 같은 다양한 표현과 동적인 정보를 표현할 수 있는 발전된 기능을 쓸 수 있도록 하였다. ECMAScript는 유럽 쪽의 표준화를 담당하고 있는 ECMA에서 채택한 WWW[6]의 표준 스크립트로써, 널리 알려진 JavaScript[7]나 Jscript와 부합된 특성을 갖고 있으며, 최초로 웹 스크립팅을 위해 설계된 언어이다. ECMAScript는 동적인 문서를 기술할 수 있으며 프로그래밍 언어와 마찬가지로 알고리즘을 수행할 수 있는 언어로 개발되었다.

객체 지향 프로그래밍 언어인 ECMAScript 언어는 다른 프로그래밍 언어들과 마찬가지로 객체 및 함수, 연산자들을 제공한다. 객체 및 함수들은 자신의 범주(scope)를 갖고 있어서 범주 밖의 객체나 변수들은 이용할 수 없다. 변수가 가질 수 있는 값의 형태로는 Undefined, Null, Boolean, Number, String, Object 등이 있으며, 객체의 메쏘드(method)들은 속성(property)으로 정의되어 있다. ECMAScript에서 제공되는 객체로는 Global, Object, Function, Array, String, Boolean, Number, Math, Date, RegExp, Error 객체들이 있다. 또한 ECMAScript는 단항 연산자, 다항 연산자, 비트 연산자, 할당 연산자, 생성 연산자 등 많은 연산자를 제공한다.

ECMAScript 프로그램은 Unicode[8]로 표현되므로, Lexical 분석기는 16비트 코드와 8비트 코드를 모두 처리할 수 있도록 명시되어 있다. ECMAScript의 문법중 많은 부분이 C 언어와 비슷하며, 코멘트와 명령문(statement), 연산자 등은 C 언어와 동일하다. ECMAScript의 명령문들은 반드시 세미콜론(semicolon)으로 끝나야 하며 대부분의 스크립트언어처럼 ECMAScript도 세미콜론 자동 삽입을 지원하고 있다.

2.3 VXML에서 ECMAScript의 해석

VXML과 ECMAScript는 <표 1>에서 정리한 것과 같이

그 특성이 서로 다르다. 따라서 ECMAScript는 VXML과 같은 마크업 언어의 일부분으로서 존재해야 하며, VXML은 ECMAScript의 존재 여부에 상관없이 동작할 수 있어야 한다. VXML 언어 명세에서 ECMAScript를 하나의 구성요소로 사용하는 것은 VXML이 할 수 없는 일을 ECMAScript를 통해서 보완 하고자 하는 것이다. <표 2>의 예를 보면 VXML은 현재 시간에 대한 시스템의 정보를 이용할 수 없고, 오전 및 오후를 구별하는 알고리즘도 처리하지 못한다. 이러한 VXML의 단점을 보완하기 위하여 ECMAScript를 하나의 요소로 삽입하였다.

〈표 1〉 VXML과 ECMAScript

VXML	ECMAScript
마크업 언어	스크립트 언어
정보의 표현 및 이용 기술	알고리즘 처리
사용자 입력 처리	시스템 정보의 이용
음성 대화를 정의	프로그래밍 연산

ECMAScript 언어 표준에는 VXML과의 연동부분이 정의 되지 않았으므로, ECMAScript를 VXML에서 사용하기 위해서는 VXML을 위한 인터페이스가 필요하다. 즉, 두 언어가 서로 독립적으로 수행되나 그 관계는 완전히 독립적이지 않다. 변수 선언에서 살펴보면 VXML과 ECMAScript의 변수들은 독립적으로 동작하지 않는다. VXML 내부에서는 변수를 갖지 않으며, 모든 변수는 ECMAScript 내부에서 관리된다. ECMAScript 언어 표준에서는 각 변수들이 독립적인 영역을 갖기 때문에, 상호간의 공유가 어려워 정보의 전달 및 이용이 어렵다. 그러므로 VXML에서는 모든 표현식을 ECMAScript에서 해결해 줄 것을 요청하고 있으며 변수의 생성, 변경 및 삭제 연산에 대해 ECMAScript에 작업을 요청함으로써 해결하고 있다.

ECMAScript는 함수를 선언, 정의하고 호출할 수 있지만 VXML에서 호출할 수 있는 인터페이스를 지원하지 않고 있다. 또한 ECMAScript에는 시스템 정보를 이용하고, 수학적 계산을 도와주는 다양한 내장함수를 가지고 있으나, VXML에서 내장함수를 이용할 수 있는 인터페이스의 설계에 대한 내용을 ECMAScript 언어 표준에서는 언급하지 않는다. 따라서 ECMAScript에서 VXML과 연동하여 ECMAScript 함수에서 VXML를 호출하는 것이 어렵다. 이런 문제점을 해결하기 위하여 VXML 안에 ECMAScript를 포함시켜 VXML에서 ECMAScript 함수를 호출하도록 하고 있다. <표 3>의 예에서 보면 ECMAScript내에 "factorial"이라는 함수를 정의하고 VXML에서 factorial 함수를 호출하고 있다. VXML은 함수를 호출하는 부분인 "factorial(fact)"을 factorial(3)을 호출한 결과 값인 "6"이라는 값으로 대치한다.

〈표 3〉 사용자 정의 함수를 정의하고 호출하는 예

```
                    :
                    :
< script > < ![CDATA [
        function factorial (n)
        { return ( n <= 1 ) ? 1 : n * factorial ( n-1 ) ; }
```

〈표 2〉 VXML과 ECMAScript의 예

VXML

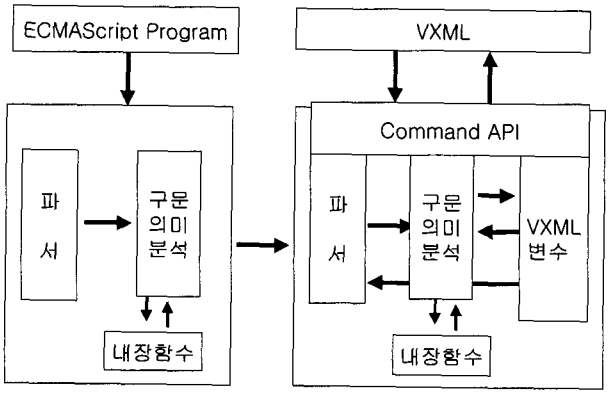
```
< ?xml version = "1.0"? >
< !DOCTYPE vxml SYSTEM "vxml.dtd" >
< vxml version = "1.0" >
  < form >
    < var name = "h" expr = "1"/ >
    < var name = "m" expr = "34"/ >
    < var name = "s" expr = "19"/ >
    < field name = "r" type = "boolean" >
      < prompt > VXML만으로는 현재시간을 알 수 없습니다.
                 미리 입력된 시간은
        < value expr = "h"/ > 시,
        < value expr = "m"/ > 분,
        < value expr = "s"/ > 초 입니다.
      < /prompt >
      < prompt >
        다시 들으시겠습니까?
      < /prompt >
      < filled >
        < if cond = "hear_another" >
          < clear/ >
        < /if >
      < /filled >
    < /field >
  < /form >
< /vxml >
```

ECMAScript

```
function currentTime( ) {
  var d = new Date( ) ;
  var h = d.getHours( ) ;
  var m = d.getMinutes( ) ;
  var s = d.getSeconds( ) ;
 if (h >= 12) {
    h = h - 12 ;
    if (h == 12)
      return
        '오전,' + h + '시,' + m + '분,' + s + '초' ;
    else
      return
        '오후,' + h + '시,' + m + '분,' + s + '초' ;
  } else
   return
     '오전,' + h + '시,' + m + '분,' + s + '초' ;
}
```

```
        ]]>
</script>
        <var name = "face" expr = "3"/>
                        :
                        :
        팩토리알은 <value expr = "factorial(fact)"/> 입니다.
                        :
                        :
```

다른 문제점으로는 ECMAScript 언어 표준에서는 단순 변수만이 존재하기 때문에 한 변수 내에 여러 개의 값을 저장할 수 없다는 것이다. 그러나 VXML에 있는 SUBDIA LOG 요소는 한 변수 내에 여러 값을 저장할 수 있도록 정의되고 있다. VXML에서 필요로 하는 변수를 생성, 수정하고 삭제할 수 있는 인터페이스를 ECMAScript가 지원해야하나, ECMAScript 언어 표준에는 이부분에 대한 설계가 언급되어 있지 않으므로 다중 변수의 생성 및 이용이 어렵다. 이 문제점을 해결하기 위하여 본 연구에서는 ECMAScript와 VXML의 변수를 서로 공유하기 위하여 기존의 ECMAScript 표준 언어의 구조를 (그림 2)에서와 같이 변경하여 사용하고자 한다.



(그림 2) ECMAScript 표준 언어의 구조 및 변경된 구조

3. 설 계

3.1 설계시 고려사항

VXML 수행을 위한 ECMAScript 인터프리터를 설계하기 위해서는 순수한 ECMAScript 언어 명세 외에 다음 사항들을 고려하여야 한다.

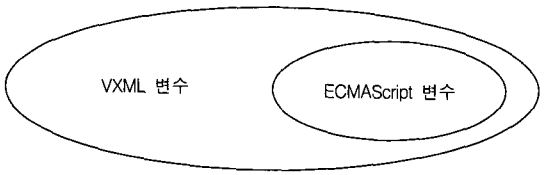
① 하나의 변수 이름에 여러 값을 갖는 새로운 변수를 추가한다.
② VXML의 요청 또는 VXML의 생존 주기에 따라 내부 데이터들을 유지한다.
③ VXML의 엔티티(entity)들을 고려하여 엔티티와 동일한 연산자 및 키워드들을 변경한다.
④ 스트링을 입력으로 받는 Evaluate 및 변수에 대한 상태를 알려주는 인터페이스를 추가한다.
⑤ VXML 및 ECMAScript의 변수를 동일하게 관리한다.
⑥ ECMAScript에는 없으나 VXML에 존재하는 "undefined" 리터럴 (literal)을 수용한다.

VXML은 <SCRIPT>와 </SCRIPT> 사이의 텍스트들을 ECMAScript로 간주한다. 이 태그(tag)가 나타나면 VXML은 ECMAScript의 해석을 요청한다. 이 경우, 해석은 계산된 결과 값이 아닌 문장을 분석한 자료를 갖게 된다. ECMAScript의 생존 기간은 VXML과 같고, 여러 번 이용되기 때문에 ECMAScript 인터프리터는 ECMAScript를 해석할 수 있는 정보를 보존하여야 한다.

〈표 4〉 Assign 요소의 DTD 와 DI

DTD	< !ELEMENT assign EMPTY > < !ATTLIST assign *name* %field.name ; #REQUIRED *expr* %expression ; #REQUIRED >
DI 1	< assign name = "card" expr = "1"/ >
DI 2	< assign name = "card" expr = "dealCard( )"/ >

VXML에서 여러 가지 표현식들이 등장하는데 이것은 VXML의 변수, ECMAScript의 변수 또는 함수이기 때문에 ECMAScript 인터프리터에게 해석을 요청하고 결과를 전달받는다. <표 4>에서 보는바와 같이 VXML의 Element인 <ASSIGN>태그는 'expr'에 표현식이 오도록 되어 있다. 표현식에 '1' 이라는 값이 올 때와 'dealCard()'라는 함수 이름이 올 때 VXML에서는 같은 표현식 텍스트로 인식한다. 따라서 ECMAScript의 문법과 같은 표현식을 VXML에서 해석할 때 별도의 표현식 처리가 필요하다는 부담이 발생된다. 그러나 이 부분의 해석을 ECMAScript 인터프리터에게 요청할 경우 VXML은 표현식 처리 부분을 줄일 수 있으므로 간소화될 수 있다.



(그림 3) VXML 변수와 ECMAScript 변수의 포함 관계

VXML에서 ECMAScript를 구현하기 위하여, 한·변수를 두 곳에 독립적으로 정의하는 것이 아니라, 한 곳에서 정의되고 다른 쪽에서 이용할 수 있도록 공유하여야 한다. (그림 3)과 같이 ECMAScript의 변수들은 VXML의 변수의 범주에 속해 있지만 VXML의 변수들은 ECMAScript의 변수 범주에 속해있지 않다. 따라서 모든 변수의 관리는 ECMA

Script에서 한다. 또한 VXML의 변수를 관리하면 그 변수가 속한 범주도 같이 관리해야 하므로, ECMAScript의 범주가 아닌 VXML의 범주를 조정할 수 있는 환경을 제공한다. 즉, VXML 변수를 위한 별도의 Scope를 제공한다.

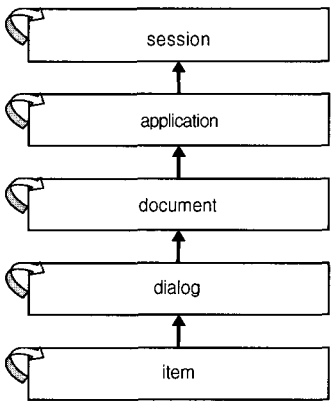
〈표 5〉 변수의 선언 및 정의

선언 후 정의	< var name = "variable"/ > < assign name = "variable" expr = "'value1'"/ >
선언 없이 정의	< assign name = "variable" expr = "'value1'"/ >

구현시 Scope 관리 및 공유와 함께 생성 및 삭제 기능도 제공한다. VXML에서는 표현식과 개체의 구분 없이 하나의 텍스트 스트림으로 보기 때문에, 생성 기능의 제공시 개체를 받아들여 생성하는 방법과 표현식을 해석하여 생성하는 방법을 동시에 제공한다. 일반적인 프로그래밍 언어에서는 함수의 선언 후, 함수의 정의 부분이 뒤따르나 VXML 및 ECMAScript에서는 선언 없이 바로 정의될 수 있다.

ECMAScript 언어에는 여러 가지 데이터 타입이 있지만 사용자의 입장에서는 타입의 구별을 고려하지 않는다. 하나의 연산을 처리할 때마다 연산 결과 값이 생성되고, 이 값을 이용하여 다른 문장을 수행한다. 따라서 같은 내용의 연산을 수행해도 다른 타입의 결과 값이 생성될 수 있다. 그러므로 각 개체들은 여러 가지 데이터 타입을 수용해야 하며, 연산 결과에 따라 적절한 데이터 타입을 결정한다.

'<SCRIPT>'와 '</SCRIPT>' 사이에 오는 텍스트 스트림은 ECMAScript 인터프리터에게 해석되는 1차적 해석 과정을 거친다. 즉, 파싱 및 글로벌 개체의 연산을 수행한다. 이 연산은 VXML Element들의 ECMAScript 표현식의 형태를 갖는 Attribute에서 실제로 사용된다. 또한, ECMAScript의 생존 주기는 VXML과 동일하기 때문에 해석 후 바로 삭제하지 않고, VXML의 수행에 의하여 ECMAScript의 초기화가 요청되기 전까지는 Parse Tree 및 모든 데이터들을 보존한다.



(그림 4) VXML Scope hierarchy

ECMAScript에서 사용되는 데이터 및 함수들은 범주(scope)를 갖는데, 연산 수행에 필요한 데이터 및 함수들을 찾을 때 자신이 속해있는 범위에서부터 전역 범주 순으로 찾아간다. 따라서 ECMAScript 내의 범주 정보와는 별도로 VXML의 데이터를 위한 범주가 지원되어야 한다. VXML 및 ECMAScript의 범주 관계에서 우선순위가 높은 순으로 나열하면 session, application, document, dialog, item, global 그리고 local의 순서가 된다. session에서 item까지는 VXML 변수 범위이고 global과 local은 ECMAScript의 변수 범주이다. 이러한 범주 정보와 함께 Active Execute Context도 함수의 호출 및 종료할 때마다 재조정되어야 한다.

VXML의 subdialog 요소를 위해서는 다중 값(multiple value)을 갖는 타입이 지원되어야 한다. subdialog 요소는 다중 값뿐 아니라 단일 데이터도 처리해야하기 때문에, 단일 데이터일 때는 값을 치환하고, 다중 값일 때는 값을 추가한다. 일반적으로 변수는 하나의 값을 갖도록 되어 있지만 subdialog 요소에 의해 반환되는 변수는 내부에 여러 값을 갖고 있다고 해석된다. 이 요소의 "name" 속성(attribute)도 하나의 변수이므로, "name" 값으로 변수를 찾아 이용하게 된다. 예를 들어 "subdialog" 요소의 이름은 "sub 1"이고 반환되는 namelist는 "var 1"과 "var 2"가 있다고 하자. VXML은 변수 중에서 "sub 1"이라는 이름을 갖는 변수를 찾고, 이것이 단순 변수인지 여러 값을 갖는 변수인지를 조사하게 되고, 다중 값을 갖는 변수일 때에는 "sub 1"의 값을 사용하지 않고 "var 1"과 "var 2"의 값을 사용하게 된다. "var 1"과 "var 2"를 표현하는 표현식은 dot(.)연산으로 나타내며, "var 1"또는 "var 2"역시 다중 값이 될 수 있다.

3.2 자료 구조

ECMAScript의 해석 과정을 살펴보면, 먼저 Parse Tree를 생성하고 생성된 Tree를 검색하여 해석한다. 본 연구에서 구현한 인터프리터는 계층적 구조를 갖는 클래스들을 사용하였기 때문에, 관련된 클래스는 다음과 같다. 파서의 가장 중요한 클래스로는 비종단 노드(non-terminal node)를 표현한 Symbol 클래스와 종단 노드를 표현한 Token 클래스가 있으며, 이 두 클래스의 기본 클래스인 Node 클래스가 있다. Node는 다음 Sibling을 가리키는 포인터를 가지고 있으며, Symbol에는 Token을, Token에는 실제 데이터를 가리키는 포인터가 있다. 실제 데이터를 만들어 주는 것은 Lexical 분석기이다. Lexical 분석기는 16비트가 한 character가 되는 배열을 생성하여 Terminal Token을 저장한다.

ECMAScript의 여러 가지 데이터 타입들을 표현하기 위하여 각 타입마다 하나의 클래스를 생성하였고, Semantic 처리 루틴의 반환 값을 하나로 통일하기 위하여 하나의 기본 클래스로부터 파생되도록 하였다. 또한 Object 타입은

여러 개의 속성 집합을 리스트 형태로 가질 수 있도록 설계하였다. 이 타입으로부터 생성된 Object 클래스는 VXML의 변수 및 범주 관리에 사용된다. 즉 하나의 객체는 일정한 범주를 형성하고, 그 형성된 범주 내부에 다른 객체들이 존재할 수 있다. 변수는 객체 내부에 속성 형태로 저장되게 된다. 각각의 객체마다 이것이 새로 생성된 객체인지 아니면 다른 객체 내에 있는 속성인지를 표시하여 자원 반환 여부를 결정짓도록 하였다.

내장 함수들 역시 하나의 속성의 형태로 해당 객체에 속하게 된다. 모든 내장 함수들은 해당 객체의 프로토타입 객체에 속하게 된다. 프로토타입에는 내장 함수의 기능을 수행하는 실제적인 함수가 있고, 각 함수에 아이디를 부여한다. 사용자는 내장 함수를 이름으로 호출하게 되는데, 내장 함수를 찾는 알고리즘은 변수를 찾는 알고리즘과 동일하다. 해당 객체에서 찾으려는 함수 이름이 발견되지 않으면 프로토타입 객체에서 찾게 된다. 특정 타입에 대한 인스턴스 객체는 여러 개가 있을 수 있지만 프로토타입 객체는 한 개이기 때문에 프로토타입에 대한 인스턴스를 만들 필요는 없다.

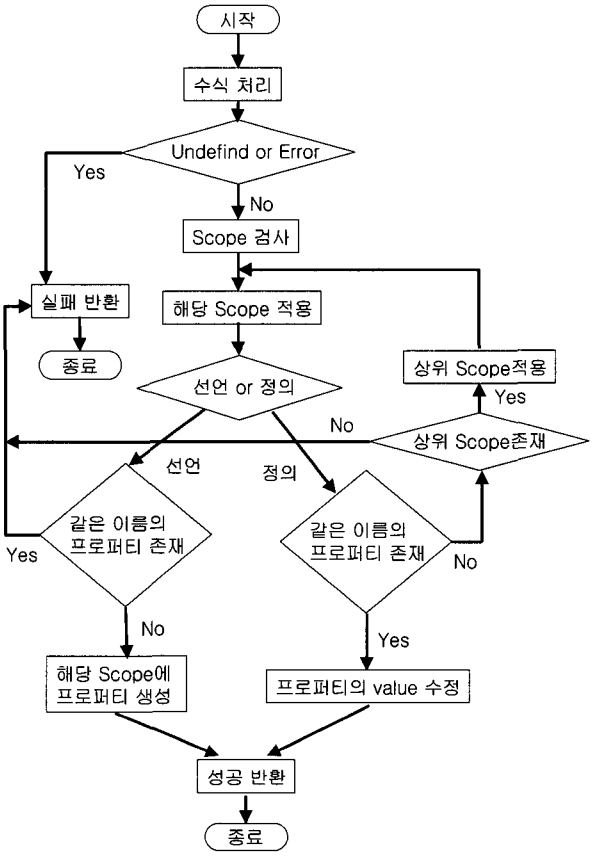
8비트의 char형 대신 16비트로된 특별한 char형을 사용하여 unicode를 지원할 수 있도록 설계하였다. 즉, 입력 흐름이 8비트일지라도 16비트로 변환하여 저장하고 처리함으로써, 모든 스트링 연산은 16비트 연산으로 이루어진다.

객체는 내부에 정렬되지 않은 속성들을 가질 수 있다. 속성에 있는 값의 형태로는 ECMAScript에서 사용되는 모든 값이 지정될 수 있다. 범위 중에서 히스토리 기억이 필요한 document, dialog, item은 별도의 히스토리 스택을 갖고 있다. 이 히스토리 스택에는 현재의 모든 변수들을 저장할 수 있으며 변수 이름 찾기 규칙에 방해되어서는 안 된다. 또한 객체의 변수 및 내장 함수들뿐만 아니라 범위 정보까지 같은 구조를 갖게 함으로써 일괄 알고리즘(batch algorithm)을 적용할 수 있도록 하였다.

4. 구 현

본 연구에서 구현한 ECMAScript 인터프리터는 IBM-PC 호환 기종에서 Windows 운용체제에서 표준 C++ 언어로 프로그램되었고, Visual C++ 컴파일러를 이용하였다. KDE libraries 2.0의 kjs를 수정 및 보완하여 구현하였으며, 70여 개의 클래스로 구성되었다. 구현된 인터프리터는 독립 애플리케이션이 아닌 VXML 인터프리터의 한 모듈로써 작동하기때문에, 인터프리터가 해석한 결과는 VXML의 수행에 있어서 중요한 역할을 담당한다. ECMAScript 인터프리터는 파서 모듈, 구문-의미(syntax-semantic) 분석 모듈, VXML 변수 관리 모듈 및 내장 함수 모듈로 나뉜다.

구현된 인터프리터와 VXML과의 연동을 위하여 입력 및 출력 함수를 추가하였다. 입력으로는 파일 입력과 텍스트 스트림 입력을 허용하며, 모든 출력은 처리 결과 상태를 알려주는 상태 플래그와 결과-값이 되는 스트링으로 통일하였다. 오류가 발생할 경우 오류 원인을 결과-값으로 반환한다. 내부에 오류가 발생하여 프로그램이 종료되거나 잘못된 연산을 수행해서는 안되기때문에, 어떠한 경우에도 프로그램이 비정상적으로 종료하지 않도록 구현하였다. 리소스 관리를 위하여 KSO라는 클래스를 삭제하였고, Node로부터 파생된 클래스들의 evaluate 멤버 함수는 base 클래스의 포인터를 반환하도록 수정하였다.



(그림 5) VXML 변수의 선언 및 정의 순서도

ECMAScript 내에 VXML의 변수를 생성시키고 변경 및 삭제할 수 있는 다양한 인터페이스를 구현하였다. 선언과 정의를 할 수 있도록 CREATE, MULTIPLE_CREATE, CREATE_OR_STORE, STORE, STORE_ANYWAY 등 5가지 모드를 지원하도록 하였다. 특히 STORE_ANYWAY 모드는 어떠한 경우에도 변수가 정의되도록 하였다. 예를 들어 "dialog.field1.field2.field3.name"과 같은 표현식에서 field1, field2, field3이 없는 상태에서 name을 저장하려 할 때에도 저장이 가능하여야 한다. 이 경우 "dialog.field1"을 처리하

고 같은 방식으로 다음 필드도 처리하도록 구현하였다. 이러한 기능은 VXML의 subdialog 요소를 수행할 때 반드시 필요한 기능이다. 또한 KScript 클래스를 수정하여 다양한 인터페이스를 구현하였다. 또한 MULTIPLE_CREATE 옵션을 사용하여 여러 변수를 생성할 수 있는데, 이 모드는 VXML을 위하여 별도로 추가된 모드이며, KSOMultiple이라는 클래스를 추가하여 구현하였다.

이외에도 변수의 선언과 정의 상태를 알 수 있는 인터페이스를 구현하였고, 변수의 잘못된 사용을 막기 위해 단순 변수와 복합 변수를 구별할 수 있는 인터페이스를 구현하였다. 이 알고리즘도 변수의 선언 및 정의 알고리즘과 비슷하며, 차이점은 프로퍼티의 생성 및 수정부분이 없다는 것이다.

ECMAScript의 "부울 변수 + 스트링" 연산은 숫자의 더하기 연산이 아닌 문자열 병합 연산이 적용되는 것과 같이 서로 다른 타입의 변수간에도 다양한 연산이 가능하다. 따라서 VXML 변수를 ECMAScript 내에 생성할 때 객체가 아닌 표현식을 받아들이도록 구현하였다.

VXML 변수는 ECMAScript에서 사용할 수 있기 때문에 ECMAScript 인터프리터에서 관리하도록 구현하였으며, VXMLScope라는 클래스를 추가하여 VXML의 변수를 관리하도록 하였다. VXML 변수에 ECMAScript에 있는 이름 찾기 규칙을 똑같이 적용하기 위해 VXML 변수만을 위한 별도의 범주를 구현하였다. VXML 컨텍스트(context)가 변경되면 이전에 사용하던 변수를 삭제하기도 하지만 과거의 변수를 다시 사용하는 경우도 있다. VXML의 요청에 따라 과거에 사용하던 변수를 히스토리에 저장하였다가 컨텍스트가 복구되면 과거의 변수를 사용할 수 있도록 구현하였다. ECMAScript 인터프리터에 VXML 변수 범위를 저장할 수 있는 스택을 두어 히스토리를 저장하도록 하였으며, 히스토리 저장은 item, dialog, document 범위에 대해서만 이루어진다.

사용자 정의 함수와 내장 함수의 사용은 ECMAScript 프로그램을 통하여 이루어진다. VXML에서 ECMAScript의 함수를 편리하게 이용할 수 있도록 표현식을 통한 함수 이용 부분을 구현하였다. 이 인터페이스는 간단한 표현식 처리를 위하여 사용되기도 하고 ECMAScript 내의 객체를 이용할 수도 있다. VXML로부터 입력되어지는 텍스트 스트림에 대해서는 파싱과 구문-의미 분석을 수행하는 방법으로 구현하였다.

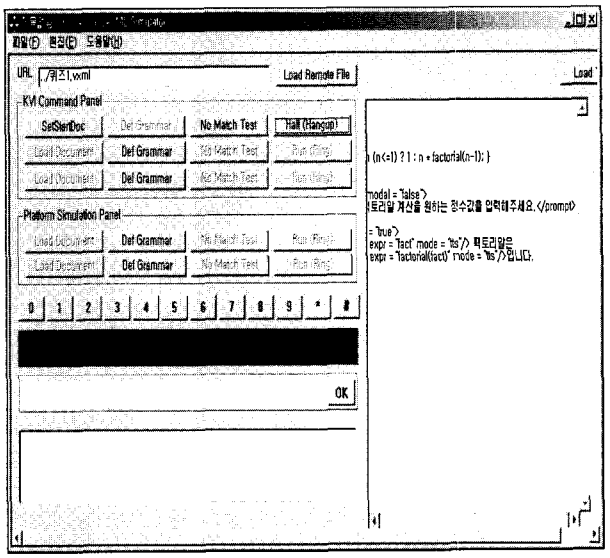
〈표 6〉 ECMAScript를 포함하고 있는 VXML 문서 예제

```
<?xml version = "1.0"?>
<!DOCTYPE vxml SYSTEM "vxml.dtd">
<vxml version = "1.0">
    <script><![CDATA[
    function factorial (n)
        { return (n <= 1) ? 1 : n * factorial(n-1); }
    ]]></script>
    <form id = "form">
        <field name = "fact" type = "number">
        <prompt>
        팩토리알 계산을 원하는 정수값을 입력해주세요.
        </prompt>
        <filled>
            <prompt>
            <value expr = "fact"/> 팩토리알은
            <value expr = "factorial(fact)"/>입니다.
            </prompt>
        </filled>
        </field>
    </form>
</vxml>
```

(그림 6)과 (그림 7)은 ECMAScript가 VXML에 의해 수행되는 예를 보여주고 있다. <표 6>은 수행되는 VXML 문서이다. <표 7>은 <표 6>에 있는 VXML이 수행 되었을 때의 컴퓨터와 사용자의 대화를 보인다.

〈표 7〉 VXML 분석 예제

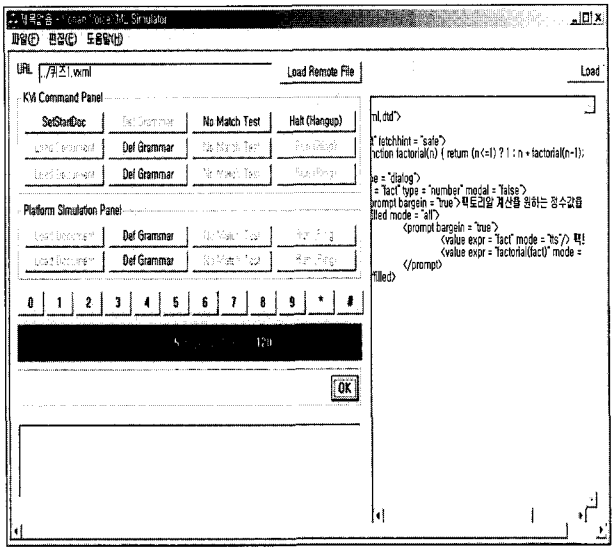
```
컴퓨터 : 팩토리알 계산을 원하는 정수 값을 입력해주세요.
사용자 : 삼
컴퓨터 : 삼 팩토리알은 6입니다.
```



(그림 6) ECMAScript 내의 factorial 함수 수행 화면

(그림 6)은 VXML 문서를 로드하고 수행하는 화면이다. VXML 스크립트를 수행하기 전에 ECMAScript 인터프리터에게 ECMAScript 프로그램을 처리하도록 요청하며, ECMAScript 인터프리터는 VXML로부터 요청 받은 프로그램을 분석(parse)하고 그 결과인 분석 트리(parsing tree)를 가

진다. 이 트리에서 VXML의 "field" 요소에 있는 "name" 속성 값은 변수로 사용된다. VXML은 "fact"를 변수로서 선언하도록 ECMAScript 인터프리터에게 요청하며, 범위는 "dialog"가 된다. 선언과 동시에 fact 변수의 값은 "undefined"이다. field 요소의 "name" 속성 값이 "undefined"가 된다는 것은 해당 필드를 아직 수행하지 않았다는 것을 의미한다. 따라서 VXML은 값이 "undefined"인 필드를 찾아 수행하게 되는 것이다. 여기에서 변수 "fact"를 먼저 해석하여 그 값을 "factorial" 함수의 파라미터로 넘겨주는 것은 아니다. VXML은 모든 표현식을 처리하지 않고 하나의 텍스트 스트림으로 간주한다. ECMAScript에서의 "factorial(fact)" 해석은 파라미터인 "fact"의 값을 찾기 위해, 자신의 위치와 가장 가까운 범주부터 검색한다. 순수한 ECMAScript의 가장 상위 레벨인 "GLOBAL"에도 없으면 VXML 범주를 검색한다. 변수 "fact"는 VXML 범위 중에서 "dialog"에 존재하고 있다. 다음으로 함수 "factorial"을 찾아 "fact" 값을 전달하여 함수를 수행시킨다. (그림 7)은 "factorial(fact)" 처리를 ECMAScript 인터프리터에게 처리하도록 요청한 후 결과 값을 받아 출력하고 있다.



(그림 7) ECMAScript 수행 후 결과 값 출력 화면

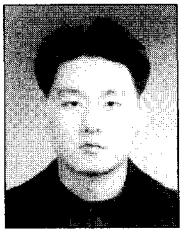
본 연구에서 구현한 VXML 시뮬레이터는 VXML 인터프리터의 모듈로서 작동하도록 하였고 IBM PC호환 기종의 윈도즈 운영체제 환경에서 시험하였다. 다양한 스크립트 프로그램을 만들어 시험한 결과, 모든 Built-in 함수들이 제대로 작동되었고, 타입에러 및 참조(reference) 에러가 발생할 때 에러 처리를 하여 어떠한 경우에도 인터프리터가 비정상 종료되지 않음을 확인할 수 있었다. VXML의 요청에 의한 범주별 변수 관리는 정상적으로 이루어 졌으며, Name Lookup Rule도 ECMAScript와 동일하게 적용됨을 확인하였다. 또한, 모든 처리의 수행 전과 수행 후의 시스템 리소스를 검사한 결과 리소스의 할당 및 반환이 정상적으로 이루어졌다.

5. 결 론

본 연구에서는 VXML 수행을 위한 ECMAScript 인터프리터를 MS 윈도우 환경에서 표준 C++를 이용하여 설계하고 구현하였다. 구현된 시스템에서는 VXML을 위하여 다양한 인터페이스를 제공하고 VXML에서 요구하는 기능들을 추가 구현하였다. 또한 구현된 ECMAScript 인터프리터를 이용함으로써 다양하고 고급의 VXML 문서를 작성할 수 있다. 이 인터프리터는 VXML 뿐만 아니라 스크립트 언어의 지원을 필요로 하는 모든 문서 형식이나 도구 제작에 도움을 줄 것으로 기대 된다.

참 고 문 헌

[1] VoiceXML Forum, VoiceXML 1.0 Specification Document.
[2] Standard ECMAScript-262, "ECMAScript Language Specification," iUniverse, 3rd Edition, 1998.
[3] David Barron, "The World of Scripting Language," John Wiley & Sons, 2000.
[4] Klein, Jeannine M. E., "Building Enhanced Html Help With Dhtml and Css," Prentice Hall Computer Books, 2000.
[5] Stein, L., "Web CGI Scripting with perl," Longman.
[6] Sears, Andrew, "International Journal of Human-Computer Interaction-Computer Interaction (WWW Usability)," Lawrence Erlbaum Associates, 2000.
[7] Heinle, Nick, "Designing with JavaScript : Creating Dynamic Web Pages," O'Reily, 1997.
[8] Unicode Consortium, "The Unicode Standard, Version 3.0," Addison-Wesley Longman Pub., 2000.
[9] Alfred V. Aho, Jeffrey D. Ullman, "Principles of Compiler Design," Addison-Wesley Publishing, 1977.
[10] Ellis Howitz, Sartaj Sahni, Dinesh Mehta, "Fundamentals of Data Structures in C++," W. H. Freeman and Company, 1995.
[11] Andrew W. Appel, "Modern compiler implementation in C," Cambridge University Press, 1997.



신 동 혁
e-mail : hyeok@konantech.com
1999년 한남대학교 정보통신공학과(공학사)
2001년 한남대학교 정보통신공학과(공학석사)
2000년~현재 (주)코난 테크놀로지 연구원
관심분야 : 자연언어처리, 인터넷, Java 등

윤 영 선

e-mail : ysyun@mail.hannam.ac.kr

1990년 한국과학기술원 전산학과(공학사)

1992년 한국과학기술원 전산학과(공학석사)

1992년~1995년 (주)핸디소프트 주임연구원

2001년 한국과학기술원 전산학과(공학박사)

2001년~현재 한남대학교 정보통신·멀티미디어공학부 조교수

관심분야 : 음성인식, 패턴인식, 음성 정보 검색 등

은 성 배

e-mail : sbeun@octacomm.net

1985년 서울대학교 컴퓨터공학과(공학사)

1987년 한국과학기술원 전산학과(공학석사)

1987년~1990년 한국전자통신연구원 연구원

1995년 한국과학기술원 전산학과(공학박사)

1995년~현재 한남대학교 정보통신·멀티미디어공학부 부교수

2000년~현재 (주)옥타컴 대표이사

2001년~현재 무선 인터넷 연구회 총무

관심분야 : 임베디드 시스템, 홈 네트워킹, 망관리 시스템 등