스마트폰에서 실시간 음성 통신을 위한 UDP Socket Server 구현

*강지희 **손한비 ***임양미

덕성여자대학교 디지털미디어학과

***yosimi@duksung.ac.kr

Implement UDP Socket Server for Real-time Voice Communication on Smart-phone

*Kang, Ji-Hee **Son, Han-Bee ***Lim, Yang-Mi

***Dept. of Digital Media, Duksung Women's University

요약

최근 오디오 기반의 그룹 대화 통신 기술이 급격히 발전하고 있는데 이는 원거리 간의 회의 또는 긴급 구조망, 음성 인식을 활용한 기술 분야에서 필요로 하기 때문이다. 과거 오디오 그룹 간의 실시간 서비스는 영상 통신보다 타이밍에 있어서 사용자에게 딜레이 되는 값을 전송하는 즉 버퍼 컨트롤이 문제가 되어 잘 사용되지 않았었다. 하지만 최근 다중경로 라우팅, QoS 전송량 감소 기술들이 소개되면서 N:N의 대화가 가능하게 되었다. 본 연구에서는 UDP Socket 방식을 활용하여 N:N 실시간 음성 서비스를 개발한다. 이는 무선단말기를 활용하여 3~4인이 그룹핑 되어 노래 경쟁을 할 수 있는 앱에 적용하여 개발하였다. 운전자가 혼자 운전할 때, 다른 지역에서 운전하는 사람들과 음성인식 인터페이스를 활용하여 즉각적인 그룹을 만들고, 자신과 다른 사람들이 노래를 부르고, 듣고 평가하는 과정에서 재미를 느끼게 함으로써 졸음을 방지할 수 있도록 개발하였다.

1. 서론

최근 오디오 기반의 그룹 채팅 또는 그룹 대화 통신을 이용한 앱 개발이 상당히 활발하다. 오디오 그룹을 통한 실시간 서비스는 영상 통신보다 타이밍에 민감하여 사용자에게 딜레이 되는 전송 값 즉 버퍼 컨트롤의 문제로 인해 이를 해결해야 하는 문제점이 많았지만, 이제는 IBM, Google, NHN, Daum 등과 같은 업체들이 무선 Ad Hoc에서 실시간 음성서비스의 필요를 시급하게 느끼고 다중경로 라우팅, 블루투스 스캐터넷 라우팅, QoS, 전송량 감소 등에 대한 기술들을 다양하게 선보이고 있다 [1,2,3]. 일반적으로 애드혹 매쉬 네트워크 환경에서 음성/영상 스트리밍 데이터의 멀티캐스팅 라우팅 기술에는 MPR(multi-point relay) 기반의 1:1통화 기능 지원과 MMP(multimedia multicast routing protocol)의 N:N 통화 기능을 지원하는 기술이 있다 [4]. 이러한 무선 실시간 음성 지원 기술은 동호회, 긴급 구조 운영팀, 원거리 회의 그룹 등 다양한 곳에서 효율적으로 사용할 수 있다. 본 연구에서 무선 단말기를 활용하여 3~4인이 모여 노래 경쟁을 할 수 있는 앱으로 운전자가 혼자 운전할 때, 다른 지역에서 운전하는 사람들과 즉각적인 그룹을 만들어 자신과 다른 사람들의 노래를 부르고, 듣고 평가하는 과정에서 재미를 통해 졸음을 방지하는 데에 있다. 본 앱을 개발하기 위해서 UDP 기반의 스마트폰과 서버와의 네트워크 환경 설정이 중요하다. 네트워크 통신 방법에는 UDP와 TCP가 있는데 실시간 음성 스트리밍을 위해 UDP Socket방식을 기반으로 한 N:N 실시간 음성 서비스를 연구하였으며, 이 방식은 실시간 상호작용이 필요한 앱 서비스에 적합하다.

본 연구의 2장에서는 UDP Socket을 사용한 스마트 폰과 서버의 통신 방법과 서버 내 음성 전달 방식을 설명한다. 3장에서는 UDP Socket을 이용해 스마트 폰에서 음성을 전달하는 구현 과정을 소개하고 4장에서 결론을 맺는다.

2. 스마트 폰과 서버와의 통신에 대한 모델

본 장에서는 서버를 중심으로 스마트 폰 간의 실시간 음성 통신을 가능하게 하기 위해 주고받을 정보에 대한 프로토콜을 설계한다. 사용자가 스마트 폰 앱을 통해 서버로 접속한 후 자신의 정보를 전달하고 자신이 속한 그룹의 정보를 받는 등의 실행 모드(대결하기 모드)에서 데이터를 처리를 위해 서버와 클라이언트 간의 프로토콜이 필요하다.

상호작용 데이터는 크게 CP(Common Part)영역과 OP(Option Part)영역으로 나뉘며, CP영역은 상호작용 데이터에 대한 전반적인 정보를 담고 있고 OP영역은 서버로 전달하고자 하는 실제 데이터를 담고 있다. 표1은 CP영역을 정의한 내용으로 각 항목들은 byte단위로 이루어져있다. Course항목은 해당 데이터의 첫 부분 1byte를 차지하고 있다. 스마트 폰에서 서버로 보내는 데이터인 경우에는 0 값을, 서버에서 스마트 폰으로 보내는 데이터인 경우에는 1 값을 갖는다. CP영역의 Group항목과 Command항목을 통해 어떤 명령에 대한 데이터인지를

제어하도록 하였다. 데이터를 받으면 Group항목과 Command항목을 비교하여 맞는 함수를 실행하도록 구현하였다.

표 1. 데이터의 CP
Table 1. The CP of data

Contents	Length	Contents
Course	1	0: HP→Server, 1: Server→HP
Length	2	Byte unit of control message length
Type	1	0: Transmission, 1: Reply
Error Status	1	Default 0
Group	1	
Command	1	

Group항목과 그에 해당하는 Command항목들을 표2를 통해 알 수 있다. Group항목은 데이터를 보내는 주체가 클라이언트인지 서버인지에 따라 GROUP_INFO, GROUP_SERVER를 결정하고 세션 데이터의 경우 GROUP_MONITOR로 분류된다. GROUP_INFO의 Command는 대결하기 모드 시에 클라이언트가 서버에 접속하거나 노래의 시작과 끝, 점수와 관련된 데이터를 보낼 때의 명령을 정의하고 있다. GROUP_SERVER의 Command는 서버에서 클라이언트 그룹에게 그룹원의 정보, 그룹의 singer 정보, 점수 요청 및 singer의 점수 정보, 그룹의 노래 종료를 보내는 데이터 명령을 정의하고 있다. GROUP_MONITOR의 Command는 클라이언트가 접속해 있는지 세션을 관리하는 명령을 정의하고 있다.

표 2. CP영역 중 Group과 Command 항목
Table 2. The group and command list in the CP area

Group	Command	Value	Comment
GROUP_INFO	INFO_HP	1	HP(user) ID Transmit.
	INFO_START	2	HP Sing Start.
	INFO_STOP	3	HP Sing Stop.
	INFO_SCORE	4	HP Sing Score.
GROUP_SERVER	SERVER_CONNLIST	1	Notify group members of new HP connections.
	SERVER_HPSTART	2	Notify start signal to singer.
	SERVER_RSCORE	3	Ask for a singer's score.
	SERVER_SCORE	4	Notify singer's score to all members in group.
	SERVER_FINISH	5	All members in group are finished sing.
GROUP_MONITOR	MONITOR_SESSION	1	HP sends signal to server every 3 seconds.
	MONITOR_MD	2	Member exited.
	MONITOR_GD	3	Group destroyed.

이렇게 정의한 프로토콜을 통해 서버와 클라이언트가 데이터를 주고받는데 서버는 클라이언트들로부터 ID를 전달 받아 관리하고 최대 4명의 클라이언트로 구성된 Group을 만들어 Group 내 Member 간의 음성 데이터를 전달할 수 있도록 하는 전반적인 구조를 가진다.

서버와 클라이언트 사이의 음성 전달 방법은 그림1과 같다. 그림 1은 전반적인 음성 전달과정에서 A가 singer로 선정되었을 때이다. 서버가 음성 전달을 위한 50003번 포트를 열어두고 A가 서버의 해당 포트로 음성을 전달한다. 서버는 전달 받은 음성 데이터를 그룹 내 멤버들에게 전달한다. 이 때, 그룹 내 현재 singer ID와 그룹 멤버들의 ID를 차례로 비교하여 singer가 아닌 경우에만 해당 멤버의 50030번 포트로 전달받은 음성을 그대로 전달한다.

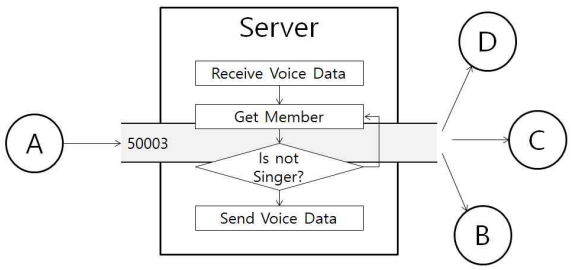

그림 1. 서버음성전달 흐름도
Fig. 1. The voice delivery flow-chart in server

3. 구현

3.1 서버 내 클라이언트 Group 생성

스마트 폰 간의 음성을 전달하기 위해 중간에서 스마트 폰 클라이언트들을 관리하고 그룹을 만들어주는 등의 일을 하는 서버를 생성하였다.

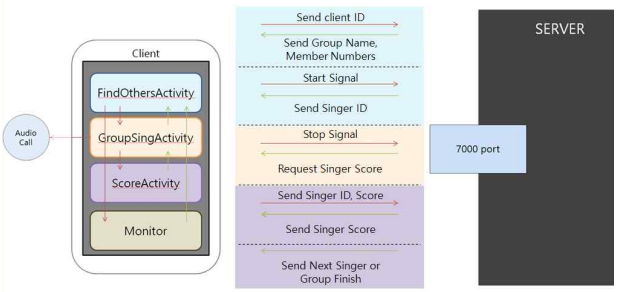

그림 2. 서버와 클라이언트 간 통신 흐름도
Fig. 2. The communication flow between server and client

그림 2는 서버와 클라이언트 간의 통신의 흐름을 전반적으로 보여준다. 클라이언트는 대결하기 모드를 실행한 후에 부를 노래 검색을 완료하면, FindOthersActivity로 넘어와 Command를 INFO_HP로 설정하고 OP에 클라이언트의 ID값을 넣어 서버의 7000번 포트로 전달하고 대기한다. 서버는 클라이언트를 Member 형태에 맞게 만든 후 MemberMap에 추가하고 기존 추가 가능한 Group이 있는지 파악한 후 없다면 Group을 하나 생성하여 클라이언트를 추가한다. 그리고 해당 Group의 Name과 Group 내 Member 수를 클라이언트에게 전달한다. 클라이언트는 계속 FindOthersActivity에서 대기하다가 서버로부터 전달 받은 Command가 SERVER_CONNLIST인 메시지를 통해 자신이 추가된 Group의 이름과 해당 Group 내 Member 수를 확인할 수

있다. 클라이언트가 노래를 시작할 때 Command를 INFO_START로 설정하여 보내면 서버는 해당 클라이언트가 속한 Group에서 임의로 singer를 설정하여 Group 내 전체 Member에게 전달하고 해당 Group의 음성 전달 Thread를 실행 시키도록 구현하였다. 클라이언트는 서버로부터 singer에 대한 정보가 전달되면 자신이 singer인지 아닌지 판단한다. 자신이 singer라면 미리 검색한 노래와 가수를 이용하여 Youtube API를 통해 노래방 영상을 검색 후 GroupSingActivity로 넘어간다. 자신이 singer가 아니라면 바로 GroupSingActivity로 넘어간다. 그 후 GroupSingActivity에서 AudioCall을 실행한다. AudioCall은 노래를 부르는 음성을 전달하거나 전달 받은 음성을 들을 수 있도록 하는 클래스이기 때문에 클라이언트는 서버로부터 음성이외의 다른 신호가 올 경우를 대비해 GroupSingActivity에서 대기한다. 노래가 끝나면 singer인 클라이언트가 Command를 INFO_STOP로 설정하여 서버로 전달하고 서버는 이 정보를 받아 Group 내 Member들에게 점수를 요청하기 위해 Command를 SERVER_RSCORE로 설정한 메시지를 보낸다. 점수 요청에 대한 메시지를 받은 클라이언트들은 ScoreActivity로 넘어가서 음성인식을 통해 사용자로부터 singer의 점수를 입력 받아 Command를 INFO_SCORE로 설정하여 서버로 전달한 후 대기한다. 서버는 Member들이 보낸 점수를 받아 이를 종합하여 다시 Group 내 Member들에게 Command를 SERVER_SCORE로 설정한 메시지를 보내 singer의 최종 점수를 전달한다. 이후 Group의 Member들이 노래를 다 불렀는지 파악하여 노래를 다 부르지 않은 경우, 해당 그룹에서 다시 singer를 선정하고 Command를 SERVER_HPSTART로 설정하여 Group 내 Member들에게 전달한 후, 음성 전달 Thread를 실행되도록 하였다. 노래를 다 부른 경우에는 Member들에게 Group의 노래가 모두 끝났다는 의미로 Command를 SERVER_FINISH로 설정하여 전달한 후 Group을 해체시키도록 하였다. 클라이언트는 ScoreActivity에서 계속 대기하고 있다가 SERVER_HPSTART 메시지가 오는 경우에는 다시 FindOthersActivity로 넘어가서 위에서 진행했던 과정을 반복하고 SERVER_FINISH 메시지가 오는 경우에는 그룹 종료에 대한 음성 안내 후 앱의 설정 탭으로 이동하게 된다.

3.2 서버를 중심으로 한 클라이언트 간의 음성 통신

스마트 폰 클라이언트가 노래 시작에 대한 신호를 보내오면 서버는 음성 전달 Thread를 실행하도록 구현하였다. 그림 3은 클라이언트들이 서버를 중심으로 음성 데이터를 전달하는 흐름을 설명한 것이다. 서버 내 음성전달 Thread에서 새로운 port를 열고 Group의 singer가 보내오는 음성 데이터를 그대로 Group 내 singer를 제외한 다른 Member들에게 전달하도록 하였다. 서버로부터 임의로 선정된 singer 클라이언트는 오디오 녹음 하드웨어 장치를 켜고 AudioRecord 클래스를 이용하여 사용자의 음성을 바이트 단위로 녹음하여 서버로 전달하도록 하였다. singer가 아닌 클라이언트는 스피커를 켠 후 AudioTrack 클래스를 이용하여 알맞은 포맷에 맞게 서버로부터 전달된 바이트 단위 음성을 출력하도록 하였다.

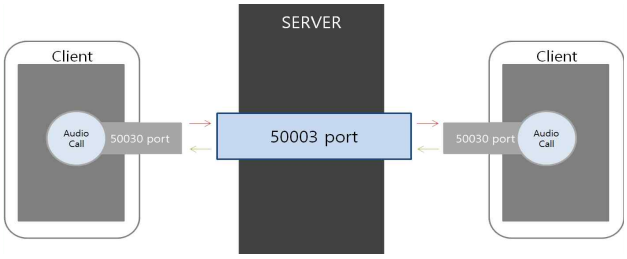

그림 3. 서버기반의 클라이언트 간 음성 전달

Fig. 3. The voice delivery flow between clients based on the server

4. 결론

운전 중에 졸음이 온다면 운전자 스스로가 졸린 상태를 인지하고 빠른 조치를 취하는 경우가 아니면 졸음을 깨는 데는 현재 정해진 방식이 없다. 따라서 본 연구에서는 스마트 폰에서 실시간 음성 통신으로 노래 부르기 앱 구현을 통해 UDP Socket 통신 방식과 안드로이드 음성 처리 방식에 대해 살펴보고, UDP 방식의 음성을 전달하는 서버와 클라이언트 모델을 구현했다. 클라이언트와 서버 간의 데이터를 처리하기 위한 프로토콜을 설계한 후, 서버 내에 Group을 생성하여 접속한 클라이언트를 관리하고, 서버가 Group 내 Member간에 음성 데이터를 전달할 수 있도록 음성 전달 Thread를 구현했다.

본 연구의 N:N 실시간 음성 서비스 기술을 다양한 콘텐츠 분야에 적용할 수 있도록 운전자 대상으로 한 앱을 제공하였다. 이는 졸음으로 인한 사고를 줄이는 효과적인 방법이 될 수 있다. 향후 더 다양한 미디어 콘텐츠 분야에 적용 가능하도록 더 많은 사용자가 참여할 수 있도록 다중 참여 그룹핑의 효율성을 높이고자 한다.

참 고 문 헌 (References)

[1] M. Kwong, S. Cherkaoui, R. Lefebvre, “Multiple description and multi-path routing for robust voice transmission over ad-hoc networks," in IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob, pp.262-267, 2006.

[2] P. Gupta and P. R. Kumar, “The capacity of wireless networks," IEEE Trans on Info Theory, Mar.2000.

[3] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee, and R.Morris, “Capacity of ad-hoc wireless networks," in MOBICOM, 2001.

[4] P. Jung and H. Sangho, “A Multi-hop Protocol Design for the Transfer of Voice Data,” Journal of KIISE : Computing Practices and Letters, Vol. 19, No. 4, pp. 205-208, 2013.