

ARS 제어를 위한 동작 및 데이터 모델

민경석, 김석일, 전주남
충북대학교 전자계산학과
e-mail:ksmin@john.chungbuk.ac.kr

An Operational and Data Model for ARS Control

Kyoungseok Min, Sukil Kim, Joongnam Jeon

Department of Computer Science, Chungbuk National University

요 약

ARS(Audio Response System)를 구현하기 위하여, 응용 분야를 분석하여 필요한 자료구조를 설계 및 처리 과정을 설계한 후, ARS 처리용 하드웨어 생산 업체에서 제공하는 원시 라이브러리와 C언어를 사용하여 구현하는 것이 일반적이다.

본 논문에서는 ARS의 처리 과정을 분석하여 동작을 제어하는 부분과 동작을 표현하는 부분으로 분리한 ARS 구현 모델을 제시하였다. 응용분야와 무관한 동작제어 부분은 대기상태, 처리상태, 종료상태로 구성되는 유한상태 기계 모델을 제시하였고, 응용분야에 따라 결정되는 동작표현에 필요한 정보를 체계적으로 구성한 자료구조를 제시하였다. 본 논문에서 제시하는 모델에 의하여 동작표현만 제공함으로써 ARS를 구현할 수 있다.

keyword : CTI, ARS, 유한 상태 기계

1. 서 론

CTI(Computer Telephony Integration)는 그 동안 각 발전해 온 전화 시스템과 컴퓨터 시스템을 서로 통합하는 일종의 응용 기술로, 컴퓨터 분야의 음성, 팩스, 정보처리 등의 통신 처리기능과 교환 분야의 호 제어(call management) 기능을 결합하여 다양한 서비스를 제공하는 것이다[1]. CTI의 관련 기술에는 음성처리(voice processing), 음성인식, 문자/음성 변환(text to speech), 팩스 기술 등이 포함된다[2]. ARS(Audio Response System)는 CTI 기술 중 음성처리에 속하는 것으로 전화기의 버튼을 누를 때 발생하는 DTMF[3] 신호와 음성 안내를 이용하여 컴퓨터가 발신자에게 자동 응답하는 시스템이다.

CTI 기술의 구성 요소는 하드웨어와 소프트웨어로 구분할 수 있다. 컴퓨터 기술의 발전에 따라 이 두 부분 역시 빠르게 변하고 있지만 국내의 개발 환경은 이러한 변화에 적응하지 못하고 있다. 하드웨어 기술은 이미 미국의 Dialogic, GammaLink 등의 회사에서 전세계 시장을 석권하고 있으며, 국내에서는 이 회사의 제품을 수입하여 사용하고 있다. 하드웨어 제조업체에서는 하드웨어

를 제어하기 위한 소프트웨어 라이브러리를 함께 제공하고 있으며, 이것을 사용하여 응용 분야별로 CTI 소프트웨어를 작성하게 된다[7].

ARS 소프트웨어를 구현하기 위해서는 하드웨어에 관한 지식, 제조업체에서 제공하는 라이브러리의 기능, Window SDK, MFC, 그리고 응용 분야에서의 데이터 처리 방식 등에 대하여 알아야 하기 때문에 생산성 있는 프로그램을 작성하기 어렵다.

본 논문에서는 ARS 동작을 유한상태 기계(finite-state machine)를 사용하여 분석함으로써, 응용 분야와는 무관하게 실행되는 유한상태 기계 엔진과 응용 분야에 따라 달라지는 동작 표현을 위한 자료 구조로 ARS 모델을 설계하였다. 이 모델을 사용하면, ARS 개발과정이 하드웨어에 의존하는 부분은 실행 엔진으로 캡슐화되기 때문에, 응용분야에 대한 데이터 처리 방식을 유한상태 기계 상태로 표현해 줌으로써 간단히 ARS 시스템을 구현할 수 있다.

2. ARS의 기본 동작 분석

ARS는 전화를 받고 사용자가 입력한 자료를 분석하

여 적절한 안내 음성을 제공하고, 필요시 호스트 컴퓨터로부터 자료를 받아 사용자에게 제공하는 전화기와 안내원의 역할을 담당한다. 사용자 입장에서 본 ARS 서비스의 동작은 다음(그림 1)과 같다.

- ① ARS 시스템은 사용자에게 필요한 음성 안내를 내보낸다.
- ② 사용자는 음성 안내에 따라 전화기의 버튼으로 필요한 입력을 한다.
- ③ 이 입력에 따라 ARS 시스템은 사용자가 요구하는 정보를 조합하여 음성으로 내보낸다.
- ④ 다음 서비스에 필요한 입력을 사용자로부터 받는다.
- ⑤ 위의 과정이 반복되다 정상적으로 종료되면 마지막 음성을 내보내고 서비스가 끝나게 된다.

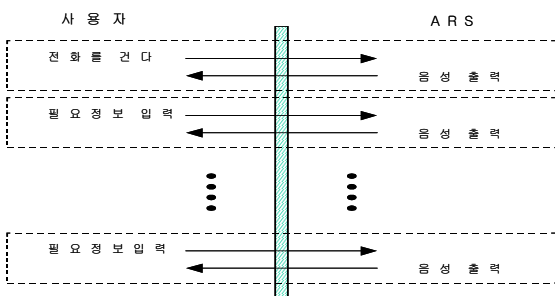


그림 1. ARS 사용자 측면의 일반적 동작

위 동작 중 전화 사용자와 컴퓨터와의 상호동작을 제어하기 위해서는 전화와 컴퓨터를 연결해 주는 하드웨어를 다루는 동작이 필요하다[4][5]. 표 1과 같이 하드웨어는 사용자와 소프트웨어의 중계 역할을 담당한다. 즉, 사용자의 입력을 소프트웨어에게 전달하고, 소프트웨어가 사용자에게 제공하는 음성을 출력하는 데 필요한 제어를 담당한다.

여기서 주목할 점은 표1의 순서 1에서 5의 과정과 n+8에서 n+9의 과정은 서비스의 시작과 끝에 한번씩 실행되고, 순서 n에서 n+7의 과정은 서비스가 진행되는 동안 반복한다는 것이다.

표 2. ARS 내부동작 순서

순서	사용자	하드웨어	소프트웨어
1			전화를 기다린다
2	전화를 건다		
3		전화가 온 것 감지	
4			전화를 받는다
5		전화 연결 감지	
n			음성을 출력한다
n+1	버튼을 누른다		
n+2		버튼 입력 감지	
n+3			입력을 받기 위한 설정
n+4	입력을 마친다		
n+5		입력 종료 감지	
n+6			입력 값을 확인
n+7			입력 값에 따른 서비스 선택
n+8			음성출력
n+9			서비스종료

3. 유한 상태 기계를 이용한 ARS 동작 모델

ARS 동작은 사용자에게 보여지는 외부 동작과 외부 동작을 수행하기 위하여 하드웨어를 제어하는 내부 동작으로 분리하여 생각할 수 있으며, 이들을 유한상태 기계로 표현할 수 있다.

3.1 외부동작

ARS가 사용자에게 보여지는 동작은 다음과 같이 세 가지 형태로 나눌 수 있으며, 유한 상태 기계로 나타내면 그림 2와 같다.

- ① 대기상태(순서 1~5): 사용자가 전화 연결을 시작한다.
- ② 처리상태(순서 n~n+7): 사용자에게 서비스를 제공한다. 이 상태는 반복하여 수행된다.
- ③ 종료상태(순서 n+8~n+9): 사용자가 서비스를 종료한다. 이 후, 사용자를 기다리는 대기상태로 변경된다.

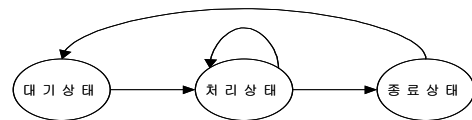


그림 2. ARS 동작의 세가지 상태

3.2 내부동작

ARS의 외부동작은 다시 하드웨어[4],[5],[9]를 다루는 내부적인 동작을 포함하는 상태로 세분화된다. 각 외부 상태를 처리하기 위한 내부동작은 다음과 같다.

3.2.1 대기상태

외부 동작의 대기상태는 하드웨어로부터 전화가 왔다는 신호를 받고 전화연결을 실시하는 단계이다. 대기상태는 그림 3과 같이 기다림상태, 연결상태의 두 가지 상태로 기계로 표현할 수 있다.

- ① 기다림상태(state10): 사용자로부터 전화가 오기를 기다려 전화받기 명령(Set OffHook)을 하드웨어에게 전달한다.
- ② 연결상태(state11): 전화받기 동작 완료를 확인하고, 처리상태로 이동한다.

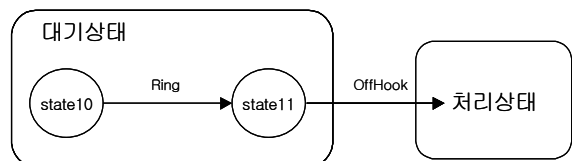


그림 3. 대기상태의 내부동작

3.2.2 처리상태

처리상태는 ARS 동작의 중심이 되는 부분이며, 사용자와의 직접적인 동작을 수행하는 상태로 그림 4와 같이 출력상태, 출력종료 상태, 입력준비 상태, 확인상태, 저장상태, 결정상태의 여섯 가지 상태로 구분된다.

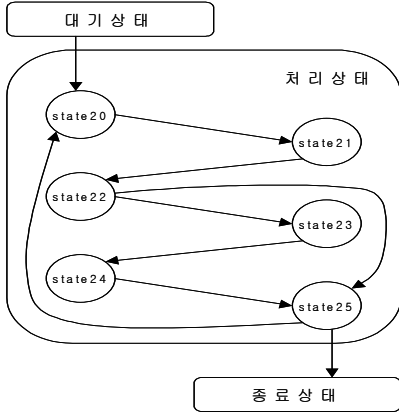


그림 4 처리상태의 내부동작

- ① 출력상태(state20): 음성 출력을 지시한다. 하드웨어는 음성파일 목록을 받아 음성출력을 시작하게 된다.
- ② 출력종료상태(state21): 음성이 출력 중임을 확인하고, 다음상태로 이동한다.
- ③ 입력준비상태(state22): 사용자가 입력할 데이터의 형태를 설정하고, 확인상태로 이동한다. 만약 입력받을 필요가 없으면 결정상태로 이동한다. 이 때에도 음성은 출력 중이다.
- ④ 확인상태(state23): 음성출력이 완료된 후, 또는 음성 출력 중이라도, 사용자가 입력준비상태에서 설정한 데이터를 모두 입력하기를 기다려, 저장상태로 이동한다.
- ⑤ 저장상태(state24): 사용자의 입력 값을 지정된 변수에 저장하고, 결정상태로 이동한다.
- ⑥ 결정상태(state25): ARS 제어 흐름에서 더 입력받을 내용이 있으면 출력상태로 이동하고, 제어 흐름의 마지막이면, 종료상태로 이동한다.

3.2.3 종료상태

한 통화의 서비스를 종료하는 단계이다. 그림 5와 같이 전화를 끊는 과정(끊기상태)과, 다음 통화를 위한 준비 과정(초기화상태)으로 구성된다.

- ① 끊기상태(state30): 전화끊기 명령(Set OnHook) 명령을 하드웨어에게 전달한다.
- ② 초기화상태(state31): 전화가 끊어진 것을 확인하고, 다음 전화를 받기 위하여 관련된 제어 변수들을 초기화한 후, 대기상태로 이동한다.



그림 5. 종료상태의 내부동작

3.3 상태도의 통합

앞에서 살펴본 것처럼 ARS의 외부동작은 대기상태, 처리상태, 종료상태로 표현할 수 있으며, 각 외부동작은 하드웨어와 연관된 세부적인 내부동작으로 표현된다. 그러나 세부동작 설명에는 예외 발생의 경우가 제외되어 있다. 즉, 사용자가 도중에 전화를 끊거나, 사용자가 데이터를 입력하지 않고 장시간 전화를 들고 있는 경우 등의 예외가 발생했을 때 필요한 처리조건이 제외되었다. 그림 6은 예외 처리를 포함한 통합 ARS 처리과정이다.

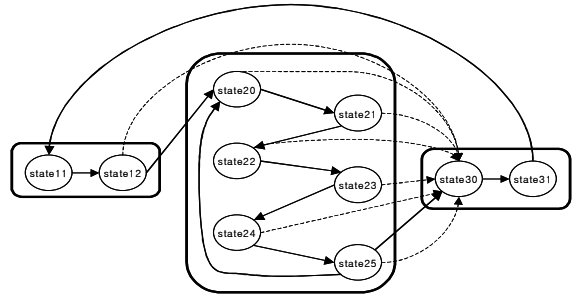


그림 6. 예외 처리를 포함한 ARS 처리과정

그림 6에서 점선으로 나타난 것이 예외 상황이 발생하는 경우이다. 예를 들면, state30으로의 점선입력은 현상태와 무관하게 사용자가 전화를 끊는 경우에 해당된다.

4. 처리상태의 표현

3절에서 ARS의 동작을 분석하고 계층적 유한 상태 기계틀을 이용하여 일반적인 모델을 제시하였다. 이 모델은 ARS 동작을 제어하기 위한 엔진 부분에 해당되며, ARS 응용 분야 별로 결정되는 구체적인 서비스에 관한 내용을 포함되지 않는다. 어떤 음성이 출력되고, 어떤 입력을 받아서 어떠한 서비스를 할 것인가 등에 관한 정보는 포함되어 있지 않다. 외부동작 중에 ARS 서비스를 처리하는 과정은 처리상태이고, 대기상태와 종료상태는 ARS 서비스와 관계가 없다. 그러므로, 처리상태를 표현하기 위한 자료구조가 필요하다.

사용자 입장에서 본 ARS 처리과정의 예는 다음과 같다.

- ① “000 ARS서비스입니다. 00정보를 원하시면 0번을, 00정보를 원하시면 0번을 눌러주세요!”와 같은 안내

문을 들은 후, 번호를 입력한다.

- ② “번호가 잘못되었습니다. 다시 입력하세요!”, 혹은 “비밀번호가 필요합니다. 비밀번호 0자리를 눌러주세요!” 등의 안내문이 나오고, 번호를 다시 누르거나 비밀번호를 입력한다.
- ③ “문의하신 내용은 0000입니다. 다른 정보를 원하시면 0번을 종료하시려면 0번을 누르세요!”와 같은 안내문이 나온다.

위 예의 각 항목이 하나의 처리상태에 해당되며, 실제 ARS의 동작은 처리상태의 연속이다. 반복되는 처리상태는 내부적으로 같은 일을 처리하지만 외부적으로는 서로 다른 파라미터를 가지고 동작하는 것을 알 수 있다. 따라서, 처리상태가 필요로 하는 정보를 표현하기 위한 자료구조를 구체적으로 분석할 필요가 있다.

처리상태의 각 내부상태가 필요로 하는 정보는 다음과 같다.

- ① 출력상태: 출력할 음성에 관한 정보가 필요하다. 음성이 여러 개의 파일로 나누어지는 경우도 있으므로, 음성파일 목록이 필요하다.
- ② 출력종료상태: 출력이 종료되면 파일을 닫아야 하므로 출력상태에서 사용한 파일의 이름이 필요하다.
- ③ 입력준비 상태: 사용자가 입력할 데이터의 형태에 관한 정보가 필요하다. 입력할 버튼의 숫자가 제한되는 경우와 자리수가 제한되지 않고 종료기호(* 또는 # 버튼)를 사용하는 경우가 있다.
- ④ 확인상태: 사용자의 입력의 완료여부를 확인하는 상태로 별도의 정보가 필요 없다.
- ⑤ 저장상태: 사용자의 입력 값을 저장할 변수가 필요하다. 이 변수에 저장된 값은 처리상태를 제어하는 데 사용되며, 입력 값 중 일부는 ARS 서비스가 종료된 후에 데이터베이스에 저장되기도 한다.
- ⑥ 결정상태: 결정상태는 다음의 처리상태 또는 종료상태로의 이동을 결정하기 위한 정보가 필요하다.

외부상태인 처리상태에 이름을 부여하고, 위에서 열거한 내부동작이 필요로 하는 정보를 표현하기 위한 자료구조를 표2에 정리하였다.

표 3. 처리상태 정보에 대한 자료구조

상태	필요한 자료구조
처리상태	처리상태이름
출력상태	음성파일 목록
입력준비상태	입력 버튼 수 종료기호 사용여부 및 종료기호
저장상태	저장될 변수의 이름, 변수의 값을 저장하기 위한 변수
결정상태	입력 값의 형태 다음처리상태 이름

이 논문에서는 각 처리상태가 필요로 하는 자료들로 구성되는 클래스 ArsNode를 정의하였다.

```
class CArsNode
{
public:
    CString m_NodeName; // 현재 상태의 이름
    CString m_FileNameList; // 출력할 음성파일명
    int m_Length; // 사용자입력을 위한
    int m_TermFlag; // 형태
    CVariable m_Variable; // 입력 값을 저장하기 위한 변수명
    CPtrArray m_NextNode; // 입력형태에 따른 다음 상태정보

public:
    CArsNode();
    virtual ~CArsNode();
};
```

그림 7 처리상태 수행 시 필요한 정보클래스

- ▶ m_NodeName: 처리상태이름으로 처리상태를 구분하기 위한 실별자(ID, identifier)로 사용된다.
- ▶ m_FileNameList: 처리상태에서 출력할 음성파일 목록이다. 출력할 음성은 일반적인 안내문, 사용자의 숫자 입력을 음성으로 재확인, 그리고 숫자로 구성된 코드를 디코드한 음성을 조합하여 생성한다. 즉, 정보에 해당하는 단어 단위의 음성을 조합하여 하나의 문장을 만들어 주어야 한다. 따라서 음성안내에 필요한 정보들과 음성파일의 이름에 관한 목록이 필요하다.
- ▶ m_Length, m_TermFlag: m_Length는 사용자로부터 입력 받을 버튼의 최대수를 저장한다. m_TermFlag는 종료조건을 저장하며, 값이 0일 때는 종료조건이 없고 고정된 수만큼 입력받는 경우이고, 값이 1일 때는 종료조건으로 “*”를 사용하며, 그리고 값이 2일 때는 종료조건으로 “#”을 사용함을 나타낸다.
- ▶ m_Variable: (m_Name, m_Value)의 쌍으로 구성된 클래스의 형태를 갖는 자료이다. m_Name은 현재 상태에서 입력된 정보를 저장할 변수의 이름이고, m_Value는 입력된 정보를 저장할 변수이다.
- ▶ m_NextNode: (m_Input, m_NodeName)의 쌍으로 구성된 클래스의 형태를 갖는 배열이다. m_Input은 입력된 값의 범위이고, m_NodeName은 다음 처리상태의 이름이다. 다음 처리상태 결정 방법은 세 가지이다. 첫째는 입력된 값이 직접 사용되는 경우, 두 번째는 입력된 값으로 데이터베이스를 참조하는 경우, 그리고 잘못된 입력에 의하여 이전 처리상태를 다시 반복하는 경우이다. 첫 번째는 입력 값의 범위를 m_Input에 저장하고, 두 번째는 m_Input에 “match”를 저장하며, 세 번째는 m_Input에 “no”를 저장하여 각 구분한다.

ArsNode는 ARS의 실제동작을 유한상태 기계로 표현할 때 하나의 처리상태에 해당된다. 따라서 전체적인 ARS의 동작을 나타내기 위해서는 각 처리상태마다

ArsNode 정보를 표현해 주어야 한다.

그림 8과 같이 ARS의 실제동작을 처리상태 목록으로 만들고 ARS 동작 엔진과 처리상태 표현을 분리함으로써 처리상태의 표현만 변경함으로써 다른 형태의 ARS를 구현할 수 있다. 또한 ARS의 내부동작을 이해하지 못하더라도 처리상태만 표현하면 쉽게 ARS의 구현이 가능하다.

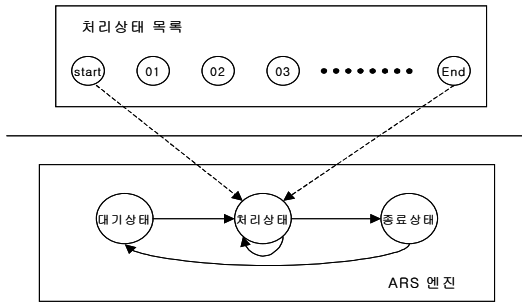


그림 8 ARS의 계층적 실행도

5. 구현

이 논문의 구현을 위한 시스템 환경은 전화와 컴퓨터를 연결하기 위한 하드웨어로 Dialogic사의 D/41D[4,5]를 사용하였다.

본 논문에서는 구현할 시스템을 객체지향 모델링 기호[6]를 사용하여 그림 9와 같이 표현하였다. VoicBoard는 D/41D의 동작을 제어하는 클래스로 4개의 Channel 클래스를 가진다. Channel 클래스는 ARS를 직접적으로 서비스하는 클래스로 실제 ARS 처리 엔진이 구현되어 있고, 처리상태가 필요로 하는 정보를 저장할 자료구조를 가지고 있다. 내부상태들이 멤버함수의 형태로 구현이 되어 있으며, 하드웨어에서 들어오는 신호에 따라 상태가 전이된다.

Channel 클래스는 한 번의 통화를 제어하기 위하여 필요한 자료 목록을 갖는다. 이 자료 목록은 ID, 패스워드, 기타 ARS 응용분야 별로 저장하여야 할 자료 등을 저장하기 위하여 사용된다. 그리고, 처리상태를 표현하는 ArsNode 클래스의 배열을 멤버변수로 갖는다.

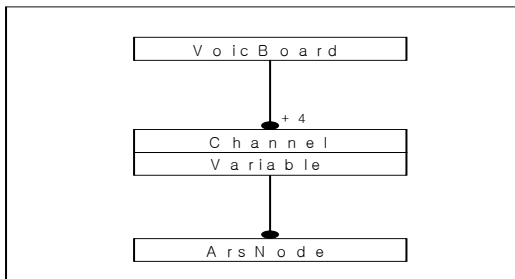


그림 9 ARS구현을 위한 클래스 다이어그램

그림 10은 처리상태 목록의 일부를 표현한 예이다. "NodeName"에서 "NodeList"까지가 하나의 상태를 나타내며 ArsNode 클래스의 멤버변수에 대응된다. 이 처리상태 목록들은 ARS 프로그램이 초기화 될 때 파일의 형태로 주어지며, Channel 클래스의 ArsNode 배열에 저장된다.

```

=====//start
NodeName=Greeting                               : Nodename
VoiceFile='D:\WWWvoice\WWgreeting.vox'         : Voicefile name
VariableName= :
Digits=1                                         : # of digits with terminator
NodeLists=(1,CarInPut)(2,CarSerch)(3,LuggageInput)
           (4,LuggageSerch)(0,EEND)             : list of next nodes
=====//공차입력확인
NodeName=CarReSult                               : Nodename
VoiceFile='D:\WWWvoice\WW지역.vox',(ECDeparture,지역)
           'D:\WWWvoice\WWend.vox'             : Voicefile names
VariableName= :
Digits=1                                         :
NodeLists=(*,CarReSult)(0,CarChange)(#,CarSearch)(No,CarReset) list of next nodes
=====//공차입력완료
NodeName=End                                     : Nodename
VoiceFile='D:\WWWvoice\WW13입력완료.vox'       : Voicefile name
VariableName= :
Digits=0                                         : # of digits with terminator
NodeLists= : list of next nodes
    
```

그림 10 ARS 외부동작 모듈의 표현

그림 10이 나타내는 의미는 다음과 같다.

- ▶ NodeName=CarReset: 처리상태의 이름이 CarReset임을 의미한다.
- ▶ VoiceFile='D:\WWWvoice\WW지역.vox',(BCDeparture,지역),'D:\WWWvoice\Wend.vox': 지역.vox의 음성파, 사용자의 입력이 저장된 BCDeparture의 값과 '지역'이라는 정보를 가지고 데이터베이스에서 출력할 음성을 찾고, end.vox를 결합하여 하나의 안내음성을 내보내라는 의미이다.
- ▶ VariableName= : CarReset 상태에서는 입력받은 값을 저장하지 않는다는 의미이다.
- ▶ Digits=1: 사용자로부터 한번의 버튼입력을 받는다는 의미이다.
- ▶ NodeList=(*,CarResult)(0,CarChange)(#,CarSearch): 사용자의 입력값이 '*'이면 CarResult의 상태로, '0'이면 CarChange의 상태로, '#'이면 CarSearch의 상태로 처리상태가 변경된다. 그리고 사용자의 입력이 '*',0,#'이 외의 값이면 CarResult의 상태, 즉 자신의 처리상태를 반복한다는 의미이다.

6. 결론

본 논문에서는 ARS의 동작을 분석하여 대기상태, 처리상태, 종료상태의 세 가지 상태에 의해 수행되는 외부동작과 실질적인 ARS의 동작을 표현하는 내부동작으로 분리하여 ARS를 구현하는 동작 및 데이터 모델을 제시하였다. 동작 모델은 응용 분야와 무관한 실행 엔진에 해당되고, 데이터 모델은 처리상태에서 필요로 하는 정보를 체계적으로 정리한 것이다.

본 논문에서 제안하는 모델을 사용하면, ARS 내부동작을 이해하지 못하더라도 외부동작에 대한 정보의 표현만으로 ARS를 구현할 수 있다. 따라서 기존에 하드웨어가 제공하는 원시 함수를 이용하여 어렵게 프로그램 하던 것을 실행동작의 표현만으로 쉽게 구현할 수 있는 방법을 제시하였으며, 구현에 있어 하드웨어의 동작을 제어하는 내부 동작을 객체 지향 기법에 의하여 캡슐화함으로써, 재사용성을 높이고 하드웨어 인터페이스를 편리하게 하였다.

앞으로 ARS의 외부동작의 표현을 파일로 직접 만들어주던 것을 필요한 정보를 입력하면 자동으로 설계할 수 있도록 하는 것을 개발할 것이며, 정보처리 부분의 자료를 데이터베이스화하는 것과 음성/문자 변환의 기법을 사용하는 부분에 대한 연구가 필요하다.

6. 참고문헌

- [1] 배장만, 이승범 “CTI 구성 요소 및 통신망 고려사항” 전자공학회지 vol. 24 no. 7 . 1997
- [2] 김희동 “컴퓨터텔레포니 결합 기술” 전자공학회지 vol. 24 no. 7. 1997
- [3] K.Edwards, K.Quine, P.B.Dalziel and M.A.Jack "Evaluating Commercial Speech Recognition and DTMF Technology For Automated Telephone Banking Services" IEE 1997
- [4] Voice Features Guide for Windows NT Volume 1 Manual Dialogic
- [5] Voice Programmer's Guide for Windows NT Volume 2 Manual Dialogic
- [6] “Object-Oriented Modeling and Design James” Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorenson, PRENTICE HALL
- [7] <http://itelab.inchon.ac.kr/nainsys/resourse/html/cti1.html>
- [8] <http://www.speed.co.kr/ars1.html>
- [9] <http://www.dialogic.com/>