홍 기 형 교수지도
석사학위 청구논문

한국어를 기반으로한 음성 사용자 인터페이스 설계방법론

2006

성신여자대학교 교육대학원
전자계산 교육학과
권 지 혜

한국어를 기반으로한 음성 사용자 인터페이스 설계방법론

홍 기 형 교수지도

이 논문을 석사학위논문으로 제출함

2006년 5월

성신여자대학교 교육대학원
전자계산 교육학과
권 지 혜

논문 개요

자동차용 네비게이션(navigation) 서비스, 일부 관공서의 시각 장애우를 위한 음성 브라우저 서비스, 철도청의 ARS를 이용한 철도 예매 서비스 등과 같이 점차 음성 사용자 인터페이스의 수요가 늘고 있다. 이는 '음성'이라는 인간의 기본적인 의사소통 도구를 이용한다는 점에서 사용자가 보다 편리하게 이용할 수 있기 때문이다.

본 논문에서는 음성 사용자 인터페이스 설계 방법론에 대하여 알아보고, 우리말의 특성을 반영한 프롬프트 및 인식 문법의 설계 방법을 제시하였다. 또한 음성 사용자 인터페이스의 기본 요소인 대화(다이얼로그)를 폼 필링 시스템 주도(Form-Filling Directed) 대화, 폼 필링 상호 주도(Form-Filling Mixed Initiative)대화, 메뉴 선택(Menu Select) 대화의 유형으로 나누어 정형화 하여 모델링 하였다.

목 차

그림 목차

표 목차

I. 서 론

공상과학 영화에서나 보아왔던 음성 기술이 유비쿼터스(Ubiquitous) 기술의 비약적인 발전으로 이제 텔레메틱스(Telematics), 게임, 로봇공학, 생활가전 등의 영역에서 우리가 직접 체험할 수 있는 시대가 도래 하였다. 음성 기술은 기계와 인간사이의 상호소통을 인간의 가장 기본적이며 자연스러운 의사소통의 수단인 음성을 이용한다는 점에서 사용자에 큰 매력을 준다. 음성 기술의 핵심은 음성 사용자 인터페이스의 설계이다. 일반적으로 지금까지 시스템의 사용자 인터페이스는 그래픽 사용자 인터페이스 (GUI, Graphical User Interface)에 많은 초점이 맞추어져 왔고, 다양한 응용에서의 다양한 GUI의 설계 방법론에 대해서도 많은 연구가 있어 왔다[1]. 그에 반하여 음성사용자 인터페이스 (VUI, Voice User Interface)에 관한 연구는 매우 드물게 진행되어 왔으며, 최근의 음성 기술의 발전과 음성이 전화망 기반 시스템의 인터페이스에서 다양한 휴대기기 및 무선인터넷 등 그 응용 범위가 확대됨에 따라 음성 사용자 인터페이스 설계에 관한 연구가 본격적으로 시작되고 있다 [2, 3, 4].

현재까지 연구된 음성 사용자 인터페이스 설계 방법론은 영어권 언어를 중심으로 한 방법론이 대부분이다. 기본적인 VUI 설계 방법론은 영어권의 VUI 설계 방법론을 사용해도 큰 무리는 없으나, 프롬프트 설계단계, 억양계획, 인식문법의 설계 단계에서는 한국어 및 우리 문화가 가진 고유한 특징을 설계 단계에 반영해야 한다.

본 논문에서는 음성 사용자 인터페이스 설계 방법론을 알아보고, 프롬프트 설계 단계, 억양계획 인식 문법의 설계 단계에서 한국어의 특징을 고려한 설계 기법을 제시한다. 또한 음성 인터페이스의 기본 단위인 대화(다이얼로그)를 폼 필링 시스템 주도(Form-Filling Directed) 대화, 폼 필링 상호 주도(Form-Filling Mixed Initiative) 대화, 메뉴 선택(Menu-Selection) 대화의 유형

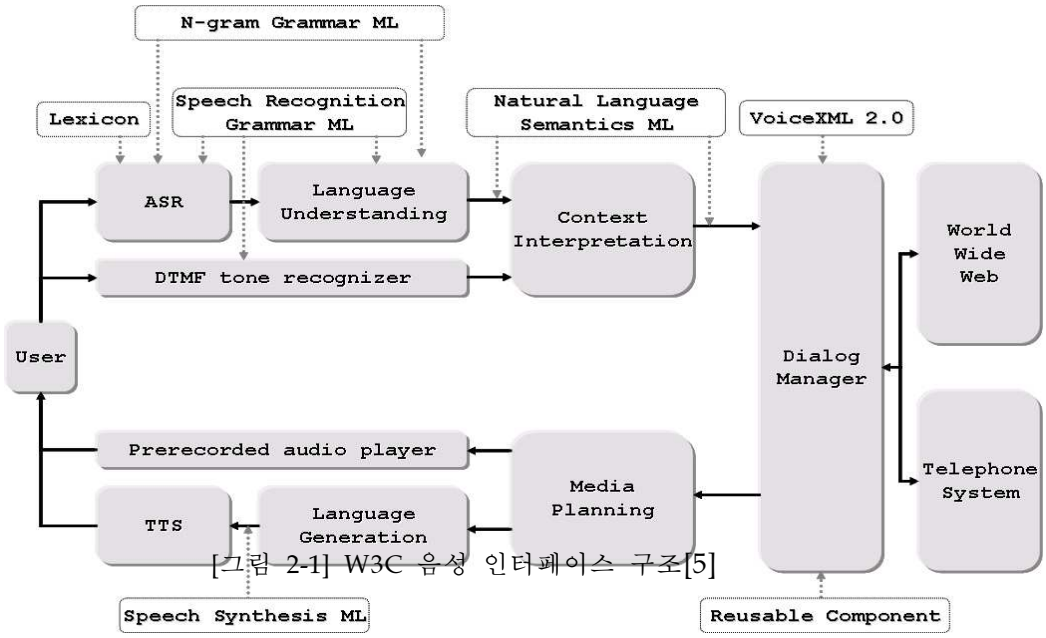
으로 분류하고 정형화 하여 모델링한다.

본 논문의 구성은 다음과 같다. 2장에서는 현재의 음성 인터페이스의 구조와 음성 사용자 인터페이스의 특징 대해 개괄적으로 기술한다. 3장에서는 음성 사용자 인터페이스의 설계 단계별 방법론에 대하여 설명한다. 4장에서는 음성 사용자 인터페이스 설계를 위해 대화의 유형별 모델링을 한다. 마지막으로, 5장에서 결론과 향후 과제를 논하겠다.

II. 관련연구

1. 음성 인터페이스 구조

음성 인터페이스의 구조는 현재 World Wide Web Consortium(이하 W3C)에서 [그림 2-1]과 같이 제안하고 있다[5].



[그림 2-1] W3C 음성 인터페이스 구조[5]

음성 인터페이스 시스템의 구성 요소는 다음과 같다.

▪ 음성 인식기(Automatic Speech Recognition : ASR)

음성 인식기는 사용자로부터 음성을 받아들여 텍스트(TEXT)를 생성한다. 음성 인식기는 사용자의 음성을 인식하기 위해 문법을 사용하는데

SGML(Speech Grammar Markup Language[5])을 이용하거나 음성 데이터의 코퍼스로부터 생성된 통계적 문법을 사용하기도 한다. 이러한 문법은 NGML(N-gram Grammar Markup Language[23])를 이용하여 나타낸다.

▪ DTMF 톤 인식기(DTMF Tone Recognizer)

DTMF 톤(Tone) 인식기는 사용자가 전화기의 키패드의 키를 누름으로써 발생되는 터치음을 받아들인다.

▪ 언어 해석기(Language Understanding)

언어 해석기는 미리 명세 된 문법을 이용하여 문자열에 대한 의미를 추출한다. 사용자의 입력이 음성이라면 음성 합성기를 통하여 생성된 문자열을 이용할 것이고, 사용자의 입력이 DTMF 라면 바로 DTMF 톤(Tone) 데이터가 언어 해석기로 전달된다. 언어 해석기 역시 SGML이나 NGML로 명세 된 문법을 사용하며, 언어 해석기의 출력은 NLSML(Natural Language Semantics Markup Language[24])형태로 출력된다.

▪ 문맥 해석기(Context Interpreter)

문맥 해석기는 대화의 이전 기록으로부터 문맥 정보를 획득함으로써 언어 이해 모듈로부터 의미를 강화 시키는 역할을 한다. 문맥 해석기의 입력과 출력은 NLSML형태로 출력된다.

▪ 대화 관리자(Dialog Manager)

대화 관리자는 사용자의 입력에 대한 프롬프트를 제공하며 사용자의 입력에 따라 다음에 무슨 일을 해야 할지를 결정한다. 대화 관리자는 VoiceXML 2.1(Voice Extensible Markup Language 2.1[9])을 이용하여 대화를 작성 할 수 있으며 이는 다른 어플리케이션이나 후위 시스템과도 연동 할 수 있다. 대화

관리자는 NLSML로 명세 된 입력을 수락 할 수 있으며, 대화 스크립트는 다른 대화 스크립트에서 재사용 할 수 있다.

▪ 미디어 설계자(Media Planner)

미디어 설계자는 대화 관리자로 부터의 출력이 사용자에게 합성된 음성으로 나타낼 것인지 아니면 녹음된 오디오로 나타낼 것인지를 결정한다.

▪ 녹음된 오디오 재생기

녹음된 오디오 재생기는 사용자에게 미리 녹음된 오디오 파일을 재생한다.

▪ 언어 생성기(Language Generator)

미디어 설계자로부터 문자열을 받아들여 TTS(Text To Speech)합성기를 통하여 음성으로 사용자에게 제공할 준비를 한다. 이때의 문자열은 SSML(Speech Synthesis Markup Language[25])을 포함하여 사용자에게 어떤 음향학적인 효과를 낼 것인가에 대한 정보를 담을 수 있다.

▪ TTS 합성기

TTS 합성기는 언어 생성기로부터 문자열을 받아들이고 SSML[25]에 명세 된 대로 사용자가 마치 사람의 음성처럼 들을 수 있도록 하는 음성 신호를 생성해 낸다.

2. 음성 사용자 인터페이스의 특징

음성 사용자 인터페이스(VUI)란 사람과 음성언어 어플리케이션이 서로 상호작용하는 것이다. 음성 사용자 인터페이스는 프롬프트(Prompt), 문법(Grammar), 그리고 대화 로직(Dialog logic)으로 구성되어 있다. 프롬프트는 시스템이 사용자에게 들려주는 합성된 음성 또는 미리 녹음된 음성을 말한다. 문법은 각각의 프롬프트에 대한 사용자의 모든 가능한 응답을 정의하고 있고, 대화 로직은 시스템에 의해 발생할 수 있는 모든 가능한 행위를 정의하고 있다. 이러한 구성요소를 가지고 있는 음성 사용자 인터페이스는 아래의 두 가지 특징을 가지고 있다[2].

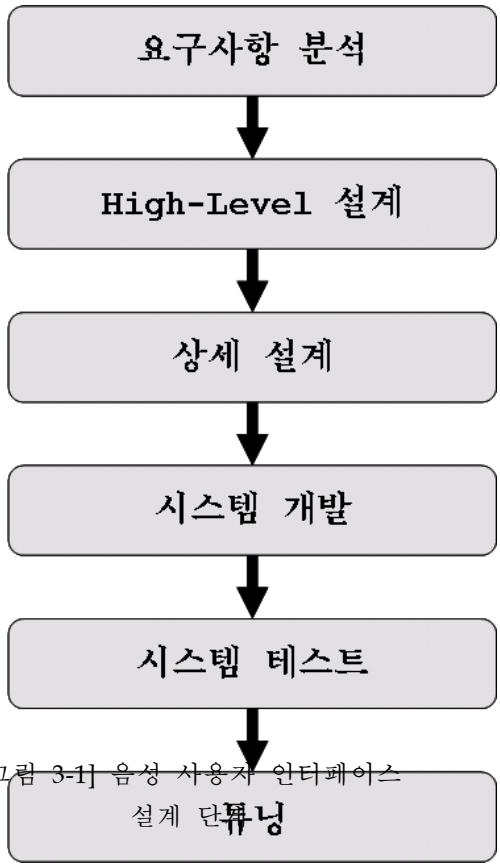
첫째, 음성 사용자 인터페이스의 입력과 출력이 청각을 이용한다는 것이다. 즉, 시스템으로부터의 음성 출력과 사용자로부터의 음성 입력을 소리를 통하여 서로 소통한다는 것이다. 청각을 이용한 음성 인터페이스는 비주얼 웹(Visual-Web) 인터페이스와 달리 지속성 없이 순간적으로 스쳐 지나가는 메시지를 이용하여 사용자와 의사소통을 하기 때문에 설계의 어려움이 많다.

둘째, 음성 사용자 인터페이스는 음성언어에 기반을 두고 있다는 점이다. 음성 사용자 인터페이스와 사용자 간의 의사소통을 이해하기 위해서는 인간 대 인간의 대화방식을 이해하여 그것을 인간과 기계사이의 의사소통에 적용하여야 한다. 인간은 많은 대화의 관습과 인습을 가지고 있고, 또한 대화를 통해 상대방이 다음에 무슨 말을 할 것인지에 대한 기대감을 가진다. 인간의 대화방식에서는 이러한 요소들이 앞으로 전개될 대화의 방향을 결정하는데, 이것을 얼마나 이해하느냐가 성공적인 음성언어를 이용한 인터페이스의 설계에 있어서 가장 중요하다고 할 수 있겠다.

III. 음성 사용자 인터페이스 설계 단계별 방법론

1. 음성 사용자 인터페이스 설계 단계별 방법론

음성 사용자 인터페이스 설계의 단계는 [그림 3-1]과 같이 요구 사항 분석, 고급설계, 상세 설계, 시스템 개발, 테스트, 조율 및 조정으로 크게 여섯 단계로 나눌 수 있다[2].



[그림 3-1] 음성 사용자 인터페이스 설계 단계

(가) 요구 사항 분석 단계에서는 사업상의 목적, 사용자, 어플리케이션 이 세 가지의 입장에서 요구사항을 분석한다.

- 사업상의 목적 측면에서는 사업적 동기, 기업이 추구하는 이미지와 브랜드, 그리고 고객에게 무엇을 기여하고자 하는지를 고려해야 한다.
- 사용자의 요구 사항 수집을 위해서는 누가 시스템을 사용할 것인지, 시스템을 어떻게 사용할 것인지에 대해 알아야 한다. 전자는 시스템을 사용할 대상 사용자가 속해있는 계층, 지적 수준, 유사한 시스템의 사용 경험, 사용자가 바라보는 브랜드이미지 등과 같이 사용자의 특징과 요구를 파악하는 것이다. 후자는 사용자가 시스템을 사용하는 목적이 무엇인지, 시스템의 대상, 용도 등에 대한 정보를 포함한다.
- 어플리케이션의 입장에서 요구사항 분석은 어플리케이션의 각 기능마다 사용자 입력은 어떻게 받고, 사용자의 입력에 대한 결과 값으로서 시스템은 어떤 정보를 반환 할 것이며, 시스템 내부에서는 어떤 정보를 필요로 하는지를 자세히 정의하는 것이다.

(나) 고급 설계 단계에서는 설계 기준의 확립, 대화 설계 전략 및 문법 형식의 결정, 기본 명령어(Universal Command)의 설계, 오류 회복 전략의 설계, 용어 일치, 은유(Metaphor)기법의 설정, 개성부여, 비음성(non-verbal) 오디오 설정 과 같은 작업을 수행한다.

- 설계 기준은 요구 사항 분석에서의 분석 결과를 바탕으로 VUI 설계 기준을 정하는 것이다. 설계기준은 사용자 측면이나 사업상 목적의 측면에서 하나에서 여러 개까지 정할 수 있으나, 가장 통일성 있는 시스템을 구축하기 위해서는 가급적 설계기준이 하나인 것이 가장 좋다.

▪ 대화 설계 전략에는 시스템 주도(Directed) 대화 방식과 상호 주도(Mixed-initiative) 대화 방식이 있다. [그림 3-2]과 [그림 3-3]는 철도 예약 안내 시스템을 두 가지 방식으로 설계한 것이다.

시스템 : 열차 잔여석 안내입니다. 승차 월일 네 자리를 눌러주십시오.
사용자 : 1201 (DTMF)
시스템 : 출발역을 말씀해 주십시오.
사용자 : 서울
시스템 : 도착역을 말씀해 주십시오.
사용자 : 부산
시스템 : 승차 시각 네 자리를 입력해 주십시오.
사용자 : 1000 (DTMF)

[그림 3-2] 시스템 주도 대화(현재, 철도청 ARS 예약 시스템 [5])

시스템 : 열차 잔여석 안내입니다. 여행 일정을 말씀해 주십시오.
예를 들어 서울에서 대구까지 오전 10시 출발 이라고 말씀해 주시면 됩니다.
사용자 : 서울에서 부산까지
시스템 : 몇 시에 출발할 예정입니까?

[그림 3-3] 상호 주도 대화

시스템 주도 대화 방식을 이용하여 여행시스템을 설계한다면, [그림 3-2] 과 같이 시스템은 승차월일, 출발역, 도착역, 승차시각을 차례로 요청하고, 사용자는 시스템이 제시하는 프롬프트에 따라 순서대로 응답하도록 설계해야 한다. 이와 달리 [그림 3-3]와 같이 상호 주도 대화 방식으로 한다면, 시스템의 여행일정 요청에 대한 응답으로 사용자는 다양한 응답이 가능하다. 상호 주도 대화 방식은 시스템 주도 대화 방식보다 시스템과 사용자간의 대화진행이 능동적이며 더욱 자연스럽게 진행된다.

▪ 기본명령어는 시스템이 어떤 대화를 진행하고 있는지에 관계없이

사용자가 시스템을 사용하는 전 과정에서 사용할 수 있는 음성 명령어이다. 가장 흔히 사용되는 기본명령어는 '도움말'이다. 이는 사용자가 어떤 상태에서든 "도움말" 이라고 발화하면, 그 어플리케이션은 현재 대화 상태에서의 상세한 정보를 사용자에게 제공하게 된다.

- 오류 회복 기법은 음성 인식이 부정확할 경우, 시스템이 어떻게 처리하는지를 다룬다. 보통 인식 오류의 종류는 '거절(reject)'과 '입력 시간제한(no speech timeout)'이 있다. '거절'인 경우는 사용자 발화가 인식기의 문법 등에 일치하지 못하는 경우이다. 이 때 시스템은 사용자에게 올바른 답을 유도하기 위하여 단계적 프롬프트 방식과 재 프롬프트 방식을 사용할 수 있다. 단계적 프롬프트는 Yankelovich 등[6]이 제시한 것으로, 사용자의 입력에 대해 계속적인 '거절'이 반복될 경우 사용된다. 이 방법은 처음에는 짧은 오류 프롬프트로 시작하여 단계적으로 점점 더 상세한 내용으로 사용자에게 프롬프트 하는 방법이다. 다른 방법으로는 인식 거절 프롬프트를 좀 더 간결하게 하는 신속한 재 프롬프트(Rapid reprompt)방법이 있다. 이 방법은 시스템이 오류의 내용에 대해 장황하게 설명하지 않고 간략하게 표현함으로써 사용자들로 하여금 장황한 오류 프롬프트에 지치지 않도록 한다. 이 방법은 단계적 프롬프트 보다 효율적인 오류 회복 기법으로 사용 될 수 있다.
- 용어의 일치는 만약 VUI 어플리케이션이 웹 사이트와 같은 다른 시스템과 통합되어 사용될 때 시스템의 통일성을 유지하기 위하여 지켜야 하는 것이다.
- 은유기법은 사용자를 위한 개념 모델(Mental Model)-사용자가 시스템과 어떻게 의사소통 하는지를 나타내는 내부적인 모델-의 생성을 돕기 위해 사용자 인터페이스에서 많이 사용되고 있다. 다시 말하

면, 사용자가 실생활에서 생각하는 물건이나 행동들을 시스템에 적용하여 형상화 하는 것이다.

- 개성부여는 음성을 통하여 조직의 이미지나 서비스를 나타낼 수 있는 매개물이다. 개성부여 설계하기 위해서는 사용자에 대한 시스템의 역할이 무엇인지, 회사가 추구하는 이미지는 무엇인지 사용자 대상은 어떠한지를 고려해야 한다.
- 비 음성 오디오는 시스템에 설계할 음성(프롬프트) 외의 모든 소리를 포함한다. 배경음이나 효과음과 같은 소리음의 사용으로 어플리케이션의 사용자 환경을 보다 강화시켜준다. 그러나 너무 많은 소리음의 사용은 오히려 시스템의 효율성을 저하시키므로 주의해야 한다.

(다) 상세 설계 단계에서는 콜 흐름 및 프롬프트를 설계하고, 에러 회복을 위한 확인(Confirmation) 기법을 설정한다. 또한 운율계획을 세우며, 사용자 테스트를 수행한다.

- 콜 흐름은 시스템을 구성하는 대화의 기본 구조를 나타내는 것이다. 사용자가 시스템을 어떻게 사용할 것인지, 메뉴는 어떻게 구성할 것인지, 사용자에게 정보를 어떻게 표현할지 효과음은 어디에서 재생할지, 다른 시스템과 어느 부분에서 통합을 할지 등과 같은 사항들을 고려하여 콜 흐름도를 설계해야 한다.
- VUI 프롬프트는 음성 언어를 통하여 사용자와 어플리케이션이 서로 의사소통을 하도록 하는 것이다. 프롬프트를 설계하기 위해서 선행해야 할 일은 인간의 자연 발생적인 언어인 담화(Discourse) 형식으로서의 대화를 이해해야 하기 때문에 담화적 특성인 담화표지나 문장의 정보구조를 고려하여 프롬프트 설계를 해야 한다. 또한 문장의 결속성을 위하여 대명사나 시간부사 등을 사용하여 보다 인간의 대

화에 가까운 자연스러운 프롬프트를 설계하는 것이 중요하다.

▪ 확인(Confirmation) 기법은 시스템의 신뢰도가 낮거나, 사용자의 입력으로부터 인식 오류를 막기 위해 사용 한다. 확인 기법으로는 [그림 3-4]와 같이 항목마다 확인하는 기법과, [그림 3-5]와 같이 하나의 그룹을 묶어 확인하는 기법이 있다. 항목별 확인 과정은 그룹으로 묶어 확인 하는 기법을 사용하도록 하되, 4개 이상의 항목을 그룹화 하는 것은 피하도록 한다.

시스템 : 출발역을 말씀해 주십시오. *(Item1 : 출발역)*
사용자 : 서울
시스템 : 출발역은 '서울'입니다. *(Item1 확인)*
아니면 별표를 눌러 주십시오.
시스템 : 도착역을 말씀해 주십시오.
사용자 : 부산
시스템 : 도착역은 '부산'입니다. *(Item2 확인)*
아니면 별표를 눌러 주십시오.
시스템 : 승차 시각 네 자리를 입력해 주십시오. *(Item3 : 승차시각)*
사용자 : 1000 *(DTMF)*
시스템 : 승차시각은 오전 10시입니다.*(Item3 확인)*
아니면 별표를 눌러 주십시오.

[그림 3-4] 항목별 확인

시스템 : 출발역을 말씀해 주십시오. *(Item1 : 출발역)*
사용자 : 서울
시스템 : 도착역을 말씀해 주십시오. *(Item2 : 도착역)*
사용자 : 부산
시스템 : 승차 시각 네자리를 입력해 주십시오. *(Item3 : 승차시각)*
사용자 : 1000 *(DTMF)*
시스템 : 요청하신 정보를 확인해 드리겠습니다.
출발역 서울, 도착역 부산, 승차시각 오전10시입니다.
(Item1 ,Item2, Item3 확인)

[그림 3-5] 그룹으로 확인

- 운율 계획은 사용자에게 올바른 내용 전달을 하기 위하여 적절한 억양, 강세, 리듬, 음성의 톤, 문장의 휴지부(pause) 등을 사용하도록 계획하는 것이다. 운율을 계획 할 시에는 문장의 종류나 태도와 감에 따라 적절한 억양을 사용하도록 하며, 사용자의 기억을 쉽게 하고 자연스러운 발화가 되기 위해서 휴지부를 두어 발화 하도록 한다.
- 사용자 테스트는 앞서 설계한 콜 흐름 설계와 프롬프트 설계를 위한 테스트이다. 시스템 설계 초기에 사용자테스트를 수행하면 시스템에 발생할 수 있는 위험요소를 사전에 예방할 수 있다.

(라) 시스템 개발 단계에서는 어플리케이션과 인식 문법을 개발하고, 음성·비음성오디오를 제작하는 작업을 수행한다.

- VUI 어플리케이션의 개발은 설계한 대화를 코딩하고 다른 소프트웨어 시스템과 데이터베이스와 통합을 하는 것이다. 어플리케이션의 개발은 여러 가지 프로그래밍 언어로 구현될 수 있는데, 음성기반의 어플리케이션 개발은 음성 플랫폼을 이용한다는 특성에 맞게 VoiceXML [4][9]을 이용하여 음성 대화 코딩을 구현할 수 있다.

▪ 인식 문법의 개발은 실제 사용자가 말할 수 있는 가능한 모든 범위를 다룰 수 있도록 해야 한다. 인식 문법 개발의 과정은 규칙기반 문법 방법과 통계적 문법 방법이 서로 다르다.

- 규칙기반 문법방식은 완전한 규칙에 정의된 대로 문법을 생성하는 것이다. 설계자는 문법을 생성할 때 문법에 중심 정보(Slot)와 주변 정보(Filler)를 분리하여 작성하여야 한다. 주변정보는 중심정보를 감싸고 있는 단어나 구(句)이다. 문법을 생성하는 규칙은 ABNF, XML Form형식으로 나타낼 수 있다[10][11][12].
- 통계적 언어 모델은 사용자의 입력이 일정한 규칙 기반 문법으로 다루기 어려울 정도로 광범위 할 때 사용된다. 통계적 언어 모델을 위한 문법의 개발은 사용자가 입력할 수 있는 데이터를 수집하여 반복적인 학습을 통하여 접근한다. 설계자는 사용자들의 언어를 훈련과정을 거쳐서 통계적 언어모델을 생성한다.

▪ 오디오의 제작은 시스템에 사용할 모든 프롬프트 및 비 음성 오디오를 녹음·제작하는 작업이다. 프롬프트를 녹음하기 위해서는 시스템의 목적과 사용자의 요구사항에 부합하는 음성 행위자 (Voice Actor)를 선택해야 한다. 음성 행위자는 시스템과 사용자 사이에서 실제로 사용하는 프롬프트를 낭독해야 하기 때문에 한국어 표준어를 사용하여야 한다. 또한 음성 행위자가 낭독할 스크립트의 내용을 충실히 작성하여야 한다. 스크립트의 내용에는 아이템 번호, 프롬프트 할 내용, 각 프롬프트에 대한 보충설명, 녹음될 파일이름 등을 기입한다. 보충설명에는 프롬프트 할 내용의 발음을 표시하거나 억양이나 강세를 표시하도록 하여 음성 행위자가 낭독하기 편하도록 한다.

(마) 시스템 테스트 단계에서는 어플리케이션 테스트, 인식 및 사용성 테스트 작업을 수행한다.

▪ 어플리케이션 테스트는 음성 대화의 설계가 계획대로 짜임새 있게 구현 되었는지, 사용자로부터의 콜을 시스템이 적절히 수용하는지를 테스트한다. 어플리케이션 테스트는 음성 대화 테스트, 시스템 QA테스트, 작업부하 테스트를 통하여 수행될 수 있다.

  ▫ 음성 대화 테스트는 모든 대화의 상태를 테스트하며 사용자의 입력에 올바른 프롬프트가 재생되는지, 사용자의 입력에 따라 시스템이 올바른 방향으로 수행되는지를 살펴본다. 특히 기본명령어는 제대로 수행 되는지, 오류 처리는 제대로 수행되는지, 등을 테스트 한다.

  ▫ 시스템 QA 테스트는 다른 시스템과 통합하여 테스트를 수행하는 것으로 모든 통합 환경에서 모든 조건을 테스트 한다.

  ▫ 부하테스트는 사용자가 가장 많이 사용하는 시간대에서 부하가 최고점에 이를 때 시스템이 이를 얼마나 효율적으로 대처할 수 있는지를 테스트 한다. 부하테스트는 일반적으로 콜 센터나 서비스 업자에게 소프트웨어를 설치한 후에 시행하며, 동시에 많은 사용자가 접속하여 테스트한다.

▪ 인식 테스트는 허용 범위 안에서 음성 인식이 얼마나 잘 되는지를 테스트 한다. 사용성 테스트는 시스템을 공개하기 전에 수행하며 시스템 운용 중 발생하는 문제점(예: 사용자로 인한 시스템 지연문제, 잡음 환경에서의 반복된 오인식 등)을 발견하기 위해 시행한다.

(바) 튜닝(Tuning) 단계에서는 테스트 단계의 결과를 이용하여 보다 나은 성능을 확보하기 위하여, 음성 대화 튜닝, 인식 튜닝 작업을 수행한다.

▪ 음성 대화 튜닝은 사용자와 시스템 간에 커뮤니케이션이 보다 효율적으로 진행될 수 있도록 조정하는 것이다. 음성 대화 튜닝은 콜 모니터링, 콜 로그 분석을 위한 유틸리티를 사용하거나, 사용자 경험 연구로

음성 대화 성능을 평가할 수 있다.

▪ 인식튜닝은 인식 문법에 대한 인식의 정확성을 높이기 위한 것이다. 인식튜닝의 첫 단계는 인식 문법 성능을 측정하는 것이다. 이것은 사용자의 발화에 대해 인식기의 인식 결과를 서로 비교해 봄으로써 이루어진다. 그 결과는 인식대상문법(**I**n-grammar) 데이터와 비 인식대상문법(Out-of-grammar) 데이터로 나눌 수 있다. 인식대상문법(**I**n-grammar) 데이터는 대화의 각 상태와 관련된 문법에 정의된 것을 사용자가 말하는 경우이고, 비 인식대상 문법(Out-of-grammar)은 문법에 정의되지 않은 것을 사용자가 말하는 경우이다. 인식대상문법(**I**n-grammar) 데이터는 경우에 따라 세 가지로 분류된다. 인식기가 정확한 답을 반환하는 일치 수락(Correct Accept), 인식기가 틀린 답을 반환하는 오인 수락(False Accept), 인식기가 문법의 어떤 경로에서도 입력 값에 일치되는 답이 없을 때 답을 반환하지 않고 거절하는, 오인 거부(False Reject)로 분류된다. 비 인식대상문법(Out-of-grammar) 데이터는 두 가지로 분류된다. 인식기가 사용자의 입력에 정확히 거절되는 일치 거부(Correct Reject)와 문법의 정의에 없는 입력에 대한 잘못된 답을 반환하는 오인 수락(False Accept)로 분류된다. 인식 성능은 이 값들의 비율에 따라 결정되며, 인식기의 인식 파라미터를 조정함으로써 인식 성능이 개선될 수 있다.

2. 한국어 음성 특성을 반영한 VUI 설계

한국어를 기반으로 한 음성 사용자 인터페이스를 설계하기 위해서는 한국어 음성의 고유한 특징을 설계 단계에 반영해야 한다. 한국어의 고유한 특징이 반영되어야 할 설계 부분으로는 프롬프트 설계, 운율 계획, 인식 문법의 개발 단계이다.

2.1 한국어 프롬프트의 설계

프롬프트의 설계는 인간의 담화 맥락에 맞도록 설계해야 한다고 했다. 담화는 대화가 하나의 구(句)나 단어로만 의미전달이 되는 것이 아니라 문장 전체의 맥락에 의존한다[7][8]. 효과적으로 사용자에게 담화의 내용을 전달하기 위해서는 한국어 담화표지나 정보구조를 이용한다.

담화표지의 기능은 화자가 청자에게 발화 의도나 심적 태도를 보다 잘 드러내어 청자를 이해시키기 위해 사용하는 것이다. 한국어 담화 표지는 [표 3-1]와 같이 다양하게 사용할 수 있다.

어휘	담화표지
부사	그래, 그래도, 그래서, 그러나, 그러니까(근까), 그런데(근데), 그럼, 그러면, 그렇지만, 그리고, 그 담에, 정말로, 진짜로, 아무튼, 어쨌든,
감탄사	이제(인제, 인자, 인저), 참, 하여간, 어디, 가만 자, 뭐, 뭘, 저, 아니, 글쎄, 어, 음, 아, 아이고, 어머, 오, 야, 이봐,
대명사	거시기, 저기(요), 이거, 그거, 저거
용언	뭐지, 뭐야, 뭐랄까, 있잖아. 있지, 말입니다, 말하자면, 말이지
관형사	이, 그, 저, 이런, 그런, 어떤, 무슨
조사	-요, -그래, -만

[표 3-1] 한국어 담화 표지의 실현[13]

정보구조는 대화의 내용을 신(New)정보와 구(Old)정보로 구분한다. 상대방에게 전달하고자 하는 중심 정보를 신정보라 하고 주변 정보를 구정보라고 한다. 사용자의 기억력의 한계로 인하여 신정보의 위치를 구문의 후반부에 두는 것이 사용자에게 전달하기에 효과적이다. 한국어 대화체 구문의 선호 담화 정보 구조는 아래와 같다[14].

첫째, 한 개의 절에는 한 개 이하의 신정보 명사구가 실현된다.

둘째, 타동사의 주어는 신정보의 기능을 하지 않는다.

셋째, 정보구조는 구정보-신정보의 순으로 실현된다.

한국어는 영어권이나 여타의 언어와 달리 다음과 같은 고유한 특징을 갖는다.

▪ 한국어 대용현상

하나의 문장이나 문장 사이에 같은 내용이 되풀이 될 때 언어 사용의 경제성을 위하여 잉여적 표현을 제거 하는 것으로 되풀이된 요소를 대명사와 같은 간략한 언어 표현으로 대치하여 쓰는 현상이다. 한국어는 다양한 대용어를 사용할 수 있으나, 대용어의 선행어에 따라 대용어의 어휘가 달라진다[표 3-2]. 음성 인터페이스 설계자는 이를 유념하고 선행어에 맞는 대용어를 사용해야 한다. 그러나 대용어의 사용은 프롬프트를 경제적으로 설계 할 수 있다는 이점이 있으나 사용자가 대용어를 사용하여 응답 했을 경우에 이를 해석하기 위해서는 구문 전체 문맥을 고려하여야 하는 경우가 발생할 수 있다. 그렇기 때문에 대용어를 이용하여 프롬프트를 설계할 때에는 화자의 의도를 고려하여 이전 문맥으로부터 화자의 의도를 유추할 수 있거나 보다 제한적으로 화자에게 명확한 의도를 나타낼 수 있도록 프롬프트를 설계한다.

선행어	대용어
1인칭 사람	본인, 자기, 자신, 그자신, 그
3인칭 사람	그사람, 그여자, 그남자, 그분,
사물	그것, 저것, 거시기, 것, 그거, 고거
방향	그쪽, 그리, 저쪽, 저리, 이쪽, 이리

[표 3-2] 선행사에 따른 대용어 표현

▪ 한국어 구어체와 문어체의 특징

한국어 구어체에서는 발화 상황에 따라 주어나 목적어가 생략되는 경우가 많고, 어순이 자유롭다. [그림 3-6]의 ㄱ. 은 주어가 생략되었고, ㄴ. 은 주어와 목적어가 생략되었으며 ㄷ. 은 어순이 바뀌었다. 비록 주어와 목적어가 생략되거나 어순이 바뀌었다 할지라도 충분히 사용자에게 정보를 전달 할 수 있음을 알 수 있다.

ㄱ. 서비스를 계속 진행 하시겠습니까?

ㄴ. 종료하시겠습니까?

ㄷ. 계속 서비스를 진행 하시겠습니까?

ㄹ. 당신은 서비스를 계속 진행 하시겠습니까?

[그림 3-6] 생략 현상과 어순 뒤바뀜

그러나 문어체에서는 문장성분들이 생략되는 경우가 그리 많지 않다. [그림 3-6]의 ㄹ.과 같은 문어체의 사용은 오히려 사용자와 시스템간의 대화체에서는 부자연스럽다. 그렇지만 사용자에게 보다 구체적으로 정보를 전달하거나 요청할 경우에는 주어와 목적어를 모두 생략하기 보다는 문어체에 가깝되 구어적 특징을 반영하여 프롬프트를 설계하는 것이 효과적이다.

▪ 한국어의 다양한 음운 현상

한국어는 된소리 현상, 장단음, 자음 탈락, 모음 탈락, 자음 동화 등과

같이 여러 가지 음운현상이 존재한다. 이러한 음운 현상은 미리 녹음된 프롬프트를 이용한다면 자연스러운 프롬프트를 설계 할 수 있으나, 합성기를 통해 프롬프트 할 경우 음운적 특징을 고려하여야 한다. 또한 사용자가 음운현상을 일으키는 응답을 했을 경우에 인식기나 문법이 이러한 음운현상을 일으킨 단어나 문장에 대해서 원래 기본 값으로 인식할 수 있어야 한다. 그러나 현재 인식기의 성능으로는 고립된 숫자음보다 연속된 숫자음의 오인식률이 더 크다. 특히 일-이, 일-칠, 이-일, 오-구 의 숫자음이 오인식이 많이 발생한다[15]. 이는 화자의 발음 습관에 따라 '삼'의 경우 '삼', '쌈', '싸'으로, '육'의 경우 '육', '륙' 으로, '일이'의 경우 '일이', '일리'와 같이 다양하게 발음되기 때문이다. 기차예매나 증권 서비스와 같이 사용자의 응답으로 연속된 숫자음을 많이 요하는 서비스라면 프롬프트 설계 시 오 인식을 막기 위해 DTMF로 사용자에게 숫자를 입력해 달라고 설계 하는 것도 하나의 방법이 될 수 있겠다.

▪ 한국어의 대우법

한국어 대우법은 계층, 성별, 연령 등에 따라 다양하게 표현한다. 대우법은 화자와 청자 그리고 제 3자 사이에서 성립한다. 대우법은 동사에 -시-, -사오-, -자오-, -오-, -옵-, -사옵-, 등의 선어말 어미를 이용하여 나타낸다. 또한 진지, 말씀, 존함, 성함, 드리다, 생신 등과 같이 존대어휘 혹은 -께서, -께와 같은 조사를 이용하여 표현할 수 있다[16]. 한국어 대우법에 맞도록 표현하려면 어휘의 사용도 함께 호응해야 한다. [그림 3-7]의 ㄱ.은 대우법과 존대어휘 간의 호응이 부적절한 경우이나, ㄴ.은 적절한 경우이다.

ㄱ. 선생님, 생일 축하해요.

ㄴ. 선생님, 생신 축하드립니다.

[그림 3-7] 대우법의 사용

▪ 호칭어의 다양성

비교적 단순한 호칭 체계를 지닌 영어권과 달리 한국어는 호칭 체계가 다양하다. 호칭어란 화자가 청자와의 의사소통 과정에서 상대를 부르기 위해 사용되는 부름말을 말한다[17].

한국어 호칭어는 이름 호칭어, 직함 호칭어, 친족 호칭어, 대명사 호칭어, 통칭적 호칭 유형으로 분류된다. 이름 호칭어는 이름에 호격 조사 (-아/야, -씨, -님 등)가 붙어 화자와 청자의 관계를 나타낸다. 직함 호칭어는 직함에 접미사 -님을 붙여 상대방을 높이는 역할을 한다. 친족 호칭어는 친족어(아버지, 아빠, 어머니, 엄마, 아버님, 어머님, 형님, 형아)과 같이 친족 간의 호칭을 나타낸다. 대명사 호칭어는 2인칭 단수 대명사(너, 자네, 당신, 그대, 자기, 어르신)를 지칭하기 위해 나타낸다. 마지막으로, 통칭적 호칭어는 청자의 직함이나 청자와 화자와의 관계와 상관없이 지칭하는 대상(선생님, 손님, 고객님)을 지칭한다[18].

한국어를 위한 음성 인터페이스의 프롬프트 설계에 있어서 설계자는 사용자에게 한국적 정서인 공손한 태도로 프롬프트하기 위해 청자를 존대하는 경어법과 사용자층에 맞는 호칭어를 사용해야 한다. 특히, 개인적인 정보를 요구할 경우에 이름 호칭어(예. OOO고객님, OOO님)를 사용하여 사용자가 자신의 정보임을 확인할 수 있도록 하는 것도 좋은 방법이다. 그러나 현재 ARS와 같은 음성 인터페이스를 이용한 서비스에서의 호칭은 통칭적 호칭(예. 고객님)을 많이 사용한다.

지금까지 한국어가 가지는 고유적 특성이 프롬프트 설계 영역에 어떻게 반영 되어야 할지를 논하였다.

마지막으로 프롬프트 설계 시 담화 태도에 대해 논해 보겠다.

[그림 3-8]은 휴대전화 요금납부에 관한 사용자의 문의에 대한 시스템의 답

변에 대한 담화 태도의 예이다. 사용자와 시스템간의 응답문을 통한 시스템의 담화 태도에 있어서 시스템은 사용자 입장에서 전문용어를 일상적인 언어와 표현으로 바꾸어 쓰고 답변의 내용이 일반적인 것과 좀더 세부적인 것이 있다면 일반적인 것을 먼저 언급하고 세부적인 것으로 옮겨 가도록 한다. 또한 사용자가 요청한 질문에 핵심 정보를 제공하는 것이 의미적 응집성과 응결 구조를 견고하게 한다. 마지막으로 상호간 의사소통의 마지막 발화 교환의 시점에서 발화 교환의 또 다른 연쇄를 기대하며 개방하는 궁금한 사항이 있으면 언제든 도와주겠다는 표현이나 사용자 자신이 발화 교환의 기회를 매우 긍정적으로 받아들이고 있다는 것을 표시할 수 있는 감사의 표현을 덧붙이는 것도 발화 교환의 완결성과 개방성을 표시하는데 도움을 준다[19].

안녕하세요! XX 텔레콤 도우미 입니다.

~~OOO 고객님, 먼저 XX텔레콤 서비스를 이용 해 주신점 감사드립니다.~~

OOO 고객님께서는 요금 결제일 변경과 관련 하여 문의 주셨습니다.

문의 주신 내용에 대해 안내를 해 드리겠습니다.

문의 주신 아이디의 번호를 조회 해 보니,

현재 고객님께서는 납부 방법이 은행자동이체로 등록되어 있습니다.

은행자동이체의 결제일은 매월 22일이오나,

22일이 주말이나 공휴일일 경우에는 다음날로 연장됩니다.

은행 이체일이 정해져 있어 고객님께서 요청 해 주신 26일로 결제일을 변경하는 것은 가능하지 않음을 양해 바랍니다.

부족하나마 저희 답변이 OOO고객님께 도움이 되었길 바랍니다.

다른 궁금한 사항이 있으시면 언제든지 문의 주십시오.

상담실을 이용 해 주셔서 감사드립니다.

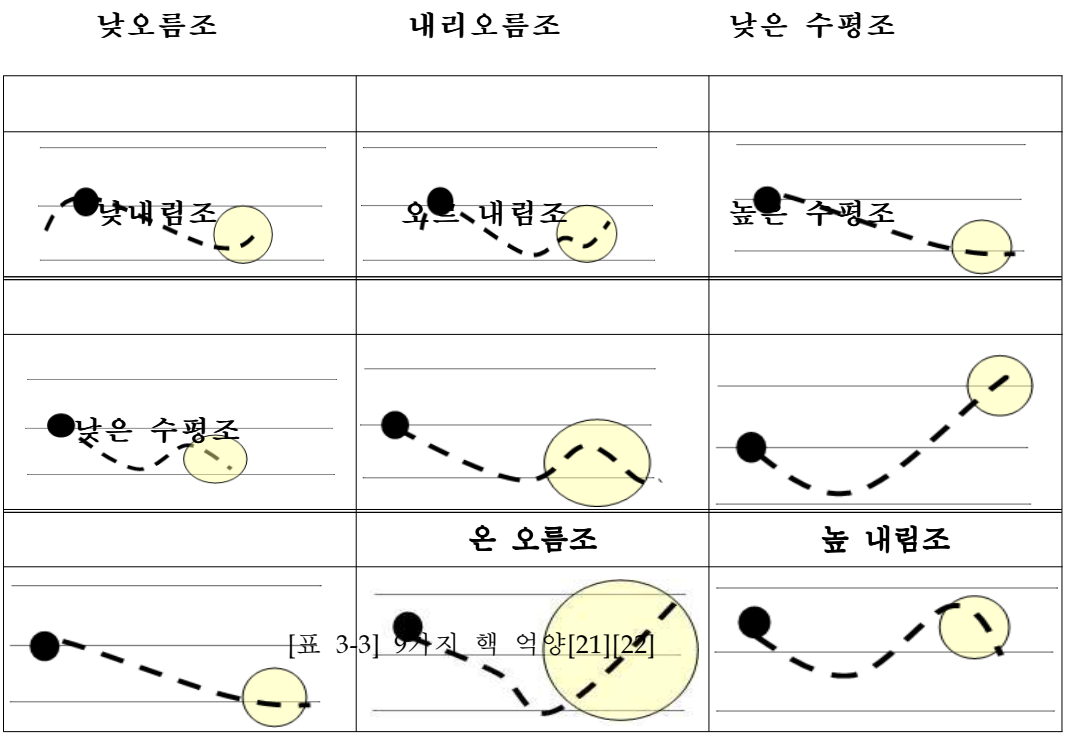
편안하고 행복한 하루 되십시오.

[그림 3-8] 담화태도의 예

2.2 한국어 억양 계획

억양이라는 개념은 크게 발화시 음높이가 일정한 유형을 나타내면서 만들어 내는 말의 가락으로써, 화자의 감정을 표현하는 감정적, 정서적 기능을 한다[20][21][22].

한국어의 억양은, 문장 보다는 작지만 말토막보다는 큰 '말마디'에 음의 높낮이를 얹어 9개의 핵 억양으로 구성할 수 있다[21][22]. 말토막이란, 리듬의 단위인 동시에 의미 및 정보의 단위로서 하나의 강세 음절을 가지며 말토막의 경계에는 휴지부가 온다[21]. 9가지 핵억양은 [표 3-3]과 같다. 9가지 억양계획은 문장의 종류나([표 3-4]), 화자의 태도 및 감정([표 3-5])에 따라 달리 사용될 수 있다[22].



[표 3-3] 9가지 핵 억양[21][22]

문장의 유형	핵 억양
평서문	낮은 수평조가 주로 사용, 낮내림조, 가운데 수평조, 오르내림조, 낮오름조, 내리 오름조
예-아니오 의문문	높은 수평조가 주로 사용, 높 내림조, 온 오름조, 낮은 수평조, 오르내림조, 낮오름조, 내리 오름조
의문사 의문문	낮은 수평조, 낮 내림조, 오르내림조, 가운데 수평조, 낮 오름조, 내리오름조
선택 의문문	앞마디에는 가운데 수평조, 낮오름조가 선택되고, 뒷마디에는 낮은 수평조, 낮내림조가 선택
되물음 의문문	높은 수평조, 높내림조, 온오름조, 내리오름조, 낮오름조, 내리오름조
명령문	낮은 수평조, 낮내림조, 오르내림조, 낮오름조, 내리 오름조선택
청유문	낮은 수평조, 낮내림조, 오르내림조,가운데수평조, 낮오름조,내리오름조

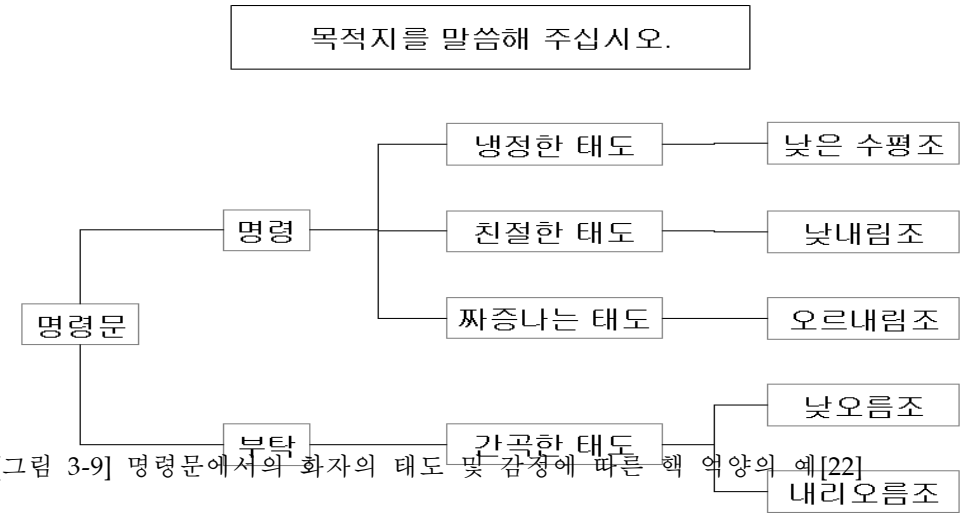
[표 3-4] 문장의 유형에 따른 핵 억양[21][22]

화자의 태도 및 감정	핵 억양
냉정한, 사무적인	낮은 수평조
친절한 부드러운	낮내림조
달래는 안심시키는	낮오름조, 내리 오름조
흥미로운	높은 수평조
감동적인, 비꼬는	오르내림조
질책하는	내리오름조
퉁명스러운	가운데 수평조
놀라운, 의심스러운	온오름조, 높내림조, 높은수평조

[표 3-5] 화자의 태도 및 감정에 따른 핵 억양[21][22]

한국어 프롬프트의 설계시 전달하고자하는 내용과 문장 유형 및 태도에 따라 9가지 핵 억양을 이용하여 적절한 억양 계획을 수립하여 자연스러운 발화가 될 수 있도록 해야 한다. [그림 3-9]는 시스템의 "목적지를 말씀해 주십시

오"라고 사용자에게 요청했을 때의 억양 계획이다. 명령문은 문장의 유형과 화자의 태도 및 감정에 따라 낮은 수평조, 낮내림조, 오르내림조, 낮오름조, 내리 오름조가 올 수 있다[22]. 이때의 경우에는 간곡한 태도를 나타내는 낮 오름조나 내리 오름조가 혹은 친절한 태도를 나타내는 낮 내림조로써 억양 계획을 수립하면 적절할 것이다.



[그림 3-9] 명령문에서의 화자의 태도 및 감정에 따른 핵 억양의 예[22]

비교적 고립단어의 발화시에는 억양이 바르게 지켜진다 할지라도 전체적이며 연속적인 문장에서 억양이 부자연스럽다면 사용자에게 문맥 이해의 혼란을 야기 할 수 있으며, 전반적인 음성인터페이스 서비스의 질이 낮은 평가를 받을 수 있기 때문에, 비록 단어의 발화는 부정확 하더라도 발화 문장 전체의 억양이 자연스럽도록 하는 것이 사용자에게 훨씬 유창하게 인식될 것이다.

2.3 한국어 인식 문법의 개발

인식 문법의 개발 시에는 한국어의 다양하고 고유한 특징을 반영하여 한국어를 모국어로 하는 사용자의 언어 습관을 고려하여 설계해야 한다.

한국어는 여타의 언어와 달리 사물을 설명하는 표현이 다양하다. 가장 대표적인 예가 색채의 표현이다. 한국어의 색채어는 '검다'를 '거뭇거뭇하다', '거무퉤퉤하다', '시커멓다', '새까맣다' 등과같이 기본 색채어에 화자의 주관적 감정이 개입되어 표현되기도 하고, '팥색', '쥐색', '쥐색', '비둘기색' 등과 같이 다른 사물에 비유하여 표현되기도 한다. 또한 젊은 층 사이에서는 '카키색'이 중장년층에서는 '국방색'으로 표현 하는 것과 같이 연령에 따라 그 표현 양식이 다양하다.

또한 한국어 대화체에서는 간투사가 빈번히 사용된다. 간투사는 "음..날씨정보로 이동" 에서의 '음..'과 같이 화자가 전달하고자하는 핵심내용과는 상관없는 사용자의 습관적인 언어 행위이다.

이와 같이 한국어의 다양한 표현을 인식 문법에서 다루어 사용자의 언어습관이 베어든 다양한 표현이 가능하도록 해야 할 것이다.

IV. 음성 사용자 인터페이스를 위한 대화 설계 모델링

음성 사용자 인터페이스의 가장 기본 요소는 대화이다. 대화는 시스템과 사용자의 커뮤니케이션을 위한 최소 단위이다.

대화는 [그림 4-1]과 같이 시스템이 사용자에게 여러 개의 정보를 요청하는 형태가 있고, [그림 4-2]와 같이 시스템이 사용자에게 하나의 정보만을 선택하도록 하는 형태가 있다.

대화 설계 모델링은 이러한 대화 형태들을 정형화한 것이다. 대화 설계 모델은 음성 사용자 인터페이스에서의 대화를 설계할 때 설계자가 보다 용이하게 설계할 수 있는 틀을 제공한다.

시스템 : 승차월일을 말씀해 주십시오.
사용자 : 12월 2일
시스템 : 목적지를 말씀해 주십시오.
사용자 : 서울
시스템 : 출발지를 말씀해 주십시오.
사용자 : 부산
시스템 : 승차시각을 말씀해 주십시오.
사용자 : 오전 10시

[그림4-1] 양식 채우기 대화 형식

시스템 : 날씨정보와 증권서비스 중 원하는 서비스를 말씀해 주십시오.
사용자 : 날씨

[그림 4-2] 메뉴 선택 대화 형식

1. 대화의 분류

대화는 다음과 같이 크게 2가지로 분류할 수 있다.

- 양식 채우기(Form-Filling) 형식 : 이 형식은 다음 작업을 수행하기 위하여 사용자로부터 하나 이상의 정보 항목 입력을 요구하는 대화이다. 여기서 양식은 하나 이상의 정보 입력 항목(field)으로 구성되며, 양식을 구성하는 모든 정보 항목이 입력되어야 시스템이 다음 작업을 진행할 수 있다. 기차예약의 경우에는 승차월일, 목적지, 출발지, 승차시각 등의 정보 항목을 요청하는 것이 그 예이다. 이 대화 방식은 시스템과 사용자 간의 질의·응답의 흐름의 순서가 시스템에 의해 주도되는지, 사용자와 시스템이 상호 주도 하는지에 따라 시스템 주도 대화 방식과 상호 주도 대화 방식으로 세분된다.
  - 시스템 주도(Directed) 대화 : 프롬프트를 통한 시스템의 질의에 사용자는 하나의 응답만 할 수 있으며, 프롬프트의 순서가 미리 정해져 있다.
  - 상호 주도(Mixed Initiative) 대화 : 프롬프트를 통한 시스템의 질의에 사용자는 여러 개의 응답을 할 수 있으며, 사용자의 응답에 따라 시스템의 다음 질의가 결정된다.

- 메뉴 선택(Menu-Selection) 형식 : 이 대화 방식은 사용자가 시스템이 제공하는 정보 리스트 중에서 하나를 선택하는 방식이다. [그림 4-2]에서 "날씨정보와 증권서비스 중 원하는 서비스를 말씀해 주십시오"처럼 시스템은 사용자가 어떤 응답을 선택 할 수 있는지를 말해준다.

2. 대화 설계의 공통 객체

앞서 분류한 대화 모델은 각 모델별 대화 형식은 다르지만, 다음과 같은 공통 객체가 사용된다.

첫째, 사용자에게 입력받을 문법 객체,

둘째, 사용자에게 정보를 요구하는 프롬프트(prompt) 객체,

셋째, 현재의 대화가 종료되었을 때 다음에 수행할 행위를 기술하는 객체,

넷째, 현재의 대화가 하위 대화를 호출할 때 사용하는 하위 대화 호출 객체,

마지막으로, 대화에서 발생하는 이벤트를 위한 이벤트 처리 객체이다.

2.1 문법객체

문법 객체(Grammar object)는 대화에서 사용자의 음성 인식을 위한 문법을 정의하는 것이다. 본 논문에 정의한 문법 객체는 ABNF 문법형식을 따른다[10][11][12]. 문법 객체는 대화가 폼 필링 시스템 주도 방식, 폼 필링 상호 주도 방식, 그리고 메뉴 선택 방식에 따라서 문법의 복잡도가 다르다. 이는 각 대화의 방식에 따라 사용자로부터 입력받는 경우의 수가 다르기 때문이다.

기본적으로 문법 객체를 참조하기 위해서는 다음과 같은 파라미터와 값이 필요하다.

파라미터	값
location	*external* | *inline*
mode	*dtmf* | *speech*
ref	*GrammarFileURI#rule_name*

- 문법객체의 위치(**location**) : 문법 객체의 위치가 외부 파일로 존재하면 external, 대화 내에서 직접 작성할 수 있으면 inline 으로 값을 설정한다.
- 문법객체의 형식(**mode**) : 사용자에게 DTMF로 입력을 받을 것이면 dtmf, 음성으로 입력을 받을 것이면 speech 로 값을 설정한다.
- 문법객체의 참조(**ref**) : 대화에 적용할 문법 객체의 근원지로 URI을 지정한다. URI는 *GrammarFileURI#rule_name*의 형태로 기술된다.

문법 객체가 외부 파일로 존재하든, 대화 내에 직접 작성하든 간에 문법 객체는 다음과 같이 모델링할 수 있다([표 4-1]).

구성 요소	값
grammar_id	*String*
scope	*dialog | field*
grammatical_rule	$\$rule\_name_1$ = *rule_expansion*$_1$; *rule_expansion*$_1$ = $(PV_{1.1}^1 \mid ... \mid PV_{1.1}^{m_1})\{RV_{1.1}\}$ ... $(PV_{1.n}^1 \mid ... \mid PV_{1.n}^{m_n})\{RV_{1.n}\}$ ... $\$rule\_name_k$ = *rule_expansion*$_k$; *rule_expansion*$_k$ = $(PV_{k.1}^1 \mid ... \mid PV_{k.1}^{t_1})\{RV_{k.1}\}$ ... $(PV_{1.r}^1 \mid ... \mid PV_{1.r}^{t_r})\{RV_{k.r}\}$ $\$rule\_name_v$ = $\$rule\_name_1 \mid .. \mid \$rule\_name_k$ *PV=Possible value,* *RV= Representative value*

[표 4-1] 문법 객체의 구성요소

▪ 문법 식별자(**grammar_id**) 는 문자열(을 그 값으로 갖는다.

(예. **grammar_id** = *stock*)

String)

▪ 문법 범위(**scope**)는 그 값으로 나 중 하나의 값을 가질 수 있다.

(예. **scope** = *dialog* *field*

dialog)

▪ 문법 규칙(**grammatical_rule**)은 문법이 적용될 필드의 이름을 규칙이름으로 한다. 는 그 문법 규칙에 적법한 표현 값으로 을 정의한다. 는 과 동

$\$rule\_name$ $\$rule\_name_k$

$RV_{k.1}$, ..., $RV_{k.r}$ $(PV_{k.1}^1 \mid ... \mid PV_{k.1}^{t_1})$ $RV_{k.1}$

일한 표현 값으로 올 수 있는 값들이다. 문법 규칙은 와 $rule_name$v 같이 다른 문법 규칙을 참조하여 사용할 수도 있다.

- 예) $도착지 = (서울|설|쎄울){서울}

2.2 프롬프트(prompt)객체

프롬프트 객체는 합성된 음성이나, 녹음된 음성을 이용하여 사용자에게 입력을 요청하는 것이다.

프롬프트 객체는 **tts**와 **audio**타입을 가지며, 두 타입 중 하나만 선택해서 사용할 수 있다.

프롬프트 객체는 **tts**와 **audio** 타입에 상관없이 의 속성을 가지고 있다. 속성은 사용자가 프롬프트 중에 발화한 경우 즉각적으로 사용자의 입력을 받기 위해 사용한다. 프롬프트에서 기능을 사용하려면 을 명시하도록 한다.

barge-in

barge-in

barge-in

barge-in

▪ **tts** 타입

: tts 타입은 문자열( )을 값으로 받아 합성기를 통하여 사용자에게 프롬프트 한다. tts 타입은 속성으로 를 가지고, 속성 값으로는 사용자에게 프롬프트 할 내용을 문자열로 입력한다.

String

text

▪ **audio** 타입

: **audio** 타입은 미리 녹음된 파일로 사용자에게 프롬프트 할 때 사용한다. **audio** 타입은 속성 를 가지고, 속성 값으로 프롬프트 할 audio 파일의 *URI*를 받는다.

src

src

2.3 후속 작업(next_action) 객체

후속 작업 객체는 현재의 대화가 종료되어 다른 대화나 후위 시스템으로 이동할 때 사용된다. 후속 작업 객체는 **transfer, submit, return** 타입으로 구성된다([표 4-2]). 후속 작업 객체의 세 가지 타입(**transfer, submit, return**) 중 한 가지만 선택되어 사용할 수 있다.

	타입	속성	값
next_action	transfer	next	*dialog_id*
	submit	target	*URI*
	return	-	-

[표 4-2] 후속 작업 객체의 종류와 값

▪ **transfer** 타입

: **transfer**타입은 현재의 대화가 종료되면 다른 대화로 이동할 때 사용된다. **transfer**타입은 속성으로 next 를 가지며, 이동할 대화의 식별자를 값으로 가진다.

▪ **submit** 타입

: **submit**타입은 현재의 대화가 종료되어 다른 시스템이나 후위 시스템으로 이동할 때 사용된다. **submit**타입은 현재의 대화에서 지정한 변수를 받아 다른 시스템이나 후위 시스템으로 변수를 값으로 전달 할 수 있다. **submit**타입은 target 속성을 가지고 있다. target 속성은 이동할 다른 대화나 후위 시스템의 주소를 *URI* 값으로 가진다.

▪ **return** 타입

: **return**타입은 현재의 대화가 하위 대화로 사용되었을 경우에 사용한다.

2.4 하위 대화 호출(subdialog_call) 객체

하위대화 호출 객체는 현재의 대화가 수행 도중에 하위대화를 호출 할 때 사용된다. 하위 대화 호출 객체 속성은 호출할 대화의 식별자를 값으로 가진다.

속성	값
subdialog_call	
src	*dialog_id*

[표 4-3] 하위 대화 호출 객체의 속성

2.5 이벤트 처리(event_handling) 객체

대화 영역이나 필드 영역에서 이벤트가 발생했을 때 이벤트를 처리하기 위한 것이다.

이벤트 처리 객체는 세부객체로 noinput, nomatch, help로 분류된다.

각각의 이벤트 처리 객체는 noinput, nomatch, help 이벤트의 발생 허용 횟수를 [표 4-4]와 같이 미리 지정한다.

이벤트 발생 허용 횟수 (#define)	값	설명
noinput_count	*integer*	noinput이벤트의 발생 허용 횟수를 지정한다.
nomatch_count	*integer*	nomatch이벤트의 발생 허용 횟수를 지정한다.
help_count	*integer*	help 이벤트의 발생 허용 횟수를 지정한다.

[표 4-4] 이벤트의 발생 허용 횟수 지정

미리 지정된 noinput, nomatch, help이벤트의 발생 허용 횟수 값에 따라 그 수만큼 이벤트 처리 객체가 필요하다.

▪ noinput 이벤트 처리 객체

: noinput 이벤트 처리 객체는 일정시간 동안 사용자의 입력이 없을 때 발생하는 이벤트를 처리하기 위해 사용한다. 하나의 noinput 이벤트 처리 객체는 [표 4-5]와 같은 속성을 가진다.

속 성	값
count	*integer*
timeout	*integer*
prompt	(프롬프트 객체 참고)
next_action	(후속 작업 객체 참고)

[표 4-5] **noinput** 이벤트의 속성

▫ **count**

: **count**는 noinput 이벤트 처리 객체가 몇 번째 발생한 noinput 이벤트 처리 객체인지를 나타낸다.

▫ **timeout**

: **timeout**은 시스템이 사용자 입력을 받기 위해 대기하는 시간을 나타낸다. noinput 이벤트의 발생 횟수에 따라 timeout의 값을 달리 지정할 수 있다.

▫ **prompt**

: noinput 이벤트가 발생 했을 때 사용자에게 noinput 이벤트 처리를 위해 프롬프트 객체를 이용하여 정보를 요구한다.

▫ **next_action**

: **next_action**은 미리 지정한 noinput 발생의 허용횟수만큼 noinput 이벤트가 발생했을 때, 다른 대화나 후위 시스템으로 이동하기위해 사용한다.

▪ **nomatch** 이벤트 처리 객체

: **nomatch** 이벤트 처리 객체는 사용자의 입력이 해당 문법에 없을 때 발

생하는 이벤트를 처리하기 위해 사용한다. 속성은 **timeout**을 제외한 **noinput** 이벤트 처리 객체 속성과 동일하다(**noinput** 이벤트 처리 객체 속성참조). 단, prompt 속성에서 prompt 객체를 정의할 때 prompt 내용이 **nomatch** 이벤트 발생 이유에 맞도록 사용자의 응답이 문법과 일치 하지 않는다는 의미를 포함 시키도록 한다.

▪ **help** 이벤트 처리 객체

: **help** 이벤트 처리 객체는 사용자가 '도움말'이라고 발화할 때 발생하는 이벤트를 처리하는 객체이다. **help** 이벤트 처리 객체는 도움말 정보를 담은 다른 하위 대화를 호출한다. **help** 이벤트 처리 객체의 속성은 다음과 같다([표 4-6]).

속 성	값
count	*Integer*
subdialog_call	(하위 대화 호출 객체 참조)

[표 4-6] **help** 이벤트 처리 객체 의 속성

▫ **count**

: **count**는 **help** 이벤트 객체가 몇 번째 발생한 **help**이벤트 객체인지를 나타낸다.

▫ **subdialog_call**

: **help** 이벤트가 발생하면 도움말 정보를 담은 하위 대화를 호출한다.

▪ 이벤트 처리 객체의 예

: noinput이벤트 발생 허용 횟수 2, nomatch 이벤트 발생 허용 횟수 2, help 이벤트 발생 허용 횟수가 2인 이벤트 처리 객체는 다음과 같이 정의 된다.

{이벤트 처리 객체]

#define : noinput_count = 2

#define : nomatch_count = 2

#define : help_count = 2

noinput 이벤트 처리 객체

count	*1*
timeout	*60*
prompt	*"죄송합니다."*

noinput 이벤트 처리 객체

text		*barge_in*

count	*2*
timeout	*60*
prompt	*nonput1.wav*

nomatch이벤트 처리 객체

src		*barge_in*

count	*1*
prompt	*"입력이 되지 않았습니다.."*

nomatch이벤트 처리 객체

text		*barge_in*

count	*2*
prompt	*"죄송합니다. 다시한번...."*

help이벤트 처리 객체

text		*barge_in*

count	*1*
return	*help_dialog1*

help이벤트 처리 객체

count	*2*
return	*help_dialog2*

3. 대화 설계

3.1 폼 필링 시스템 주도(Form Filling Directed) 대화

폼 필링 시스템 주도 대화는 [그림 4-3]에서와 같이 하나의 대화에서 채워야 하는 정보항목, 즉 필드(field)의 입력이 일련의 순서를 가지며 한 번에 하나의 입력을 가질 수 있다. 폼 필링 시스템 주도 대화에서의 필드란, 사용자로부터 해당 인식 문법에 유효한 하나의 정보를 입력받는 단위이다. [그림 4-3]는 4개의 필드로 구성된 대화이다.

시스템 : 승차 월일을 말씀해 주십시오. 사용자 : 12월 2일	승차월일 필드(Field)
시스템 : 출발역을 말씀해 주십시오. 사용자 : 서울	출발역 필드(Field)
시스템 : 도착역을 말씀해 주십시오. 사용자 : 부산	도착역 필드(Field)
시스템 : 승차 시각을 말씀해 주십시오. 사용자 : 오전 10시	승차시각 필드(Field)
[그림 4-3] 폼 필링 시스템 주도 대화	

폼 필링 시스템 주도 대화는 대화 정보 영역과 필드 영역으로 구성된다.

3.1.1 폼 필링 시스템 주도 대화의 대화 정보 영역

폼 필링 시스템 주도 대화의 대화 정보 영역은 대화 정보를 담고 있으며, 구성 요소는 다음과 같다([표 4-7]).

구성 요소	값
dialog_id	
field_count	*string*
field_list	*integer*
variable_list	*field_name list*
next_action	*variable list* (후속 사용자에게 질의)
event_handling	(이벤트 처리 객체 참조)

표 [4-7] 폼 필링 시스템 주도 대화의 정보영역의 구성 요소

폼 필링 시스템 주도 대화의 필드 영역은 3.1.2절에서 설명하겠다.

▪ **dialog_id**

: **dialog_id**는 대화 식별자로 문자열을 그 값으로 갖는다.

(예. **dialog_id** = *advanced_ticket*)

▪ **field_count**

: **field_count**는 대화에서 사용자에게 요구하는 정보항목(field)의 개수이다. 이 개수만큼 필드가 필요하다.

(예. [그림 4-3]의 대화는 4개(승차월일, 출발역, 도착역, 승차시각)의 필드를 사용자에게 요구한다. **field_count**)

= 4

▪ **field_list**

: **field_list**는 사용자로부터 입력받을 순서에 따라 열거한 필드 이름(**field_name**)의 리스트를 갖는다.

(예. '**field_list** = 승차월일, 목적지, 도착지, 승차시각' 이라면, 이 대화는 승차월일, 목적지, 도착지, 승차 시각의 필드를 가지며, 필드의 입력 순서가 승차월일, 목적지, 도착지, 승차 시각임을 뜻한다.)

▪ **variable_list**

: **variable_list**는 사용자에게 입력받은 필드의 값을 저장하는 변수이다. **variable_list**의 개수는 **field_count**와 동일하며, 순서 역시 **field_list**의 순서와 동일하다.

variable_list는 현재의 대화가 모든 필드를 수행하여 종료하면, 후위 수행 객체의 **submit**을 이용하여 다른 대화나 후위 정보 시스템에 전달하여 처리될 수 있다.

(예. 대화가 승차월일, 출발역, 도착역, 승차시각 필드로 구성되어 있고, 각 필드에 대응되는 변수를 *depart_date, depart_station, arrival_station, depart_time*으로 정의 할 수 있다. 승차월일 필드에서는 12월 2일을, 출발역 필드에서는 서울을, 도착역 필드에서는 부산을, 승차시각에서는 오전 10시라고 인식되었다면, 해당 변수와 그 값은 [표 4-8]과 같다.)

(*variable name*) **variable_list**	**variable_list** **(*value*)**
depart_date	*1202*
depart_station	*서울*
arrival_station	*부산*
depart_time	*1000*

[표4-8] **variable_list**의 속성과 속성 값의 예

▪ **next_action**

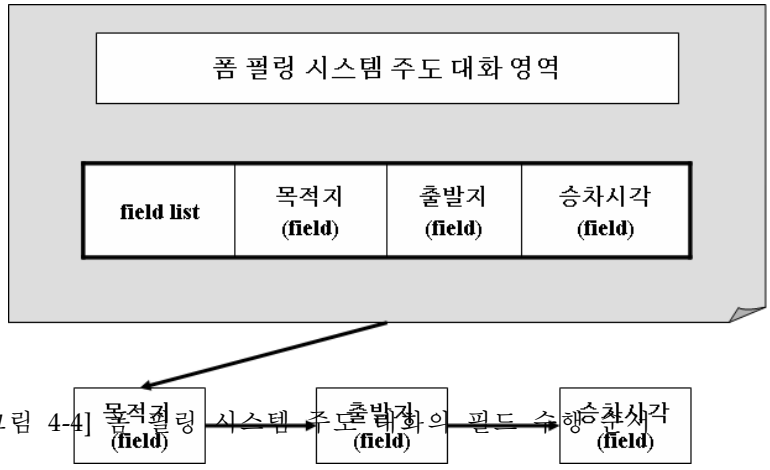
: 대화 영역에서의 후위 수행 객체는 대화의 모든 필드가 입력 된 후의 다음 동작을 명시한다. (후위 수행 객체 참고)

▪ **event_handling**

: 대화 영역에서 정의되는 이벤트 처리 객체는, 필드에 이벤트 처리 객체가 명시 되어 있지 않을 경우에 기본 이벤트 처리 객체로 사용된다. 만약 필드에 이벤트 처리 객체가 명시 되어 있다면, 필드의 이벤트 처리 객체가 우선한다.

3.1.2 폼 필링 시스템 주도 대화의 필드 영역

폼 필링 시스템 주도 대화는 대화의 필드 수행 순서가 대화영역의 **field_list**에 명시된 순서대로 대화가 진행된다([그림4-4]).



[그림 4-4] 폼 필링 시스템 주도 대화의 필드 수행 순서

폼 필링 시스템 주도 대화의 필드 영역은 필드 정보를 담고 있으며, 구성 요소는 다음과 같다([표 4-9]).

구성 요소	값
field_name	*string*
prompt	(프롬프트 객체 참조)
grammar_ref	(문법 객체 참조)
subdialog_call	(하위 대화 객체 참조)
event_handling	(이벤트 처리 객체 참조)

[표 4-9] 폼 필링 시스템 주도 대화의 필드 영역

▪ **field_name**

: **field_name**은 필드를 식별할 수 있는 이름을 문자열로 받는다. 같은 대화에서 같은 **field_name**을 사용할 수 없다. **field_name**은 대화 정보 영역 **field_list**에 명시한 이름과 동일해야 한다. **field_name**을 지정할 때에는 필드의 특징을 나타낼 수 있는 의미 있는 문자열을 사용한다.

▪ **prompt**

: 현재의 필드에서 사용자에게 요구하는 정보를 프롬프트 한다. 프롬프트 할 때에는 현재 필드 정보가 포함되도록 프롬프트 내용을 작성한다.

▪ **grammar_ref**

: 현재의 필드에 적용되는 문법을 명시한다.

▪ **subdialog_call**

: 현재의 필드가 다른 하위 대화를 호출 할 경우 하위 대화 호출 객체를 사용한다. (예를 들어, 현재의 필드가 우편번호를 받는 필드인 경우, 이 필드는 우편번호를 검색하는 다른 하위 대화를 호출할 수 있다.)

▪ **event_handling**

: 필드 영역에서의 이벤트 처리 객체는, 현재 필드에서 발생하는 이벤트를 처리하기 위해 정의한다. 만약 필드에 이벤트 처리 객체가 명시되어 있다면, 대화 정보영역에서의 이벤트 처리 객체보다 필드의 이벤트 처리 객체가 우선한다. 필드의 이벤트 처리 객체 프롬프트 내용에는 현재 필드와 관련된 내용이 포함되어 있어야 한다. (예를 들어, 현재

필드가 목적지를 입력받는 필드라면, noinput 이벤트를 처리하기 위한 프롬프트는 목적지에 관한 정보가 입력되지 않았음을 나타내는 내용을 사용해야 할 것이고, nomatch 이벤트를 처리하기 위한 프롬프트는 목적지에 관한 정보가 잘못 되었다는 내용이 포함되어야 한다. 현재의 필드에서 help 이벤트를 처리하기 위해서는 현재 필드에서 받아야 할 정보에 관한 도움말이 있는 대화를 호출해야 한다.)

3.2 폼 필링 상호 주도 대화

폼 필링 상호 주도 대화는 폼 필링 시스템 주도 대화와 달리, 채워야 하는 정보 항목 입력의 순서를 가지지 않고, 사용자가 한 번에 발화하여 입력한 값이 복수개가 될 수도 있다. 폼 필링 상호 주도 대화는 [그림 4-5]와 [그림 4-6]에서와 같이 초기 프롬프트가 존재하여 대화가 시작될 때 실행되며 사용자의 응답에 따라 대화의 흐름이 다양하게 전개된다. 폼 필링 상호 주도 대화에서 사용자는 초기 프롬프트에 대한 응답으로 대화가 요구하는 정보 항목들을 모두 입력 하거나, 몇 개만 입력 할 수 있다. 전자의 경우는 대화가 요구하는 정보 항목들이 모두 입력되기 때문에 대화를 종료하거나, 다른 대화나 후위 시스템으로 이동한다. 그러나 후자의 경우는 나머지 필요한 정보 항목들을 사용자에게 요구하며, 그 순서는 각 필드에서 사용자의 입력이 무엇인가에 따라 결정된다.

시스템 : 피자 주문 서비스입니다.
피자 종류, 크기, 수량을 말씀해 주십시오 *초기 프롬프트*

사용자 : 하와이언, 라지	*사용자 응답*
시스템 : 피자 수량을 말씀해 주십시오 사용자 : 1개	*수량 필드*

[그림 4-5] 폼 필링 상호 주도 대화의 예 1

시스템 : 피자 주문 서비스입니다. 피자 종류, 크기, 수량을 말씀해 주십시오	*초기 프롬프트*
사용자 : 라지	*사용자 응답*
시스템 : 피자 종류와 수량을 말씀해 주십시오	*종류_수량 필드*
사용자 : 하와이언 시스템 : 피자 수량을 말씀해 주십시오.	*수량 필드*
사용자 : 1개	

[그림 4-6] 폼 필링 상호 주도 대화의 예 2

3.2.1 폼 필링 상호 주도 대화의 정보 영역

폼 필링 상호 주도 대화의 정보 영역은 대화 정보를 담고 있으며, 구성 요소는 다음과 같다([표 4-10]).

구성 요소	값
dialog_id	*string*
initial_prompt	(프롬프트 객체 참고)
grammar_ref	(문법 객체 참고)
field_count	*integer*
field_items	*field_items*
variable_items	*variable_items*
not_yet_filled_patterns	*not_yet_filled_fields*
not_yet_filled_fields	*not_yet_filled_fields*
dialog_control	
next_action	(후위 작업 객체 참고)
event_handling	(이벤트 처리 객체 참고)

[표 4-10] 폼 필링 상호 주도 대화의 대화 영역 구성요소

폼 필링 상호 주도 대화의 필드 영역은 3.2.2절에서 설명하겠다.

- **dialog_id**

  : **dialog_id**는 대화 식별자로 문자열을 값으로 가진다.

  (예. **dialog_id** = *pizza_order*)

- **initial prompt**

  : **initial prompt**는 사용자와의 대화에서 제일 처음으로 프롬프트를 하

는 것으로 프롬프트는 내용에 있어서 현재의 대화가 사용자로부터 요구하는 정보항목을 모두 포함할 수 있도록 해야 한다.

▪ **grammar_ref**

: 대화 정보 영역에서 참조 받는 문법 객체는 초기 프롬프트에 대한 다양한 사용자 응답의 경우를 모두 다룰 수 있는 문법 객체이다.

▪ **field_items**

: **field_items**는 현재의 대화가 사용자에게 요구하는 정보 항목들이다. (예. 만약 현재의 대화가 사용자에게 크기, 종류, 수량을 요구한다면, **field_items**는 {종류, 크기, 수량}이다.)

▪ **field_count**

: **field_count**는 대화에서 사용자에게 요구하는 정보 항목의 개수를 가진다.

▪ **not_yet_filled_patterns**

: **not_yet_filled_patterns**는 대화에서 요구하는 정보 항목을 사용자가 다양하게 입력할 수 있는 모든 경우를 정의 한 것이다. 만약, **field_items**가 {종류, 크기, 수량}이라고 한다면, 사용자는 {종류, 크기, 수량} 정보를 다음과 같이 다양한 패턴으로 입력할 수 있다.

- 종류만 입력할 경우
- 크기만 입력할 경우
- 수량만 입력할 경우
- 종류와 크기를 함께 입력할 경우
- 종류와 수량을 함께 입력할 경우

- 크기와 수량을 함께 입력할 경우
- 종류, 크기, 수량을 모두 입력했을 경우

not_yet_filled_patterns는 이러한 사용자의 다양한 입력 패턴들을 정의해 놓은 것이다.

▪ **not_yet_filled_fields**

: **not_yet_filled_fields**는 현재의 대화에서, 사용자로부터 정보 항목을 입력받을 수 있는 모든 경우(**not_yet_filled_field**)를 정의한다. 이것은 **field_items**로부터 구할 수 있으며, **not_yet_filled_patterns**의 경우의 수와 같다. 예를 들어, 현재 대화의 **field_items**가 {종류, 크기, 수량}이라고 한다면, 대화에서 이 정보를 입력 받기 위한 경우는 다음과 같이 다양한 패턴으로 입력 받을 수 있다.

- 종류만 입력받을 경우
- 크기만 입력받을 경우
- 수량만 입력받을 경우
- 종류와 크기를 함께 입력받을 경우
- 종류와 수량을 함께 입력받을 경우
- 크기와 수량을 함께 입력받을 경우
- 종류, 크기, 수량을 모두 입력받을 경우

위의 경우에서 종류, 크기, 수량을 모두 입력받을 경우는 대화의 초기 프롬프트를 통해 사용자에게 입력받는 경우이기 때문에 대화가 시작되자마자 수행된다.

▪ **dialog_control**

: 폼 필링 상호 주도 대화는 대화의 흐름이 미리 순서가 지정되어 있는 것이 아니라, 사용자의 입력에 따라 대화 흐름이 달라진다. 그렇기 때

문에 대화 영역에서는 시스템 내부적으로 대화 흐름을 제어 할 수 있는 메커니즘을 이용하여, 사용자의 음성 입력에 따른 미입력 필드들(**not_yet_filled_fields**)을 수행시킬 수 있도록 해야 한다. **dialog_control**은 다음 수행할 미입력 필드(**not_yet_filled_field**)를 결정한다.

dialog_control은 다음과 같은 동작으로 이루어진다.

사용자가 지금까지 입력한 정보 항목을 제외하고, 아직 입력되지 않은 정보 항목을 요구하는 미입력 필드(**not_yet_filled_field**)를 **not_yet_filled_fields**에서 찾아 해당 미입력 필드(**not_yet_filled_field)**를 수행시키도록 한다.

예를 들면, **field_items**가 {종류, 크기, 수량}이라고 하고, 사용자가 지금까지 입력한 정보 항목이 {종류}일 때, 이때 수행되는 미입력 필드는 전체 항목{종류, 크기, 수량}에서 입력된 항목{종류}을 제외한 나머지{종류, 크기}를 요구하는 미입력 필드가 수행된다. [표 4-11]은 **field_items**가 {종류, 크기, 수량}인 경우 사용자가 지금까지 입력한 정보 항목에 따른 **dialog_control** 동작을 보여주고 있다.

not_yet_filled_patterns	**field_items**			**not_yet_filled_fields**
	종류	**크기**	**수량**	
{종류}	입력	미입력	미입력	{크기, 수량}
{크기}	미입력	입력	미입력	{종류, 수량}
{수량}	미입력	미입력	입력	{종류, 크기}
{종류, 크기}	입력	입력	미입력	{수량}
{종류, 수량}	입력	미입력	입력	{크기}
{크기, 수량}	미입력	입력	입력	{종류}
{종류, 크기, 수량}	입력	입력	입력	*대화 종료*

[표 4-11] dialog control의 예

▪ **variable_items**

: **variable_items**는 사용자에게 입력받은 필드의 값을 저장하는 변수다.

variable_items는 현재의 대화가 모든 필드를 입력받아 종료하면, 후위 수행 객체의 **submit**타입을 이용하여 다른 대화나 후위 정보 시스템에 전달하여 처리 될 수 있다.

▪ **next_action**

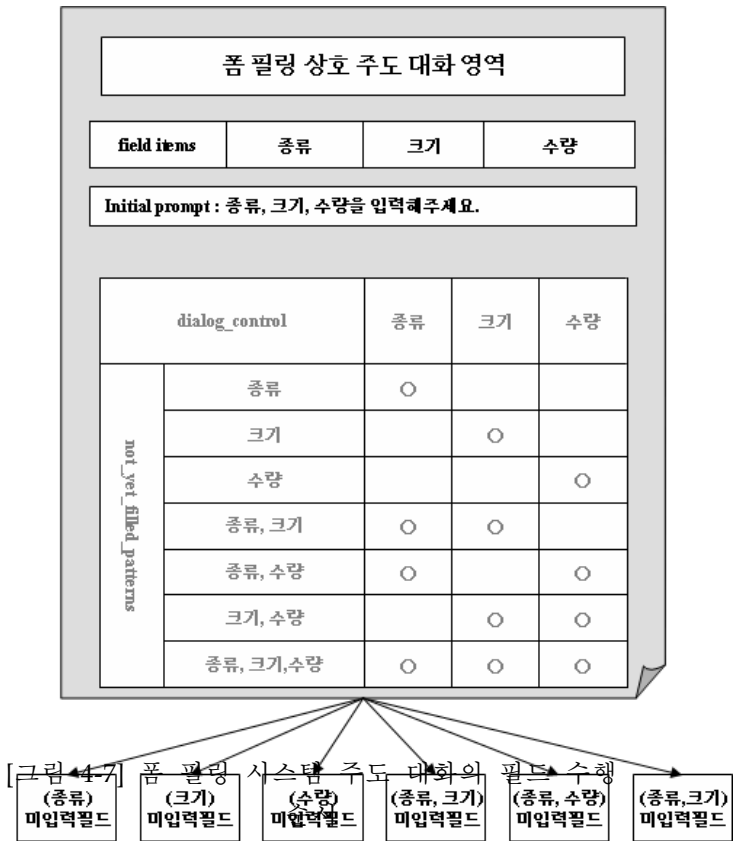
: 대화 영역에서의 후위 수행 객체는 대화의 모든 필드가 입력 된 후의 다음 동작을 명시한다. (후위 수행 객체 참고)

▪ **event_handling**

: 대화 영역에서 정의되는 이벤트 처리 객체는, 필드에 이벤트 처리 객체가 명시 되어 있지 않을 경우에 기본 이벤트 처리 객체로 사용된다. 만약 필드에 이벤트 처리 객체가 명시 되어 있다면, 필드의 이벤트 처리 객체가 우선한다.

3.2.2 폼 필링 상호 주도 대화의 미입력 (not_yet_filled_field) 필드 영역

폼 필링 상호 주도 대화의 미입력 필드(**not_yet_filled_field**)는 다이얼로그 영역의 **dialog_control**에 의해 수행된다([그림 4-7]).



[그림 4-7] 폼 필링 시스템 주도 대화의 필드 수행

폼 필링 상호 주도 대화의 미입력 필드(**not_yet_filled_field**) 영역의 구성 요소는 다음과 같다([표 4-12]).

구성 요소	값
not_yet_filled_field_name	*string*
prompt	(프롬프트 객체 참조)
grammar_ref	(문법 객체 참조)
sub_dialog_call	(하위 대화 호출 객체 참조)
event_handling	(이벤트 처리 객체참조)

[표 4-12] 폼 필링 상호 주도 대화의 필드 영역의 구성 요소

▪ **not_yet_filled_field_name**

: **not_yet_filled_field_name**은 필드를 식별할 수 있는 이름으로, 같은 대화에서 같은 **not_yet_filled_field_name**을 사용할 수 없다.

▪ **prompt**

: 현재의 미입력 필드에서 사용자에게 요구하는 정보를 프롬프트 한다. 프롬프트 할 때에는 현재 미입력 필드 정보가 포함되도록 프롬프트 내용을 작성 하도록 한다.

▪ **grammar_ref**

: 현재의 필드가 참조 받는 문법을 명시한다.

3.3 메뉴 선택(Menu-Selection) 대화

메뉴 선택 형식의 대화는 [그림 4-8]와 같이 하나의 대화가 여러 개의 정보 항목(menu)중에 한 가지를 사용자에게 선택하도록 요청한다. 사용자에 의해 선택된 정보 항목에 따라서 각각 다음 수행될 동작이 다르다.

메뉴 선택 형식의 대화는 사용자로부터 하나의 정보 항목 입력을 받기 때문에 따로 필드가 필요하지 않고 대화 정보영역만을 정의하면 된다.

시스템 : 안녕하세요. 날씨, 증권, 스포츠 정보 중
원하는 서비스를 말씀해 주세요.
사용자 : 날씨.

<날씨정보 대화로 이동....>

[그림 4-8] 메뉴 선택 대화의 예

3.3.1 메뉴 선택 대화의 대화 정보 영역

메뉴 선택 대화의 대화 정보 영역은 다음과 같이 구성된다([표 4-13]).

구성 요소	값
dialog_id	*string*
grammar_ref	(문법 객체 참조)
prompt	(프롬프트 객체 참조)
menu_items	*menu_items*
menu_control_table	(menu control table에서 별도 설명)
event	(이벤트 처리 객체 참조)

[표 4-13] 메뉴 선택 대화의 정보 영역 구성 요소

▪ **dialog_id**

: **dialog_id**는 대화 식별자로 문자열을 값으로 가진다.

(예. **dialog_id** = *main_menu*)

▪ **grammar_ref**

: 현재의 대화가 참조 받는 문법을 명시한다.

▪ **prompt**

: 메뉴 선택 대화에서의 **prompt**는 프롬프트 내용에 현재의 대화가 사용자에게 제공할 수 있는 메뉴를 모두 명시해야한다.

▪ **menu_items**

: **menu_items**는 현재 메뉴 선택 대화에서 사용자가 선택할 수 있는 정

보 항목들을 나타낸다.

(예. 현재의 메뉴 선택 대화에서 사용자가 선택할 수 있는 정보 항목이 '증권', '날씨', '스포츠' 라면, **menu_items**는 {증권, 날씨, 스포츠}이다.)

▪ **menu_control_table**

: **menu_control_table**은 **menu_items**( )의 각 정보항목( )이 선택되었을 때, 하위대화, 호출, 객체를 이용하여 다른 하위 대화를 호출을 하거나, 후속 대화 객체를 이용하여 다음 수행 할 동작을 정의한다([표 4-14]).

$menu\_item_1$ $menu\_item_m$ $menu\_item_1$ $menu\_item_m$

▫ **menu_control_table**은 [표 4-14]과 같이 나타낸다.

menu_items	**다음 수행 동작**
	next_action
$menu\_item_1$	**subdialog_call**
	…
. . .	**next_action**
$menu\_item_m$	**subdialog_call**

[표 4-14] **menu_control_table**

(예. menu_items가 {증권, 기상, 스포츠}일 경우, '증권'이 선택 되었을 경우에는 증권 정보 하위 대화를, '기상'이 선택 되었을 경우에는 기상 정보 하위 대화를, '스포츠'가 선택되었을 경우에는 스포츠 정보 하위 대화를 각각 호출한다고 한다면, **menu_control_table**은 다음과 같다.)

menu_items	**다음 수행 동작**
증권	subdialog_call(= 증권 정보) src
기상	subdialog_call(= 기상 정보) src
스포츠	subdialog_call(= 스포츠 정보) src

V. 결론 및 향후과제

인간의 가장 자연스러운 의사소통 수단인 음성을 사용하는 음성 사용자 인터페이스는, 최근 모바일 장치 등이 급속히 보급됨에 따라 그 활용이 증대 되고 있다.

본 논문에서는 음성 사용자 인터페이스(VUI) 개발을 위한 설계 방법론을 소개 하였다. VUI 설계 방법론은 요구 사항 분석, 고급 설계, 상세 설계, 시스템 개발, 테스트, 조율 및 조정으로 크게 여섯 단계로 구성된다. 특히, 상세 설계 단계에서는 한국어 음성 특성을 반영한 설계 기법을 제안하였다.

보다 자연스러운 음성 인터페이스의 설계는 인간의 대화 패턴을 모델링하여 음성 인터페이스 시스템 개발에 적용할 수 있어야 한다. 그러기 위해서는 대화 설계 영역에 대한 연구와 그 설계가 중요하다. 대화 설계는 대화 패턴에 따른 모델링을 통하여 각 패턴별로 대화를 설계하였다. 폼 필링 시스템 주도 대화 방식은 음성 인터페이스 사용에 경험이 없는 사용자층을 대상으로 할 때 효과적이고, 폼 필링 상호 주도 대화 방식은 시스템에 경험이 있는 사용자층을 대상으로 할 때 효과적이다. 두 대화 방식을 혼용하여 사용하는 것도 모든 사용자층을 아우를 수 있을 것이라 본다.

무엇보다도 음성 인터페이스 설계를 위해서 설계자는 다양한 관점에서 설계에 임해야 하며, 설계자 자신이 최악의 사용자임을 잊지 말고 설계해야 할 것이다.

본 논문에서는 음성 인터페이스 설계의 대화 설계에 있어서 대화 패턴별로 각각의 템플릿을 분류하였으나 보다 공용성 있고 추후 개발의 용이성을 위해서는 모든 대화 패턴을 적용시킬 수 있는 공용의 템플릿 모델링이 필요하다.

참고 문헌

[1] D. Redmond-Pyle and A. Moore, "Graphical User Interface Design and Evaluation Guide," Prentice Hall, 1995.

[2] M. H. Cohen, J. P. Giangolam, J. Balogh, "Voice User Interface Design," Addison-Wesley, 2004.

[3] M. F. McTear, "Spoken dialogue technology," ACM Computing Surveys (CSUR), Vol.34, Issue.1, pp.90~169, 2002.

[4] C. Sharma, J. Kunins, "VoiceXML: Strategies and Techniques for Effective Voice Application Development with VoiceXML 2.0," Willey Computer Publishing, 2002.

[5] J. A. Larson, "Introduction and Overview of W3C Speech Interface Framework," http://www.w3.org/TR/voice-intro/, 2000

[5] 철도청 ARS 예약시스템, 060-700-1188 (전화).

[6] N. Yankelovich, G.-A. Levow, M. Marx, “Designing SpeechActs: Issues in Speech User Interfaces,” Conference on Human Factors in Computing Systems, CHI ’95 Proceedings, pp369~376, 1995.

[7] D.Schiffrin, "Approaches to discourse," Cambridge University Press, 1998.

[8] 김태엽, "담화 표지 되기와 문법화," 우리말글, 26집, pp61~80, 2002.

[9] M. Oshry, R. Auburn, P. Baggia, M. Bodell, D. Burke, D. C. Burnett, E. Candell, H. Kilic, S. McGlashan, A. Lee, B. Porter, K. Rehor, "Voice Extensible Markup Language (VoiceXML) 2.1(W3C Candidate Recommendation 13 June 2005),"

http://www.w3.org/TR/2005/CR-voicexml21-20050613/, 2005.

[10] Nuance Communications Inc, "Nuance Speech Recognition System Version 8.5: Grammar Developer's Guide," 2003.

[11] Andrew Hunt, Scott McGlashan, "Speech Recognition Grammar Specification Version 1.0(W3C Recommendation 16 March 2004)," http://www.w3.org/TR/2004/REC-speech-grammar-20040316/, 2004

[12] Bevocal Inc, "Grammar Reference," 2005

[13] 전영옥, "전통화법과 화법교육: 한국어 담화 표지의 특징 연구," 화법연구, pp.113~145, 2002.

[14] 김미경, "한국어 대화체 구문의 정보구조," 담화와 인지, 제 8권 1호, pp.21~42, 2001.

[15] 김민성, 정성윤, 손종목, 배건성, 김상훈, "한국어 연속 숫자음 전화 음성 인식에서의 오인식 유형 분석," 말소리, 제 46호, pp.77~86, 2003.

[16] 이주행, "한국어 청자 경어법의 교육 방안에 관한 고찰," 국어교육 119, pp.371~396, 2006.

[17] B. Friederike. "Terms of Address. Problems of Patterns and Usage in Various Languages and Cultures," Berlin: Mouton de Gruyter, 1988.

[18] 박정운, "한국어 호칭어 체계," 사회언어학, 제5권 2호, pp.507~527, 1997.

[19] 신선경, "웹 상에 나타난 공무원과 민원인의 질의응답 실태 연구," 화법연구, pp.113~132, 2003.

[20] 정명숙, "한국어 발음 교육을 위한 음성DB 구축 방안," 말소리, 제 47호. pp.51~72, 2003.

[21] 이호영, "한국어의 억양 교육," 제 8회 국외 한국어 교사 연수회, 2004

[22] 이호영, "한국어 문장 억양의 선택 과정," 한글, 225, pp.7~33 1995.

[23] M. K. Brown, A. Kellner, D. Raggett, "Stochastic Language Models (N-Gram) Specification(W3C Working Draft 3 January 2001),"

http://www.w3.org/TR/ngram-spec/, 2001.

[24] D. A. Dahl, "Natural Language Semantics Markup Language for the Speech Interface Framework(W3C Working Draft 20 November 2000)," 2000.

[25] D. C. Burnett, M. R. Walker, A. Hunt, "Speech Synthesis Markup Language (SSML) Version 1.0(W3C Recommendation 7 September 2004)," 2004.

ABSTRACT

A Study on Korean Voice User Interface Design Methodology

Kwon Ji-Hye
Department of Computer Science Education
Graduated School of SungShin. Women's University

Voice is the most natural way in human communication. Demand for VUI(Voice User Interface) has been increasing in many systems such as automobile navigation system, voice browser for blinds and ARS(Audio Response System) for advanced ticket purchasing system.

In this paper, we introduce the VUI design methodology for English. For some parts of the VUI design methodology such as prompt design, intonation planning and recognition grammar development, we propose the detailed strategies in order to reflect the characteristics of Korean spoken language.

According to the proposed design strategies, we also present a formal model for dialog design. The dialog is the basic unit in VUI. We classify dialogs into tree types : Form Filling Directed dialog, Form Filling Mixed Initiative Dialog and Menu Selection Dialog. For each type of dialogs, we develop a formal model that can be used in dialog design.