WESC-14-09-개발완료보고서

한 석 봉

참가부문 : 오픈 플랫폼

팀 명 : kobot_op

팀구성원

No.	구분	성명	소속명	No.	구분	성명	소속명
1	**팀장**	**박상열**	**국민대학교 컴퓨터공학부**	6	팀원		
2	팀원	김호연지기	국민대학교 컴퓨터공학부	7	팀원		
3	팀원	박혜련	국민대학교 컴퓨터공학부	8	팀원		
4	팀원	신혜수	국민대학교 컴퓨터공학부	9	팀원		
5	팀원			10	팀원		

2014. 10. 5

1. 목차

4.2.2.4. void onEndOfSpeech( )

4.2.2.5. void onResults(Bundle results)

4.2.2.6. void onBufferReceived(byte[] buffer)

4.2.2.7. void onError(int error)

4.2.2.8. void run( ) //SendManager

4.2.2.9. void run( ) //RecvManager

4.2.3. Message.java

4.2.3.1. void setMode(int val)

4.2.3.2. void setTime(Date date)

4.2.3.3. void buildMessage(Boolean isGen, String text)

4.2.3.4. byte[] getMessage( )

4.2.4. hsb_analyze.c

4.2.4.1. 회의록 분석을 위한 함수

4.2.4.1.1. void analyze( )

4.2.4.1.2. void findKeyword(node* m, char* keyword)

4.2.4.1.3. void printKeyword(char* keyword)

4.2.4.1.4. void analyzeParticipation(node* u, int total)

4.2.4.1.5. void analyzeKeywordHit(node* u, char* keyword)

4.2.4.1.6. void printKeywordHit(node* u, char* keyword)

4.2.4.1.7. void incKey(node* u, char* id)

4.2.4.1.8. void decKey(node* u, char* id)

4.2.4.1.9. void makeKeywordFile(int numKeyFile, node* morp)

4.2.4.1.10. void writeKeyword(FILE* keyFile, node* morp)

4.2.4.1.11. void report(node* morp, node* user, char* keyword, int total)

4.2.4.2. 트리 구조를 위한 함수

4.2.4.2.1. void insert(node* r, char* id)

4.2.4.2.2. void freeNode(node* r)

4.2.4.3. 스택 구조를 위한 함수

4.2.4.3.1. int isEmpty(stack* s)

4.2.4.3.2. void push(stack* s, char* data)

4.2.4.3.3. void pop(stack* s, char* data)

4.2.5. hsb_sc2html.c

4.2.5.1. void makeHtmlFile( )

2. 개요

2.1. 작품명

한석봉(난 말을 할 테니, 넌 받아 적거라)

2.2. 작품 개요

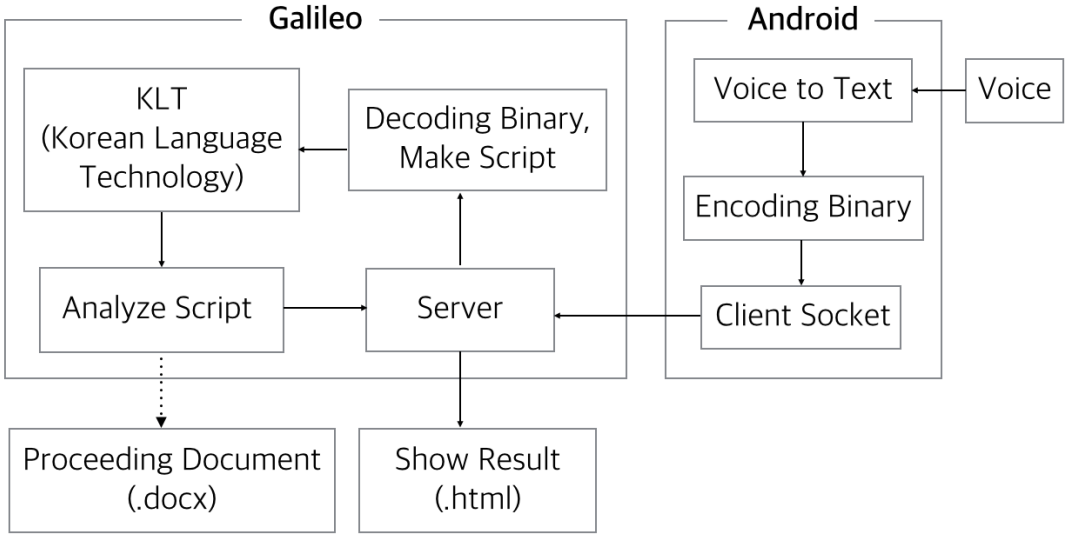
'한석봉'은 각 회의 참석자의 발언을 정리하여 정해진 양식의 회의록을 작성해주는 임베디드 장치이다. '한석봉'은 3단계의 과정을 거친다. 첫 번째 단계는 각 회의 참여자의 안드로이드 기기를 통해, 회의 참여자의 음성을 자동으로 인식하여 문자로 변환하는 과정이다. 두 번째 단계에서는 안드로이드 기기가 변환된 문자를 네트워크 통신을 통해 갈릴레오 보드 기반의 중앙 장치에 전달한다. 세 번째 단계에서는 갈릴레오 보드가 안드로이드 기기로부터 전송된 문자 데이터를 기반으로 회의록을 만들고, 형태소 분석을 이용하여 회의록을 분석한다.

2.3. 목적

'회의'는 의견을 수렴하는 보편적인 과정이다. 회의는 회의 참여자뿐만 아니라, 회의 내용을 알아야 할 많은 사람들을 위해서 기록되어야 할 필요가 있다. 이때 매번 서기가 회의 기록을 작성해야 하는 번거로움이 있을 수 있고, 작성 과정에서 서기의 주관적 관점이 들어갈 여지가 있을 수 있다는 문제가 발생한다. 또한 회의록을 통해 '회의' 상황 및 '회의 참여자'들에 대한 여러 가지 정보를 제공한다면, 참여자들이 효율적인 회의를 할 수 있게 된다. 이에 착안하여 회의록을 자동으로 작성해주는 장치를 고안하게 되었다.

3. 작품 설명

3.1. Software 구성



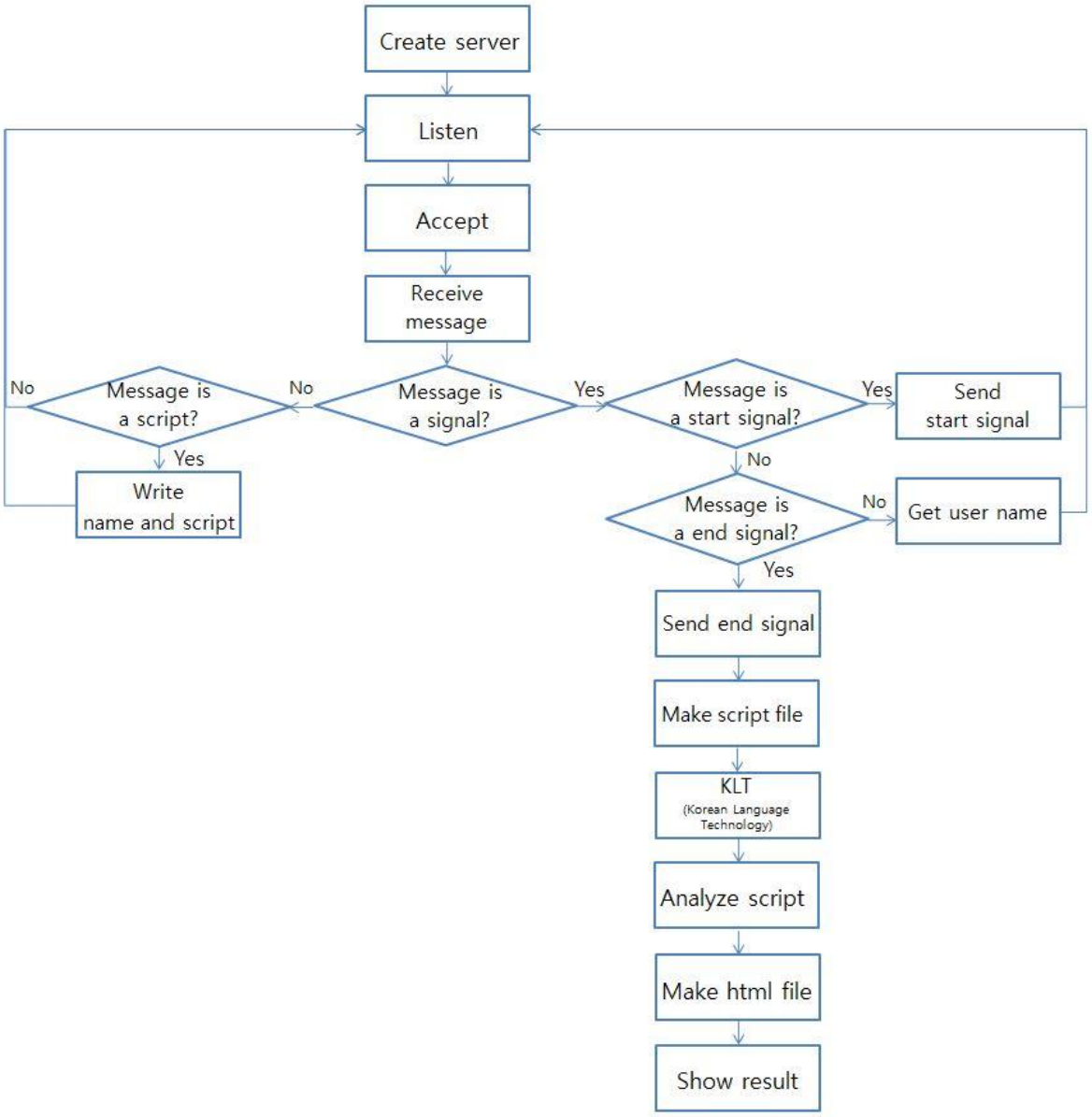
'한석봉'은 갈릴레오 보드와 안드로이드 기기의 상호작용으로 동작한다. 안드로이드 기기의 소프트웨어는 크게 세 가지 부분으로 구성되어 있다. 첫 번째는 음성을 문자로 변환하는

부분이다. 두 번째는 변환된 문자를 정해진 포맷(<6.1. 전송에 이용된 메시지 프로토콜> 참고)에 맞춰 바이트 스트림으로 인코딩하는 부분이다. 세 번째는 네트워크 통신을 위한 소켓 구현 부분이다.

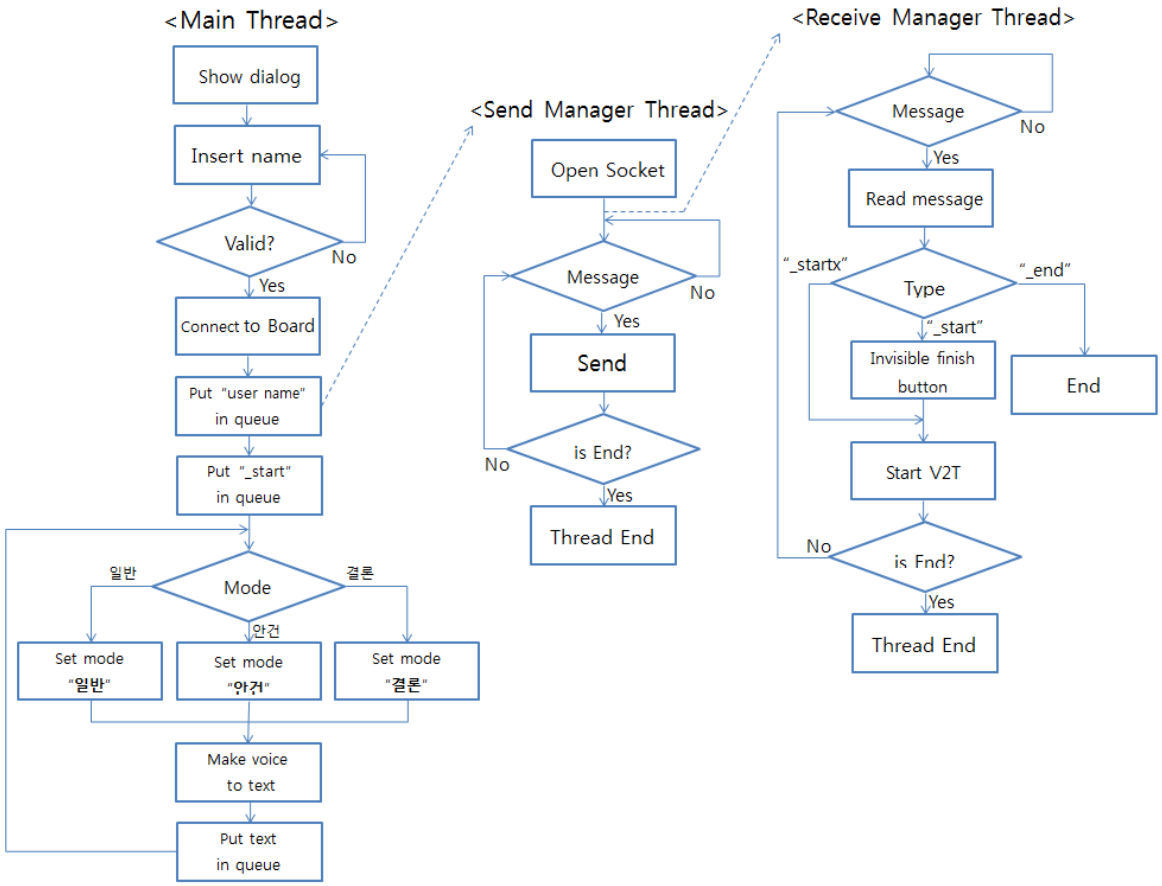
갈릴레오 보드 소프트웨어는 크게 네 가지 부분으로 구성되어 있다. 첫 번째는 안드로이드 기기에서 받아온 바이트 스트림을 디코딩하여 스크립트 파일을 만드는 부분이다. 두 번째는 스크립트에서 색인어를 추출하는 부분이다. 세 번째는 추출된 색인어를 분석하여 스크립트에 분석 정보를 더하는 부분이다. 네 번째는 네트워크 통신을 위한 서버 구현 부분이다.

3.2. Software 흐름도 및 클래스 다이어그램 (개발언어에 따라 선택)

3.2.1. Galileo Board Application



3.2.2. Android Mobile Application



3.3. Software 기능 (알고리즘 설명 포함)

3.3.1. 음성 인식 및 문자 변환 기능

음성 인식 및 문자 변환 기능은 안드로이드 API에서 제공하는 SpeechRecongizer 클래스와 RecognitionListener 인터페이스를 상속하여 구현된다. SpeechRecognizer 클래스는 안드로이드 기기의 마이크로부터 회의 참여자의 발언을 입력 받아 이를 문자열로 변환하는 역할을 수행한다. SpeechRecognizer 클래스가 반환하는 문자열을 인코딩하여 갈릴레오 보드로 보내기 위한 메시지 패킷을 만든다. 문자열을 인코딩하기 위해 Date 클래스를 이용하여 시간을 기록하며, 만들어진 메시지 패킷을 Queue에 보관했다가 네트워크 통신에 의해 갈릴레오 보드로 전송한다.

3.3.2. 안드로이드 기기와 갈릴레오 보드 통신 기능

회의 참여자들이 사용하는 안드로이드 기기들과 갈릴레오 보드는 클라이언트-서버 구조를 이루며, 소켓을 통해 서로 통신한다. 클라이언트 역할을 하는 안드로이드 기기의 소프트웨어의 서버에게 메시지를 보내는 부분과 서버로부터의 메시지를 받는 부분은 서로 다른 쓰레드에서 구현되어 있다. 즉, 안드로이드 기기의 소프트웨어는 멀티쓰레딩(Multi-threading)을 통해 서버로부터 메시지를 기다리는 동안 애플리케이션이 멈추지 않도록 했다. 안드로이드 기기가 서버에게 보내는 메시지는 회의 참여자의 음성이 문자로 변환된 데이터이며, 이 데이터는 서버인 갈릴레오 보드로 보내져 이후

모든 과정이 서버 사이드(Server-side)로 처리된다. 서버 역할을 하는 갈릴레오 보드의 소프트웨어는 Node.js로 구현되었으며, 클라이언트에서 연결을 요청할 때마다 콜백(callback) 함수를 호출하여 클라이언트로부터의 데이터를 기록한다.

3.3.3. 회의록 작성 기능

통신을 통해 회의 참여자가 발언한 메시지를 정해진 양식의 회의록으로 자동 작성한다. 기본 양식은 다음과 같다.

<안건> 오픈플랫폼 부문 참가를 위한 아이디어
혜련 : 매번 회의 때마다 누군가가 회의록을 써야 한다는 것이 불편한 것 같아.
지기 : 그냥 자동으로 되면 회의록 쓰는 사람도 부담이 없을 텐데.
혜수 : 그럼 자동으로 회의록 쓰는 장치를 만들자.
상열 : 그냥 회의록만 쓰면 뭔가 부족하지 않아?
<결론> 자동으로 회의록 작성을 하는 장치로 오픈플랫폼 부문을 참가하자.

회의록 작성 알고리즘은 다음과 같다. 회의 중 안드로이드 기기로부터 전송되는 패킷을 '/hsb/inputs/message.bin'라는 파일에 바이너리 형태로 출력하여 저장한다. 이때 패킷을 보내온 안드로이드 기기의 IP를 판단하여 회의 참여자의 ID를 '/hsb/inputs/id.txt'라는 파일에 출력한다. 한 줄에 하나의 ID가 쓰여진다. 회의가 끝나면 '/hsb/inputs.id.txt' 파일에서 메시지 발신자에 대한 정보를 추출하고, '/hsb/inputs/message.bin' 파일에서 메시지 정보를 추출하여(<6.1 전송에 이용된 메시지 프로토콜> 참고) 다음과 같은 XML 형식으로 '/hsb/inputs/input.xml'에 출력한다.

```
<Message>
  <ID>지기</ID>
  <Time>15:33:10</Time>
  <Text> 그냥 자동으로 되면 회의록 쓰는 사람도 부담이 없을 텐데.</Text>
</Message>
```

만들어진 '/hsb/inputs/input.xml' 파일에서 ID 정보를 추출하여 각 ID마다 XML 파일을 만들고, 해당 메시지를 '/hsb/outputs/id.xml' 파일에 출력한다. 각 ID 파일을 참조하여 시간 순서로 정렬한 뒤, User 구조체(<4.1.1.1.2. struct User> 참고)에 ID 정보와 메시지 정보, 출력될 메시지의 인덱스 정보를 저장한다. 각 User 구조체의 인덱스 정보를 참조하여 더 이른 시간 정보를 가진 User의 메시지를 스크립트에 출력한다. 이때 메시지의 내용이 "***"이 나올 때까지 출력한다. 이는 회의 중에 생기는 여러 문제 상황을 처리하기 위함이다(<6.2. 문제점과 해결 알고리즘> 참고).

3.3.4. 회의록 분석 기능

회의록 하단부에 회의록 분석 결과에 대한 정보를 다음과 같이 출력한다.

<Keyword>

본 회의록의 키워드는 '오픈플랫폼'(으)로 추정됩니다.

<Rate of Participation>

[[혜련]]의 회의 참여율은 25 % 입니다.

[[지기]]의 회의 참여율은 25 % 입니다.

[[혜수]]의 회의 참여율은 25 % 입니다.

[[상열]]의 회의 참여율은 25 % 입니다.

<Keyword Hit>

[[혜련]]은 키워드 '오픈플랫폼'(을)를 2 번 언급했습니다.

[[지기]]은 키워드 '오픈플랫폼'(을)를 3 번 언급했습니다.

[[혜수]]은 키워드 '오픈플랫폼'(을)를 3 번 언급했습니다

[[상열]]은 키워드 '오픈플랫폼'(을)를 2 번 언급했습니다

'한석봉'이 제공하는 회의록 분석 정보는 다음과 같다.

3.3.4.1. 키워드

'한석봉'은 현재 회의록에서 가장 많이 언급된 단어를 회의의 키워드로 추정하여 사용자들에게 제공한다. 키워드를 추정하기 위해, 회의록에 등장하는 단어들에 대하여 빈도수를 검사한다. 이때 의미상의 빈도수를 체크해야 하므로 형태소 분석을 할 필요가 있다. 예를 들면, '대학을'과 '대학에서'라는 어절에 대하여 '대학'이라는 형태소를 분리해서 키워드로 추정해야 한다. 스크립트의 각 어절로부터 형태소를 분리하기 위해서 '형태소 분석기(KLT: Korean Language Technology, 강승식 개발)'를 사용했다. KLT는 '대학을'과 '대학에서'라는 입력에 대하여 '대학'이라는 형태소를 추출한다. 이때 '이', '저'와 같은 지시어나 '나', '우리', '것'과 같은 대명사가 키워드로 추정될 가능성을 방지하기 위해, 접속사, 지시어, 대명사 등을 위한 사전을 미리 만들어두고, 이와 같은 단어들에 대하여는 키워드로 추정하는 가능성을 배제한다.

3.3.4.2. 회의 참여자들의 참여율

'한석봉'은 회의 중에 각 회의 참여자들이 얼마나 참여를 했는지에 대한 수치 정보를 제공한다. 회의 참여자들의 참여율(%)은 다음과 같은 공식으로 추정한다.

$$\text{회의 참여자 i의 참여율(\%)} = \frac{\text{회의 참여자 i의 발언 횟수}}{\sum_{i=1}^{n} \text{회의 참여자 i의 발언 횟수}} * 100$$

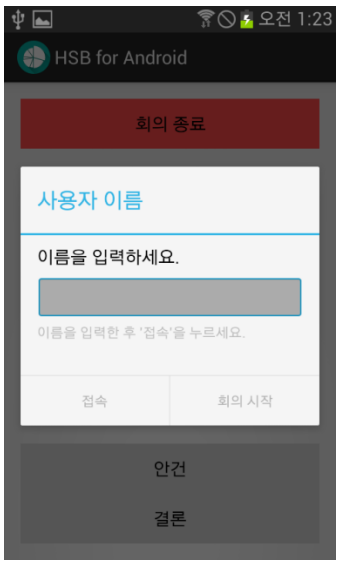
이때, n은 전체 회의 참여자의 숫자(명)이다. 즉, 회의 참여자의 참여율은 '특정 회의

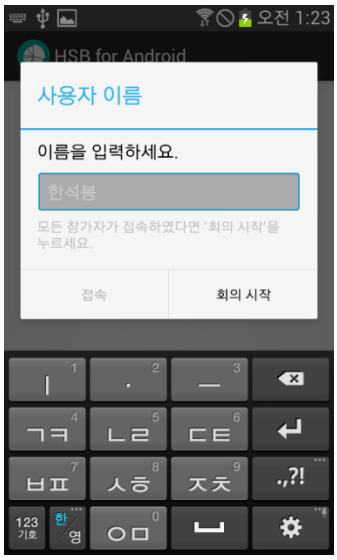
참여자의 발언 횟수'를 '전체 회의 참여자의 총 발언 횟수'로 나눈 값이다. 회의 참여자의 발언은 만들어진 스크립트로부터 "id : "인 부분을 인식함으로써 셀 수 있다. 또한 각 회의 참여자의 발언 횟수를 기록하기 위해 트리 형태로 회의 참여자에 대한 정보를 기록했다(<4.1.1.3.1. struct TreeNode> 참고).

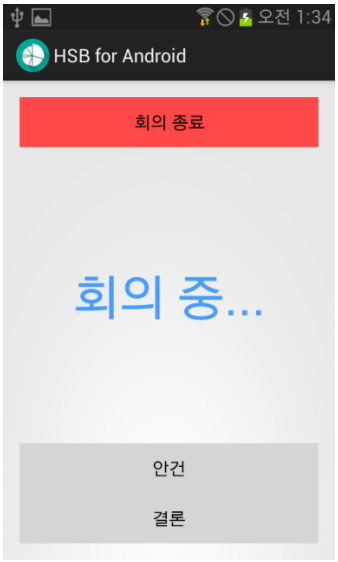
3.3.4.3. 회의 참여자들의 키워드 언급 횟수

'한석봉'은 회의 중에 각 회의 참여자들이 얼마나 회의에 관련된 발언을 했는지에 대한 정보를 제공한다. 이는 각 회의 참여자들의 발언에 키워드가 포함되었는지를 조사함으로써 추정할 수 있다.

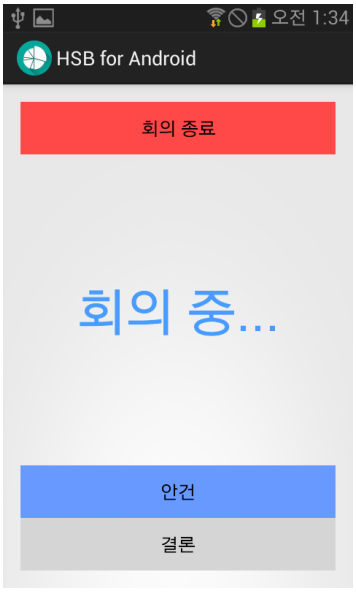
3.4. 프로그램 사용법 (Interface)

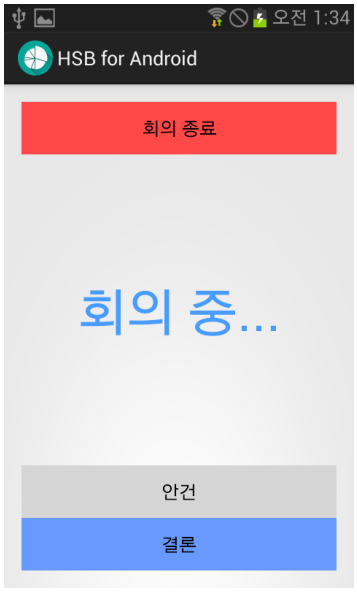






갈릴레오 보드가 서버를 키면, 안드로이드 기기가 접속이 가능한 상태가 된다. 안드로이드 기기가 '한석봉' 안드로이드 애플리케이션을 실행시키면 이름 입력 창과 함께 '접속' 버튼과 '회의 시작' 버튼이 나타난다. 이름을 입력해야 '접속' 버튼을 누를 수 있으며, '접속' 버튼을 눌러야 '회의 시작' 버튼을 누를 수 있다. 여러 명의 회의 참여자 중 한 명이 서버에 접속한 후 이름 입력 창 하단부에서 '회의 시작' 버튼을 누르면, 갈릴레오 보드가 회의가 시작됨을 인식하고, 회의 참여자들의 모든 안드로이드 기기에 회의가 시작되었음을 알린다. 갈릴레오 보드로부터 회의가 시작되었음을 전달받은 안드로이드 기기의 화면에는 '회의 중'이라는 문자가 가운데에 나타나며, 하단부에 '안건' 버튼과 '결론' 버튼이 나타난다. 단, '회의 시작' 버튼을 누른 회의 참여자의 화면에만 상단에 '회의 종료' 버튼이 나타난다.





회의 중 참여자들의 발언은 자동으로 녹음되어 갈릴레오 보드로 전송된다. 또한 회의 중 참여자들은 '안건' 또는 '결론' 버튼을 누를 수 있다. '안건' 버튼을 누르면 참여자의 다음 발언이 '안건'으로서 갈릴레오 보드에 전송되며, '결론' 버튼을 누르면 참여자의 다음 발언이 '결론'으로서 갈릴레오 보드에 전송된다. '회의 시작' 버튼을 누른 참여자가 '회의 종료' 버튼을 누르면, 갈릴레오 보드는 회의가 종료되었음을 인식하고, 모든 회의 참여자들의 안드로이드 기기에 회의가 종료되었음을 알린다. 갈릴레오 보드로부터 회의가 종료되었음을 전달 받은 안드로이드 기기에서는 '한석봉' 애플리케이션이 종료된다. 회의가 종료되면 갈릴레오 보드는 전송된 발언들을 처리하여 정해진 양식의 회의록을 작성하고, 웹 페이지에 게시하여 모바일 또는 PC의 웹 브라우저를 통해 회의록을 볼 수 있게 한다.

3.5. 개발환경 (언어, Tool, 사용시스템 등)

3.5.1 Android Application

언어	Java
IDE	Android Studio 0.8.9
테스트 단말기	SHV-E160S(Android 4.1.2 Jelly Bean)
	SHW-M250S(Android 4.1.2 Jelly Bean)

3.5.2. Galileo Board Application

언어	C, Javascript
Cross Compiler	i586-poky-linux-uclibc-gcc

- Target System : i586(Poky Linux)
- Host System : x86_64(Ubuntu 12.04 LTS)

Framework	Node.js

4. 프로그램 설명

4.1. 파일 구성

4.1.1. Galileo Board Application

4.1.1.1. hsb.c

4.1.1.1.1. struct Message

안드로이드 기기로부터 전송되는 메시지의 정보를 저장하기 위한 구조체이다. 멤버는 char[] time, char[] text이다. time은 메시지가 보내지는 시간 정보를 "hh:mm:ss"(h는 시간을 나타내는 한 자리 정수이며, m은 분을 나타내는 한 자리 정수이다. 그리고 s는 초를 나타내는 한 자리 정수이다.) 형식으로 저장한다. text는 안드로이드 기기에서 녹음되고 변환된 문자 정보를 저장한다.

4.1.1.1.2. struct User

각 메시지가 회의 참여자 중 누구의 메시지인지에 대한 정보를 저장하기 위한 구조체이다. 멤버는 int numMessage, int idx, char[] id, struct Message*[] messages이다. numMessage는 참여자가 몇 개의 메시지를 보냈는지에 대한 정보를 저장한다. idx는 메시지를 기반으로 스크립트를 만들 때, 현재 어느 메시지까지 출력되었나에 대한 정보를 저장한다. messages는 참여자가 보낸 메시지 정보를 저장한다.

4.1.1.1.3. Function Prototype

int compareTime(char* t1, char* t2)

void makeIdFiles( )

void mergeIdFiles( )

void makeLog( )

4.1.1.2. hsb.sh

'한석봉'을 실행시키기 위한 셸 스크립트 파일이다. 이 파일이 하는 역할은 다음과 같다.

- '한석봉' 프로그램을 실행시키기 위한 초기 작업(inputs 폴더와 outputs 폴더 비우기)
- 'server.js' 백그라운드에서 실행
- '/hsb/hsb.exe' 실행
- '/hsb/KLT/EXE/index' 실행
- '/hsb/hsb_analyze.exe' 실행
- '/hsb/hsb_sc2html.exe' 실행

4.1.1.3. hsb_analyze.c

4.1.1.3.1. struct TreeNode

회의 참여자 또는 형태소 정보를 저장하기 위한 트리 구조체이다. 멤버는 char[]

data, struct TreeNode* left, struct TreeNode* right, int frequency, int keyword이다. data는 현재 트리 노드에 저장될 데이터 값이다. 회의 참여자의 정보를 저장하기 위한 트리인 경우(소스 상 user 트리), 데이터 값은 각 회의 참여자의 ID를 나타낸다. 그러나 형태소 정보를 저장하기 위한 트리인 경우(소스 상 morp 트리), 데이터 값은 각 형태소 문자열 자체를 나타낸다. left와 right는 이진 트리(Binary Tree) 검색을 하기 위한 변수이며, 각각 left child와 right child를 가리키는 트리 노드 포인터이다. frequency는 현재 노드의 데이터 값이 몇 번 삽입되었는지를 나타내는 정수 값이다. 따라서 frequency의 값을 통해 회의 참여자의 발언 횟수나, 형태소의 등장 횟수를 확인할 수 있다. keyword는 회의 참여자의 정보를 저장하기 위한 트리인 경우에만 사용된다. keyword는 현재 트리 노드의 데이터 값을 ID로 가진 회의 참여자가 키워드를 언급한 횟수를 의미한다.

4.1.1.3.2. struct Stack

스크립트의 각 어절로부터 분리된 형태소들을 저장하기 위한 구조체이다. 멤버는 char[][] stk, int top이다. stk는 형태소를 나타내는 문자열의 배열이다. top은 현재 스택의 제일 마지막으로 넣은 데이터 값을 가리키는 인덱스 값이다.

4.1.1.3.3. Function Prototype

4.1.1.3.3.1. 회의록 분석을 위한 함수

void analyze( )

void findKeyword(node* m, int max, char* keyword)

void printKeyword(char* keyword)

void analyzeParticipation(node* u, int total)

void analyzeKeywordHit(node* u, char* keyword)

void printKeywordHit(node* u, char* keyword)

void incKey(node* u, char* id)

void decKey(node* u, char* id)

void makeKeywordFile(int numKeyFile, node* morp)

void writeKeyword(FILE* keyFile, node* morp)

void report(node* morp, node* user, char* keyword, int total)

4.1.1.3.3.2. 트리 구조를 위한 함수

void insert(node* r, char* id)

void freeNode(node* r)

4.1.1.3.3.3. 스택 구조를 위한 함수

int isEmpty(stack* s)

void push(stack* s, char* data)

void pop(stack* s, char* data)

4.1.1.4. hsb_sc2html.c

'한석봉'이 만들어 낸 회의록을 웹 페이지에 게시하기 위해서, 스크립트를 HTML 문서로 변환하는 일을 하는 파일이다.

4.1.1.4.1. Function Prototype

void makeHtmlFile( )

4.1.1.5. server.js

갈릴레오 보드에서 실행되는 서버 프로그램으로, 안드로이드 기기와 통신하고, http 서버로서 회의록에 대한 스크립트 HTML 파일을 웹페이지로 호스팅한다. Node.js 프레임워크를 기반으로 작성된 서버 사이드(Server-side) 프로그램이다.

4.1.1.5.1. Array user_name

서버에 접속하는 클라이언트인 안드로이드 기기의 IP 정보와 사용자의 이름을 함께 관리하기 위한 리스트 객체이다. user_name[0]을 통해 안드로이드 기기의 IP를 저장하며, user_name[1]을 통해 사용자의 이름을 저장한다.

4.1.1.5.2. Array sockets

서버에 접속하는 클라이언트인 안드로이드 기기와 통신하기 위한 소켓을 관리하기 위한 리스트 객체이다.

4.1.1.5.3. Function Prototype

function callback_server_connection(socket)

http.createServer(function (request, response) { })

4.1.1.6. KLT(Korean Language Technology)

형태소 분석을 하는 소프트웨어이다. 국민대학교 컴퓨터공학부 강승식 교수에 의해 개발되었다. KLT 프로그램은 한글 음절 정보를 기반으로 한 형태소 분석 알고리즘에 의해 동작한다. KLT 프로그램은 입력 파일에서 각 어절을 분리하고, 분리된 어절 별로 형태소를 분석하여 다음과 같은 형식으로 출력파일을 만들어주는 프로그램이다.

[[어절]]
    형태소1 형태소2 형태소3 …

4.1.2. Android Application

4.1.2.1. V2T.java

4.1.2.1.1. Message msg

Message.java에 정의된 Message클래스의 인스턴스를 저장한다.

4.1.2.1.2. Inner Class

4.1.2.1.2.1. class SendManager

음성 인식 결과 생성된 메시지를 갈릴레오 보드로 보내는 역할을 한다. Thread 클래스를 상속하여 별도의 쓰레드에서 실행된다. SendManager의 모든 작업은 '한석봉' 안드로이드 애플리케이션이 작동되는 동안 백그라운드 환경에서 이루어진다.

4.1.2.1.2.2. class RecvManager

갈릴레오 보드로부터 전송된 신호를 받아, 적절하게 처리하는 역할을 한다. Thread 클래스를 상속하여 별도의 쓰레드에서 실행된다. 갈릴레오 보드로부터 전송된 신호는 64바이트로 구성된 메시지이다. RecvManager의 모든 작업은 '한석봉' 안드로이드 애플리케이션이 작동되는 동안 백그라운드 환경에서 이루어진다.

4.1.2.1.3. Function Prototype

void onCreate(Bundle saveInstance State)

void onReadyForSpeech(Bundle params)

void onBeginningOfSpeech( )

void onEndOfSpeech( )

void onResults(Bundle results)

void onBufferReceived(byte[] buffer)

void onError(int error)

void run( ) //SendManager

void run( ) //RecvManager

4.1.2.2. Message.java

4.1.2.2.1. byte[] message

각 회의 참여자의 발언은 패킷(바이트 스트림)화 되어 갈릴레오 보드로 전송되는데, 이때 전송될 패킷이다.

4.1.2.2.2. Queue<byte[]> q

각 회의 참여자의 발언은 패킷(바이트 스트림)화 되어 갈릴레오 보드로 전송되는데, 이때 전송 대기 중인 패킷이 저장될 큐이다.

4.1.2.2.3. Function Prototype

void setMode(int mode)

void setTime(Date date)

void buildMessage(Boolean isGen, String text)

byte[] getMessage( )

4.2. 함수별 기능

4.2.1. hsb.c

4.2.1.1. int compareTime(char* t1, char* t2)

시간을 나타내는 “hh:mm:ss” 형식의 문자열 두 개를 인자로 받아서 비교하는 함수이다. 첫 번째 시간(t1)이 두 번째 시간(t2)보다 이른 시간인 경우 1을 반환하고, t1이 t2보다 늦은 경우 -1을 반환한다. 두 시간이 같은 경우 0을 반환한다.

4.2.1.2. void makeLog( )

여러 안드로이드 기기에서 보내온 여러 개의 메시지를 하나의 로그 파일에 출력하는 함수이다. 각 메시지는 다음과 같은 xml 형식으로 ‘/hsb/inputs/input.xml’에 출력된다.

```
<Message>
  <ID>지기</ID>
  <Time>15:33:10</Time>
  <Text> 그냥 자동으로 되면 회의록 쓰는 사람도 부담이 없을 텐데.</Text>
</Message>
```

단, 안드로이드 기기로부터 ‘안건’이나 ‘결론’으로 지정되어 보내진 메시지는 ID에 보낸 참여자의 ID 대신에 각각 ‘안건’과 ‘결론’을 적는다.

4.2.1.3. void makeIdFiles( )

‘/hsb/inputs/input.xml’에서 메시지를 추출해 내어 각 ID 별로 파일을 만든다. 각 ID 이름으로 된 xml 파일(ex. /hsb/outputs/id.xml)에는 각 ID를 가진 참여자가 보낸 메시지가 다음과 같이 출력된다.

```
<Message>
  <Time>15:33:10</Time>
  <Text> 그냥 자동으로 되면 회의록 쓰는 사람도 부담이 없을 텐데.</Text>
</Message>
```

4.2.1.4. void mergeIdFiles( )

참여자마다 ID 파일이 만들어지면, User 구조체를 이용해서 메시지 정보를 저장한다. 이 User 구조체를 이용하여 시간을 기준으로 메시지를 정렬한다. 이 결과로 각 참여자들에 대한 User 구조체가 만들어지며, 각 User 구조체는 정렬된 메시지들에 대한 정보를 갖는다. 정렬된 메시지에서 더 이른 시간을 나타내는 메시지를 스크립트 파일(/hsb/outputs/script.txt)에 출력한다. 각 메시지는 다음과 같은 형식으로 스크립트 파일에 출력된다.

```
지기 : 그냥 자동으로 되면 회의록 쓰는 사람도 부담이 없을 텐데.
```

단, ID가 '안건'또는 '결론'이면 다음과 같은 형식으로 스크립트 파일에 출력된다.

<안건> 오픈플랫폼 부문 참가를 위한 아이디어.

<결론> 자동으로 회의록 작성을 하는 장치로 오픈플랫폼 부문을 참가하자.

4.2.2. V2T.java

4.2.2.1. void onCreate(Bundle savedInstanceState)

'한석봉' 안드로이드 애플리케이션이 실행될 때, 이벤트를 설정하는 등 초기화 과정을 수행한다. 설정되는 이벤트는 초기 실행 시 팝업되는 다이얼로그 창의 '이름 입력' 창과 '접속' 버튼, '회의 시작' 버튼에 대한 이벤트와 메인 액티비티에 위치한 '안건' 버튼, '결론' 버튼에 대한 이벤트이다.

4.2.2.2. void onReadyForSpeech(Bundle params)

회의 참여자가 발언을 시작하고 음성 인식을 위해 대기하는 과정을 수행한다.

4.2.2.3. void onBeginningOfSpeech( )

회의 참여자가 발언을 시작할 때 호출되는 이벤트 핸들러이다. 이 이벤트 핸들러가 호출되면, 패킷에 기록하여 전송하기 위해 발언 시작 시간을 기록한다. 발언 시작 시간을 기록하기 위해 Message 클래스의 setTime 함수를 호출한다(<4.2.3.2. void setTime(Date date)> 참고).

4.2.2.4. void onEndOfSpeech( )

회의 참여자가 발언을 끝마치면 호출되는 이벤트 핸들러이다. 발언하는 동안 버퍼에 저장된 음성 파일을 텍스트로 변환하기 위한 준비를 한다.

4.2.2.5. void onResults(Bundle results)

회의 참여자의 발언이 문자열로 변환되면 호출되는 이벤트 핸들러이다. 문자열로 변환된 참여자의 발언 정보는 매개변수인 Bundle results를 통해서 전달된다. 매개변수인 results를 이용하여 Message 클래스의 buildMessage 함수를 호출한다(<4.2.3.3. void buildMessage(String text)> 참고). buildMessage 함수를 호출한 뒤에는 새로운 음성 인식 이벤트를 실행시켜 다음 발언을 인식하기 위한 준비를 한다.

4.2.2.6. void onBufferReceived(byte[] buffer)

음성 인식 과정 중에 회의 참여자의 음성은 버퍼에 저장된다. 이때 버퍼 사이즈보다 음성이 길어지는 경우 발생하는 이벤트에 대한 이벤트 처리기이다.

4.2.2.7. void onError(int error)

음성 인식 이벤트 중에 발생하는 에러를 처리한다. 음성 인식 이벤트 중에 발생하는 에러는 다음과 같다.

ERROR_AUDIO	녹음 과정에서 발생
ERROR_CLIENT	다른 클라이언트에서 발생

ERROR_INSUFFICIENT_PERMISSIONS	권한이 충분치 않을 때 발생
ERROR_NETWORK	네트워크와 관련되어 발생
ERROR_NETWORK_TIMEOUT	네트워크 작업에서 'time out' 이 발생
ERROR_NO_MATCH	매칭되는 음성 인식 결과가 없는 경우 발생
ERROR_RECOGNIZER_BUSY	음성인식 서비스가 바쁜 경우 발생
ERROR_SERVER	서버가 에러 상태를 보내온 경우 발생
ERROR_SPEECH_TIMEOUT	음성 입력이 없는 경우 발생

'한석봉' 안드로이드 애플리케이션은 ERROR_NO_MATCH와 ERROR_SPEECH_TIMEOUT에 대해서만 에러 처리를 하였다. ERROR_NO_MATCH와 ERROR_SPEECH_TIMEOUT이 발생한 경우, 음성 인식 이벤트를 다시 시작한다.

4.2.2.8. void run( ) //SendManager

Message 클래스의 멤버 변수인 q(큐)에서 저장하고 있는 전송 대기 중 메시지를 꺼내어 네트워크 소켓을 통해 연결된 갈릴레오 보드로 전송한다.

4.2.2.9. void run( ) //RecvManager

64 바이트 메시지를 받아 상황에 맞는 작업을 진행한다. 진행 가능한 작업으로는 '회의 시작', '회의 종료'가 있으며, '회의 시작'에 대한 작업은 '회의 시작' 버튼을 누른 사용자와 단순히 접속만 한 사용자를 구분하여 실행된다.

4.2.3. Message.java

4.2.3.1. void setMode(int val)

갈릴레오 보드로 보내지는 메시지의 모드를 바이트 스트림의 0번지에 기록한다(<6.1 전송에 이용된 메시지 프로토콜> 참고). 갈릴레오 보드로 보내지는 메시지의 모드는 '안건', '결론', '발언의 끝'을 갈릴레오 보드에서 인식하기 위해서 사용된다.

4.2.3.2. void setTime(Date date)

갈릴레오 보드로 보내지는 메시지의 시간을 바이트 스트림의 1~3번지에 기록한다. 시간을 나타내는 정수(0~23)는 1번지에 기록되며, 분을 나타내는 정수(0~59)는 2번지에 기록된다. 그리고 초를 나타내는 정수(0~59)는 3번지에 기록된다. 시, 분, 초에 대한 정보는 범위가 정해져 있으므로 모두 1 바이트만을 이용해 전달되도록 했다.

4.2.3.3. void buildMessage(boolean isGen, String text)

보내고자 하는 메시지 형식에 따라 메시지를 구성한다. 평범한 메시지의 경우 1024 바이트 스트림을 구성하며, 보드에게 전송하는 알림 메시지의 경우 64 바이트 스트림을 구성한다. 구성된 메시지는 Message 클래스의 멤버 변수인 q(큐)에 저장된다. buildMessage 함수는 음성 인식 및 문자 변환 후에 호출되는 이벤트 핸들러인 onResults와 사용자 조작으로 이루어지는 다양한 이벤트 리스너에 의해 호출된다.

4.2.3.4. byte[] getMessage( )

갈릴레오 보드로 보내질 패킷화 된 메시지는 Message 클래스의 멤버인 q(큐)에 저장되는데, 이때 q에서 하나의 메시지를 꺼내 반환하는 함수이다. q가 비어있는 경우 null을 반환한다.

4.2.4. hsb_analyze.c

4.2.4.1. 회의록 분석을 위한 함수

4.2.4.1.1. void analyze( )

회의록 분석을 하기 위한 초기 작업을 하는 함수이다. 형태소 분석기 프로그램이 만들어 낸 '/hsb/outputs/result.txt' 파일을 참고하여, 회의 참여자에 대한 정보와 형태소에 대한 정보를 트리로 저장한다. 회의 참여자에 대한 정보는 각 회의 참여자의 ID, 발언 횟수이다. 형태소에 대한 정보는 형태소에 대한 문자열 값, 등장 횟수이다. 회의 참여자에 대한 정보와 형태소에 대한 정보는 각각 user, morp 라는 트리에 저장된다. 형태소 분석기 프로그램이 만들어 낸 '/hsb/outputs/result.txt' 파일은 다음과 같은 구성을 가지고 있다.

```
[[어절]]
 형태소1  형태소2  형태소3 …
```

각 형태소는 탭 문자를 통해 구별된다. 형태소와 회의 참여자 ID를 구분하기 위해서 토큰(어절 정보)을 저장할 변수를 두 개 사용하였다. 하나는 새로운 토큰을 저장하기 위한 변수이고, 하나는 이전 토큰을 저장하기 위한 변수이다. 새로운 토큰이 ":"이 아닌 경우 이전 토큰은 의미를 가진 어절을 의미하게 된다. 따라서 이 경우 스택에 형태소를 저장해야 한다. 또한 현재 스택에 있는 이전 토큰에 대한 형태소를 morp 트리에 삽입하는 연산을 한다. 새로운 토큰이 ":"인 경우, 이전 토큰은 회의 참여자의 ID를 의미하게 된다. 따라서 이 경우 user 트리에 토큰을 삽입하는 연산을 한다. 또한 현재 스택에는 이전 토큰에 대한 형태소, 즉 회의 참여자의 ID에 대한 형태소가 들어 있게 되므로 스택을 모두 비워주는 연산을 한다.

4.2.4.1.2. void findKeyword(node* m, char* keyword)

스크립트의 모든 형태소 정보를 기반으로 키워드를 찾아내는 함수이다. 전달되는 인자는 node* m, char* keyword이다. m은 형태소 정보를 저장하고 있는 트리이다. 형태소 발생 횟수는 트리 m의 frequency 변수를 통해 알 수 있다. keyword는 현재 최빈값을 발생 횟수로 가진 형태소가 저장되는 변수이다. 트리 m의 모든 노드를 방문해야 하므로, pre-order 재귀 방식으로 호출된다. 이때 접속사나 대명사를 키워드로 인식하면 안되므로, 접속사와 대명사 등을 예외 처리 하였다.

4.2.4.1.3. void printKeyword(char* keyword)

인자로 전달되는 키워드를 스크립트 파일 하단부에 출력하는 함수이다. 출력은 다음과 같은 방식으로 이루어진다.

<Keyword>
본 회의록의 키워드는 '오픈플랫폼'(으)로 추정됩니다.

4.2.4.1.4. void analyzeParticipation(node* u, int total)

현재 방문한 트리 노드의 data 변수를 참조하여 회의 참여자의 ID 값을 얻어내고, frequency 변수와 매개변수인 total을 이용하여 참여율을 계산(<3.3.4.2. 회의 참여자들의 참여율> 참고)하는 함수이다. 전달되는 인자는 node* u, int total이다. u는 회의 참여자들의 정보가 저장된 트리이다. total은 회의 참여자들의 총 발언 횟수이다. 트리 u의 모든 노드를 방문해야 하므로, pre-order 재귀 방식으로 호출된다. 계산된 참여율은 스크립트 파일 하단부에 출력되며, 출력은 다음과 같은 양식으로 출력된다.

<Rate of Participation>
[[혜련]]의 회의 참여율은 25 % 입니다.
[[지기]]의 회의 참여율은 25 % 입니다.
[[혜수]]의 회의 참여율은 25 % 입니다.
[[상열]]의 회의 참여율은 25 % 입니다.

4.2.4.1.5. void analyzeKeywordHit(node* u, char* keyword)

각 회의 참여자들이 키워드를 언급한 횟수를 계산하는 함수이다. 인자 node* u, char* keyword는 각각 회의 참여자들의 정보를 저장한 트리와 키워드이다. 키워드 언급 횟수를 계산하기 위해 스크립트 파일의 모든 토큰이 키워드를 포함하는지 검사해야 한다. 스크립트 파일의 토큰이 키워드를 포함하는 경우 incKey 함수를 호출한다. 단, 회의 참여자의 ID가 키워드를 포함하는 경우 decKey 함수를 호출한다. 회의 참여자의 ID는 다음 토큰이 ":"인지를 검사함으로써 인식할 수 있다.

4.2.4.1.6. void printKeywordHit(node* u, char* keyword)

각 회의 참여자들의 키워드를 언급한 횟수를 스크립트 하단부에 출력하는 함수이다. 인자는 node* u, char* keyword이며, 각각 회의 참여자의 정보를 저장하고 있는 트리와 키워드이다. 트리 u의 모든 노드를 방문해야 하므로, pre-order 재귀 방식으로 호출된다. 계산된 참여율은 스크립트 파일 하단부에 출력되며, 출력은 다음과 같은 양식으로 출력된다.

<Keyword Hit>

[[혜련]]은 키워드 '오픈플랫폼'(을)를 2 번 언급했습니다.

[[지기]]은 키워드 '오픈플랫폼'(을)를 3 번 언급했습니다.

[[혜수]]은 키워드 '오픈플랫폼'(을)를 3 번 언급했습니다

[[상열]]은 키워드 '오픈플랫폼'(을)를 2 번 언급했습니다

4.2.4.1.7. void incKey(node* u, char* id)

인자 node* u는 회의 참여자들의 정보가 들어있는 트리의 루트이며, char* id는 검색하고자 하는 회의 참여자의 ID이다. u에서 id를 가진 회의 참여자의 트리 노드를 찾아내어, keyword 변수를 증가시킨다.

4.2.4.1.8. void decKey(node* u, char* id)

인자 node* u는 회의 참여자들의 정보가 들어있는 트리의 루트이며, char* id는 검색하고자 하는 회의 참여자의 ID이다. u에서 id를 가진 회의 참여자의 트리 노드를 찾아내어, keyword 변수를 감소시킨다.

4.2.4.1.9. void makeKeywordFile(int numKeyFile, node* morp)

형태소 정보는 회의록의 안건 별로 분리되어 저장되어야 한다. 왜냐하면, 각 안건마다 키워드가 다를 것이기 때문이다. 따라서 안건 별로 형태소 정보를 파일로 출력할 필요가 있다. 인자 int numKeyFile은 현재까지 만들어진 키워드 파일(형태소 정보)의 수이다. 0부터 시작한다. 인자 node* morp는 파일에 출력할 형태소 정보를 저장하고 있는 트리의 루트이다. morp에 저장된 형태소 정보를 기록할 수 있는 파일 '/hsb/outputs/keywordN.txt'(N = 0, 1, 2, 3, …)을 만들고 writeKeyword 함수를 호출하여 형태소 정보를 파일에 출력한다.

4.2.4.1.10. void writeKeyword(FILE* keyFile, node* morp)

인자 FILE* keyFile은 형태소 정보가 출력될 파일이고, node* morp는 형태소 정보를 담고 있는 트리 노드이다. morp에 저장된 정보를 keyFile에 출력한다. writeKeyword는 재귀적으로 호출된다.

4.2.3.1.11. report(node* morp, node* user, char* keyword, int total)

회의록 분석을 하기 위해 다른 함수를 호출하는 함수이다. 인자 node* morp는 형태소 정보를 저장하고 있는 트리이다. node* user는 회의 참여자 정보를 저장하고 있는 트리이다. char* keyword는 키워드가 저장될 변수이며, report 함수 호출 단계에서는 회의록의 키워드가 저장되지 않은 상태이다. int total은 회의 참여자들의 총 발언 횟수이다. report 함수는 회의록 분석을 위해 인자 morp의 정보를 참조하여 키워드(최빈값으로 추정)를 찾아내는 findKeyword 함수를 호출한다. findKeyword함수의 결과는 keyword 변수에 저장되며, 이를 이용하여 analyzeKeywordHit 함수를 호출한다. analyzeKeywordHit 함수는 각 회의 참여자들이 키워드를 몇 번 언급했는지를 계산한다. 회의록 분석을 마친 뒤에 스크립트 파일에

키워드, 회의 참여자의 참여율, 각 참여자의 키워드 언급 횟수를 출력한다.

4.2.4.2. 트리 구조를 위한 함수

4.2.4.2.1. void insert(node* r, char* id)

트리 구조에 새로운 트리 노드를 삽입하기 위한 함수이다. 인자 node* r, char* id는 각각 삽입할 트리의 루트, 삽입할 데이터 값을 의미한다. 데이터 값은 문자열이므로, 이진 트리(Binary Tree) 형태를 만들기 위해 strcmp 함수를 이용하였다. strcmp 함수는 두 문자열을 인자로 받아 첫 번째 인자가 사전적으로 먼저 나오는 단어이면 음수를 반환하고, 같으면 0을 반환하며, 두 번째 인자가 사전적으로 먼저 나오는 단어이면 양수를 반환한다. 삽입할 문자열과 현재 트리 노드의 데이터 값이 같은 경우 현재 트리 노드의 frequency 변수를 하나 증가시킨다.

4.2.4.2.2. void freeNode(node* r)

인자로 전달된 서브 트리의 각 노드의 메모리를 해제하는 함수이다. 트리의 루트가 가장 마지막에 메모리 해제 되어야 하므로, post-order 재귀 방식으로 호출된다.

4.2.4.3. 스택 구조를 위한 함수

4.2.4.3.1. int isEmpty(stack* s)

인자로 전달된 스택이 비었는지를 검사하는 함수이다. 인자로 전달된 스택의 top 변수가 음수이면 스택이 비었다는 의미이므로 1을 반환하고, 비지 않은 경우 0을 반환한다.

4.2.4.3.2. void push(stack* s, char* data)

인자로 전달된 스택에 데이터를 추가하는 함수이다. 이때 인자로 전달된 스택의 top 변수를 하나 증가시켜야 한다.

4.2.4.3.3. void pop(stack* s, char* data)

인자로 전달된 스택에서 데이터를 꺼내는 함수이다. 꺼내진 데이터는 인자로 전달된 데이터를 나타내는 포인터에 저장되며, 이때 스택의 top 변수를 하나 감소시켜야 한다.

4.2.5. hsb_sc2html.c

4.2.5.1. void makeHtmlFile( )

만들어진 스크립트 내용을 html 파일로 만드는 함수이다. 특수 문자나 개행 문자에 대하여 HTML 문법에 맞게 처리를 하였다.

4.2.6. server.js

4.2.6.1. function callback_server_connection(socket)

클라이언트로부터 접속이 발생하는 경우 호출되는 콜백 함수이다. sockets 리스트 변수에 접속을 시도한 클라이언트의 소켓을 저장한다.

4.2.6.1.1. socket.on('data', function(data) { })

접속되어 있는 클라이언트 소켓으로부터 데이터를 받는 경우 호출되는 콜백 함수이다. 받은 데이터의 크기를 통해 신호인지 메시지인지를 구분한다. 신호의 경우 64바이트의 크기를 가지고 있으며, 처리하는 신호는 '회의 시작', '회의 종료'이다. '회의 시작' 신호를 받는 경우 연결된 모든 소켓에게 회의 시작에 대한 정보를 브로드캐스트(broadcast)한다. '회의 종료' 신호를 받는 경우 소켓을 닫는다. 메시지의 경우 1024바이트의 크기를 가지고 있으며, 메시지를 받는 경우 './inputs/id.txt'에 메시지를 보내온 클라이언트의 이름을 출력하고 './inputs/message.bin'에 메시지의 내용을 출력한다.

4.2.6.1.2. socket.on('end', function( ) { })

접속되어 있는 클라이언트 소켓이 닫히는 경우 호출되는 콜백 함수이다. 닫힌 소켓의 정보를 콘솔에 출력한다.

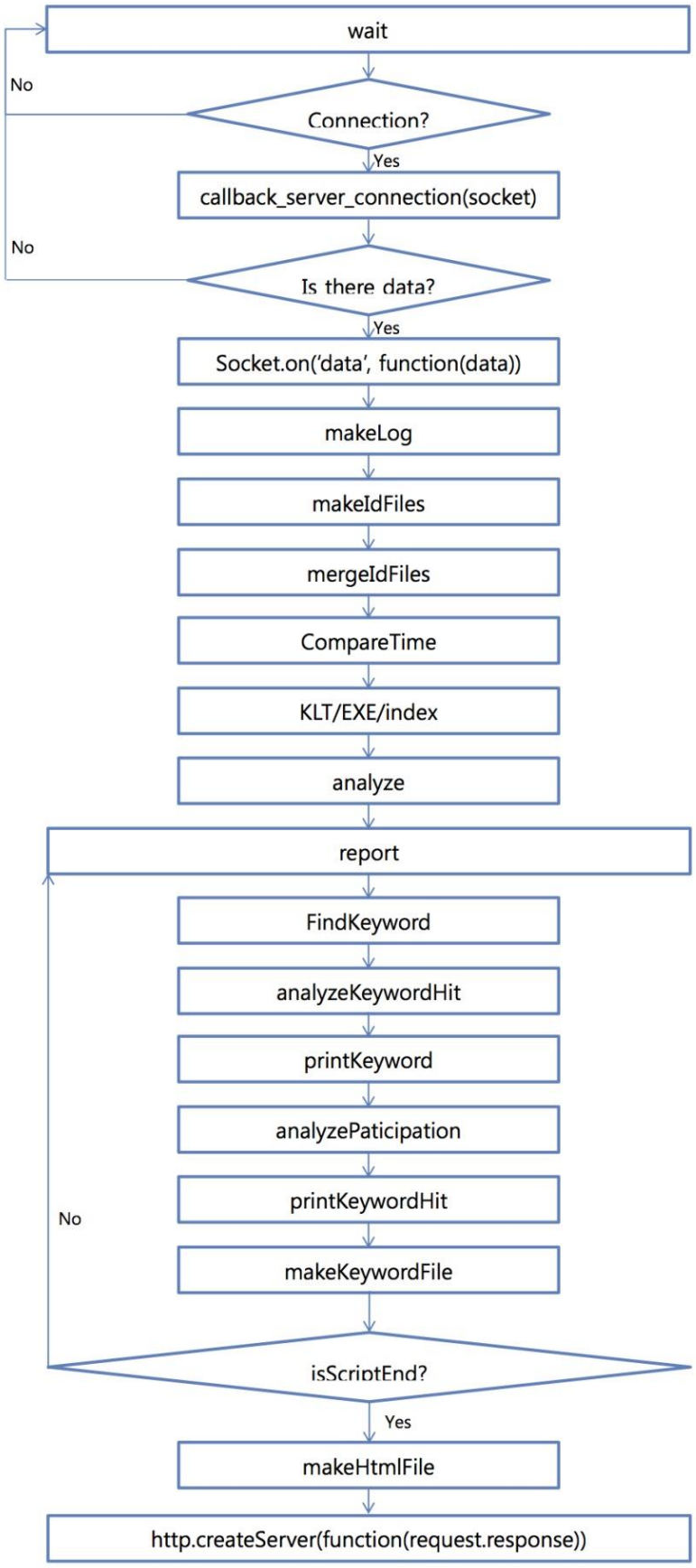
4.2.6.2. http.createServer(function (request, response) { })

'한석봉'이 회의록을 HTML 파일로 만든 후에 실행된다. '한석봉'으로부터 만들어진 './outputs/hsb.html' 파일을 http 호스트로서 웹 페이지에 게시한다.

4.3. 주요 함수의 흐름도

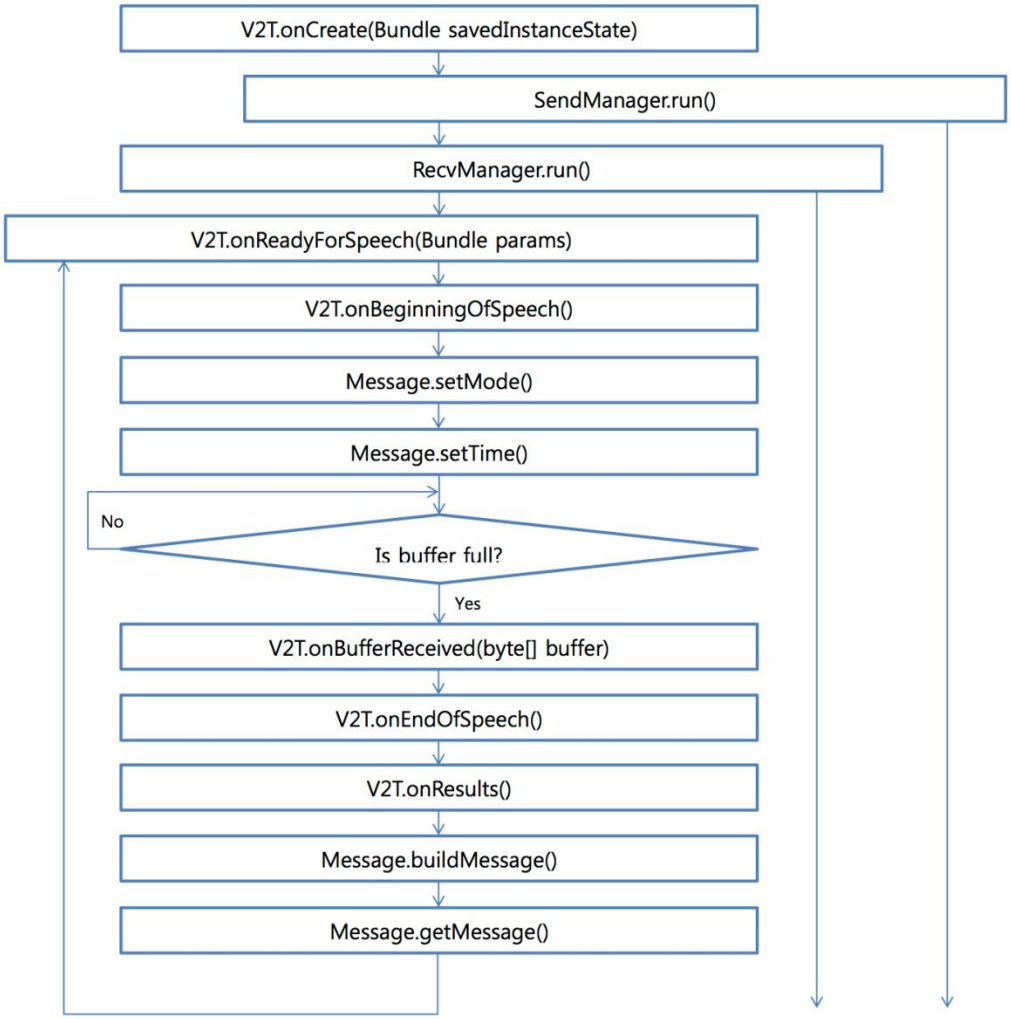
4.3.1. Galileo Board Application

갈릴레오 보드에서 실행되는 '한석봉' 갈릴레오 보드 애플리케이션이다. 주요 함수의 흐름도는 다음과 같다.


wait
No
Connection?
Yes
callback_server_connection(socket)
No
Is there data?
Yes
Socket.on('data', function(data))
makeLog
makeIdFiles
mergeIdFiles
CompareTime
KLT/EXE/index
analyze
report
FindKeyword
analyzeKeywordHit
printKeyword
analyzePaticipation
printKeywordHit
makeKeywordFile
No
isScriptEnd?
Yes
makeHtmlFile
http.createServer(function(request.response))

4.3.2. Android Mobile Application

회의 참여자들이 사용하게 될 안드로이드 기기에서 실행되는 '한석봉' 안드로이드 애플리케이션이다. 주요 함수의 흐름도는 다음과 같다.



4.4. 기술적 차별성

4.4.1. 웹을 통한 회의록 열람

보통의 회의록은 파일로 만들어져, 해당 회의록 파일이 요구하는 뷰어 애플리케이션이 있어야 회의록을 열람할 수 있다. 예를 들어, 회의록이 '.hwp'로 만들어져 있다면 아래아 한글 애플리케이션이 있어야 회의록을 열람할 수 있다. 또는 회의록 파일을 다운로드 받아야만 회의록의 내용을 열람할 수 있다. '한석봉'은 회의 후 만들어진 회의록 파일을 HTML 파일로 만들어, 웹 페이지에 게시한다. 이를 통해, 사용자는 회의록의 내용을 별도의 뷰어 애플리케이션 없이도 회의록을 열람할 수 있다. 또한, 별도의 다운로드 절차 없이도 회의록의 내용을 확인할 수 있다.

4.4.2. KLT(Korean Language Technology) 프로그램 사용

한글은 영어에 비해 최소 의미를 갖는 단위인 형태소 분석이 힘들다는 특징이 있다. '한석봉'은 KLT 프로그램을 사용하여 형태소 분석을 함으로써 거의 모든 어절에서 형태소를 정확하게 분리해낸다.

4.4.3. 독자적 통신 프로토콜 사용

'한석봉'은 갈릴레오 보드와 안드로이드 기기가 통신하기 위한 독자적 통신 프로토콜을 사용한다. 따라서 전송되는 데이터는 꼭 필요한 정보만을 포함하게 되며, 이를 통해 바이트 스트림을 최적화하여 사용할 수 있다.

4.4.4. 빈도수 체크를 통한 회의 키워드 추출

'한석봉'은회의 결과 만들어진 스크립트로부터 형태소 분석을 통해 키워드를 추출한다. 이 키워드는 안건 별로 키워드 파일로 기록되며, 가장 많이 언급된 키워드를 정렬함으로써 회의 내용을 유추할 수 있다. 이는 아주 긴 회의록이나 복잡한 회의록에서 중요한 내용을 이해하는 데에 도움을 줄 수 있다.

5. 응용 분야

5.1. 효율적인 회의 기록

회의를 기록하기 위해서는 회의 기록을 담당하는 사람이 일일이 회의 때마다 회의에 대한 정보를 기록해야 한다. 즉, 회의록을 작성하기 위해서는 회의 기록을 담당하는 사람이 매번 회의에 참여해야 한다는 것이 전제되어야 한다. 또한 회의에 대한 정보가 회의 기록을 담당하는 사람의 주관에 의해 잘못 기록될 가능성도 있다. '한석봉'의 기본적인 기능은 회의록 자동 작성 기능이다. 회의가 필요한 다양한 기관 또는 집단에서 '한석봉'을 통해 효율적인 회의 기록을 할 수 있다. '한석봉'을 통해서 따로 회의 기록을 담당하는 사람을 정할 필요 없으며, 객관적인 회의록 작성이 가능하다.

5.2. 회의 참여자 평가

'한석봉'은 회의록 분석을 통해 회의 참여자의 회의 태도를 추정할 수 있도록 돕는다. 발언 횟수를 통해 회의 참여자의 참여율을 추정할 수 있고, 키워드 언급 횟수를 통해 회의 참여자가 얼마나 회의와 관련된 발언을 했는지를 추정할 수 있다. 이와 같은 회의 참여자의 평가를 통해 더 효율적인 회의 문화를 만들 수 있을 것이다.

5.3. 음성 인식 인터페이스 사용

최근 음성 인식에 대한 관심이 커져가고 있다. 이와 같은 상황에서 '한석봉'의 음성 인식 인터페이스 사용은 훌륭한 음성 인식 애플리케이션의 예시가 될 수 있다.

6. 기타

6.1. 전송에 이용된 메시지 프로토콜

각 회의 참여자들의 안드로이드 기기는 참여자의 음성을 인식하여 문자로 변환한 뒤, 패킷화 하여 갈릴레오 보드로 정보를 전송해야 한다. 이때 갈릴레오 보드가 필요로 하는 정보는 다음과 같다.

6.1.1. 메시지를 보낸 회의 참여자의 ID

스크립트에 출력하기 위해서 메시지를 보낸 회의 참여자가 누구인지 식별할 수 있어야 한다. 하지만 이는 굳이 회의 참여자의 안드로이드 기기에서 알려줄 필요 없이, 안드로이드 기기가 갈릴레오 보드에 네트워크 접속을 할 때 갈릴레오 보드에서 정보를 저장해 놓음으로써 알 수 있는 정보이다. 안드로이드 기기가 갈릴레오 보드에 네트워크 접속을 할 때 갈릴레오 보드는 각 안드로이드 기기의 IP와 ID를 매칭하여 저장할 수 있기 때문이다. 따라서 안드로이드 기기에서 전송될 패킷에 회의 참여자의 ID가 포함될 경우 불필요한 메모리 차지를 할 수 있다고 생각되어, 회의 참여자의 ID 정보는 패킷의 정보로서 생략되었다.

6.1.2. 메시지의 종류

회의 중에 발생하는 참여자의 발언에는 세 가지 종류가 있다고 판단된다. 첫 번째는 '안건'이며, 두 번째는 '결과'이며, 세 번째는 '일반 발언' 이다. 이를 구분하여 다른 형식으로 스크립트에 출력하는 것이 더 좋다고 생각되었다. 따라서 갈릴레오 보드에서는 안드로이드 기기가 보내는 메시지가 어떤 종류인지를 판단할 수 있어야 한다.

6.1.3. 메시지를 보낸 시간

스크립트에 대화를 출력하기 위해, 각 메시지의 순서를 알 필요가 있다. 각 메시지의 순서는 갈릴레오 보드에서 수신 받은 시간이 아니라, 안드로이드 기기에서 음성 인식을 한 시간을 기준으로 한다(<4.2.2.3. void onBeginningOfSpeech( )> 참고). 여러 대의 안드로이드 기기에서 갈릴레오 보드로 여러 개의 패킷을 전송할 경우 갈릴레오 보드가 수신하는 패킷 순서와 실제 대화의 순서가 불일치할 수 있기 때문이다.

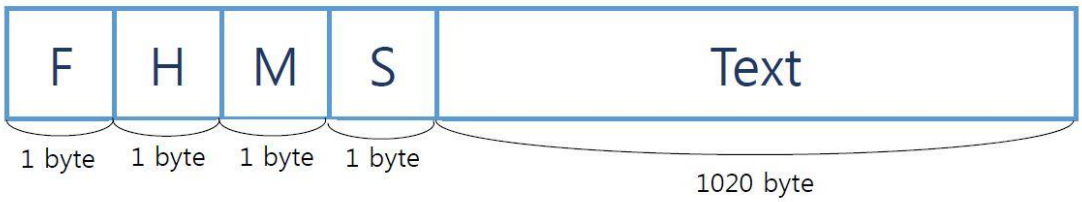
6.1.4. 메시지의 내용

실질적으로 스크립트에 쓰여질 대화의 내용이 필요하다.

이와 같은 정보를 담기 위해 1024 바이트의 스트림을 이용하였으며, 각 바이트는 0~1023의 번지 수로 참조가 가능하다. 이때 첫 번째 바이트인 0번지는 '메시지의 종류'에 대한 정보를 담기 위해 이용되었으며, 1~3번지는 '메시지를 보낸 시간'에 대한 정보를 담기 위해 이용되었다. 그리고 나머지 4~1023번지는 '메시지의 내용'에 대한 정보를 담기 위해 이용되었다.

C : 결론(Conclusion)인지 아닌지를 나타내는 bit
A : 안건(Agenda)인지 아닌지를 나타내는 bit
E : 발언의 끝(End)인지 아닌지를 나타내는 bit

0번지의 바이트 중에서 LSB(Least Significant Bit)는 회의 참여자의 발언이 끝이 난 경우와 아직 끝이 나지 않은 경우를 판단하기 위해 사용된다. 이는 <6.2. 문제점과 해결 알고리즘>에 나타난 문제 상황을 처리하기 위한 flag이다. LSB가 1이면 발언이 끝이 난 경우이며, 0이면 아직 발언이 끝이 나지 않은 경우를 의미한다. 또한 LSB의 옆 비트는 메시지의 종류가 '안건'인지 아닌 지를 판단하기 위해 사용된다. 1이면 메시지의 종류가 '안건'인 경우이며, 0이면 메시지의 종류가 '안건'이 아닌 경우이다. '안건'을 나타내는 비트의 왼쪽 비트는 메시지의 종류가 '결론'인지 아닌 지를 판단하기 위해 사용된다.



F : 메시지의 종류에 대한 플래그(flag)를 나타내는 byte
H : 메시지를 보낸 시간(hour)을 나타내는 byte
M : 메시지를 보낸 분(minute)을 나타내는 byte
S : 메시지를 보낸 초(second)를 나타내는 byte
Text : 메시지의 내용이 담길 byte stream

1번지 바이트는 시간을 나타내는 정수(0~23)가 저장된다. 2번지 바이트는 분을 나타내는 정수(0~59)가 저장된다. 3번지 바이트는 초를 나타내는 정수(0~59)가 저장된다. 나머지 4~1023번지 바이트는 메시지의 내용이 저장된다.

6.2. 문제점과 해결 알고리즘

6.2.1. 동시 발언 상황

회의 중에 여러 참여자가 동신에 발언을 할 가능성이 있다. 이 경우 스크립트에 출력될 순서가 애매하다는 문제가 있다. 예를 들어 한 명이 발언을 하는 중에 다른 누군가 끼어들어 말을 거나 동조의 발언을 할 수 있다.

혜수 : 오픈 플랫폼 분야 개발 완료 보고서 마감 시기가 얼마 남지 않아서 보고서 작업을 우선적으로

지기 : 하지만 개발이 조금 더 빨리 완성되어야 하는 것 아닙니까?

혜수 : 해야 한다고 생각합니다.

이 경우 스크립트에는 다음과 같은 형식으로 출력되는 것이 바람직하다.

혜수 : 오픈 플랫폼 분야 개발 완료 보고서 마감 시기가 얼마 남지 않아서 보고서 작업을 우선적으로 해야 한다고 생각합니다.

지기 : 하지만 개발이 조금 더 빨리 완성되어야 하는 것 아닙니까?

6.2.2. 지나치게 긴 발언 상황

회의 참여자의 안드로이드 기기는 패킷화 된 메시지를 갈릴레오 보드로 전송한다. 이때 패킷의 크기는 1024바이트 이므로, 메시지의 내용이 길어지는 경우 안드로이드 기기는 여러 개의 패킷을 만들어 갈릴레오 보드로 전송하게 된다.

혜련 : 아니 그러면 여러 사용자가 메시지를 동시에 보내게 될 수도 있는 거 아냐? 그런 상황에 대해서 처리를 해야 하는데, 지금 이 방식으로는 문제가 분명히 생길 것 같아. 내 생각에는 이런 문제를 해

혜련: 결할 방법이 필요한데 어떻게 해볼까?

이 경우 스크립트에는 다음과 같은 형식으로 출력되는 것이 바람직하다.

혜련 : 아니 그러면 여러 사용자가 메시지를 동시에 보내게 될 수도 있는 거 아냐? 그런 상황에 대해서 처리를 해야 하는데, 지금 이 방식으로는 문제가 분명히 생길 것 같아. 내 생각에는 이런 문제를 해결할 방법이 필요한데 어떻게 해볼까?

6.2.3. 해결 알고리즘

두 문제 상황 모두 갈릴레오 보드로 전송되는 메시지가 발언의 끝인지 아닌지를 판단할 수 있다면 해결이 된다. 즉, 갈릴레오 보드는 전송된 메시지가 발언의 끝인지 아닌지를 판단하여 스크립트에 출력해야 한다. 이를 위해 패킷화 된 메시지의 0번지를 이용해서 전달된 메시지가 발언의 끝인지 아닌지를 판단하게 했다. 갈릴레오 보드는 전달받은 메시지가 발언의 끝인지를 인식하고, 발언의 끝인 경우 해당 ID에 다음과 같은 메시지를 추가하여 정보를 저장할 수 있도록 한다.

```
<Message>
  <ID>지기</ID>
  <Time>15:33:10</Time>
  <Text> 그냥 자동으로 되면 회의록 쓰는 사람도 부담이 없을 텐데.</Text>
</Message>
```

즉, Text 태그의 값이 "***"이 될 때까지 스크립트에 해당 ID의 메시지 내용을 출력할 수 있도록 한다.

6.3. 보완할 사항 및 보완 방안

6.3.1. 스크립트 문서화(txt to docx)

지금 상태의 '한석봉'이 만드는 회의록은 스크립트 형식이다. 즉 단순한 텍스트 파일의 형식이다. 그러나 회의록은 출력 가능한 형태인 것이 더 바람직하다고 생각된다. 따라서 단순한 텍스트 파일이 아니라 오피스 오픈 XML을 이용해서 회의록을 만드는 방식으로 변경하고자 한다.

6.3.2. 게시판 식의 웹 페이지 게시

지금 상태의 '한석봉'이 만드는 회의록은 HTML 파일로 만들어져, 웹페이지에 게시된다. 그러나 여러 개의 파일이 게시판 형식을 통해 게시된다면, 여러 번의 회의 내용을 한 번에 볼 수 있을 것이다. 따라서 여러 개의 회의록 HTML 파일을 서버에서 관리하고, 게시판 형식을 통해 회의록 열람을 할 수 있도록 하고자 한다.

6.3.3. 전자 서명

일반적으로 회의록은 각 회의 참여자들의 서명을 통해서 공시 능력을 가지게 된다. 즉 회의 참여자들이 회의록에 서명함으로써, 회의록은 각 회의 참여자의 공식적인 입장을 대변할 수 있는 능력을 가지게 된다. 지금 상태의 '한석봉'은 각 회의 참여자들로부터 서명을 받을 수 있는 기능이 없다. 이를 보완하기 위해 전자 서명 기능을 추가하여 회의록에 공시 능력을 주고자 한다.

7. 제작자 정보

No.	구분	성명	소속(학교)	부서(학과)	입학년도	담당업무
1	팀장	박상열	국민대학교	컴퓨터공학부	2012년	보드 환경 구축, 네트워크 통신 서버 설계
2	팀원	김호연지기	국민대학교	컴퓨터공학부	2012년	'한석봉' 안드로이드 애플리케이션 개발
3	팀원	박혜련	국민대학교	컴퓨터공학부	2011년	보드 환경 구축, 자연어처리, 데모 영상 촬영 및 편집
4	팀원	신혜수	국민대학교	컴퓨터공학부	2012년	'한석봉' 갈릴레오 보드 애플리케이션 개발, 자연어처리

8. 개발 단계별 기간 및 투입 인원

개발단계	기간(주)								인원수 (명)	투입 인원
	7/4	8/1	8/2	8/3	8/4	9/1	9/2	9/3		
보드 환경 구축									2	박상열,박혜련
네트워크 서버 설계									1	박상열
안드로이드 모바일 애플리케이션									1	김호연지기
갈릴레오 보드 애플리케이션									1	신혜수
자연어처리									2	신혜수,박혜련
테스트 및 디버깅									4	전 팀원